This International Student Edition is for use outside of the U.S.

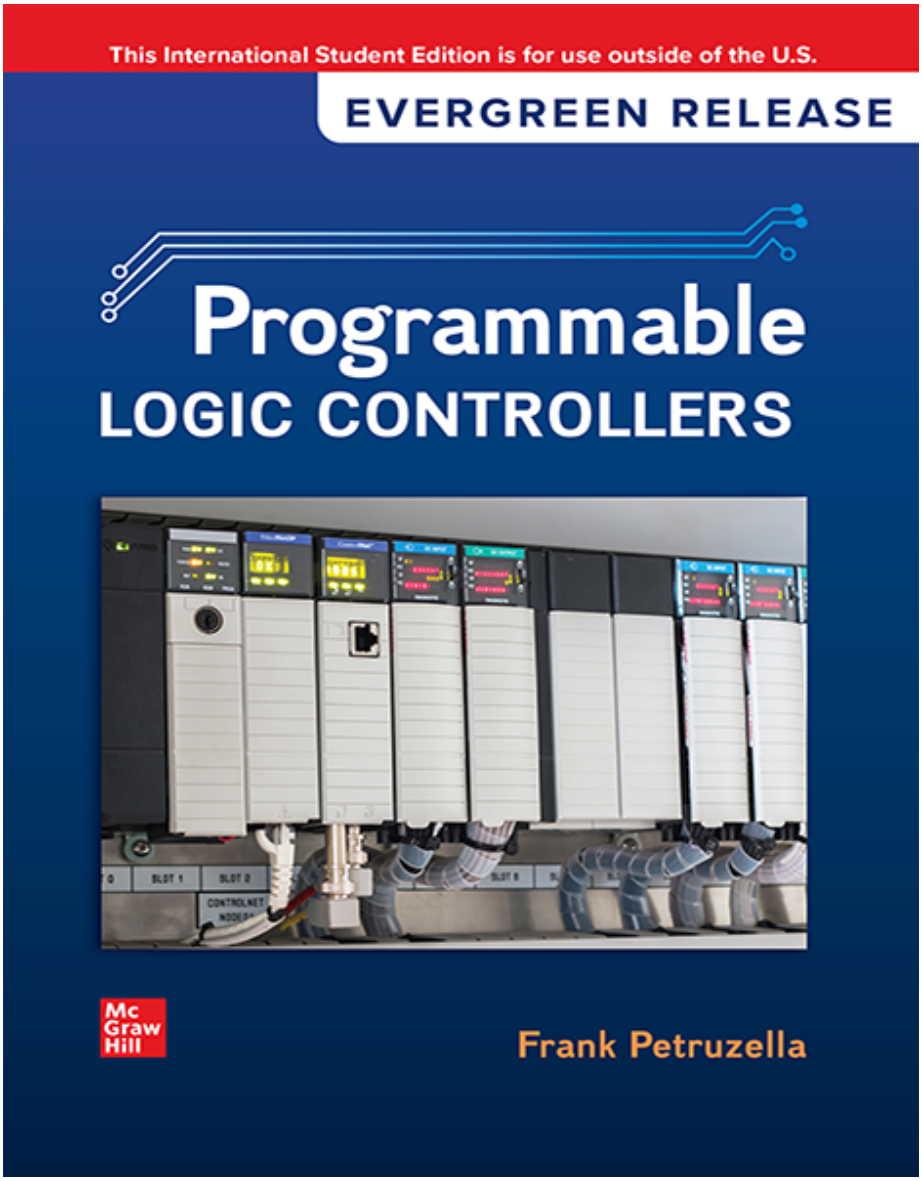
EVERGREEN RELEASE

Programmable LOGIC CONTROLLERS



Mc Graw Hill

Frank Petruzella



OceanofPDF.com

Programmable Logic Controllers

2025 Release

Frank D. Petruzella

McGraw Hill

OceanofPDF.com



PROGRAMMABLE LOGIC CONTROLLERS

Published by McGraw Hill LLC, 1325 Avenue of the Americas, New York, NY 10019. Copyright ©2025 by McGraw Hill LLC. All rights reserved. Printed in the United States of America. No part of this publication may be reproduced or distributed in any form or by any means, or stored in a database or retrieval system, without the prior written consent of McGraw Hill LLC, including, but not limited to, in any network or other electronic storage or transmission, or broadcast for distance learning.

Some ancillaries, including electronic and print components, may not be available to customers outside the United States.

This book is printed on acid-free paper.

1 2 3 4 5 6 7 8 9 LWI 30 29 28 27 26 25

ISBN 978-1-265-20789-2

MHID 1-265-20789-5

Cover Image: *Xmentoys/Shutterstock*

All credits appearing on page or at the end of the book are considered to be an extension of the copyright page.

The Internet addresses listed in the text were accurate at the time of publication. The inclusion of a website does not indicate an endorsement by the authors or McGraw Hill LLC, and McGraw Hill LLC does not guarantee the accuracy of the information presented at these sites.

mheducation.com/highered

OceanofPDF.com

About the Author

Frank D. Petruzella has extensive practical experience in the electrical control field, as well as many years of experience teaching and authoring textbooks. Before becoming a full-time educator, he was employed as an apprentice and electrician in areas of electrical installation and maintenance. He holds a Master of Science degree from Niagara University, a Bachelor of Science degree from the State University of New York College–Buffalo, as well as diplomas in Electrical Power and Electronics from the Erie County Technical Institute.

OceanofPDF.com

Contents

About the Author Page iii

Preface Page vii

Acknowledgments Page ix

Chapter 1 Programmable Logic Controllers (PLCs): An Overview Page 1

1.1 Programmable Logic Controllers Page 2
1.2 Parts of a PLC Page 4
1.3 Principles of Operation Page 6
1.4 Modifying the Operation Page 9
1.5 PLCs versus Computers Page 10
1.6 PLC Size and Application Page 11

Review Questions Page 12
Problems Page 13

Chapter 2 PLC Hardware Components Page 14

2.1 The I/O Section Page 15
2.2 Discrete I/O Modules Page 19
2.3 Analog I/O Modules Page 23
2.4 Special I/O Modules Page 27
2.5 I/O Specifications Page 27
Typical Discrete (Digital) I/O Module Specifications Page 27
Typical Analog I/O Module Specifications Page 29
2.6 The Central Processing Unit (CPU) Page 29
2.7 Memory Design Page 31
2.8 Memory Types Page 32
2.9 Programming Terminal Devices Page 33
2.10 Recording and Retrieving Data Page 33
2.11 Human Machine Interfaces (HMIs) Page 34

Review Questions Page 36
Problems Page 38

Chapter 3 Number Systems and Codes Page 39

3.1 Decimal System **Page 40**
3.2 Binary System **Page 40**
3.3 Negative Numbers **Page 42**
3.4 Octal System **Page 43**
3.5 Hexadecimal System **Page 43**
3.6 Binary Coded Decimal (BCD) System **Page 43**
3.7 Gray Code **Page 45**
3.8 ASCII Code **Page 46**
3.9 Parity Bit **Page 47**
3.10 Binary Arithmetic **Page 47**
3.11 Floating Point Arithmetic **Page 49**

Review Questions **Page 51**
Problems **Page 52**

Chapter 4 Fundamentals of Logic Page 53

4.1 The Binary Concept **Page 54**
4.2 AND, OR, and NOT Functions **Page 54**
The AND Function ***Page 54***
The OR Function ***Page 55***
The NOT Function ***Page 55***
The Exclusive-OR (XOR) Function ***Page 57***
4.3 Boolean Algebra **Page 57**
4.4 Developing Logic Gate Circuits from Boolean Expressions **Page 59**
4.5 Producing the Boolean Equation for a Given Logic Gate Circuit **Page 59**
4.6 Hardwired Logic versus Programmed Logic **Page 60**
4.7 Programming Word Level Logic Instructions **Page 63**

Review Questions **Page 65**
Problems **Page 65**

Chapter 5 Basics of PLC Programming Page 67

5.1 Processor Memory Organization **Page 68**
Program Files ***Page 68***
Data Files ***Page 68***
5.2 Program Scan **Page 71**
5.3 PLC Programming Languages **Page 73**

5.4 Bit-Level Logic Instructions **Page 76**
5.5 Instruction Addressing **Page 79**
5.6 Branch Instructions **Page 79**
5.7 Internal Relay Instructions **Page 81**
5.8 Programming Examine If Closed and Examine If Open Instructions **Page 82**
5.9 Entering the Ladder Diagram **Page 83**
5.10 Modes of Operation **Page 85**
5.11 Connecting with Analog Devices **Page 85**

Review Questions **Page 87**
Problems **Page 88**

Chapter 6 Developing Fundamental PLC Wiring Diagrams and Ladder Logic Programs **Page 90**

6.1 Electromagnetic Control Relays **Page 91**
6.2 Contactors **Page 92**
6.3 Motor Starters **Page 93**
6.4 Manually Operated Switches **Page 94**
6.5 Mechanically Operated Switches **Page 95**
6.6 Sensors **Page 96**
Proximity Sensor ***Page 96***
Magnetic Reed Switch ***Page 99***
Light Sensors ***Page 99***
Ultrasonic Sensors ***Page 101***
Strain/Weight Sensors ***Page 102***
Temperature Sensors ***Page 102***
Flow Measurement ***Page 102***
Velocity and Position Sensors ***Page 103***
6.7 Output Control Devices **Page 103**
6.8 Seal-In Circuits **Page 106**
6.9 Electrical Interlocking Circuits **Page 106**
6.10 Latching Relays **Page 108**
6.11 Converting Relay Schematics into PLC Ladder Programs **Page 111**
6.12 Writing a Ladder Logic Program Directly from a Narrative Description **Page 114**
6.13 Instrumentation **Page 117**

Review Questions **Page 118**
Problems **Page 119**

Chapter 7 Programming Timers Page 121

7.1 Mechanical Timing Relays **Page 122**
7.2 Timer Instructions **Page 123**
7.3 On-Delay Timer Instruction **Page 124**
7.4 Off-Delay Timer Instruction **Page 130**
7.5 Retentive Timer **Page 133**
7.6 Cascading Timers **Page 136**

Review Questions **Page 140**
Problems **Page 140**

Chapter 8 Programming Counters Page 145

8.1 Counter Instructions **Page 146**
8.2 Up-Counter **Page 147**
One-Shot Instruction ***Page 151***
8.3 Down-Counter **Page 155**
8.4 Cascading Counters **Page 158**
8.5 Incremental Encoder-Counter Applications **Page 161**
8.6 Combining Counter and Timer Functions **Page 162**
8.7 High-Speed Counters **Page 165**

Review Questions **Page 167**
Problems **Page 167**

Chapter 9 Program Control Instructions Page 172

9.1 Program Control **Page 173**
9.2 Master Control Reset Instruction **Page 173**
9.3 Jump and Label Instructions **Page 176**
9.4 Subroutine Functions **Page 178**
9.5 Immediate Input and Immediate Output Instructions **Page 181**
9.6 Forcing External I/O Addresses **Page 182**
9.7 Safety Circuitry **Page 185**
9.8 Fault Routine **Page 188**
9.9 Temporary End Instruction **Page 188**

Review Questions **Page 189**
Problems **Page 189**

Chapter 10 Data Manipulation Instructions Page 193

10.1 Data Manipulation Page 194
10.2 Data Transfer Operations Page 194
10.3 Data Compare Instructions Page 202
10.4 Data Manipulation Programs Page 206
10.5 Numerical Data I/O Interfaces Page 209
10.6 Closed-Loop Control Page 212

Review Questions Page 215
Problems Page 216

Chapter 11 Math Instructions Page 219

11.1 Math Instructions Page 220
11.2 Addition Instruction Page 221
11.3 Subtraction Instruction Page 223
11.4 Multiplication Instruction Page 224
11.5 Division Instruction Page 225
11.6 Other Word-Level Math Instructions Page 227
11.7 File Arithmetic Operations Page 230

Review Questions Page 232
Problems Page 233

Chapter 12 Sequencer and Shift Register Instructions Page 237

12.1 Mechanical Sequencers Page 238
12.2 Sequencer Instructions Page 240
12.3 Sequencer Programs Page 244
12.4 Bit Shift Registers Page 249
12.5 Word Shift Operations Page 257

Review Questions Page 262
Problems Page 262

Chapter 13 PLC Installation Practices, Editing, and Troubleshooting Page 266

13.1 PLC Enclosures Page 267
13.2 Electrical Noise Page 269
13.3 Leaky Inputs and Outputs Page 270
13.4 Grounding Page 270
13.5 Voltage Variations and Surges Page 272

13.6 Program Editing and Commissioning **Page 273**
13.7 Programming and Monitoring **Page 274**
13.8 Preventive Maintenance **Page 276**
13.9 Troubleshooting **Page 276**
Processor Module ***Page 277***
Input Malfunctions ***Page 277***
Output Malfunctions ***Page 278***
Ladder Logic Program ***Page 280***
13.10 PLC Programming Software **Page 284**

Review Questions **Page 287**
Problems **Page 287**

Chapter 14 Process Control, Network Systems, and SCADA Page 290

14.1 Types of Processes **Page 291**
14.2 Structure of Control Systems **Page 293**
14.3 On/Off Control **Page 295**
14.4 PID Control **Page 296**
14.5 Motion Control **Page 300**
14.6 Data Communications **Page 301**
Data Highway ***Page 307***
Serial Communication ***Page 307***
DeviceNet ***Page 307***
ControlNet ***Page 308***
EtherNet/IP ***Page 309***
Modbus ***Page 311***
Fieldbus ***Page 311***
PROFIBUS-DP ***Page 311***
14.7 Supervisory Control and Data Acquisition (SCADA) **Page 313**
14.8 Variable Frequency Drive (VFD) with PLC Control **Page 314**
Drive Control ***Page 315***
Drive Status ***Page 316***
14.9 PLC and HMI Programming **Page 318**

Review Questions **Page 321**
Problems **Page 322**

Chapter 15 ControlLogix Controllers Page 323

Part 1 Memory and Project Organization **Page 324**

Memory Layout Page 324
Configuration Page 324
Project Page 325
Tasks Page 326
Programs Page 326
Routines Page 327
Tags Page 327
Structures Page 330
Creating Tags Page 331
Monitoring and Editing Tags Page 332
Array Page 332
Review Questions Page 334
Part 2 Bit-Level Programming Page 335
Program Scan Page 335
Creating Ladder Logic Page 336
Tag-Based Addressing Page 337
Adding Ladder Logic to the Main Routine Page 338
Internal Relay Instructions Page 341
Latch and Unlatch Instructions Page 343
One-Shot Instruction Page 345
Review Questions Page 348
Problems Page 348
Part 3 Programming Timers Page 350
Timer Predefined Structure Page 350
On-Delay Timer (TON) Page 351
Off-Delay Timer (TOF) Page 354
Retentive Timer On (RTO) Page 356
Cascading of Timers Page 357
Review Questions Page 359
Problems Page 359
Part 4 Programming Counters Page 360
Counters Page 360
Count-Up (CTU) Counter Page 361
Count-Down (CTD) Counter Page 363
Combining Counter and Timer Functions Page 364
Review Questions Page 366
Problems Page 366
Part 5 Math, Comparison, and Move Instructions Page 367

Math Instructions Page 367
Comparison Instructions Page 369
Move Instructions Page 372
Combining Math, Comparison, and Move Instructions Page 373
Review Questions Page 380
Problems Page 380
Part 6 Function Block Programming Page 381
Function Block Diagram (FBD) Page 381
FBD Programming Page 385
Review Questions Page 391
Problems Page 391
Glossary Page 392
Index Page 404



OceanofPDF.com

Preface

Programmable logic controllers (PLCs) continue to evolve as new technologies are added to their capabilities. As PLC technology has advanced, so have programming languages and communications capabilities. Today's PLCs offer faster scan times, space efficient high-density input/output systems, and special interfaces to allow non-traditional devices to be attached directly to the PLC.

The primary source of information for a particular PLC is always the accompanying user manuals provided by the manufacturer. This textbook is not intended to replace the vendor's reference material, but rather to complement, clarify, and expand on this information. The text covers the basics of programmable logic controllers in a manner that complements instruction with an RSLogix 500 or RSLogix 5000 platform. The underlying PLC principles and concepts covered in the text are common to most manufacturers. They serve to maximize the knowledge gained through on-the-job training and programs offered by different vendors.

The text is written in an easy-to-read style that is designed for students with no prior PLC experience. For example, when the operation of a program is called for, a bulleted list is used to summarize its execution. The bulleted list replaces a lengthy paragraph and is especially helpful when covering the different steps related to the execution of a program.

Each chapter begins with a brief introduction outlining chapter coverage and learning objectives. When applicable, the relay equivalent of the virtual programmed instruction is explained first, followed by the appropriate PLC instruction. Chapters conclude with a set of review questions and problems. The review questions are closely related to the chapter objectives and require students to recall and apply information covered in the chapter. The problems range from easy to difficult, thus challenging students at various levels of competence.

OceanofPDF.com

Chapter Updates to Note

Chapter 2

- New RSLogix 5000 program used to execute a three-wire motor control circuit.

Chapter 4

- New RSLogix 5000 AND logic rung.
- New RSLogix 5000 OR logic rung.
- New RSLogix 5000 NOT logic rung.
- New RSLogix 5000 NAND logic rung.
- New RSLogix 5000 XOR logic rung.
- New RSLogix 5000 word-level Bitwise AND logic rung.

Chapter 6

- New RSLogix 5000 push-to-start/push-to-stop program.

Chapter 7

- New RSLogix 5000 automatic sequential control system replacement program.
- New RSLogix 5000 off-delay timer relay circuit replacement program.
- New RSLogix 5000 retentive on-delay timer alarm replacement program.
- New RSLogix 5000 annunciator replacement program.

Chapter 8

- New RSLogix 5000 counter used for length measurement replacement program.

Chapter 14

The new section 14.5 covers pairing of an HMI terminal with its related PLC program. Includes the procedure involved setting up communication and data transfer connections between the two through a network. Contains a summary of the basics associated with the creating and navigating a simple motor control application using Studio 5000 and Factory Talk View software.

Chapter 15 Part 2

- New RSLogix 5000 XIC and XIO instructions logic evaluation exercise.
- New RSLogix 5000 dual compressor application program.

Chapter 15 Part 4

- New RSLogix 5000 timer driving a counterprogram.

Chapter 15 Part 5

- New RSLogix 5000 used to display the lapsed time period of a timer program.
- New RSLogix 5000 cooling tower system program.
- New RSLogix 5000 Move instruction exercise.
- New RSLogix 5000 Masked Move instruction exercise.
- New RSLogix 5000 Bit Field Distribute instruction exercise.

Minimal updates were made to the remaining chapters.

OceanofPDF.com

Acknowledgments

I would like to thank the following reviewers for their comments and suggestions:

Iftekhar Basith
Sam Houston State University

Joseph Cuiffi
Pennsylvania State University-New Kensington

Narendra Datta
University of West Alabama

Carroll Garrett
Alamo Community College

Mary Sue Gurka
Joliet Junior College

Sean Hickman
Lansing Community College

Patrick Kiely
Volunteer State Community College

George Lister
Del Mar College

Joe Long
University of Central Missouri

Brian Sparks
Southcentral Kentucky Community and Technical College

Steve Swailes
Grand Rapids Community College

Wade Wittmus
Lakeshore Technical College

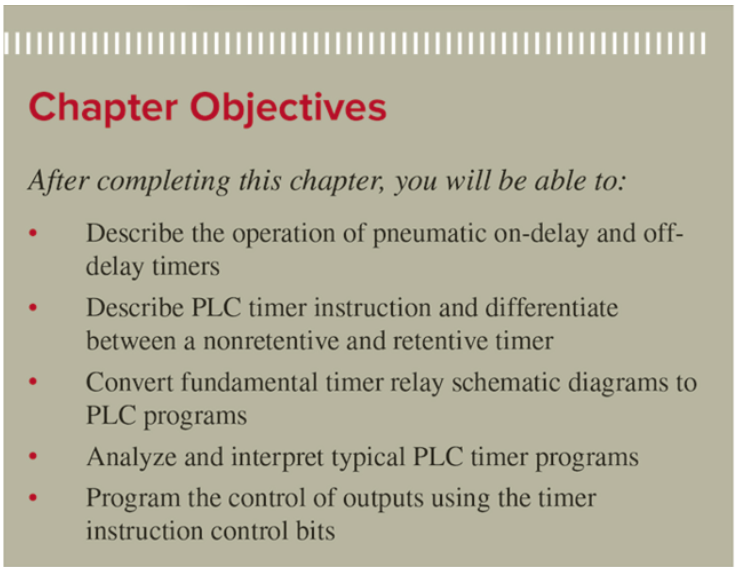
Wai Yung
Johnson and Wales University

Frank D. Petruzella

OceanofPDF.com

Programmable Logic Controllers **makes it easy to learn PLCs from the ground up! Up-to-the-minute revisions include all the newest developments in programming, installing, and maintaining processes. Clearly developed chapters deliver the organizing objectives, explanatory content with helpful diagrams and illustrations, and closing review problems that evaluate retention of the chapter objectives.**

CHAPTER OBJECTIVES overview the chapter, letting students and instructors focus on the main points to better grasp concepts and retain information.



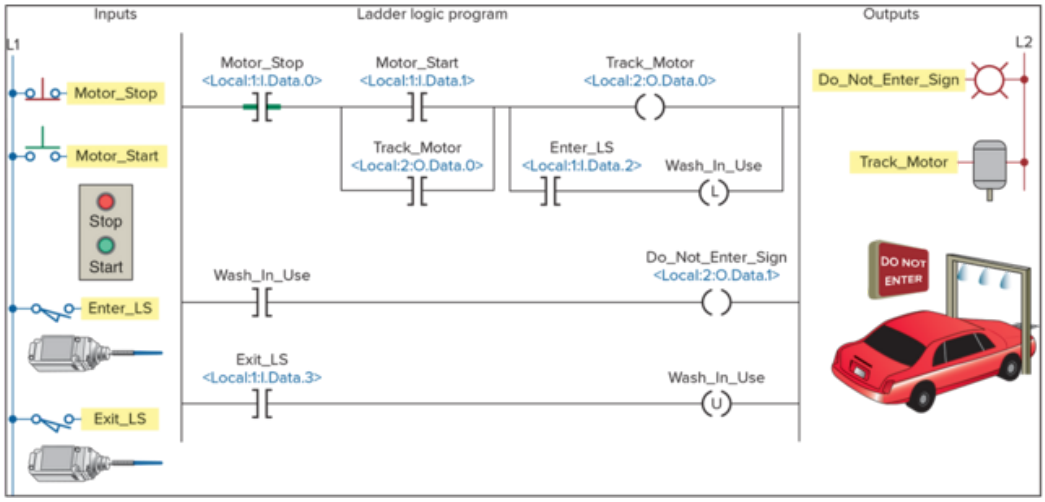
Chapter Objectives

After completing this chapter, you will be able to:

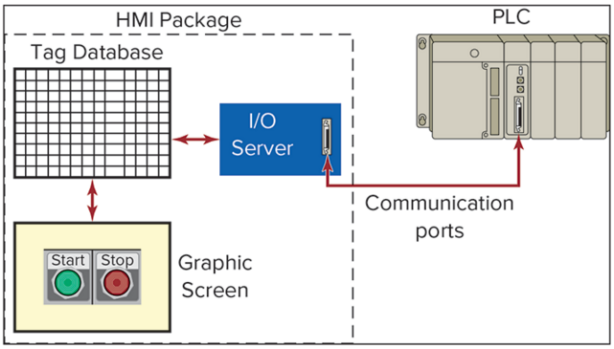
- Describe the operation of pneumatic on-delay and off-delay timers
- Describe PLC timer instruction and differentiate between a nonretentive and retentive timer
- Convert fundamental timer relay schematic diagrams to PLC programs
- Analyze and interpret typical PLC timer programs
- Program the control of outputs using the timer instruction control bits

Chapter content includes rich illustrative detail and extensive visual aids, allowing students to grasp concepts more quickly and understand practical applications

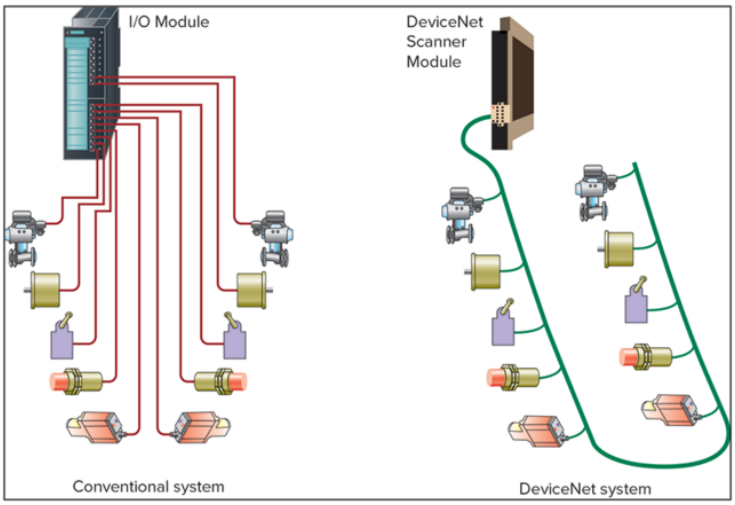
Here, drawings of realworld input and output devices have been included



In chapter 2, students not only read about but can also see how HMIs fit into an overall PLC system, giving them a practical introduction to the topics



Coverage of communications and control networks utilizes clear graphics to demonstrate how things work



BULLETED LISTS break down processes to helpfully summarize execution of tasks

Creating Ladder Logic

Although other programming languages are available, ladder logic is the most common programming language for PLCs. The instructions in ladder logic programming can be divided into two broad categories: input and output instructions. The most common input instruction is equivalent to a relay contact and the most common output instruction is the equivalent of a relay coil (Figure 15-27). When creating ladder I/O bit instructions, the following rules apply:

- All input instructions must be to the left of an output instruction.
- A rung cannot begin with an output instruction if it also contains an input instruction. This is because the controller tests all inputs for true or false before deciding what value the output instruction should be.
- A rung does not need to contain any input instructions, but it must contain at least one output instruction.

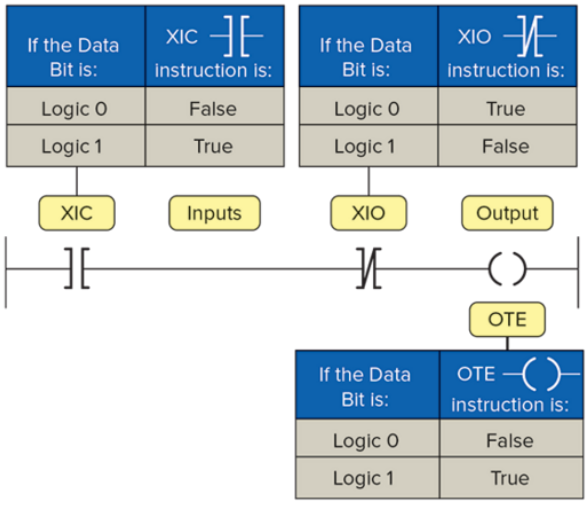


Figure 15-27 Contacts and coil instructions.

Diagrams, such as this one illustrating an overview of the function block programming language, help students put the pieces together

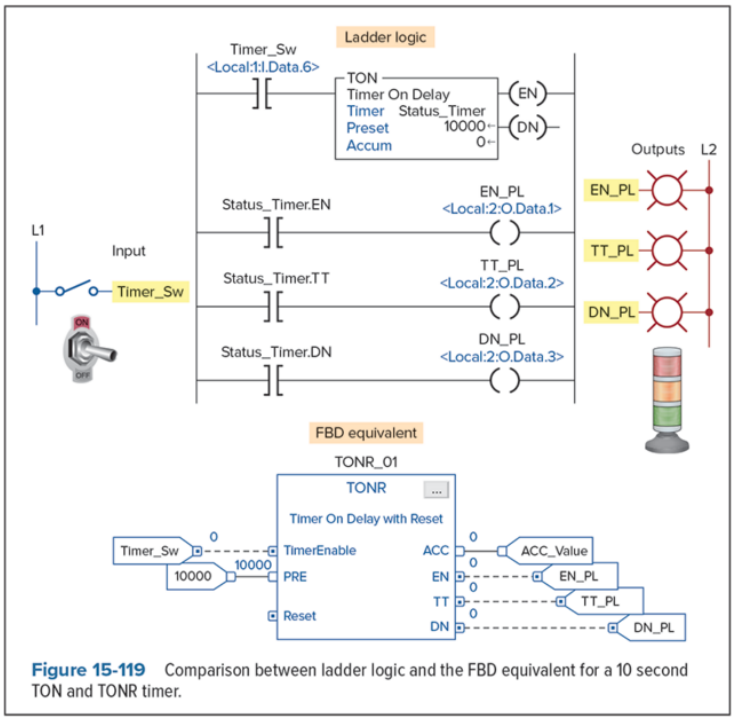


Figure 15-119 Comparison between ladder logic and the FBD equivalent for a 10 second TON and TONR timer.

END-OF-CHAPTER REVIEWS are structured to reinforce chapter objectives

CHAPTER 6 REVIEW QUESTIONS

1. Explain the basic operating principle of an electromagnetic control relay.
2. What is the operating difference between a normally open and a normally closed relay contact?
3. In what ways are control relay coils and contacts rated?
4. How do contactors differ from relays?
5. What is the main difference between a contactor and a magnetic motor starter?
6. **a.** Draw the schematic for an across-the-line AC magnetic motor starter.
   b. With reference to this schematic, explain the function of each of the following parts:
   i. Main contact M
   ii. Control contact M
   iii. Starter coil M
   iv. OL relay coils
   v. OL relay contact
7. The current requirement for the control circuit of a magnetic starter is normally much smaller than that required by the power circuit. Why?
8. Compare the method of operation of each of the following types of switches:
   a. Manually operated switch
   b. Mechanically operated switch
   c. Proximity switch
15. Compare the operation of the reflective-type and through-beam photoelectric sensors.
16. Give an explanation of how a scanner and a decoder act in conjunction with each other to read a bar code.
17. How does an ultrasonic sensor operate?
18. Explain the principle of operation of a strain gauge.
19. Explain the principle of operation of a thermocouple.
20. What is the most common approach taken with regard to the measurement of fluid flow?
21. Explain how a tachometer is used to measure rotational speed.
22. How does an optical encoder work?
23. Draw an electrical symbol used to represent each of the following PLC control devices:
   a. Pilot light
   b. Relay
   c. Motor starter coil
   d. OL relay contact
   e. Alarm
   f. Heater
   g. Solenoid
   h. Solenoid valve
   i. Motor
   j. Horn
24. Explain the function of each of the following actuators:
   a. Solenoid
   b. Solenoid valve

EXAMPLE PROBLEMS help bring home the applicability of chapter concepts

CHAPTER 6 PROBLEMS

1. Design and draw the schematic for a conventional hardwired relay circuit that will perform each of the following circuit functions when a normally closed pushbutton is pressed:
   - Switch a pilot light on
   - De-energize a solenoid
   - Start a motor running
   - Sound a horn
2. Design and draw the schematic for a conventional hardwired circuit that will perform the following circuit functions using two break-make pushbuttons:
   - Turn on light L1 when pushbutton PB1 is pressed.
   - Turn on light L2 when pushbutton PB2 is pressed.
   - Electrically interlock the pushbuttons so that L1 and L2 cannot both be turned on at the same time.
3. Study the ladder logic program in Figure 6-74, and answer the questions that follow:
   a. Under what condition will the latch rung 1 be true?
   b. Under what conditions will the unlatch rung 2 be true?
   c. Under what condition will rung 3 be true?
   d. When PL1 is on, the relay is in what state (latched or unlatched)?
   e. When PL2 is on, the relay is in what state (latched or unlatched)?
   f. If AC power is removed and then restored to the circuit, what pilot light will automatically come on when the power is restored?
   g. Assume the relay is in its latched state and all three inputs are false. What input change(s) must occur

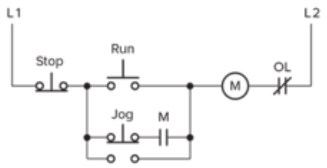
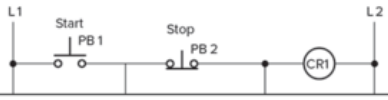


Figure 6-75 Hardwired control circuit for Problem 4.

will correctly execute the hardwired control circuit in Figure 6-75.
Assume: Stop pushbutton used is an NO type.
Run pushbutton used is an NO type.
Jog pushbutton used has one set of NO contacts.
OL contact is hardwired.

5. Design a PLC program and prepare a typical I/O connection diagram and ladder logic program that will correctly execute the hardwired control circuit in Figure 6-76.
   Assume: PB1 pushbutton used is an NO type.
   PB2 pushbutton used is an NC type.
   PS1 pressure switch used is an NO type.
   LS1 limit switch used has only one set of NC contacts.



OceanofPDF.com

Ancillaries that Work

Expanded on and updated from the previous edition, this release includes an outstanding instructor support package available through Connect:

- Question banks for each chapter.
- Lab Manual question banks for several chapters.
- Lecture PowerPoints.
- Answers to the questions and problems in the textbook, LogixPro Manual, and RSLogix 5000 Manual.

PLC SIMULATOR LAB MANUALS

Two lab manuals are available to support Programmable Logic Controllers (RsLogix 500 Lab Manual and RSLogix 5000 Lab Manual). These lab manuals are offered as eBooks only (within Connect or standalone). Further details on how best to make use of these manuals are outlined below:

As an Instructor Resource

- **Dual-purpose** programming assignments designed to be used as **standalone** or in **conjunction** with a wide variety of classroom PLC controller training equipment.
- **All assignments** refer to the execution of programs covered in the textbox.
- **Student progress** is evaluated based on how well their simulated program meets the stated requirements.
- All fully **solved and operational** simulator files can be used by instructors to walk students through the operating stages of a program.
- **Fully integrated** with the **CONNECT** e-learning platform featuring an extensive bank of objective test questions for each Chapter.

As a Student Resource

- **Validating** the execution of a program design prior to downloading it to the PLC controller.
- **Experimenting** with different PLC control scenarios.
- Provides a useful application **alternative** when PLC trainer hardware is not available.
- Helps students, via **CONNECT**, with managing and submitting assignments online and accessing course-specific resources.

RSLogix 500 Lab Manual and Simulator Program

- This simulator emulates programs based on RSLogix 500 software.
- It must be installed on your computer to run or produce all programs from the related Lab Manual.

- **It is not included as part of the Lab Manual and can be purchased from the LogixPro 500 website: https://canadu.com/pp/buyakey.asp**

RSLogix 5000 Lab Manual and Simulator Program

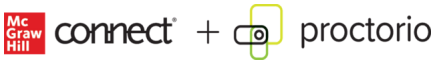
- This simulator emulates programs based on RSLogix 5000 software.
- It must be installed on your computer to run or produce all programs from the related Lab Manual.
- **It is not included as part of the Lab Manual and can be purchased from the PLCLogix 5000 website: https://www.plclogix.com/**

OceanofPDF.com

Digital Resources

Proctorio

Remote Proctoring & Browser-Locking Capabilities



Remote proctoring and browser-locking capabilities, hosted by Proctorio within Connect, provide control of the assessment environment by enabling security options and verifying the identity of the student.

Seamlessly integrated within Connect, these services allow instructors to control the assessment experience by verifying identification, restricting browser activity, and monitoring student actions.

Instant and detailed reporting gives instructors an at-a-glance view of potential academic integrity concerns, thereby avoiding personal bias and supporting evidence-based claims.

ReadAnywhere® App

Read or study when it's convenient with McGraw Hill's free ReadAnywhere app. Available for iOS and Android smartphones or tablets, the ReadAnywhere app gives users access to McGraw Hill tools including the eBook and SmartBook® or Adaptive Learning Assignments in McGraw Hill Connect®. Students can take notes, highlight, and complete assignments offline—all their work will sync when connected to Wi-Fi. Students log in with their Connect username and password to start learning—anytime, anywhere!

Online Learning Consortium-Aligned Courses

Implementing High-Quality Instruction and Assessment through Preconfigured Courseware

In consultation with the Online Learning Consortium (OLC) and our certified Faculty Consultants, McGraw Hill has created preconfigured courseware using OLC's quality scorecard to align with best practices in online course delivery. This turnkey courseware contains a combination of formative and summative assessments, homework, and application activities, and can easily be customized to meet an individual instructor's needs and desired course outcomes. For more information, visit **mheducation.com/highered/olc**.

Test Builder in McGraw Hill Connect®

Available within Connect, Test Builder is a cloud-based tool that enables instructors to format tests to be printed, administered within a Learning Management System, or exported as a Word document. Test Builder offers a modern, streamlined interface for easy content configuration that matches course needs, without requiring a download.

Test Builder allows you to:

- access all test bank content from a particular title.
- easily pinpoint the most relevant content through robust filtering options.
- manipulate the order of questions or scramble questions and/or answers.
- pin questions to a specific location within an exam.
- determine your preferred treatment of algorithmic questions.
- choose the layout and spacing.
- add instructions and configure default settings.

Test Builder provides a secure interface for better protection of content and allows for just-in-time updates to flow directly into assessments.

Application-Based Activities in McGraw Hill Connect®

Prepare students for the real world with Application-Based Activities in Connect. These highly interactive, assignable exercises boost engagement and provide a safe space to applyconcepts learned to real-world, course-specific problems. Each Application-Based Activity involves the application of multiple concepts, providing the ability to synthesize information and use critical thinking skills to solve realistic scenarios.

Polling

Every learner has unique needs. Uncover where and when you're needed with the new Polling tool in McGraw Hill Connect®! Polling allows you to discover where students are in real time. Engage students and help them create connections with your course content while gaining valuable insight during lectures. Leverage polling data to deliver personalized instruction when and where it is needed most.



McGraw Hill GO

McGraw Hill GO is an easy-to-use, quick-to-set-up eBook+ that lives within the Learning Management System (LMS). GO equips instructors and students with essential course materials: trusted, assignable content and auto-graded chapter questions that sync directly into the LMS. GO makes it easier to keep up with student progress and direct attention where it matters.

Evergreen

Content and technology are ever-changing, and it is important that you can keep your course up to date with the latest content, assessments, and accessibility. That's why we want to deliver the most current and relevant content for your course, hassle-free.

Petruzella, Programmable Logic Controllers uses an Evergreen delivery model, which is a product delivery model and experience that provides customers with the most relevant and up-to-date content, tools, and accessibility. Engage students and freshen up assignments with up-to-date coverage of select topics and new questions, all without having to switch editions or build a new course.

Create

Your Book, Your Way
McGraw Hill's Content Collections Powered by Create® is a self-service website that enables instructors to create custom course materials—print and eBooks—by drawing upon McGraw Hill's comprehensive, cross-disciplinary content. Choose what you want from our high-quality textbooks, digital products, articles, cases, and more. Combine it with your own content quickly and easily, and tap into other rights-secured, third-party content such as cases, articles, readings, cartoons, and labs. Content can be arranged in a way that makes the most sense for your course, and you can select your own cover and include the course name and school information as well. Choose the best format for your course: color print, black-and-white print, or eBook. The eBook can be included in your Connect course and is available on the free ReadAnywhere® app for smartphone or tablet access as well. When you are finished customizing, you will receive a free digital copy to review in just minutes! Visit McGraw Hill Create®—mcgrawhillcreate.com—today and begin building!

Education for All

McGraw Hill is fully committed to Education for All, which we support by clearing paths to achieve equity for every individual and by working together toward an education of value. We believe in unlocking the potential of every learner at every stage of life. To accomplish that, we are dedicated to creating products that reflect, and are accessible to, all the diverse, global customers we serve. Within McGraw Hill, we foster a culture of belonging, and we work with partners who share our commitment to equity, inclusion, and diversity in all forms. In McGraw Hill Higher Education, this includes, but is not limited to, the following:

- Refreshing and implementing inclusive content guidelines around topics including generalizations and stereotypes, gender, abilities/disabilities, race/ethnicity, sexual orientation, diversity of names, and age.
- Enhancing best practices in assessment creation to eliminate cultural, cognitive, and affective bias.
- Maintaining and continually updating a robust photo library of diverse images that reflect our student populations.
- Including more diverse voices in the development and review of our content.
- Using appropriate color contrast, non-text color, and use of color with text elements and art to improve accessibility by ensuring text and art are distinguishable and perceivable by users with

limited color vision and moderately low vision.

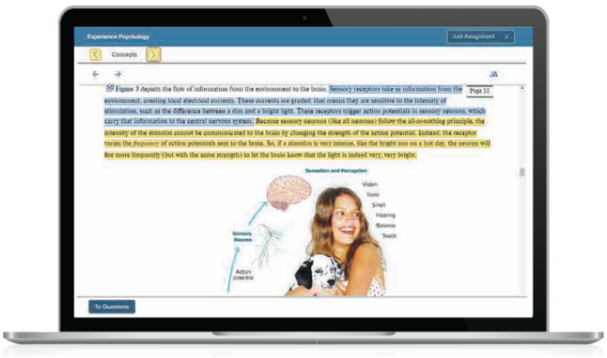
OceanofPDF.com

Mc Graw Hill connect®

Your complete course platform

Connect enables you to build deeper connections with your students through cohesive digital content and tools, creating engaging learning experiences. We are committed to providing you with the right resources and tools to support all your students along their personal learning journeys.

65%
Less Time Grading



Laptop: Getty Images; Woman/dog: George Doyle/Getty Images

Every learner is unique

In Connect, instructors can assign an adaptive reading experience with SmartBook®. Rooted in advanced learning science principles, SmartBook delivers a personalized experience, focusing students on their learning gaps and ensuring the time they spend studying is time well spent. **mheducation.com/highered/connect/smartbook**

Study anytime, anywhere

Encourage your students to download the free ReadAnywhere® app so they can access their online eBook, SmartBook®, or Adaptive Learning Assignments when it's convenient, even when they're offline. Because the app automatically syncs with students' Connect accounts, all their work is available every time they open it. **mheducation.com/readanywhere**

"I really liked this app—it made it easy to study when you don't have your textbook in front of you."

Jordan Cunningham, a student at *Eastern Washington University*

Effective tools for efficient studying

Connect is designed to help students be more productive with simple, flexible, intuitive tools that maximize study time and meet students' individual learning needs. Get learning that works for everyone with Connect.

Education for all

McGraw Hill works directly with Accessibility Services departments and faculty to meet the learning needs of all students. Please contact your Accessibility Services Office, and ask them to email **accessibility@mheducation.com**, or visit **mheducation.com/about/accessibility** for more information.

Affordable solutions, added value

Make technology work for you with LMS integration for single sign-on access, mobile access to the digital textbook, and reports to quickly show you how each of your students is doing. And with our Inclusive Access program, you can provide all these tools at the lowest available market price to your students. Ask your McGraw Hill representative for more information.

Solutions for your challenges

A product isn't a solution. Real solutions are affordable, reliable, and come with training and ongoing support when you need it and how you want it. Visit **supportateverystep.com** for videos and resources both you and your students can use throughout the term.

Updated and relevant content

Our Evergreen delivery model provides you with the most relevant and up-to-date content, tools, and accessibility. Engage students and freshen up assignments with up-to-date coverage of select topics and new questions, all without having to switch editions or build a new course.



OceanofPDF.com

1

Programmable Logic Controllers (PLCs): An Overview



Image Courtesy of Rockwell Automation, Inc.

Chapter Objectives

After completing this chapter, you will be able to:

- Define what a programmable logic controller (PLC) is and list its advantages over relay systems
- Identify the main parts of a PLC and describe their functions
- Outline the basic sequence of operation for a PLC
- Identify the general classifications of PLCs

This chapter gives a brief history of the evolution of the programmable logic controller, or PLC. The reasons for changing from relay control systems to PLCs are discussed. You will learn the basic parts of a PLC, how a PLC is used to control a process, and the different kinds of PLCs and their applications. The ladder logic language, which was developed to simplify the task of programming PLCs, is introduced.

OceanofPDF.com

1.1 Programmable Logic Controllers

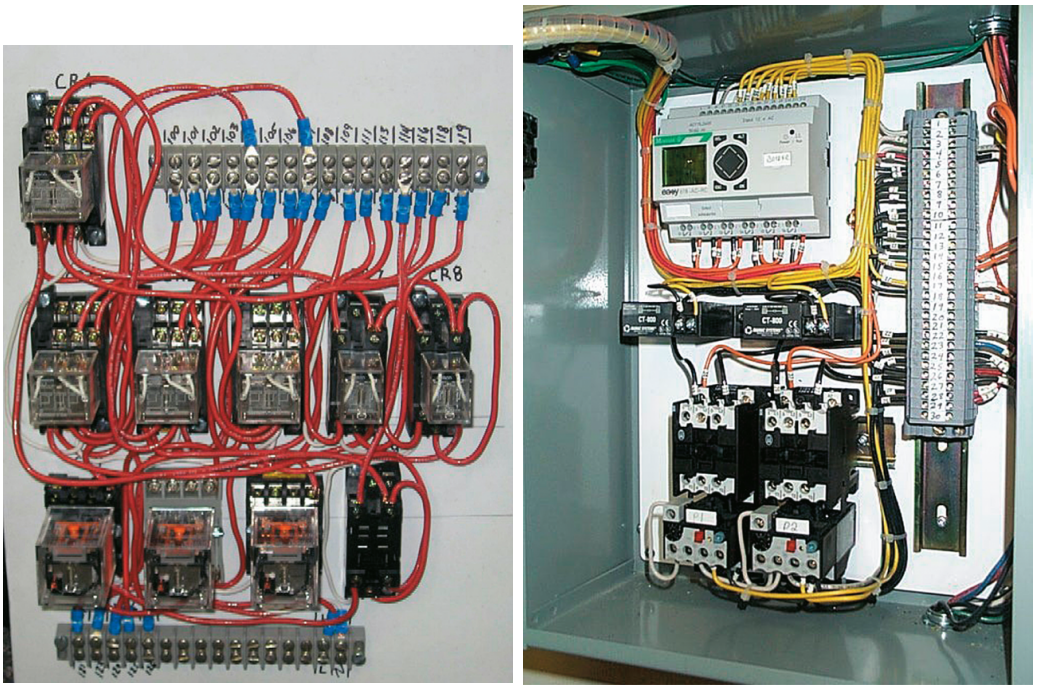
Programmable logic controllers are now the most widely used industrial process control technology. A **programmable logic controller (PLC)** is an industrial grade computer that is capable of being programmed to perform control functions. The programmable controller has eliminated much of the hardwiring associated with conventional relay control circuits. Other benefits include fast response, easy programming and installation, high control speed, network compatibility, troubleshooting and testing convenience, and high reliability.

The PLC is designed for multiple input and output arrangements, extended temperature ranges, immunity to electrical noise, and resistance to vibration and impact. Programs for the control and operation of manufacturing process equipment and machinery are typically stored in battery-backed or nonvolatile memory. A PLC is an example of a **real-time system** since the output of the system controlled by the PLC depends on the input conditions.

The PLC is, then, basically a digital computer designed for use in machine control. Unlike a personal computer, it has been designed to operate in the industrial environment and is equipped with special input/output interfaces and a control programming language.

Initially the PLC was used to replace **relay logic**, but its ever-increasing range of functions means that it is found in many and more complex applications. Because the structure of a PLC is based on the same principles as those employed in computer architecture, it is capable not only of performing relay switching tasks but also of performing other applications such as timing, counting, calculating, comparing, and the processing of analog signals.

Programmable controllers offer several advantages over a conventional relay type of control. Relays have to be hardwired to perform a specific function. When the system requirements change, the relay wiring has to be changed or modified. In extreme cases, such as in the auto industry, complete control panels had to be replaced since it was not economically feasible to rewire the old panels with each model changeover. The programmable controller has eliminated much of the hardwiring associated with conventional relay control circuits (**Figure 1-1**). It is small and inexpensive compared to equivalent relay-based process control systems. Modern control systems still include relays, but these are rarely used for logic.



(*a*) (*b*)

Figure 1-1 Relay- and PLC-based control panels. (*a*) Relay-based control panel. (*b*) PLC-based control panel.

Source: (*a*) Courtesy Mid-Illini Technical Group, Inc.; (*b*) Photo courtesy Ramco Electric Ltd., Toronto.

PLCs provide many other benefits including:

- ***Increased Reliability.*** Once a program has been written and tested, it can be easily downloaded to other PLCs. Since all the logic is contained in the PLC's memory, there is no chance of making a logic wiring error (**Figure 1-2**). The program takes the place of much of the external wiring that would normally be required for control of a process. Hardwiring, though still required to connect field devices, is less intensive. PLCs also offer the reliability associated with solid-state components.

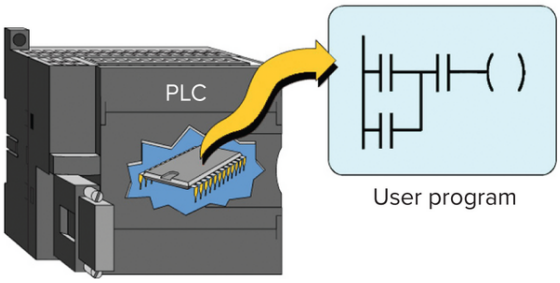


Figure 1-2 All the logic is contained in the PLC's memory.

- ***More Flexibility.*** It is easier to create and change a program in a PLC than to wire and rewire a circuit. With a PLC the relationships between the inputs and outputs are determined by the user program instead of the manner in which they are interconnected (**Figure 1-3**). Original

equipment manufacturers can provide system updates by simply sending out a new program. End users can modify the program in the field, or if desired, security can be provided by hardware features such as key locks and by software passwords.

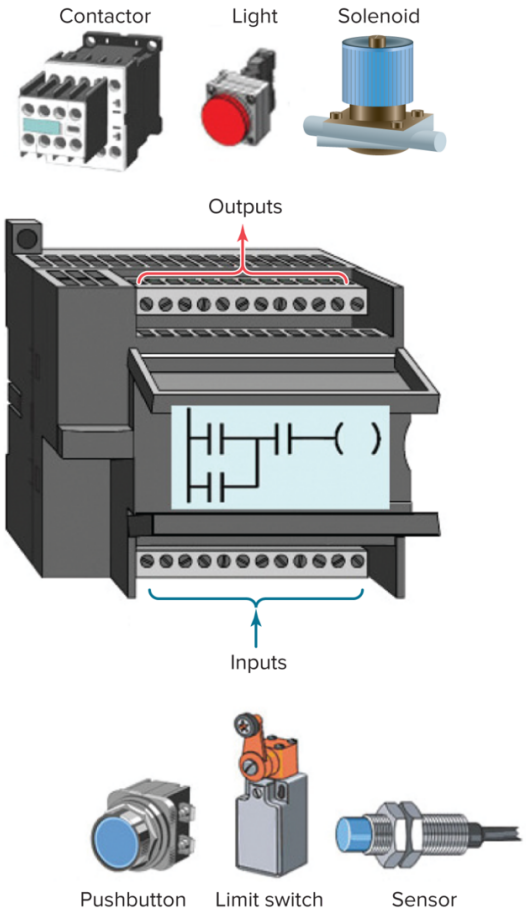


Figure 1-3 Relationships between the inputs and outputs are determined by the user program.

- ***Lower Cost.*** PLCs were originally designed to replace relay control logic, and the cost savings have been so significant that relay control is becoming obsolete except for power applications. Generally, if an application has more than about a half-dozen control relays, it will probably be less expensive to install a PLC.
- ***Communications Capability.*** A PLC can communicate with other controllers or computer equipment to perform such functions as supervisory control, data gathering, monitoring devices and process parameters, and download and upload of programs.
- ***Faster Response Time.*** PLCs are designed for high-speed and real-time applications (**Figure 1-4**). The programmable controller operates in real time, which means that an event taking place in the field will result in the execution of an operation or output. Machines that process thousands of items per second and objects that spend only a fraction of a second in front of a sensor require the PLC's quick-response capability.

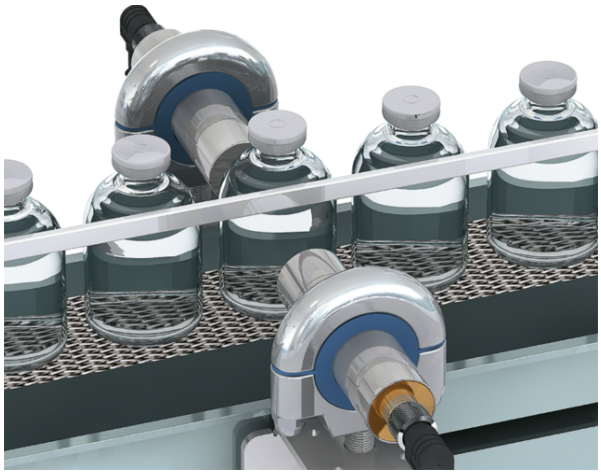


Figure 1-4 High-speed counting.
Source: Courtesy Banner Engineering Corp.

- ***Easier to Troubleshoot.*** PLCs have resident diagnostics and override functions that allow users to easily trace and correct software and hardware problems. To find and fix problems, users can display the control program on a monitor and watch it in real time as it executes (**Figure 1-5**).

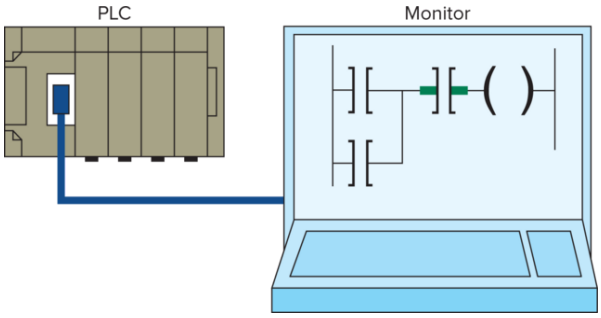


Figure 1-5 Control program can be displayed on a monitor in real time.

- ***Easier to Test Field Devices.*** A PLC control panel has the ability to check field devices at a common point. For example, a control system consisting of hundreds of input and output field devices may be contained within a very large manufacturing area. Thus, it would take a considerable amount of time to check each device at its location. By having each device wired back to a common point on a PLC module, each device could be checked for operation fairly quickly.

OceanofPDF.com

1.2 Parts of a PLC

A typical PLC can be divided into parts, as illustrated in **Figure 1-6**. These are the *central processing unit (CPU)*, the *input/output (I/O)* section, the *power supply,* and the *programming device.* The term *architecture* can refer to PLC hardware, to PLC software, or to a combination of both. An *open* architecture design allows the system to be connected easily to devices and programs made by other manufacturers. Open architectures use off-the-shelf components that conform to approved standards. A system with a *closed* architecture is one whose design is proprietary, making it more difficult to connect to other systems. Most PLC systems are in fact **proprietary**, so you must be sure that any generic hardware or software you may use is compatible with your particular PLC. Also, although the principal concepts are the same in all methods of programming, there are differences in addressing, memory allocation, retrieval, and data handling for different models. Consequently, PLC programs cannot be interchanged among different PLC manufacturers.

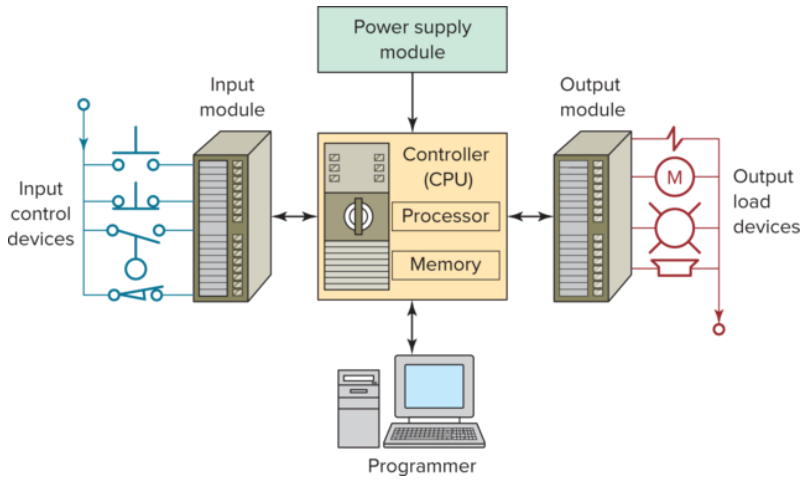


Figure 1-6 Typical parts of a programmable logic controller.

There are two ways in which I/Os (Inputs/Outputs) are incorporated into the PLC: fixed and modular. ***Fixed I/O*** (**Figure 1-7**) is typical of small PLCs that come in one package with no separate, removable units. The processor and I/O are packaged together, and the I/O terminals will have a fixed number of connections built in for inputs and outputs. The main advantage of this type of packaging is lower cost. The number of available I/O points varies and usually can be expanded by buying additional units of fixed I/O. One disadvantage of fixed I/O is its lack of flexibility; you are limited in what you can get in the quantities and types dictated by the packaging. Also, for some models, if any part in the unit fails, the whole unit has to be replaced.

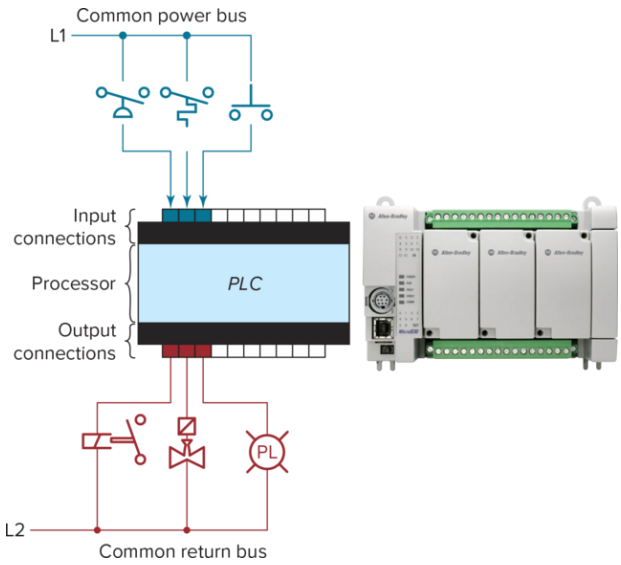


Figure 1-7 Fixed I/O configuration.

Source: Mitsubishi Electric Automation, Inc.

Modular I/O (Figure 1-8) is divided by compartments into which separate modules can be plugged. This feature greatly increases your options and the unit's flexibility. You can choose from the modules available from the manufacturer and mix them any way you desire. The basic modular controller consists of a rack, power supply, processor module (CPU), input/output (I/O modules), and an operator interface for programming and monitoring. The modules plug into a rack. When a module is slid into the rack, it makes an electrical connection with a series of contacts called the backplane, located at the rear of the rack. Communication between modules is accomplished by the backplane rail that each module plugs into.

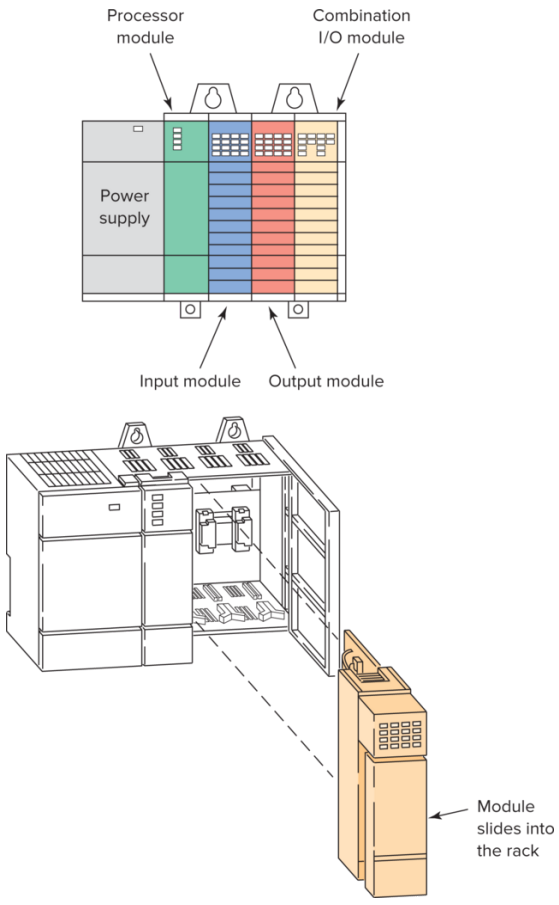


Figure 1-8 Modular I/O configuration.

The ***power supply*** supplies DC power through the backplane to the processor and the other modules that plug into the rack (**Figure 1-9**). For large PLC systems, this power supply does not normally supply power to the field devices. With most systems, power to field devices is provided by external alternating current (AC) or direct current (DC) supplies. For some small micro PLC systems, the power supply may be used to power field devices.



Figure 1-9 The power supply supplies DC power to other modules that plug into the rack.

Source: Photo of PLC Modicon M340 ©Schneider Electric, 2010. **www.schneider-electric.com.**

The ***processor*** **(CPU)** is the "brain" of the PLC. A typical processor usually consists of a microprocessor for implementing the logic and controlling the communications among the modules. The processor requires memory for storing user program instructions, numerical values, and I/O devices status.

The CPU controls all PLC activity and is responsible for running the program. The PLC program is executed as part of a repetitive process referred to as a scan (**Figure 1-10**). A typical PLC scan starts with the CPU reading the status of inputs. Then, the program logic is executed. Once the program execution is completed, the status of all outputs is updated. Next, the CPU performs internal diagnostic and communication tasks. This process is repeated continuously as long as the PLC is in the run mode.

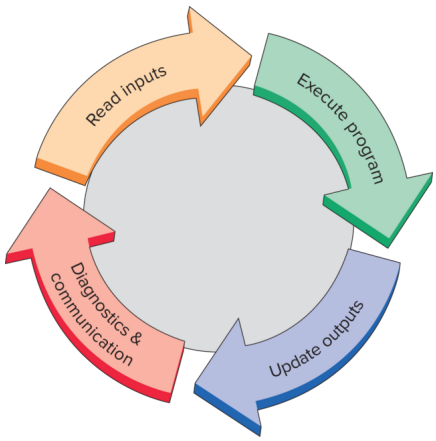


Figure 1-10 Typical PLC scan cycle.

The ***I/O system*** forms the interface by which field devices are connected to the controller (**Figure 1-11**). The purpose of this interface is to condition the various signals received from or sent to external field devices. Input devices such as pushbuttons, limit switches, and sensors are hardwired to the input terminals. Output devices such as small motors, motor starters, solenoid valves, and indicator lights are hardwired to the output terminals. To electrically isolate the internal components from the input and output terminals, PLCs commonly employ an optical isolator, which uses light to couple the circuits together. The external devices are also referred to as "field" or "real-world" inputs and outputs. The terms ***field*** or ***real world*** are used to distinguish actual external devices that exist and must be physically wired from the internal user program that emulates the function of relays, timers, and counters.

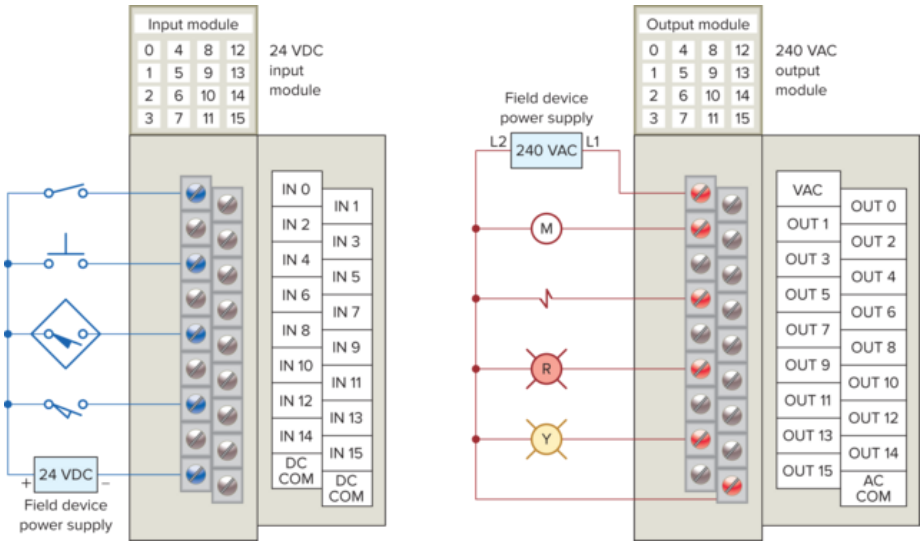


Figure 1-11 Typical PLC input/output (I/O) system connections.

A ***programming device*** is used to interface with the PLC in order to develop and transfer logic programs, download or upload data, or supply diagnostic functions to troubleshoot PLC systems. The device may be a dedicated handheld type or a personal computer running special application software. Removing the programming device will not affect the operation of the program.

A laptop computer is the most commonly used programming device. Most brands of PLCs have software available so that a laptop can be used as the programming device. This software allows users to create, edit, document, store, and troubleshoot ladder logic programs. The computer monitor is able to display more logic on the screen than can hand-held types, thus simplifying the interpretation of the program. The programming device communicates with the PLC processor via a serial or parallel data communications link, or Ethernet.

The logic ***program*** is a user-developed series of instructions that directs the PLC to execute actions. A *programming language* provides rules for combining the instructions so that they produce the desired actions. ***Relay ladder logic (RLL)*** is the standard programming language used with PLCs. Its origin is based on electromechanical relay control. The relay ladder logic program graphically represents rungs of contacts, coils, and special instruction blocks. RLL was originally designed for easy use and understanding for its users and has been modified to keep up with the increasing demands of industry's control needs.

OceanofPDF.com

1.3 Principles of Operation

To get an idea of how a PLC operates, consider the simple process control problem illustrated in Figure 1-12. Here a mixer motor is to be used to automatically stir the liquid in a vat when the temperature and pressure reach preset values. In addition, direct manual operation of the motor is provided by means of a separate pushbutton station. The process is monitored with temperature and pressure sensor switches that close their respective contacts when conditions reach their preset values.

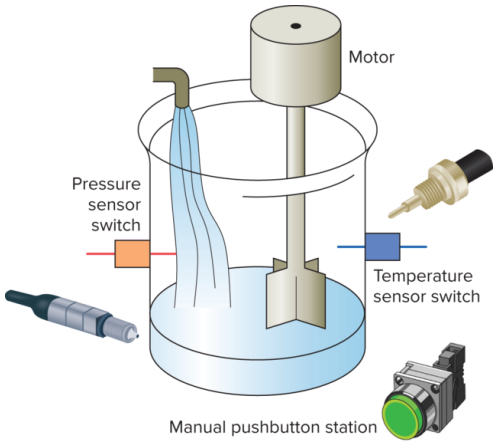


Figure 1-12 Mixer process control problem.

This control problem can be solved using the relay method for motor control shown in the relay ladder diagram of Figure 1-13. The motor starter coil (M) is energized when both the pressure and temperature switches are closed or when the manual pushbutton is pressed.

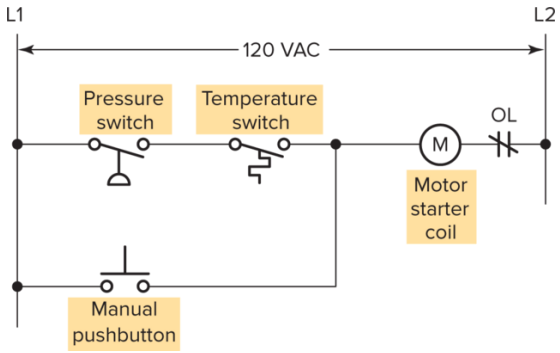


Figure 1-13 Process control relay ladder diagram.

Now let's look at how a programmable logic controller might be used for this application. The same input field devices (pressure switch, temperature switch, and pushbutton) are used. Each of these

devices is wired to a terminal on the 120 VAC input module as shown in Figure 1-14. The processor memory location addresses used are:

I/1 for the Pressure switch

I/2 for the Temperature switch

I/3 for the Manual pushbutton

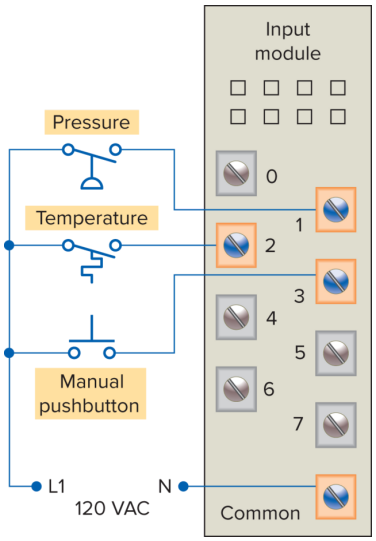


Figure 1-14 Typical wiring connections for a 120 VAC modular configured input module.

The same output field device (motor starter coil) would also be used. This device is wired to a terminal on the 120 VAC output module as shown in Figure 1-15. The processor memory location address used for the Motor starter coil is O/1.

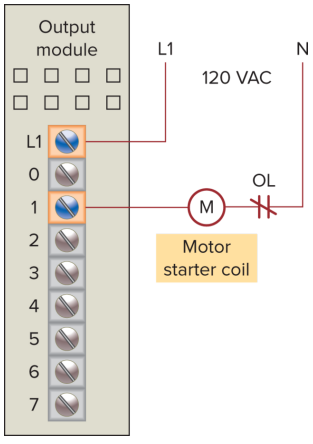


Figure 1-15 Typical wiring connections for a 120 VAC modular configured output module.

Next, the PLC ladder logic program would be constructed and entered into the memory of the CPU. A typical ladder logic program for this process is shown in **Figure 1-16**. The format used is similar to the layout of the hardwired relay ladder circuit. The individual symbols represent instructions, whereas the numbers represent the instruction memory location addresses. To program the controller, you enter these instructions one by one into the processor memory from the programming device. Each input and output device is given an address, which lets the PLC know where it is physically connected. Note that the I/O address format will differ, depending on the PLC model and manufacturer. Instructions are stored in the user program portion of the processor memory. During the program scan the controller monitors the inputs, executes the control program, and changes the output accordingly.

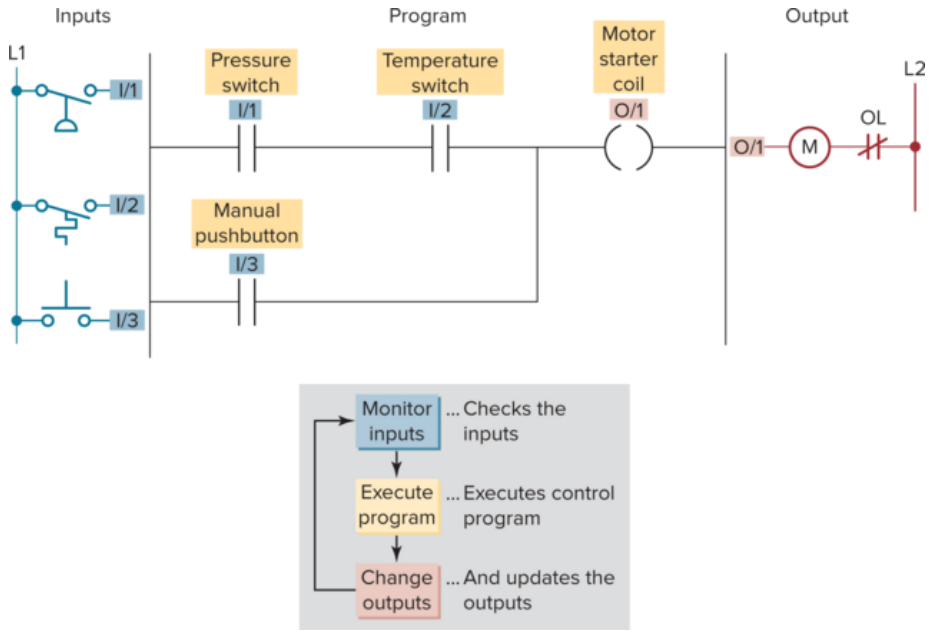


Figure 1-16 Process control PLC ladder logic program with typical addressing scheme.

For the program to operate, the controller is placed in the RUN mode, or operating cycle. During the program scan, the controller monitors the inputs, executes the control program, and changes the output accordingly. Each ⫫ symbol (looks like a normally open contact) is an instruction. The ( ) symbol is considered to represent a coil that, when energized, will energize the device that is wired to the respective output. In the ladder logic program of **Figure 1-16**, the coil O/1 is energized when contacts I/1 and I/2 are closed or when contact I/3 is closed. Either of these conditions provides a continuous logic path from left to right across the rung that includes the coil.

A programmable logic controller operates in real time in that an event taking place in the field will result in an operation or output taking place. The RUN operation for the process control logic can be described by the following sequence of events:

- First, the pressure switch, temperature switch, and pushbutton inputs are examined and their status is recorded in the controller's memory.

- A closed contact is recorded in memory as logic 1 and an open contact as logic 0.
- Next the ladder diagram is evaluated, with each internal contact given an OPEN or CLOSED status according to its recorded 1 or 0 state.
- When the states of the input contacts provide logic continuity from left to right across the rung, the output coil memory location is given a logic 1 value and the output module interface contacts will close.
- When there is no logic continuity of the program rung, the output coil memory location is set to logic 0 and the output module interface contacts will be open.
- The completion of one cycle of this sequence by the controller is called a *scan.* The scan time, the time required for one full cycle, provides a measure of the speed of response of the PLC.
- Generally, the output memory location is updated during the scan but the actual output is not updated until the end of the program scan during the I/O scan.

Figure 1-17 shows the typical wiring required to implement the process control scheme using a fixed PLC controller. In this example, the Allen-Bradley Pico controller equipped with 8 inputs and 4 outputs is used to control and monitor the process. Installation can be summarized as follows:

- Fused power lines, of the specified voltage type and level, are connected to the controller's L1 and L2 terminals.
- The pressure switch, temperature switch, and pushbutton field input devices are hardwired between L1 and controller input terminals I1, I2, and I3, respectively.
- The motor starter coil connects directly to L2 and in series with Q1 relay output contacts to L1.
- The ladder logic program is entered using the front keypad and LCD display.
- Pico programming software is also available that allows you to create as well as test your program using a personal computer.

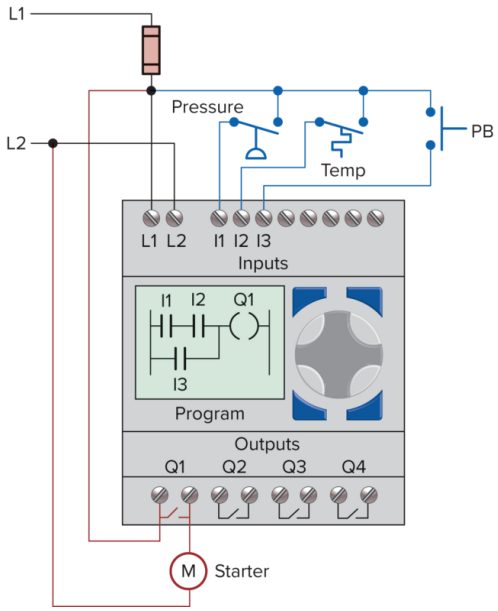


Figure 1-17 Typical wiring required to implement the process control scheme using a

fixed PLC controller.

OceanofPDF.com

1.4 Modifying the Operation

One of the important features of a PLC is the ease with which the program can be changed. For example, assume that the original process control circuit for the mixing operation must be modified as shown in the relay ladder diagram of **Figure 1-18**. The change requires that the manual pushbutton control be permitted to operate at any pressure, but not unless the specified temperature setting has been reached.

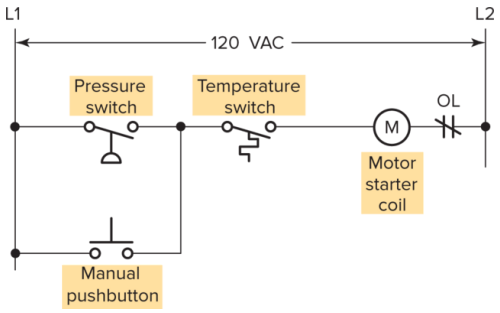


Figure 1-18 Relay ladder diagram for the modified process.

If a relay system were used, it would require some rewiring of the circuit shown in **Figure 1-18** to achieve the desired change. However, if a PLC system were used, no rewiring would be necessary. The inputs and outputs are still the same. All that is required is to change the PLC ladder logic program as shown in **Figure 1-19**.

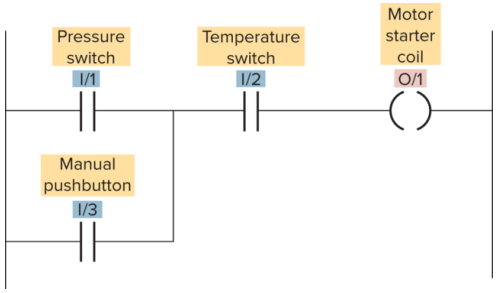


Figure 1-19 PLC ladder logic program for the modified process.

At times, a process may call for additional real input or output field devices to be added to the circuit. The relay ladder diagram of **Figure 1-20** shows the circuit further modified to include a process ON/OFF switch. To accomplish this, using hard-wired circuit control requires accessing both the pressure switch and manual pushbutton and rewiring the circuit so that the two are in parallel with each other and in series with the ON/OFF switch. The modification implemented using a PLC ladder logic program is shown in **Figure 1-21**. Note that the original wiring of the existing PLC input

circuit remains the same. All that is required is the connection of the ON/OFF switch to I/4 input and the related change in the PLC program.

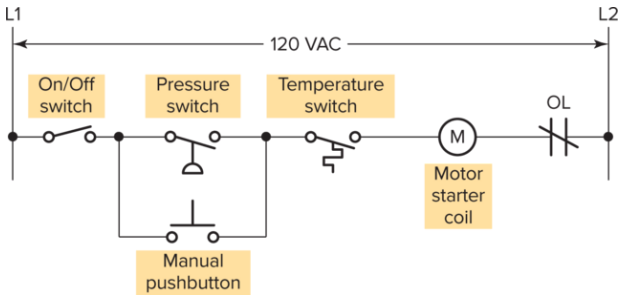


Figure 1-20 Modified relay ladder for the addition of an ON/OFF switch.

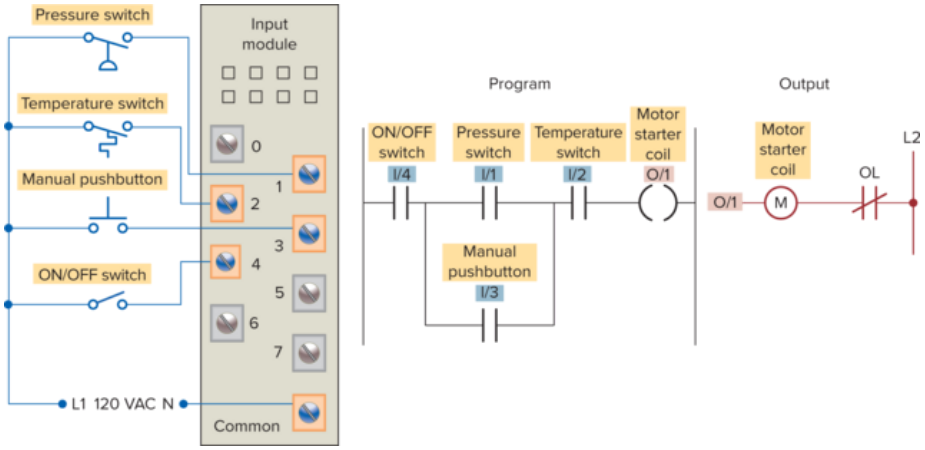


Figure 1-21 Modified PLC ladder logic program and ON/OFF switch module connection.

OceanofPDF.com

1.5 PLCs versus Computers

The architecture of a PLC is basically the same as that of a personal computer (PC). Characteristics that distinguish PLCs from PC include:

- ***Hardware*.** Unlike PCs, the PLC has no permanently attached keyboard, CD drive, or monitor.
- ***Operating Environment.*** PLCs are designed to operate in the industrial environment a wide range of ambient temperature, humidity, and electrical noise factors.
- ***Programming.*** The PLC is programmed in relay ladder logic or four other types of programming languages and comes with its program language built into its memory.
- ***Program Execution.*** PLCs execute a single program in a sequential order, while computers execute several programs or tasks simultaneously in any order.

Software associated with a PLC but written and run on a personal computer falls into the following two broad categories:

- PLC software that allows the user to program and document gives the user the tools to write a PLC program—using ladder logic or another programming language—and document or explain the program in as much detail as is necessary.
- HMI software that allows the user to monitor and control the process is called a ***human machine interface (HMI)***. It enables the user to view a process—or a graphical representation of a process—on a monitor, operate the machine, trend values, and receive alarm conditions (**Figure 1-22**). PLCs can be integrated with HMIs but the same software does not program both devices.



Figure 1-22 Human Machine Interface (HMI).

Source: Image Courtesy of Rockwell Automation, Inc.

Most recently automation manufacturers have responded to the increased requirements of industrial control systems by blending the advantages of PLC-style control with that of PC-based systems. Such a device has been termed a *programmable automation controller*, or *PAC*. Programmable automation controllers combine PLC ruggedness with PC functionality. Using PACs, you can build advanced systems incorporating software capabilities such as advanced control, communication, data logging,

and signal processing with rugged hardware performing logic, motion, process control, and vision. One main difference between PLC- and PAC-based programs is how the program is executed. A PLC mixes scan-based and event-driven program execution, whereas PAC software is typically event-driven. The scan-based execution of a PLC program might take longer because the system needs to complete the higher priority actions in the cycle first.

OceanofPDF.com

1.6 PLC Size and Application

The criteria used in categorizing PLCs include functionality, number of inputs and outputs, cost, and physical size. Of these, the ***I/O count*** is the most important factor.

There are three major types of PLC application: single-ended, multitask, and control management. A ***single-ended*** or **stand-alone** PLC application involves one PLC controlling one process. This would be a stand-alone unit and would not be used for communicating with other computers or PLCs.

A ***multitask*** PLC application involves one PLC controlling several processes. Adequate I/O capacity is a significant factor in this type of installation. In addition, if the PLC would be a subsystem of a larger process and would have to communicate with a central PLC or computer, provisions for a data communications network are also required.

A ***control management*** PLC application involves one PLC controlling several others (**Figure 1-23**). This kind of application requires a large PLC processor designed to communicate with other PLCs and possibly with a computer. The control management PLC supervises several PLCs by downloading programs that tell the other PLCs what has to be done. It must be capable of connection to all PLCs so that by proper addressing it can communicate with any one it wishes to.

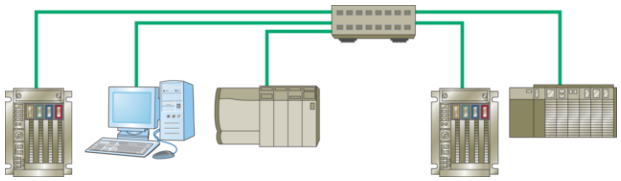


Figure 1-23 Control management PLC application.

Memory is the part of a PLC that stores data, instructions, and the control program. Memory size is usually expressed in K values: 1 K, 6 K, 12 K, and so on. The measurement kilo, abbreviated K, normally refers to 1000 units. When dealing with computer or PLC memory, however, 1 K means 1024, because this measurement is based on the binary number system $\left(2^{10} = 1024\right)$.

Although it is common for us to measure the memory capacity of PLCs in words, we need to know the number of bits in each word before memory size can be accurately compared. Modern computers usually have a word size of 16, 32, or 64 bits. The amount of memory required depends on the application. Factors affecting the memory size needed for a particular PLC installation include:

- Number of I/O points used
- Size of control program
- Data-collecting requirements
- Supervisory functions required
- Future expansion

The ***instruction set*** for a particular PLC lists the different types of instructions supported. Typically, this ranges from 15 instructions on smaller units up to 100 instructions on larger, more powerful units (see **Table 1-1**).

Table 1-1 Typical PLC Instructions

Instruction	Operation
XIC (Examine ON)	Examine a bit for an ON or 1 condition
XIO (Examine OFF)	Examine a bit for an OFF or 0 condition
OTE (Output Energize)	Turn ON a bit (nonretentive)
OTL (Output Latch)	Latch a bit (retentive)
OTU (Output Unlatch)	Unlatch a bit (retentive)
TOF (Timer Off-Delay)	Turn an output ON or OFF after its rung has been OFF for a preset time interval
TON (Timer On-Delay)	Turn an output ON or OFF after its rung has been ON for a preset time interval
CTD (Count Down)	Use a software counter to count down from a specified value
CTU (Count Up)	Use a software counter to count up to a specified value

OceanofPDF.com



CHAPTER 1 REVIEW QUESTIONS

1. What is a programmable logic controller (PLC)?
2. Identify four tasks in addition to relay switching operations that PLCs are capable of performing.
3. List six distinct advantages that PLCs offer over conventional relay-based control systems.
4. Explain the differences between open and proprietary PLC architecture.
5. State two ways in which I/O is incorporated into the PLC.
6. Describe how the I/O modules connect to the processor in a modular-type PLC configuration.
7. Explain the main function of each of the following major components of a PLC:
   a. Processor module (CPU)
   b. I/O modules
   c. Programming device
   d. Power supply module
8. What are the two most common types of PLC programming devices?
9. Explain the terms *program* and *programming language* as they apply to a PLC.
10. What is the standard programming language used with PLCs?
11. Answer the following with reference to the process control relay ladder diagram of **Figure 1-13** of this chapter:
    a. When do the pressure switch contacts close?
    b. When do the temperature switch contacts close?
    c. How are the pressure and temperature switches connected with respect to each other?
    d. Describe the two conditions under which the motor starter coil will become energized.
    e. What is the approximate value of the voltage drop across each of the following when their contacts are open?

    (1) Pressure switch
    (2) Temperature switch
    (3) Manual pushbutton
12. The programmable controller operates in real time. What does this mean?
13. Answer the following with reference to the process control PLC ladder logic diagram of **Figure 1-16** of this chapter:
    a. What do the individual symbols represent?
    b. What do the numbers represent?
    c. What field device is the number I/2 identified with?

d. What field device is the number O/1 identified with?

e. What two conditions will provide a continuous path from left to right across the rung?

f. Describe the sequence of operation of the controller for one scan of the program.

14. Compare the method by which the process control operation is changed in a relay-based system to the method used for a PLC-based system.

15. Compare the PLC and PC with regard to:

a. Physical hardware differences

b. Operating environment

c. Method of programming

d. Execution of program

16. What two categories of software written and run on PCs are used in conjunction with PLCs?

17. What is a programmable automation controller (PAC)?

18. List four criteria by which PLCs are categorized.

19. Compare the single-ended, multitask, and control management types of PLC applications.

20. What is the memory capacity, expressed in bits, for a PLC that uses 16-bit words and has an 8 K word capacity?

21. List five factors affecting the memory size needed for a particular PLC installation.

22. What does the instruction set for a particular PLC refer to?

OceanofPDF.com



CHAPTER 1 PROBLEMS

1. Given two single-pole switches, write a program that will turn on an output when both switch *A* and switch *B* are closed.
2. Given two single-pole switches, write a program that will turn on an output when either switch *A* or switch *B* is closed.
3. Given four NO (Normally Open) pushbuttons (*A-B-C-D*), write a program that will turn a lamp on if pushbuttons *A* and *B* or *C* and *D* are closed.
4. Write a program for the relay ladder diagram shown in **Figure 1-24**.

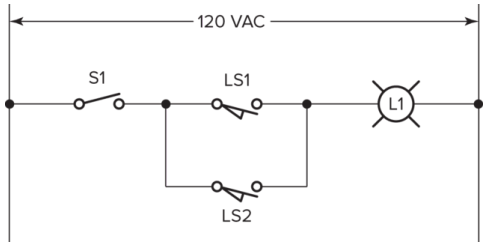


Figure 1-24 Circuit for Problem 4.

5. Write a program for the relay ladder diagram shown in **Figure 1-25**.

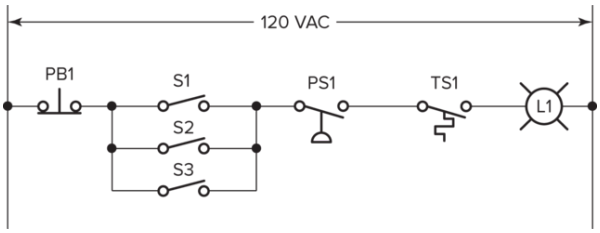


Figure 1-25 Circuit for Problem 5.

OceanofPDF.com

2

PLC Hardware Components



genkur/Shutterstock

Chapter Objectives

After completing this chapter, you will be able to:

- List and describe the function of the hardware components used in PLC systems
- Describe the basic circuitry and applications for discrete and analog I/O modules, and interpret typical I/O and CPU specifications
- Differentiate between tag-based and rack/slot-based addressing.
- Describe the general classes and types of PLC memory devices
- List and describe the different types of PLC peripheral support devices available
- Be familiar with the execution of the three-wire PLC motor control program

This chapter exposes you to the details of PLC hardware and modules that make up a PLC control system. The chapter's illustrations show the various parts of a PLC as well as general connection paths. In this chapter we discuss the CPU and memory hardware components, including the various types of memory that are available, and we describe the hardware of the input/output section, including the difference between the discrete and analog types of modules.

OceanofPDF.com

2.1 The I/O Section

The **input/output (I/O)** section of a PLC is the section to which all field devices are connected and provides the interface between them and the CPU. Input/output arrangements are built into a fixed PLC while modular types use external I/O modules that plug into the PLC.

Figure 2-1 illustrates a rack-based I/O section made up of individual I/O modules. Input interface modules accept signals from the machine or process devices and convert them into signals that can be used by the controller. Output interface modules convert controller signals into external signals used to control the machine or process. A typical PLC has room for several I/O modules, allowing it to be customized for a particular application by selecting the appropriate modules. Each slot in the rack is capable of accommodating any type of I/O module.

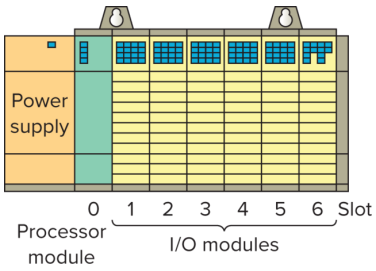


Figure 2-1 Rack-based I/O section.

The I/O system provides an interface between the hardwired components in the field and the CPU. The input interface allows *status information* regarding processes to be communicated to the CPU, and thus allows the CPU to communicate *operating signals* through the output interface to the process devices under its control.

One benefit of a PLC system is the ability to locate the I/O modules near the field devices. **Distributed I/O** refers to inputs and outputs located at a machine and controlled remotely from the PLC, as illustrated in **Figure 2-2**, in order to minimize the amount of wiring required. The processor receives signals from the remote input modules and sends signals back to their output modules via the communication module.

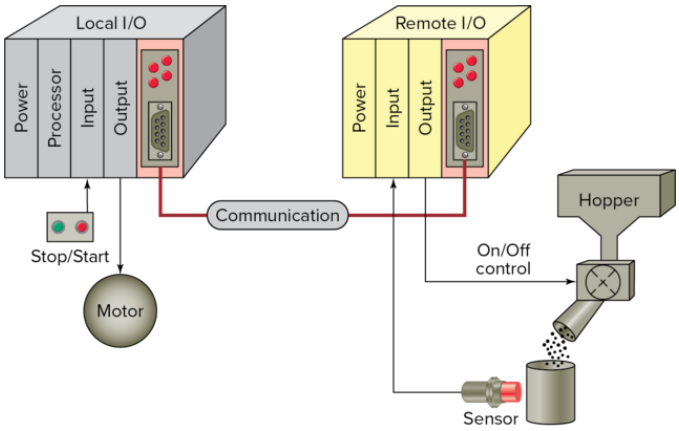


Figure 2-2 Remote I/O rack.

A rack is referred to as a ***remote*** rack when it is located away from the processor module. To communicate with the processor, the remote rack uses a special communications network. Each remote rack requires a unique station number to distinguish one from another. The remote racks are linked to the local rack through a ***communications module***. Cables connect the modules with each other. If fiber optic cable is used between the CPU and I/O rack, it is possible to operate I/O points from distances greater than 20 miles with no voltage drop. Coaxial cable will allow remote I/O to be installed at distances greater than two miles. Fiber optic cable will not pick up noise caused by adjacent high power lines or equipment normally found in an industrial environment. Coaxial cable is more susceptible to this type of noise.

The PLC's memory system stores information about the status of all the inputs and outputs. To keep track of all this information, it uses a system called *addressing.* An **address** is a label or number that indicates where a certain piece of information is located in a PLC's memory. Just as your home address tells where you live in your city, a device's or a piece of data's address tells where information about it resides in the PLC's memory. That way, if a PLC wants to find out information about a field device, it knows to look in its corresponding address location. Examples of addressing schemes include ***rack/slot-based***, versions of which are used in Allen-Bradley SLC 500 controllers, ***tag-based*** used in Allen-Bradley ControlLogix and CompactLogix controllers.

Page 16

In general, rack/slot-based addressing elements include:

Type–The type determines if an input or output is being addressed.

Slot–The slot number is the physical location of the I/O module. This may be a combination of the rack number and the slot number when using expansion racks.

Word and Slot–The word and slot are used to identify the actual terminal connection in a particular I/O module. A discrete module usually uses only one word, and each connection corresponds to a different bit that makes up the word.

With a rack/slot address system the location of a module within a rack and the terminal number of a module to which an input or output device is connected will determine the device's address.

O:4/15	**Output module in slot 4, terminal 15**
I:3/8	**Input module in slot 3, terminal 8**

O:6.0	**Output module, slot 6**
I:5.0	**Input module, slot 5**

Every input and output device connected to a discrete I/O module is addressed to a specific ***bit*** in the PLC's memory. A bit is a binary digit that can be either 1 or 0. Analog I/O modules use a *word* addressing format, which allows the entire words to be addressed. The bit part of the address is usually not used; however, bits of the digital representation of the analog value can be addressed by the programmer if necessary. **Figure 2-3** illustrates bit level and word level addressing.

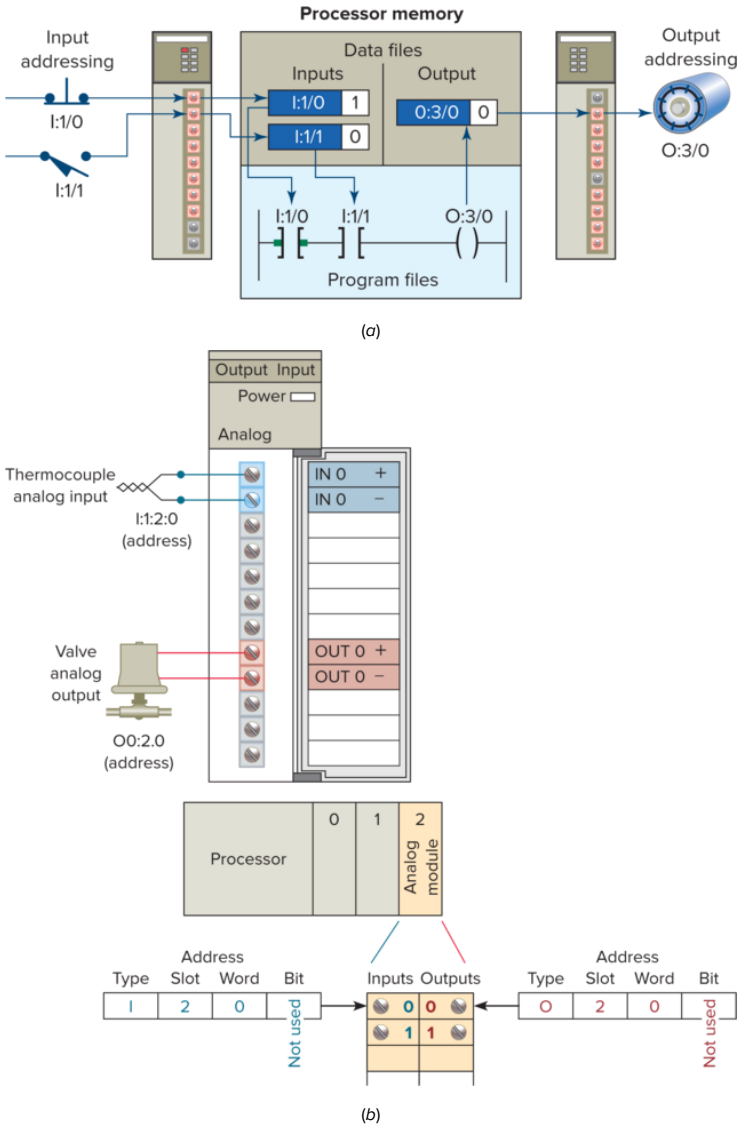


Figure 2-3 RSLogix 500 bit level and word level addressing. (*a*) Bit level addressing. (*b*) Word level addressing.

The **RSLogix 5000** family of PLCs uses a tag-based memory structure with all data being assigned an alphanumeric name called a **tag.** **Figure 2-4** illustrates the tag-based addressing format. Memory locations are defined by using base and alias tags. A **base tag** defines a memory location where data are stored. An **alias tag** is used to create an alternate name (alias) for a tag. The alias tag is often used to create a tag name to represent a real-world input or output.



Figure 2-4 Allen-Bradley RSLogix 5000 tag-based addressing format.

Source: Image Courtesy of Rockwell Automation, Inc.

Figure 2-5 shows an RSLogix 5000 program used to execute a three-wire motor control circuit. The input/output (I/O) section of the PLC, where all field devices are connected, is also shown. All data is assigned to the tags. The addressing format is summarized as follows:

- The addresses are used by the processor to identify where the device is located in order to monitor or control it.
- LEDs are connected to each terminal of the input and output module that indicate the ON or OFF status of each I/O circuit.
- Stop PB address I:1/0 refers to the Input module in slot 1, terminal 0.
- Start PB address I:1/1 refers to the Input module in slot 1, terminal 1.
- Motor address O:2/0 refers to the Output module in slot 2, terminal 0.
- Green PL address O:2/1 refers to the Output module in slot 2, terminal 1.
- Red PL address O:2/2 refers to the Output module in slot 2, terminal 2.

The operation of the circuit is based on the logical relationship between the input and output programmed instructions and is summarized as follows:

- When power is first applied, the motor stater coil M is de-energized.
- The Red PL will be on due to the state of its series associated Motor instruction.
- The Green PL will be off due to the state of its series associated Motor instruction.
- When the Start PB is momentarily closed the starter coil M is energized and remains energized by its associated parallel Motor associated instruction.
- The Red PL is switched off due to a change in state of the series associated Motor instruction.
- The Green PL is switched on due to a change in state of the series associated Motor instruction. Pressing the stop open button, at any, time rests the operation.

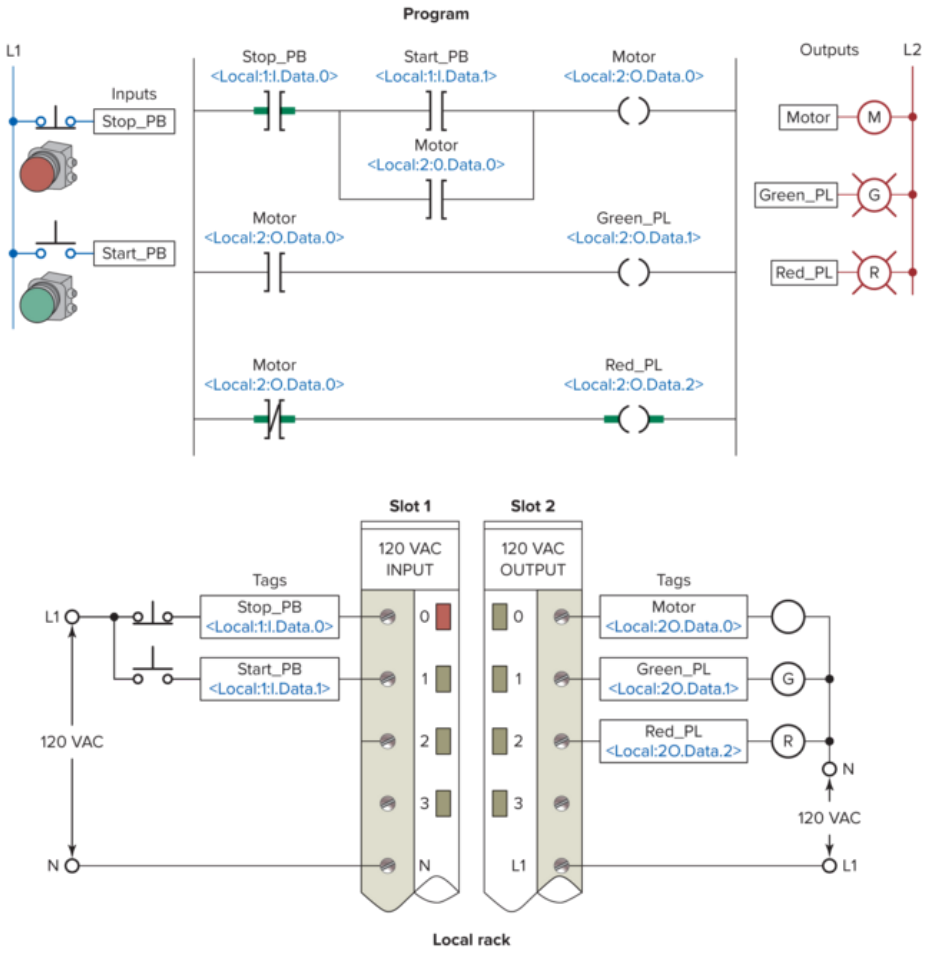


Figure 2-5 RSLogix 5000 program used to execute a three-wire motor control circuit.

An I/O module is made up of a printed circuit board and a terminal assembly. The printed circuit board contains the electronic circuitry used to interface the circuit of the processor with that of the input or output device. Modules are designed to plug into a slot or connector in the I/O rack or directly into the processor. The terminal assembly, which is attached to the front edge of the printed circuit board, is used for making field-wiring connections. Modules contain terminals for each input and output connection, status lights for each of the inputs and outputs, and connections to the power supply used to power the inputs and outputs. Terminal and status light arrangements vary with different manufacturers.

Most PLC modules have plug-in wiring terminal strips. The terminal block is plugged into the actual module as illustrated in **Figure 2-6**. If there is a problem with a module, the entire strip is removed, a new module is inserted, and the terminal strip is plugged into the new module. Unless otherwise specified, never install or remove I/O modules or terminal blocks while the PLC is powered. A module inserted into the wrong slot could be damaged by improper voltages connected through the wiring arm. Most faceplates and I/O modules are keyed to prevent putting the wrong faceplate on the wrong module. In other words, an output module cannot be placed in the slot where an input module was originally located.

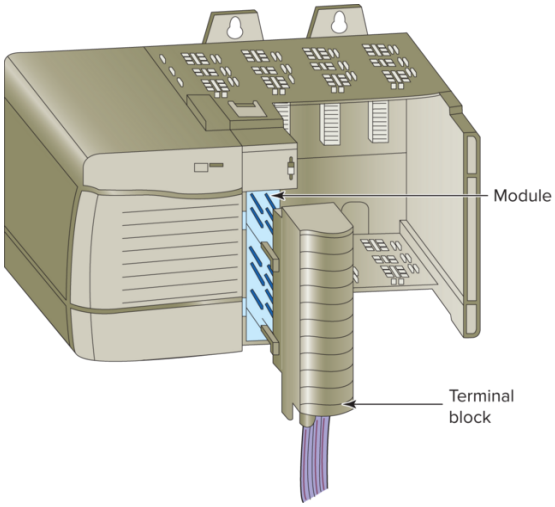


Figure 2-6 Plug-in terminal block.

Input and output modules can be placed anywhere in a rack, but they are normally grouped together for ease of wiring. I/O modules can be 8, 16, 32, or 64 point cards. The number refers to the number of inputs or outputs available. The standard I/O module has eight inputs or outputs. A ***high-density*** module may have up to 64 inputs or outputs. The advantage with the high-density module is that it is possible to install up to 64 inputs or outputs in one slot for greater space savings. The only disadvantage is that the high-density output modules cannot handle as much current per output.

OceanofPDF.com

2.2 Discrete I/O Modules

Discrete or **digital** type **input** modules (Figure 2-7) are designed to **monitor** ON/OFF devices such as selector switches, pushbuttons, and limit switches. Likewise, discrete output modules control devices such as lights, relays, solenoids, and motor starters that require simple ON/OFF switching. The classification of discrete I/O covers ***bit-oriented*** inputs and outputs. In this type of input or output, each bit represents a complete information element in itself and provides the status of some external contact or advises of the presence or absence of power in a process circuit.

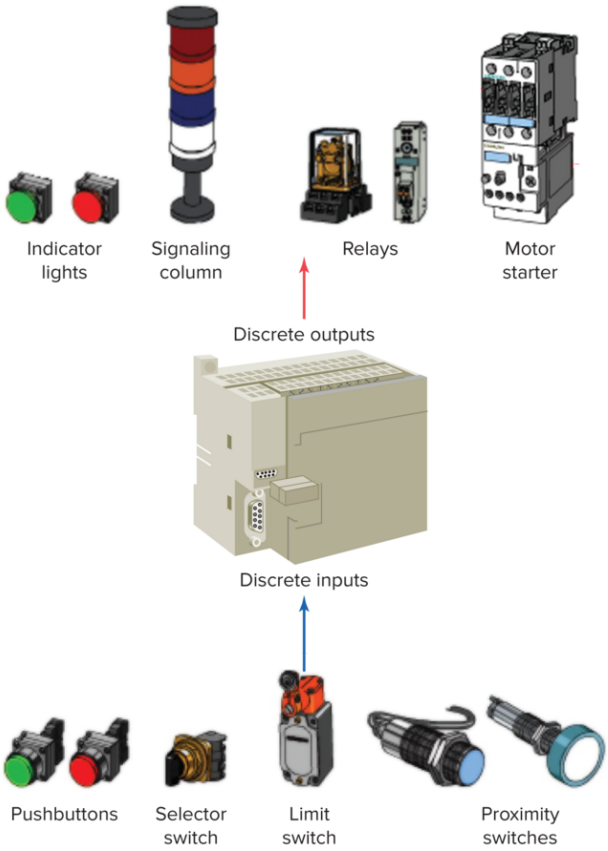


Figure 2-7 Discrete input and output devices.

Each discrete I/O module is powered by some *field-supplied* voltage source. Since these voltages can be of different magnitude or type, I/O modules are available at various AC and DC voltage ratings, as listed in Table 2-1.

Table 2-1 Common Ratings for Discrete I/O Interface Modules

Input Interfaces	Output Interfaces
12 V **AC/DC** /24 V **AC/DC**	12–48 V **AC**

Input Interfaces	Output Interfaces
48 V **AC/DC**	120 V **AC**
120 V **AC/DC**	230 V **AC**
230 V **AC/DC**	120 V **DC**
5 V **DC** (TTL level)	230 V **DC**
	5 V **DC** (TTL level)
	24 V **DC**

The modules themselves receive their voltage and current for proper operation from the backplane of the rack enclosure into which they are inserted, as illustrated in **Figure 2-8**. **Backplane** power is provided by the PLC module power supply and is used to power the electronics that reside on the I/O module circuit board. The relatively higher currents required by the loads of an output module are normally provided by user-supplied power. Module power supplies typically may be rated for 3 A, 4 A, 12 A, or 16 A depending on the type and number of modules used.

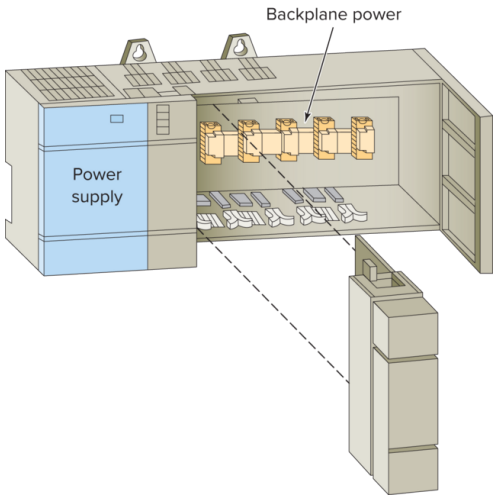


Figure 2-8 Modules receive their voltage and current from the backplane.

Figure 2-9 shows the block diagrams for one input of a typical alternating current (AC) *discrete input module.* The input circuit is composed of two basic sections: the ***power*** section and the ***logic*** section. An optical isolator is used to provide electrical isolation between the field wiring and the PLC backplane internal circuitry. The input LED turns on or off, indicating the status of the input device. Logic circuits process the digital signal to the processor. Internal PLC control circuitry typically operates at 5 VDC or less volts.

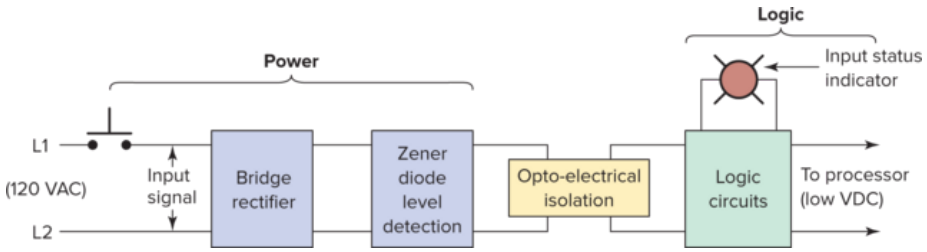


Figure 2-9 Discrete AC input module block diagram.

A simplified diagram for a single input of a discrete AC input module is shown in **Figure 2-10**. The operation of the circuit can be summarized as follows:

- The input noise filter consisting of the capacitor and resistors R1 and R2 removes false signals that are due to contact bounce or electrical interference.
- When the pushbutton is closed, 120 VAC is applied to the bridge rectifier input.
- This results in a low-level DC output voltage that is applied across the LED of the optical isolator.
- The Zener diode (Z_D) voltage rating sets the minimum threshold level of voltage that can be detected.
- When light from the LED strikes the phototransistor, it switches into conduction and the status of the pushbutton is communicated in logic to the processor.
- The optical isolator not only separates the higher AC input voltage from the logic circuits but also prevents damage to the processor due to line voltage transients. In addition, this isolation also helps reduce the effects of electrical noise, common in the industrial environment, which can cause erratic operation of the processor.
- For fault diagnosis, an input state LED indicator is on when the input pushbutton is closed. This indicator may be wired on either side of the optical isolator.
- An AC/DC type of input module is used for both AC and DC inputs as the input polarity does not matter.
- A PLC input module will have either all inputs isolated from each other with no common input connections or groups of inputs that share a common connection.

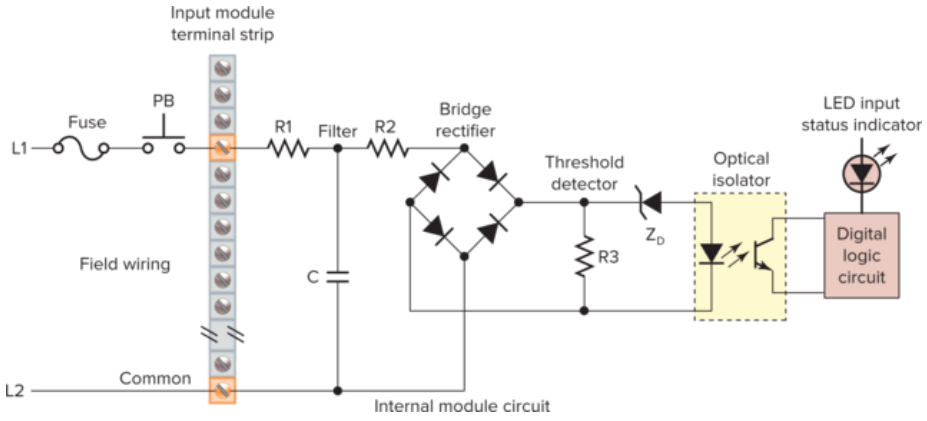


Figure 2-10 Simplified diagram for a single input of a discrete AC input module.

Discrete input modules perform four tasks in the PLC control system. They:

- Sense when a signal is received from a field device.
- Convert the input signal to the correct voltage level for the particular PLC.
- Isolate the PLC from fluctuations in the input signal's voltage or current.
- Send a signal to the processor indicating which sensor originated the signal.

Figure 2-11 shows the block diagram for one output of a typical discrete output module. Like the input module, it is composed of two basic sections: the power section and the logic section, coupled by an isolation circuit. The output interface can be thought of as an electronic switch that turns the output load device on and off. Logic circuits determine the output status. An output LED indicates the status of the output signal.

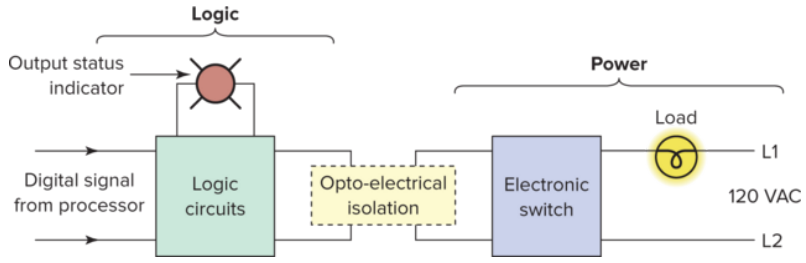


Figure 2-11 Discrete AC output module block diagram.

A simplified diagram for a single output of a discrete AC output module is shown in **Figure 2-12**. The operation of the circuit can be summarized as follows:

- As part of its normal operation, the digital logic circuit of the processor sets the output status according to the program.
- When the processor calls for an output load to be energized, a voltage is applied across the LED of the opto-isolator.
- The LED then emits light, which switches the phototransistor into conduction.
- This in turn triggers the triac AC semiconductor switch into conduction, allowing current to flow to the output load.
- Since the triac conducts in either direction, the output to the load is alternating current.
- The triac, rather than having ON and OFF status, actually has LOW and HIGH resistance levels, respectively. In its OFF state (HIGH resistance), a small leakage current of a few milliamperes still flows through the triac.
- As with input circuits, the output interface is usually provided with LEDs that indicate the status of each output.
- Fuses are normally required for the output module, and they are provided on a per circuit basis, thus allowing for each circuit to be protected and operated separately. Some modules also provide visual indicators for fuse status.
- The triac cannot be used to switch a DC load.

- For fault diagnosis, the LED output status indicator is on whenever the PLC is commanding that the output load be switched on.

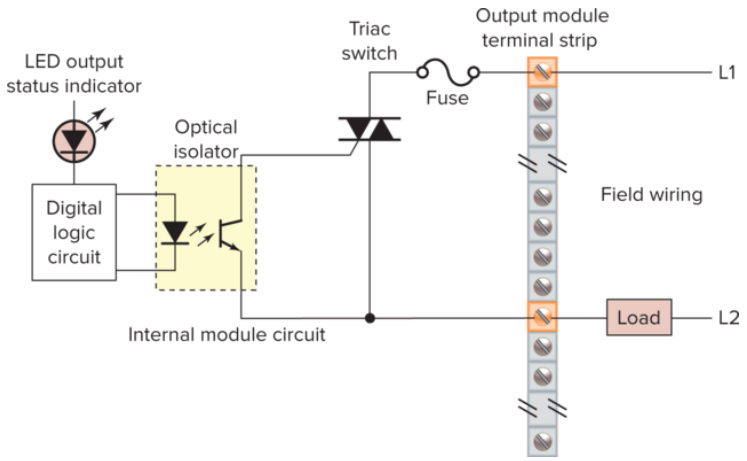


Figure 2-12 Simplified diagram for a single output of a discrete AC output module.

Individual AC outputs are usually limited by the size of the triac to 1 A or 2 A. The maximum current load for any one module is also specified. To protect the output module circuits, specified current ratings should not be exceeded. For controlling larger loads, such as large motors, a standard control relay is connected to the output module. The contacts of the relay can then be used to control a larger load or motor starter, as shown in **Figure 2-13**. When a control relay is used in this manner, it is called an ***interposing*** **relay.**

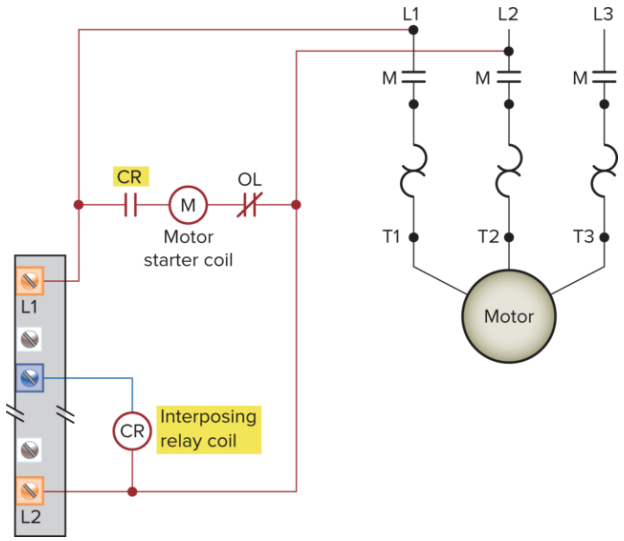


Figure 2-13 Interposing relay connection.

Discrete output modules are used to turn field output devices either on or off. These modules can be used to control any two-state device, and they are available in AC and DC versions and in various voltage ranges and current ratings. Output modules can be purchased with *transistor, triac,* or *relay* output as illustrated in **Figure 2-14**. Triac outputs can be used only for control of AC devices, whereas transistor outputs can be used only for control of DC devices. The discrete relay contact output module uses electromechanical as the switching element. These relay outputs can be used with AC or DC devices, but they have a much slower switching time compared to solid-state outputs.

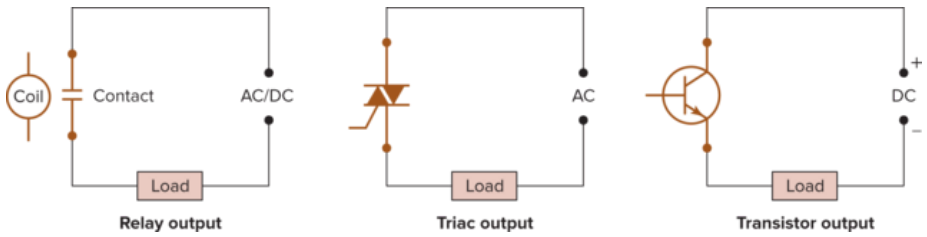


Figure 2-14 Relay, transistor, and triac switching elements.

Certain DC I/O modules specify whether the module is designed for interfacing with current-source (PNP) or current-sink (NPN) devices. If the module is a current-sourcing module, then the input or output device must be a current-sinking device. Conversely, if the module is specified as current-sinking, then the connected device must be current-sourcing. Some modules allow the user to select whether the module will act as current sinking or current sourcing, thereby allowing it to be set to whatever the field devices require. Sinking and sourcing terminology applies only to DC input and output circuits.

Allen-Bradley delineates between the various digital DC modules by sorting them into two categories: **Sinking** and **Sourcing.** These terms are used to describe a current signal flow relationship between field input and output devices. If a device provides current when it is ON, it is said to be sourcing current. Conversely, if a device receives current when it is ON, it is said to be sinking current.

Figure 2-15 and **Figure 2-16** show device connections for both sourcing and sinking configurations:

- Conventional current $(+ \text{ to } -)$ is assumed.
- In sinking devices, current flows into the device's terminal from the module (the module provides, or sources the current).
- In sourcing devices, current flows out of the device's terminal into the module (the module receives, or sinks, the current).
- A sourcing I/O device or I/O module will always have a connection directly to the positive side of the DC power supply.
- A sinking I/O device or I/O module will always have a connection directly to the negative side of the DC power supply.
- Input and output points that are sinking or sourcing only can conduct current in only one direction. Therefore, it is possible to connect the external supply and field device to the I/O point with current trying to flow in the wrong direction, and the circuit **will not operate.**

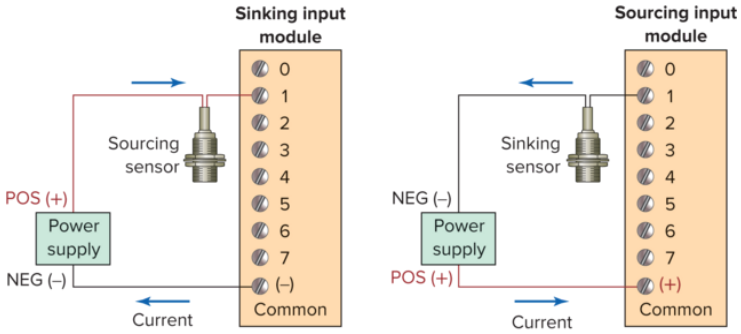


Figure 2-15 Sinking and sourcing inputs.

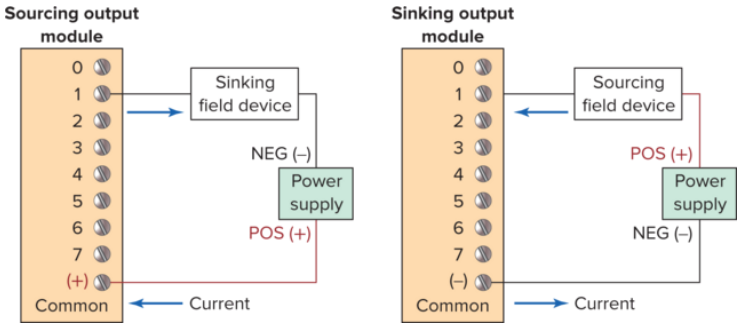


Figure 2-16 Sinking and sourcing outputs.

OceanofPDF.com

2.3 Analog I/O Modules

Analog input and output modules are used whenever the control process requires the **continuously variable** type of control, in contrast to the discrete or digital ON/OFF types. Typical analog input detection devices include temperature sensors, potentiometers, and ultrasonic proximity sensors. Typical analog output control devices include control valves, meters, and stepper motors (**Figure 2-17**).

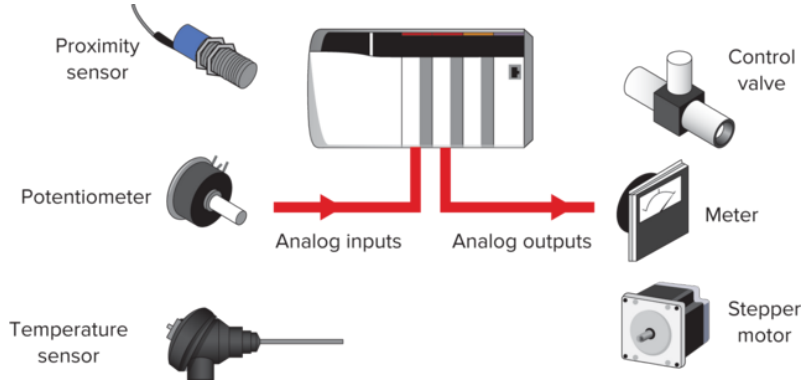


Figure 2-17 Analog input and output devices.

Discrete devices are inputs and outputs that have only two states: on and off. In comparison, **analog** devices represent physical quantities that can have an infinite number of values. Typical analog inputs and outputs vary from 0 to 20 mA, 4 to 20 mA, or 0 to 10 V. **Figure 2-18** illustrates how PLC analog input and output modules are used in measuring and displaying the level of fluid in a tank. The analog input interface module contains the circuitry necessary to accept an analog voltage or current signal from the level transmitter field device. This input is converted from an analog to a digital value for use by the processor. The circuitry of the analog output module accepts the digital value from the processor and converts it back to an analog signal that drives the field tank level meter.

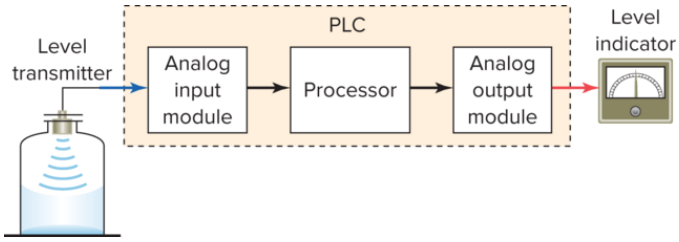


Figure 2-18 Analog input and output to a PLC.

Analog input modules normally have multiple input channels that allow 4, 8, or 16 devices to be interface to the PLC. The two basic types of analog input modules are ***voltage*** **sensing** and ***current*** **sensing.** Analog sensors measure a varying physical quantity over a specific range and generate a corresponding voltage or current signal. Common physical quantities measured by a PLC analog

module include temperature, speed, level, flow, weight, pressure, and position. For example, a sensor may measure temperature over a range of 0 to 500°C, and output a corresponding voltage signal that varies between 0 and 50 mV.

Figure 2-19 illustrates an example of a voltage sensing input analog module used to measure temperature. The connection diagram applies to an Allen-Bradley Micro-Logic 4-channel analog thermocouple input module. A varying DC voltage in the low millivolt range, proportional to the temperature being monitored, is produced by the thermocouple. This voltage is amplified and digitized by the analog input module and then sent to the processor on command from a program instruction. Because of the low voltage level of the input signal, a twisted shielded pair cable is used in wiring the circuit to reduce unwanted electrical noise signals that can be induced in the conductors from other wiring. When using an ungrounded thermocouple, the shield must be connected to ground at the module end. To obtain accurate readings from each of the channels, the temperature between the thermocouple wire and the input channel must be compensated for. A cold junction compensating (CJC) thermistor is integrated in the terminal block for this purpose.

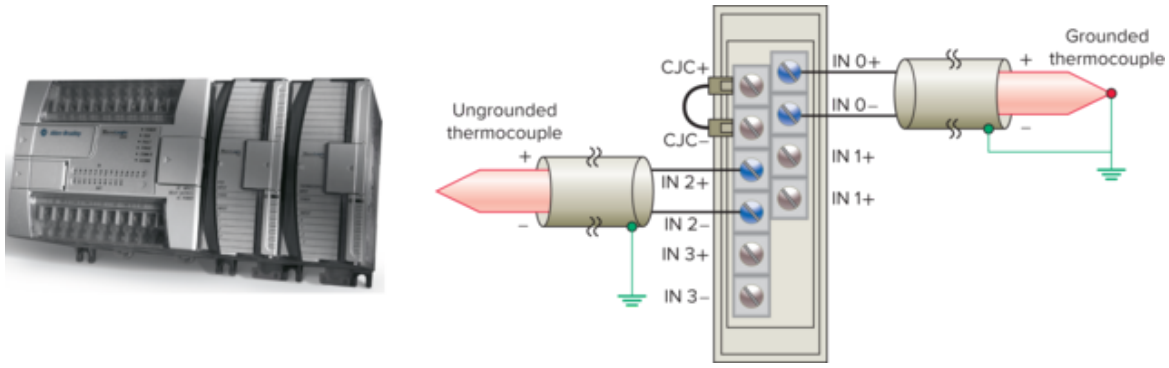


Figure 2-19 MicroLogix 4-channel analog thermocouple input module.

Source: Image Courtesy of Rockwell Automation, Inc.

Only digital signals can be handled by PLCs. The transition of an analog signal to digital values is accomplished by an analog-to-digital (A/D) converter, the main element of the analog input module. Analog voltage input modules are available in two types: unipolar and bipolar. ***Unipolar*** modules can accept an input signal that varies in the positive direction only. For example, if the field device outputs 0 to +10 V, then the unipolar modules would be used. Bipolar signals swing between a maximum negative value and a maximum positive value. For example, if the field device outputs −10 V to +10 V a bipolar module would be used. The ***resolution*** of an analog input channel refers to the smallest change in input signal value that can be sensed and is based on the number of bits used in the digital representation. Analog input modules must produce a range of digital values between a maximum and minimum value to represent the analog signal over its entire span. Typical specifications are as follows:

Span of analog input	Bipolar	10 V	−10 to +10 V
		5 V	−5 to +5 V
	Unipolar	10 V	0 to +10 V
		5 V	0 to +5 V
Resolution			0.3 mV

When connecting voltage sensing inputs, close adherence to specified requirements regarding wire length is important to minimize signal degrading and the effects of electromagnetic noise interference induced along the connecting conductors. Current input signals, which are not as sensitive to noise as voltage signals, are typically not distance limited. Current sensing input modules typically accept analog data over the range of 4 to 20 mA, but can accommodate signal ranges of −20 to +20 mA. The loop power may be supplied by the sensor or may be provided by the analog output module as illustrated in **Figure 2-20**. Shielded twisted pair cable is normally recommended for connecting any type of analog input signal.

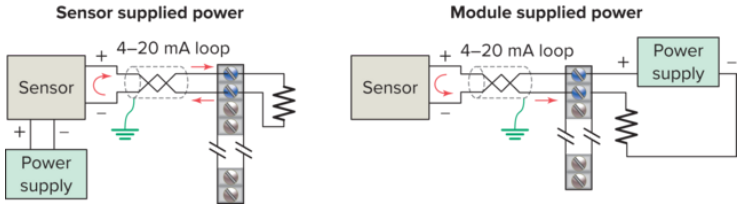


Figure 2-20 Sensor and analog module supplied power.

Field devices that provide an analog output as their signal may be connected to transmitters, or may contain their own internal transmitters to generate the analog signal to the module, as illustrated in **Figure 2-21**. A **transducer** converts a field device's variable (e.g., pressure, temperature etc.) into a very low-level electric signal (current or voltage) that can be amplified by a **transmitter** and then input into the analog module.

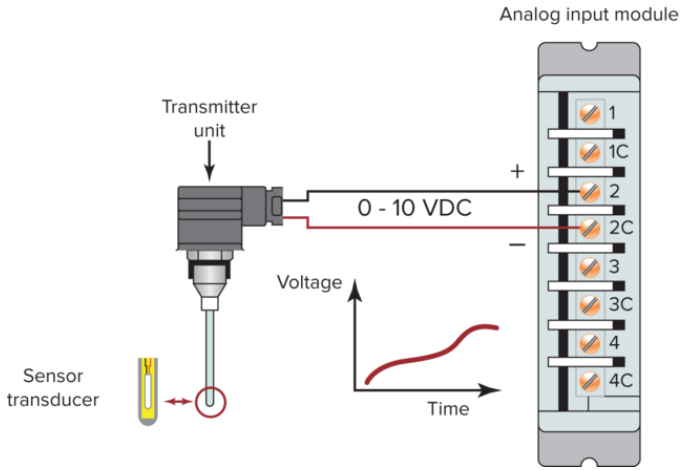


Figure 2-21 Transmitter input to an analog module.

The method used to wire two-, three-, and four-wire sensors to an analog input module is illustrated in Figure 2-22. The module does not provide loop power for analog inputs. A separate power that matches the transmitter specifications is used. All analog common (ANL COM) points are electrically connected together inside the module but not to earth ground. When wiring single-ended analog input devices to the analog input card, the number of total wires necessary can be limited by using the ANALOG COMMON terminal, as shown in Figure 2-23. Note that differential inputs are more immune to noise than single-ended inputs.

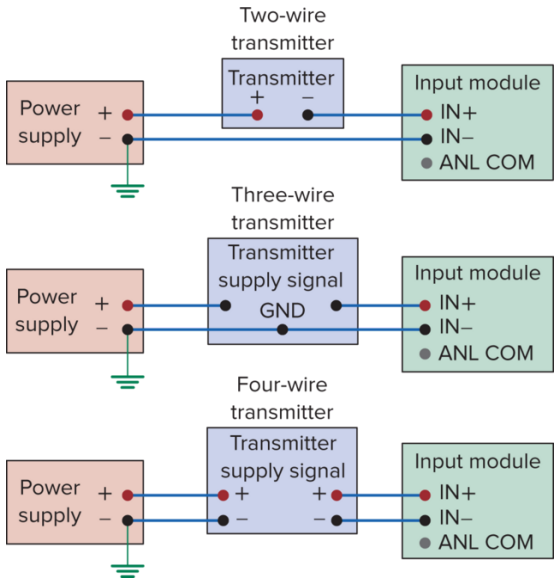


Figure 2-22 Wiring two-, three-, and four-wire sensors to an analog input module.

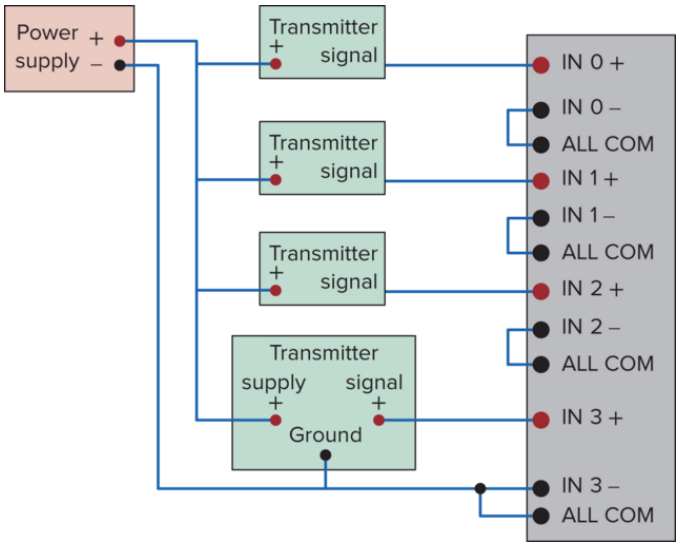


Figure 2-23 Wiring single-ended analog input devices.

The ***analog output interface module*** receives from the processor digital data, which are converted into a proportional voltage or current to control an analog field device. The transition of a digital signal to analog values is accomplished by a **digital-to-analog (D/A)** converter, the main element of the analog output module. An analog output signal is a continuous and changing signal that is varied under the control of the PLC program. Common devices controlled by a PLC analog output module include instruments, control valves, chart recorder, electronic drives, and other types of control devices that respond to analog signals. They employ standard analog output ranges such as ± 5 V, $\pm$ 10 V, 0 to 5 V, 0 to 10 V, 4 to 20 mA, or 0 to 20 mA.

Figure 2-24 illustrates the use of analog I/O modules in a typical PLC control system. In this application the PLC controls the amount of fluid placed in a holding tank. The level sensor detects the fluid level in the tank and transmits a signal to the analog input. The analog output from the PLC is used to control the flow by adjusting the percentage of the valve opening. The valve is initially open 100%. As the fluid level in the tank approaches the preset point, the processor modifies the output, which adjusts the valve to maintain a set point.

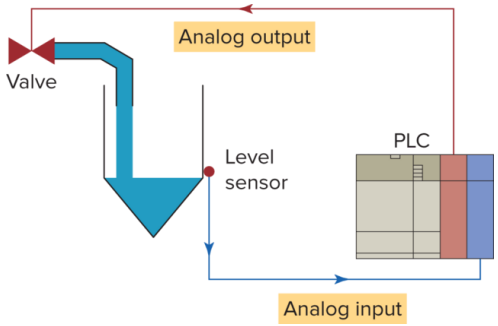


Figure 2-24 Typical analog I/O control system.

Transducers produce either voltage or current proportional to some engineering units such as temperature ($^\circ$C or $^\circ$F), pressure (lb/in^2), distance (cm), etc. **Scaling** refers to changing a quantity from one notation to another and involves:

Engineering units: The units a human uses and understands

Transducer units: Either a voltage or current

Binary, raw, or machine units: The units the processor requires

Scaling is used to produce an output value that has a linear relationship between the input and scaled values. It allows you to take an analog input from a sensor and convert it to the engineering units required by the application. **Figure 2-25** illustrates a typical application involving temperature measurement.

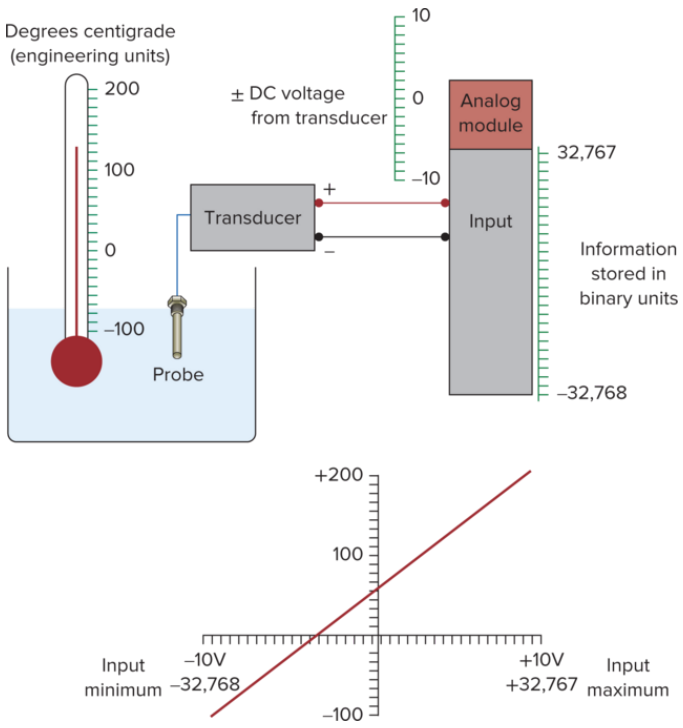


Figure 2-25 Measuring temperature.

OceanofPDF.com

2.4 Special I/O Modules

Many different types of I/O modules have been developed to meet special needs. These include:

HIGH-SPEED COUNTER MODULE

The high-speed counter module is used to provide an interface for applications requiring counter speeds that surpass the capability of the PLC ladder program. High-speed counter modules are used to count pulses (Figure 2-26) from sensors, encoders, and switches that operate at very high speeds. They have the electronics needed to count independently of the processor. A typical count rate available is 0 to 100 kHz, which means the module would be able to count 100,000 pulses per second.

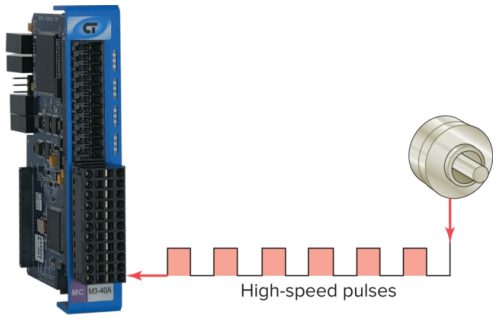


Figure 2-26 High-speed counter module.

Source: Courtesy Control Technology Corporation

ENCODER-COUNTER MODULE

An encoder-counter module allows the user to read the signal from an encoder (Figure 2-27) on a real-time basis and stores this information so it can be read later by the processor. An encoder is mainly used for feedback purpose to know the **position and distance** measurement in process applications. Encoders convert motion to electrical signal pulses. For example, if a rotary encoder is designed to produce 100 pulses per revolution, each pulse would indicate 3.6 degrees of rotation.

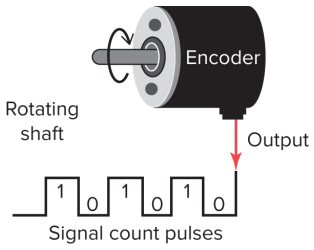


Figure 2-27 Rotary encoder.

STEPPER-MOTOR MODULE

The stepper-motor module provides pulse trains to a stepper-motor translator, which enables control of a stepper motor. The commands for the module are determined by the control program in the PLC.

MOTION AND POSITION CONTROL MODULE

Motion and position control modules are used in applications involving accurate high-speed machining and packaging operations. Intelligent position and motion control modules permit PLCs to control stepper and servo motors. These systems require a drive, which contains the power electronics that translate the signals from the PLC module into signals required by the motor (**Figure 2-28**).

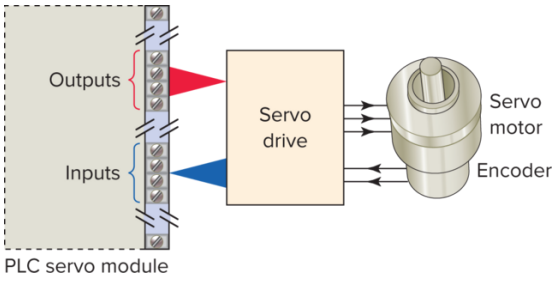


Figure 2-28 PLC servo module.

COMMUNICATION MODULES

Serial communications modules are used to establish point-to-point connections with other intelligent devices for the exchange of data. Such connections are normally established with computers, operator stations, process control systems, and other PLCs. Communication modules allow the user to connect the PLC to high-speed local networks that may be different from the network communication provided with the PLC.

OceanofPDF.com

2.5 I/O Specifications

Manufacturers' specifications provide information about how an interface device is correctly and safely used. These specifications place certain limitations not only on the I/O module but also on the field equipment that it can operate. Some PLC systems support *hot swappable* I/O modules designed to be changed with the power on and the PLC operating. The following is a list of some typical manufacturers' I/O specifications, along with a short description of what is specified.

Typical Discrete (Digital) I/O Module Specifications

NOMINAL INPUT VOLTAGE

This discrete input module voltage value specifies the magnitude (e.g., 5, 24, 230 V) and type (AC or DC) of user-supplied voltage that a module is designed to accept. Input modules are typically designed to operate correctly without damage within a range of plus or minus 10% of the input voltage rating. With DC input modules, the input voltage may also be expressed as an operating range (e.g., 24 to 60 V DC) over which the module will operate.

INPUT THRESHOLD VOLTAGES

This discrete input module specification specifies two values: a minimum ON-state voltage that is the minimum voltage at which logic 1 is recognized as absolutely ON; and a maximum OFF-state voltage which is the voltage at which logic 0 is recognized as absolutely OFF.

NOMINAL CURRENT PER INPUT

This value specifies the minimum input current that the discrete input devices must be capable of driving to operate the input circuit. This input current value, in conjunction with the input voltage, functions as a threshold to protect against detecting noise or leakage currents as valid signals.

AMBIENT TEMPERATURE RATING

This value specifies what the maximum temperature of the air surrounding the I/O modules should be for best operating conditions.

INPUT ON/OFF DELAY

Also known as *response time,* this value specifies the maximum time duration required by an input module's circuitry to recognize that a field device has switched ON (input ON-delay) or switched OFF (input OFF-delay). This delay is a result of filtering circuitry provided to protect against contact bounce and voltage transients. This input delay is typically in the 9 to 25 ms range.

OUTPUT VOLTAGE

This AC or DC value specifies the magnitude (e.g., 5 V, 115 V, 230 V) and type (AC or DC) of user-supplied voltage at which a discrete output module is designed to operate. The output field device that the module interfaces to the PLC must be matched to this specification. Output modules are typically designed to operate within a range of plus or minus 10% of the nominal output voltage rating.

OUTPUT CURRENT

These values specify the maximum current that a single output and the module as a whole can safely carry under load (at rated voltage). This rating is a function of the module's components and heat dissipation characteristics. A device drawing more than the rated output current results in overloading, causing the output fuse to blow. As an example, the specification may give each output a current limit of 1 A. The overall rating of the module current will normally be less than the total of the individuals. The overall rating might be 6 A because each of the eight devices would not normally draw their 1 A at the same time. Other names for the output current rating are *maximum continuous current* and *maximum load current.*

INRUSH CURRENT

An inrush current is a momentary surge of current that an AC or DC output circuit encounters when energizing inductive, capacitive, or filament loads. This value specifies the maximum inrush current and duration (e.g., 20 A for 0.1 s) for which an output circuit can exceed its maximum continuous current rating.

SHORT CIRCUIT PROTECTION

Short circuit protection is provided for AC and DC output modules by either fuses or some other current-limiting circuitry. This specification will designate whether the particular module's design has individual protection for each circuit or if fuse protection is provided for groups (e.g., 4 or 8) of outputs.

LEAKAGE CURRENT

This value specifies the amount of current still conducting through an output circuit even after the output has been turned off. Leakage current is a characteristic exhibited by solid-state switching devices such as transistors and triacs and is normally 1 to 2 mA. Leakage current is normally not large enough to falsely trigger an output device but must be taken into consideration when switching very low current sensitive devices.

ELECTRICAL ISOLATION

Recall that I/O module circuitry is electrically isolated to protect the low-level internal circuitry of the PLC from high voltages that can be encountered from field device connections. The specification for electrical isolation, typically 1500 or 2500 V AC, rates the module's capacity for sustaining an excessive voltage at its input or output terminals.

POINTS PER MODULE

This specification defines the number of field inputs or outputs that can be connected to a single module. Most commonly, a discrete module will have 8, 16, or 32 circuits. Modules with 32 or 64 input or output bits are referred to as *high-density* modules. Some modules provide more than one common terminal, which allows the user to use different voltage ranges on the same card as well as to distribute the current more effectively.

BACKPLANE CURRENT DRAW

This value indicates the amount of current the module requires from the backplane. The sum of the backplane current drawn for all modules in a chassis is used to select the appropriate chassis power supply rating.

Typical Analog I/O Module Specifications

CHANNELS PER MODULE

Whereas individual circuits on discrete I/O modules are referred to as points, circuits on analog I/O modules are referred to as channels. These modules normally have 4, 8, or 16 channels. Analog modules may allow for either single-ended or differential connections. *Single-ended* connections use a single ground terminal for all channels or for groups of channels. *Differential* connections use a separate positive and negative terminal for each channel. If the module normally allows 16 single-ended connections, it will generally allow only 8 differential connections. Single-ended connections are more susceptible to electrical noise.

INPUT CURRENT/VOLTAGE RANGE(S)

These are the voltage or current signal ranges that an analog input module is designed to accept. The input ranges must be matched accordingly to the varying current or voltage signals generated by the analog sensors.

OUTPUT CURRENT/VOLTAGE RANGE(S)

This specification defines the current or voltage signal ranges that a particular analog output module is designed to output under program control. The output ranges must be matched according to the varying voltage or current signals that will be required to drive the analog output devices.

INPUT PROTECTION

Analog input circuits are usually protected against accidentally connecting a voltage that exceeds the specified input voltage range.

RESOLUTION

The resolution of an analog I/O module specifies how accurately an analog value can be represented digitally. This specification determines the smallest measurable unit of current or voltage. The higher the resolution (typically specified in bits or mV), the more accurately an analog value can be represented.

INPUT IMPEDANCE AND CAPACITANCE

For analog I/Os, these values must be matched to the external device connected to the module. Typical ratings are in Megohm (MΩ) and picofarads (pF).

COMMON-MODE REJECTION

Noise is generally caused by electromagnetic interference, radio frequency interference, and ground loops. Common-mode noise rejection applies only to differential inputs and refers to an analog module's ability to prevent noise from interfering with data integrity on a single channel and from channel to channel on the module. Noise that is picked up equally in parallel wires is rejected because the difference is zero. Twisted pair wires are used to ensure that this type of noise is equal on both wires. Common-mode rejection is normally expressed in decibels or as a ratio.

OceanofPDF.com

2.6 The Central Processing Unit (CPU)

The central processing unit (CPU) is built into single-unit fixed PLCs while modular rack types typically use a plug-in module. CPU, controller, and processor are all terms used by different manufacturers to denote the same module that performs basically the same functions. Processors vary in processing speed and memory options. A processor module can be divided into two sections: the **CPU section** and the **memory section** (Figure 2-29). The CPU section executes the program and makes the decisions needed by the PLC to operate and communicate with other modules. The memory section electronically stores the PLC program along with other retrievable digital information.

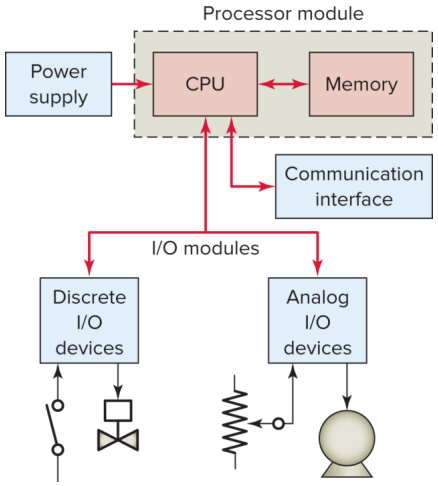


Figure 2-29 Sections of a PLC processor module.

The PLC power supply provides the necessary power (typically 5 VDC) to the processor and I/O modules plugged into the backplane of the rack (Figure 2-30). Power supplies are available for most voltage sources encountered. The power supply converts 24 VDC, 115 VAC, or 230 VAC into the usable DC voltage required by the CPU, memory, and I/O electronic circuitry. PLC power supplies are normally designed to withstand momentary losses of power without affecting the operation of the PLC. *Hold-up time,* which is the length of time a PLC can tolerate a power loss, typically ranges from 10 ms to 3 s.

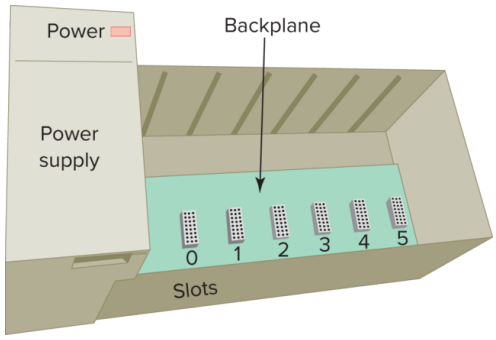


Figure 2-30 PLC power supply.

The CPU contains the similar type of microprocessor found in a personal computer. The difference is that the program used with the microprocessor is designed to facilitate industrial control rather than provide general-purpose computing. The CPU executes the operating system, manages memory, monitors inputs, evaluates the user logic (ladder program), and turns on the appropriate outputs.

Every PLC processor handles some type of communication. The **communications** done by the PLC CPU include:

- Communications to the programmer through a serial, Ethernet, or USB port on the processor module.
- Communications to the input and output (I/O) modules through the chassis backplane.
- Communications to other PLCs and other industrial automation devices through Ethernet and other network types.

The CPU of a PLC system may contain more than one processor. One advantage of using multiprocessing is that the overall operating speed is improved. Each processor has its own memory and programs, which operate simultaneously and independently. In such configurations the scan of each processor is parallel and independent thus reducing the total response time. Fault-tolerant PLC systems support dual processors for critical processes. These systems allow the user to configure the system with ***redundant*** (two) processors, which allows transfer of control to the second processor in the event of a processor fault.

Associated with the processor unit will be a number of status LED indicators to provide system diagnostic information to the operator (**Figure 2-31**). Also, a keyswitch may be provided that allows you to select one of the following three modes of operation: RUN, PROG, and REM.

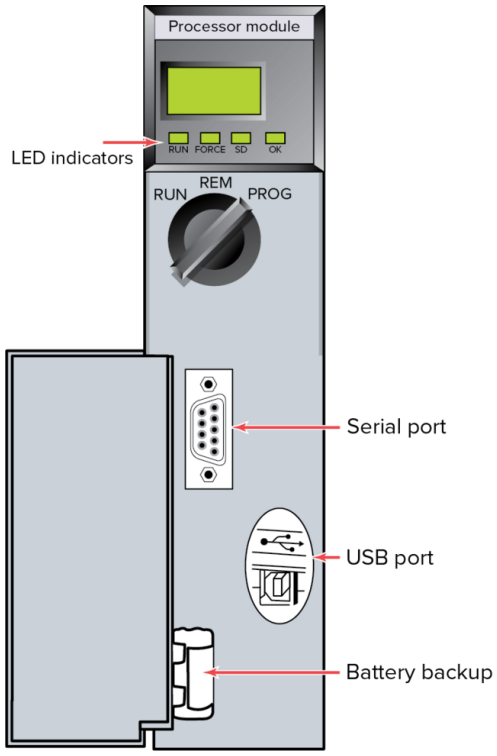


Figure 2-31 Typical processor module.

RUN Position

- Places the processor in the Run mode
- Executes the ladder program and energizes output devices
- Prevents you from performing online program editing in this position
- Prevents you from using a programmer/operator interface device to change the processor mode

PROG Position

- Places the processor in the Program mode
- Prevents the processor from scanning or executing the ladder program, and the controller outputs are de-energized
- Allows you to perform program entry and editing
- Prevents you from using a programmer/operator interface device to change the processor mode

REM Position

- Places the processor in the Remote mode: either the REMote Run, REMote Program, or REMote Test mode
- Allows you to change the processor mode from a programmer/operator interface device
- Allows you to perform online program editing

The processor module also contains circuitry to communicate with the programming device. Somewhere on the module you will find a connector that allows the PLC to be connected to an

external programming device. The decision-making capabilities of PLC processors go far beyond simple logic processing. The processor performs other functions such as timing, counting, latching, comparing, motion control and complex math functions.

PLC processors have changed constantly due to advancements in computer technology and greater demand from applications. Today, processors are faster and have additional instructions added as new models are introduced. Because PLCs are microprocessor based, they can be made to perform tasks that a computer can do. In addition to their control functions, PLCs can be networked to do supervisory control and data acquisition (SCADA).

Many electronic components found in processors and other types of PLC modules are sensitive to *electrostatic* voltages that can degrade their performance or damage them. The following static control procedures should be followed when handling and working with static-sensitive devices and modules:

- Ground yourself by touching a conductive surface before handling static-sensitive components.
- Wear a wrist strap that provides a path to bleed off any charge that may build up during work.
- Be careful not to touch the backplane connector or connector pins of the PLC system (always handle the circuit cards by the edge if possible).
- Be careful not to touch other circuit components in a module when you configure or replace its internal components.
- When not in use, store module in its static-shield bag.
- If available, use a static-safe work station.

OceanofPDF.com

2.7 Memory Design

Memory is the element that stores information, programs, and data in a PLC. The user memory of a PLC includes space for the user program as well as addressable memory locations for storage of data. Data are stored in memory locations by a process called *writing.* Data are retrieved from memory by what is referred to as *reading*.

The complexity of the program determines the amount of memory required. Memory elements store individual pieces of information called *bits* (for *binary digits*). The amount of memory capacity is specified in increments of 1000 or in "K" increments, where 1 K is 1024 bytes of memory storage (a byte is 8 bits).

The program is stored in the memory as 1s and 0s, which are typically assembled in the form of 16-bit or 32 bit words. Memory sizes are commonly expressed in thousands of words that can be stored in the system; thus 2 K is a memory of 2000 words, and 64 K is a memory of 64,000 words. The memory size varies from as small as 1 K for small systems to 32 MB for very large systems. Memory capacity is an important prerequisite for determining whether a particular processor will handle the requirements of the specific application.

Memory *location* refers to an address in the CPU's memory where a binary word can be stored. The RSLogix 5000 memory is built on a 32-bit platform. The controller can process 32 bits of information at once and memory is grouped in 32-bit sections. Allen-Bradley refers to 32 bits of data as a double integer (**DINT**). Each bit in memory has its own unique address that can be used by instructions to read and/or write states. Some program instructions work with single bits of memory and are called **Boolean** instructions. Other instructions work with all 32 bits at on time and are classified as word instructions. **Figure 2-32** shows how tag names are classified in terms of memory reserved. Memory *utilization* refers to the number of memory locations required to store each type of instruction. A rule of thumb for memory locations is one location per coil or contact. One K of memory would then allow a program containing 1000 coils and contacts to be stored in memory.

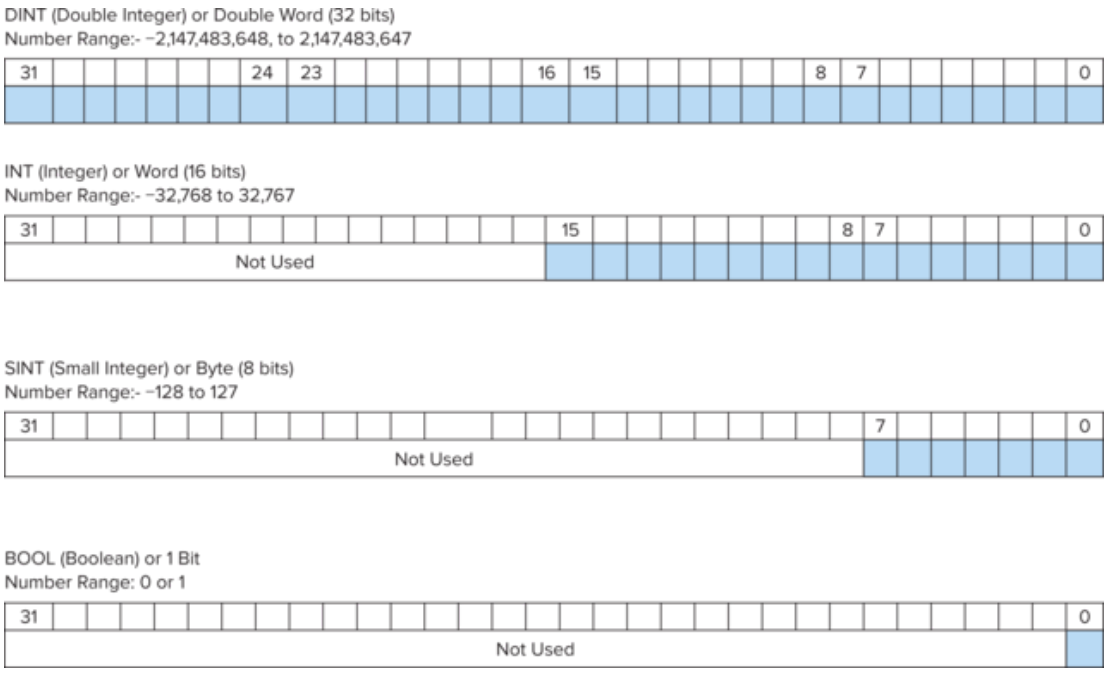


Figure 2-32 32 bit data types.

The memory of a PLC may be broken into sections that have specific functions. Sections of memory used to store the status of inputs and outputs are called input status files or tables and output status files or tables. These terms simply refer to a location where the status of an input or output device is stored. Each bit is either a 1 or 0, depending on whether the input is open or closed. A closed contact would have a binary 1 stored in its respective location in the input table, whereas an open contact would have a 0 stored. A lamp that is ON would have a 1 stored in its respective location in the output table, whereas a lamp that is OFF would have a 0 stored. Input and output image tables are constantly being revised by the CPU. Each time a memory location is examined, the table changes if the contact or coil has changed state.

PLCs execute memory-checking routines to be sure that the PLC memory has not been corrupted. This memory checking is undertaken for safety reasons. It helps ensure that the PLC will not execute if memory is corrupted.

OceanofPDF.com

2.8 Memory Types

Memory can be placed into two general categories: volatile and nonvolatile. Volatile memory will lose its stored information if all operating power is lost or removed. Volatile memory is easily altered and is quite suitable for most applications when supported by battery backup.

Nonvolatile memory has the ability to retain stored information when power is removed accidentally or intentionally. As the name implies, programmable logic controllers have programmable memory that allows users to develop and modify control programs. This memory is made nonvolatile so that if power is lost, the PLC holds its programming.

Read Only Memory (ROM) stores programs, and data cannot be changed after the memory chip has been manufactured. ROM is normally used to store the programs and data that define the capabilities of the PLC. ROM memory is nonvolatile, meaning that its contents will not be lost if power is lost. ROM is used by the PLC for the operating system. The operating system is burned into ROM by the PLC manufacturer and controls the system software that the user uses to program the PLC. When Allen Bradley burns the operating system into memory it is called PROM (programmable read-only memory).

Random Access Memory (RAM), sometimes referred to as *read-write (R/W) memory,* is designed so that information can be written into or read from the memory. RAM is used as a temporary storage area of data that may need to be quickly changed. RAM is volatile, meaning that the data stored in RAM will be lost if power is lost. A battery backup is required to avoid losing data in the event of a power loss (**Figure 2-33**). Most PLCs use CMOS-RAM technology for user memory. CMOS-RAM chips have very low current draw and can maintain memory with a lithium battery for an extended time, two to five years in many cases. Some processors have a capacitor that provides at least 30 minutes of battery backup when the battery is disconnected and power is OFF.

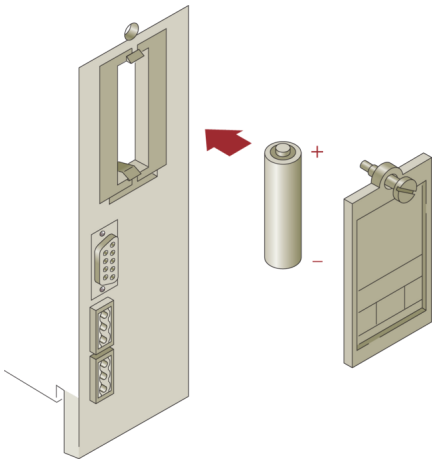


Figure 2-33 Battery used to back up processor RAM.

Erasable Programmable Read-Only Memory (EPROM) provides some level of security against unauthorized or unwanted changes in a program. EPROMs are designed so that data stored in them

can be read, but not easily altered without special equipment. For example, UV EPROMs (ultraviolet erasable programmable read-only memory) can only be erased with an ultraviolet light. EPROM memory is used to back up, store, or transfer PLC programs.

Electrically erasable programmable read-only memory (EEPROM) is a nonvolatile memory that offers the same programming flexibility as does RAM. The EEPROM can be electrically overwritten with new data instead of being erased with ultraviolet light. Because the EEPROM is nonvolatile memory, it does not require battery backup. It provides permanent storage of the program and can be changed easily using standard programming devices. Typically, an EEPROM memory module is used to store, back up, or transfer PLC programs (Figure 2-34).

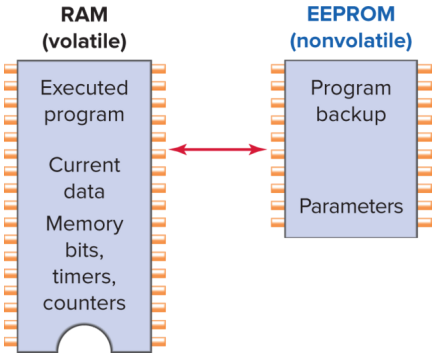


Figure 2-34 EEPROM memory module is used to store, back up, or transfer PLC programs.

Flash EEPROMs are similar to EEPROMs in that they can only be used for backup storage. The main difference comes in the flash memory: they are extremely fast at saving and retrieving files. In addition, they do not need to be physically removed from the processor for reprogramming; this can be done using the circuitry within the processor module in which they reside. Flash memory is also sometimes built into the processor module (Figure 2-35), where it automatically backs up parts of RAM. If power fails while a PLC with flash memory is running, the PLC will resume running without having lost any working data after power is restored.

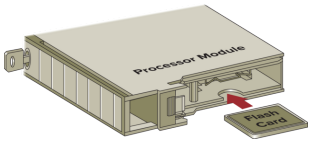


Figure 2-35 Flash memory card installed in a socket on the processor.

OceanofPDF.com

2.9 Programming Terminal Devices

A programming terminal device is needed to enter, modify, and troubleshoot the PLC program. PLC manufacturers use various types of programming devices. The simplest type is the hand-held type programmer shown in Figure 2-36. This proprietary programming device has a connecting cable so that it can be plugged into a PLC's programming port. Hand-held programmers have limited display capabilities. Some units will display only the last instruction that has been programmed, whereas other units will display from two to four rungs of ladder logic.

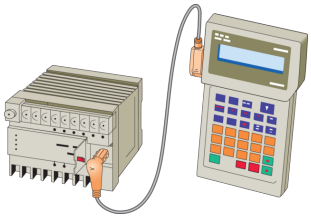


Figure 2-36 Hand-held programming terminal.

The most popular method of PLC programming is to use a personal computer (PC) in conjunction with the manufacturer's programming software (Figure 2-37). Typical capabilities of the programming software include online and offline program editing, online program monitoring, program documentation, diagnosing malfunctions in the PLC, and troubleshooting the controlled system. Hard-copy reports generated in the software can be printed on the computer's printer. Most software packages will not allow you to develop programs on another manufacturer's PLC. In some cases, a single manufacturer will have multiple PLC families, each requiring its own software to program.

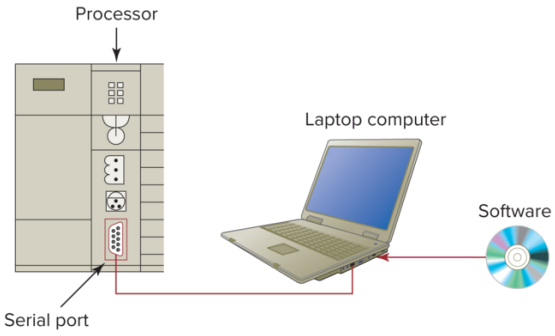


Figure 2-37 Personal computer used as the programming device.

OceanofPDF.com

2.10 Recording and Retrieving Data

Printers are used to provide hard-copy printouts of the processor's memory in ladder program format. Lengthy ladder programs cannot be shown completely on a screen. Typically, a screen shows a maximum of five rungs at a time. A printout can show programs of any length and analyze the complete program.

The PLC can have only one program in memory at a time. To change the program in the PLC, it is necessary either to enter a new program directly from the keyboard or to download one from the computer hard drive. Some CPUs support the use of a memory cartridge that provides portable EEPROM storage for the user program. The cartridge can be used to copy a program from one PLC to another similar type PLC.

OceanofPDF.com

2.11 Human Machine Interfaces (HMIs)

In the past, the typical user interface to a control system consisted of a panel with switches, pushbuttons, pilot lights, gauges, analog meters, and the like. With the advent of digital control systems, larger hard-wired panels have been replaced by a computer screen with process graphics and operator commands entered via a keyboard (Figure 2-38).



Figure 2-38 Human Machine Interface (HMI).
Source: genkur/Shutterstock

Human machine interfaces give the ability to the operator and to management to view the operation in real time. Through personal computer–based setup software, you can configure display screens to:

- Replace hardwired pushbuttons and pilot lights with realistic-looking icons. The machine operator need only touch the display panel to activate the pushbuttons.
- Show operations in graphic format for easier viewing.
- Allow the operator to change timer and counter presets by touching the numeric keypad graphic on the touch screen.
- Show alarms, complete with time of occurrence and location.
- Display variables as they change over time.

Human Machine Interfaces (HMIs) are also referred to as User Interface, Operator Panel, or Terminal and provide a means of *controlling*, *monitoring, managing*, and/or visualizing device processes. They can be located on the machine or in centralized control rooms. The general structure of an HMI package is shown in Figure 2-39. The tag database variables are programmed to interact with the graphic screen objects and communicate with the PLC through the I/O server. If required the screen light can be programmed to change from Green to Red in response whenever the process transitions from Stop to Start.

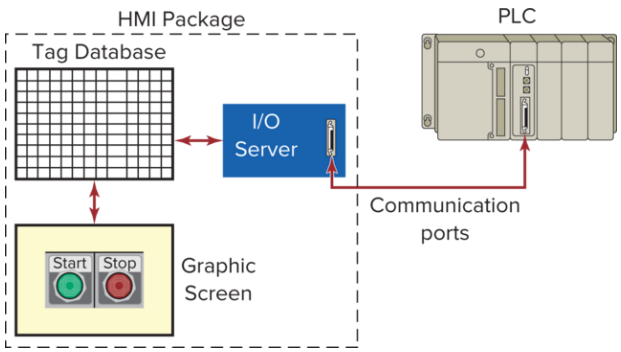


Figure 2-39 General structure of an HMI package.

The design of the HMI application plays a critical role in determining the operator's ability to effectively manage the operation, particularly in response to abnormal situations. The major tasks in the development of an HMI application are:

- *Set up the communication with the PLC*. This involves configuring all necessary software and hardware components.
- *Create the tag database*. Most HMI packages provide a way to import tags from the PLC programming software.
- *Insert the graphical objects on the screen*. Graphics are drawn or imported from a library of common objects.
- *Animate the objects*. There are two basic types of animation: user input and display. User input types allow an operator to change tag values. A display animation allows a value to be displayed and also allows an object to change shape, position, and color.

Many different types of HMI hardware and software features are available. These include:

HMI MONITOR AND ENCLOSURE

HMI operator panels typically contain monochrome or 256 color display screens. These systems often communicate directly with the PLC to read or write memory locations.

- A monochrome monitor uses one color for the background and another to display text or images on the screen.
- Color displays enable clearer process representation and in general brighten up their systems. The color convention for status and alarms should follow the same convention as their hardwired equivalents, namely:
  - *Red*—for alarm, danger, and stop
  - *Yellow*—for caution and risk of danger
  - *Green*—for ready, running, and safe condition
- Screen resolution is expressed as $\text{width} \times \text{height}$, with the units in pixels.
- Screen memory is expressed in Megabytes (MB).

- The environmental certification refers to the type of electrical enclosure used to protect their contents from troublesome operating conditions such as dust, liquids, and extreme variations in temperature.
- The screen is usually touch-sensitive. The touch-sensitive screen allows for more devices and data to be displayed in a smaller area. Detailed information about an object can be accessed by touching the object.

Serial or Ethernet connections are most commonly used when connecting a human machine interface to a PLC. **Serial-based** connection can handle fair amounts of data over distances of up to 100 feet. Ethernet-based connections can handle very large amounts of data more quickly over moderate distances up to 300 feet.

ALARMS

Alarms are messages which indicate that a fault condition is present (**Figure 2-40**). An alarm summary can present a complete list of time-stamped active alarms. Typically an alarm can exist in the following states:

- *Inactive*—The condition being monitored does not have any faults present, and there is no associated alarm message waiting to be acknowledged.
- *Active*—A fault condition is present, and the alarm message has not been acknowledged by the operator.
- *Acknowledged*—The fault condition is present, and the operator has acknowledged the alarm message.
- *OK*—The fault condition is no longer present, but the operator has not acknowledged the alarm message yet.

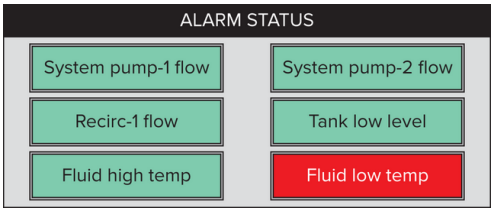


Figure 2-40 Typical alarm status screen.

EVENT HISTORY

An event history presents a time-stamped list of all significant events that have occurred in the process. Many problems within the plant or equipment may occur when no one is monitoring the system, and intermittent problems may be difficult to diagnose without a history of previous issues.

TREND

Values of important process variables, such as flow, temperature, and production rate, over a period of time are shown by this type of display. This type of display provides the ability to chart the progress of the process in real time, providing the same function as a strip chart recorder. For example, suppose

you are monitoring pressure of a Pounds per Square Inch Gauge (psig) as shown in Figure 2-41. According to the table, you can see that it's OK right now, but that's all you know. This trend shows the pressure oscillating around a known good level. We may want to check on the cause of oscillation, but there appears to be no immediate problem.

Current Pressure	Alarm Level	Shutdown Level
235.2 psig	250 psig	300 psig

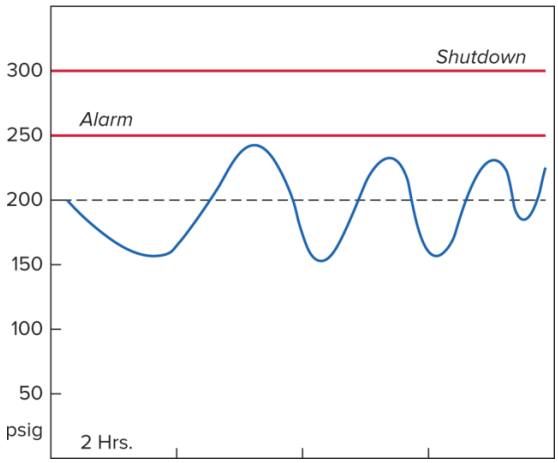


Figure 2-41 Trend monitoring of a pressure gauge.

GRAPHICS LIBRARY

The graphics library contained within an HMI development package provides buttons, lights, switches, sliders, meters, fills, and other graphic objects (Figure 2-42). It saves design time by providing graphics and faceplates for numerous industrial control devices that would otherwise have to be created manually. Librarian applications may include easy-to-use features for resizing, changing color scheme, and orientation of objects, as well as building your own graphics into the library.

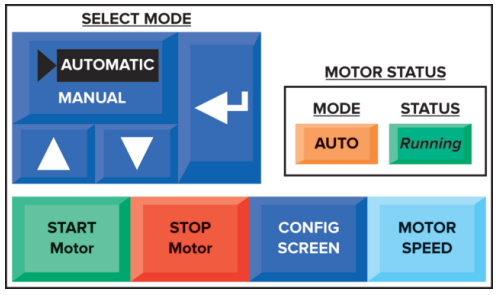


Figure 2-42 Typical motor control graphics.

OceanofPDF.com



CHAPTER 2 REVIEW QUESTIONS

1. What is the function of a PLC input interface module?
2. What is the function of a PLC output interface module?
3. With reference to a PLC rack:
   a. What is a *remote rack?*
   b. Why are remote racks used?
4. How does the processor identify the location of a specific input or output device?
5. List the three basic elements of rack/slot-based addressing.
6. Compare bit level and word level addressing.
7. In what way does tag-based addressing differ from rack/slot-based addressing?
8. What do PC-based control systems use to interface with field devices?
9. What type of I/O modules have both inputs and outputs connected to them?
10. In addition to field devices, what other connections are made to a PLC module?
11. Most PLC modules use plug-in wiring terminal strips. Why?
12. What are the advantage and the disadvantage of using high-density modules?
13. With reference to PLC discrete input modules:
    a. What types of field input devices are suitable for use with them?
    b. List three examples of discrete input devices.
14. With reference to PLC discrete output modules:
    a. What types of field output devices are suitable for use with them?
    b. List three examples of discrete output devices.
15. Explain the function of the backplane of a PLC rack.
16. What is the function of the optical isolator circuit used in discrete I/O module circuits?
17. Name the two distinct sections of an I/O module.
18. List four tasks performed by a discrete input module.
19. What electronic element can be used as the switching device for a 120 VAC discrete output interface module?
20. With reference to discrete output module current ratings:
    a. What is the maximum current rating for a typical 120 VAC output module?
    b. Explain one method of handling outputs with larger current requirements.
21. What electronic element can be used as the switching device for DC discrete output modules?

22. A discrete relay type output module can be used to switch either AC or DC load devices. Why?

23. With reference to sourcing and sinking I/O modules:

- **a.** What current relationship are the terms *sourcing* and *sinking* used to describe?
- **b.** If an I/O module is specified as a current-sinking type, then which type of field device (sinking or sourcing) it is electrically compatible with?

24. Compare discrete and analog I/O modules with respect to the type of input or output devices with which they can be used.

25. Explain the function of the analog-to-digital (A/D) converter circuit used in analog input modules.

26. Explain the function of the digital-to-analog (D/A) converter circuit used in analog output modules.

27. Name the two general sensing classifications for analog input modules.

28. List five common physical quantities measured by a PLC analog input module.

29. What type of cable is used when connecting a thermocouple to a voltage sensing analog input module? Why?

30. Explain the difference between a unipolar and bipolar analog input module.

31. The resolution of an analog input channel is specified as $0.3\ \mathrm{mV}$. What does this tell you?

32. In what two ways can the loop power for current sensing input modules be supplied?

33. List three field devices that are commonly controlled by a PLC analog output module.

34. State one application for each of the following special I/O modules:

- **a.** High-speed counter module
- **b.** Encoder-counter module
- **c.** Stepper-motor module

35. List one application for each of the following intelligent I/O modules:

- **a.** Motion and position control module
- **b.** Communication module

36. Write a short explanation for each of the following discrete I/O module specifications:

- **a.** Nominal input voltage
- **b.** Input threshold voltages
- **c.** Nominal current per input
- **d.** Ambient temperature rating
- **e.** Input ON/OFF delay
- **f.** Output voltage
- **g.** Output current
- **h.** Inrush current
- **i.** Short circuit protection
- **j.** Leakage current
- **k.** Electrical isolation

l. Points per module

m. Backplane current draw

37. Write a short explanation for each of the following analog I/O module specifications:

a. Channels per module

b. Input current/voltage range(s)

c. Output current/voltage range(s)

d. Input protection

e. Resolution

f. Input impedance and capacitance

g. Common-mode rejection

38. Compare the function of the CPU and memory sections of a PLC processor.

39. With reference to the PLC chassis power supply:

a. What conversion of power takes place within the power supply circuit?

b. Explain the term *hold-up time* as it applies to the power supply.

40. Explain the purpose of a redundant PLC processor.

41. Describe three typical modes of operation that can be selected by the keyswitch of a processor.

42. State five other functions, in addition to simple logic processing, that PLC processors are capable of performing.

43. List five important procedures to follow when handling static-sensitive PLC components.

44. Define each of the following terms as they apply to the memory element of a PLC:

a. writing

b. reading

c. bits

d. location

e. utilization

45. With reference to the I/O image tables:

a. What information is stored in PLC input and output tables?

b. What is the input status of a closed switch stored as?

c. What is the input status of an open switch stored as?

d. What is the status of an output that is ON stored as?

e. What is the status of an output that is OFF stored as?

46. Why do PLCs execute memory-checking routines?

47. Compare the memory storage characteristics of volatile and nonvolatile memory elements.

48. What information is normally stored in the ROM memory of a PLC?

49. What information is normally stored in the RAM memory of a PLC?

50. What information is normally stored in an EEPROM memory module?

51. What are the advantages of a processor that utilizes a flash memory card?

52. List three functions of a PLC programming terminal device.

53. Give one advantage and one limitation to the use of hand-held programming devices.

54. What is required for a personal computer to be used as a PLC programming terminal?

55. List four important capabilities of PLC programming software.

56. How many programs can a PLC have stored in memory at any one time?

57. Outline four functions that an HMI interface screen can be configured to perform.

58. List the four major tasks in the development of an HMI application.

59. What information does an HMI trend display convey?

60. Define the term scaling as it applies to PLC inputs and outputs.

61. What is the function of a transducer?

62. In a tag based PLC memory structure, what is the function of a base tag and an alias tag?

OceanofPDF.com



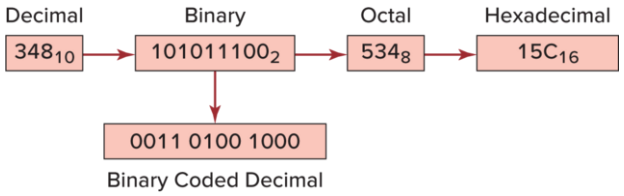
CHAPTER 2 PROBLEMS

1. A discrete 120 VAC output module is to be used to control a 230 VDC solenoid valve. Draw a diagram showing how this could be accomplished using an interposing relay.
2. Assume a thermocouple, which supplies the input to an analog input module, generates a linear voltage of from 20 to 50 mV when the temperature changes from 750 to 1250°F. How much voltage will be generated when the temperature of the thermocouple is at 1000°F?
3. With reference to I/O module specifications:
   a. If the ON-delay time of a given discrete input module is specified as 12 ms, how much is this expressed in seconds?
   b. If the output leakage current of a discrete output module is specified as 950 μA, how much is this expressed in amperes?
   c. If the ambient temperature rating for an I/O module is specified as 60°C, how much is this expressed in degrees Fahrenheit?
4. Assume the triac of an AC discrete output module fails in the shorted state. How would this affect the device connected to this output?
5. A personal computer is to be used to program several different PLCs from different manufacturers. What would be required?
6. Create RSLogix 5000 base and alias tags for each of the following memory locations:
   a. A limit switch connected to terminal 3 of a 16-point input module located in slot 1 of the local chassis.
   b. A solenoid connected to terminal 8 of a 16-point output module located in slot 2 of the local chassis.

OceanofPDF.com

3

Number Systems and Codes



Chapter Objectives

After completing this chapter, you will be able to:

- Define the decimal, binary, octal, and hexadecimal numbering systems and be able to convert from one numbering or coding system to another
- Explain the BCD, Gray, and ASCII code systems
- Define the terms *bit, byte, word, least significant bit (LSB),* and *most significant bit (MSB)* as they apply to binary memory locations
- Add, subtract, multiply, and divide binary numbers

Using PLCs requires us to become familiar with other number systems besides decimal. Some PLC models and individual PLC functions use other numbering systems. This chapter deals with some of these numbering systems, including binary, octal, hexadecimal, BCD, Gray, and ASCII codes. The basics of each system, as well as conversion from one system to another, are explained.

OceanofPDF.com

3.1 Decimal System

Knowledge of different number systems and digital codes is quite useful when working with PLCs or with almost any type of digital computer. This is true because a basic requirement of these devices is to represent, store, and operate on numbers. In general, PLCs work on binary numbers in one form or another; these are used to represent various codes or quantities.

The ***decimal system***, which is most common to us, has a **base of 10.** The radix or base of a number system determines the total number of different symbols or digits used by that system. For instance, in the decimal system, 10 unique numbers or digits—that is, the digits 0 through 9—are used: the total number of symbols is the same as the base, and the symbol with the largest value is 1 less than the base.

The value of a decimal number depends on the digits that make up the number and the place value of each digit. A place (weight) value is assigned to each position that a digit would hold from right to left. In the decimal system the first position, starting from the rightmost position, is 0; the second is 1; the third is 2; and so on up to the last position. The weighted value of each position can be expressed as the base (10 in this case) raised to the power of the position. For the decimal system then, the position weights are 1, 10, 100, 1000, and so on. **Figure 3-1** illustrates how the value of a decimal number can be calculated by multiplying each digit by the weight of its position and summing the results.

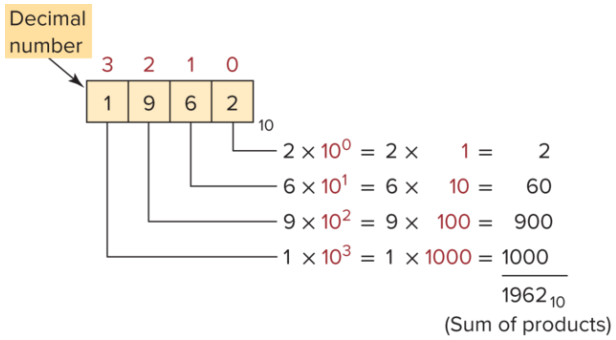


Figure 3-1 Weighted value in the decimal system.

OceanofPDF.com

3.2 Binary System

The ***binary system*** uses the number **2 as the base.** The only allowable digits are 0 and 1. With digital circuits it is easy to distinguish between two voltage levels (i.e., +5 V and 0 V), which can be related to the binary digits 1 and 0 (Figure 3-2). Therefore, the binary system can be applied quite easily to PLCs and computer systems.

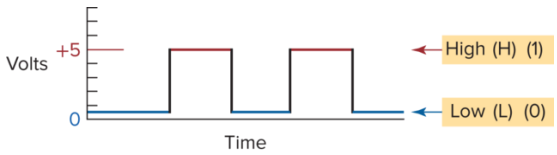


Figure 3-2 Digital signal waveform.

Since the binary system uses only two digits, each position of a binary number can go through only two changes, and then a 1 is carried to the immediate left position. Table 3-1 shows a comparison among four common number systems: decimal (base 10), octal (base 8), hexadecimal (base 16), and binary (base 2). Note that all numbering systems start at *zero.*

Table 3-1 Number System Comparisons

Decimal	Octal	Hexadecimal	Binary
0	0	0	0
1	1	1	1
2	2	2	10
3	3	3	11
4	4	4	100
5	5	5	101
6	6	6	110
7	7	7	111
8	10	8	1000
9	11	9	1001
10	12	A	1010
11	13	B	1011
12	14	C	1100
13	15	D	1101
14	16	E	1110
15	17	F	1111
16	20	10	10000

Decimal	Octal	Hexadecimal	Binary
17	21	11	10001
18	22	12	10010
19	23	13	10011
20	24	14	10100

The decimal equivalent of a binary number can be determined in a manner similar to that used for a decimal number. This time the weighted values of the positions are 1, 2, 4, 8, 16, 32, 64, and so on. The weighted value, instead of being 10 raised to the power of the position, is 2 raised to the power of the position. **Figure 3-3** illustrates how the binary number 10101101 is converted to its decimal equivalent: 173.

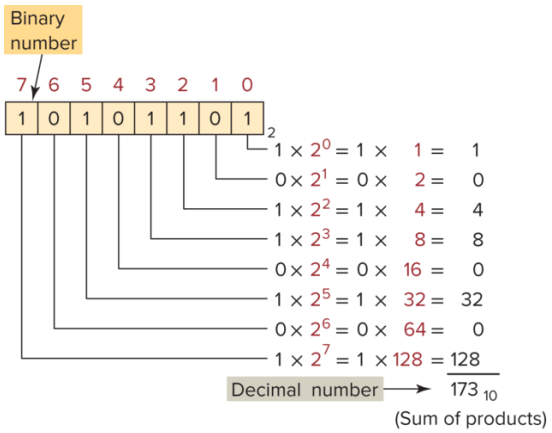


Figure 3-3 Converting a binary number to a decimal number.

Each digit of a binary number is known as a *bit.* In a PLC, the processor-memory element consists of hundreds or thousands of locations. These locations, or registers, are referred to as words. Each *word* is capable of storing data in the form of binary digits, or bits. The number of bits that a word can store depends on the type of PLC system used. Sixteen-bit and 32-bit words are the most common. Bits can also be grouped within a word into bytes. A group of 8 bits is a byte, and a group of 2 or more bytes is a word. **Figure 3-4** illustrates a 16-bit word made up of 2 bytes. The least significant bit (LSB) is the digit that represents the smallest value, and the most significant bit (MSB) is the digit that represents the largest value. A bit within the word can exist only in two states: a logical 1 (or ON) condition, or a logical 0 (or OFF) condition.

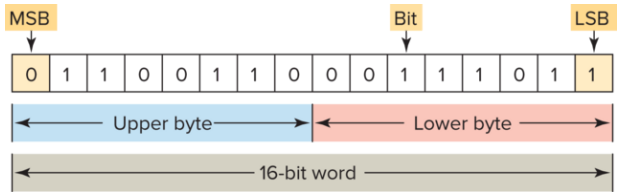


Figure 3-4 A 16-bit word.

PLC memory is organized using bytes, single words, or double words. Older PLCs use 8-bit or 16-bit memory words while newer systems, such as the ControlLogix platform from Allen-Bradley, use 32-bit double words. The size of the programmable controller memory relates to the amount of user program that can be stored. If the memory size is 1 K word (Figure 3-5), it can store 1024 words or $16,384\ (1024 \times 16)$ bits of information using 16-bit words, or $32,768\ (1024 \times 32)$ bits using 32-bit words.

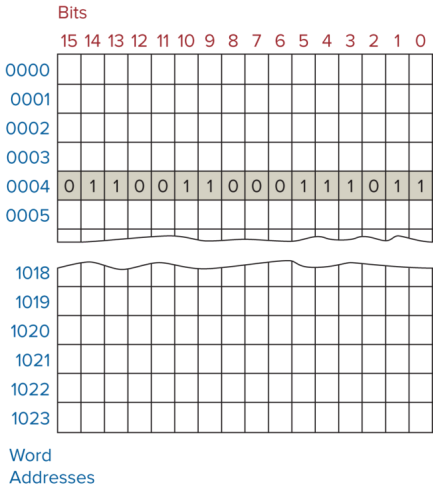


Figure 3-5 1-K word memory.

To convert a decimal number to its binary equivalent, we must perform a series of divisions by 2. Figure 3-6 illustrates the conversion of the decimal number 47 to binary. We start by dividing the decimal number by 2. If there is a remainder, it is placed in the LSB of the binary number. If there is no remainder, a 0 is placed in the LSB. The result of the division is brought down and the process is repeated until the result of successive divisions has been reduced to 0.

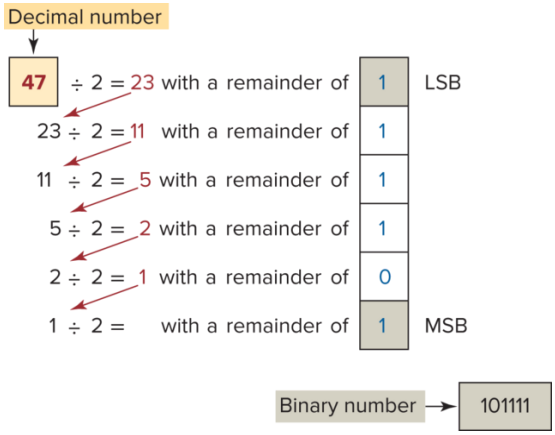


Figure 3-6 Converting a decimal number to a binary number.

Even though the binary system has only two digits, it can be used to represent any quantity that can be represented in the decimal system. All PLCs work internally in the binary system. The processor, being a digital device, understands only 0s and 1s, or binary.

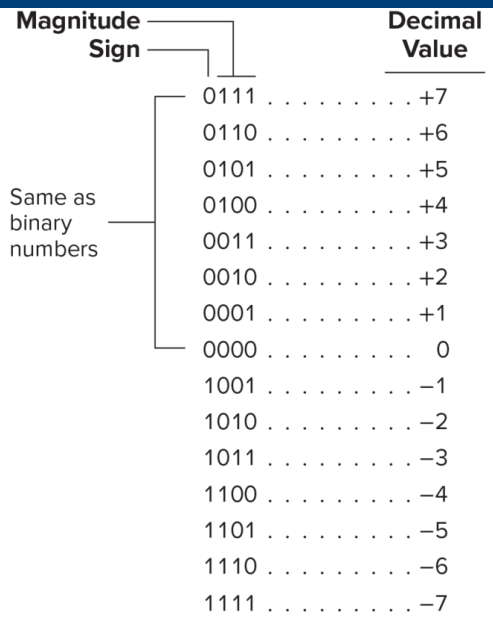
Computer memory is, then, a series of binary 1s and 0s.

OceanofPDF.com

3.3 Negative Numbers

If a decimal number is positive, it has a plus sign; if a number is negative, it has a minus sign. In binary number systems, such as used in a PLC, it is not possible to use positive and negative symbols to represent the polarity of a number. One method of representing a binary number as either a positive or negative value is to use an extra digit, or ***sign bit***, at the MSB side of the number. In the sign bit position, a 0 indicates that the number is positive, and a 1 indicates a negative number (Table 3-2).

Table 3-2 Signed Binary Numbers



Another method of expressing a negative number in a digital system is by using the complement of a binary number. To complement a binary number, change all the 1s to 0s and all the 0s to 1s. This is known as the 1's complement form of a binary number. For example, the 1's complement of 1001 is 0110.

The most common way to express a negative binary number is to show it as a **2's complement** number. The 2's complement is the binary number that results when 1 is added to the 1's complement. This system is shown in Table 3-3. A zero sign bit means a positive number, whereas a 1 sign bit means a negative number.

Table 3-3 1's and 2's Complement Representation of Positive and Negative Numbers

3.4 Octal System

To express the number in the binary system requires many more digits than in the decimal system. Too many binary digits can become cumbersome to read or write. To solve this problem, other related numbering systems are used.

The ***octal numbering system***, a **base 8** system, is used because 8 data bits make up a byte of information that can be addressed. Octal is a convenient means of handling large binary numbers. One octal digit can be used to express three binary digits. As in all other numbering systems, each digit in an octal number has a weighted decimal value according to its position. **Figure 3-7** illustrates how the octal number 462 is converted to its decimal equivalent: 306. Octal converts easily to binary equivalents. For example, the octal number 462 is converted to its binary equivalent by assembling the 3-bit groups, as illustrated in **Figure 3-8**. Notice the simplicity of the notation: the octal 462 is much easier to read and write than its binary equivalent is.

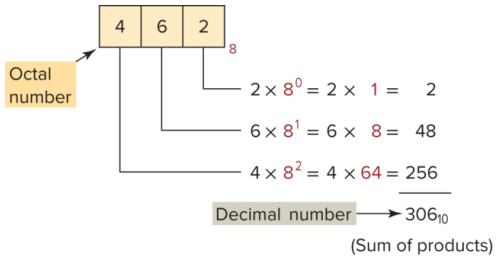


Figure 3-7 Converting an octal number to a decimal number.

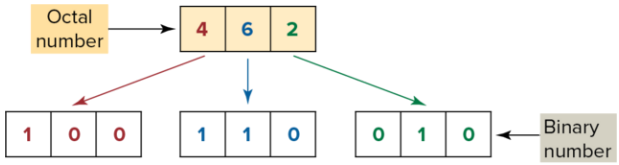


Figure 3-8 Converting an octal number to a binary number.

OceanofPDF.com

3.5 Hexadecimal System

The hexadecimal (hex) system is a **base 16** system, with A to F used to represent decimal numbers 10 to 15 (Table 3-4). The hexadecimal numbering system allows the status of a large number of binary bits to be represented in a small space, such as on a computer screen or PLC programming device display.

Table 3-4 Hexadecimal Numbering System

Hexadecimal	Binary	Decimal
0	0000	0
1	0001	1
2	0010	2
3	0011	3
4	0100	4
5	0101	5
6	0110	6
7	0111	7
8	1000	8
9	1001	9
A	1010	10
B	1011	11
C	1100	12
D	1101	13
E	1110	14
F	1111	15

To convert a hexadecimal number to its decimal equivalent, the hexadecimal digits in the columns are multiplied by the base 16 weight, depending on digit significance. Figure 3-9 illustrates how the conversion would be done for the hex number 1B7.

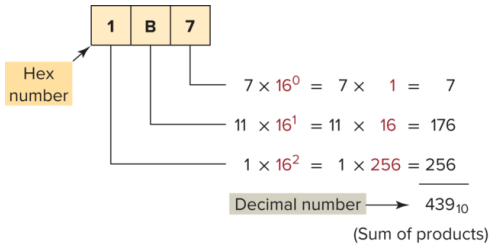


Figure 3-9 Converting a hexadecimal number to a decimal number.

Hexadecimal numbers can easily be converted to binary numbers. Conversion is accomplished by writing the 4-bit binary equivalent of the hex digit for each position, as illustrated in **Figure 3-10**.

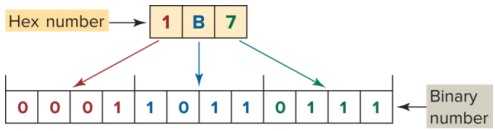


Figure 3-10 Converting a hexadecimal number to a binary number.

OceanofPDF.com

3.6 Binary Coded Decimal (BCD) System

The **binary coded decimal (BCD)** system provides a convenient way of handling large numbers that need to be input to or output from a PLC. As you can see from looking at the various number systems, there is no easy way to go from binary to decimal and back. The BCD system provides a means of converting a code readily handled by humans (decimal) to a code readily handled by the equipment (binary). PLC thumbwheel switches and LED displays are examples of PLC devices that make use of the BCD number system. Table 3-5 shows examples of numeric values in decimal, binary, BCD, and hexadecimal representation.

Table 3-5 Numeric Values in Decimal, Binary, BCD, and Hexadecimal Representation

Decimal	Binary	BCD	Hexadecimal
0	0	0000	0
1	1	0001	1
2	10	0010	2
3	11	0011	3
4	100	0100	4
5	101	0101	5
6	110	0110	6
7	111	0111	7
8	1000	1000	8
9	1001	1001	9
10	1010	0001 0000	A
11	1011	0001 0001	B
12	1100	0001 0010	C
13	1101	0001 0011	D
14	1110	0001 0100	E
15	1111	0001 0101	F
16	1 0000	0001 0110	10
17	1 0001	0001 0111	11
18	1 0010	0001 1000	12
19	1 0011	0001 1001	13
20	1 0100	0010 0000	14
126	111 1110	0001 0010 0110	7E
127	111 1111	0001 0010 0111	7F

Decimal	Binary	BCD	Hexadecimal
128	1000 0000	0001 0010 1000	80
510	1 1111 1110	0101 0001 0000	1FE
511	1 1111 1111	0101 0001 0001	1FF
512	10 0000 0000	0101 0001 0010	200

Page 44

The BCD system uses 4 bits to represent each decimal digit. The 4 bits used are the binary equivalents of the numbers from 0 to 9. In the BCD system, the largest decimal number that can be displayed by any four digits is 9.

The BCD representation of a decimal number is obtained by replacing each decimal digit by its BCD equivalent. To distinguish the BCD numbering system from a binary system, a BCD designation is placed to the right of the units digit. The BCD representation of the decimal number 7863 is shown in Figure 3-11.

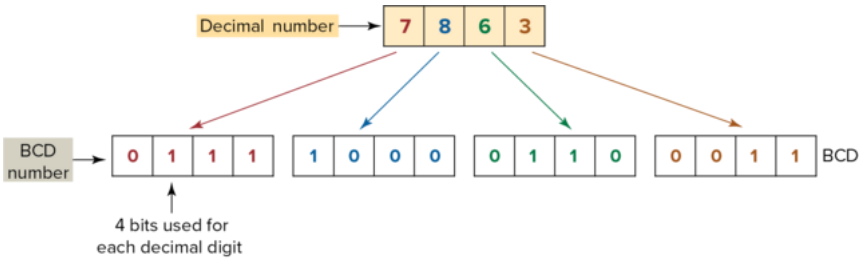


Figure 3-11 The BCD representation of a decimal number.

A thumbwheel switch is one example of an input device that uses BCD. Figure 3-12 shows a single-digit BCD thumbwheel. The circuit board attached to the thumbwheel has one connection for each bit's weight plus a common connection. The operator dials in a decimal digit between 0 and 9, and the thumbwheel switch outputs the equivalent 4 bits of BCD data. In this example, the number eight is dialed to produce the input bit pattern of 1000. A four-digit thumbwheel switch, similar to the one shown, would control a total of 16 (4×4) PLC inputs.

Page 45

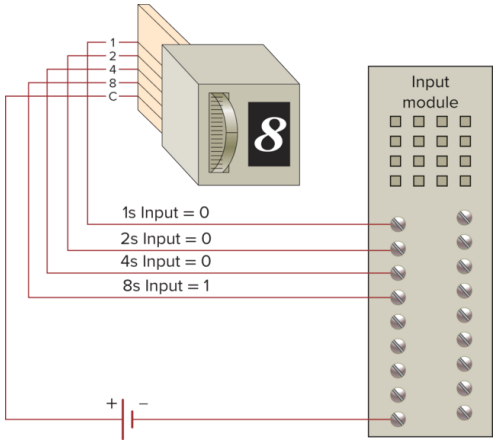


Figure 3-12 BCD thumbwheel switch interfaced to a PLC.

Scientific calculators can be used to convert numbers back and forth between decimal, binary, octal, and hexadecimal. In addition, PLCs contain number conversion functions. BCD-to-binary conversion is required for the input while binary-to-BCD conversion is required for the output. The Convert-to-BCD instruction will convert the binary bit pattern at the source address, N7:23, into a BCD bit pattern of the same decimal value and store it at the destination address, O:2/0. The instruction executes every time it is scanned, and the instruction is true.

Many PLCs allow you to change the format of the data that the data monitor displays. For example, the change radix function found on Allen-Bradley controllers allows you to change the display format of data to binary, octal, decimal, hexadecimal, or ASCII.

OceanofPDF.com

3.7 Gray Code

The **Gray code** is a special type of binary code that does not use position weighting. In other words, each position does not have a definite weight. The Gray code is set up so that as we progress from one number to the next, only one bit changes. This can be quite confusing for counting circuits, but it is ideal for encoder circuits. For example, absolute encoders are position transducers that use the Gray code to determine angular position. The Gray code has the advantage that for each "count" (each transition from one number to the next) only one digit changes. **Table 3-6** shows the Gray code and the binary equivalent for comparison.

Table 3-6 Gray Code and Binary Equivalent

Gray Code	Binary
0000	0000
0001	0001
0011	0010
0010	0011
0110	0100

In binary, as many as four digits could change for a single "count." For example, the transition from binary 0111 to 1000 (decimal 7 to 8) involves a change in all four digits. This kind of change increases the possibility for error in certain digital circuits. For this reason, the Gray code is considered to be an error-minimizing code.

Gray codes are used with position encoders for accurate control of the motion of robots, machine tools, and servomechanisms. **Figure 3-13** shows an optical encoder disk that uses a 4-bit Gray code to detect changes in angular position. In this example, the encoder disk is attached to a rotating shaft and outputs a digital Gray code signal that is used to determine the position of the shaft. A fixed array of photo diodes senses the reflected light from each of the cells across a row of the encoder path. Depending on the amount of light reflected, each cell will output a voltage corresponding to a binary 1 or 0. Thus, a different 4-bit word is generated for each row of the disk.

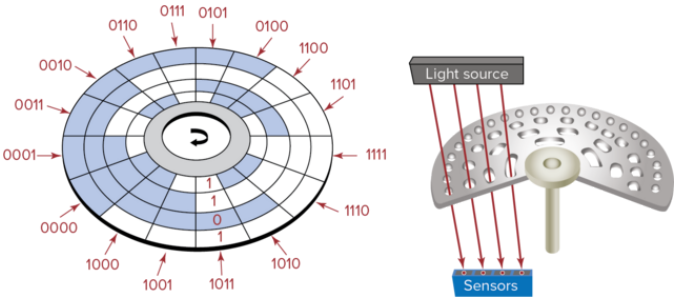


Figure 3-13 Optical encoder disk.

OceanofPDF.com

3.8 ASCII Code

ASCII stands for American Standard Code for Information Interchange. It is an alphanumeric code because it includes letters as well as numbers. The characters accessed by the ASCII code include 10 numeric digits; 26 lowercase and 26 uppercase letters of the alphabet; and about 25 special characters, including those found on a standard typewriter. **Table 3-7** shows a partial listing of the ASCII code. It is used to interface the PLC CPU with alphanumeric keyboards and printers.

Table 3-7 Partial Listing of ASCII Code

Character	7-Bit ASCII	Character	7-Bit ASCII
A	100 0001	X	101 1000
B	100 0010	Y	101 1001
C	100 0011	Z	101 1010
D	100 0100	0	011 0000
E	100 0101	1	011 0001
F	100 0110	2	011 0010
G	100 0111	3	011 0011
H	100 1000	4	011 0100
I	100 1001	5	011 0101
J	100 1010	6	011 0110
K	100 1011	7	011 0111
L	100 1100	8	011 1000
M	100 1101	9	011 1001
N	100 1110	blank	010 0000
O	100 1111	.	010 1110
P	101 0000	,	010 1100
Q	101 0001	+	010 1011
R	101 0010	−	010 1101
S	101 0011	#	010 0011
T	101 0100	(	010 1000
U	101 0101	%	010 0101
V	101 0110	=	011 1101
W	101 0111		

The ASCII code is a seven-bit code in which the decimal digits are represented by the 8-4-2-1 BCD code preceded by 011. Uppercase letters are preceded by 100 or 101. Lowercase letters are preceded by 110 or 111. Character symbols are preceded by 010, 011, 101, and 111. This seven-bit

code provides all possible combinations of characters used when communicating with peripherals or interfaces in a PLC system.

OceanofPDF.com

3.9 Parity Bit

Some PLC communication systems use a binary bit to check the accuracy of data transmission. For example, when data are transferred between PLCs, one of the binary digits may be accidentally changed from a 1 to a 0. This can happen because of a transient or a noise or because of a failure in some portion of the transmission network. A parity bit is used to detect errors that may occur while a word is moved.

Parity is a system in which each message transmitted contains one additional bit. That bit is known as a parity bit. The bit may be a binary 0 or binary 1, depending on the number of 1s and 0s in the character itself. Two systems of parity are normally used: odd and even. Odd parity means that the total number of binary 1 bits in the character, including the parity bit, is odd. Even parity means that the number of binary 1 bits in the character, including the parity bit, is even. Examples of odd and even parity are shown in Table 3-8.

Table 3-8 Odd and Even Parity

Character	Even Parity Bit	Odd Parity Bit
0000	0	1
0001	1	0
0010	1	0
0011	0	1
0100	1	0
0101	0	1
0110	0	1
0111	1	0
1000	1	0
1001	0	1

OceanofPDF.com

3.10 Binary Arithmetic

Arithmetic circuit units form a part of the CPU. Mathematical operations include addition, subtraction, multiplication, and division. Binary addition follows rules similar to decimal addition. When adding with binary numbers, there are only four conditions that can occur:

$$\begin{array}{cccc} 0 & 1 & 0 & 1 \\ \underline{+0} & \underline{+0} & \underline{+1} & \underline{+1} \\ 0 & 1 & 1 & 0 \end{array} \quad \text{carry 1}$$

The first three conditions are easy because they are like adding decimals, but the last condition is slightly different. In decimal, $1 + 1 = 2$. In binary, a 2 is written 10. Therefore, in binary, $1 + 1 = 0$, with a carry of 1 to the next most significant place value. When adding larger binary numbers, the resulting 1s are carried into higher-order columns, as shown in the following examples.

Decimal	Equivalent binary
5	101
+2	+ 10
7	111
	carry 1
10	10 | 10
+ 3	+ | 11
13	11 | 01
	carry carry 1 1
26	1 | 1010
+12	+ | 1100
38	1 0 | 0110



In arithmetic functions, the initial numeric quantities that are to be combined by subtraction are the minuend and subtrahend. The result of the subtraction process is called the difference, represented as:

$$\begin{array}{ll} A & \text{(minuend)} \\ \underline{-B} & \text{(subtrahend)} \\ C & \text{(difference)} \end{array}$$

To subtract from larger binary numbers, subtract column by column, borrowing from the adjacent column when necessary. Remember that when borrowing from the adjacent column, there are now two digits, i.e., 0 borrow 1 gives 10.

EXAMPLE 3-1

Subtract 1001 from 1101.

$$\begin{array}{r} 1101 \\ -1001 \\ \hline 0100 \end{array}$$

Subtract 0111 from 1011.

$$\begin{array}{r} 1011 \\ -0111 \\ \hline 0100 \end{array}$$

Equivalent decimal

$$\begin{array}{r} \mathbf{13} \\ \mathbf{-9} \\ \hline \mathbf{4} \end{array}$$

Equivalent decimal

$$\begin{array}{r} \mathbf{11} \\ \mathbf{-7} \\ \hline \mathbf{4} \end{array}$$

Binary numbers can also be negative. The procedure for this calculation is identical to that of decimal numbers because the smaller value is subtracted from the larger value and a negative sign is placed in front of the result.

EXAMPLE 3-2

Subtract 111 from 100.

$$\begin{array}{r} 111 \\ -100 \\ \hline -011 \end{array}$$

Subtract 11011 from 10111.

$$\begin{array}{r} 11011 \\ -10111 \\ \hline -00100 \end{array}$$

Equivalent decimal

$$\mathbf{4 - 7 = -3}$$

Equivalent decimal

$$\mathbf{23 - 27 = -4}$$

There are other methods available for doing subtraction:

1's complement

2's complement

The procedure for subtracting numbers using the 1's complement is as follows:

Step 1 Change the subtrahend to 1's complement.

Step 2 Add the two numbers.

Step 3 Remove the last carry and add it to the number (end-around carry).

Decimal	Binary		
10	1010		1010
− 6	−0110	1's complement →	+1001
4	100		10011
		End-around carry →	+1
			100



When there is a carry at the end of the result, the result is positive. When there is no carry, then the result is negative and a minus sign has to be placed in front of it.

EXAMPLE 3-3

Subtract 11011 from 01101.

01101	
$+\not{0}00100$	The 1's complement
10001	There is no carry, so we take the 1's complement and add the minus sign: -01110

Equivalent decimal

$$\mathbf{13 - 27 = -14}$$

For subtraction using the 2's complement, the 2's complement is added instead of subtracting the numbers. In the result, if the carry is a 1, then the result is positive; if the carry is a 0, then the result is negative and requires a minus sign.

EXAMPLE 3-4

Subtract 101 from 111.

111	
$+\not{0}11$	The 2's complement
1010	The first 1 indicates that the result is positive, so it is disregarded: 010

Equivalent decimal

$$7 - 5 = 2$$

EXAMPLE 3-5

Subtract 11011 from 01101.

$$\begin{array}{rl} 01101 & \\ +\not{\varnothing}\,00101 & \text{The 2's complement} \\ \hline 10010 & \text{There is no carry, so the result is negative; therefore a 1 has to be subtracted and the 1's complement taken to give the result:} \\ \text{subtract 1} & 10010 - 1 = 10001 \\ \text{1's complement} & -01110 \\ \hline \end{array}$$

Equivalent decimal

$$13 - 27 = -14$$

Binary numbers are multiplied in the same manner as decimal numbers. When multiplying binary numbers, there are only four conditions that can occur:

Page 49

$$\begin{aligned} 0 \times 0 &= 0 \\ 0 \times 1 &= 0 \\ 1 \times 0 &= 0 \\ 1 \times 1 &= 1 \end{aligned}$$

To multiply numbers with more than one digit, form partial products and add them together, as shown in the following example.

Decimal	Equivalent binary
5	101
$\times\ 6$	$\times\ 110$
30	000
	101
	101
	11110

The process for dividing one binary number by another is the same for both binary and decimal numbers, as shown in the following example.

Decimal	Equivalent binary
$\begin{array}{r} 7 \\ 2\overline{)14} \end{array}$	$\begin{array}{r} 111 \\ 10\overline{)1110} \\ \underline{10} \\ 11 \\ \underline{10} \\ 10 \\ \underline{10} \\ 00 \end{array}$

The basic function of a comparator is to compare the relative magnitude of two quantities. PLC data comparison instructions are used to compare the data stored in two words (or registers). At times, devices may need to be controlled when they are less than, equal to, or greater than other data values or set points used in the application, such as timer and counter values. The basic compare instructions are as follows:

$$A = B \, (A \text{ equals } B)$$
$$A > B \, (A \text{ is greater than } B)$$
$$A < B \, (A \text{ is less than } B)$$

OceanofPDF.com

3.11 Floating Point Arithmetic

Certain PLC-related computations are performed using **floating point** arithmetic. The term *floating point* refers to the fact that the decimal point can float or be placed anywhere relative to the significant digits of the number. The main features of floating-point representation are:

- Floating point can support a much wider range of values. It can represent numbers that are very small or numbers that are very large.
- Floating point provides an easy method of dealing with fractions. Without floating point, a PLC word can only represent an integer or whole number.

An example of a floating point number system is shown in **Figure 3-14**. The representations shown in this example vary based on the position of the decimal point, which floats to the left or right, with a corresponding change in the exponent value.

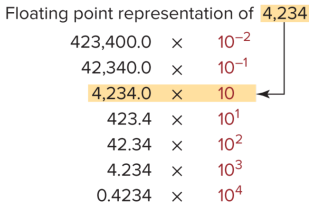


Figure 3-14 Example of a floating point number system.

- The numbers shown on the left side represent the *significand*, or *mantissa*, and the numbers shown on the right indicate the exponent value.
- A significand contains the number's digits and an exponent indicates where the decimal (or binary) point is located relative to the beginning of the significand.
- Decimal floating-point numbers usually take the form of scientific notation with a decimal point always between the 1st and 2nd digits.

Floating point numbers have three basic components: the sign, the exponent, and the mantissa, as shown in **Figure 3-15**.

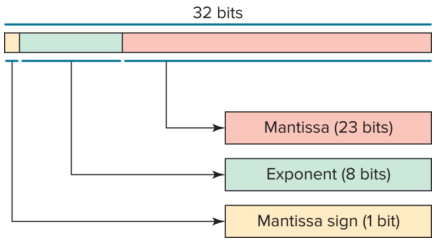


Figure 3-15 Basic components of a floating point number.

- The **sign** of a binary floating-point number is represented by a single bit. A 1 bit indicates a negative number, and a 0 bit indicates a positive number.
- The **mantissa,** always a positive number, holds the significant digits of the floating-point number.
- The **exponent** indicates the positive or negative power of the radix that the mantissa and sign should be multiplied by.

Floating point numbers are also known as *real numbers*. The IEEE 754 Standard is the most commonly used standard for representing these numbers and includes:

- **Single precision:** 32 bits, consisting of
  - Sign bit (1 bit)
  - Exponent (8 bits)
  - Mantissa (32 bits)
- **Double precision:** 64 bits, consisting of
  - Sign bit (1 bit)
  - Exponent (11 bits)
  - Mantissa (52 bits)

Figure 3-16 is an example of how the decimal number 23.5 would be represented in a single precision 32-bit floating point binary format.

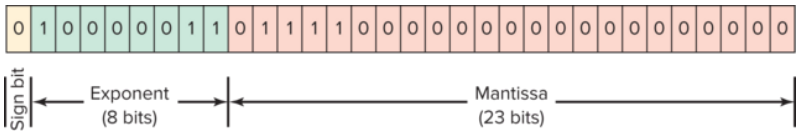


Figure 3-16 Decimal number 23.5 represented in a single precision 32-bit floating point binary format.

OceanofPDF.com



CHAPTER 3 REVIEW QUESTIONS

1. Convert each of the following binary numbers to decimal numbers:

- **a.** 10
- **b.** 100
- **c.** 111
- **d.** 1011
- **e.** 1100
- **f.** 10010
- **g.** 10101
- **h.** 11111
- **i.** 11001101
- **j.** 11100011

2. Convert each of the following decimal numbers to binary numbers:

- **a.** 7
- **b.** 19
- **c.** 28
- **d.** 46
- **e.** 57
- **f.** 86
- **g.** 94
- **h.** 112
- **i.** 148
- **j.** 230

3. Convert each of the following octal numbers to decimal numbers:

- **a.** 36
- **b.** 104
- **c.** 120
- **d.** 216
- **e.** 360
- **f.** 1516

4. Convert each of the following octal numbers to binary numbers:

a. 74

b. 130

c. 250

d. 1510

e. 2551

f. 2634

5. Convert each of the following hexadecimal numbers to decimal numbers:

a. 5A

b. C7

c. 9B5

d. 1A6

6. Convert each of the following hexadecimal numbers to binary numbers:

a. 4C

b. E8

c. 6D2

d. 31B

7. Convert each of the following decimal numbers to BCD:

a. 146

b. 389

c. 1678

d. 2502

8. What is the most important characteristic of the Gray code?

9. What makes the binary system so applicable to computer circuits?

10. Define the following as they apply to the binary memory locations or registers:

a. Bit

b. Byte

c. Word

d. LSB

e. MSB

11. State the base used for each of the following number systems:

a. Octal

b. Decimal

c. Binary

d. Hexadecimal

12. Define the term *sign bit*.

13. Explain the difference between the 1's complement of a number and the 2's complement.

14. What is ASCII code?

15. Why are parity bits used?

16. Add the following binary numbers:

a. $110 + 111$

b. $101 + 011$

c. $1100 + 1011$

17. Subtract the following binary numbers:

a. $1101 - 101$

b. $1001 - 110$

c. $10111 - 10010$

18. Multiply the following binary numbers:

a. 110×110

b. 010×101

c. 101×11

19. Divide the following unsigned binary numbers:

a. $1010 \div 10$

b. $1100 \div 11$

c. $110110 \div 10$

20. What is the subscript notation associated with each of the number systems?

a. Binary

b. Hex

c. Octal

d. Decimal

e. Binary Coded Decimal

OceanofPDF.com



CHAPTER 3 PROBLEMS

1. The following binary PLC coded information is to be programmed using the hexadecimal code. Convert each piece of binary information to the appropriate hexadecimal code for entry into the PLC from the keyboard.

a. 0001 1111

b. 0010 0101

c. 0100 1110

d. 0011 1001

2. Express the decimal number 18 in each of the following number codes:

a. Binary

b. Octal

c. Hexadecimal

d. BCD

3. Add the binary number 110 to the negative binary number -101.

4. Subtract $1011_2 - 110_2$.

5. Multiply the following unsigned binary numbers: 101.1 and 10.11.

6. Divide the following unsigned binary numbers: 1110 by 10.

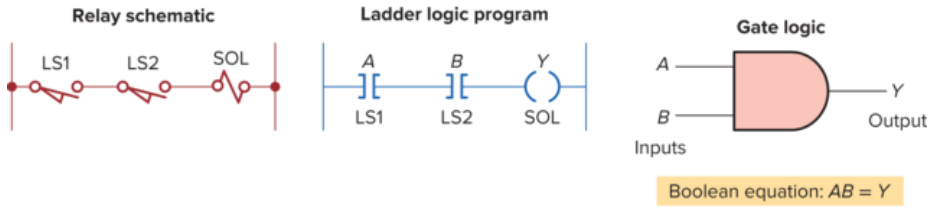
7. State two instances that might call for the use of a floating-point numbering system.

8. What are the three basic components of a floating-point number?

OceanofPDF.com

4

Fundamentals of Logic



Chapter Objectives

After completing this chapter, you will be able to:

- Describe the binary concept and the functions of gates
- Draw the logic symbol, construct a truth table, and state the Boolean equation for the AND, OR, and NOT functions
- Construct circuits from Boolean expressions and derive Boolean equations for given logic circuits
- Convert relay ladder schematics to ladder logic programs
- Develop elementary programs based on logic gate functions
- Program instructions that perform logical operations
- Be familiar with the execution of the AND, OR, NOT, NAND and XOR, PLC logic rungs
- Be familiar with the execution of the word-level Bitwise AND instruction

This chapter gives an overview of digital logic gates and illustrates how to duplicate this type of control on a PLC. Boolean algebra, which is a shorthand way of writing digital gate diagrams, is discussed briefly. Some small hand-held programmers have digital logic keys, such as AND, OR, and NOT, and are programmed using Boolean expressions.

OceanofPDF.com

4.1 The Binary Concept

The PLC, like all digital equipment, operates on the binary principle. The term ***binary principle*** refers to the idea that many things can be thought of as existing in only one of two states. These states are 1 and 0. The 1 and 0 can represent ON or OFF, open or closed, true or false, high or low, or any other two conditions. The key to the speed and accuracy with which binary information can be processed is that there are only two states, each of which is distinctly different. There is no in-between state so when information is processed the outcome is either yes or no.

All digital equipment, regardless of its complexity, is constructed from combinations of logic gates. A ***logic gate*** is a circuit with several inputs but only one output that is activated by particular combinations of input conditions. The two-state binary concept, applied to gates, can be the basis for making decisions. The high beam automobile lighting circuit of **Figure 4-1** is an example of a logical AND decision. For this application, the high beam light can be turned on only when the light switch AND the high beam switch are closed.

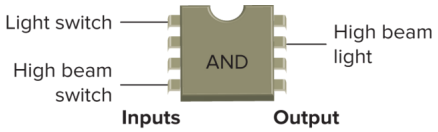


Figure 4-1 The logical AND.

The dome light automobile circuit of **Figure 4-2** is an example of a logical OR decision. For this application, the dome light will be turned on whenever the passenger door switch OR the driver door switch is activated.

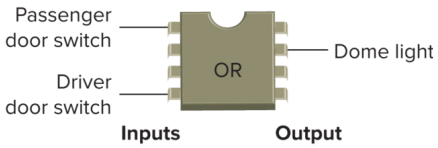


Figure 4-2 The logical OR.

Logic is the ability to make decisions when one or more different factors must be taken into account before an action is taken. This is the basis for the operation of the PLC, where it is required for a device to operate when certain conditions have been met.

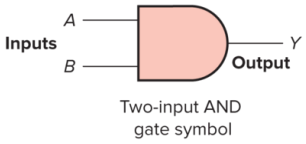
OceanofPDF.com

4.2 AND, OR, and NOT Functions

The operations performed by digital equipment are based on three fundamental logic functions: AND, OR, and NOT. Each function has a rule that will determine the outcome and a *symbol* that represents the operation. For the purpose of this discussion, the outcome or output is called *Y* and the signal inputs are called *A, B, C,* and so on. Also, binary 1 represents the presence of a signal or the occurrence of some event, and binary 0 represents the absence of the signal or nonoccurrence of the event.

The AND Function

The symbol drawn in **Figure 4-3** is that of an AND gate. An **AND gate** is a device with two or more inputs and one output. The AND gate output is 1 only if all inputs are 1. The AND truth table in **Figure 4-3** shows the resulting output from each of the possible input combinations.



Two-input AND gate symbol

AND truth table

Inputs		Output
A	*B*	*Y*
0	0	0
0	1	0
1	0	0
1	1	1

Figure 4-3 AND gate.

Logic gate ***truth tables*** show each possible input to the gate or circuit and the resultant output depending upon the combination of the input(s).

Since logic gates are digital ICs (Integrated Circuits) their input and output signals can be in only one of two possible digital states, that is, logic 0 or logic 1. Thus, the logic state of the output of a logic gate depends on the logic states of each of its individual inputs. **Figure 4-4** illustrates the four possible combinations of inputs for a 2-input AND gate. The basic rules that apply to an AND gate are:

- If all inputs are 1, the output will be 1.
- If any input is 0, the output will be 0.

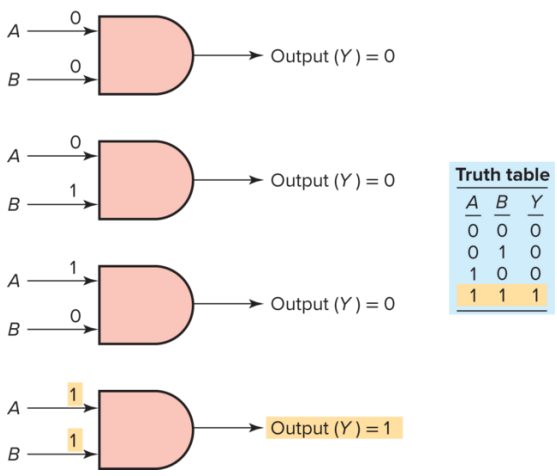


Figure 4-4 AND logic gate digital signal states.

The AND logic gate operates similarly to control devices connected in ***series,*** as illustrated in Figure 4-5. The light will be on only when both switch A and switch B are closed. Also shown is an RSLogix 5000 program rung used to execute the AND logic function.

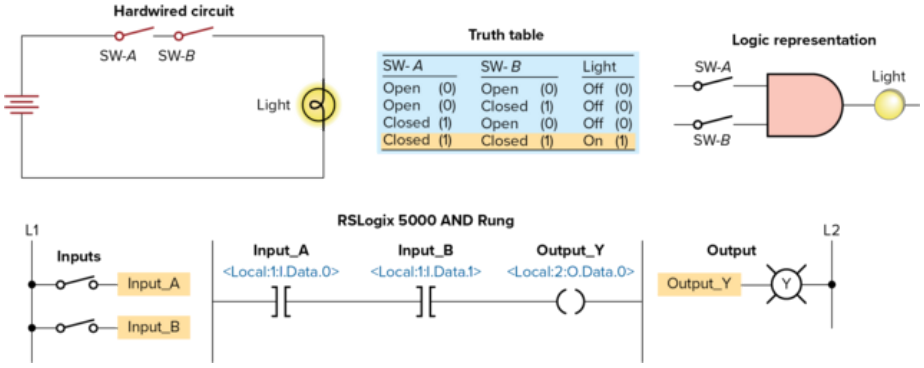


Figure 4-5 AND logic gate operates similarly to control devices connected in series.

The OR Function

The symbol drawn in Figure 4-6 is that of an OR gate. An **OR gate** can have any number of inputs but only one output. The OR gate output is 1 if one or more inputs are 1. The truth table in Figure 4-6 shows the resulting output Y from each possible input combination.

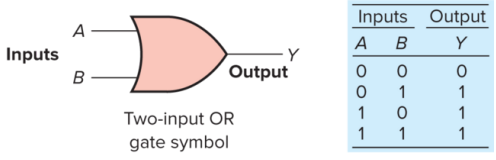


Figure 4-6 OR gate.

Figure 4-7 illustrates the four possible combinations of inputs for a 2-input OR gate. The basic rules that apply to an OR gate are:

- If one or more inputs are 1, the output is 1.
- If all inputs are 0, the output will be 0.

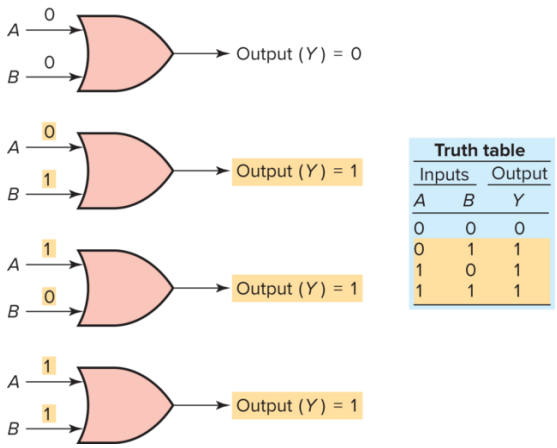


Figure 4-7 OR logic gate digital signal states.

The OR logic gate operates similarly to control devices connected in ***parallel,*** as illustrated in Figure 4-8. The light will be on if switch A or switch B or both are closed. Also shown is an RSLogix 5000 program rung used to execute the OR logic function.

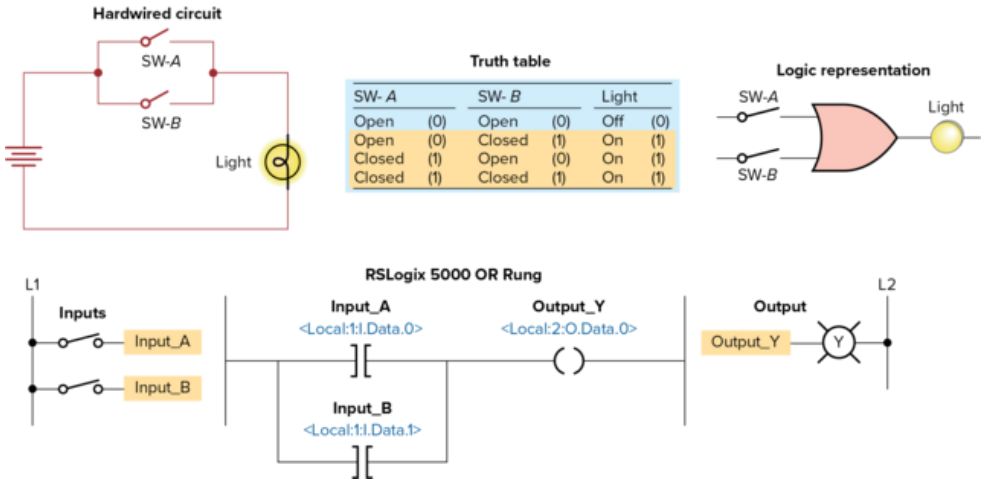


Figure 4-8 OR logic gate operates similarly to control devices connected in parallel.

The NOT Function

The symbol drawn in Figure 4-9 is that of a NOT function. Unlike the AND and OR functions, the **NOT function** can have *only one* input. The NOT output is 1 if the input is 0. The output is 0 if the input is 1. The result of the NOT operation is always the inverse of the input, and the NOT function is, therefore, called an ***inverter***. The NOT function is often depicted by using a bar across the top of the letter, indicating an inverted output. The small circle at the output of the inverter is referred to as a bubble and indicates that an inversion of the logical function has taken place.

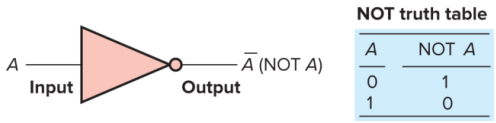


Figure 4-9 NOT function.

The logical NOT function can be performed on a contact input simply by using a normally closed instead of a normally open contact. Figure 4-10 shows an example of the NOT function constructed using a switch series with a lamp. When the switch is *not* actuated, the output lamp is ON. When the input pushbutton is actuated, the output lamp switches OFF. Also shown is an RSLogix 5000 program rung used to execute the NOT logic function.

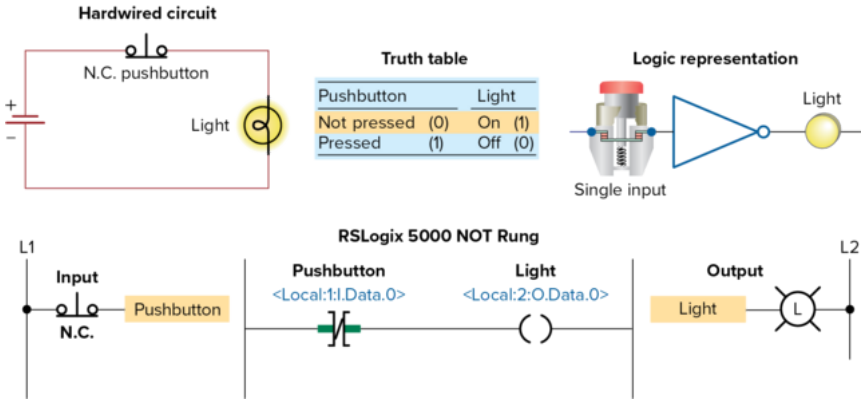


Figure 4-10 NOT function constructed using a switch.

The NOT function is most often used in conjunction with the AND or the OR gate. Figure 4-11 shows the NOT function connected to one input of an AND gate for a low-pressure indicator circuit. If the power is on (1) and the pressure switch is not closed (0), the warning light will be on (1).

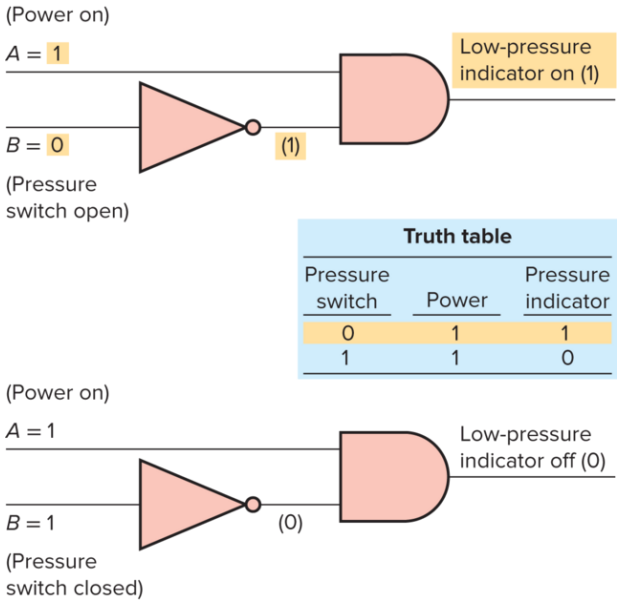


Figure 4-11 NOT function is most often used in conjunction with an AND gate.

The NOT symbol placed at the output of an AND gate would invert the normal output result. An AND gate with an inverted output is called a ***NAND*** gate. The NAND gate symbol and truth table are shown in Figure 4-12. The NAND function is often used in integrated circuit logic arrays and can be used in programmable controllers to solve complex logic. Also shown is an RSLogix 5000 program rung used to execute the NAND logic function.

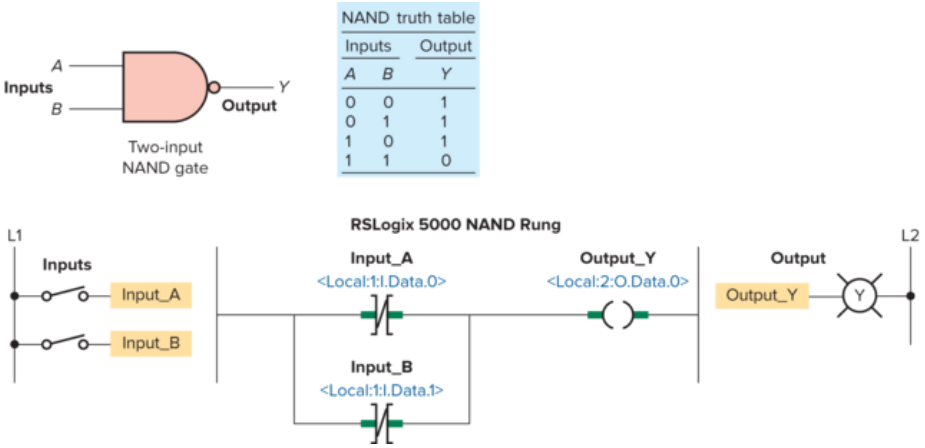
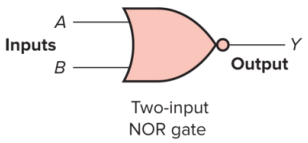


Figure 4-12 NAND gate symbol and truth table.

The same rule about inverting the normal output result applies if a NOT symbol is placed at the output of the OR gate. The normal output is inverted, and the function is referred to as a ***NOR*** gate. The NOR gate symbol and truth table are shown in Figure 4-13.

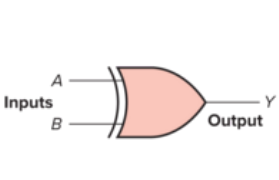


NOR truth table		
Inputs		Output
A	*B*	*Y*
0	0	1
0	1	0
1	0	0
1	1	0

Figure 4-13 NOR gate symbol and truth table.

The Exclusive-OR (XOR) Function

An often-used combination of gates is the exclusive-OR (XOR) function. The ***XOR*** gate symbol and truth table are shown in **Figure 4-14**. The output of this circuit is HIGH only when one input or the other is HIGH, but not both. The exclusive-OR gate is commonly used for the *comparison* of two binary numbers. Also shown is an RSLogix 5000 program rung used to execute the Exclusive-OR (XOR) logic function.



Truth table		
Inputs		Output
A	*B*	*Y*
0	0	0
0	1	1
1	0	1
1	1	0

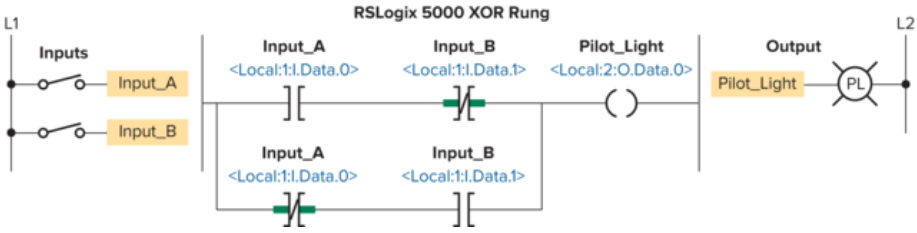


Figure 4-14 The XOR gate symbol and truth table.

OceanofPDF.com

4.3 Boolean Algebra

The mathematical study of the binary number system and logic is called ***Boolean algebra.*** The purpose of this algebra is to provide a simple way of writing complicated combinations of logic statements. There are many applications where Boolean algebra could be applied to solving PLC programming problems.

Figure 4-15 summarizes the basic operators of Boolean algebra as they relate to the basic AND, OR, and NOT functions. Inputs are represented by capital letters A, B, C, and so on, and the output by a capital Y. The dot (•), or no symbol, represents the AND operation, an addition sign (+) represents the OR operation, the circle with an addition sign $\oplus$ represents the exclusive-OR operation, and a bar over the letter ($\bar{A}$) represents the NOT operation. The Boolean equations are used to express the mathematical function of the logic gate.

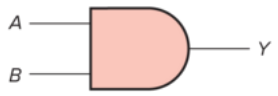
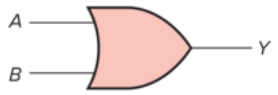
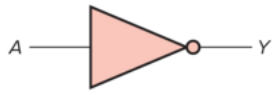
Logic symbol	Logic statement	Boolean equation	Boolean notations
	Y is 1 if A and B are 1	$Y = A \cdot B$ or $Y = AB$	Symbol Meaning • and + or – not ° invert = result in
	Y is 1 if A or B is 1	$Y = A + B$	
	Y is 1 if A is 0 Y is 0 if A is 1	$Y = \bar{A}$	

Figure 4-15 Boolean algebra as related to AND, OR, and NOT functions.

PLC digital systems may be designed using Boolean algebra. Circuit functions are represented by Boolean equations. **Figure 4-16** illustrates how logic operators AND, NAND, OR, NOR, and NOT are used singly to form logical statements. **Figure 4-17** illustrates how basic logic operators are used in combination to form Boolean equations.

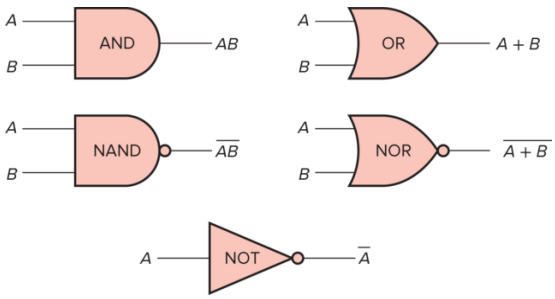


Figure 4-16 Logic operators used singly to form logical statements.

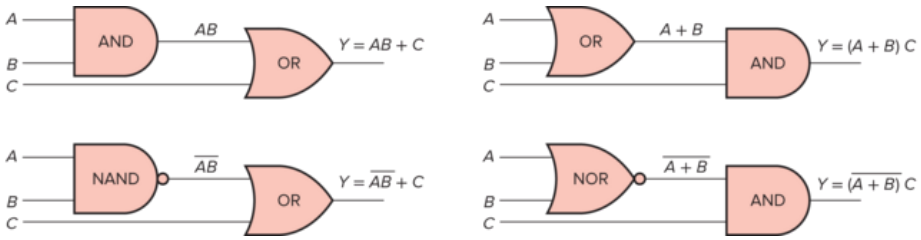


Figure 4-17 Logic operators used in combination to form Boolean equations.

An understanding of the technique of writing simplified Boolean equations for complex logical statements is a useful tool when creating PLC control programs. Some laws of Boolean algebra are different from those of ordinary algebra. These three basic laws illustrate the close comparison between Boolean algebra and ordinary algebra, as well as one major difference between the two:

Commutative Law

$$A + B = B + A$$
$$A \cdot B = B \cdot A$$

Associative Law

$$(A + B) + C = A + (B + C)$$
$$(A \cdot B) \cdot C = A \cdot (B \cdot C)$$

Distributive Law

$$A \cdot (B + C) = (A \cdot B) + (A \cdot C)$$
$$A + (B \cdot C) = (A + B) \cdot (A + C)$$

This law holds true only in Boolean algebra.

OceanofPDF.com

4.4 Developing Logic Gate Circuits from Boolean Expressions

As logic gate circuits become more complex, the need to express these circuits in Boolean form becomes greater. **Figure 4-18** shows a logic gate circuit developed from the Boolean expression $Y = AB + C$. The procedure is as follows:

Boolean expression: $Y = AB + C$

Gates required: (by inspection)

1 - AND gate with input A and B
1 - OR gate with input C and output from previous AND gate

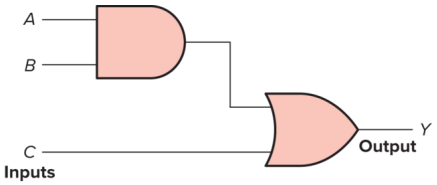


Figure 4-18 Logic gate circuit developed from the Boolean expression $Y = AB + C$.

Figure 4-19 shows a logic gate circuit developed from the Boolean expression $Y = A(BC + D)$. The procedure is as follows:

Boolean expression: $Y = A(BC + D)$

Gates required: (by inspection)

1 - AND gate with inputs B and C
1 - OR gate with inputs BC and D
1 - AND gate with input A and the output from the OR gate

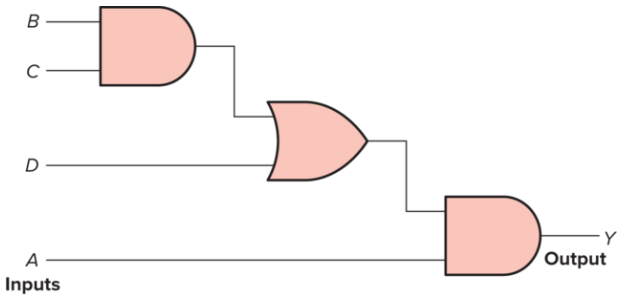


Figure 4-19 Logic gate circuit developed from the Boolean expression $Y = A(BC + D)$.

OceanofPDF.com

4.5 Producing the Boolean Equation for a Given Logic Gate Circuit

A simple logic gate is quite straightforward in its operation. However, by grouping these gates into combinations, it becomes more difficult to determine which combinations of inputs will produce an output. The Boolean equation for the logic circuit of Figure 4-20 is determined as follows:

- The output of the OR gate is $A + B$
- The output of the inverter is D
- Based on the input combination applied to the AND gate the Boolean equation for the circuit is $Y = C\,D\,(A + B)$

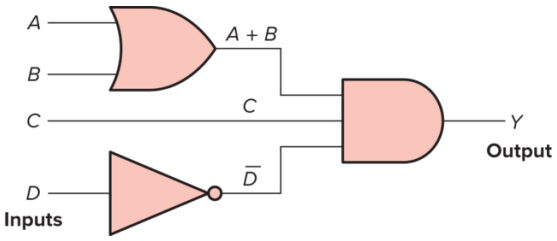


Figure 4-20 Determining the Boolean equation for a logic circuit.

The Boolean equation for the logic circuit of Figure 4-21 is determined as follows:

- The output of AND gate 1 is AB
- The output of AND gate 2 is AB
- Based on the combination of inputs applied to the OR gate the Boolean equation for the circuit is $Y = AB + AB$

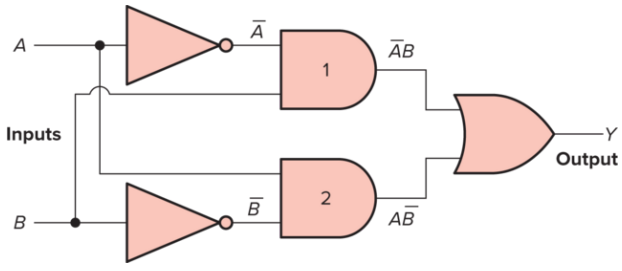


Figure 4-21 Determining the Boolean equation for a logic circuit.

OceanofPDF.com

4.6 Hardwired Logic versus Programmed Logic

The term ***hardwired logic*** refers to logic control functions that are determined by the way devices are electrically interconnected. Hardwired logic can be implemented using relays and relay ladder schematics. Relay ladder schematics are universally used and understood in industry. **Figure 4-22** shows a typical relay ladder schematic of a motor stop/start control station with pilot lights. The control scheme is drawn between two vertical supply lines. All the components are placed between these two lines, called rails or legs, connecting the two power lines with what look like rungs of a ladder—thus the name, relay ladder schematic.

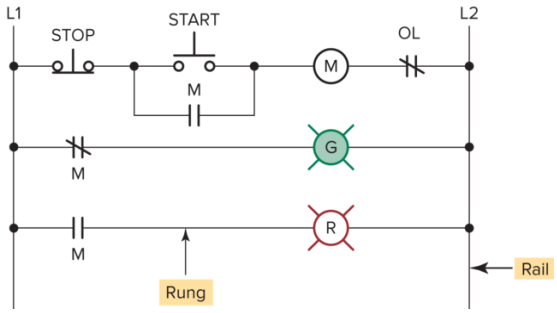


Figure 4-22 Motor stop/start relay ladder schematic.

Hardwired logic is fixed; it is changeable only by altering the way devices are electrically interconnected. In contrast, programmable control is based on the basic logic functions, which are programmable and easily changed. These functions (AND, OR, NOT) are used either singly or in combinations to form instructions that will determine if a device is to be switched on or off. The form in which these instructions are implemented to convey commands to the PLC is called the ***language***. The most common PLC language is *ladder logic*. **Figure 4-23** shows a typical ***ladder logic program*** for the motor start/stop circuit. The instructions used are the relay equivalent of normally open (NO) and normally closed (NC) contacts and coils.

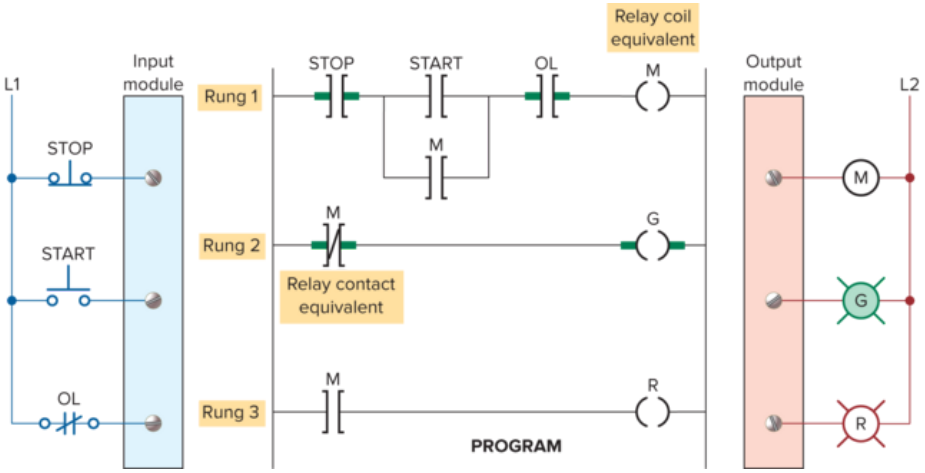
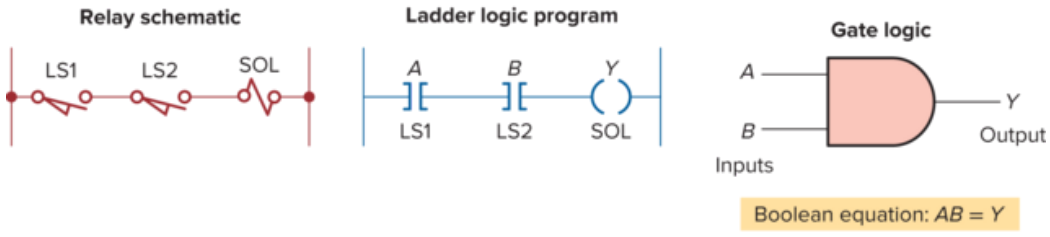


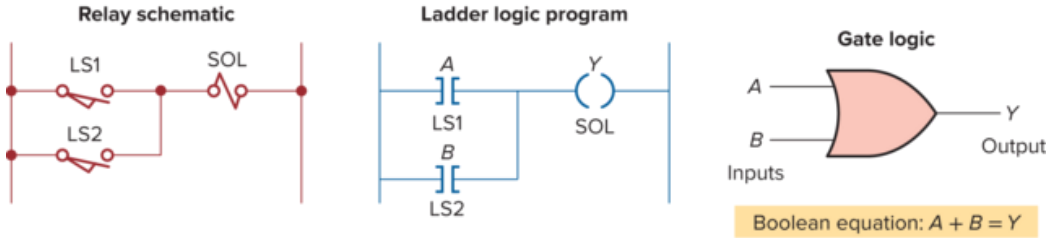
Figure 4-23 Motor stop/start ladder logic program.

PLC contact symbolism is a simple way of expressing the control logic in terms of symbols. These symbols are basically the same as those used for representing hardwired relay control circuits. A **rung** is the contact symbolism required to control an output. Some PLCs allow a rung to have multiple outputs while others allow only one output per rung. A complete ladder logic program then consists of several rungs, each of which controls an output. In programmed logic all mechanical switch contacts are represented by a software contact symbol and all electromagnetic coils are represented by a software coil symbol.

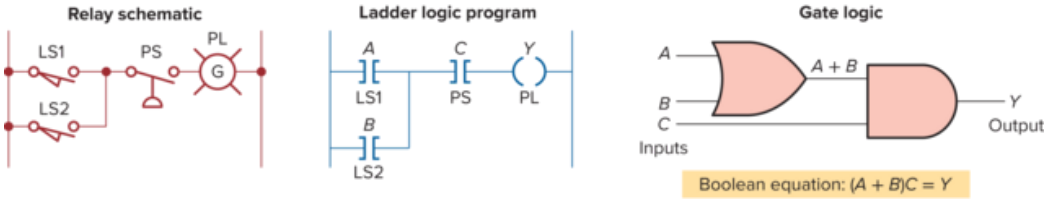
Because the PLC uses ladder logic diagrams, the conversion from any existing relay logic to programmed logic is simplified. Each rung is a combination of **input** conditions (symbols) connected from left to right, with the symbol that represents the **output** at the far right. The symbols that represent the inputs are connected in series, parallel, or some combination of the two to obtain the desired logic. The following group of examples illustrates the relationship between the relay ladder schematic, the ladder logic program, and the equivalent logic gate circuit.



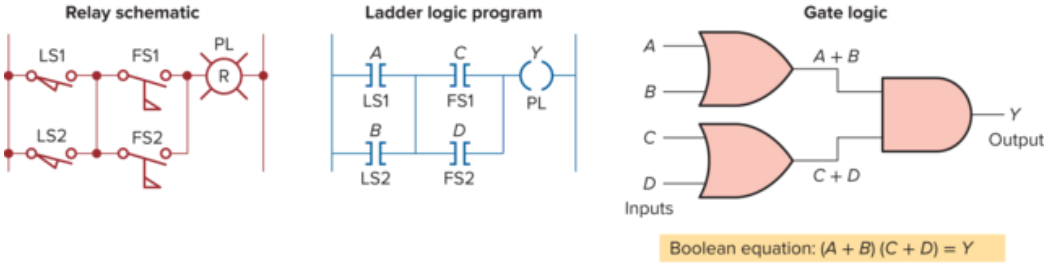
Example 4-1 Two limit switches connected in series and used to control a solenoid valve.



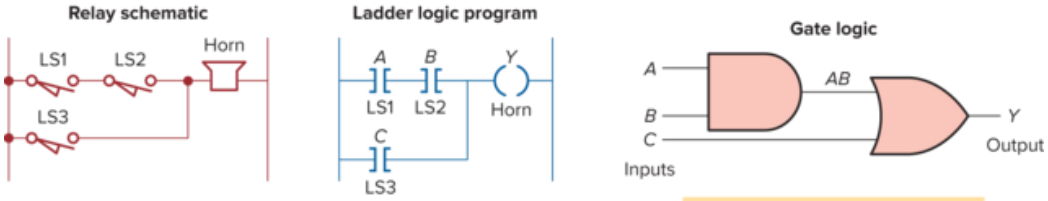
Example 4-2 Two limit switches connected in parallel and used to control a solenoid valve.



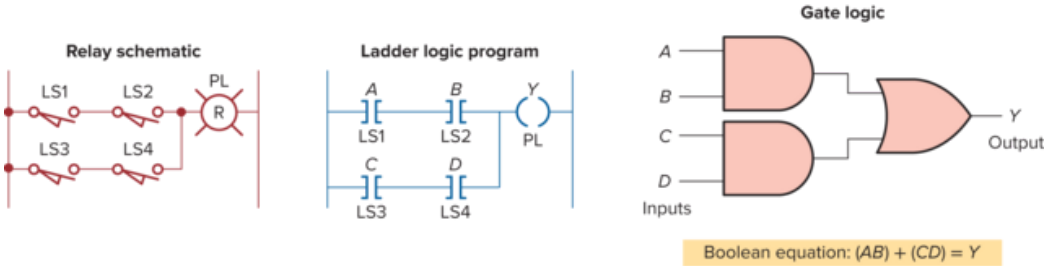
Example 4-3 Two limit switches connected in parallel with each other and in series with a pressure switch.



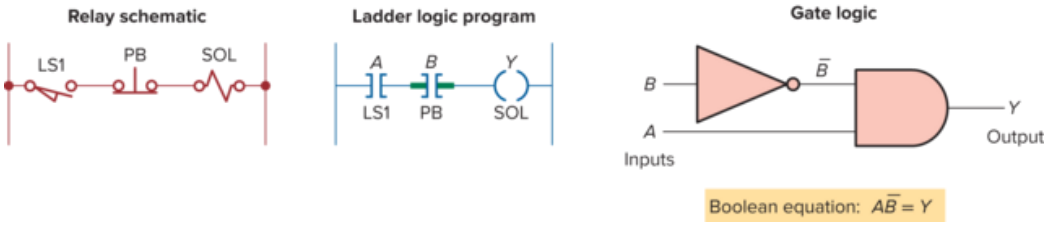
Example 4-4 Two limit switches connected in parallel with each other and in series with two sets of flow switches (that are connected in parallel with each other), and used to control a pilot light.



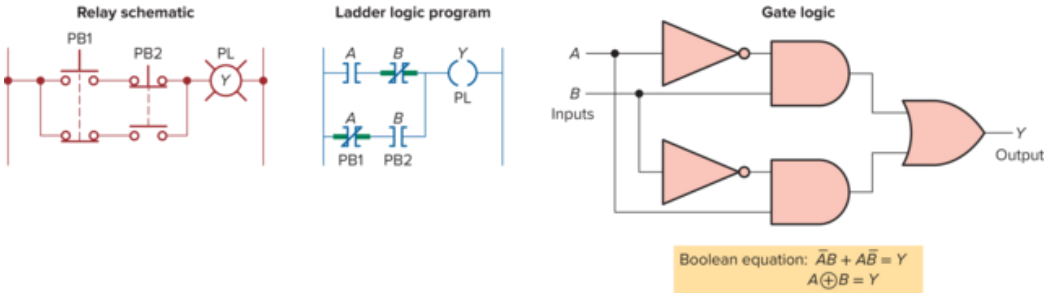
Example 4-5 Two limit switches connected in series with each other and in parallel with a third limit switch, and used to control a warning horn.



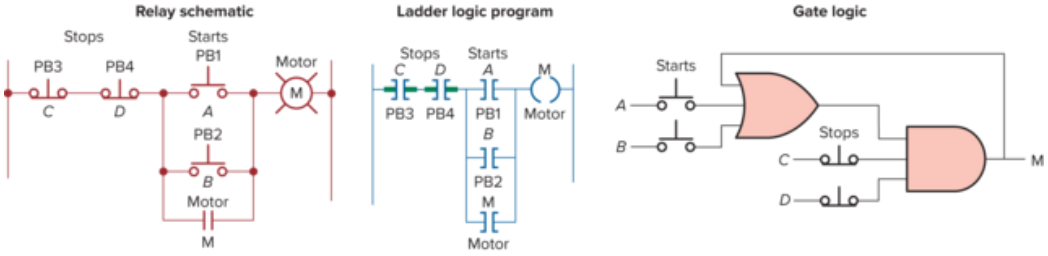
Example 4-6 Two limit switches connected in series with each other and in parallel with two other limit switches (that are connected in series with each other), and used to control a pilot light.



Example 4-7 One limit switch connected in series with a normally closed pushbutton and used to control a solenoid valve. This circuit is programmed so that the output solenoid will be turned on when the limit switch is closed and the pushbutton is *not pushed*.



Example 4-8 Exclusive-OR circuit. The output lamp of this circuit is ON only when pushbutton A or B is pressed, but not both. This circuit has been programmed using only the normally open A and B pushbutton contacts as the inputs to the program.



Example 4-9 A motor control circuit with two start/stop buttons. When either start button is depressed, the motor runs. By use of a seal-in contact, it continues to run when the start button is released. Either stop button stops the motor when it is depressed.

OceanofPDF.com

4.7 Programming Word Level Logic Instructions

PLCs provide **word-level** logic instructions as part of their instruction set. Table 4-1 shows how to select the correct word logic instruction for different situations.

Table 4-1 Selecting Logic Instructions

If you want to . . .	. . . use this instruction.
Know when matching bits in two different words are both ON	AND
Know when one or both matching bits in two different words are ON	OR
Know when one or the other bit of matching bits in two different words is ON	XOR
Reverse the state of bits in a word	NOT

Figure 4-24 illustrates the operation of the AND instruction to perform a word-level AND operation using the bits in the two source addresses. This instruction tells the processor to perform an AND operation on B3:5 and B3:7 and to store the result in destination B3:10 when input device A is true. The destination bits are a result of the logical AND operation.

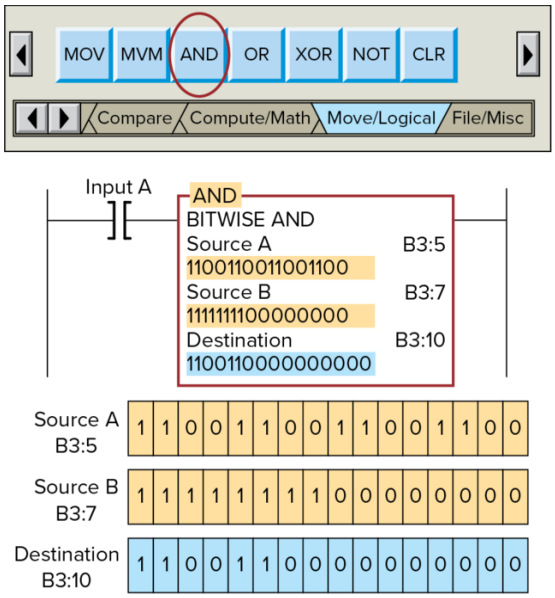


Figure 4-24 Word-level AND instruction.

Figure 4-25 illustrates the operation of a word-level OR instruction, which ORs the data in Source A, bit by bit, with the data in Source B and stores the result at the destination address. The address of Source A is B3:1, the address of Source B is B3:2, and the destination address is B3:20. The

instruction may be programmed conditionally, with input instruction(s) preceding it, or unconditionally, as shown, without any input instructions preceding it.

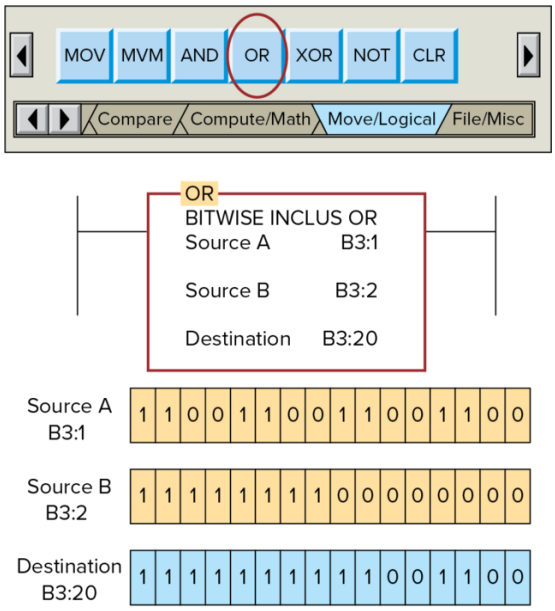


Figure 4-25 Word-level OR instruction.

Figure 4-26 illustrates the operation of a word-level XOR instruction. In this example, data from input I:1.0 are compared, bit by bit, with data from input I:3.0. Any mismatches energize the corresponding bit in word O:4.0. As you can see, there is a 1 in every bit location in the destination corresponding to the bit locations where Source A and Source B are different, and a 0 in the destination where Source A and Source B are the same. The XOR is often used in diagnostics, where real-world inputs, such as rotary cam limit switches, are compared with their desired states.

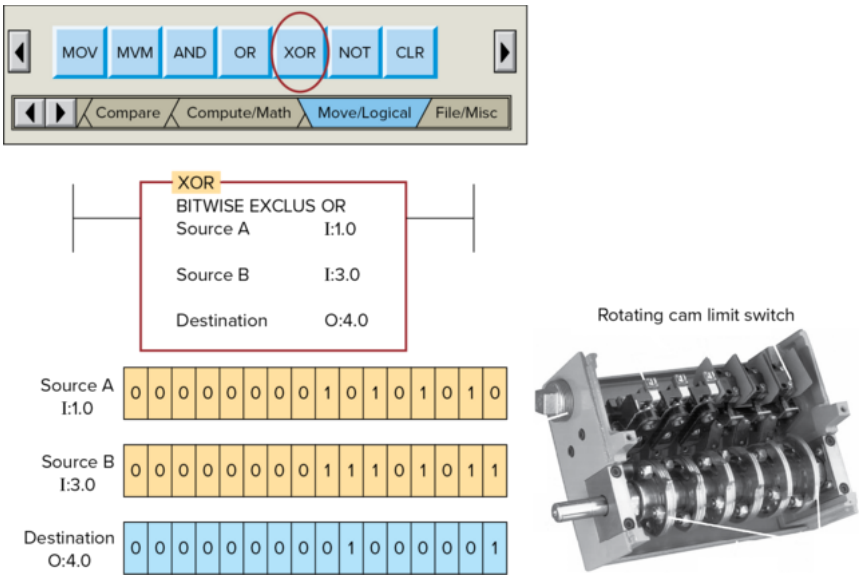


Figure 4-26 Word-level XOR instruction.

Source: Image Courtesy of Rockwell Automation, Inc.

Figure 4-27 shows a programmed RSLogix 5000 word-level Bitwise AND logic rung. The operation is preformed using the ON/OFF (1 or 0) state of two sets of input switches. The execution of the program is summarized as follows:

- Source Input A switches are set ON/OFF according to binary bit pattern 1010_1010.
- Source Input B switches are set ON/OFF according to binary bit pattern 1110_0011.
- The AND instruction performs the logical AND operation between each pair of corresponding bits.
- The resultant binary bit pattern 1010_0010, is set to the Destination address Bitwise_AND_Display and on to output module Local:2:0.Data.
- Note that a 1 is placed in the destination bit location only when both paired bits are 1.
- The ON/OFF light sequence of the output pilot lights reflect the resultant AND operation.

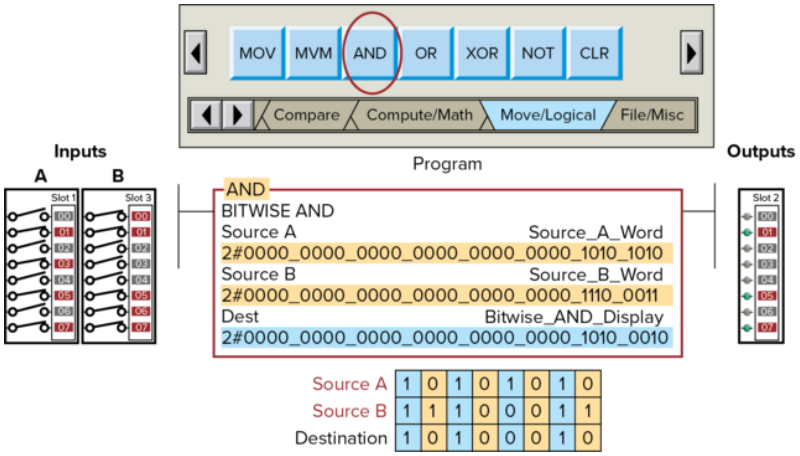


Figure 4-27 RSLogix 5000 word-level Bitwise AND logic rung.

OceanofPDF.com

CHAPTER 4 REVIEW QUESTIONS

1. Explain the binary principle.
2. What is a logic gate?
3. Draw the logic symbol, construct a truth table, and state the Boolean equation for each of the following:
   - **a.** Two-input AND gate
   - **b.** NOT function
   - **c.** Three-input OR gate
   - **d.** XOR function
4. Express each of the following equations as a ladder logic program:
   - **a.** $Y = (A + B)CD$
   - **b.** $Y = A\overline{B}C + \overline{D} + E$
   - **c.** $Y = [(\overline{A} + \overline{B})\,C\,] + DE$
   - **d.** $Y = (\overline{A}B\overline{C}) + (D\overline{E}F)$
5. Write the ladder logic program, draw the logic gate circuit, and state the Boolean equation for the two relay ladder diagrams in **Figure 4-28**.

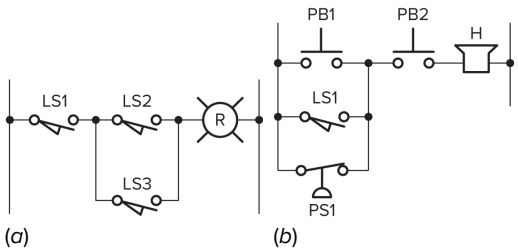


Figure 4-28 Question 5 relay ladder diagrams.

6. Develop a logic gate circuit for each of the following Boolean expressions using AND, OR, and NOT gates:
   - **a.** $Y = ABC + D$
   - **b.** $Y = AB + CD$
   - **c.** $Y = (A + B)(\overline{C} + D)$
   - **d.** $Y = \overline{A}\,(B + CD)$

e. $Y = \overline{A}B + C$

f. $Y = (ABC + D)(E\overline{F})$

7. State the logic instruction you would use when you want to:

a. Know when one or both matching bits in two different words are 1.

b. Reverse the state of bits in a word.

c. Know when matching bits in two different words are both 1.

d. Know when one or the other bit of matching bits, but not both, in two different words is 1.

8. For the logic gate circuit shown in **Figure 4-29**:

a. Determine the Boolean equation.

b. Draw an equivalent ladder logic program for the gate circuit.

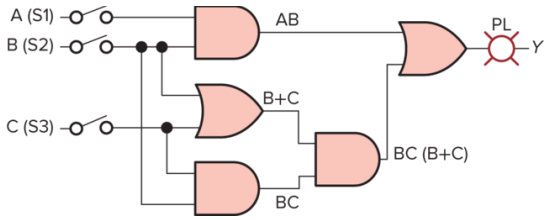


Figure 4-29

9. For the logic gate circuit shown in **Figure 4-30**:

a. Determine the Boolean equation.

b. Draw an equivalent ladder logic program for the gate circuit.

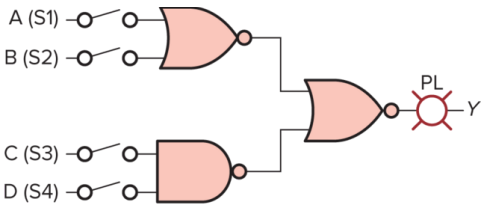


Figure 4-30

OceanofPDF.com



CHAPTER 4 PROBLEMS

1. It is required to have a pilot light come on when all of the following circuit requirements are met:

- All four circuit pressure switches must be closed.
- At least two out of three circuit limit switches must be closed.
- The reset switch must not be closed.

Using AND, OR, and NOT gates, design a logic circuit that will solve this hypothetical problem.

2. Write the Boolean equation for each of the logic gate circuits in **Figure 4-31a–f**.

Page 66

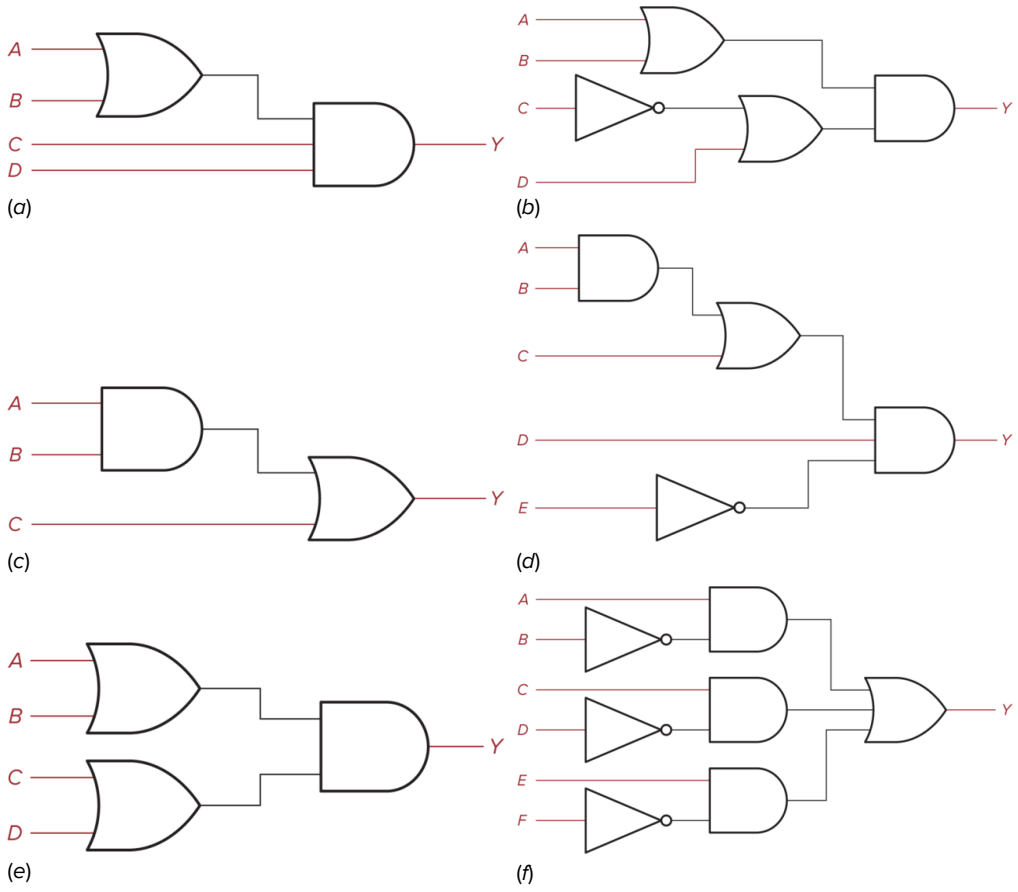


Figure 4-31 Logic gate circuits for Problem 2.

3. The logic circuit of **Figure 4-32** is used to activate an alarm when its output Y is logic HIGH or 1. Draw a truth table for the circuit showing the resulting output for all 16 of the possible input

conditions.

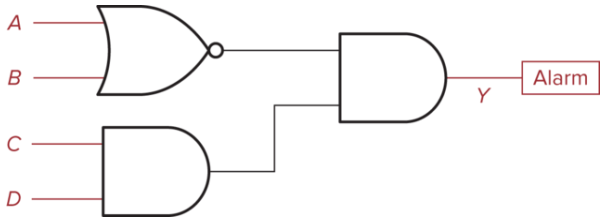


Figure 4-32 Logic circuit for Problem 3.

4. What will be the data stored in the destination address of **Figure 4-33** for each of the following logical operations?

a. AND operation

b. OR operation

c. XOR operation

Source A	0	0	0	0	0	0	0	0	1	0	1	0	1	0	1	0
Source B	0	0	0	0	0	0	0	0	1	1	1	0	1	0	1	1
Destination																

Figure 4-33 Data for Problem 4.

5. Write the Boolean expression and draw the gate logic diagram and typical PLC ladder logic diagram for a control system wherein a fan is to run only when all of the following conditions are met:

- Input A is OFF
- Input B is ON or input C is ON, or both B and C are ON
- Inputs D and E are both ON
- One or more of inputs F, G, or H are ON

OceanofPDF.com

5

Basics of PLC Programming

Symbol	Name	Bit status	Instruction status
	XIC	0	FALSE
		1	TRUE
	XIO	0	TRUE
		1	FALSE



Chapter Objectives

After completing this chapter, you will be able to:

- Define and identify the functions of a PLC memory map
- Describe input and output image table files and types of data files
- Describe the PLC program scan sequence
- Understand how ladder diagram language, Boolean language, and function chart programming language are used to communicate information to the PLC
- Define and identify the function of internal relay instructions
- Identify the common operating modes found in PLCs
- Write and enter ladder logic programs

Each input and output PLC module terminal is identified by a unique address. In PLCs, the internal symbol for any input is a contact. Similarly, in most cases, the internal PLC symbol for all outputs is a coil. This chapter shows how these contact/coil functions are used to program a PLC for circuit operation. This chapter covers only the basic set of instructions that perform functions similar to relay functions. You will also learn more about the program scan cycle and the scan time of a PLC.

OceanofPDF.com

5.1 Processor Memory Organization

While the fundamental concepts of PLC programming are common to all manufacturers, differences in memory organization, I/O addressing, and instruction sets mean that PLC programs are never perfectly interchangeable among different makers. Even within the same product line of a single manufacturer, different models may not be directly compatible.

The **memory** map or structure for a PLC processor consists of several areas, some of these having specific roles. Allen-Bradley PLCs have two different memory structures identified by the terms *rack-based* systems and *tag-based* systems. The RSLogix 500 controller is built on a 16-bit **rack-based** fixed memory structure. The I/O addresses are derived using the slot location of the input and output modules within the PLC rack. In comparison, the RSLogix 5000 controller is built on a 32-bit **tag-based** memory structure for assigning and referencing memory locations. A tag is a friendly name for a memory location. In tag-based memory structures there are no fixed areas of memory allocated for I/O addresses or other types of data. **The memory organization for rack-based systems will be covered in detail in this chapter and that for tag-based systems in Chapter 15.**

Memory organization takes into account the way a PLC divides the available memory into different sections. The memory space can be divided into two broad categories: *program files* and *data files.* Individual sections, their order, and the sections' length will vary and may be fixed or variable, depending on the manufacturer and model.

Program files are the part of the processor memory that stores the user ladder logic program. The program accounts for most of the total memory of a given PLC system. It contains the ladder logic that controls the machine operation. This logic consists of instructions that are programmed in a ladder logic format. Most instructions require one word of memory.

The **data files** store the information needed to carry out the user program. This includes information such as the status of input and output devices, timer and counter values, data storage, and so on. Contents of the data table can be divided into two categories: status data and numbers or codes. Status is ON/OFF type of information represented by 1s and 0s, stored in unique bit locations. Number or code information is represented by groups of bits that are stored in unique byte or word locations.

Figure 5-1 shows the program and data file organization for the RSLogix 500 controller. The contents of each file are as follows.

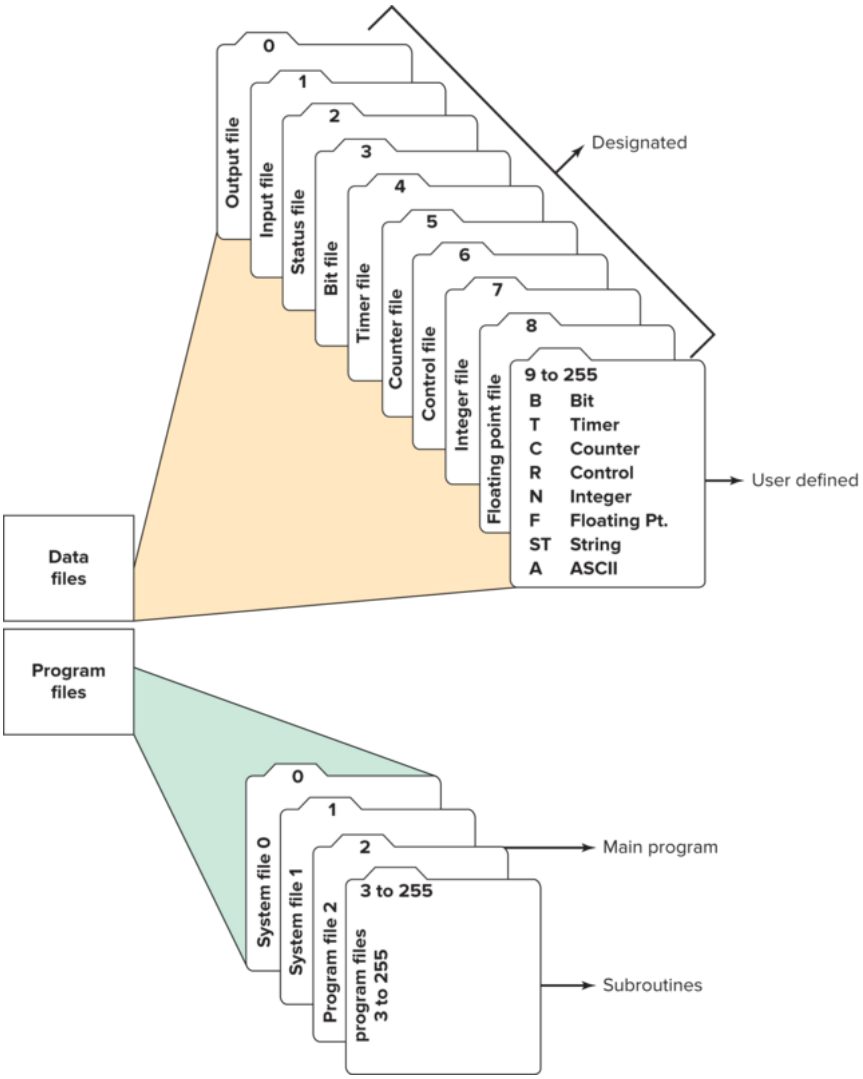


Figure 5-1 Program and data file organization for the RSLogix 500 controller.

Program Files

Program files (Figure 5-2) are the areas of processor memory where ladder logic programming is stored. They may include:

- **System functions (file 0)**—This file is always included and contains various system-related information and user-programmed information such as processor type, I/O configuration, processor file name, and password.
- **Reserved (file 1)**—This file is reserved by the processor and is not accessible to the user.
- **Main ladder program (file 2)**—This file is always included and contains user-programmed instructions that define how the controller is to operate.

- **Subroutine ladder program (files 3–255)—**These files are user-created and are activated according to subroutine instructions residing in the main ladder program file.

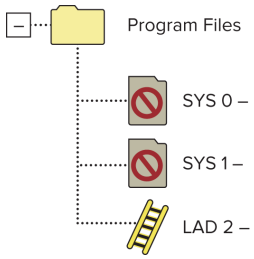


Figure 5-2 Program file tree.

Data Files

The data file portion (Figure 5-3) of the processor's memory stores input and output status, processor status, the status of various bits, and numerical data. All this information is accessed via the ladder logic program. These files are organized by the type of data they contain and may include:

- **Output (file 0)—**This file stores the state of the output terminals for the controller.
- **Input (file 1)—**This file stores the status of the input terminals for the controller.
- **Status (file 2)—**This file stores controller operation information and is useful for troubleshooting controller and program operation.
- **Bit (file 3)—**This file is used for internal relay logic storage.
- **Timer (file 4)—**This file stores the timer accumulated and preset values and status bits.
- **Counter (file 5)—**This file stores the counter accumulated and preset values and status bits.
- **Control (file 6)—**This file stores the length, pointer position, and status bit for specific instructions such as shift registers and sequencers.
- **Integer (file 7)—**This file is used to store whole number values or bit information.
- **Float (file 8)—**The floating point file is used to store fractional numerical data or numerical values greater than 32,767. This file applies to selected PLC processors.

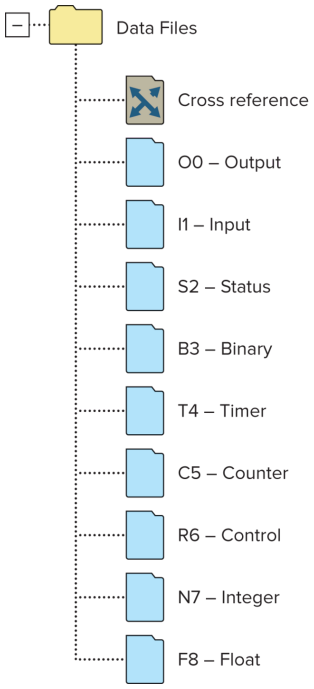


Figure 5-3 Data file tree.

The I/O address format for the SLC family of PLCs is shown in **Figure 5-4**. The format consists of the following three parts:

Part 1: I for input, and a colon to separate the module type from the slot.
O for output and a colon to separate the module type from the slot.

Part 2: The module slot number and a forward slash to separate the slot from the terminal screw.

Part 3: The screw terminal number.

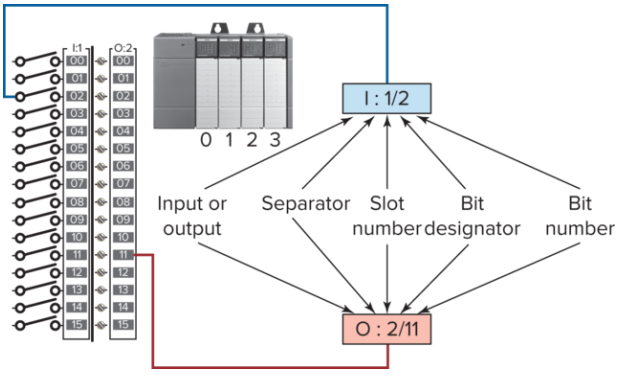


Figure 5-4 I/O address format for the SLC family of PLCs.

Source: Image Courtesy of Rockwell Automation, Inc.

The RSLogix 500 stores data in data tables that are based on 16-bit words. The ***input image table file*** is that part of the program memory allocated to storing the on/off status of connected discrete inputs.

Figure 5-5 shows the connection of an open and closed switch to the input image table file through the input module. Its operation can be summarized as follows.

- For the switch that is closed, the processor detects a voltage at the input terminal and records that information by storing a binary 1 in its bit location.
- For the switch that is open, the processor detects no voltage at the input terminal and records that information by storing a binary 0 in its bit location.
- Each connected input has a bit in the input image table file that corresponds exactly to the terminal to which the input is connected.

Page 70

- The input image table file is changed to reflect the current status of the switch during the I/O scan phase of operation.
- If the input is on (switch closed), its corresponding bit in the table is set to 1.
- If the input is off (switch open), the corresponding bit is cleared, or reset to 0.
- The processor continually reads the current input status and updates the input image table file.

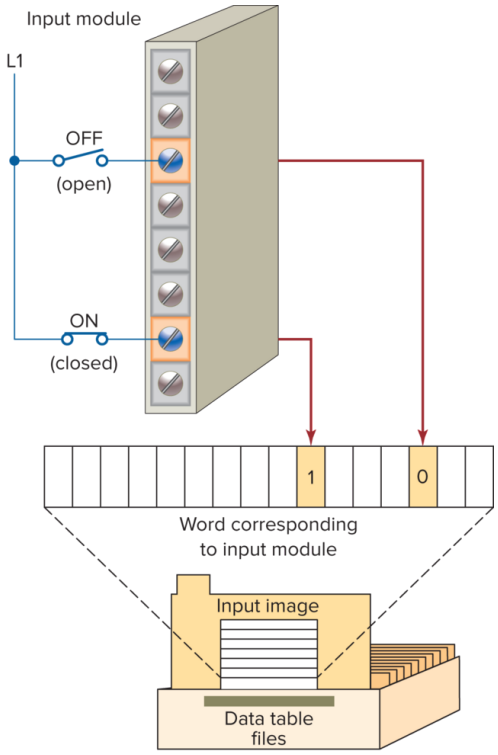


Figure 5-5 Connection of an open and closed switch to the input image table file through the input module.

The ***output image table file*** is that part of the program memory allocated to storing the actual on/off status of connected discrete outputs. **Figure 5-6** shows a typical connection of two pilot lights to the output image table file through the output module. Its operation can be summarized as follows.

- The status of each light (ON/OFF) is controlled by the user program and is indicated by the

Page 71

presence of 1 (ON) and 0 (OFF).

- Each connected output has a bit in the output image table file that corresponds exactly to the terminal to which the output is connected.
- If the program calls for a specific output to be ON, its corresponding bit in the table is set to 1.
- If the program calls for the output to be OFF, its corresponding bit in the table is set to 0.
- The processor continually activates or deactivates the output status according to the output table file status.

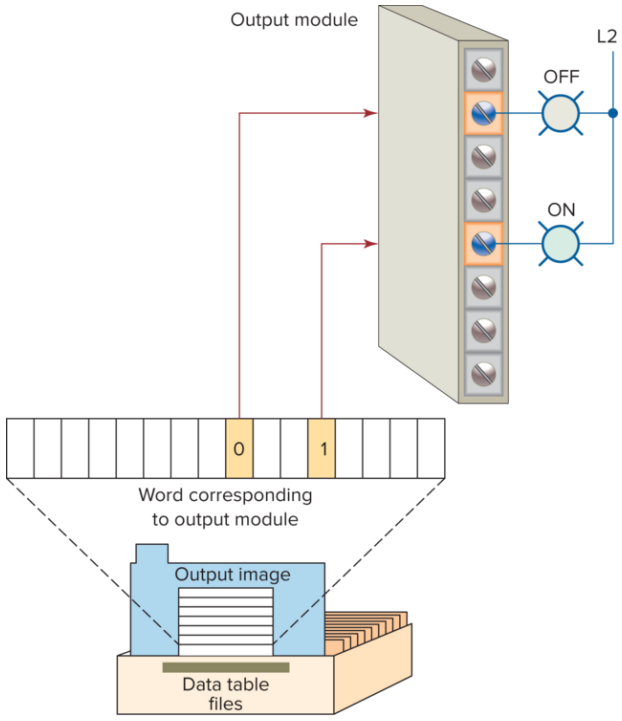


Figure 5-6 Connections of pilot lights to the output image table file through the output module.

Typically, micro PLCs have a fixed number of inputs and outputs. **Figure 5-7** shows the MicroLogix controller from the Allen-Bradley MicroLogix 1000 family of controllers. The controller has 20 discrete inputs with predefined addresses I/0 through I/19 and 12 discrete outputs with predefined addresses O/0 through O/11. Some units also contain analog inputs and outputs embedded into the base unit or available through add-on modules.

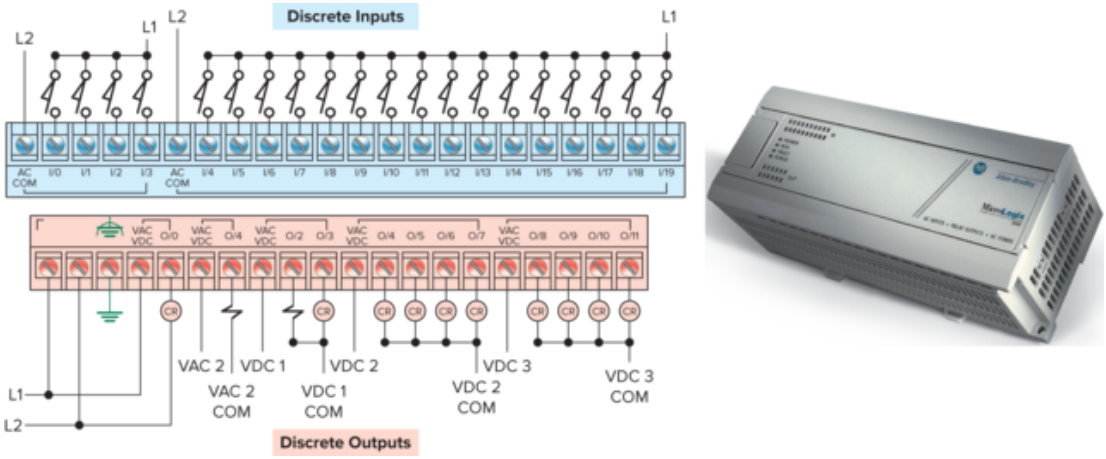


Figure 5-7 Typical micro PLC with predefined addresses.

Source: Image Courtesy of Rockwell Automation, Inc.

OceanofPDF.com

5.2 Program Scan

When a PLC executes a program, it must know—in real time—when external devices controlling a process are changing. During each operating cycle, the processor reads all the inputs, takes these values, and energizes or de-energizes the outputs according to the user program. This process is known as a ***program scan cycle***. **Figure 5-8** illustrates a single PLC operating cycle consisting of the *input scan, program scan, output scan,* and housekeeping duties. Because the inputs can change at any time, it constantly repeats this cycle as long as the PLC is in the RUN mode.

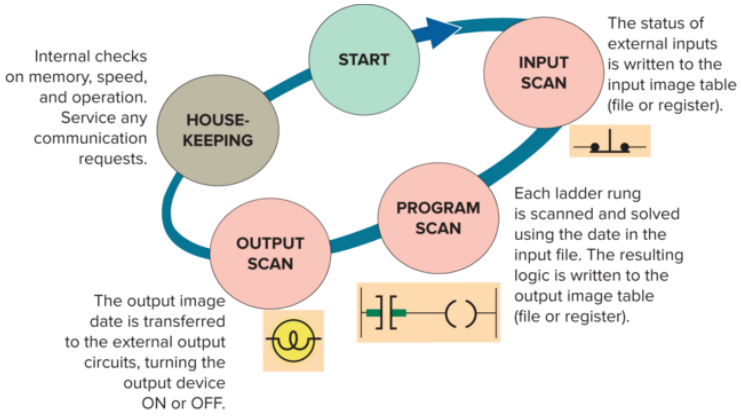


Figure 5-8 PLC program scan cycle.

The time it takes to complete a scan cycle is called the **scan cycle time** and indicates how fast the controller can react to changes in inputs. The time required to make a single scan can vary from about 1 to 20 ms. If a controller has to react to an input signal that changes states twice during the scan time, it is possible that the PLC will never be able to detect this change. For example, if it takes 8 ms for the CPU to scan a program, and an input contact is opening and closing every 4 ms, the program may not respond to the contact changing state. The CPU will detect a change if it occurs during the update of the input image table file, but the CPU will not respond to every change. The scan time is a function of the following:

- The speed of the processor module
- The length of the ladder program
- The type of instructions executed
- The actual ladder true/false conditions

The actual scan time is calculated and stored in the PLC's memory. The PLC computes the scan time each time the END instruction is executed. Scan time data can be monitored via the PLC programming. Typical scan time data include the maximum scan time and the last scan time.

The scan is normally a continuous and sequential process of reading the status of inputs, evaluating the control logic, and updating the outputs. **Figure 5-9** shows an overview of the data flow during the scan process. For each rung executed, the PLC processor will:

- Examine the status of the input image table bits.
- Copy the status of all of the input terminals to the input image table. If an input is active (i.e., there is electrical continuity), the corresponding bit in the input image table will be set to a 1.
- Solve the ladder logic in order to determine logical continuity.
- Update the appropriate output image table bits, if necessary.
- Copy the output image table status to all of the output terminals. Power is applied to the output device if the output image table bit has been previously set to a 1.

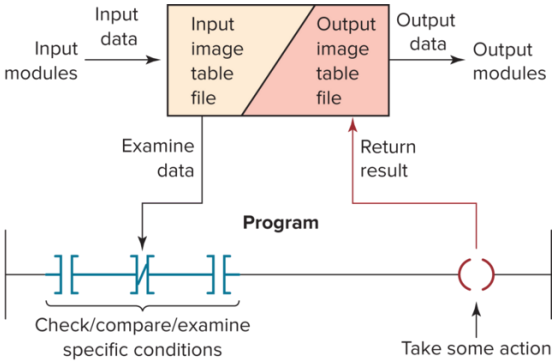
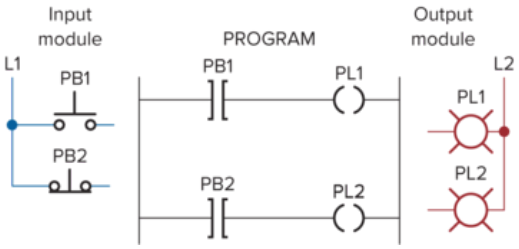


Figure 5-9 Overview of the data flow during the scan process.

In a rung of any hardwired circuit there must be **electrical continuity** in order for the load to energize. The rung has electrical continuity only when the current flow is established in a path from one side of the power rail to the other. There is no electrical continuity in the PLC ladder logic program. Instead, the rung must be evaluated in terms of **logical continuity** rather than electrical continuity. When there is a continuous path of true conditional instructions in a rung, logical continuity exists; accordingly the output instruction is true and the status bit will be set to a 1 (ON).

The controller evaluates logic rung base on the rung conditions preceding the output instruction. **Figure 5-10** and its corresponding truth table show the state of the outputs for all possible input conditions. Its operation is summarized as follows:

- The state of the inputs is determined by input device and is completely independent of the program.
- Conversely, the state of the outputs is solely dependent on the program.
- PB1 is wired with a normally open contact so it will have a 0 in its memory location when it is not pressed and a 1 and logic continuity when it is.
- PB2 is wired with a normally closed contact so it will have a 1 and logic continuity when it is not pressed and a 0 and no logic continuity when it is.
- Any time PB1 is pressed, logic continuity of the rung is established, creating a 1 in the PL1 memory location to switch the light from off to on.
- Any time PB2 is pressed, logic continuity of the rung is lost, resulting in a 0 in the PL2 memory location to switch the light from on to off.



Truth Table

Input Device State	INPUT		OUTPUT	
	PB1	PB2	PL1	PL2
No PBs are pressed	0	1	0	1
PB1 is pressed	1	1	1	1
PB2 is pressed	0	0	0	0
Both PBs are pressed	1	0	1	0

Figure 5-10 Evaluating logic rung conditions.

Ladder programs process inputs at the beginning of a scan and outputs at the end of a scan, as illustrated in Figure 5-11. For each rung scan executed, the PLC processor will:

Step 1 Update the input image table by sensing the voltage of the input terminals. Based on the absence or presence of a voltage, a 0 or a 1 is stored into the memory bit location designated for a particular input terminal.

Step 2 Solve the ladder logic in order to determine logical continuity. The processor scans the ladder program and evaluates the logical continuity of each rung by referring to the input image table to see if the input conditions are met. If the conditions controlling an output are met, the processor immediately writes a 1 in its memory location, indicating that the output will be turned ON; conversely, if the conditions are not met a 0 indicating that the device will be turned OFF is written into its memory location.

Step 3 The final step of the scan process is to update the actual states of the output devices by transferring the output table results to the output module, thereby switching the connected output devices ON (1) or OFF (0). If the status of any input devices changes when the processor is in step 2 or 3, the output condition will not react to them until the next processor scan.

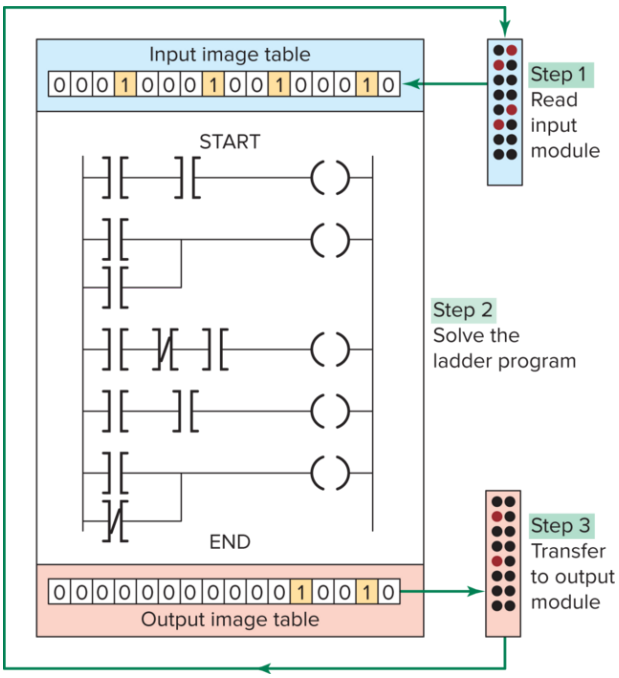


Figure 5-11 Scan process applied to a multiple rung program.

Each instruction entered into a program requires a certain amount of time for the instruction to be executed. The amount of time required depends on the instruction. For example, it takes less time for a processor to read the status of an input contact than it does to read the accumulated value of a timer or counter. The time taken to scan the user program is also dependent on the clock frequency of the microprocessor system. The higher the clock frequency, the faster is the scan rate. Typical processor clock frequencies range from 1 to 10 MHz.

It is important to understand how the PLC scans your program to understand fully how it is going to react. Allen-Bradley PLCs use the ***horizontal*** scan by rung method (**Figure 5-12**). In this system, the processor examines input and output instructions from the first command, top left in the program, horizontally, rung by rung. Some PLCs use the ***vertical*** scan by column method. In this system, the processor examines input and output instructions from the top left command entered in the ladder diagram, vertically, column by column and page by page. Pages are executed in sequence. Both methods are appropriate; however, misunderstanding the way the PLC scans a program can cause programming bugs.

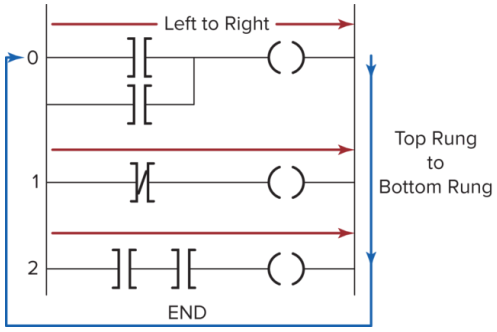


Figure 5-12 Horizontal PLC scanning.

OceanofPDF.com

5.3 PLC Programming Languages

The term ***PLC programming language*** refers to the method by which the user communicates information to the PLC. The standard IEC 61131 (Figure 5-13) was established to standardize the multiple languages associated with PLC programming by defining the following five standard languages:

- **Ladder Diagram (LD)**—a symbolic depiction of instructions arranged in rungs similar to ladder-formatted schematic diagrams.
- **Function Block Diagram (FBD)**—a graphical depiction of process flow using simple and complex interconnecting blocks.
- **Sequential Function Chart (SFC)**—a graphical depiction of interconnecting steps, actions, and transitions.
- **Instruction List (IL)**—a low-level, text-based language that uses mnemonic instructions.
- **Structured Text (ST)**—a high-level, text-based language such as BASIC, C, or PASCAL specifically developed for industrial control applications.

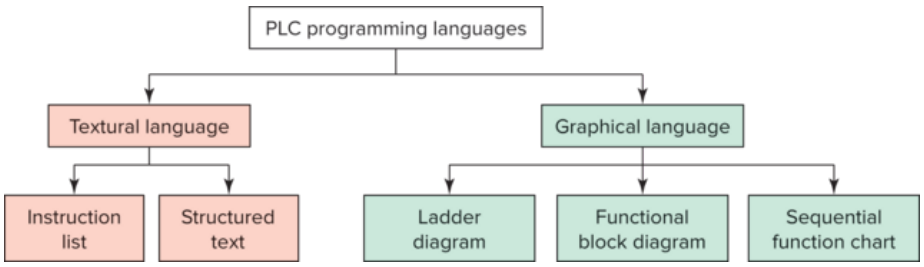
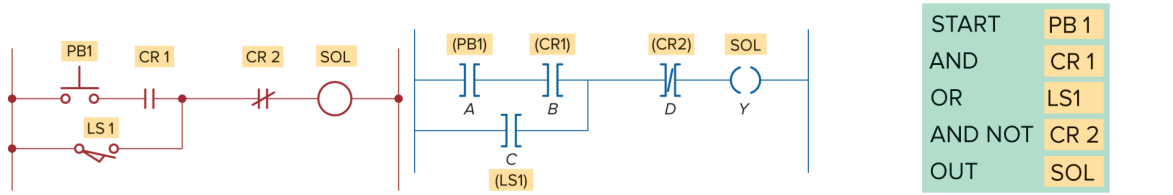


Figure 5-13 Standard IEC 61131 languages associated with PLC programming.

Ladder diagram language is the most commonly used PLC language and is designed to mimic relay logic. The ladder diagram is popular for those who prefer to define control actions in terms of relay contacts and coils, and other functions as block instructions. Figure 5-14 shows a comparison of ladder diagram programming and instruction list programming. Figure 5-14a shows the original relay hardwired control circuit. Figure 5-14b shows the equivalent ladder diagram programmed into a controller. Note how closely the ladder diagram program resembles the hardwired relay circuit. The input/output addressing is generally different for each PLC manufacturer. Figure 5-14c shows how the original hardwired circuit could be programmed using the instruction list programming language. Note that the instructional list consists of a series of instructions that refer to the basic AND, OR, and NOT logic gate functions.



(*a*) Hardwired relay control circuit (*b*) Equivalent ladder diagram (LD) program (c) Equivalent instruction list (IL) program

Figure 5-14 Comparison of ladder diagram and instruction list programming.

Functional block diagram programming uses instructions that are programmed as blocks wired together on screen to accomplish certain functions. Typical types of function blocks include logic, timers, and counters. Functional block diagrams are similar in layout to electrical/electronic block diagrams used to simplify complex systems by showing blocks of functionality. The primary concept behind a functional block diagram is data flow. Function blocks are linked together to complete a circuit that satisfies a control requirement. Data flow on a path from inputs, through function blocks or instructions, and then to outputs.

The use of function blocks for programming of programmable logic controllers (PLCs) is gaining wider acceptance. Rather than the classic contact and coil representation of ladder diagram or relay ladder logic programming, function blocks present a graphical image to the programmer with underlying algorithms already defined. The programmer simply completes needed information within the block to complete that phase of the program. **Figure 5-15** shows function block diagram equivalents to ladder logic contacts.

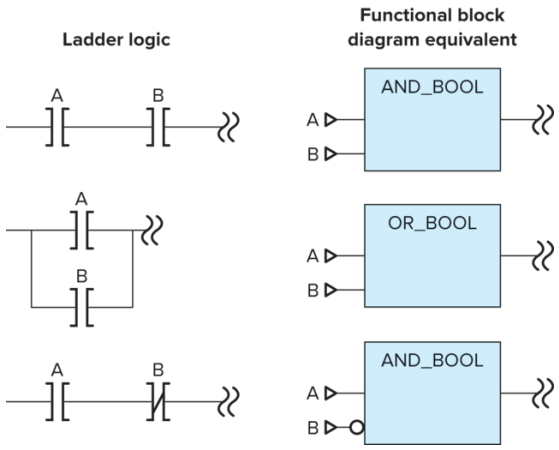


Figure 5-15 Function block diagram equivalents to ladder logic contacts.

Figure 5-16 illustrates how ladder diagram and functional block diagram programming could be used to produce the same logical output. For this application, the objective is to turn on caution pilot light PL 1 whenever both sensor switch 1 and sensor switch 2 are closed. The ladder logic consists of a single rung across the power rails. This rung contains the two input sensor instructions programmed in series with the pilot light output instruction. The function block solution consists of a logic *Boolean*

And function block with two input references tags for the sensors and a single output reference tag for the pilot light. Note there are no power rails in the function block diagram.

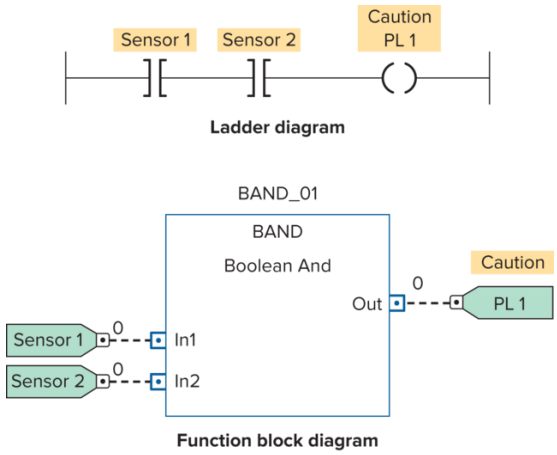


Figure 5-16 PLC ladder and equivalent function block diagram.

Sequential function chart programming language is similar to a flowchart of your process. SFC programming is designed to accommodate the programming of more advanced processes. This type of program can be split into steps with multiple operations happening in parallel branches. The basic elements of a sequential function chart program are shown in **Figure 5-17**.

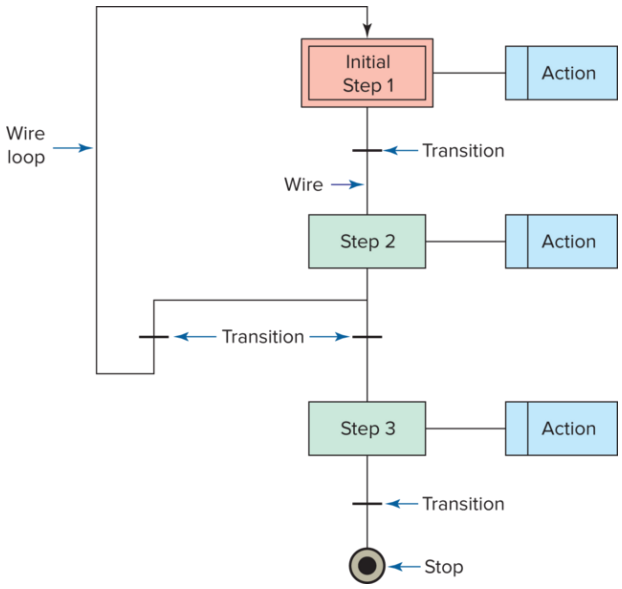
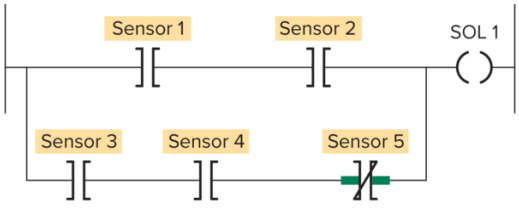


Figure 5-17 Major elements of a sequential function chart program.

Structured text is a high-level text language primarily used to implement complex procedures that cannot be easily expressed with graphical languages. Structured text uses statements to define what to execute. **Figure 5-18** illustrates how structured text and ladder diagram programming could be used

to produce the same logical output. For this application, the objective is to energize SOL 1 whenever either one of the two following circuit conditions exists:

- Sensor 1 and Sensor 2 switches are both closed.
- Sensor 3 and Sensor 4 switches are both closed and Sensor 5 switch is open.



Ladder diagram (LD) program

```
IF Sensor_1 AND Sensor_2 THEN
        SOL_1 := 1;
ELSEIF Sensor_3 AND Sensor_4 AND NOT Sensor_5 THEN
        SOL_1 := 1;
END_IF;
```

Structured text (ST) program

Figure 5-18 PLC ladder and equivalent structured text program.

OceanofPDF.com

5.4 Bit-Level Logic Instructions

The ladder diagram language is basically a ***symbolic*** set of instructions used to create the controller program. Bit-level symbolic instructions fall into two separate categories: instructions that examine data and instructions that control data. Each symbolic instruction is a command to perform a specific operation. These ladder instruction symbols are arranged to obtain the desired control logic that is to be entered into the memory of the PLC.

Representations of contacts and coils are the basic symbols of the logic ladder diagram instruction set. The three fundamental symbols that are used to translate relay control logic to contact symbolic logic are Examine If Closed (XIC), Examine If Open (XIO), and Output Energize (OTE). Each of these instructions relates to a **single bit** of PLC memory that is specified by the instruction's address. While the XIO and XIC are represented by symbols that resemble a normally-open and normally-closed relay contact, they **do not** operate like relay contacts. Instead, they operate as commands that examine the value (0 or 1) of a bit of data to determine its true or false logical condition.

The symbol for the ***Examine If Closed (XIC)*** instruction is shown in **Figure 5-19**. The XIC instruction is also called the Examine-on instruction. Associated with each XIC instruction is a memory bit linked to the status of an input device or an internal logical condition in a rung. This instruction asks the PLC's processor to examine if the contact is *closed.* It does this by examining the bit at the memory location specified by the address in the following manner:

- The memory bit is set to 1 or 0 depending on the status of the input (physical) device or internal (logical) relay address associated with that bit.
- A 1 corresponds to a true status or on condition.
- A 0 corresponds to a false status or off condition.
- When the Examine-on instruction is associated with a physical input, the instruction will be set to 1 when a physical input is present (voltage is applied to the input terminal), and 0 when there is no physical input present (no voltage applied to the input terminal).
- When the Examine-on instruction is associated by address with an internal relay, then the status of the bit is dependent on the logical status of the internal bit with the same address as the instruction.
- If the instruction memory bit is a 1 (true) this instruction will allow rung continuity through itself, like a closed relay contact.
- If the instruction memory bit is a 0 (false) this instruction will not allow rung continuity through itself and will assume a normally open state just like an open relay contact.

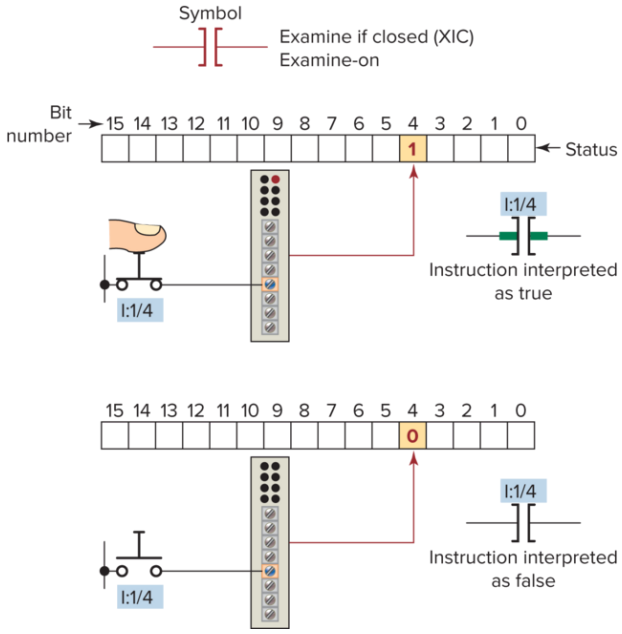


Figure 5-19 Examine If Closed (XIC) instruction.

The symbol for the ***Examine If Open (XIO)*** instruction is shown in **Figure 5-20**. The XIO instruction, which is also called the Examine-off instruction, looks and operates like a normally closed relay contact. Associated with each XIO instruction is a memory bit linked to the status of an input device or an internal logical condition in a rung. This instruction asks the PLC's processor to examine if the contact is *open.* It does this by examining the bit at the memory location specified by the address in the following manner:

- As with any other input the memory bit is set to 1 or 0 depending on the status of the input (physical) device or internal (logical) relay address associated with that bit.
- A 0 corresponds to a true status.
- A 1 corresponds to a false status.
- When the Examine-off instruction is used to examine a physical input, then the instruction will be interpreted as false when there is a physical input (voltage) present (the bit is 1) and will be interpreted as true when there is no physical input present (the bit is 0).
- If the Examine-off instruction were associated by address with an internal relay, then the status of the bit would be dependent on the logical status of the internal bit with the same address as the instruction.
- Like the Examine-on instruction, the status of the instruction (true or false) determines if the instruction will allow rung continuity through itself, like a closed relay contact.
- The memory bit always follows the status (true = 1 or false = 0) of the input address or internal address assigned to it. The interpretation of that bit, however, is determined by which instruction is used to examine it.
- Examine-on instructions always interpret a 1 status as true and a 0 status as false, while Examine-off

instructions interpret a 1 status as false and a 0 status as true, as illustrated in Figure 5-21.

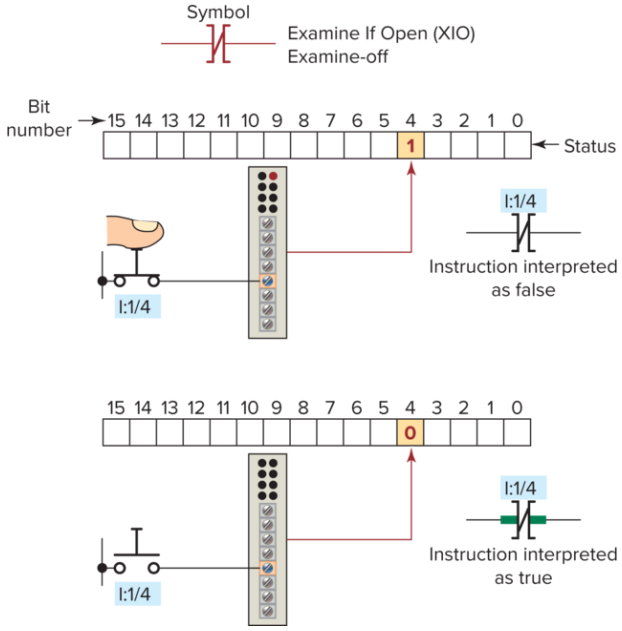


Figure 5-20 Examine If Open (XIO) instruction.

Instruction	Name	Symbol	Description
XIC	Examine If Closed	]\[	Examines a bit for an ON state 0 (false) 1 (true)
XIO	Examine If Open	]/[	Examines a bit for an OFF state 0 (true) 1 (false)
OTE	Output Energize	( )	Energize Output (1) De-energize Output (0)

Figure 5-21 Fundamental ladder diagram instructions.

The symbol for the ***Output Energize (OTE)*** instruction is shown in Figure 5-22. The OTE instruction looks and operates like a relay coil and is associated with a memory bit. This instruction signals the PLC to energize (switch on) or de-energize (switch off) the output. The processor makes this instruction true (analogous to energizing a coil) when there is a logical path of true XIC and XIO instructions in the rung. The operation of the Output Energize instruction can be summarized as follows:

- The status bit of the addressed Output Energize instruction is set to 1 to energize the output and to 0 to de-energize the output.
- If a true logic path is established with the input instructions in the rung, the OTE instruction is energized and the output device wired to its terminal is energized.
- If a true logic path cannot be established or rung conditions go false, the OTE instruction is de-energized and the output device wired to it is switched off.

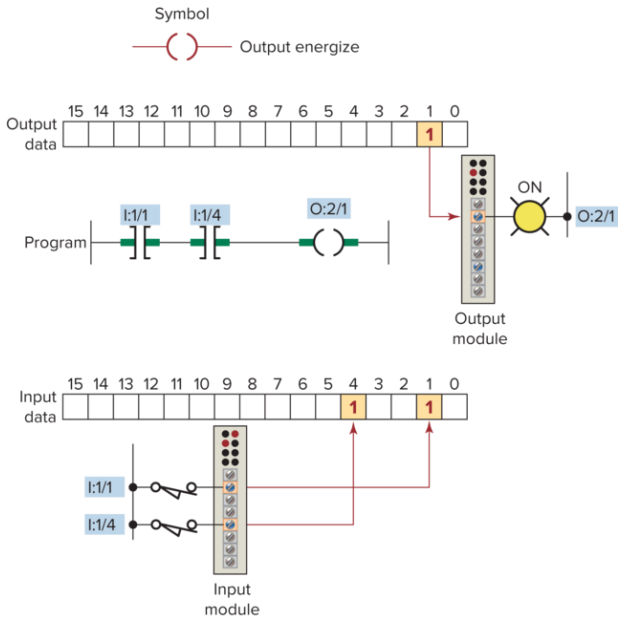


Figure 5-22 Output Energize (OTE) instruction.

Sometimes beginner programmers who are used to thinking in terms of hardwired relay control circuits tend to use the same type of contact (NO or NC) in the ladder logic program that corresponds to the type of field switch wired to the discrete input. While this is true in many instances, it is not the best way to think of the concept. A better approach is to separate the action of the field device from the action of the PLC bits as illustrated in **Figure 5-23**. A signal present makes the NO bit (1) true; a signal absent makes the NO bit (0) false. The reverse is true for an NC bit. A signal present makes the NC bit (1) false; a signal absent makes the NC bit (0) true.

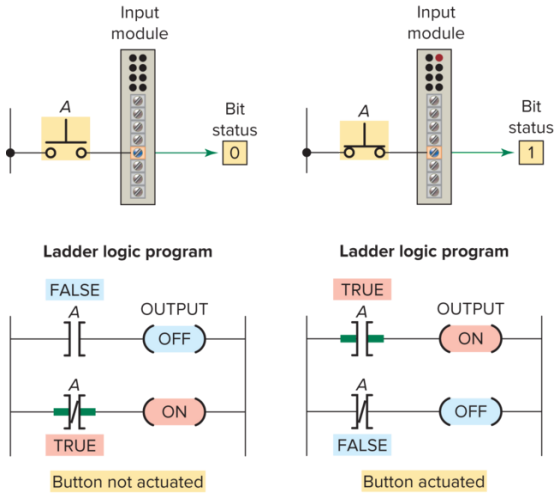


Figure 5-23 Separating the action of the field device and PLC bit.

The main function of the ladder logic diagram program is to control outputs based on input conditions, as illustrated in Figure 5-24. This control is accomplished through the use of what is referred to as a ladder rung. In general, a rung consists of a set of input conditions, represented by contact instructions, and an output instruction at the end of the rung, represented by the coil symbol. Each contact or coil symbol is referenced with an address that identifies what is being evaluated and what is being controlled. The same contact instruction can be used throughout the program whenever that condition needs to be evaluated. While this is true for the XIO and XIC contact instructions, the same **cannot** be said for the **OTE coil** instruction. A common mistake for the novice programmer is to place the same addressed OTE instruction on multiple rungs within the same program. This practice is to be avoided since it will lead to unpredictable program outcomes. The number of ladder logic relays and input and output instructions is limited only by memory size. Most PLCs allow more than one output per rung.

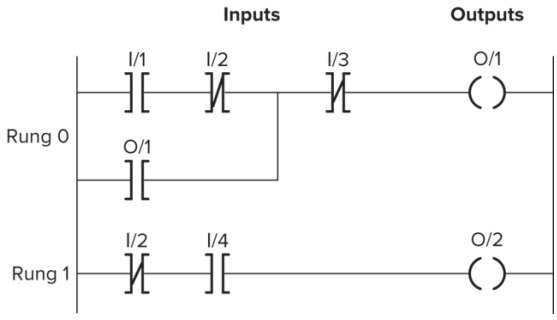


Figure 5-24 Ladder logic diagram rungs.

For an output to be activated or energized, at least one left-to-right true logical path must exist, as illustrated in Figure 5-25. A complete closed path is referred to as having logical continuity. When logical continuity exists in at least one path, the rung condition and Output Energize instruction are

said to be true. The rung condition and OTE instruction are false if no logical continuity path has been established. During controller operation, the processor evaluates the rung logic and changes the state of the outputs according to the logical continuity of rungs.

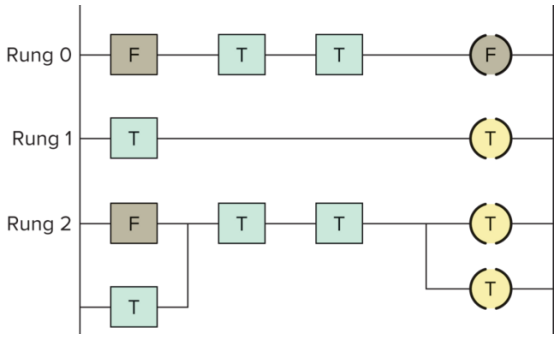


Figure 5-25 Logical continuity.

OceanofPDF.com

5.5 Instruction Addressing

To complete the entry of a relay-type instruction, you must assign an **address** or tag to each instruction. This address indicates what PLC input is connected to what input device and what PLC output will drive what output device.

The addressing of real inputs and outputs, as well as internals, depends on the PLC model used. Addressing formats can vary from one PLC family to another as well as for different manufacturers. These addresses can be represented in decimal, octal, hexadecimal or text, depending on the number system used by the PLC. The address identifies the function of an instruction and links it to a particular location in memory. **Figure 5-26** shows the addressing format for an RSLogix 500 controller. Addresses contain the slot number of the module where input or output devices are connected. Addresses are formatted as file type, file number, slot number, and bit.

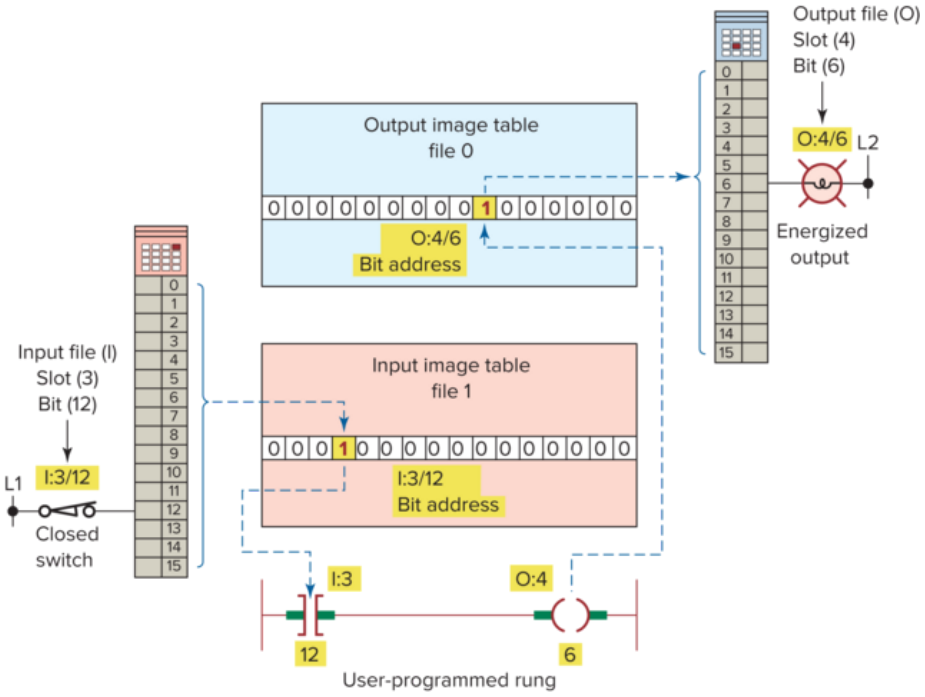


Figure 5-26 Addressing format for an RSLogix 500 controller.

RSLogix 5000 controllers offer a more flexible method of addressing memory space. Instead of a fixed device with a fixed address space, **tags** are used for assigning and referencing memory spaces. Tags are text based and can be descriptive.

The assignment of an I/O address can be included in the I/O connection diagram, as shown in **Figure 5-27**. Inputs and outputs are typically represented by squares and diamonds, respectively.

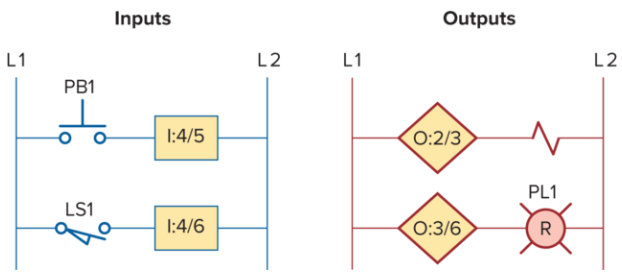


Figure 5-27 I/O connection diagram.

OceanofPDF.com

5.6 Branch Instructions

Branch instructions are used to create parallel paths of input condition instructions. This allows more than one combination of input conditions (OR logic) to establish logic continuity in a rung.

Figure 5-28 illustrates a typical branch instruction. The rung will be true if either instruction *A* or *B* is true.

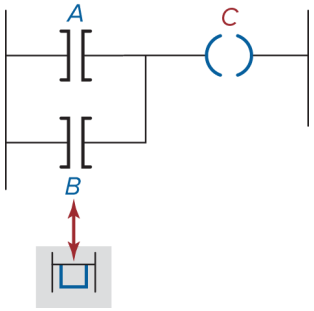


Figure 5-28 Typical branch instruction.

Input branching by formation of parallel branches can be used in your application program to allow more than one combination of input conditions. If at least one of these parallel branches forms a true logic path, the rung logic is true and the output will be energized. If none of the parallel branches complete a logical path, logic rung continuity is not established and the output will be de-energized. In the example shown in **Figure 5-29**, either *A* and NOT *B,* or *C* provides logical continuity and energizes output *D.*

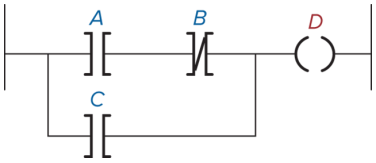


Figure 5-29 Parallel input branches.

On most PLC models, branches can be established at both input and output portions of a rung. With **output branching,** you can program parallel outputs on a rung to allow a true logic path to control multiple outputs, as illustrated in **Figure 5-30**. When there is a true logic rung path, all parallel outputs become true. In the example shown, either *A* or *B* provides a true logical path to all three output instructions: *C, D,* and *E.*

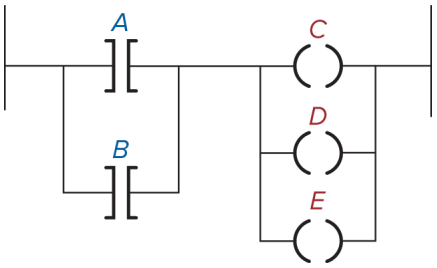


Figure 5-30 Parallel output branches.

Additional input logic instructions (conditions) can be programmed in the output branches to enhance conditional control of the outputs. When there is a true logic path, including extra input conditions on an output branch, that branch becomes true. In the example shown in **Figure 5-31**, either *A* and *D* or *B* and *D* provide a true logic path to *E*.

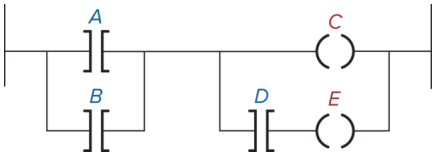


Figure 5-31 Parallel output branching with conditions.

Input and output branches can be ***nested*** to avoid redundant instructions and to speed up processor scan time. **Figure 5-32** illustrates nested input and output branches. A nested branch starts or ends within another branch.

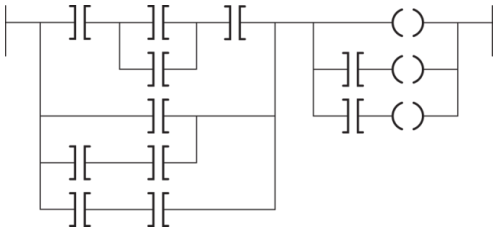


Figure 5-32 Nested input and output branches.

In some PLC models, the programming of a branch circuit within a branch circuit or a *nested* branch cannot be done directly. It is possible, however, to program a logically equivalent branching condition. **Figure 5-33** shows an example of a circuit that contains a nested contact *D*. To obtain the required logic, the circuit would be programmed as shown in **Figure 5-34**. The duplication of contact *C* eliminates the nested contact *D*. Nested branching can be converted into non-nested branches by repeating instructions to make parallel equivalents.

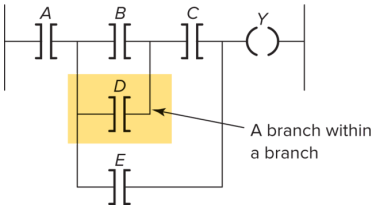


Figure 5-33 Nested contact program.

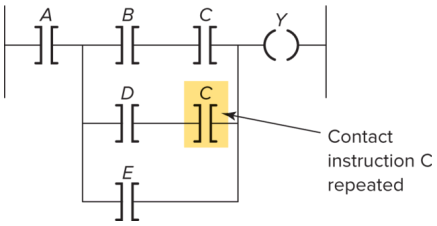
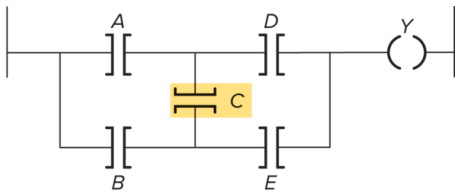


Figure 5-34 Program required to eliminate nested contact.

Some PLC manufacturers have no limitations on allowable series elements, parallel branches, or outputs. For others, there may be limitations to the number of series contact instructions that can be included in one rung of a ladder diagram as well as limitations to the number of parallel branches. Also, there is an additional limitation with some PLCs: only one output per rung and the output must be located at the end of the rung. The only limitation on the number of rungs is memory size.

Another limitation to branch circuit programming is that the PLC will not allow for programming of vertical contacts. A typical example of this limitation is contact *C* of the user program drawn in **Figure 5-35**. To obtain the required logic, the circuit would be reprogrammed as shown in **Figure 5-36**.



Boolean equation: $Y = (AD) + (BCD) + (BE) + (ACE)$

Figure 5-35 Program with vertical contact.

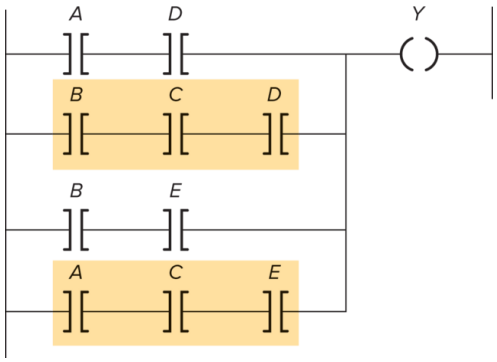
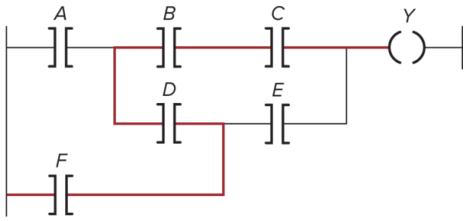


Figure 5-36 Reprogrammed to eliminate vertical contact.

The processor examines the ladder logic rung for logic continuity from left to right *only*. The processor never allows for flow from right to left. This situation presents a problem for user program circuits similar to that shown in **Figure 5-37**. If programmed as shown, contact combination *FDBC* would be ignored. To obtain the required logic, the circuit would be reprogrammed as shown in **Figure 5-38**. The best way to avoid nesting is to keep in mind that all branches must start at a common point and end up at the same location. With newer controllers, such as RSLogix 5000, nesting can be used with some restriction such as the number of branch levels that can be used.



Boolean equation: $Y = (ABC) + (ADE) + (FE) + (FDBC)$

Figure 5-37 Original circuit.

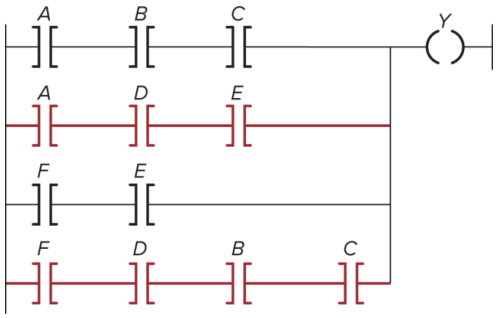


Figure 5-38 Reprogrammed circuit.

OceanofPDF.com

5.7 Internal Relay Instructions

Most PLCs have an area of the memory allocated for what are known as internal storage bits. These storage bits are also called ***internal outputs, internal coils, internal control relays,*** or simply ***internal bits.*** Internal outputs are on/off signals generated by programmed logic. Unlike a discrete output, an internal output does not directly control an output field device. The internal output operates just like any output that is controlled by programmed logic; however, the output is used strictly for internal purposes.

The advantage of using internal outputs is that there are many situations in which an output instruction is required in a program but no physical connection to a field device is needed. If there are no physical outputs wired to a bit address, the address can be used as an internal storage point. Internal storage bits or points can be programmed by the user to perform relay functions without occupying a physical output. In this way internal outputs can minimize output module point requirements whenever practical.

Internal outputs are single-bit storage locations in memory and are addressed as such. RSLogix 500 controllers use bit file B3 for storage and addressing of internal output bits. In an RSLogix 5000 controller all you need to do for an internal coil is create a Tag with a data type of BOOL.

An internal control relay can be used when a program requires more series contacts than the rung allows. **Figure 5-39** shows a circuit that allows for only 7 series contacts when 12 are actually required for the programmed logic. To solve this problem, the contacts are split into two rungs. Rung 1 contains seven of the required contacts and is programmed to control the internal relay coil. When the logic controlling the internal output is true, the referenced bit is turned on or set to 1. The advantage of an internal storage bit in this manner is that it saves an output bit from being used.

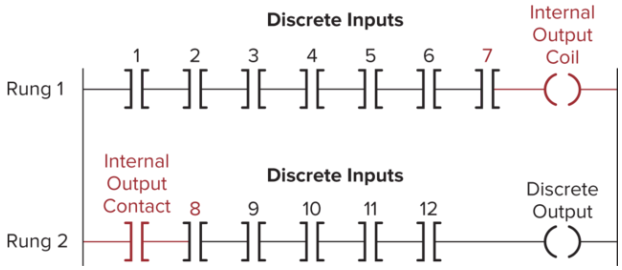


Figure 5-39 Programmed internal relay control.

OceanofPDF.com

5.8 Programming Examine If Closed and Examine If Open Instructions

A simple program using the Examine If Closed (XIC) instruction is shown in Figure 5-40. This figure shows a hardwired circuit and a user program that provides the same results. You will note that *both the NO and the NC* pushbuttons are represented by the Examine If Closed symbol. This is because the normal state of an input (NO or NC) does not matter to the controller. What does matter is that if contacts need to close to energize the output, then the Examine If Closed instruction is used. Since both PB1 and PB2 must be closed to energize the pilot light, the Examine If Closed instruction is used for both.

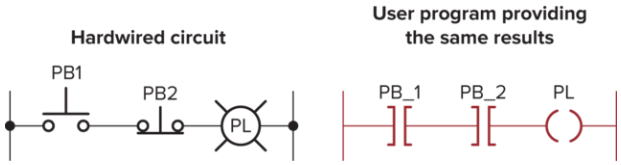


Figure 5-40 Simple program that uses the Examine If Closed (XIC) instruction.

A simple program using the Examine If Open (XIO) instruction is shown in Figure 5-41. Both the hardwired circuit and user program are shown. In the hardwired circuit, when the pushbutton is *open* relay coil CR is de-energized and its NO contact closes to switch the pilot light on. When the pushbutton is *closed,* relay coil CR is energized and its NC contact opens to switch the pilot light off. The pushbutton is represented in the user program by an Examine If Open instruction. This is because the rung must be true when the external pushbutton is open and false when the pushbutton is closed. Using an Examine If Open instruction to represent the pushbutton satisfies these requirements. The NO or NC mechanical action of the pushbutton is not a consideration. It is important to remember that the user program is not an electrical circuit but a *logic* circuit. In effect, we are interested in logic continuity when establishing an output.

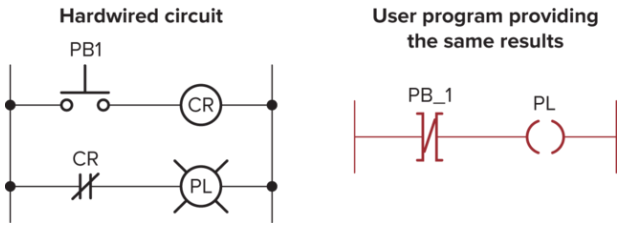


Figure 5-41 Simple program that uses the Examine If Open (XIO) instruction.

Figure 5-42 shows a simple program using both the XIC and XIO instructions. The logic states (0 or 1) indicate whether an instruction is true or false and is the basis of controller operation. The figure summarizes the on/off state of the output as determined by the changing states of the inputs in the

rung. The time aspect relates to the repeated scans of the program, wherein the input table is updated with the most current status bits.

If the data table bit is	The status of the instruction is		
	XIC EXAMINE IF CLOSED	XIO EXAMINE IF OPEN	OTE OUTPUT ENERGIZE
Logic 0	False	True	False
Logic 1	True	False	False



Time	Instruction outcome			Input bit status		
	XIC	XIO	OTE	XIC	XIO	OTE
t_1 (initial)	False	True	False	0	0	0
t_2	True	True	Goes true	1	0	1
t_3	True	False	Goes false	1	1	0
t_4	False	False	Remains false	0	1	0

Figure 5-42 Simple program using both the XIC and XIO instructions.

OceanofPDF.com

5.9 Entering the Ladder Diagram

Most of today's PLC programming packages operate in the **Windows environment**. For example, Allen-Bradley's RSLogix software packages are Windows programming packages used to develop ladder logic programs. This software, in various versions, can be used to program the ControlLogix, MicroLogix, and CompactLogix family of processors.

Because hardware and programming techniques vary with each manufacturer, it is necessary to refer to the programming manual for a specific PLC to determine how the instructions are entered.

Figure 5-43 shows the RSLogix 500 main window. Different screens, toolbars, and dialog boxes are used to navigate through the Windows environment. It is important that you understand the purpose of the various screens, toolbars, and windows to make the most effective use of the software. This information is available from the software reference manual for the particular PLC family and will become more familiar to you as you develop programs using the software.

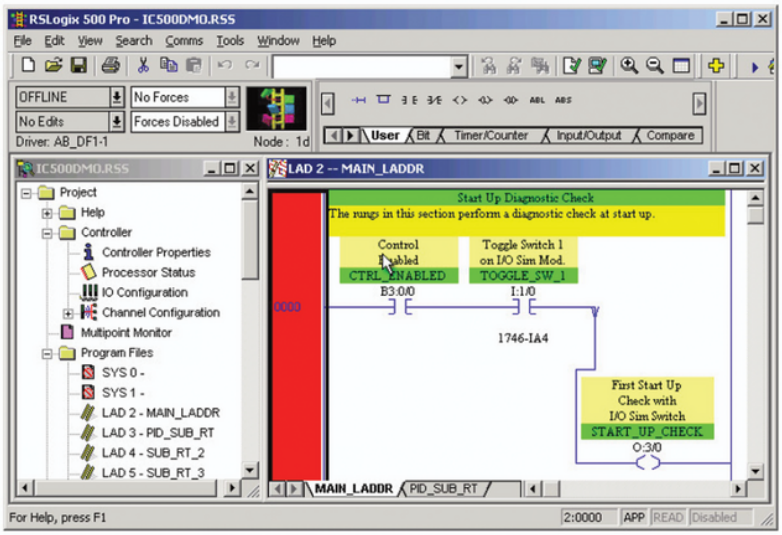


Figure 5-43 RSLogix 500 main window.

Source: Image Courtesy of Rockwell Automation, Inc.

Figure 5-44 shows a typical instruction toolbar with bit instructions selected. To place an instruction on a rung, click its icon on the toolbar and simply drag the instruction straight off the toolbar onto the rung of the ladder. Drop points are shown on the ladder to help position the instruction. In addition, instructions can also be dragged from other rungs in the project. There are several different methods that you can use to address instructions. You can enter an address by manually typing it in or by dragging the address from data files or other instructions.



Figure 5-44 Typical instruction toolbar with bit instructions selected.

Some of the windows you will need to use when working with RSLogix 500 software include:

- **Main Window**—This window opens each time you create a new project or open an existing one. Some of the features associated with this window include the following:
  - Window Title Bar—The title bar is located at the topmost strip of the window and displays the name of the program as well as that of the opened file.
  - Menu Bar—The menu bar is located below the title bar. The menu contains key words associated with menus that are opened by clicking on the key word.
  - Windows Toolbar—The Windows toolbar buttons execute standard Windows commands when you click on them.
  - Program/Processor Status Toolbar—This toolbar contains four drop-down lists that identify the current processor operating mode, current online edit status, and whether forces are present and enabled.
  - Project Window—This window displays the file folders listed in the project tree.
  - Project Tree—The project tree is a visual representation of all folders and their associated files contained in the current project. From the project tree, you can open files, create files, modify file parameters, copy files, hide or unhide files, delete files, and rename files.
  - Result Window—This window displays the results of either a search or a verify operation. The verify operation is used to check the ladder program for errors.
  - Active Tab—This tab identifies which program is currently active.
  - Status Bar—This bar contains information relevant to the current file.
  - Split Bar—The split bar is used to split the ladder window to display two different program files or groups of ladder rungs.
  - Tabbed Instruction Toolbar—This toolbar displays the instruction set as a group of tabbed categories.
  - Instruction Palette—This tool contains all the available instructions displayed in one table to make the selection of instructions easier.
  - Ladder Window—This window displays the currently open ladder program file and is used to develop and edit ladder programs.
  - Ladder Window Properties—This window allows you to change the display of your ladder program and its associated addressing and documentation.
- **Select Processor Type**—The programming software needs to know what processor is being used in conjunction with the user program. The Select Processor Type screen (**Figure 5-45**) contains a

list of the different processors that the RSLogix software can program. You simply scroll down the list until you find the processor you are using and select it.

- **I/O Configuration**—The I/O Configuration screen (Figure 5-46) lets you click or drag-and-drop a module from an all-inclusive list to assign it to a slot in your configuration.
- **Data Files**—Data File screens contain data that are used in conjunction with ladder program instructions and include input and output files as well as timer, counter, integer, and bit files. Figure 5-47 shows an example of the bit file B3, which is used for internal relays. Note that all the addresses from this file start with B3.

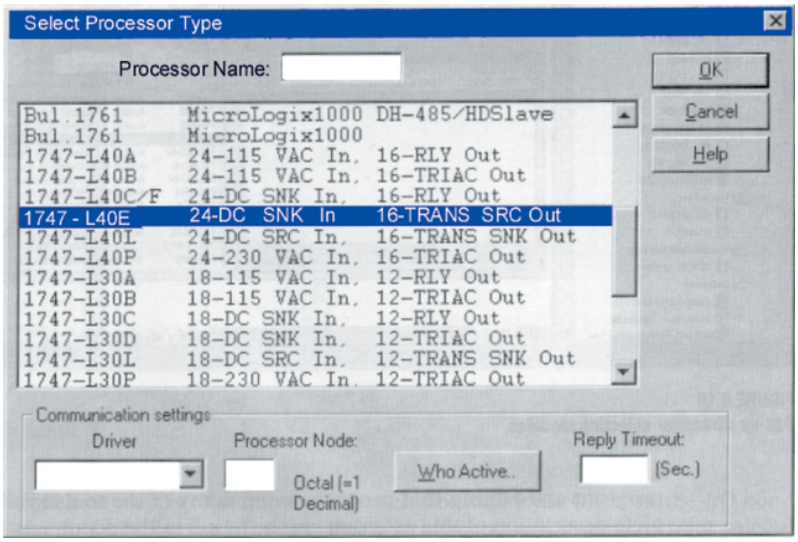


Figure 5-45 Select processor type screen.

Source: Image Courtesy of Rockwell Automation, Inc.

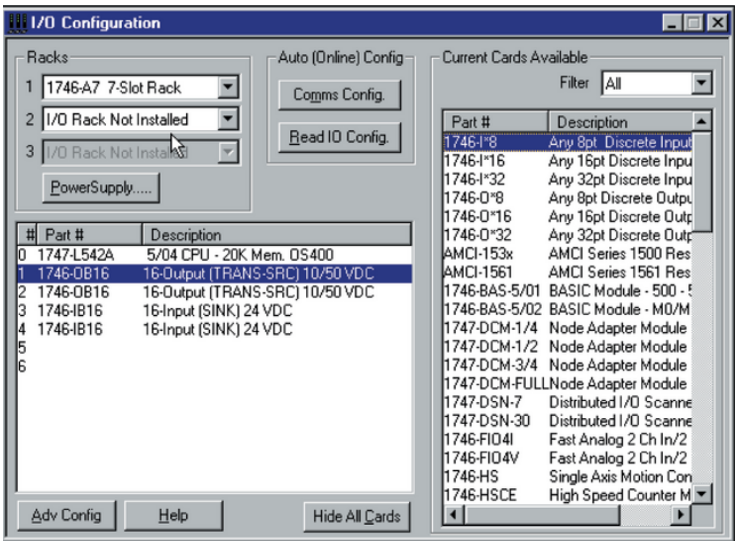


Figure 5-46 I/O configuration screen.

Source: Image Courtesy of Rockwell Automation, Inc.

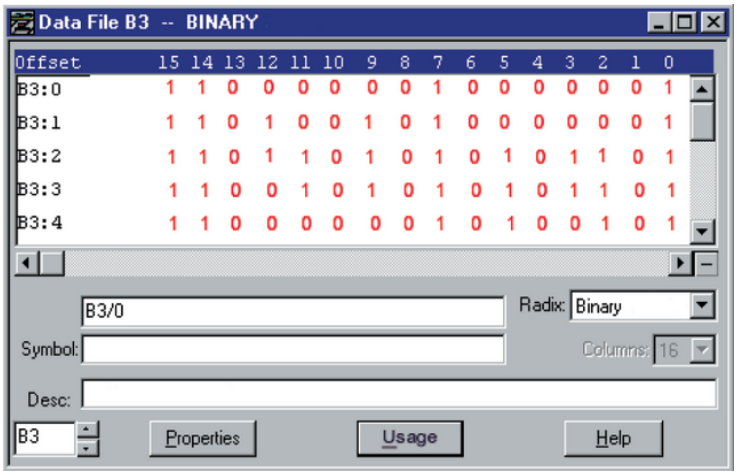


Figure 5-47 Data bit file B3 screen.

Source: Courtesy of TheLearningPit.

Ladder logic programming is particularly effective in an online mode when the PLC is actually performing control. Operation of the logic is apparent from the highlighting of rungs of the various instructions on-screen, which identifies the logic state of contacts in real time (**Figure 5-48**) and which rungs have logic continuity.

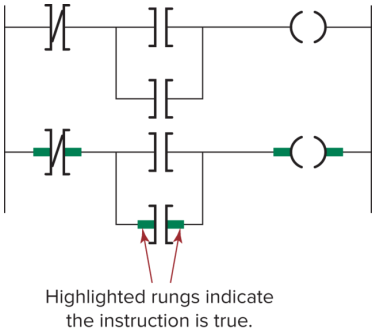


Figure 5-48 Monitoring a ladder logic program.

For most PLC systems, each Examine If Closed and Examine If Open contact, each output, and each branch Start/End instruction requires one word of user memory. You can refer to the RSLogix 500 Controller Properties to see the number of instruction words used and the number left as the program is being developed.

OceanofPDF.com

5.10 Modes of Operation

A processor has basically two modes of operation: the *program mode* and some variation of the *run mode.* The number of different operating modes and the method of accessing them varies with the manufacturer. **Figure 5-49** shows a typical three-position keyswitch used to select different processor modes of operation.

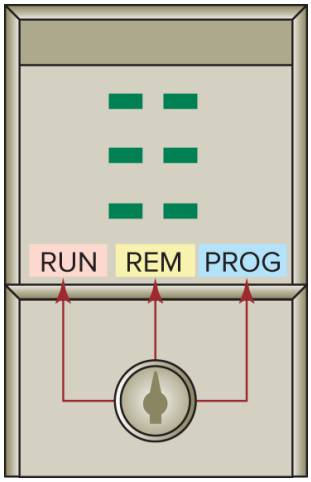


Figure 5-49 Three-position keyswitch used to select different processor modes of operation.

Some common operating modes are explained in the following paragraphs.

Program Mode The program mode is used to enter a new program, edit or update an existing program, upload files, download files, document (print out) programs, or change any software configuration file in the program. When the PLC is switched into the program mode, all outputs from the PLC are forced off regardless of their rung logic status, and the ladder I/O scan sequence is halted.

Run Mode The run mode is used to execute the user program. Input devices are monitored and output devices are energized accordingly. After all instructions have been entered in a new program or all changes made to an existing program, the processor is put in the run mode.

Test Mode The test mode is used to operate or monitor the user program without energizing any outputs. The processor still reads inputs, executes the ladder program, and updates the output status table files, but without energizing the output circuits. This feature is often used after developing or editing a program to test the program execution before allowing the PLC to operate real-world outputs. Variations of the test mode can include the *single-step test mode,* which directs the processor to execute a selected single rung or group of rungs; the *single-scan test mode,* which executes a single processor operating scan or cycle; and the *continuous-scan test mode,* which directs the processor to continuously run the program for checking or troubleshooting.

Remote Mode Some processors have a three-position switch to change the processor operating mode. In the Run position, all logic is solved and the I/O is enabled. In the Program position, all logic solving is stopped and the I/O is disabled. The Remote position allows the PLC to be remotely changed between program and run mode by a personal computer connected to the PLC processor. The remote mode may be beneficial when the controller is in a location that is not easily accessible.

OceanofPDF.com

5.11 Connecting with Analog Devices

Electrical devices and signals can be divided into two categories: analog and digital. **Digital** devices operate using discrete ON or OFF signals that have only two possible values. **Analog** signals can take any shape and represent an infinite number of possible values, as illustrated in **Figure 5-50**. Analog circuits are usually much more susceptible to noise (small, undesired variations in voltage). Small changes in the voltage level of an analog signal may produce significant errors as the signal is processed. Analog signals must be coded into digital signals before they can be processed by the PLC. An analog-to-digital converter (ADC) converts analog input signals to digital signals. A digital-to-analog converter (DAC) converts digital output signals to analog signals.

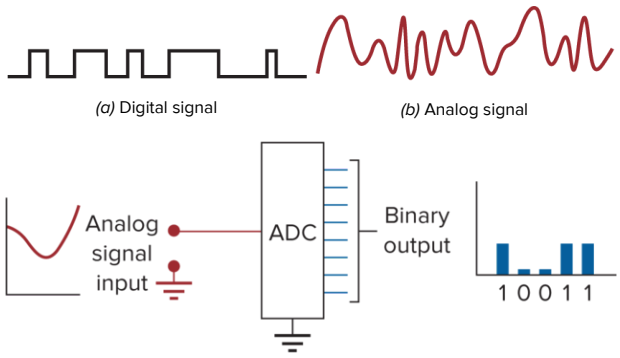


Figure 5-50 Digital and analog signals.

Analog applications are present in many forms. **Figure 5-51** shows a typical use of analog control for a tank-filling process. The operation of the circuit can be summarized as follows:

- The processor controls the amount of fluid placed in a holding tank by adjusting the percentage of the valve opening.
- The valve is initially open 100%.
- As the fluid level in the tank approaches the level preset point, the processor modifies the output to degrade, closing the valve to 90%, 80%, etc., adjusting the valve to maintain a set point.

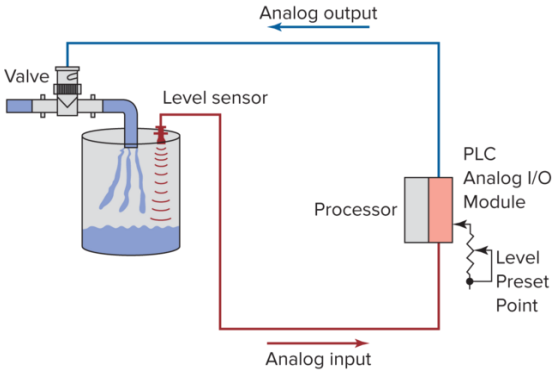


Figure 5-51 Analog control for a tank-filling process.

OceanofPDF.com



CHAPTER 5 REVIEW QUESTIONS

1. What does the memory map for a typical PLC processor consist of?
2. Compare the function of the PLC program and data files.
3. In what manner are data files organized?
4. List eight different types of data files used by an RSLogix 500 controller.
5. **a.** What information is stored in the input image table file?
   b. In what form is this information stored?
6. **a.** What information is stored in the output image table file?
   b. In what form is this information stored?
7. Outline the sequence of events involved in a PLC scan cycle.
8. List four factors that enter into the length of the scan time.
9. Compare the way horizontal and vertical scan patterns examine input and output instructions.
10. List the five standard PLC languages as defined by the International Standard for Programmable Controllers, and give a brief description of each.
11. Draw the symbol and state the equivalent instruction for each of the following: NO contact, NC contact, and coil.
12. Answer the following with regard to the Examine If Closed instruction:
    a. What is another common name for this instruction?
    b. What is this instruction asking the processor to examine?
    c. Under what condition is the status bit associated with this instruction 0?
    d. Under what condition is the status bit associated with this instruction 1?
    e. Under what condition is this instruction logically true?
    f. What state does this instruction assume when it is false?
13. Answer the following with regard to the Examine If Open instruction:
    a. What is another common name for this instruction?
    b. What is this instruction asking the processor to examine?
    c. Under what condition is the status bit associated with this instruction 0?
    d. Under what condition is the status bit associated with this instruction 1?
    e. Under what condition is this instruction logically true?
    f. What state does this instruction assume when it is false?
14. Answer the following with regard to the Output Energize instruction:

a. What part of an electromagnetic relay does this instruction look and act like?

b. What is this instruction asking the processor to do?

c. Under what condition is the status bit associated with this instruction 0?

d. Under what condition is the status bit associated with this instruction 1?

15. A normally closed pushbutton is connected to a PLC discrete input. Does this mean it must be represented by a normally closed contact in the ladder logic program? Explain why or why not.

16. Answer the following with regard to a ladder logic rung:

a. Describe the basic makeup of a ladder logic rung.

b. How are the contacts and coil of a rung identified?

c. When is the ladder rung considered as having logic continuity?

17. What does the address assigned to an instruction indicate?

18. When are input branch instructions used as part of a ladder logic program?

19. In what way does an internal output differ from a discrete output?

20. A normally open limit switch is to be programmed to control a solenoid. What determines whether an Examine-on or Examine-off contact instruction is used?

21. Explain the purpose of Windows based programming software such as RSLogix.

22. Briefly describe each of the following PLC modes of operation:

a. Program

b. Test

c. Run

23. Under what condition is a ladder logic rung said to have logic continuity?

24. Electrical devices and signals can be divided into two categories: analog and digital. What is the major difference between these two categories?

25. Compare the way memory space is assigned and referenced in rack-based and tag-based PLCs.

OceanofPDF.com



CHAPTER 5 PROBLEMS

1. Assign each of the following discrete input and output addresses based on the RSLogix 500 format.

a. Limit switch connected to terminal screw 4 of the module in slot 1 of the chassis.

b. Pressure switch connected to terminal screw 2 of the module in slot 3 of the chassis.

c. Pushbutton connected to terminal screw 0 of the module in slot 6 of the chassis.

d. Pilot light connected to terminal screw 13 of the module in slot 2 of the chassis.

e. Motor starter coil connected to terminal screw 6 of the module in slot 4 of the chassis.

f. Solenoid connected to terminal screw 8 of the module in slot 5 of the chassis.

2. Redraw the program shown in **Figure 5-52** corrected to solve the problem of a nested contact.

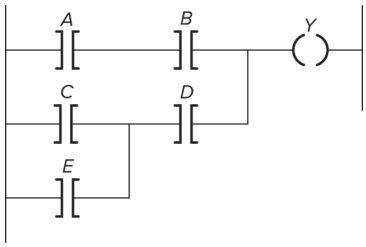


Figure 5-52 Program for Problem 2.

3. Redraw the program shown in **Figure 5-53** corrected to solve the problem of a nested vertical programmed contact.

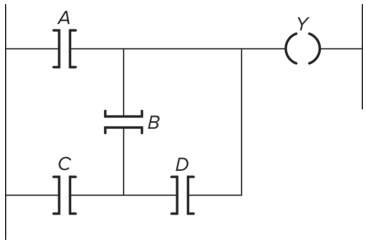


Figure 5-53 Program for Problem 3.

4. Redraw the program shown in **Figure 5-54** corrected to solve the problem of some logic ignored.

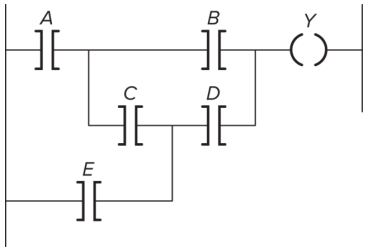


Figure 5-54 Program for Problem 4.

5. Redraw the program shown in **Figure 5-55** corrected to solve the problem of too many series contacts (only four allowed).

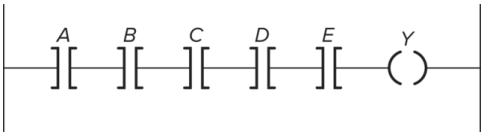


Figure 5-55 Program for Problem 5.

6. Draw the equivalent ladder logic program used to implement the hardwired circuit drawn in **Figure 5-56**, wired using:

a. A limit switch with a single NO contact connected to the PLC discrete input module

b. A limit switch with a single NC contact connected to the PLC discrete input module

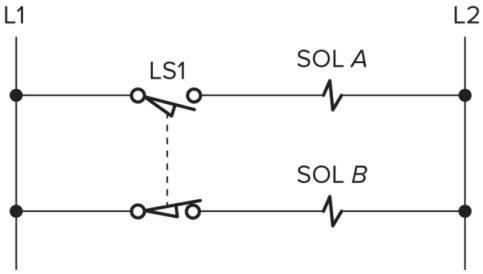


Figure 5-56 Hardwired circuit for Problem 6.

7. Assuming the hardwired circuit drawn in **Figure 5-57** is to be implemented using a PLC program, identify

a. All input field devices

b. All output field devices

c. All devices that could be programmed using internal relay instructions

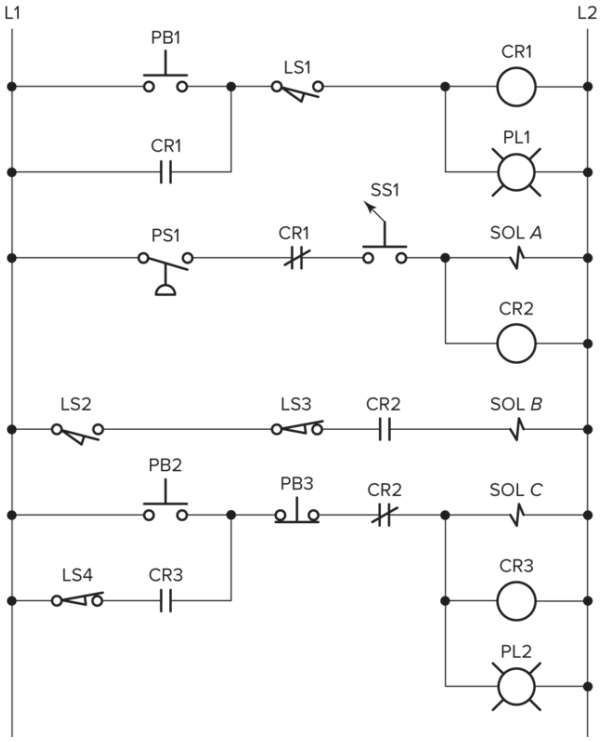


Figure 5-57 Hardwired circuit for Problem 7.

8. What instruction would you select for each of the following discrete input field devices to accomplish the desired task? (State the reason for your answer.)

a. Turn on a light when a conveyor motor is running in reverse. The input field device is a set of contacts on the conveyor start relay that close when the motor is running forward and open when it is running in reverse.

b. When a pushbutton is pressed, it operates a solenoid. The input field device is a normally open pushbutton.

c. Stop a motor from running when a pushbutton is pressed. The input field device is a normally closed pushbutton.

d. When a limit switch is closed, it triggers an instruction ON. The input field device is a limit switch that stores a 1 in a data table bit when closed.

9. Write the ladder logic program needed to implement each of the following (assume inputs *A, B,* and *C* are all normally open toggle switches):

a. When input *A* is closed, turn ON and hold ON outputs *X* and *Y* until *A* opens.

b. When input *A* is closed and either input *B* or *C* is open, turn ON output *Y*; otherwise, it should be OFF.

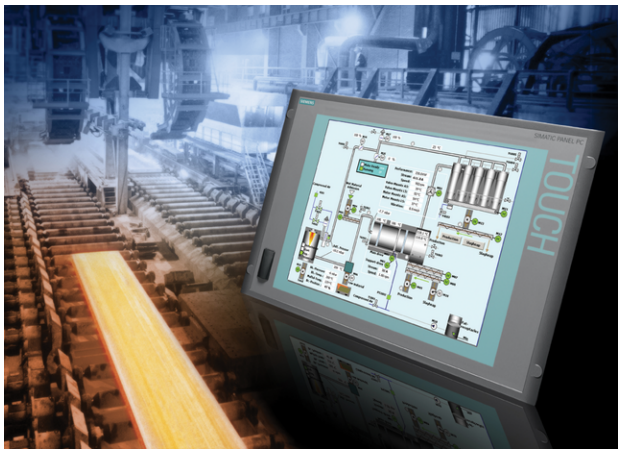
c. When input *A* is closed or open, turn ON output *Y.*

d. When input *A* is closed, turn ON output *X* and turn OFF output *Y.*

OceanofPDF.com

6

Developing Fundamental PLC Wiring Diagrams and Ladder Logic Programs



Courtesy of Siemens

Chapter Objectives

After completing this chapter, you will be able to:

- Identify the functions of electromagnetic control relays, contactors, and motor starters
- Identify switches commonly found in PLC installations
- Explain the operation of sensors commonly found in PLC installations
- Explain the operation of output control devices commonly found in PLC installations
- Describe the operation of an electromagnetic latching relay and the PLC-programmed LATCH/UNLATCH instruction
- Compare sequential and combination control processes
- Convert fundamental relay ladder diagrams to PLC ladder logic programs
- Write PLC programs directly from a narrative description

For ease of understanding, ladder logic programs can be compared to relay schematics. This chapter gives examples of how traditional relay schematics are converted into PLC ladder logic programs. You will learn more about the wide variety of field devices commonly used in connection with the I/O modules.

OceanofPDF.com

6.1 Electromagnetic Control Relays

The PLC's original purpose was the replacement of **electromagnetic relays** with a solid-state switching system that could be programmed. Although the PLC has replaced much of the relay control logic, electromagnetic relays are still used as auxiliary devices to switch I/O field devices. The programmable controller is designed to replace the physically small control relays that make logic decisions but are not designed to handle heavy current or high voltage (Figure 6-1). In addition, an understanding of electromagnetic relay operation and terminology is important for correctly converting relay schematic diagrams to ladder logic programs.

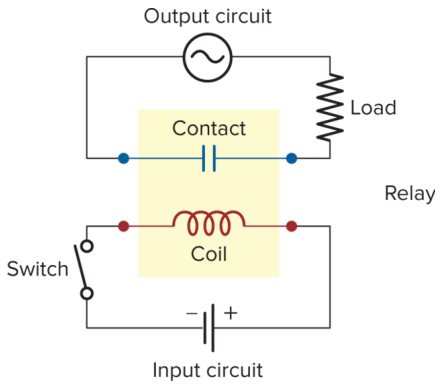


Figure 6-1 Electromechanical control relay.

An electrical relay is a magnetic switch. It uses electromagnetism to switch contacts. A relay will usually have only one coil but may have any number of different contacts. Figure 6-2 illustrates the operation of a typical control relay. With no current flow through the coil (de-energized), the armature is held away from the core of the coil by spring tension. When the coil is energized, it produces an electromagnetic field. Action of this field, in turn, causes the physical movement of the armature. Movement of the armature causes the contact points of the relay to open or close. The coil and contacts are insulated from each other; therefore, under normal conditions, no electric circuit will exist between them.

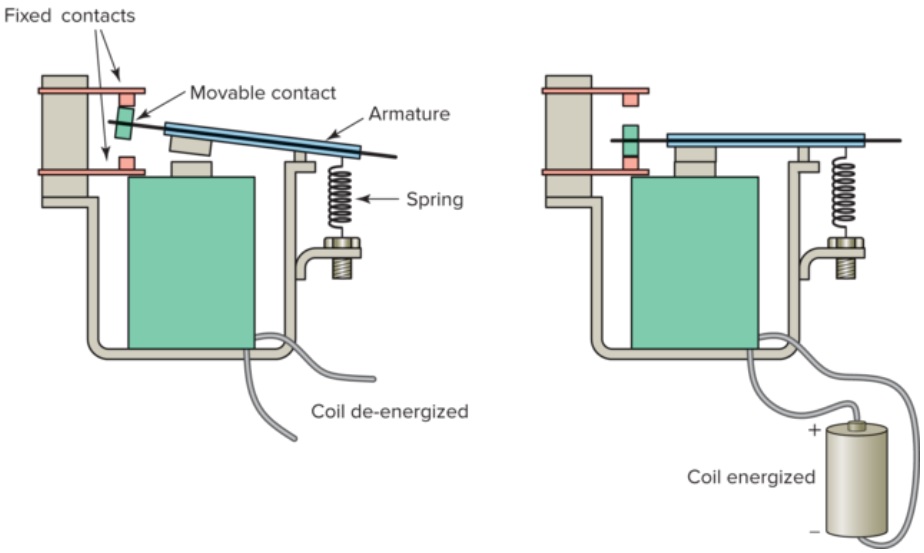


Figure 6-2 Relay operation.

The symbol used to represent a control relay is shown in Figure 6-3. The contacts are represented by a pair of short parallel lines and are identified with the coil by means of the letters. The letter M frequently indicates a motor starter, while CR is used for control relays. ***Normally open (NO) contacts*** are defined as those contacts that are open when no current flows through the coil but that *close* as soon as the coil conducts a current or is energized. ***Normally closed (NC) contacts*** ***are closed*** when the coil is de-energized and open when the coil is energized. Each contact is usually drawn as it would appear with the coil de-energized.

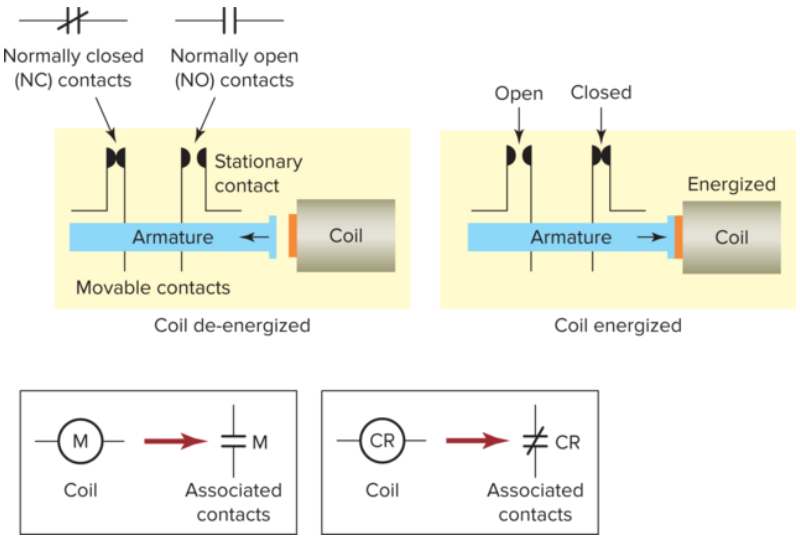


Figure 6-3 Relay normally open and normally closed contacts.

A typical control relay used to control two pilot lights is shown in Figure 6-4. The operation of the circuit can be summarized as follows:

- With the switch open, coil CR is de-energized.

- The circuit to the green pilot light is completed through the normally closed contact, so this light will be on.
- At the same time, the circuit to the red pilot light is opened through the normally open contact, so this light will be off.
- With the switch closed, the coil is energized.
- The normally open contact closes to switch the red pilot light on.
- At the same time, the normally closed contact opens to switch the green pilot light off.

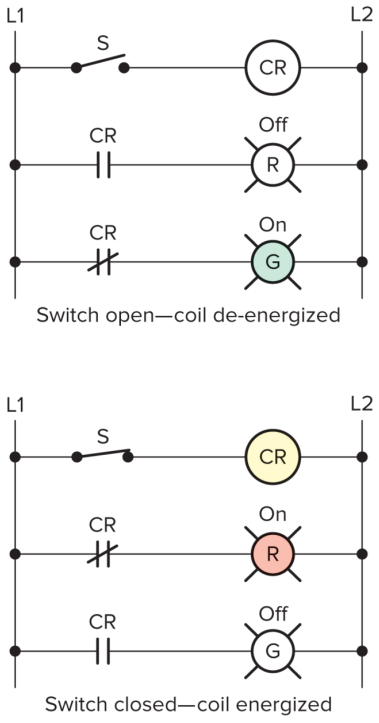


Figure 6-4 Control relay used to control two pilot lights.

Control relay coils and contacts have separate ratings. Coils are rated for the type of operating current (DC or AC) and normal operating voltage. Contacts are rated in terms of the maximum amount of current the contacts are capable of handling at a specified voltage level and type (AC or DC). Control relay contacts generally are not designed to carry heavy currents or high voltages. The contacts are usually rated between 5 and 10 Amp, with the most common rating for the coil voltage being 120 VAC.

OceanofPDF.com

6.2 Contactors

A **contactor** is a special type of relay designed to handle heavy power loads that are beyond the capability of control relays. **Figure 6-5** shows a three-pole magnetic contactor. Unlike relays, contactors are designed to make and break higher powered circuits without being damaged. Such loads include lights, heaters, transformers, capacitors, and electric motors for which overload protection is provided separately or not required.

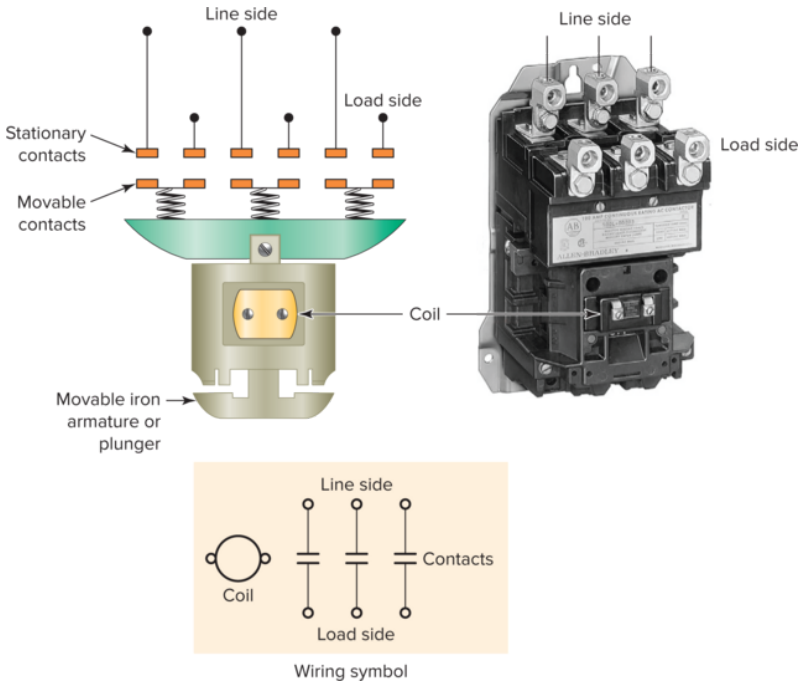


Figure 6-5 Three-pole magnetic contactor.

Source: Photo courtesy Rockwell Automation, **www.rockwellautomation.com.**

Programmable controllers normally have an output capacity capable of operating a contactor coil, but not that needed to operate heavy power loads directly. **Figure 6-6** illustrates the application of a PLC used in conjunction with a contactor to switch power on and off to a pump. The output module is connected in series with the coil to form a low-current switching circuit. The contacts of the contactor are connected in series with the pump motor to form a high-current switching circuit.

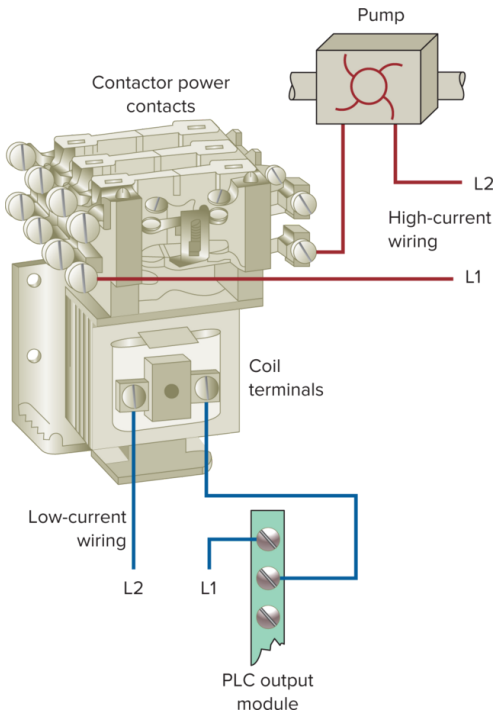


Figure 6-6 Contactor used in conjunction with a PLC output.

OceanofPDF.com

6.3 Motor Starters

A ***motor starter*** is designed to provide power to motors. The motor starter is made up of a contactor with an ***overload relay*** attached physically and electrically to it as illustrated in Figure 6-7. The function of the overload relay can be summarized as follows:

- Overload relays are designed to meet the special protective needs of motor control circuits.
- They allow harmless temporary overloads that occur when a motor starts.
- The overload relay will trip and disconnect power to the motor if an overload condition persists.
- Overload relays can be reset after the overload condition has been corrected.

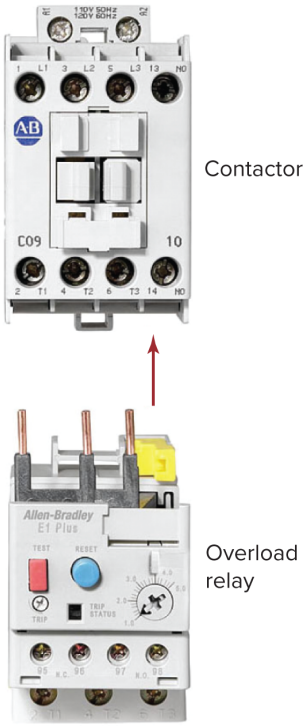


Figure 6-7 Motor starter is a contactor with an attached overload relay.

Source: Image Courtesy of Rockwell Automation, Inc.

Figure 6-8 shows the diagram for a typical three-phase, magnetic motor starter. The operation of the circuit can be summarized as follows:

- When the START button is pressed coil M is energized closing all normally open M contacts.
- The M contacts in series with the motor close to complete the current path to the motor. These contacts are part of the ***power*** circuit and must be designed to handle the full load current of the motor.

- Control contact M (across START button) closes to seal in the coil circuit when the START button is released. This contact is part of the ***control*** circuit and, as such, is only required to handle the small amount of current needed to energize the coil.
- An overload (OL) relay is provided to protect the motor against current overloads. The normally closed relay contact OL opens automatically when an overload current is sensed to de-energize the M coil and stop the motor.

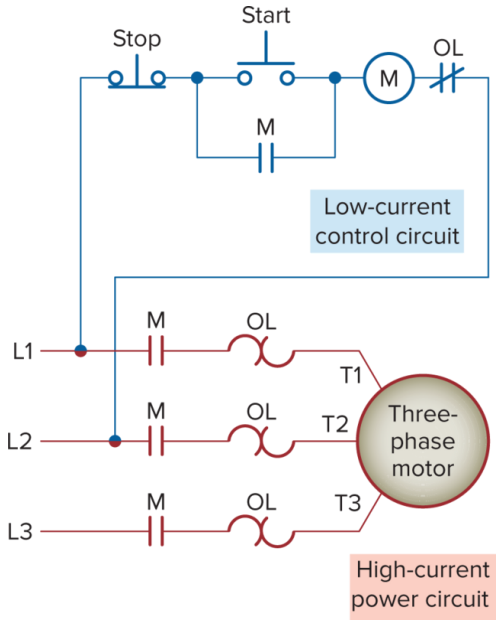


Figure 6-8 Three-phase magnetic motor starter.

Motor starters are available in various standard National Electric Manufacturers Association (NEMA) sizes and ratings. When a PLC needs to control a large motor, it must work in conjunction with a starter as illustrated in **Figure 6-9**. The power requirements for the starter coil must be within the power rating of the output module of the PLC. Note that the control logic is determined and executed by the program within the PLC and not by the hardwired arrangement of the input control devices.

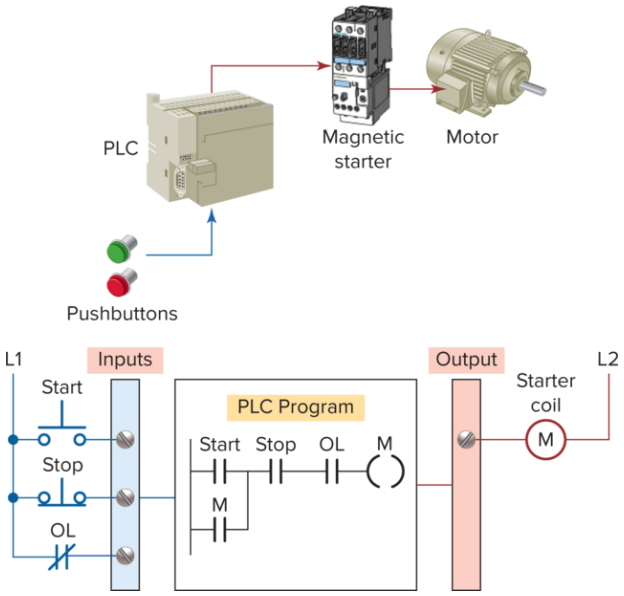


Figure 6-9 PLC control of a motor.

OceanofPDF.com

6.4 Manually Operated Switches

Manually operated switches are controlled by hand. These include toggle switches, pushbutton switches, knife switches, and selector switches.

Pushbutton switches are the most common form of manual control. A pushbutton operates by opening or closing contacts when pressed. **Figure 6-10** shows commonly used types of pushbutton switches, which include:

- *Normally open (NO) pushbutton,* which makes a circuit when it is pressed and returns to its open position when the button is released.
- *Normally closed (NC) pushbutton,* which opens the circuit when it is pressed and returns to the closed position when the button is released.
- *Break-before-make pushbutton* in which the top section contacts are NC and the bottom section contacts are NO. When the button is pressed, the top contacts open before the bottom contacts are closed.

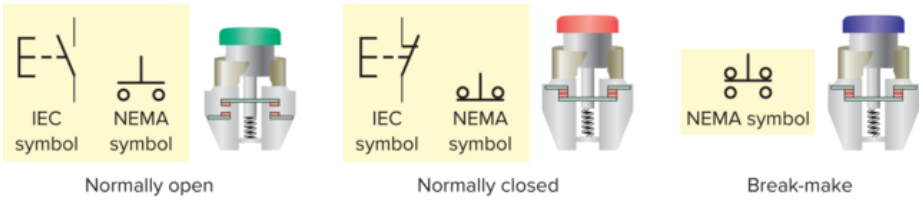


Figure 6-10 Commonly used types of pushbutton switches.

The ***selector switch*** is another common manually operated switch. The main difference between a pushbutton and selector switch is the operator mechanism. A selector switch operator is rotated (instead of pushed) to open and close contacts of the attached contact block. **Figure 6-11** shows a three-position selector switch. Switch positions are established by turning the operator knob right or left. Selector switches may have two or more selector positions, with either maintained contact position or spring return to give momentary contact operation.



Position	Contacts	
	A	B
1	X	
2		
3		X

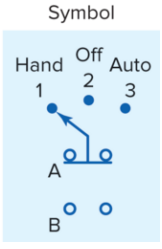


Figure 6-11 Three-position selector switch.

Source: Photo courtesy Rockwell Automation, **www.rockwellautomation.com.**

Dual in-line package (DIP) switches are small switch assemblies designed for mounting on printed circuit board modules (**Figure 6-12**). The pins or terminals on the bottom of the DIP switch are the same size and spacing as an integrated circuit (IC) chip. The individual switches may be of the toggle, rocker, or slide kind. DIP switches use binary (on/off) settings to set the parameters for a particular module. For example, the input voltage range on a particular input module may be selected by means of DIP switches located on the back of the module.

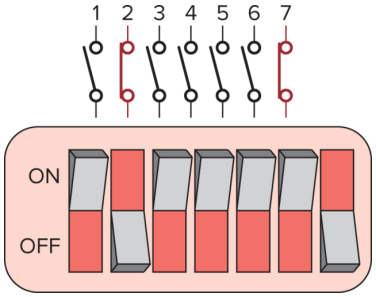


Figure 6-12 DIP switch.

OceanofPDF.com

6.5 Mechanically Operated Switches

A ***mechanically operated switch*** is controlled automatically by factors such as pressure, position, or temperature. The ***limit switch,*** shown in Figure 6-13, is a very common industrial control device. Limit switches are designed to operate only when a predetermined limit is reached, and they are usually actuated by contact with an object such as a cam. These devices take the place of a human operator. They are often used in the control circuits of machine processes to govern the starting, stopping, or reversal of motors.

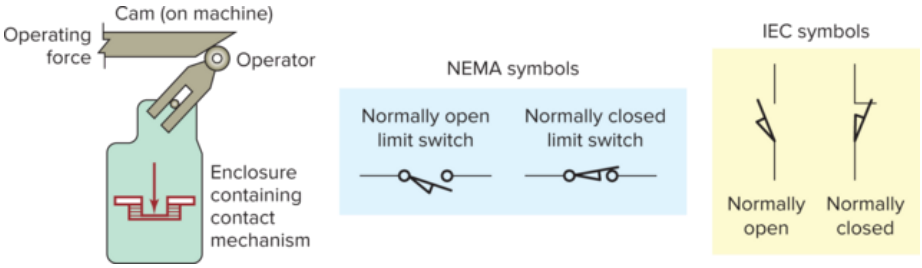


Figure 6-13 Mechanically operated limit switch.

The ***temperature switch,*** or ***thermostat*** (Figure 6-14), is used to sense temperature changes. Although there are many types available, they are all actuated by some specific environmental temperature change. Temperature switches open or close when a designated temperature is reached. Industrial applications for these devices include maintaining the desired temperature range of air, gases, liquids, or solids.

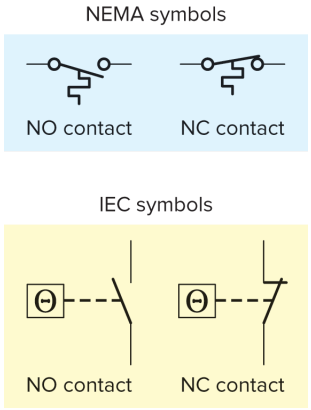


Figure 6-14 Temperature switch.

Pressure switches (Figure 6-15) are used to control the pressure of liquids and gases. Although many different types are available, they are all basically designed to actuate (open or close) their contacts when a specified pressure is reached. Pressure switches can be pneumatically (air) or

hydraulically (liquid) operated switches. Generally, bellows or a diaphragm presses up against a small microswitch and causes it to open or close.

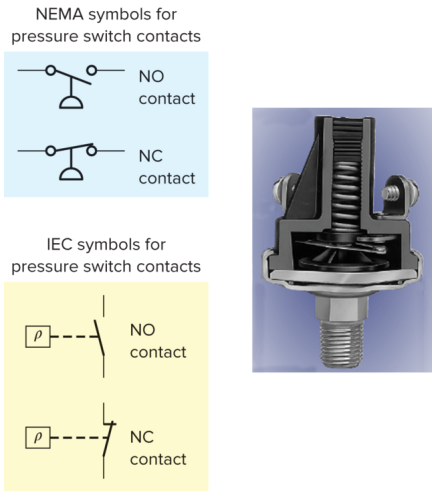


Figure 6-15 Pressure switch.

Source: Photo courtesy Honeywell, **www.honeywell.com**.

Level switches (Figure 6-16) are used to sense liquid levels in vessels and provide automatic control for motors that transfer liquids from sumps or into tanks. They are also used to open or close piping solenoid valves to control fluids. The float switch shown in Figure 6-16 is a type of level switch. This switch is weighted so that as the liquid rises the switch floats and turns upside down, actuating its internal contacts.

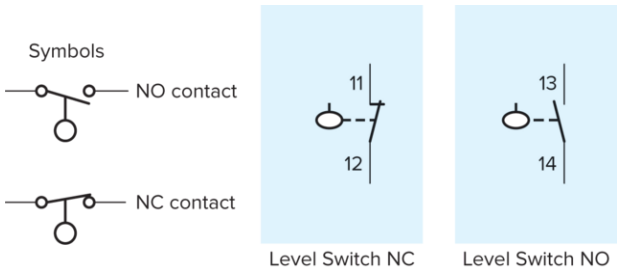


Figure 6-16 Float type level switch.

OceanofPDF.com

6.6 Sensors

Sensors are used for detecting, and often measuring, the magnitude of something. They convert mechanical, magnetic, thermal, optical, and chemical variations into electric voltages and currents. Sensors are usually categorized by what they measure, and they play an important role in modern manufacturing process control.

Proximity Sensor

Pilot control devices have limited current handling capacity and are used to control current to a secondary device, such as a contactor coil, which in turn can be used to switch heavier load currents. **Proximity sensors** or switches (**Figure 6-17**) are pilot devices that detect the presence of an object (usually called the target) *without physical contact.* These solid-state electronic devices are completely encapsulated to protect against excessive vibration, liquids, chemicals, and corrosive agents found in the industrial environment. Proximity sensors are used when:

- The object being detected is too small, lightweight, or soft to operate a mechanical switch.
- Rapid response and high switching rates are required, as in counting or ejection control applications.
- An object has to be sensed through nonmetallic barriers such as glass, plastic, and paper cartons.
- Hostile environments prevent proper operation of mechanical switches and demand improved sealing properties.
- Long life and reliable service are required.
- A fast electronic control system requires a bounce-free input signal.

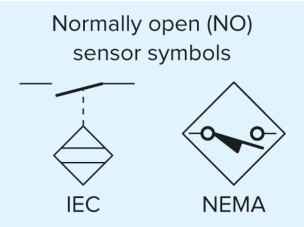


Figure 6-17 Proximity sensor.

Proximity sensors operate on different principles, depending on the type of matter being detected. When an application calls for noncontact metallic target sensing, an ***inductive-type proximity sensor*** is used. Inductive proximity sensors are used to detect both ferrous metals (containing iron) and nonferrous metals (such as copper, aluminum, and brass).

Inductive proximity sensors operate under the electrical principle of inductance, where a fluctuating current induces an electromotive force (emf) in a target object. The block diagram for an inductive proximity sensor is shown in **Figure 6-18** and its operation can be summarized as follows:

- The oscillator circuit generates a high-frequency electromagnetic field that radiates from the end of the sensor.
- When a metal object enters the field, eddy currents are induced in the surface of the object.
- The eddy currents on the object absorb some of the radiated energy from the sensor, resulting in a loss of energy and change of strength of the oscillator.
- The sensor's detection circuit monitors the oscillator's strength and triggers a solid-state output at a specific level.
- Once the metal object leaves the sensing area, the oscillator returns to its initial value.

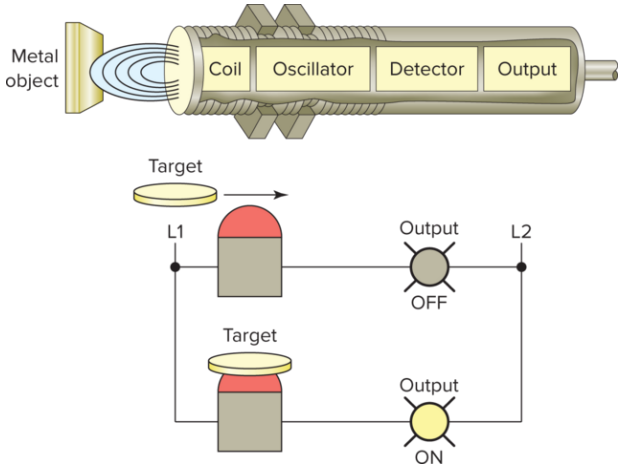


Figure 6-18 Inductive proximity sensor.

Most sensor applications operate either at 24V DC or at 120V AC. The method of connecting a proximity sensor varies with the type of sensor and its application. **Figure 6-19** shows a typical three-wire DC sensor connection. The three-wire DC proximity sensor has the positive and negative line leads connected directly to it. When the sensor is actuated, the circuit will connect the signal wire to the positive side of the line if operating normally open. If operating normally closed, the circuit will disconnect the signal wire from the positive side of the line.

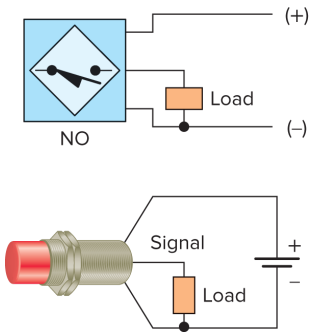


Figure 6-19 Typical three-wire DC sensor connection.

Figure 6-20 shows a typical two-wire proximity sensor connection intended to be connected in series with the load. They are manufactured for either AC or DC supply voltages. In the off state, enough current must flow through the circuit to keep the sensor active. This off state current is called leakage current and typically may range from 1 to 2 mA. When the switch is actuated, it will conduct the normal load circuit current.

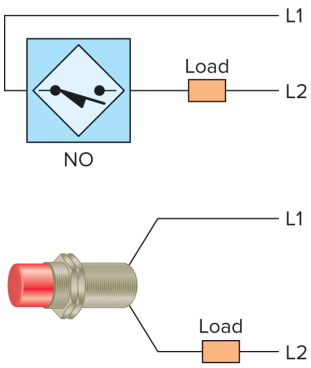


Figure 6-20 Typical two-wire proximity sensor connection.

Figure 6-21 shows the proximity sensor sensing range. Hysteresis is the distance between the operating point when the target approaches the proximity sensor face and the release point when the target is moving away from the sensor face. The object must be closer to turn the sensor on rather than to turn it off. If the target is moving toward the sensor, it will have to move to a closer point. Once the sensor turns on, it will remain on until the target moves to the release point. Hysteresis is needed to keep proximity sensors from chattering when subjected to shock and vibration, slow-moving targets, or minor disturbances such as electrical noise and temperature drift. Most proximity sensors come equipped with an LED status indicator to verify the output switching action.

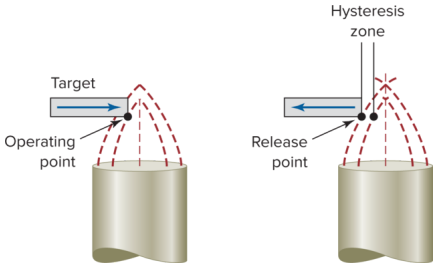


Figure 6-21 Proximity sensor sensing range.

As a result of solid-state switching of the output, a small leakage current flows through the sensor even when the output is turned off. Similarly, when the sensor is on, a small voltage drop is lost across its output terminals. To operate properly, a proximity sensor should be powered continuously.

Figure 6-22 illustrates the use of a bleeder resistor connected to allow enough current for the sensor to operate but not enough to turn on the input of the PLC.

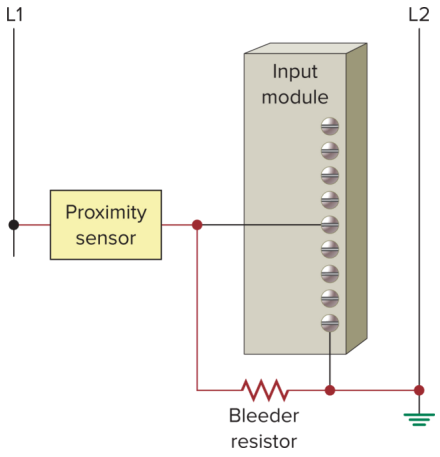


Figure 6-22 Bleeder resistor connected to continuously power a proximity sensor.

Capacitive proximity sensors are similar to inductive proximity sensors. The main differences between the two types are that capacitive proximity sensors produce an electrostatic field instead of an electromagnetic field and are actuated by both conductive and nonconductive materials.

Figure 6-23 illustrates the operation of a capacitive sensor. A capacitive sensor contains a high-frequency oscillator along with a sensing surface formed by two metal electrodes. When the target nears the sensing surface, it enters the electrostatic field of the electrodes and changes the capacitance of the oscillator. As a result, the oscillator circuit begins oscillating and changes the output state of the sensor when it reaches a certain amplitude. As the target moves away from the sensor, the oscillator's amplitude decreases, switching the sensor back to its original state.

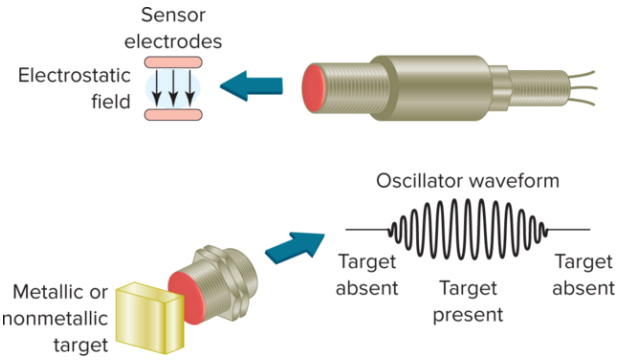


Figure 6-23 Capacitive proximity sensor.

Capacitive proximity sensors will sense metal objects as well as nonmetallic materials such as paper, glass, liquids, and cloth. They typically have a short sensing range of about 1 inch, regardless of the type of material being sensed. The larger the dielectric constant of a target, the easier it is for the capacitive sensor to detect. This makes possible the detection of materials inside nonmetallic containers as illustrated in Figure 6-24. In this example, the liquid has a much higher dielectric

constant than the cardboard container, which gives the sensor the ability to see through the container and detect the liquid. In the process shown, detected empty containers are automatically diverted via the push rod.

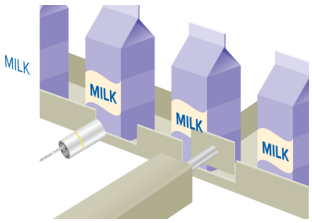


Figure 6-24 Capacitive proximity sensor liquid detection.

Inductive proximity switches may be actuated only by a metal and are insensitive to humidity, dust, dirt, and the like. Capacitive proximity switches, however, can be actuated by any dirt in their environment. For general applications, the capacitive proximity switches are not really an alternative but a supplement to the inductive proximity switches. They are a supplement when there is no metal available for the actuation (e.g., for woodworking machines and for determining the exact level of liquids or powders).

Magnetic Reed Switch

A ***magnetic reed switch*** is composed of two flat contact tabs that are hermetically sealed (airtight) in a glass tube filled with protective gas, as illustrated in **Figure 6-25**. When a magnetic force is generated parallel to the reed switch, the reeds become flux carriers in the magnetic circuit. The overlapping ends of the reeds become opposite magnetic poles, which attract each other. If the magnetic force between the poles is strong enough to overcome the restoring force of the reeds, the reeds will be drawn together to actuate the switch. Because the contacts are sealed, they are unaffected by dust, humidity, and fumes; thus, their life expectancy is quite high.

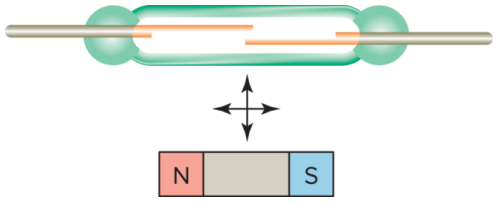


Figure 6-25 Magnetic reed switch.

One practical application for a magnetic reed switch is its use in a float switch, illustrated in **Figure 6-26**. The reed switch opens or closes a circuit as the level of a liquid rises or falls. The switch assembly is made up of a permanent magnet installed within the movable float arm and a magnetic reed switch installed within the fixed housing. The movement of the float, due to the changing liquid level, will cause the reed switch to open or close a circuit at a particular level.

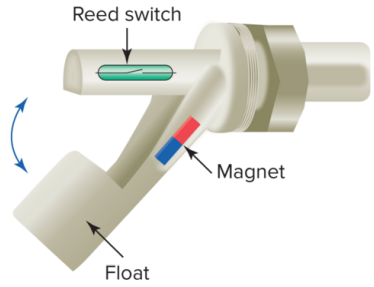


Figure 6-26 Magnetic reed float switch.

Light Sensors

The photovoltaic cell and the photoconductive cell, illustrated in Figure 6-27, are two examples of light sensors. The ***photovoltaic* or *solar cell*** reacts to light by converting the light energy directly into electric energy. The ***photoconductive cell*** (also called a *photoresistive cell*) reacts to light by change in the resistance of the cell.

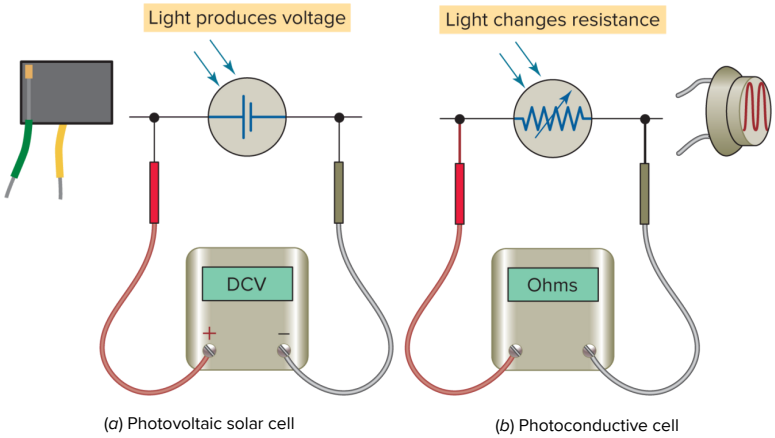


Figure 6-27 Photovoltaic and photoconductive light cells.

A ***photoelectric sensor*** is an optical control device that operates by detecting a visible or invisible beam of light and responding to a change in the received light intensity. Photoelectric sensors are composed of two basic components: a transmitter (light source) and a receiver (sensor), as shown in Figure 6-28. These two components may or may not be housed in separate units. The basic operation of a photoelectric sensor can be summarized as follows:

- The transmitter contains a light source, usually an LED along with an oscillator.
- The oscillator modulates or turns the LED on and off at a high rate of speed.
- The transmitter sends this modulated light beam to the receiver.
- The receiver decodes the light beam and switches the output device, which interfaces with the load.

- The receiver is tuned to its emitter's modulation frequency and will amplify only the light signal that pulses at the specific frequency.
- Most sensors allow adjustment of how much light will cause the output of the sensor to change state.
- Response time is related to the frequency of the light pulses. Response times may become important when an application calls for the detection of very small objects, objects moving at a high rate of speed, or both.

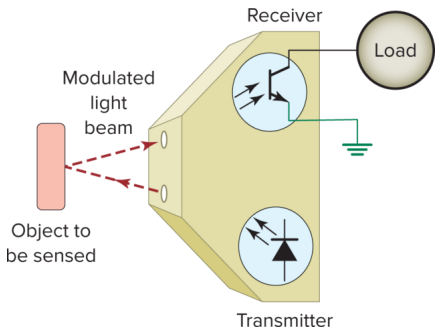


Figure 6-28 Photoelectric sensor.

The **scan technique** refers to the method used by photoelectric sensors to detect an object. The *through-beam* scan technique (also called direct scan) places the transmitter and receiver in direct line with each other, as illustrated in **Figure 6-29**. Because the light beam travels in only one direction, through-beam scanning provides long-range sensing. Quite often, a garage door opener has a through-beam photoelectric sensor mounted near the floor, across the width of the door. For this application the sensor senses that nothing is in the path of the door when it is closing.

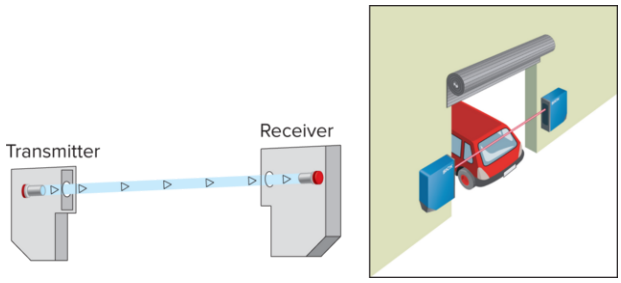


Figure 6-29 Through-beam scan.

In a *retroreflective scan,* the transmitter and receiver are housed in the same enclosure. This arrangement requires the use of a separate reflector or reflective tape mounted across from the sensor to return light back to the receiver. The retroreflective scan is designed to respond to objects that interrupt the beam normally maintained between the transmitter and receiver, as illustrated in **Figure 6-30**. In contrast to a through-beam application, retroreflective sensors are used for medium-range applications.

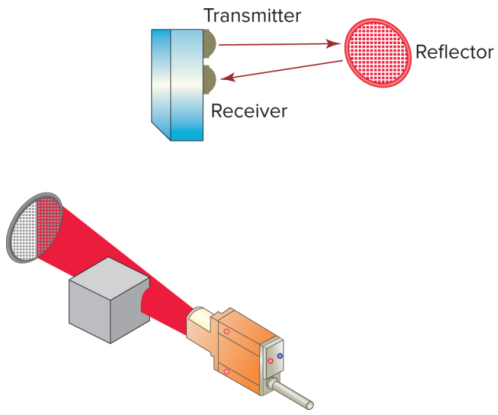


Figure 6-30 Retroreflective scan.

Fiber optics is not a scan technique, but another method for transmitting light. ***Fiber optic sensors*** use a flexible cable containing tiny fibers that channel light from emitter to receiver, as illustrated in **Figure 6-31**. Fiber optic sensor systems are completely immune to all forms of electrical interference. The fact that an optical fiber does not contain any moving parts and carries only light means that there is no possibility of a spark. This means that it can be safely used even in the most hazardous sensing environments such as a refinery for producing gases, grain bins, mining, pharmaceutical manufacturing, and chemical processing.

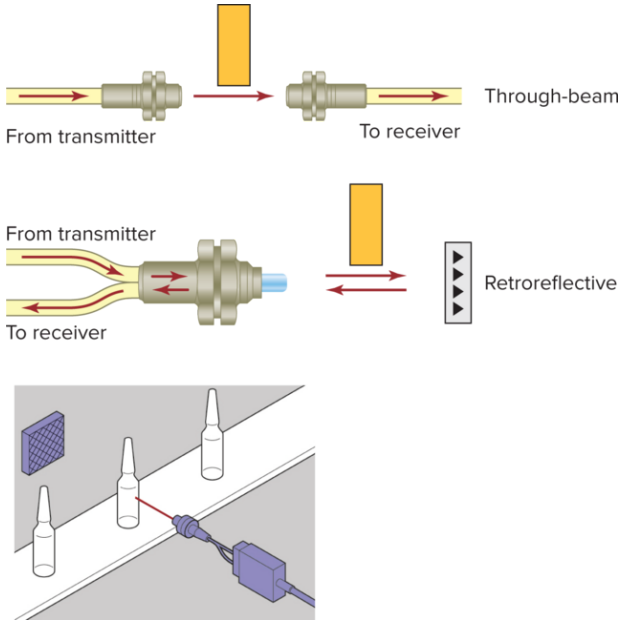


Figure 6-31 Fiber optic sensors.

Bar code technology is widely implemented in industry to enter data quickly and accurately. ***Bar code scanners*** are the eyes of the data collection system. A light source within the scanner illuminates the bar code symbol; those bars absorb light, and spaces reflect light. A photodetector collects this light in the form of an electronic-signal pattern representing the printed symbol. The decoder receives the signal from the scanner and converts these data into the character data representation of the symbol's

code. Figure 6-32 illustrates a typical PLC application which involves a bar code module reading the bar code on boxes as they move along a conveyor line. The PLC is then used to divert the boxes to the appropriate product lines according to the data read from the bar code.

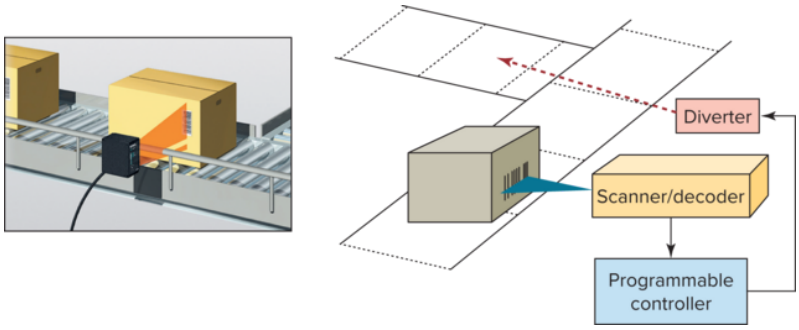


Figure 6-32 PLC bar code application.

Source: Courtesy Keyence Canada, Inc.

Ultrasonic Sensors

An ***ultrasonic sensor*** operates by sending high-frequency sound waves toward the target and measuring the time it takes for the pulses to bounce back. The time taken for this echo to return to the sensor is directly proportional to the distance or height of the object because sound has a constant velocity.

Figure 6-33 illustrates a practical application in which the returning echo signal is electronically converted to a 4- to 20-mA output, which supplies a monitored level to external control devices. The operation of this process can be summarized as follows:

- The 4 to 20-mA represents the sensor's measurement span.
- The 4-mA set point is typically placed near the bottom of the empty tank, or the greatest measurement distance from the sensor.
- The 20-mA set point is typically placed near the top of the full tank, or the shortest measurement distance from the sensor.
- The sensor will proportionately generate a 4-mA signal when the tank is empty and a 20-mA signal when the tank is full.
- Ultrasonic sensors can detect solids, fluids, granular objects, and textiles. In addition, they enable the detection of different objects irrespective of color and transparency and therefore are ideal for monitoring transparent objects.

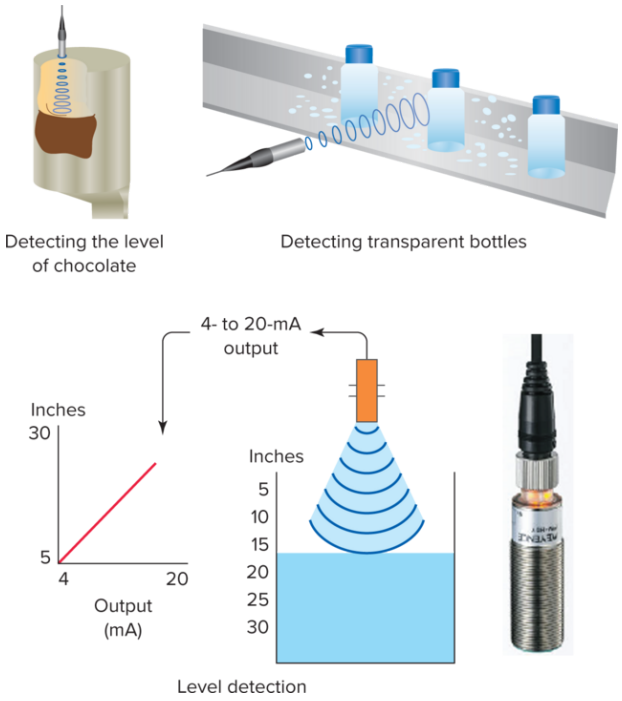


Figure 6-33 Ultrasonic sensor.

Source: Courtesy Keyence Corporation

Strain/Weight Sensors

A ***strain gauge*** converts a mechanical strain into an electric signal. Strain gauges are based on the principle that the resistance of a conductor varies with length and cross-sectional area. The force applied to the gauge causes the gauge to bend. This bending action also distorts the physical size of the gauge, which in turn changes its *resistance.* This resistance change is fed to a bridge circuit that detects small changes in the gauge's resistance. *Strain gauge load cells* are usually made with steel and sensitive strain gauges. As the load cell is loaded, the metal elongates or compresses very slightly. The strain gauge detects this movement and translates it to a varying voltage signal. Many sizes and shapes of load cells are available, and they range in sensitivity from grams to millions of pounds. Strain gauge–based load cells are used extensively for industrial weighing applications similar to the one illustrated in **Figure 6-34**.

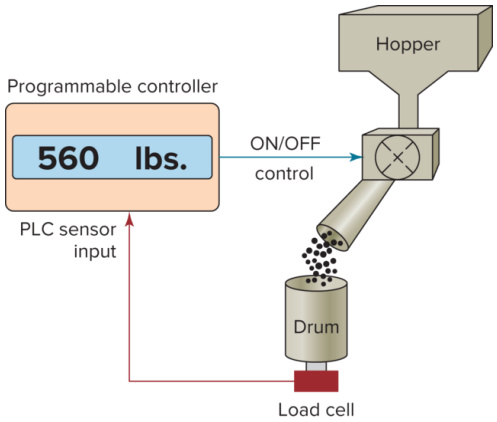


Figure 6-34 Strain gauge load cell.

Temperature Sensors

Thermocouples operate on the principle that when two dissimilar metals are joined, a predictable DC voltage will be generated that relates to the difference in temperature between the hot junction and the cold junction (Figure 6-35). The hot junction (measuring junction) is the joined end of a thermocouple that is exposed to the process where the temperature measurement is desired. The cold junction (reference junction) is the end of a thermocouple that is kept at a constant temperature to provide a reference point. For example, a K-type thermocouple, when heated to a temperature of 300°C at the hot junction, will produce 12.2 mV at the cold junction. Because of their ruggedness and wide temperature range, thermocouples are used in industry to monitor and control oven and furnace temperatures.

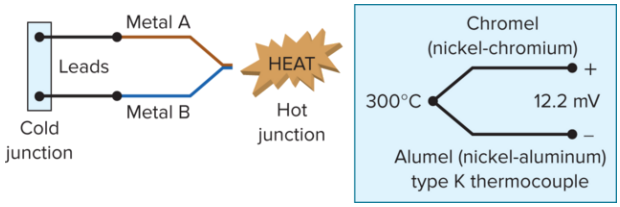


Figure 6-35 Thermocouple temperature sensor.

Resistance temperature detectors (RTDs) are wirewound temperature-sensing devices that operate on the principle of the positive temperature coefficient (PTC) of metals. That means the electrical resistance of metals is directly proportional to temperature. The hotter they become, the larger or higher the value of their electrical resistance. This proportional variation is precise and repeatable, and therefore allows the consistent measurement of temperature through electrical resistance detection. Platinum is the material most often used in RTDs because of its superiority regarding temperature limit, linearity, and stability.

RTDs are among the most precise temperature sensors available and are normally found encapsulated in probes for external temperature sensing and measurement or enclosed inside devices where they measure temperature as a part of the device's function. **Figure 6-36** illustrates how an RTD is used as part of a temperature control system. A controller uses the signal from the RTD sensor to monitor the temperature of the liquid in the tank and thereby control heating and cooling lines.

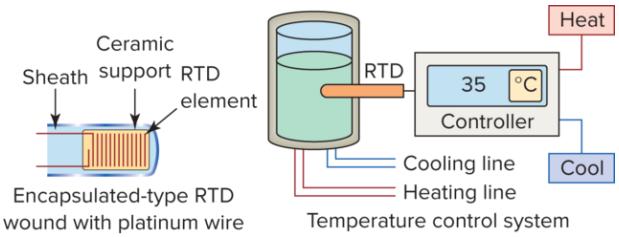


Figure 6-36 Resistance temperature detector (RTD).

Flow Measurement

Many industrial processes depend on accurate measurement of fluid flow. Although there are several ways to measure fluid flow, the usual approach is to convert the kinetic energy that the fluid has into some other measurable form.

Page 103

Turbine-type flowmeters are a popular means of measurement and control of liquid products in industrial, chemical, and petroleum operations. Turbine flowmeters, like windmills, utilize their angular velocity (rotation speed) to indicate the flow velocity. The operation of a turbine flowmeter is illustrated in **Figure 6-37**. Its basic construction consists of a bladed turbine rotor installed in a flow tube. The bladed rotor rotates on its axis in proportion to the rate of the liquid flow through the tube. A magnetic pickup sensor is positioned as close to the rotor as practical. Fluid passing through the flow tube causes the rotor to rotate, which generates pulses in the pickup coil. The frequency of the pulses is then transmitted to readout electronics and displayed as gallons per minute.

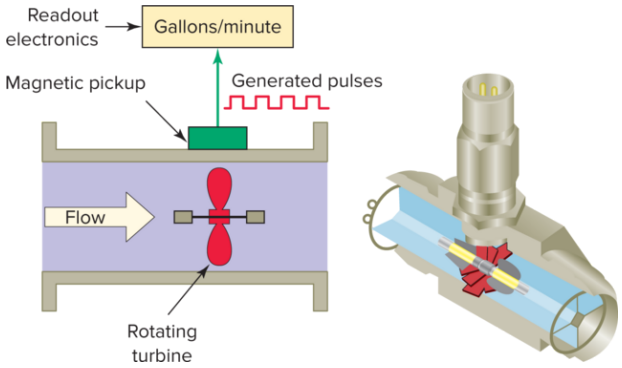


Figure 6-37 Turbine type flowmeter.

Velocity and Position Sensors

A ***Tachometer generator*** is a device that provides a means of converting rotational speed into an analog voltage signal that can be used for motor speed indication and control applications. A tachometer generator is a small AC or DC generator that develops an output voltage (proportional to its rpm) whose phase or polarity depends on the rotor's direction of rotation. The DC tachometer generator usually has permanent magnetic field excitation. The AC tachometer generator field is excited by a constant AC supply. In either case, the rotor of the tachometer is mechanically connected, directly or indirectly, to the load.

Figure 6-38 illustrates motor speed control applications in which a tachometer generator is used to provide a feedback voltage to the motor controller that is proportional to motor speed. The control motor and tachometer generator may be contained in the same or separate housings.

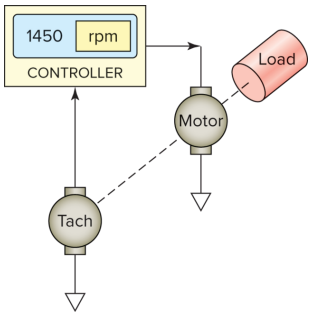


Figure 6-38 Tachometer generator feedback.

An ***encoder*** is used to convert linear or rotary motion into a binary digital signal. Encoders are used in applications where positions have to be precisely determined. The optical encoder illustrated in **Figure 6-39** uses a light source shining on an optical disk with lines or slots that interrupt the beam of light to an optical sensor. An electronic circuit counts the interruptions of the beam and generates the encoder's digital output pulses.

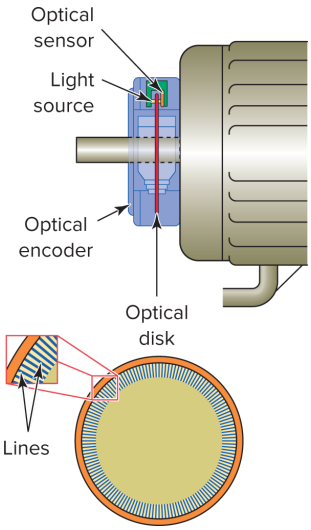


Figure 6-39 Optical encoder.

OceanofPDF.com

6.7 Output Control Devices

A variety of output control devices can be operated by the PLC output to control traditional industrial processes. These devices include pilot lights, control relays, motor starters, alarms, heaters, solenoids, solenoid valves, small motors, and horns. Similar electrical symbols are used to represent these devices both on relay schematics and PLC output connection diagrams. **Figure 6-40** shows common electrical symbols used for various output devices. Although these symbols are generally acceptable, some differences among manufacturers do exist.

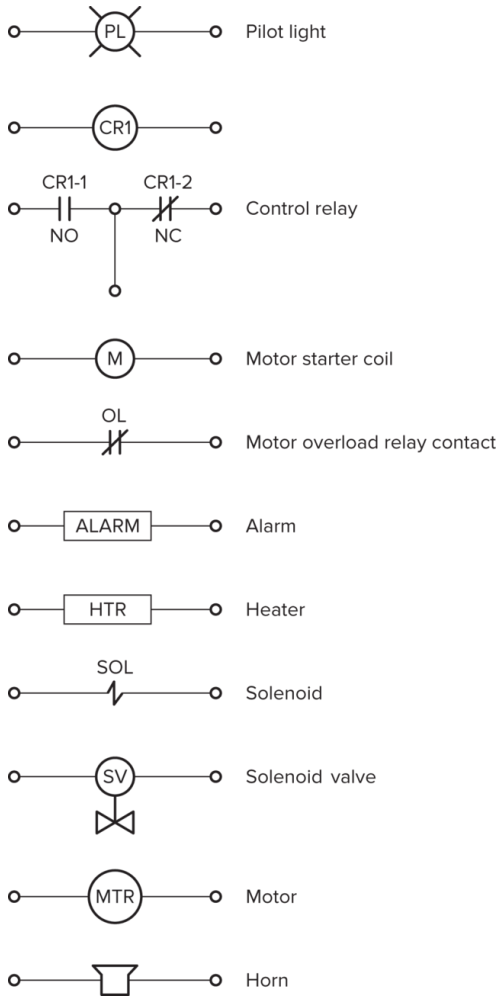


Figure 6-40 Symbols for output control devices.

An ***actuator,*** in the electrical sense, is any device that converts an electrical signal into mechanical movement. An electromechanical solenoid is an actuator that uses electrical energy to magnetically cause mechanical control action. A **solenoid** consists of a coil, frame, and plunger (or armature, as it is

sometimes called). Figure 6-41 shows the basic construction and operation of a solenoid. Its operation can be summarized as follows:

- The coil and frame form the fixed part.
- When the coil is energized, it produces a magnetic field that attracts the plunger, pulling it into the frame and thus creating mechanical motion.
- When the coil is de-energized the plunger returns to its normal position through gravity or assistance from spring assemblies within the solenoid.
- The frame and plunger of an AC-operated solenoid are constructed with laminated pieces instead of a solid piece of iron to limit eddy currents induced by the magnetic field.

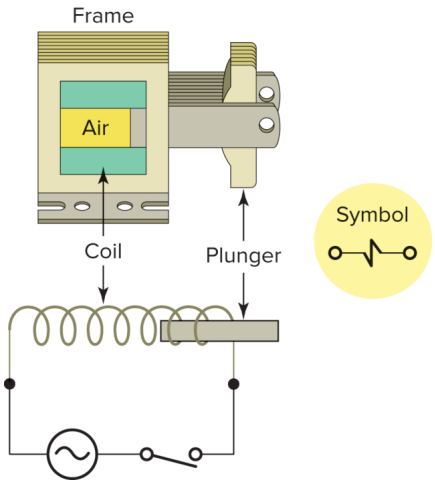


Figure 6-41 Solenoid construction and operation.

Solenoid valves are electromechanical devices that work by passing an electrical current through a solenoid, thereby changing the state of the valve. Normally, there is a mechanical element, which is often a spring, that holds the valve in its default position. A solenoid valve is a combination of a solenoid coil operator and valve, which controls the flow of liquids, gases, steam, and other media. When electrically energized, they open, shut off, or direct the flow of media.

Figure 6-42 illustrates the construction and principle of operation of a typical fluid solenoid valve. Its operation can be summarized as follows:

- The valve body contains an orifice in which a disk or plug is positioned to restrict or allow flow.

- Flow through the orifice is either restricted or allowed depending on whether the solenoid coil is energized or de-energized.
- When the coil is energized, the core is drawn into the solenoid coil to open the valve.
- The spring returns the valve to its original closed position when the coil is de-energized.
- A valve must be installed with direction of flow in accordance with the arrow cast on the side of the valve body.

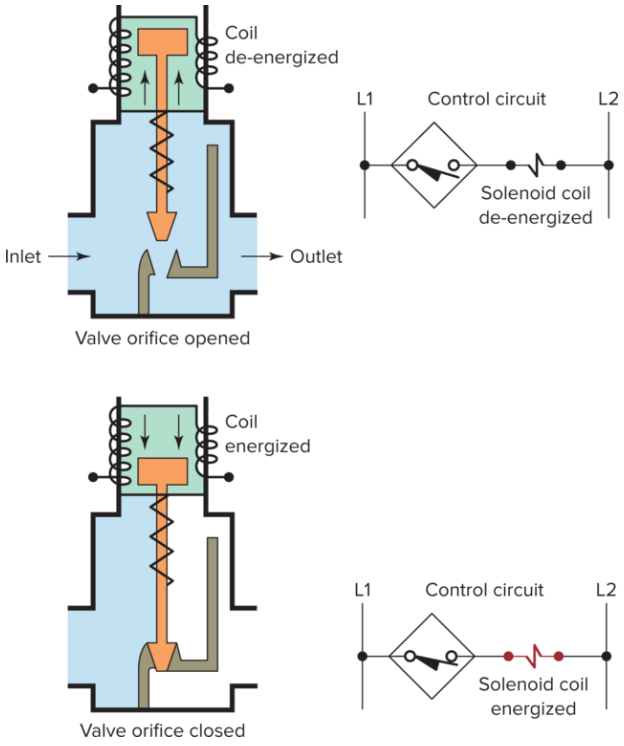


Figure 6-42 Solenoid valve construction and operation.

Stepper motors operate differently than standard motor types, which rotate continuously when voltage is applied to their terminals. The shaft of a stepper motor rotates in discrete increments when electrical command pulses are applied to it in the proper sequence. Every revolution is divided into a number of steps, and the motor must be sent a voltage pulse for each step. The amount of rotation is directly proportional to the number of pulses, and the speed of rotation is relative to the frequency of those pulses. A 1-degree-per-step motor will require 360 pulses to move through one revolution; the degrees per step are known as the ***resolution*****.** When stopped, a stepper motor inherently holds its position. Stepper systems are used most often in "open-loop" control systems, where the controller tells the motor only how many steps to move and how fast to move, but does not have any way of knowing what position the motor is at.

The movement created by each pulse is precise and repeatable, which is why stepper motors are so effective for load-positioning applications. Conversion of rotary to linear motion inside a linear actuator is accomplished through a threaded nut and lead screw. Generally, stepper motors produce less than 1 hp and are therefore frequently used in low-power position control applications. **Figure 6-43** shows typical rotary and linear stepper motor applications.

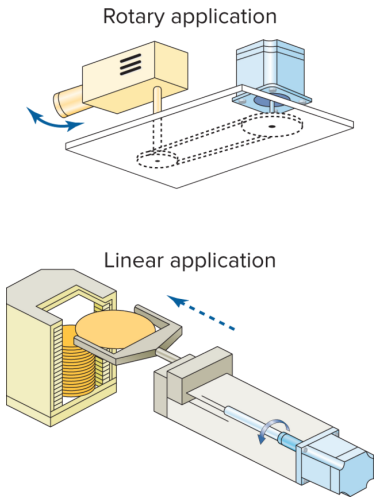


Figure 6-43 Typical stepper motor applications.

All ***servo motors*** operate in closed-loop mode, whereas most stepper motors operate in open-loop mode. Closed-loop and open-loop control schemes are illustrated in **Figure 6-44**. ***Open loop*** is control without feedback, for example, when the controller tells the stepper motor how many steps to move and how fast to move, but does not verify where the motor is. ***Closed-loop*** control compares speed or position feedback with the commanded speed or position and generates a modified command to make the error smaller. The error is the difference between the required speed or position and the actual speed or position.

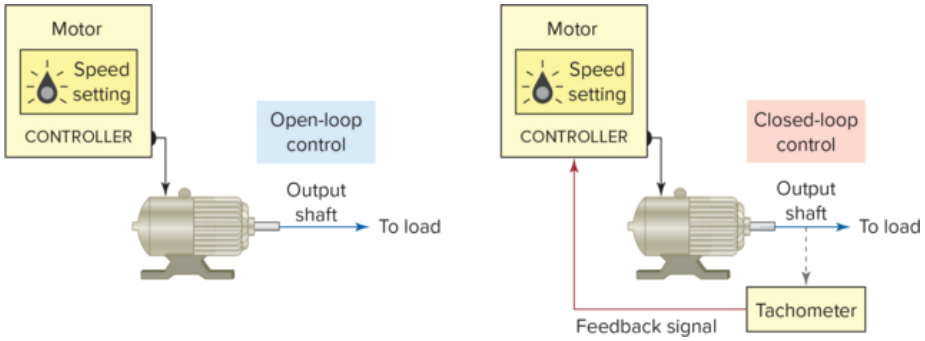


Figure 6-44 Open- and closed-loop motor control systems.

Figure 6-45 illustrates a closed-loop servo motor system. The motor controller directs operation of the servo motor by sending speed or position command signals to the amplifier, which drives the servo motor. A feedback device such as an encoder for position and a tachometer for speed are either incorporated within the servo motor or are remotely mounted, often on the load itself. These provide the servo motor's position and speed feedback information that the controller compares to its programmed motion profile and uses to alter its position or speed.

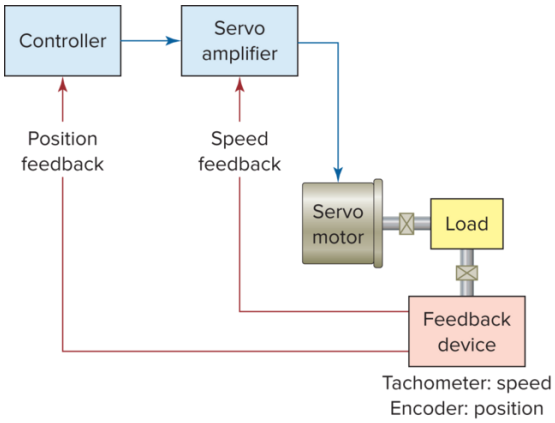


Figure 6-45 Closed-loop servo motor system.

OceanofPDF.com

6.8 Seal-In Circuits

Seal-in, or ***holding,*** circuits are very common in both relay logic and PLC logic. Essentially, a seal-in circuit is a method of maintaining current flow after a momentary switch has been pressed and released. In these types of circuits, the seal-in contact is usually in parallel with the momentary device.

The motor stop/start circuit shown in **Figure 6-46** is a typical example of a seal-in circuit. The hardwired circuit consists of a normally closed stop button in series with a normally open start button. The seal-in auxiliary contact of the starter is connected in parallel with the start button to keep the starter coil energized when the start button is released. When this circuit is programmed into a PLC, both the start and stop buttons are examined for a closed condition because both buttons must be closed to cause the motor starter to operate. Note that the motor starter auxiliary seal-in contact is used as an input device rather than a programmed contact reference. This is for safety purposes to provide positive feedback about the exact status of the motor.

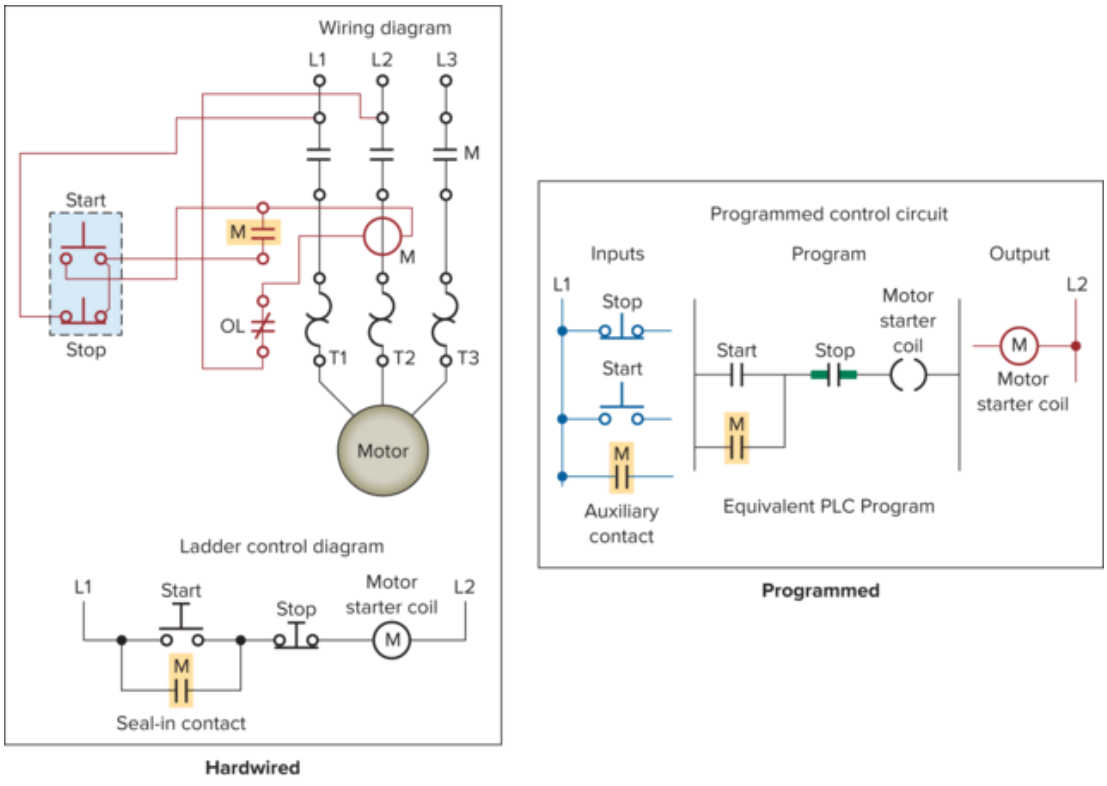


Figure 6-46 Hardwired control circuit and equivalent PLC program.

OceanofPDF.com

6.9 Electrical Interlocking Circuits

An electrical interlocking circuit is used to prevent a piece of equipment from operating under certain potentially hazardous or undesirable conditions. Figure 6-47 shows a three motor hardwired relay control circuit electrically interlocked to prevent the motors from accidently operating in an order other than their proper sequence. The interlocking feature of the circuit can be summarized as follows:

- Motor 1 has to be operating before Motor 2 can be started.
- The NO auxiliary interlocking contact M1-2 is used for this purpose.
- Motor 2 has to be operating before Motor 3 can be started.
- The NO auxiliary interlocking contact M2-2 is used for this purpose.

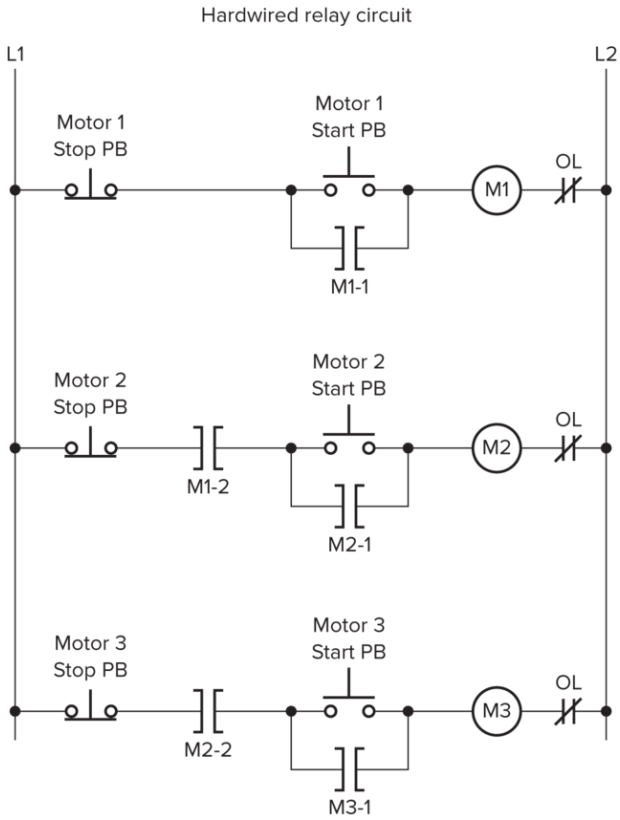


Figure 6-47 Sequential hardwired three motor relay control circuit.

Figure 6-48 shows a PLC program equivalent of the hardwired circuit.

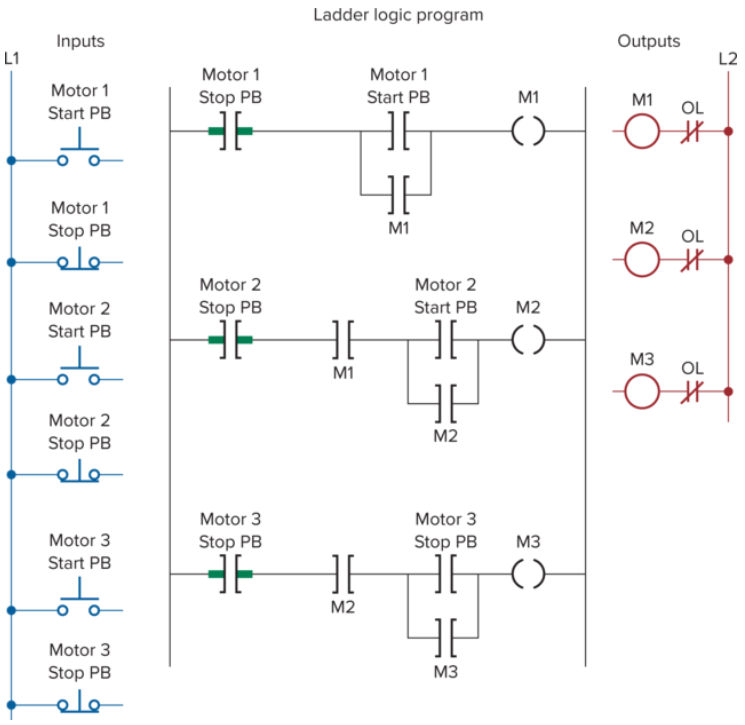


Figure 6-48 PLC program equivalent of the hardwired sequential motor control circuit.

Pushbutton interlocking is one of the methods of preventing two loads from being energized simultaneously. The hardwired pushbutton interlocking circuit of **Figure 6-49** is designed to prevent solenoids SOL-A and SOL-B from being energized at the same time. The interlocking feature of the circuit can be summarized as follows:

- Each pushbutton is equipped with a set of momentarily normally open (NO) and normally closed (NC) contacts mechanically connected together.
- The NC contact of SOL-A pushbutton is connected in series with the NO contact of SOL-B pushbutton.
- The NO contact of SOL-A pushbutton is connected in series with the NC contact of SOL-B pushbutton.
- When SOL-A pushbutton is pressed its NO contact completes the circuit to SOL-A and its NC contacts opens the current path to SOL-B.
- When SOL-B pushbutton is pressed its NO contact completes the circuit to SOL-B and its NC contacts opens the current path to SOL-A.
- When both buttons are pushed, neither solenoid will be energized.

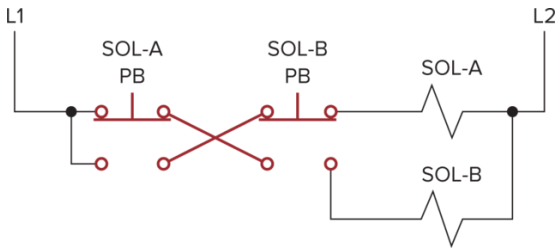


Figure 6-49 Hardwired pushbutton interlocking circuit.

Figure 6-50 shows a PLC program equivalent of the hardwired circuit implemented using two NO pushbuttons.

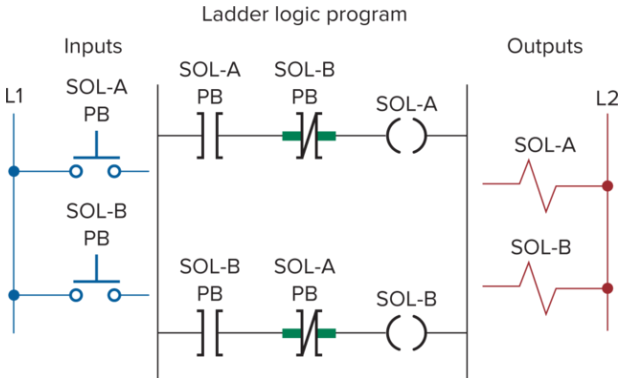


Figure 6-50 PLC program equivalent of the hardwired pushbutton interlocking circuit.

OceanofPDF.com

6.10 Latching Relays

Electromagnetic **latching relays** are designed to hold the relay closed after power has been removed from the coil. Latching relays are used where it is necessary for contacts to stay open and/or closed even though the coil is energized only momentarily. **Figure 6-51** shows a latching relay that uses two coils. The ***latch*** coil is momentarily energized to set the latch and hold the relay in the latched position. The ***unlatch*** or release coil is momentarily energized to disengage the mechanical latch and return the relay to the unlatched position.

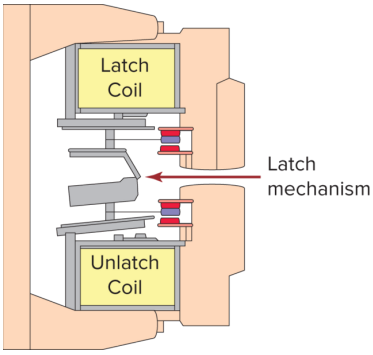


Figure 6-51 Two-coil mechanical latching relay.

Figure 6-52 shows a hardwired control circuit for an electromagnetic latching relay. The operation of the circuit can be summarized as follows:

- The contact is shown with the relay in the *unlatched* position.
- In this state the circuit to the pilot light is open and so the light is off.
- When the ON button is *momentarily* actuated, the latch coil is energized to set the relay to its latched position.
- The contacts close, completing the circuit to the pilot light, and so the light is switched on.
- The relay coil does *not* have to be continuously energized to hold the contacts closed and keep the light on.
- The only way to switch the lamp off is to actuate the OFF button, which will energize the unlatch coil and return the contacts to their open, unlatched state.
- In cases of power loss, the relay will remain in its original latched or unlatched state when power is restored.

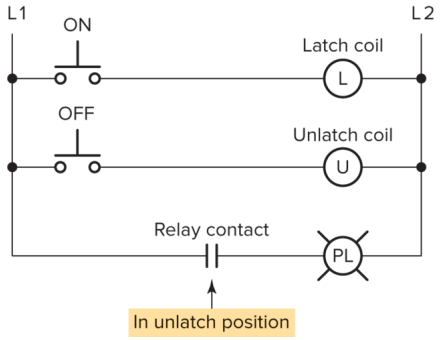


Figure 6-52 Hardwired control circuit for an electromagnetic latching relay.

An electromagnetic latching relay function can be programmed on a PLC to work like its real-world counterparts using latch and unlatch instructions. A description of the output latch **(OTL)** and output unlatch **(OTU)** instruction is given in Figure 6-53. The OTL and OTU instructions differ from the OTE instruction in that they must be used together. Both the latch and unlatch outputs must have the same address. The OTL (latch) instruction can only turn a bit on and the OTU (unlatch) instruction can only turn a bit off.

Command	Name	Symbol	Description
OTL	Output latch	(L)	OTL sets the bit to "1" when the rung becomes true and retains its state when the rung loses continuity or a power cycle occurs.
OTU	Output unlatch	(U)	OTU resets the bit to "0" when the rung becomes true and retains it.

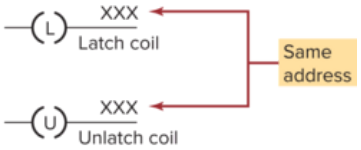


Figure 6-53 Output latch and output unlatch instruction.

The operation of the output latch and output unlatch coil instruction is illustrated in the ladder program of Figure 6-54. The operation of the program can be summarized as follows:

- Both the latch (L) and the unlatch (U) coil have the *same* address (O:2/5).
- When the on pushbutton (I:1/0) is momentarily actuated, the latch rung becomes true and the latch status bit (O:2/5) is set to 1, and so the light output is switched on.
- The status bit *will remain set to 1* when the pushbutton is released and logical continuity of the latch rung is lost.
- When the off pushbutton (I:1/1) is momentarily actuated, the unlatch rung becomes true and the latch status bit (O:2/5) is reset back to 0 and so the light is switched off.
- The status bit *will remain reset to 0* when the pushbutton is released and logical continuity of the latch rung is lost.

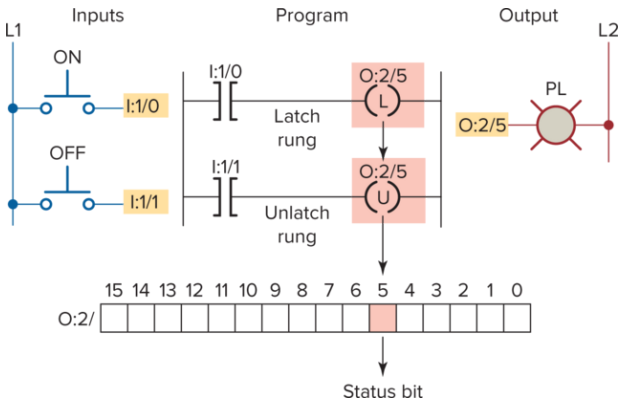


Figure 6-54 Output latch and output unlatch operation.

Output latch is an output instruction with a bit-level address. When the instruction is true, it sets a bit in the output image file. It is a retentive instruction because the bit remains set when the latch instruction goes false. In most applications it is used with an unlatch instruction. The output unlatch instruction is also an output instruction with a bit-level address. When the instruction is true, it resets a bit in the output image file. It, too, is a retentive instruction because the bit remains reset when the instruction goes false.

The process shown in **Figure 6-55** is to be used to control the level of water in a storage tank by turning a discharge pump on or off. The modes of operation are to be programmed as follows:

OFF Position–The water pump will *stop* if it is running and will *not* start if it is stopped.

Manual Mode–The pump will start if the water in the tank is at any level except low.

Automatic Mode–If the level of water in the tank *reaches a high point,* the water pump will *start* so that water can be removed from the tank, thus lowering the level.

- When the water level *reaches a low point,* the pump will *stop*.

Status Indicating Lights–Water pump running light (green)

- Low water level status light (red)
- High water level status light (yellow)

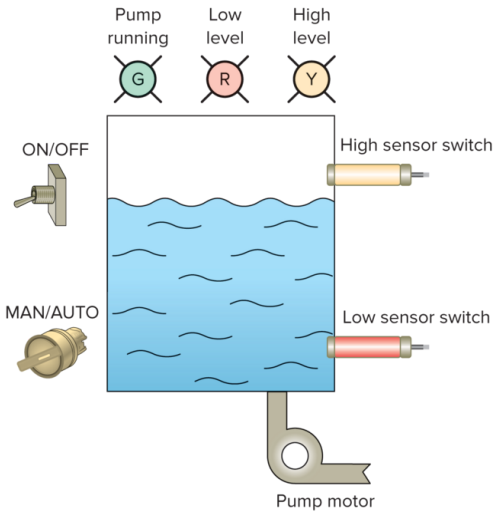


Figure 6-55 Process used to control the level of water in a storage tank.

Figure 6-56 shows a program that can be used to implement control of the water level in the storage tank. The latch and unlatch instructions form part of the program. The operation of the program can be summarized as follows:

- An internal storage bit is used for the latch and address rather than an actual discrete output address. Both have the same addresses.
- The rung 1 Examine-on instruction addressed to the off/on switch prevents the pump motor from starting under any condition when in the off (open) state.
- In the MAN mode, the rung 1 Examine-on instruction addressed to the low sensor switch allows the pump motor to operate only when the low level sensor switch is closed.
- In the AUTO mode, whenever the high sensor switch is momentarily closed the Examine-on instruction of rung 2 addressed to it will energize the latch coil. The pump will begin running and continue to operate until the unlatch coil is energized by the rung 3 Examine-off instruction addressed to the low sensor switch.
- The pump running status light is controlled by the rung 4 Examine-on instruction addressed to the motor output.
- The low-level status light is controlled by the rung 5 Examine-off instruction addressed to the low sensor switch.
- The high-level status light is controlled by the rung 6 Examine-on instruction addressed to the high sensor switch.

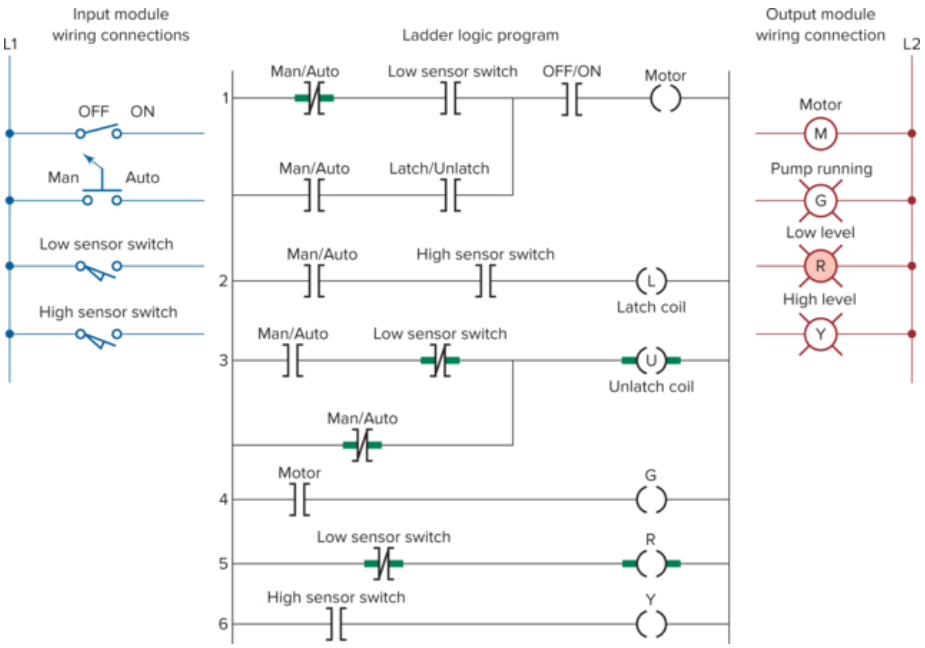


Figure 6-56 Program used to implement control of the water level in the storage tank.

Figure 6-57 shows the application of the latch instructions used in a push-to-start/push-to-stop Logix 5000 PLC program. The single normally open PB (pushbutton) performs both the On and Off functions. The exaction of the program is summarized as follows:

- The first time PB is pressed and released, the internal relay latch coil is momentarily energized making its OTE bit true (1) to switch the pilot light on.
- When this is repeated a second time, the internal relay unlatch coil is momentarily energized making its OTE bit false (0) to switch the pilot light off.

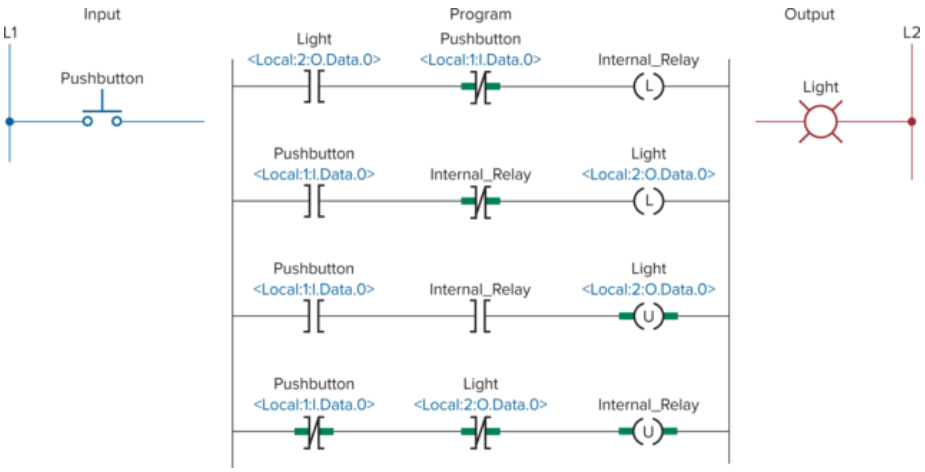


Figure 6-57 Latch instructions used in a push-to-start/push-to-stop PLC program.

OceanofPDF.com

6.11 Converting Relay Schematics into PLC Ladder Programs

The best approach to developing a PLC program from a relay schematic is to understand first the operation of each relay ladder rung. As each relay ladder rung is understood, an equivalent PLC rung can be generated. This process will require access to the relay schematic, documentation of the various input and output devices used, and possibly a process flow diagram of the operation.

Most control processes require the completion of several operations to produce the required output. Manufacturing, machining, assembling, packaging, finishing, or transporting of products requires the precise coordination of tasks.

A ***sequential*** control process is required for processes that demand that certain operations be performed in a specific order. **Figure 6-58** illustrates part of a bottle filling process. In the filling and capping operations, the tasks are (1) fill bottle and (2) press on cap. These tasks must be performed in the proper order. Obviously we could not fill the bottle after the cap is pressed on. This process, therefore, requires sequential control.

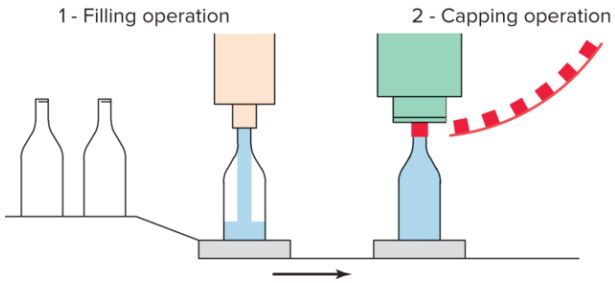


Figure 6-58 Sequential control process.

Combination controls require that certain operations be performed without regard to the order in which they are performed. **Figure 6-59** illustrates another part of the same bottle filling process. Here, the tasks are (1) place label 1 on bottle and (2) place label 2 on bottle. The order in which the tasks are performed does not really matter. In fact, however, many industrial processes that are not inherently sequential in nature are performed in a sequential manner for the most efficient order of operations.

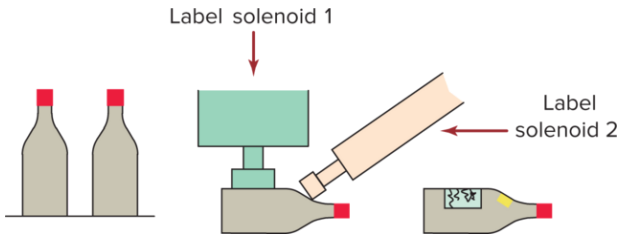


Figure 6-59 Combination control process.

Automatic control involves maintaining a desired set point at an output. One example is maintaining a certain set-point temperature in a furnace as illustrated in **Figure 6-60**. If there is deviation from

that set point, an error is determined by comparing the output against the set point and using this error to make a correction. This requires feedback from the output to the control for the input.

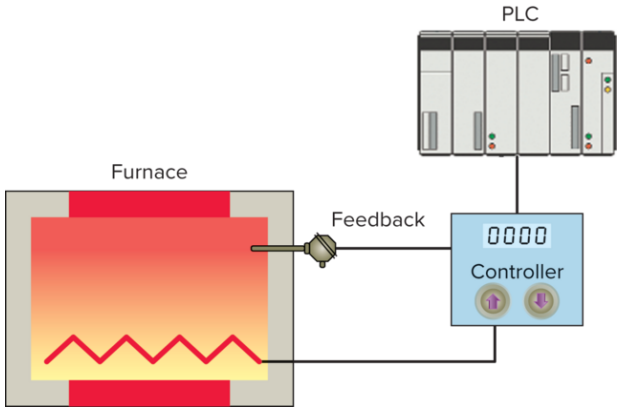


Figure 6-60 Automatic control process.

The converting of a simple sequential process can be examined with reference to the process flow diagram illustrated in **Figure 6-61**. The sequential task is as follows:

1. Start button is pressed.
2. Table motor is started.
3. Package moves to the position of the limit switch and automatically stops.

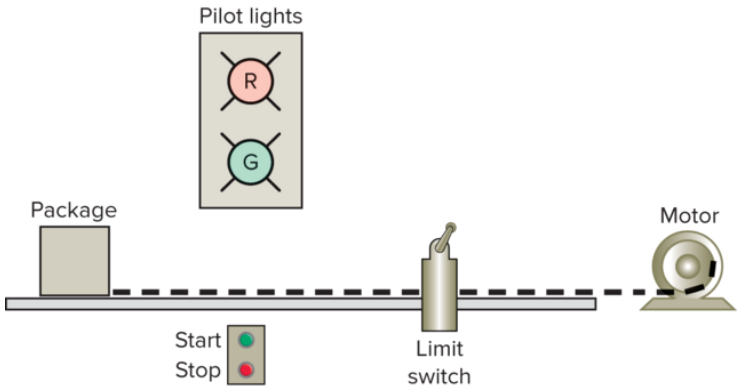


Figure 6-61 Sequential process flow diagram.

Other auxiliary features include:

- a stop button that will stop the table, for any reason, before the package reaches the limit switch position;
- a red pilot light to indicate the table is stopped; and
- a green pilot light to indicate the table is running.

A relay control schematic for the sequential process is shown in **Figure 6-62**. The operation of this hardwired circuit can be summarized as follows:

- Start button is actuated; CR is energized if stop button and limit switch are not actuated.
- Contact CR-1 closes, sealing in CR when the start button is released.
- Contact CR-2 opens, switching the red pilot light from on to off.
- Contact CR-3 closes, switching the green pilot light from off to on.
- Contact CR-4 closes to energize the motor starter coil, starting the motor and moving the package toward the limit switch.
- Limit switch is actuated, de-energizing relay coil CR.
- Contact CR-1 opens, opening the seal-in circuit.
- Contact CR-2 closes, switching the red pilot light from off to on.
- Contact CR-3 opens, switching the green pilot light from on to off.
- Contact CR-4 opens, de-energizing the motor starter coil to stop the motor and end the sequence.

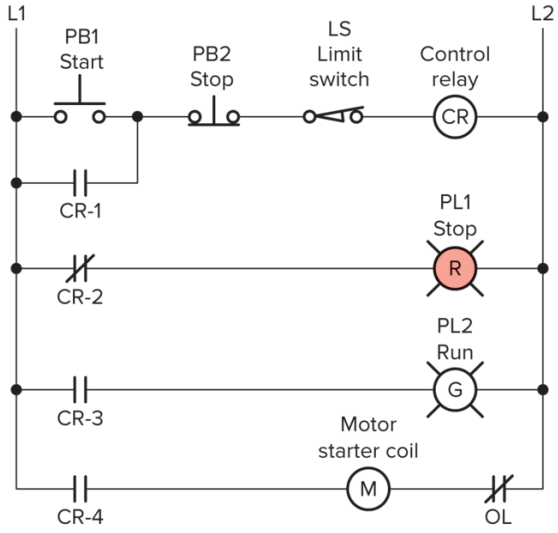


Figure 6-62 Relay schematic for the sequential process.

The control relay schematic for the sequential process can be executed using the PLC program shown in **Figure 6-63**. The operation of the program is summarized as follows:

- An internal relay is used to replace control relay CR.
- All internal relay contacts are programmed with a logically equivalent PLC contact instruction.
- The internal relay uses an output instruction in conjunction with programmed contacts which are examined for an open (0) or closed (1) state.
- The operation of the two is identical with rung continuity being established with the relay control and logic continuity with programmed control.

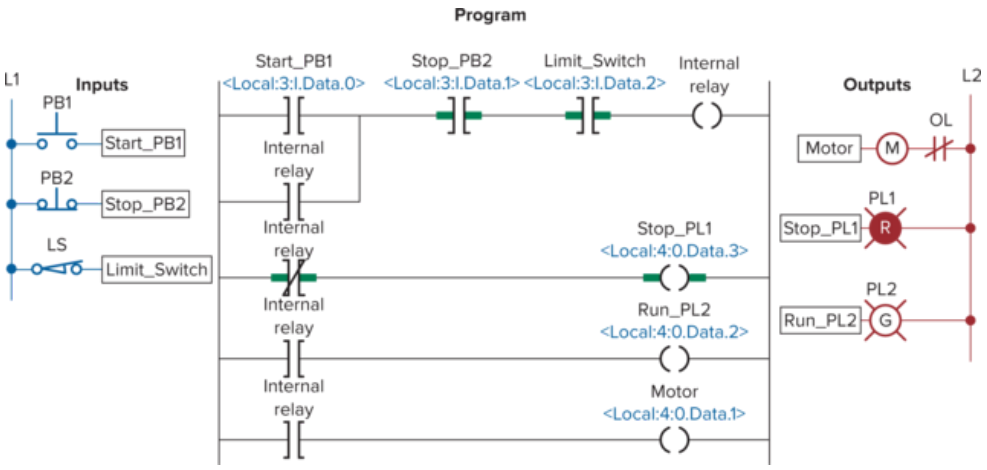


Figure 6-63 Sequential process PLC program.

There is more than one method to correctly design the ladder logic program for a given control process. In some cases one arrangement may be more efficient in terms of the amount of memory used and the time required to scan the program. **Figure 6-64** illustrates an example of an arrangement of series instructions of a rung programmed for optimum scan time. The series instructions are programmed from the most likely to be *false* (far left) to the least likely to be *false* (far right). Once the processor sees a false input instruction in series, the processor stops checking the rung at the false condition and sets the output false.

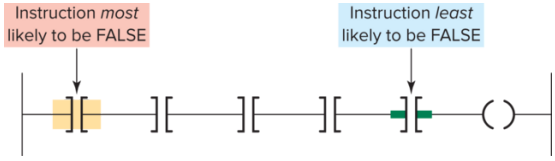


Figure 6-64 Series instructions programmed for optimum scan time.

Figure 6-65 illustrates an example of an arrangement of parallel instructions of a rung programmed for optimum scan time. The parallel path that is most often *true* is placed on the top of the rung. The processor will not look at the others unless the top path is *false.*

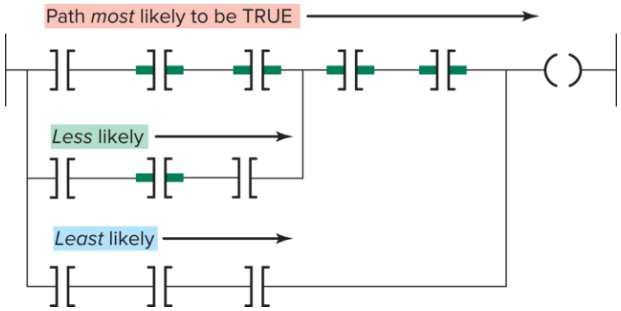


Figure 6-65 Parallel instructions programmed for optimum scan time.

Figure 6-66 shows a hardwired jog control circuit that incorporates a jog control relay. The operation of the circuit can be summarized as follows:

- Pressing the start pushbutton completes a circuit for the CR coil, closing the CR1 and CR2 contacts.
- The CR1 contact completes the circuit for the M coil, starting the motor.
- The M maintaining contact closes; this maintains the circuit for the M coil.
- Pressing the jog button energizes the M coil only, starting the motor. Both CR contacts remain open, and the CR coil is de-energized. The M coil will not remain energized when the jog push button is released.

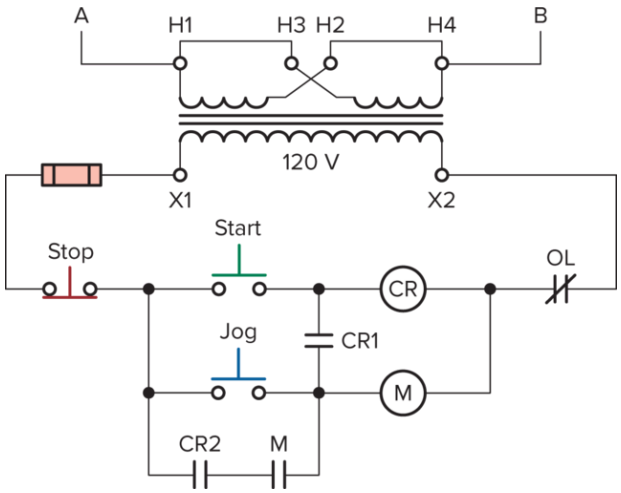


Figure 6-66 Jog circuit with control relay.

Figure 6-67 shows a PLC program equivalent of the hardwired relay jog circuit. Note that the function of the control relay is now accomplished using an internal PLC instruction (B3:1/0).

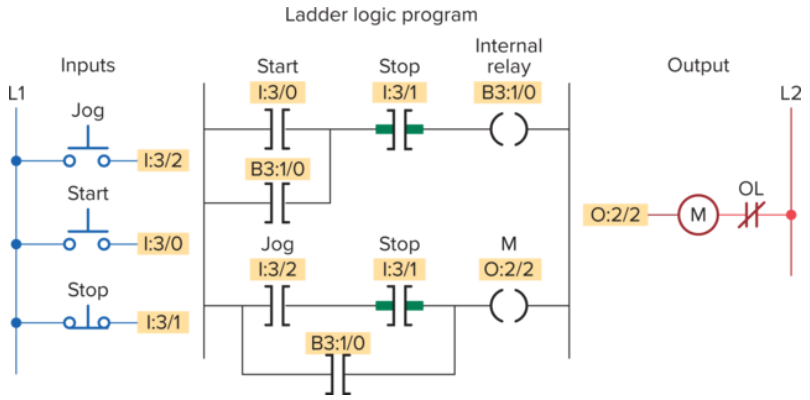


Figure 6-67 PLC program equivalent of the hardwired relay jog circuit.

OceanofPDF.com

6.12 Writing a Ladder Logic Program Directly from a Narrative Description

In most cases, it is possible to prepare a ladder logic program directly from the narrative description of a control process. Some of the steps in planning a program are as follows:

- Define the process to be controlled.
- Draw a sketch of the process, including all sensors and manual controls needed to carry out the control sequence.
- List the sequence of operational steps in as much detail as possible.
- Write the ladder logic program to be used as a basis for the PLC program.
- Consider different scenarios where the process sequence may go astray and make adjustments as needed.
- Consider the safety of operating personnel and make adjustments as needed.

The following are examples of ladder logic programs derived from narrative descriptions of control processes.

EXAMPLE 6-1

Figure 6-68 shows the sketch of a drilling process that requires the drill press to turn on only if there is a part present and the operator has one hand on each of the start switches. This precaution will ensure that the operator's hands are not in the way of the drill.

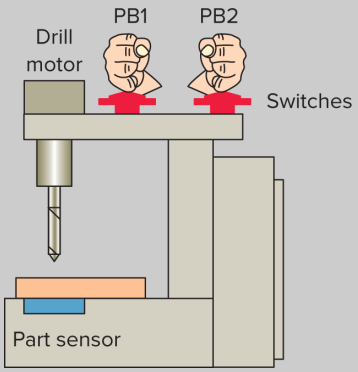


Figure 6-68 Sketch of the drilling process.

The sequence of operation requires that switches 1 and 2 and the part sensor all be activated to make the drill motor operate. **Figure 6-69** shows a PLC program used to implement the process.

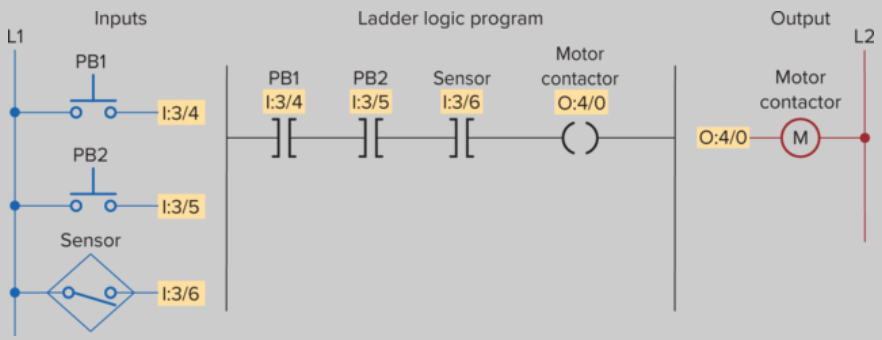


Figure 6-69 Drilling process PLC program.

EXAMPLE 6-2

A motorized overhead garage door is to be operated automatically to preset open and closed positions. The field devices include one of each of the following:

- Reversing *motor contactor* for the up and down directions.
- Normally open *down limit switch* to sense when the door is fully closed.
- Normally open-held closed *up limit switch* to sense when the door is fully opened.
- Normally open *door up button* for the up direction.
- Normally open *door down button* for the down direction.
- Normally closed *door stop button* for stopping the door.
- Red *door ajar light* to signal when the door is partially open.
- Green *door open light* to signal when the door is fully open.
- Yellow *door closed light* to signal when the door is fully closed.

The sequence of operation requires that:

- When the up button is pushed, the up motor contactor energizes and the door travels upward until the up limit switch is actuated.
- When the down button is pushed, the down motor contactor energizes and the door travels down until the down limit switch is actuated.
- When the stop button is pushed, the motor stops. The motor must be stopped before it can change direction.

Figure 6-70 shows a PLC program that could be used to operate the door.

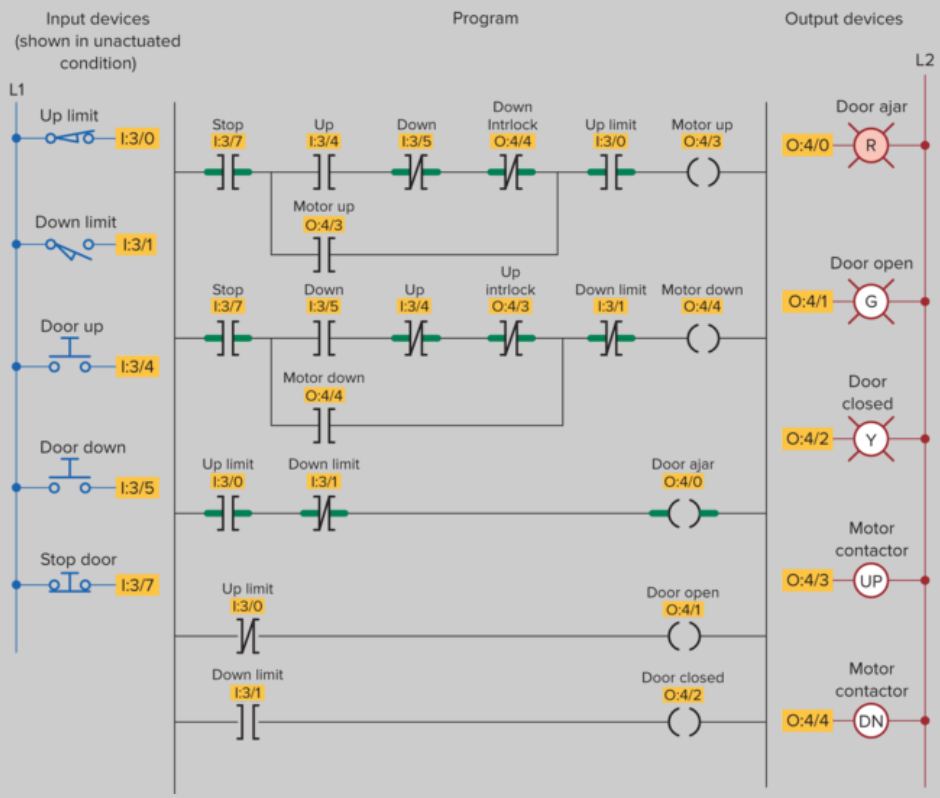


Figure 6-70 Motorized overhead garage door PLC program.

EXAMPLE 6-3

Figure 6-71 shows the sketch of a continuous filling operation. This process requires that boxes moving on a conveyor be automatically positioned and filled.

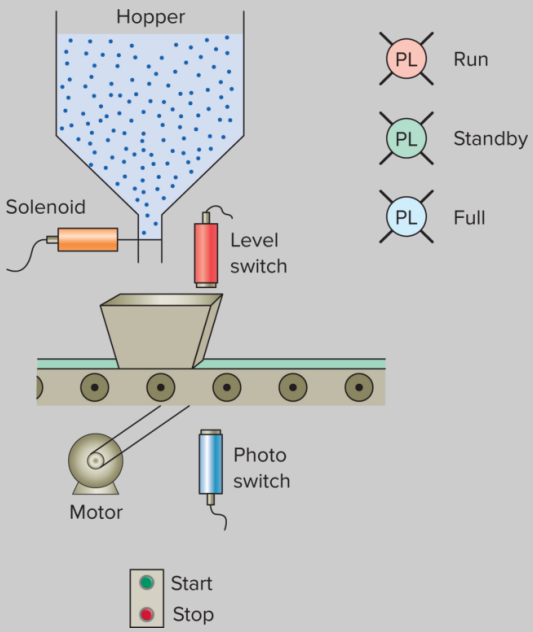


Figure 6-71 Sketch of the continuous filling operation.

The sequence of operation for the continuous filling operation is as follows:

- Start the conveyor when the start button is momentarily pressed.
- Stop the conveyor when the stop button is momentarily pressed.
- Energize the run status light when the process is operating.
- Energize the standby status light when the process is stopped.
- Stop the conveyor when the right edge of the box is first sensed by the photosensor.
- With the box in position and the conveyor stopped, open the solenoid valve and allow the box to fill. Filling should stop when the level sensor goes true.
- Energize the full light when the box is full. The full light should remain energized until the box is moved clear of the photosensor.
- When the box is full, the start motor will energize to remove the full box. The motor will remain on until a new box is detected by the photo sensor.

Figure 6-72 shows the ladder logic program required for the operation.

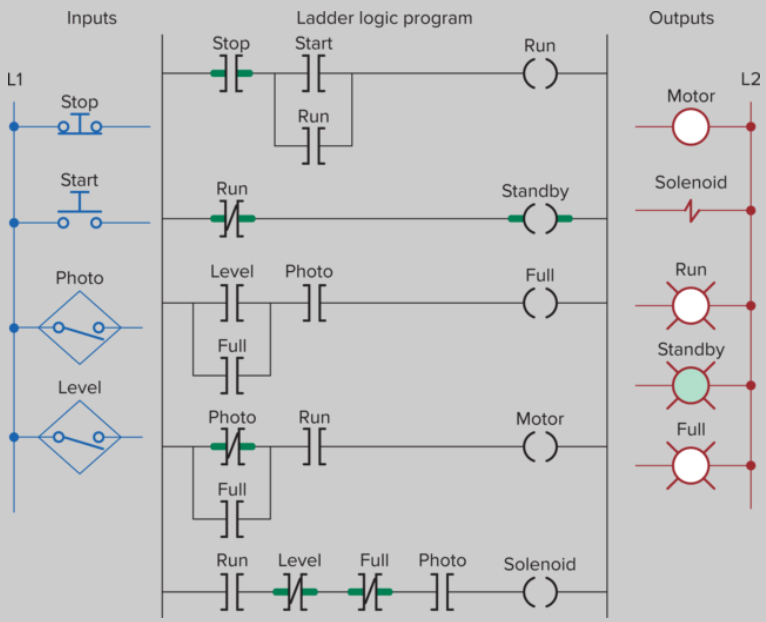


Figure 6-72 Continuous filling operation PLC program.

OceanofPDF.com

6.13 Instrumentation

Instrumentation is the use of measuring instruments to monitor and control a process. It involves the design and calibration of systems used to measure, record, and control industrial process variables. These variables may include pressure, temperature, flow rate, weight, and chemical consistency. An **instrument** is a device that measures and/or acts to control any kind of physical process and may include flow devices, level devices, thermocouples, and pressure switches.

Every instrument has at least one input and one output.

- For a pressure sensor, the input could be some fluid pressure and the output a 4- to 20-mA current signal.
- For a loop indicator, the input could be a 4- to 20-mA current signal and the output an electronic display.
- For a variable-speed motor drive, the input could be an electronic signal and the output electric power to the motor.

To **calibrate** an instrument means to check, and if necessary adjust, its response so the output accurately corresponds to its input throughout a specified range. Instrument calibration involves exposure of the instrument to an actual input stimulus of precisely known quantity. For a pressure gauge, this would mean subjecting the pressure instrument to known fluid pressures and comparing the instrument response against those known pressure quantities. **Smart instruments** (Figure 6-73) that contain microprocessors have built-in diagnostic ability, greater accuracy, and the ability to communicate digitally with host devices for reporting of various parameters.

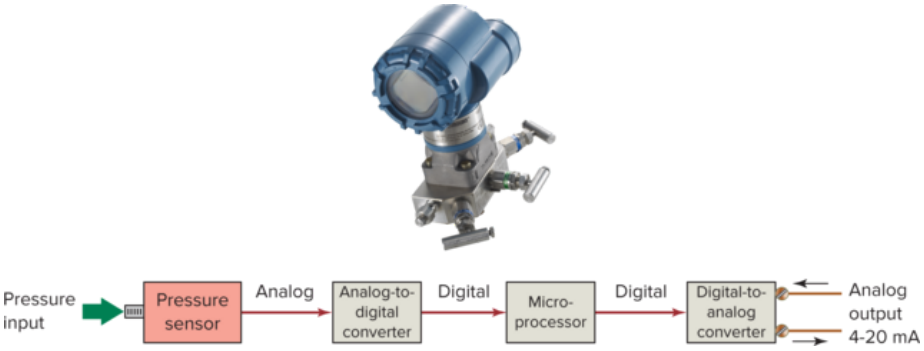


Figure 6-73 Smart instruments.

Source: Photo courtesy Emerson

The PLC's role as part of an industrial instrumentation system is to receive, process, and send signals from input and to output devices. With the use of programming software the PLC can control, monitor, and display all the parameters associated with a given process.

OceanofPDF.com



CHAPTER 6 REVIEW QUESTIONS

1. Explain the basic operating principle of an electromagnetic control relay.
2. What is the operating difference between a normally open and a normally closed relay contact?
3. In what ways are control relay coils and contacts rated?
4. How do contactors differ from relays?
5. What is the main difference between a contactor and a magnetic motor starter?
6. **a.** Draw the schematic for an across-the-line AC magnetic motor starter.
   b. With reference to this schematic, explain the function of each of the following parts:
      i. Main contact M
      ii. Control contact M
      iii. Starter coil M
      iv. OL relay coils
      v. OL relay contact
7. The current requirement for the control circuit of a magnetic starter is normally much smaller than that required by the power circuit. Why?
8. Compare the method of operation of each of the following types of switches:
   a. Manually operated switch
   b. Mechanically operated switch
   c. Proximity switch
9. What do the abbreviations NO and NC represent when used to describe switch contacts?
10. Draw the electrical symbol used to represent each of the following switches:
    a. NO pushbutton switch
    b. NC pushbutton switch
    c. Break-make pushbutton switch
    d. Three-position selector switch
    e. NO limit switch
    f. NC temperature switch
    g. NO pressure switch
    h. NC level switch
    i. NO proximity switch
11. Outline the method used to actuate inductive and capacitive proximity sensors.

12. How are reed switch sensors actuated?

13. Compare the operation of a photovoltaic solar cell with that of a photoconductive cell.

14. What are the two basic components of a photoelectric sensor?

15. Compare the operation of the reflective-type and through-beam photoelectric sensors.

16. Give an explanation of how a scanner and a decoder act in conjunction with each other to read a bar code.

17. How does an ultrasonic sensor operate?

18. Explain the principle of operation of a strain gauge.

19. Explain the principle of operation of a thermocouple.

20. What is the most common approach taken with regard to the measurement of fluid flow?

21. Explain how a tachometer is used to measure rotational speed.

22. How does an optical encoder work?

23. Draw an electrical symbol used to represent each of the following PLC control devices:

- **a.** Pilot light
- **b.** Relay
- **c.** Motor starter coil
- **d.** OL relay contact
- **e.** Alarm
- **f.** Heater
- **g.** Solenoid
- **h.** Solenoid valve
- **i.** Motor
- **j.** Horn

24. Explain the function of each of the following actuators:

- **a.** Solenoid
- **b.** Solenoid valve
- **c.** Stepper motor

25. Compare the operation of open-loop and closed-loop control.

26. What is a seal-in circuit?

27. In what way is the construction and operation of an electromechanical latching relay different from a standard relay?

28. Give a short description of each of the following control processes:

- **a.** Sequential
- **b.** Combination
- **c.** Automatic

29. Compare the type of sensor signal obtained from a thermocouple with that from an RTD.

30. Explain how a magnetic reed float switch works.

31. What is the function of an electrical interlocking circuit?

32. What is the role of instrumentation in an industrial process?

33. You have been assigned the task of calibrating an instrument. How would you proceed?

OceanofPDF.com



CHAPTER 6 PROBLEMS

1. Design and draw the schematic for a conventional hardwired relay circuit that will perform each of the following circuit functions when a normally closed pushbutton is pressed:

- Switch a pilot light on
- De-energize a solenoid
- Start a motor running
- Sound a horn

2. Design and draw the schematic for a conventional hardwired circuit that will perform the following circuit functions using two break-make pushbuttons:

- Turn on light L1 when pushbutton PB1 is pressed.
- Turn on light L2 when pushbutton PB2 is pressed.
- Electrically interlock the pushbuttons so that L1 and L2 cannot both be turned on at the same time.

3. Study the ladder logic program in **Figure 6-74**, and answer the questions that follow:

a. Under what condition will the latch rung 1 be true?

b. Under what conditions will the unlatch rung 2 be true?

c. Under what condition will rung 3 be true?

d. When PL1 is on, the relay is in what state (latched or unlatched)?

e. When PL2 is on, the relay is in what state (latched or unlatched)?

f. If AC power is removed and then restored to the circuit, what pilot light will automatically come on when the power is restored?

g. Assume the relay is in its latched state and all three inputs are false. What input change(s) must occur for the relay to switch into its unlatched state?

h. If the examine if closed instructions at addresses I/1, I/2, and I/3 are all true, what state will the relay remain in (latched or unlatched)?

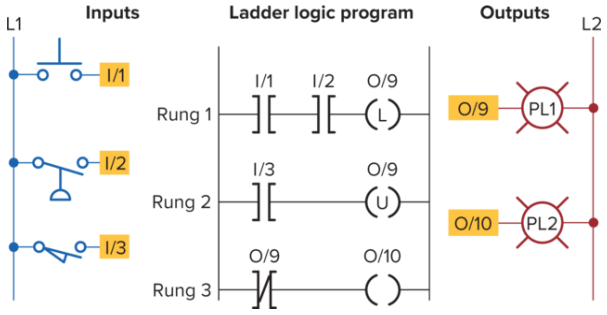


Figure 6-74 Ladder logic program for Problem 3.

4. Design a PLC program and prepare a typical I/O connection diagram and ladder logic program that will correctly execute the hardwired control circuit in **Figure 6-75**.

Assume: Stop pushbutton used is an NO type.
Run pushbutton used is an NO type.
Jog pushbutton used has one set of NO contacts.
OL contact is hardwired.

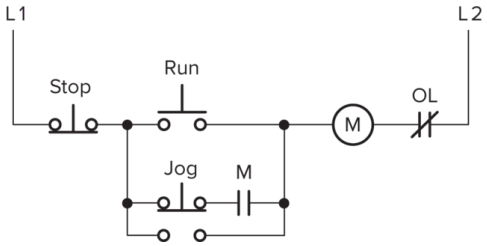


Figure 6-75 Hardwired control circuit for Problem 4.

5. Design a PLC program and prepare a typical I/O connection diagram and ladder logic program that will correctly execute the hardwired control circuit in **Figure 6-76**.

Assume: PB1 pushbutton used is an NO type.
PB2 pushbutton used is an NC type.
PS1 pressure switch used is an NO type.
LS1 limit switch used has only one set of NC contacts.

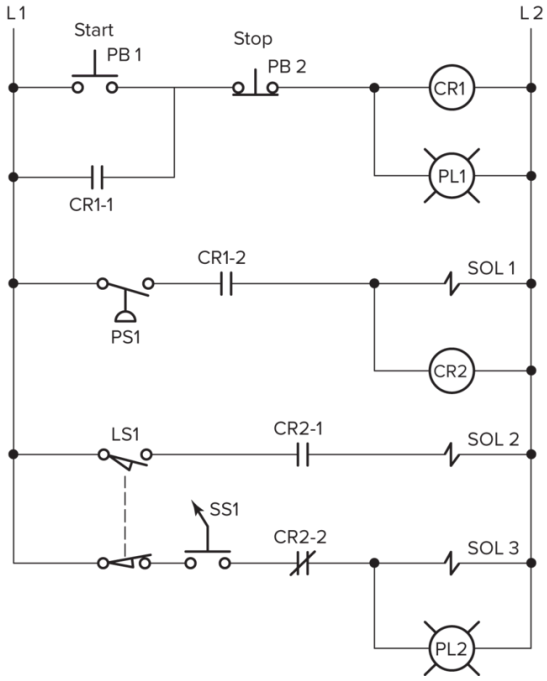


Figure 6-76 Hardwired control circuit for Problem 5.

6. Design a PLC program and prepare a typical I/O connection diagram and ladder logic program that will correctly execute the hardwired control circuit in **Figure 6-77**.

Assume: PB1 pushbutton used is an NC type.
PB2 and PB3 are each wired using one set of NO contacts.
OL contact is hardwired.

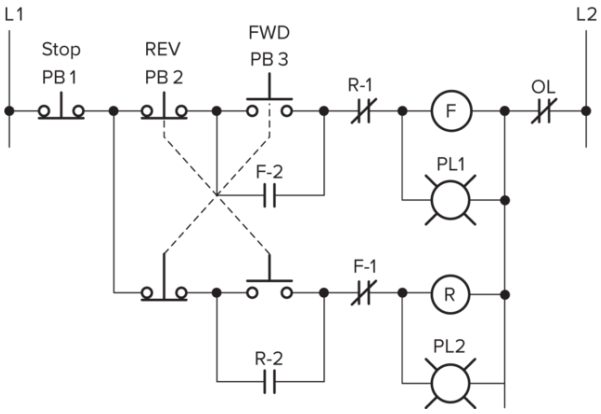


Figure 6-77 Hardwired control circuit for Problem 6.

7. Design a PLC program and prepare a typical I/O connection diagram and ladder logic program for the following motor control specifications:

- A motor must be started and stopped from any one of three start/stop pushbutton stations.
- Each start/stop station contains one NO start pushbutton and one NC stop pushbutton.
- Motor OL contacts are to be hardwired.

8. Design a PLC program and prepare a typical I/O connection diagram and ladder logic program for the following motor control specifications:

- Three starters are to be wired so that each starter is operated from its own start/stop pushbutton station.
- A master stop station is to be included that will trip out all starters when pushed.
- Overload relay contacts are to be programmed so that an overload on any one of the starters will automatically drop all of the starters.
- *All* pushbuttons are to be wired using one set of NO contacts.

9. A temperature control system consists of four thermostats controlling three heating units. The thermostat contacts are set to close at 50°, 60°, 70°, and 80°F, respectively. The PLC ladder logic program is to be designed so that at a temperature below 50°F, three heaters are to be ON. From 50° to 60°F, two heaters are to be ON. For 60° to 70°F, one heater is to be ON. Above 80°F, there is a safety shutoff for all three heaters in case one stays on because of a malfunction. A master switch is to be used to turn the system ON and OFF. Prepare a typical PLC program for this control process.

10. A pump is to be used to fill two storage tanks. The pump is manually started by the operator from a start/stop station. When the first tank is full, the control logic must be able to automatically stop flow to the first tank and direct flow to the second tank through the use of sensors and electric solenoid valves. When the second tank is full, the pump must shut down automatically. Indicator lamps are to be included to signal when each tank is full.

a. Draw a sketch of the process.

b. Prepare a typical PLC program for this control process.

11. Write the optimum ladder logic rung for each of the following scenarios, and arrange the instructions for optimum performance:

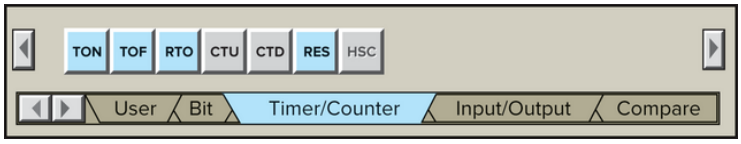
a. If limit switches LS1 or LS2 or LS3 are on, or if LS5 and LS7 are on, turn on; otherwise, turn off. (Commonly, if LS5 and LS7 are on, the other conditions rarely occur.)

b. Turn on an output when switches SW6, SW7, and SW8 are all on, or when SW55 is on. (SW55 is an indication of an alarm state, so it is rarely on; SW7 is on most often, then SW8, then SW6.)

OceanofPDF.com

7

Programming Timers



Chapter Objectives

After completing this chapter, you will be able to:

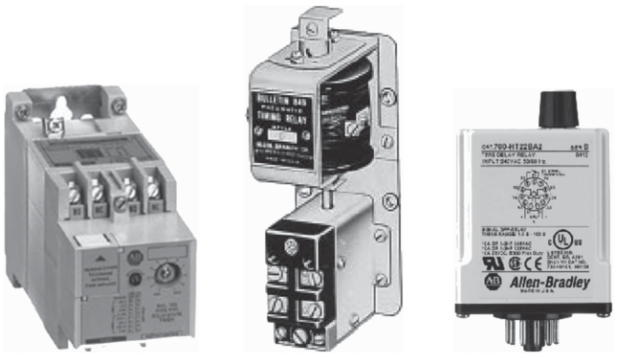
- Describe the operation of pneumatic on-delay and off-delay timers
- Describe PLC timer instruction and differentiate between a nonretentive and retentive timer
- Convert fundamental timer relay schematic diagrams to PLC programs
- Analyze and interpret typical PLC timer programs
- Program the control of outputs using the timer instruction control bits

The most commonly used PLC instruction, after coils and contacts, is the timer. This chapter deals with how timers time intervals and the way in which they can control outputs. We discuss the basic PLC on-delay timer function, as well as other timing functions derived from it, and typical industrial timing tasks.

OceanofPDF.com

7.1 Mechanical Timing Relays

There are very few industrial control systems that do not need at least one or two timed functions. **Mechanical timing relays** are used to delay the opening or closing of contacts for circuit control. The operation of a mechanical timing relay is similar to that of a control relay, except that certain of its contacts are designed to operate at a preset time interval, after the coil is energized or de-energized. Typical types of mechanical and electronic timing relays are shown in **Figure 7-1**. Timers allow a multitude of operations in a control circuit to be automatically started and stopped at different time intervals.

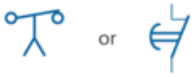


Solid-state timing relay Pneumatic timing relay Plug-in timing relay

Figure 7-1 Timing relays.

Source: Photo courtesy Rockwell Automation, **www.rockwellautomation.com.**

Mechanical timing relays provide time delay through two arrangements. The first arrangement, *on delay,* provides time delay when the relay coil is *energized.* The second arrangement, *off delay,* provides time delay when the relay coil is *de-energized.* **Figure 7-2** illustrates the different relay symbols used for timed contacts.

On-delay symbols	
	
Normally open, timed closed contact (NOTC). Contact is open when relay coil is de-energized. When relay is energized, there is a time delay in closing.	Normally closed, timed open contact (NCTO). Contact is closed when relay coil is de-energized. When relay is energized, there is a time delay in opening.

Off-delay symbols	
	
Normally open, timed open contact (NOTO). Contact is normally open when relay coil is de-energized. When relay coil is energized, contact closes instantly. When relay coil is de-energized, there is a time delay before the contact opens.	Normally closed, timed closed contact (NCTC). Contact is normally closed when relay coil is de-energized. When relay coil is energized, contact opens instantly. When relay coil is de-energized, there is a time delay before the contact closes.

Figure 7-2 Timed contact symbols.

Mechanical timing relays often have **instantaneous contacts** in addition to timed contacts. The instantaneous contacts change state as soon as the timer coil is powered while the delayed contacts change state at the end of the time delay. Instantaneous contacts are often used as holding or sealing contacts in a control circuit.

The **on-delay timer** is sometimes referred to as DOE, which stands for delay on energize. The time delay of the contacts begins once the timer is switched on; hence the term *on-delay timing.*

Figure 7-3 shows an on-delay timer circuit that uses a normally open, timed closed (NOTC) contact. The operation of the circuit can be summarized as follows:

- With S1 initially open, TD coil is de-energized so TD1 contacts are open and light L1 will be off.
- When S1 is closed, TD coil is energized and the timing period starts. TD1 contacts are delayed from closing so L1 remains off.
- After the 10 s time-delay period has elapsed, TD1 contacts close and L1 is switched on.
- When S1 is opened, TD coil is de-energized and TD1 contacts open instantly to switch L1 off.

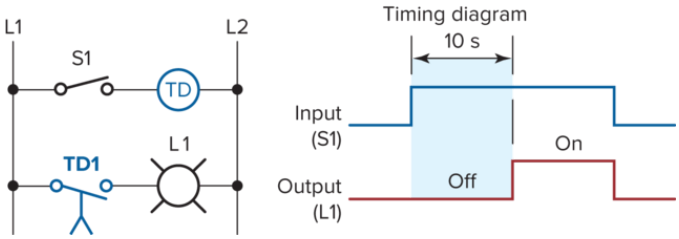


Figure 7-3 On-delay timer circuit that uses a normally open, timed closed (NOTC) contact.

Figure 7-4 shows an on-delay timer circuit that uses a normally closed, timed open (NCTO) contact. The operation of the circuit can be summarized as follows:

- With S1 initially open, TD coil is de-energized so TD1 contacts are closed and light L1 will be on.
- When S1 is closed, TD coil is energized and the timing period starts. TD1 contacts are delayed from opening so L1 remains on.
- After the 10 s time-delay period has elapsed, TD1 contacts open and L1 is switched off.
- When S1 is opened, TD coil is de-energized and TD1 contacts close instantly to switch L1 on.

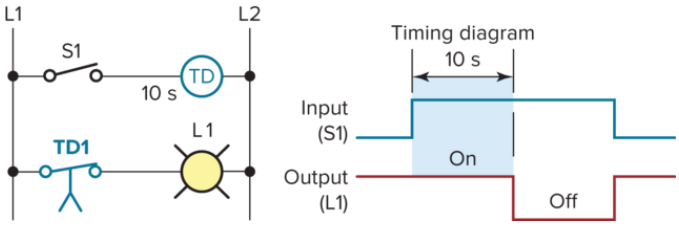


Figure 7-4 On-delay timer circuit that uses a normally closed, timed open (NCTO) contact.

Figure 7-5 shows an **off-delay timer** circuit that uses a normally open, timed open (NOTO) contact. The operation of the circuit can be summarized as follows:

- With S1 initially open, TD coil is de-energized so TD1 contacts are open and light L1 will be off.
- When S1 is closed, TD coil is energized and TD1 contacts close instantly to switch light L1 on.
- When S1 is opened, TD coil is de-energized and the timing period starts.
- After the 10 s time-delay period has elapsed, TD1 contacts open to switch the light off.

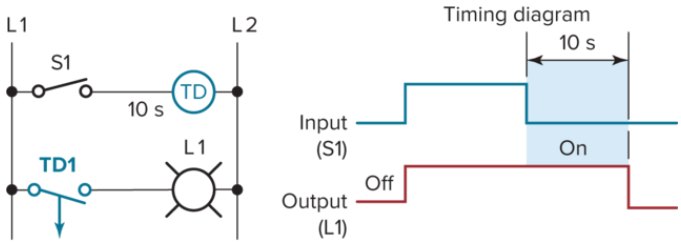


Figure 7-5 Off-delay timer circuit that uses a normally open, timed open (NOTO) contact.

Figure 7-6 shows an off-delay timer circuit that uses a normally closed, timed closed (NCTC) contact. The operation of the circuit can be summarized as follows:

- With S1 initially open, TD coil is de-energized so TD1 contacts are closed and light L1 will be on.
- When S1 is closed, TD coil is energized and TD1 contacts open instantly to switch light L1 off.
- When S1 is opened, TD coil is de-energized and the timing period starts. TD1 contacts are delayed from closing so L1 remains off.
- After the 10 s time-delay period has elapsed, TD1 contacts close to switch the light on.

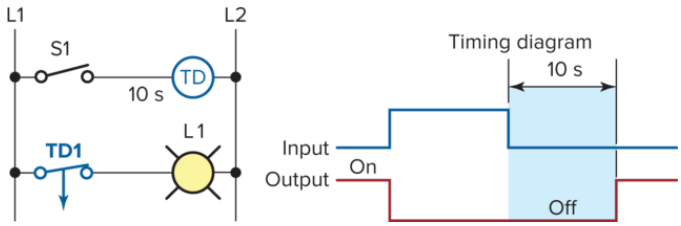


Figure 7-6 Off-delay timer circuit that uses a normally closed, timed closed (NCTC) contact.

OceanofPDF.com

7.2 Timer Instructions

PLC timers are instructions that provide the same functions as on-delay and off-delay mechanical and electronic timing relays. All PLC timers are output instructions. PLC timers offer several advantages over their mechanical and electronic counterparts. These include the fact that:

- The entire timing function occurs inside the controller.
- Time settings can be easily changed.
- The number of timers used in a circuit can be increased or decreased through the use of programming changes rather than wiring changes.
- Timer accuracy and repeatability are extremely high because time delays are generated in the PLC processor.

In general, there are three different PLC timer types: the *on-delay timer (**TON**), off-delay timer (**TOF**),* and *retentive timer on (**RTO**).* The most common is the on-delay timer, which is the basic function. There are also many other timing configurations, all of which can be derived from one or more of the basic time-delay functions. **Figure 7-7** shows the timer selection toolbar for the RSLogix 500 software. These timer commands can be summarized as follows:

Page 124

TON (Timer On Delay)—Counts time-based intervals when the instruction is true.

TOF (Timer Off Delay)—Counts time-based intervals when the instruction transitions from a true to false condition.

RTO (Retentive Timer On)—Counts time-based intervals when the instruction is true and retains the accumulated value when the instruction goes false or when power cycle occurs.

RES (Reset)—Resets a retentive timer's accumulated value to zero.

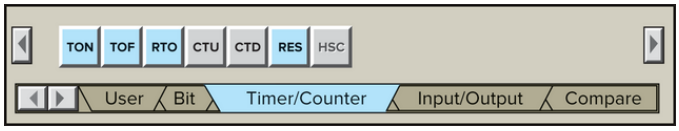


Figure 7-7 Timer selection toolbar.

Several quantities are associated with the timer instruction:

- **Time Base** The time base of a timer is the unit of time used by a timer to time an event. A timer instruction times an event by counting the number of times the time base has occurred. Depending on the manufacturer and type of PLC, time base values can be in 1 ms (0.001 s), 10 ms (0.01 s), 100 ms (0.1 s), or 1 second intervals. For example, if a timer has a time base of 1 second and it is timing something that is 5 seconds long, the PLC will wait until the time base has occurred 5 times before the timer times out. Conversely, if the PLC's time base setting is 0.01 seconds, it will wait until the time base has occurred 500 times before timing out. The smaller the time base, the better the accuracy of the timer. RSLogix 5000 controllers have a timer time base of 1 millisecond (0.001 s).

- **Preset Value** The preset value of a timer represents the time duration for the timing circuit. Total timing interval = the preset value x time base. For example, for a timer with a preset value of 10,000 and a time base of 1 millisecond $(0.001\ \text{s})$ the total timing interval would be 10 seconds.
- **Accumulated Value** The accumulated value of the timer represents the amount of time that has elapsed from the moment the timing started. It keeps track of how many times the time base has occurred since the timer instruction was initiated.

Timers are most often represented by boxes in ladder logic. **Figure 7-8** illustrates a generic **block format** for a retentive timer that requires two input lines. Its operation can be summarized as follows:

- The timer block has two input conditions associated with it, namely, the *control* and *reset.*
- The control line controls the actual timing operation of the timer. Whenever this line is true or power is supplied to this input, the timer will time. Removal of power from the control line input halts the further timing of the timer.
- The reset line resets the timer's accumulated value to zero.
- The timer instruction block contains information pertaining to the operation of the timer, including the preset time, the time base of the timer, and the current or accumulated time.
- All block-formatted timers provide at least one output signal from the timer. The timer continuously compares its current time with its preset time, and its output is false (logic 0) as long as the current time is less than the preset time. When the current time equals the preset time, the output changes to true (logic 1).

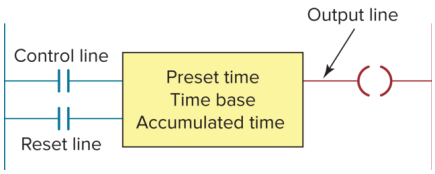


Figure 7-8 Block-formatted timer instruction.

OceanofPDF.com

7.3 On-Delay Timer Instruction

Most timers are output instructions that are conditioned by input instructions. An ***on-delay timer*** is used when you want to program a time delay before an instruction becomes true. **Figure 7-9** illustrates the principle of operation of an on-delay timer. Its operation can be summarized as follows:

- The on-delay timer operates such that when the rung containing the timer is true, the timer time-out period commences.
- The timed output becomes true sometime after the timer rung becomes true; hence, the timer is said to have an on-delay.
- The length of the time delay can be adjusted by changing the preset value.
- In addition, some PLCs allow the option of changing the time base, or resolution, of the timer. As the time base you select becomes smaller, the accuracy of the timer increases.

Page 125

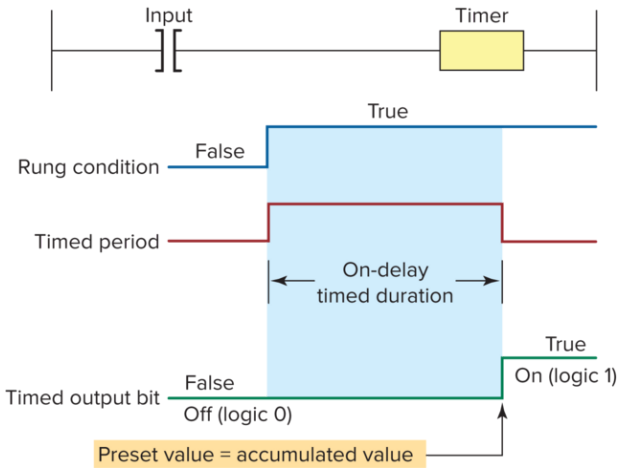


Figure 7-9 Principle of operation of an on-delay timer.

The Allen-Bradley RSLogix 500 timer file is file 4 (**Figure 7-10**). Each timer is composed of three 16-bit words, collectively called a timer element. There can be up to 256 timer elements. Addresses for timer file 4, timer element number 2 (T4:2), are listed below.

T4 = timer file 4

:2 = timer element number 2 (0–255 timer elements per file)

T4:2/DN is the address for the done bit of the timer.

T4:2/TT is the address for the timer-timing bit of the timer.

T4:2/EN is the address for the enable bit of the timer.

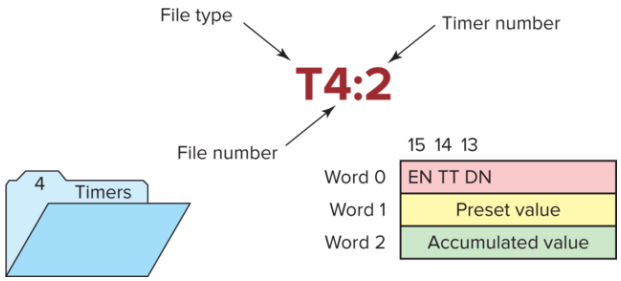


Figure 7-10 RSLogix 500 timer file.

The *control word* uses the following three control bits:

Enable (EN) bit—The enable bit is true (has a status of 1) whenever the timer instruction is true. When the timer instruction is false, the enable bit is false (has a status of 0).

Timer-timing (TT) bit—The timer-timing bit is true whenever the accumulated value of the timer is changing, which means the timer is timing. When the timer is not timing, the accumulated value is not changing, so the timer-timing bit is false.

Done (DN) bit—The done bit changes state whenever the accumulated value reaches the preset value. Its state depends on the type of timer being used.

The *preset value (PRE) word* is the set point of the timer, that is, the value up to which the timer will time. The preset word has a range of 0 through 32,767 and is stored in binary form. The preset will not store a negative number.

The *accumulated value (ACC) word* is the value that increments as the timer is timing. The accumulated value will stop incrementing when its value reaches the preset value.

The timer instruction also requires that you enter a *time base,* which is either 1.0,.1 or 0.01 s. The actual preset time interval is the time base multiplied by the value stored in the timer's preset word. The actual accumulated time interval is the time base multiplied by the value stored in the timer's accumulated word.

Figure 7-11 shows an example of the on-delay timer instruction used as part of the Allen-Bradley RSLogix 500 controller instruction sets. The information to be entered includes:

Timer number—This number must come from the timer file. In the example shown, the timer number is T4:0, which represents timer file 4, timer 0 in that file. The timer address must be unique for this timer and may not be used for any other timer.

Time base—The time base (which is always expressed in seconds) may be either 1.0,.1 or 0.01 s. In the example shown, the time base is 1.0 s.

Preset value—In the example shown, the preset value is 15. The timer preset value can range from 0 through 32,767.

Accumulated value—In the example shown, the accumulated value is 0. The timer's accumulated value normally is entered as 0, although it is possible to enter a value from 0 through 32,767. Regardless of the value that is preloaded, the timer value will become 0 whenever the timer is reset.

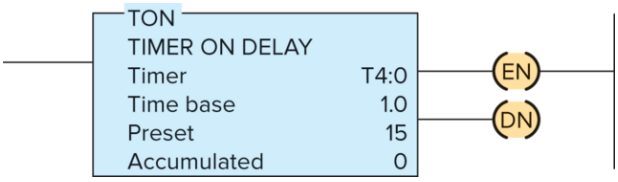
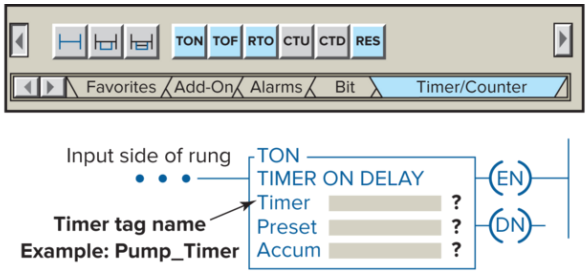


Figure 7-11 On-delay timer instruction.

The timer instruction for the RSLogix 500 and RSLogix 5000 processors operates in exactly the same manner. **Figure 7-12** shows the timer selection toolbar, on-delay timer instruction, and timer data **structure** for an RSLogix 5000 controller.

- Logix processors use a tag name, such as Pump_Motor, instead of a timer number.
- This descriptive tag name makes it easier to know what function the timer serves in the control system.
- The time base is fixed at 0.001 s $(1$ ms$)$. Therefore there is no parameter field.
- The associated timer data (PRE, ACC, EN, TT, DN) are found within the expanded timer structure.



Name	Data Type	Style	Description
Pump_Motor	Timer		Delay before starting motor
Motor_Delay.PRE	DINT	Decimal	
Motor_Delay.ACC	DINT	Decimal	
Motor_Delay.EN	BOOL	Decimal	
Motor_Delay.TT	BOOL	Decimal	
Motor_Delay.DN	BOOL	Decimal	

Figure 7-12 RSLogix 5000 timer instruction.

The on-delay timer (TON) is the most commonly used timer. **Figure 7-13** shows a PLC program that uses an on-delay timer. The operation of the program can be summarized as follows:

- The timer is activated by input switch A.
- When input switch A is closed (true or set to 1), the processor starts timer T4:0 timing and sets the EN and TT bits to true or 1.
- This turns ON outputs B and C.
- The accumulated value increases in one-second time base intervals.
- When the accumulated time equals the preset time $(10$ s$)$, the DN bit is set to 1, output D is turned ON, the TT bit is reset to 0 and output C is turned OFF.

- As long as input switch A remains closed the EN bit is set to 1 and output B will be ON.
- If input switch A is opened at any time before or after the timer has timed out, the accumulated time is automatically reset to 0 and output B is turned OFF.
- This timer configuration is termed *nonretentive* because any loss of continuity to the timer causes the timer instruction to reset.
- This timing operation is that of an on-delay timer because output D is switched on 10 s after the switch has been actuated from the off to the on position.

Timer ON DELAY (TON)		
TON Instruction OFF	Enable Bit (EN)	0
	Timer Timing Bit (TT)	0
	Done Bit (DN)	0
	Accumulating	NO
TON Instruction ON	Enable Bit (EN)	1
	Timer Timing Bit (TT)	1
	Done Bit (DN)	0
	Accumulating	YES
Timed Out Accum = Preset	Enable Bit (EN)	1
	Timer Timing Bit (TT)	0
	Done Bit (DN)	1
	Accumulating	NO
Instruction OFF after timed out	Enable Bit (EN)	0
	Timer Timing Bit (TT)	0
	Done Bit (DN)	0
	Accumulating	Reset

Table showing how each bit is effected during the program operation.

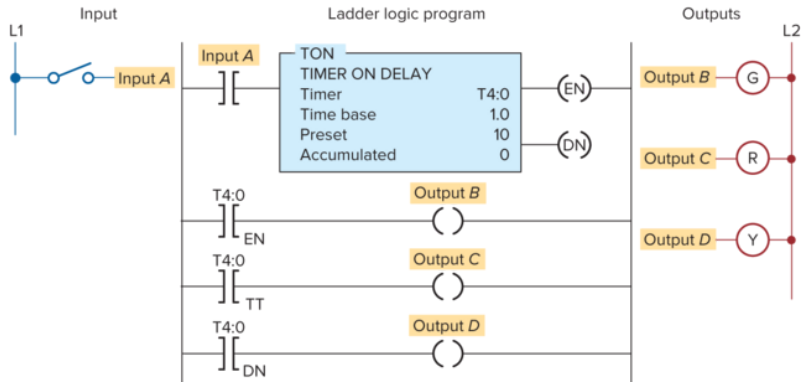


Figure 7-13 PLC on-delay timer program.

Figure 7-14 shows the timing diagram for the on-delay timer's control bits. The sequence of operation is as follows:

- The first true period of the timer rung shows the timer timing to 4 s and then going false.
- The timer resets, and both the timer-timing bit and the enable bit go false. The accumulated value also resets to 0.
- For the second true period input *A* remains true in excess of 10 s.
- When the accumulated value reaches 10 s, the done bit (DN) goes from false to true and the timer-timing bit (TT) goes from true to false.
- When input *A* goes false, the timer instruction goes false and also resets, at which time the control bits are all reset and the accumulated value resets to 0.

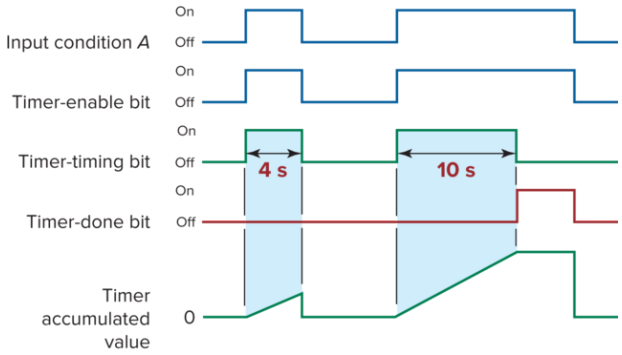


Figure 7-14 Timing diagram for an on-delay timer.

The timer table for an Allen-Bradley RSLogix 500 is shown in Figure 7-15. Addressing is done at three different levels: the element level, the word level, and the bit level. The timer uses three words per element. Each element consists of a control word, a preset word, and an accumulated word. Each word has 16 bits , which are numbered from 0 to 15. When addressing to the bit level, the address always refers to the bit within the word:

EN = Bit 15 enable

TT = Bit 14 timer timing

DN = Bit 13 done

Timer Table

	/EN	/TT	/DN	.PRE	.ACC
T4:0	0	0	0	10	0
T4:1	0	0	0	0	0
T4:2	0	0	0	0	0
T4:3	0	0	0	0	0
T4:4	0	0	0	0	0
T4:5	0	0	0	0	0

Address T4:0 Table: T4: Timer

Figure 7-15 RSLogix 500 timer table.

Timers may or may not have an instantaneous output (also known as the enable bit) signal associated with them. If an instantaneous output signal is required from a timer and it is not provided as part of the timer instruction, an equivalent instantaneous contact instruction can be programmed using an internally referenced relay coil. **Figure 7-16** shows an application of this technique. The operation of the program can be summarized as follows:

- According to the hardwired relay circuit diagram, coil M is to be energized 5 s after the start pushbutton is pressed.
- Contact TD-1 is the instantaneous contact, and contact TD-2 is the timed contact.
- The ladder logic program shows that a contact instruction referenced to an internal relay is now used to operate the timer.
- The instantaneous contact is referenced to the internal relay coil, whereas the time-delay contact is referenced to the timer output coil.

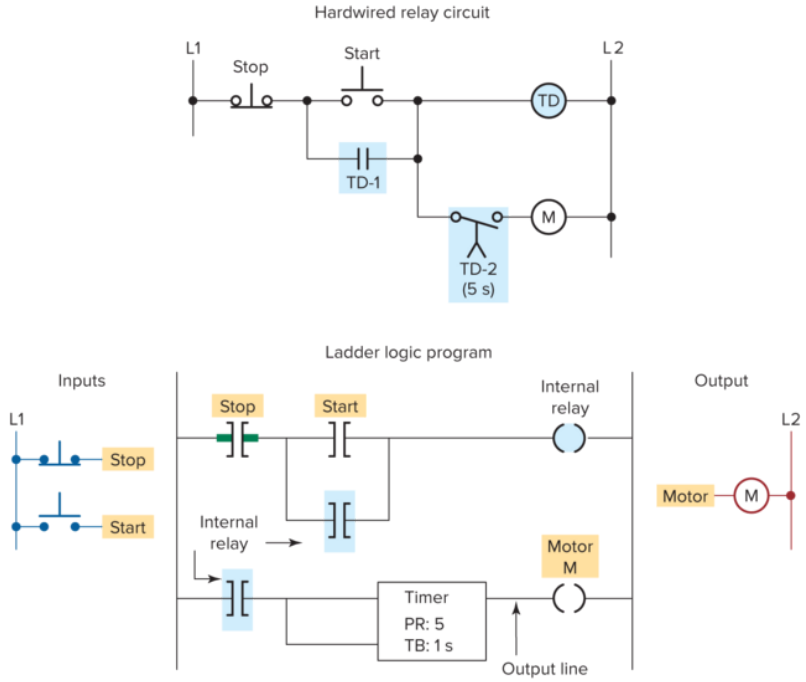


Figure 7-16 Instantaneous contact instruction can be programmed using an internally referenced relay coil.

Figure 7-17 shows an application for an on-delay timer that uses an NCTO contact. This circuit is used as a warning signal when moving equipment, such as a conveyor motor, is about to be started. The operation of the circuit can be summarized as follows:

- According to the hardwired relay circuit diagram, coil CR is energized when the start pushbutton PB1 is momentarily actuated.
- As a result, contact CR-1 closes to seal in CR coil, contact CR-2 closes to energize timer coil TD, and contact CR-3 closes to sound the horn.

- After a 10-s time-delay period, timer contact TD-1 opens to automatically switch the horn off.
- The ladder logic program shows how an equivalent circuit could be programmed using a PLC.
- The logic on the last rung is the same as the timer-timing bit and as such can be used with timers that do not have a timer-timing output.

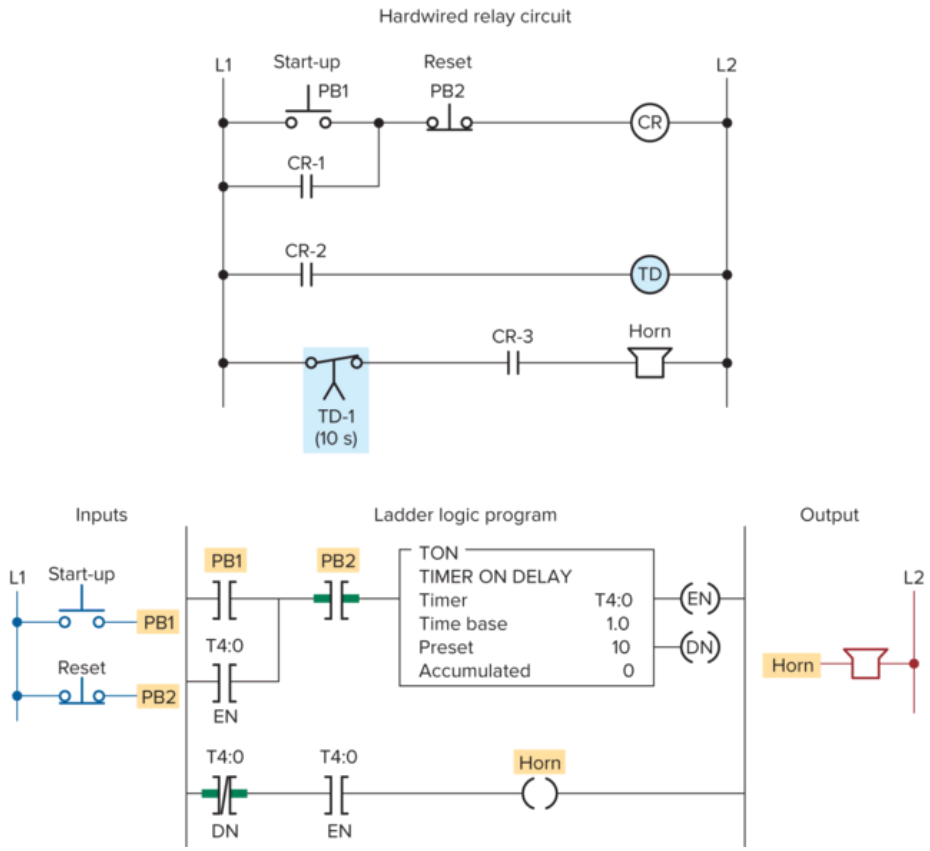


Figure 7-17 Conveyor warning signal circuit.

Timers are often used as part of **automatic sequential control** systems. **Figure 7-18** shows how a series of motors can be started automatically with only one start/stop control station. The operation of the circuit can be summarized as follows:

- According to the relay ladder schematic, lube-oil pump motor starter coil M1 is energized when the start pushbutton PB2 is momentarily actuated.
- As a result, M1-1 control contact closes to seal in M1, and the lube-oil pump motor starts.
- When the lube-oil pump builds up sufficient oil pressure, the lube-oil pressure switch PS1 closes.
- This in turn energizes coil M2 to start the main drive motor and energizes coil TD to begin the time-delay period.
- The PLC program shown has been constructed using RSLogix 5000 programming software.
- After the preset time-delay period of 15 s, TD-1 contact closes to energize coil M3 and start the feed motor.

- The ladder logic program shows how an equivalent circuit could be programmed using a PLC. The enable bit is used to seal in the timer so it continues to time until its preset value equals the accumulated value. The program sequence is reset by actuating the stop button PB1.

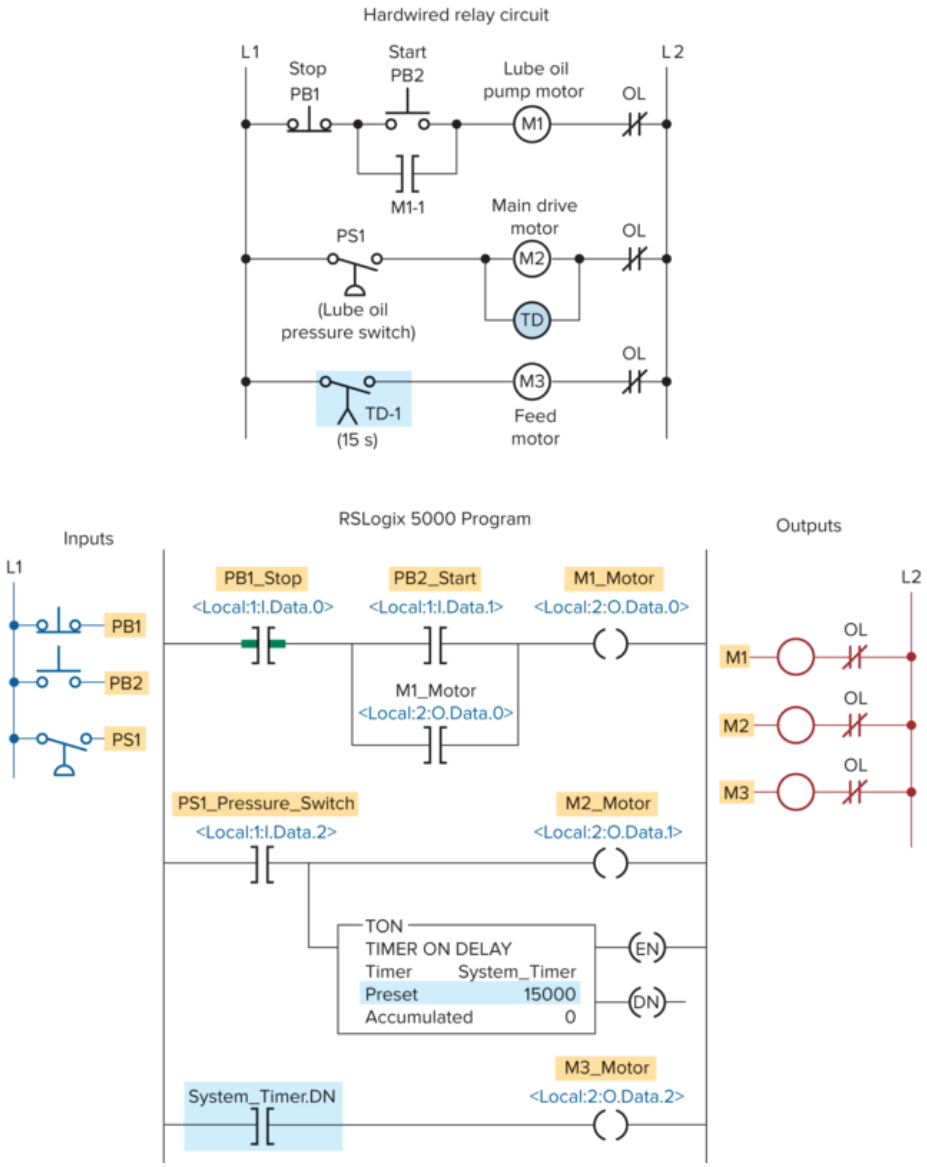


Figure 7-18 Automatic sequential control system.

OceanofPDF.com

7.4 Off-Delay Timer Instruction

The *off-delay timer (TOF)* operation will keep the output energized for a time period after the rung containing the timer has gone false. **Figure 7-19** illustrates the programming of an off-delay timer that uses the RSLogix 500 TOF timer instruction. TOF starts timing when the instruction goes from ON to OFF or from true to false. The operation of the circuit can be summarized as follows:

- When the switch connected to input I:1/0 is first closed, timed output O:2/1 is set to 1 immediately and the lamp is switched on.
- If this switch is now opened, logic continuity is lost and the timer begins counting.
- After 15 s, when the accumulated time equals the preset time, the output is reset to 0 and the lamp switches off.
- If logic continuity is gained before the timer is timed out, the accumulated time is reset to 0. For this reason, this timer is also classified as nonretentive.

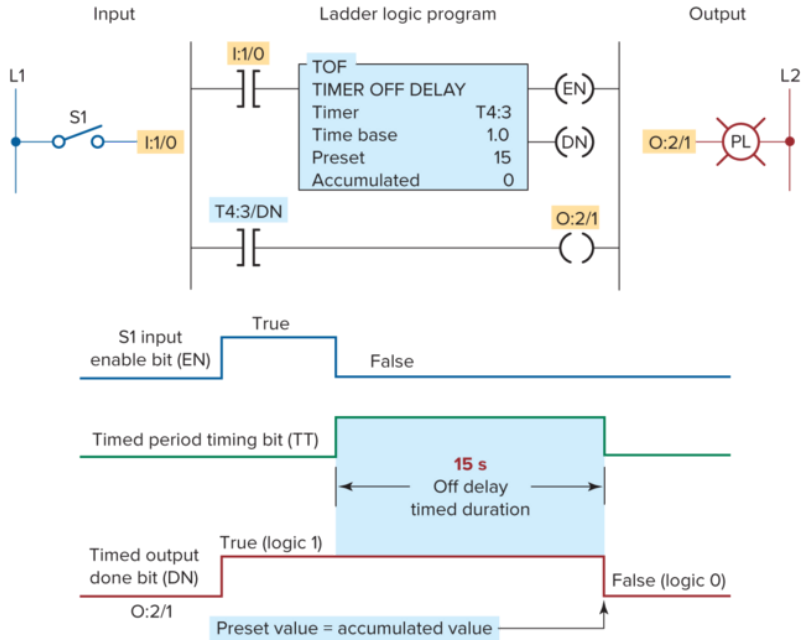


Figure 7-19 Off-delay programmed timer.

Figure 7-20 illustrates the use of an off-delay timer instruction used to switch motors *off* sequentially at 5 second intervals. The operation of the program can be summarized as follows:

- Timer preset values for T4:1, T4:2, and T4:3 are set for 5, 10, and 15 s, respectively.
- Closing the input switch SW immediately sets the done bit of each of the three off-delay timers to 1, immediately turning on motors M1, M2, and M3.
- If SW is then opened, logic continuity to all three timers is lost and each timer begins counting.
- Timer T4:1 times out after 5 s resetting its done bit to zero to de-energize motor M1.

- Timer T4:2 times out 5 s later resetting its done bit to zero to de-energize motor M2.
- Timer T4:3 times out 5 s later resetting its done bit to zero to de-energize motor M3.

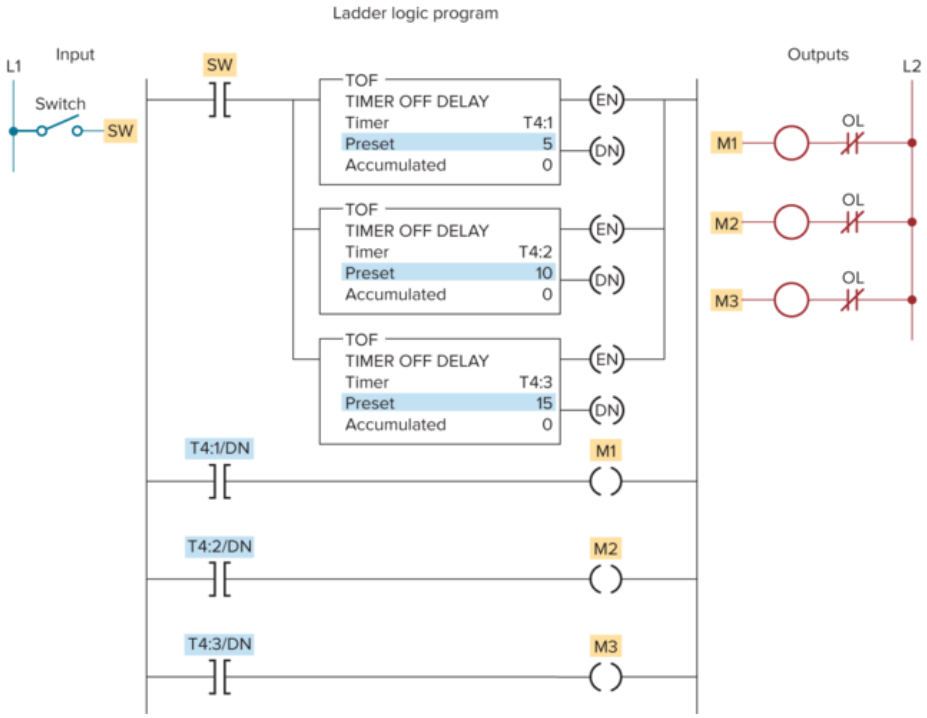


Figure 7-20 Program for switching motors off at 5 s intervals.

Timer OFF DELAY (TOF)		
TOF Instruction ON	Enable Bit (EN)	1
	Timer Timing Bit (TT)	0
	Done Bit (DN)	1
	Accumulating	NO
TOF Instruction OFF	Enable Bit (EN)	0
	Timer Timing Bit (TT)	1
	Done Bit (DN)	1
	Accumulating	YES
Timed Out Accum = Preset	Enable Bit (EN)	0
	Timer Timing Bit (TT)	0
	Done Bit (DN)	0
	Accumulating	NO
Instruction enabled again after timed out	Enable Bit (EN)	1
	Timer Timing Bit (TT)	0
	Done Bit (DN)	1
	Accumulating	Reset

Table showing how each bit is effected during the program operation.

Figure 7-21 shows a hardwired off-delay timer relay circuit with both instantaneous and timed contacts. The operation of the circuit can be summarized as follows:

- When power is first applied (limit switch LS open), motor starter coil M1 is energized and the green pilot light is on.
- At the same time, motor starter coil M2 is de-energized, and the red pilot light is off.

Page 131

- When limit switch LS closes, off-delay timer coil TD energizes.
- As a result, timed contact TD-1 opens to de-energize motor starter coil M1, timed contact TD-2 closes to energize motor starter coil M2, instantaneous contact TD-3 opens to switch the green light off, and instantaneous contact TD-4 closes to switch the red light on. The circuit remains in this state as long as limit switch LS1 is closed.
- When limit switch LS1 is opened, the off-delay timer coil TD de-energizes and the time-delay period is started.

Page 132

- Instantaneous contact TD-3 closes to switch the green light on, and instantaneous contact TD-4 opens to switch the red light off.
- After a 5-s time-delay period, timed contact TD-1 closes to energize motor starter M1, and timed contact TD-2 opens to de-energize motor starter M2.

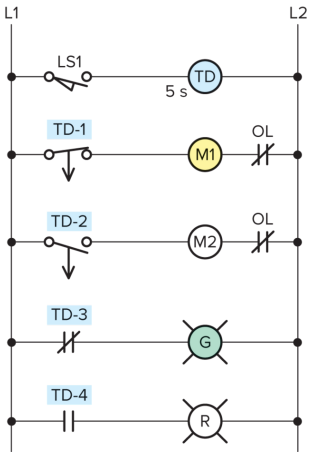


Figure 7-21 Hardwired off-delay timer relay circuit with both instantaneous and timed contacts.

Figure 7-22 shows an equivalent RSLogix 5000 PLC program of the hardwired off-delay timer relay circuit containing both instantaneous and timed contacts. The timer instruction carries out all of the functions of the original physical timer.

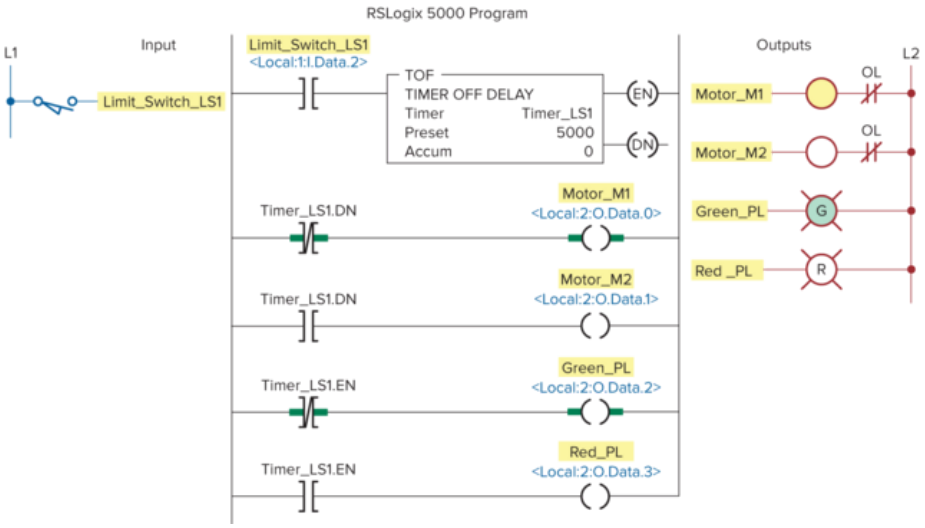


Figure 7-22 Equivalent PLC program of the hardwired off-delay timer relay circuit containing both instantaneous and timed contacts.

Figure 7-23 shows a program that uses both the on-delay and the off-delay timer instruction. The process involves pumping fluid from tank A to tank B. The operation of the process can be summarized as follows:

- Before starting, PS1 must be closed.
- When the start button is pushed, the pump starts. The button can then be released and the pump continues to operate.
- When the stop button is pushed, the pump stops.
- PS2 and PS3 must be closed 5 s after the pump starts. If either PS2 or PS3 opens, the pump will shut off and will not be able to start again for another 14 s.

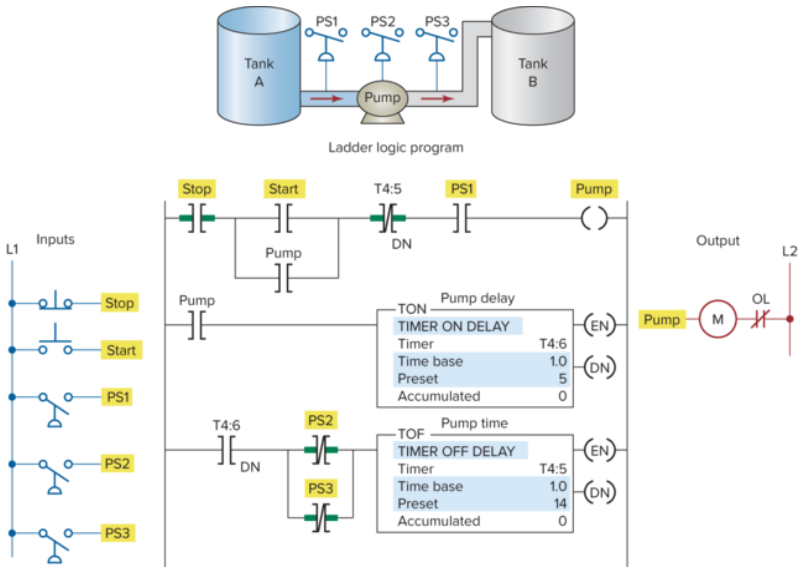


Figure 7-23 Fluid pumping process.

OceanofPDF.com

7.5 Retentive Timer

A ***retentive timer*** accumulates time whenever the device receives power, and it maintains the current time should power be removed from the device. When the timer accumulates time equal to its preset value, the contacts of the device change state. Loss of power to the timer after reaching its preset value does not affect the state of the contacts. The retentive timer must be intentionally reset with a separate signal for the accumulated time to be reset and for the contacts of the device to return to its nonenergized state.

Figure 7-24 illustrates the action of a motor-driven, electromechanical retentive timer used in some appliances. The shaft-mounted cam is driven by a motor. Once power is applied, the motor starts turning the shaft and cam. The positioning of the lobes of the cam and the gear reduction of the motor determine the time it takes for the motor to turn the cam far enough to activate the contacts. If power is removed from the motor, the shaft stops but *does not reset.*

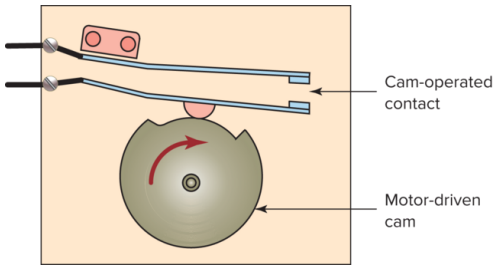


Figure 7-24 Electromechanical retentive timer.

A PLC retentive timer is used when you want to retain accumulated time values through power loss or the change in the rung state from true to false. The PLC-programmed retentive on-delay timer (RTO) is programmed in a manner similar to the nonretentive on-delay timer (TON), with one major exception—a retentive timer reset (RES) instruction. Unlike the TON, the RTO will hold its accumulated value when the timer rung goes false and will continue timing where it left off when the timer rung goes true again. This timer must be accompanied by a timer reset instruction to reset the accumulated value of the timer to 0. The RES instruction is the only automatic means of resetting the accumulated value of a retentive timer. The RES instruction has the same address as the timer it is to reset. Whenever the RES instruction is true, both the timer accumulated value and the timer done bit (DN) are reset to 0. **Figure 7-25** shows a PLC program for a retentive on-delay timer. The operation of the program can be summarized as follows:

- The timer will start to time when time pushbutton PB1 is closed.
- If the pushbutton is closed for 3 s and then opened for 3 s, the timer accumulated value will remain at 3 s.
- When the time pushbutton is closed again, the timer picks up the time at 3 s and continues timing.
- When the accumulated value (9) equals the preset value (9), the timer done bit T4:2/DN is set to 1 and the pilot light output PL is switched on.
- Whenever the momentary reset pushbutton is closed, the timer accumulated value is reset to 0.

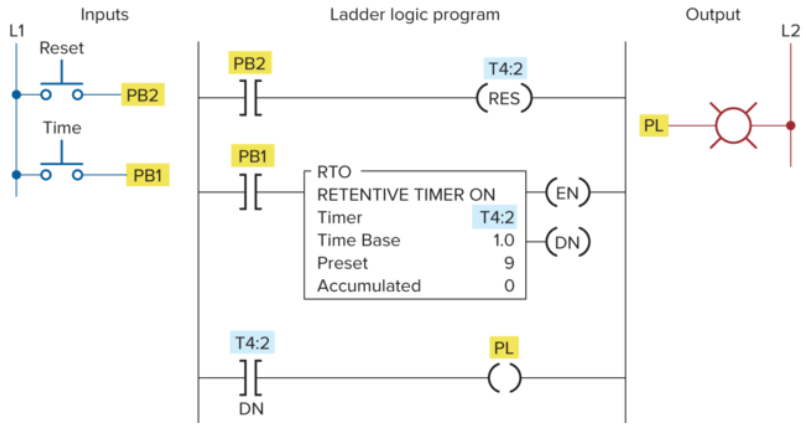


Figure 7-25 Retentive on-delay timer program.

Figure 7-26 shows a timing chart for the retentive on-delay timer program. The timing operation can be summarized as follows:

- When the timing rung is true (PB1 closed), the timer will commence timing.
- If the timing rung goes false, the timer will stop timing but will recommence timing for the stored accumulated value each time the rung goes true.
- When the reset PB2 is closed, the T4:2/DN bit is reset to 0 and turns the pilot light output off. The accumulated value is also reset and held at zero until the reset pushbutton is opened.

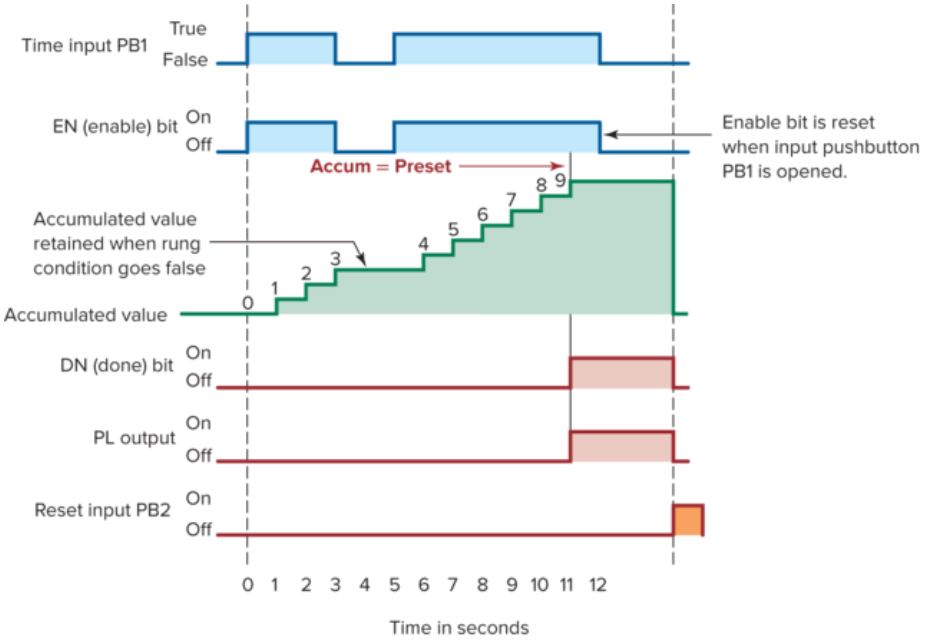


Figure 7-26 Retentive on-delay timer timing chart.

The RSLogix 5000 program drawn in **Figure 7-27** illustrates a practical application for an RTO. The purpose of the RTO timer is to detect whenever a piping system has sustained a *cumulative*

overpressure condition for 60 s. At that point, a horn is sounded automatically to call attention to the malfunction. When they are alerted, maintenance personnel can silence the alarm by switching the key switch S1 to the reset (contact closed) position. After the problem has been corrected, the alarm system can be reactivated by switching the key switch to open contact position.

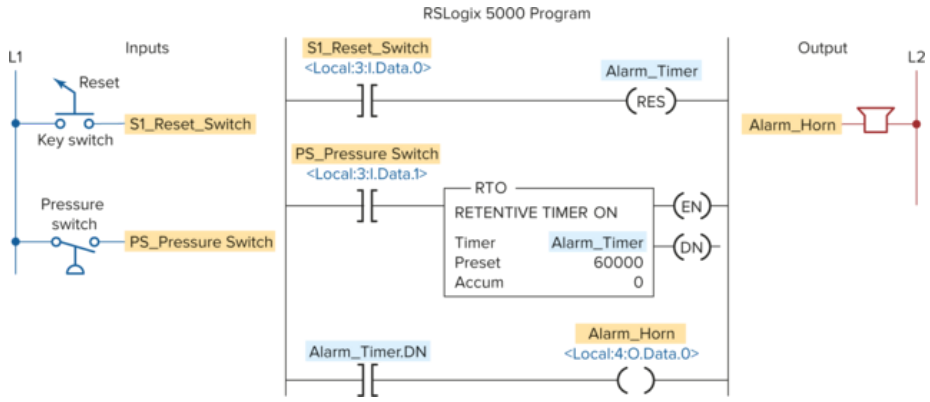


Figure 7-27 Retentive on-delay timer alarm program.

Figure 7-28 shows a practical application that uses the on-delay, off-delay, and retentive on-delay instructions in the same program. In this industrial application, there is a machine with a large steel shaft supported by babbitted bearings. This shaft is coupled to a large electric motor. The bearings need lubrication, which is supplied by an oil pump driven by a small electric motor. The operation of the program can be summarized as follows:

- To start the machine, the operator turns SW on.
- Before the *motor* shaft starts to turn, the bearings are supplied with oil by the *pump* for 10 seconds.
- The bearings also receive oil when the machine is running.
- When the operator turns SW off to stop the machine, the oil pump continues to supply oil for 15 s.
- A retentive timer is used to track the total running time of the pump. When the total running time is 3 hours, the motor is shut down and a pilot light is turned on to indicate that the filter and oil need to be changed.
- A reset button is provided to reset the process after the filter and oil have been changed.

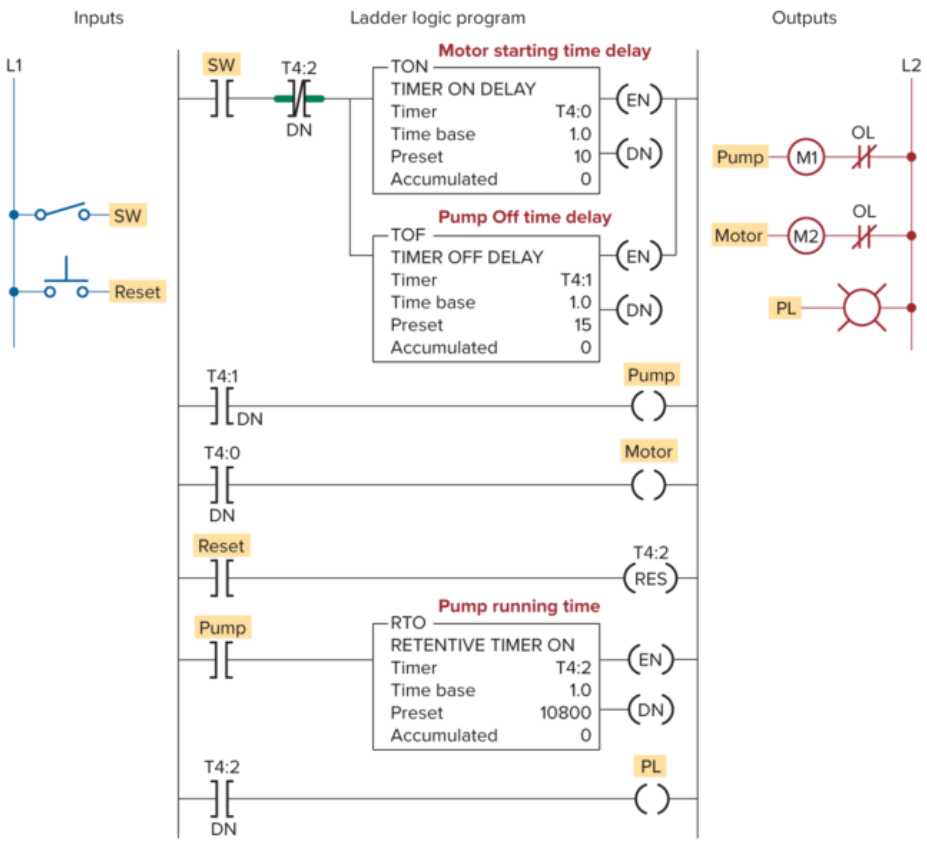


Figure 7-28 Bearing lubrication program.

Retentive timers do not have to be timed out completely to be reset. Rather, such a timer can be reset at any time during its operation. Note that the reset input to the timer will override the control input of the timer even though the control input to the timer has logic continuity.

OceanofPDF.com

7.6 Cascading Timers

When one timer's output triggers another timer's input, those timers are referred to as **cascaded**. Timers can be interconnected, or cascaded, to satisfy a number of logic control functions.

Figure 7-29 shows how three motors can be started automatically in sequence with a 20 s time delay between each using two hardwired on-delay timers. The operation of the circuit can be summarized as follows:

- Motor starter coil M1 is energized when the momentary start pushbutton PB2 is actuated.
- As a result, motor 1 starts, contact M1-1 closes to seal in M1, and timer coil TD1 is energized to begin the first time-delay period.
- After the preset time period of 20 s, TD1-1 contact closes to energize motor starter coil M2.
- As a result, motor 2 starts and timer coil TD2 is energized to begin the second time-delay period.
- After the preset time period of 20 s, TD2-1 contact closes to energize motor starter coil M3, and so motor 3 starts.

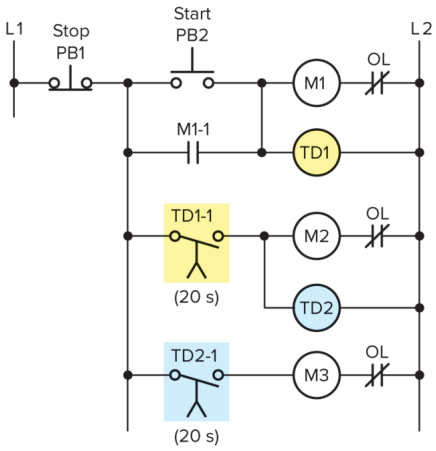


Figure 7-29 Hardwired sequential time-delayed motor-starting circuit.

Figure 7-30 shows an equivalent PLC program of the hardwired sequential time-delayed motor-starting circuit. Two programmed on-delay timers are cascaded together to obtain the same logic as the original hardwired timer relay circuit. Note that the output of timer T4:1 is used to control the input logic to timer T4:2.

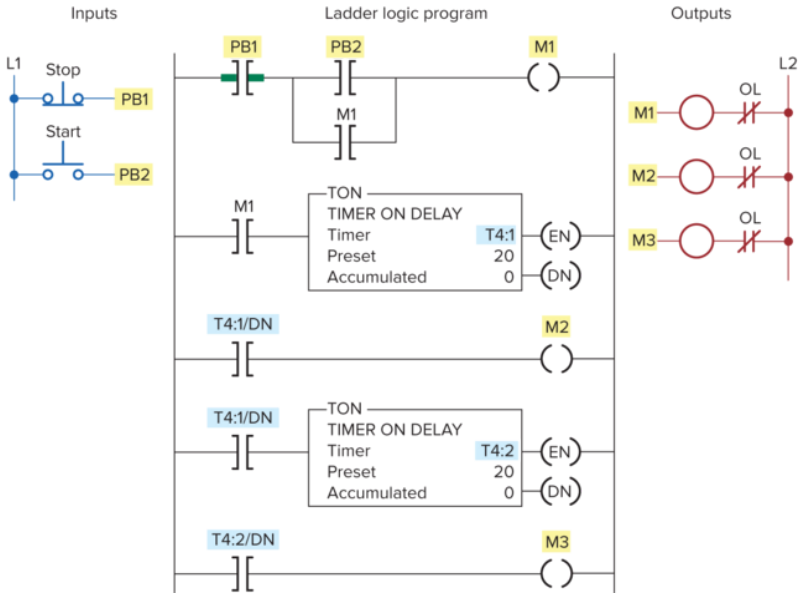


Figure 7-30 Equivalent PLC program of the sequential time-delayed motor-starting circuit.

Reciprocating timers are defined as timing functions where the output of one timer is used to reset the input of a second timer, each resetting the other. These types of timers are used in situations where a constant cycling of an output is required. For example, if a flashing light is required in the event of a control system failure, a program with reciprocating timers could be used to create the flashing output function.

Two timers can be interconnected to form an oscillator or reciprocating circuit. The oscillator logic is basically a timing circuit programmed to generate periodic output pulses of any duration. **Figure 7-31** shows the RSLogix 5000 program for an annunciator flasher circuit. Two internal timers form the oscillator circuit, which generates a timed, pulsed output. The oscillator circuit output is programmed in series with the alarm condition. If the alarm condition (temperature, pressure, or limit switch) is true, the appropriate output indicating light will flash. Note that any number of alarm conditions could be programmed using the same flasher circuit.

Page 137

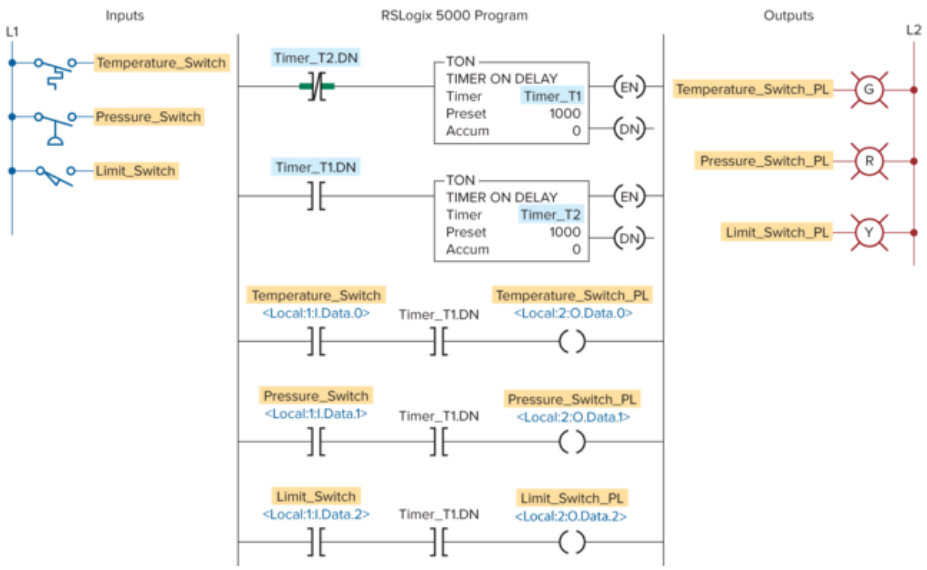


Figure 7-31 Annunciator flasher program.

At times you may require a time-delay period longer than the maximum preset time allowed for the single timer instruction of the PLC being used. When this is the case, the problem can be solved by simply cascading timers, as illustrated in **Figure 7-32**. The operation of the program can be summarized as follows:

- The total time-delay period required is 42,000 s.
- The first timer, T4:1, is programmed for a preset time of 30,000 s and begins timing when input SW is closed.
- When T4:1 completes its time-delay period 30,000 s later, the T4:1/DN bit will be set to 1.
- This in turn activates the second timer, T4:2, which is preset for the remaining 12,000 s of the total 42,000-s time delay.
- Once T4:2 reaches its preset time, the T4:2/DN bit will be set to 1, which switches on the output PL, the pilot light, to indicate the completion of the full 42,000-s time delay.
- Opening input SW at any time will reset both timers and switch output PL off.

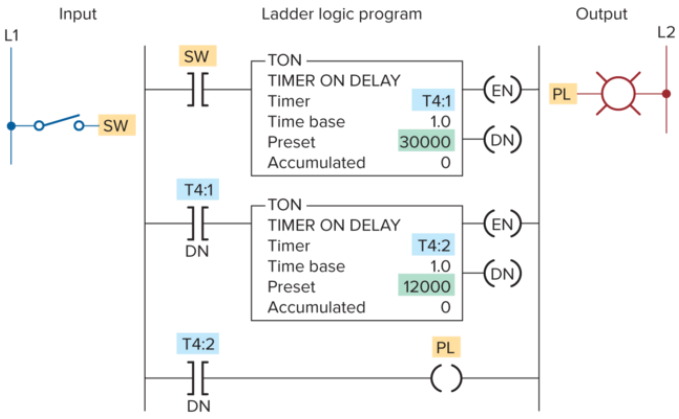


Figure 7-32 Cascading of timers for longer time delays.

A typical application for a PLC circuit combining both cascading and reciprocating functions would be the control of traffic signals. The ladder logic circuit of **Figure 7-33** illustrates a control of a set of traffic lights in one direction. The operation of the program can be summarized as follows:

Page 138

- Transition from red light to green light to amber light is accomplished by the interconnection of the three TON timer instructions.
- The input to timer T4:0 is controlled by the T4:2 done bit.
- The input to timer T4:1 is controlled by the T4:0 done bit.
- The input rung to timer T4:2 is controlled by the T4:1 done bit.
- The timed sequence of the lights is:

  Red–30 s on

  Green–25 s on

  Amber–5 s on
- The sequence then repeats itself.

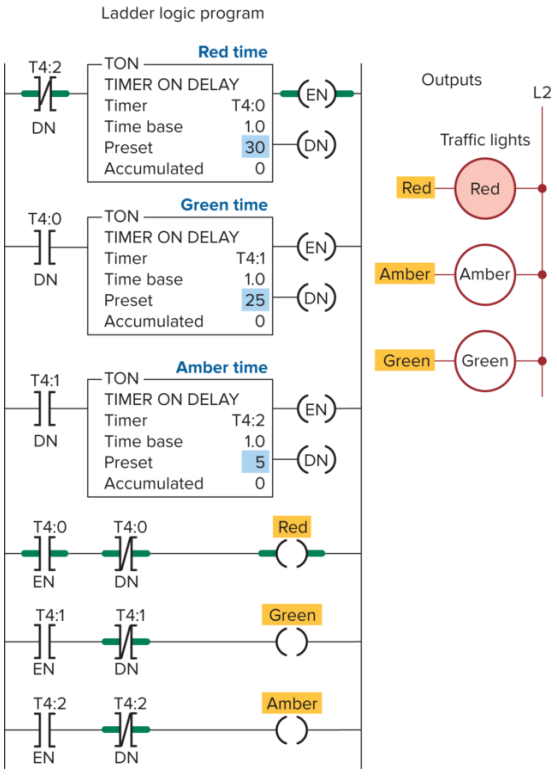


Figure 7-33 Control of traffic lights in one direction.

The chart shown in **Figure 7-34** shows the timed sequence of the lights for two-directional control of traffic lights.



Figure 7-34 Timing chart for two-directional control of traffic lights.

Figure 7-35 shows the original traffic light program modified to include three more lights that control traffic flow in two directions.

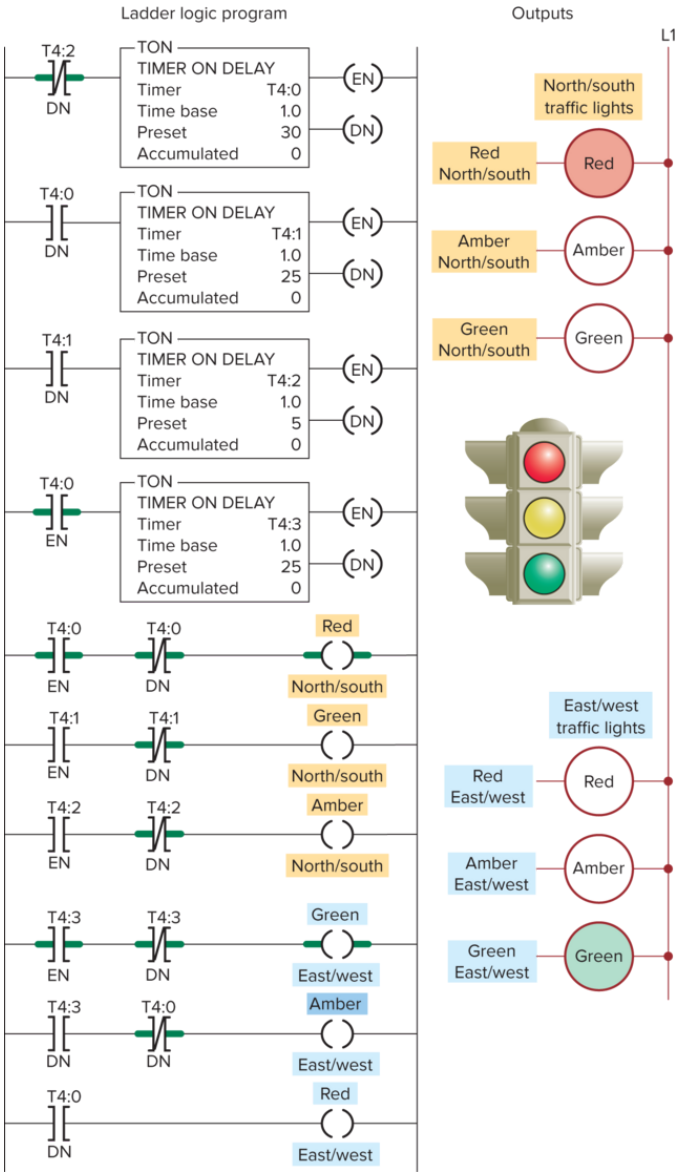


Figure 7-35 Control of traffic lights in two directions.

OceanofPDF.com

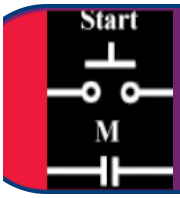


CHAPTER 7 REVIEW QUESTIONS

1. Explain the difference between the timed and instantaneous contacts of a mechanical timing relay.
2. Draw the symbol and explain the operation of each of the following timed contacts of a mechanical timing relay:
   a. On-delay timer—NOTC contact
   b. On-delay timer—NCTO contact
   c. Off-delay timer—NOTO contact
   d. Off-delay timer—NCTC contact
3. Name five pieces of information usually associated with a PLC timer instruction.
4. When is the output of a programmed timer energized?
5. What method is commonly used to represent a timer instruction within a PLC's ladder logic program?
6. a. Explain the difference between the operation of a nonretentive timer and that of a retentive timer.
   b. Explain how the accumulated count of programmed retentive and nonretentive timers is reset to zero.
7. State three advantages of using programmed PLC timers over mechanical timing relays.
8. For a TON timer:
   a. When is the enable bit of a timer instruction true?
   b. When is the timer-timing bit of a timer instruction true?
   c. When does the done bit of a timer change state?
9. For a TOF timer:
   a. When is the enable bit of a timer instruction true?
   b. When is the timer-timing bit of a timer instruction true?
   c. When does the done bit of a timer change state?
10. Explain what each of the following quantities associated with a PLC timer instruction represents:
    a. Preset time
    b. Accumulated time
    c. Time base
11. State the method used to reset the accumulated time of each of the following:
    a. TON timer
    b. TOF timer

c. RTO timer

OceanofPDF.com



CHAPTER 7 PROBLEMS

1. a. With reference to the relay schematic diagram in **Figure 7-36**, state the status of each light (on or off) after each of the following sequential events:

I. Power is first applied and switch S1 is open.

II. Switch S1 has just closed.

III. Switch S1 has been closed for 5 s.

IV. Switch S1 has just opened.

V. Switch S1 has been opened for 5 s.

b. Design a PLC program and prepare a typical I/O connection diagram and ladder logic program that will execute this hardwired control circuit correctly.

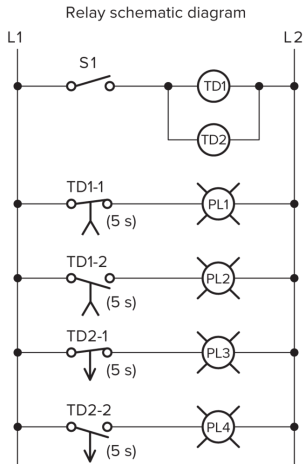


Figure 7-36 Relay schematic diagram for Problem 1.

2. Design a PLC program and prepare a typical I/O connection diagram and ladder logic program that will correctly execute the hardwired relay control circuit shown in **Figure 7-37**.

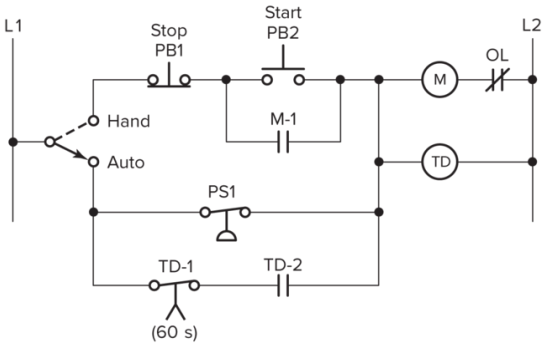


Figure 7-37 Hardwired relay control circuit for Problem 2.

3. Study the ladder logic program in **Figure 7-38** and answer the questions that follow:
   - **a.** What type of timer has been programmed?
   - **b.** What is the length of the time-delay period?
   - **c.** What is the value of the accumulated time when power is first applied?
   - **d.** When does the timer start timing?
   - **e.** When does the timer stop timing and reset itself?
   - **f.** When input LS1 is first closed, which rungs are true and which are false?
   - **g.** When input LS1 is first closed, state the status (on or off) of each output.
   - **h.** When the timer's accumulated value equals the preset value, which rungs are true and which are false?
   - **i.** When the timer's accumulated value equals the preset value, state the status (on or off) of each output.
   - **j.** Suppose that rung 1 is true for 5 s and then power is lost. What will the accumulated value of the counter be when power is restored?

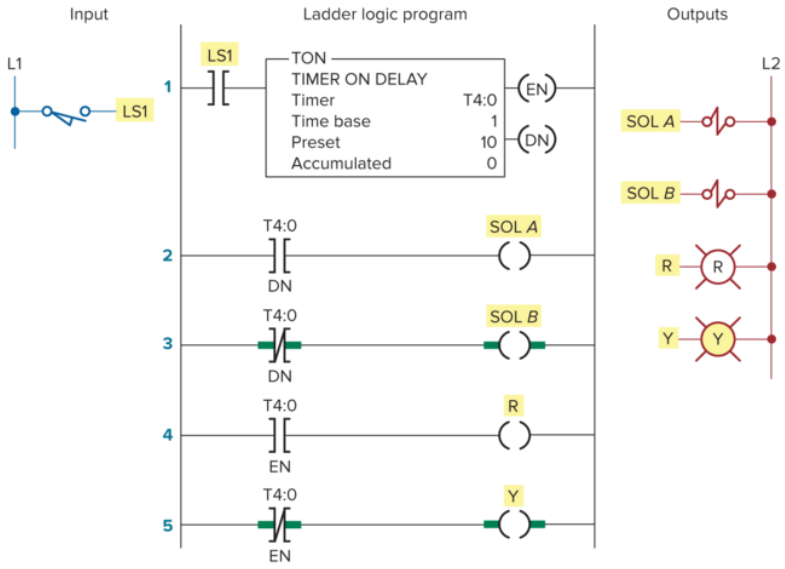


Figure 7-38 Ladder logic program for Problem 3.

4. Study the ladder logic program in **Figure 7-39** and answer the questions that follow:

a. What type of timer has been programmed?

b. What is the length of the time-delay period?

c. What is the value of the accumulated time when power is first applied?

d. When does the timer start timing?

e. When does the timer stop timing and reset itself?

f. When input LS1 is first closed, which rungs are true and which are false?

g. When input LS1 is first closed, state the status (on or off) of each output.

h. When the timer's accumulated value equals the preset value, which rungs are true and which are false?

i. When the timer's accumulated value equals the preset value, state the status (on or off) of each output.

j. Suppose that rung 1 is true for 5 s and then power is lost. What will the accumulated value of the counter be when power is restored?

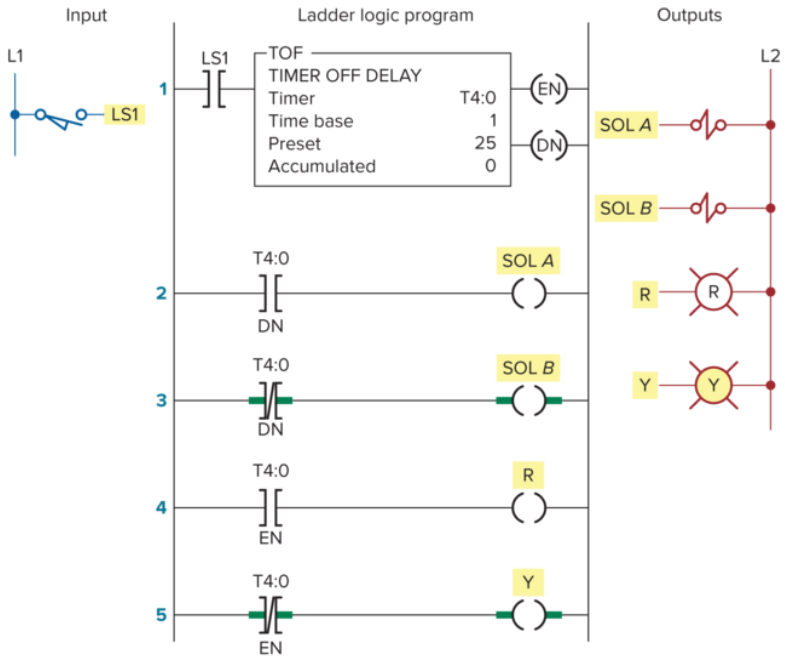


Figure 7-39 Ladder logic program for Problem 4.

5. Study the ladder logic program in **Figure 7-40**, and answer the questions that follow:

a. What type of timer has been programmed?

b. What is the length of the time-delay period?

c. When does the timer start timing?

Page 142

d. When is the timer reset?

e. When will rung 3 be true?

f. When will rung 5 be true?

g. When will output PL4 be energized?

h. Assume that your accumulated time value is up to 020 and power to your system is lost. What will your accumulated time value be when power is restored?

Page 143

i. What happens if inputs PB1 and PB2 are both true at the same time?

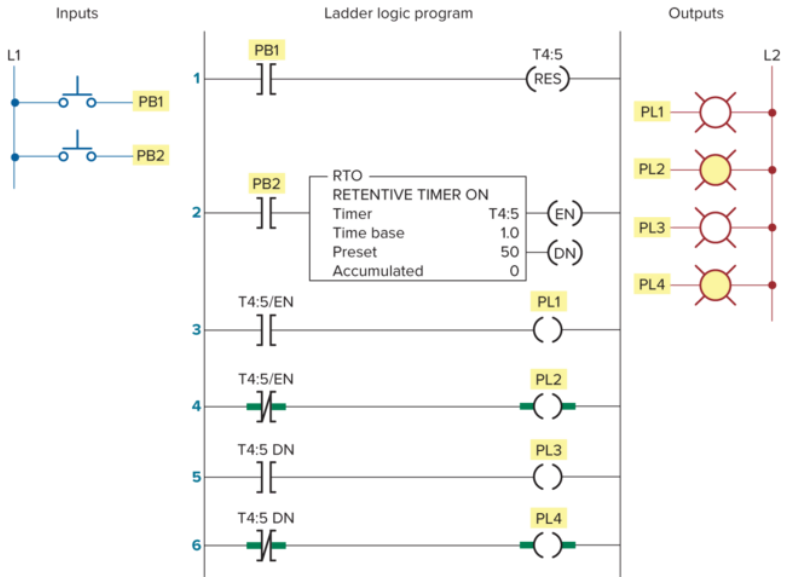


Figure 7-40 Ladder logic program for Problem 5.

6. Study the ladder logic program in **Figure 7-41** and answer the questions that follow:

   a. What is the purpose of interconnecting the two timers?

   b. How much time must elapse before output PL is energized?

   c. What two conditions must be satisfied for timer T4:2 to start timing?

   d. Assume that output PL is on and power to the system is lost. When power is restored, what will the status of this output be?

   e. When input PB2 is on, what will happen?

   f. When input PB1 is on, how much accumulated time must elapse before rung 3 will be true?

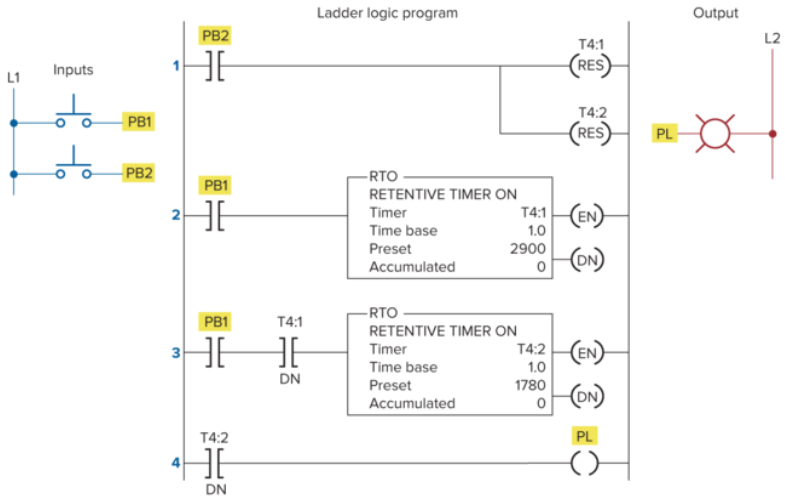


Figure 7-41 Ladder logic program for Problem 6.

7. You have a machine that cycles on and off during its operation. You need to keep a record of its total run time for maintenance purposes. Which timer would accomplish this?
8. Write a ladder logic program that will turn on a light, PL, 15 s after switch S1 has been turned on.
9. Study the on-delay timer ladder logic program in **Figure 7-42**, and from each of the conditions stated, determine whether the timer is reset, timing, or timed out or if the conditions stated are not possible.
   a. The input is true, and EN is 1, TT is 1, and DN is 0.
   b. The input is true, and EN is 1, TT is 1, and DN is 1.
   c. The input is false, and EN is 0, TT is 0, and DN is 0.
   d. The input is true, and EN is 1, TT is 0, and DN is 1.

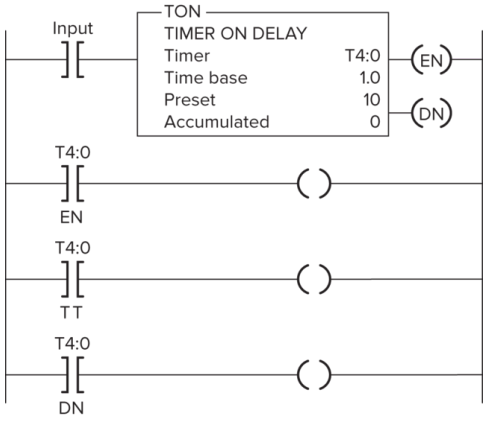


Figure 7-42 On-delay timer ladder logic program for Problem 9.

10. Study the off-delay timer ladder logic program in **Figure 7-43**, and from each of the conditions stated, determine whether the timer is reset, timing, or timed out or if the conditions stated are not possible.
    a. The input is true, and EN is 0, TT is 0, and DN is 1.
    b. The input is true, and EN is 1, TT is 1, and DN is 1.
    c. The input is true, and EN is 1, TT is 0, and DN is 1.
    d. The input is false, and EN is 0, TT is 1, and DN is 1.
    e. The input is false, and EN is 0, TT is 0, and DN is 0.

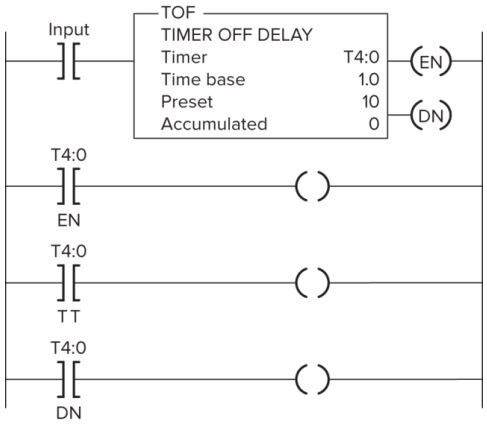


Figure 7-43 Off-delay timer ladder logic program for Problem 10.

11. Write a program for an "anti-tie down circuit" that will disallow a punch press solenoid from operating unless both hands are on the two palm start buttons. Both buttons must be pressed at the same time within 0.5 s. The circuit also will not allow the operator to tie down one of the buttons and operate the press with just one button. (Hint: Once either of the buttons is pressed, begin timing 0.5 s. Then, if both buttons are not pressed, prevent the press solenoid from operating.)
12. Modify the program for the control of traffic lights in two directions so that there is a 3-s period when both directions will have their red lights illuminated.
13. Write a program to implement the process illustrated in **Figure 7-44**. The sequence of operation is to be as follows:
    - Normally open start and normally closed stop pushbuttons are used to start and stop the process.
    - When the start button is pressed, solenoid *A* energizes to start filling the tank.
    - As the tank fills, the empty level sensor switch closes.
    - When the tank is full, the full level sensor switch closes.
    - Solenoid *A* is de-energized.
    - The agitate motor starts automatically and runs for 3 min to mix the liquid.
    - When the agitate motor stops, solenoid *B* is energized to empty the tank.
    - When the tank is completely empty, the empty sensor switch opens to de-energize solenoid *B*.
    - The start button is pressed to repeat the sequence.

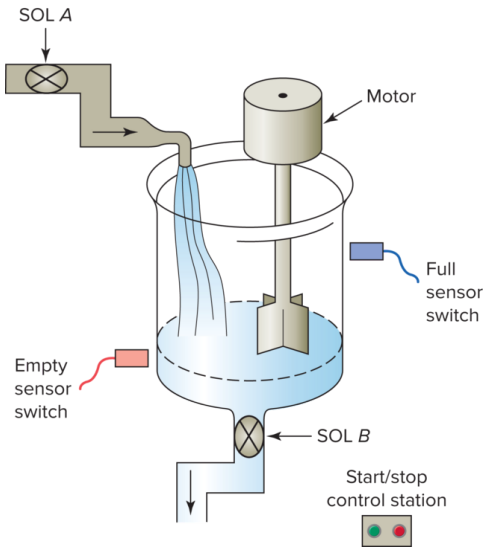


Figure 7-44 Process for Problem 13.

14. When the lights are turned off in a building, an exit door light is to remain on for an additional 2 min, and the parking lot lights are to remain on for an additional 3 min after the door light goes out. Write a program to implement this process.

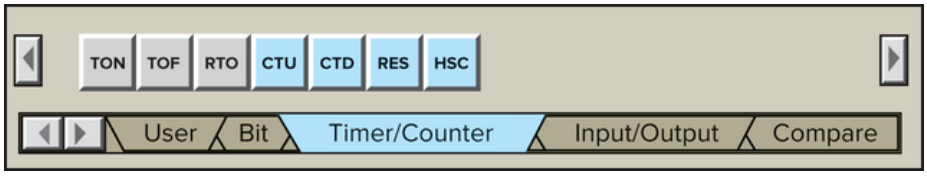
15. Write a program to simulate the operation of a sequential taillight system. The light system consists of three separate lights on each side of the car. Each set of lights will be activated separately, by either the left or right turn signal switch. There is to be a 1-s delay between the activation of each light, and a 1-s period when all the lights are off. Ensure that when both switches are on, the system will not operate. Use the least number of timers possible. The sequence of operation should be as follows:

- The switch is operated.
- Light 1 is illuminated.
- Light 2 is illuminated 1 s later.
- Light 3 is illuminated 1 s later.
- Light 3 is illuminated for 1 s.
- All lights are off for 1 s.
- The system repeats while the switch is on.

OceanofPDF.com

8

Programming Counters



Chapter Objectives

After completing this chapter, you will be able to:

- List and describe the functions of PLC counter instructions
- Describe the operating principle of a transitional, or one-shot, contact
- Analyze and interpret typical PLC counter ladder logic programs
- Apply the PLC counter function and associated circuitry to control systems
- Apply combinations of counters and timers to control systems

All PLCs include both up-counters and down-counters. Counter instructions and their function in ladder logic are explained in this chapter. Typical examples of PLC counters include the following: straight counting in a process, two counters used to give the sum of two counts, and two counters used to give the difference between two counts.

OceanofPDF.com

8.1 Counter Instructions

Programmed counters can serve the same function as mechanical counters. Figure 8-1 shows the construction of a simple **mechanical counter.** Every time the actuating lever is moved over, the counter adds one number; the actuating lever then returns automatically to its original position. Resetting to zero is done with a pushbutton located on the side of the unit.

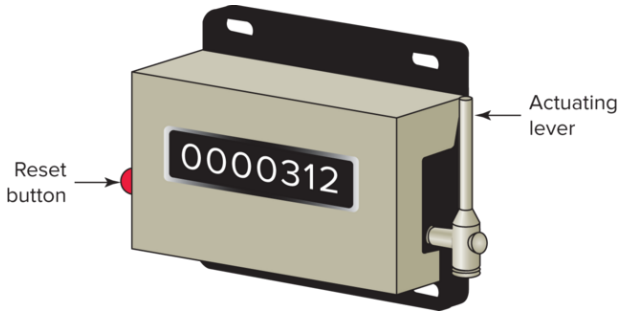


Figure 8-1 Mechanical counter.

Electronic counters can count up, count down, or be combined to count up and down. Although the majority of counters used in industry are up-counters, numerous applications require the implementation of down-counters or of combination up/down-counters.

All PLC manufacturers offer some form of counter instruction as part of their instruction set. One common counter application is keeping track of the number of items moving past a given point as illustrated in Figure 8-2.

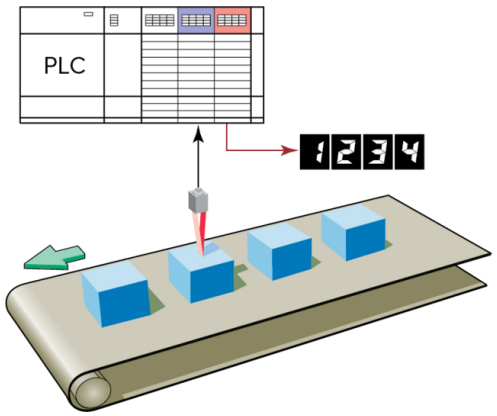


Figure 8-2 Counter application.

Counters are similar to timers except that they do not operate on an internal clock but are dependent on external or program sources for counting. The up-counter increments its accumulated value by 1 each time the counter rung makes a false-to-true transition. When the accumulated count equals the preset count the counter output is energized or set to 1. Shown as part of the instruction are the:

Counter type

Counter address

Counter preset value

Accumulated count

The counter reset instruction must be used in conjunction with the counter instruction. Up-counters are always reset to zero. Down-counters may be reset to zero or to some preset value. Some manufacturers include the reset function as a part of the general counter instruction, whereas others dedicate a separate instruction for resetting the counter.

Figure 8-3 shows a ***block-formatted*** counter. The instruction block indicates the type of counter (up or down), along with the counter's preset value and accumulated or current value. The counter has two input conditions associated with it, namely, the count and reset. All PLC counters operate, or count, on the leading edge of the input signal. The counter will either **increment** or **decrement** whenever the count input transfers from an off state to an on state. The counter will *not* operate on the trailing edge, or on-to-off transition, of the input condition.

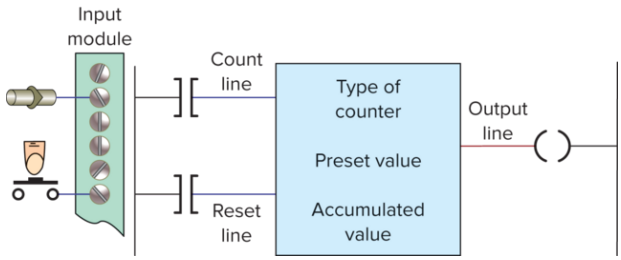


Figure 8-3 Block-formatted counter instruction.

Some manufacturers require the reset rung or line to be true to reset the counter, whereas others require it to be false to reset the counter. For this reason, it is wise to consult the PLC's operations manual before attempting any programming of counter circuits.

PLC counters are normally retentive; that is, whatever count was contained in the counter at the time of a processor shutdown will be restored to the counter on power-up. The counter may be reset, however, if the reset condition is activated at the time of power restoration.

PLC counters can be designed to count up to a preset value or to count down to a preset value. The ***up-counter*** is incremented by 1 each time the rung containing the counter goes from false to true. The ***down-counter*** decrements by 1 each time the rung containing the counter is energized. These rung transitions can result from events occurring in the program, such as parts traveling past a sensor or actuating a limit switch. The preset value of a programmable controller counter can be set by the operator or can be loaded into a memory location as a result of a program decision.

Figure 8-4 illustrates the counting sequence of an up-counter and a down-counter. The value indicated by the counter is termed the ***accumulated value*****.** The counter will increment or decrement, depending on the type of counter, until the accumulated value of the counter is equal to or greater

than the preset value, at which time an output will be produced. A counter reset is always provided to cause the counter accumulated value to be reset to a predetermined value.

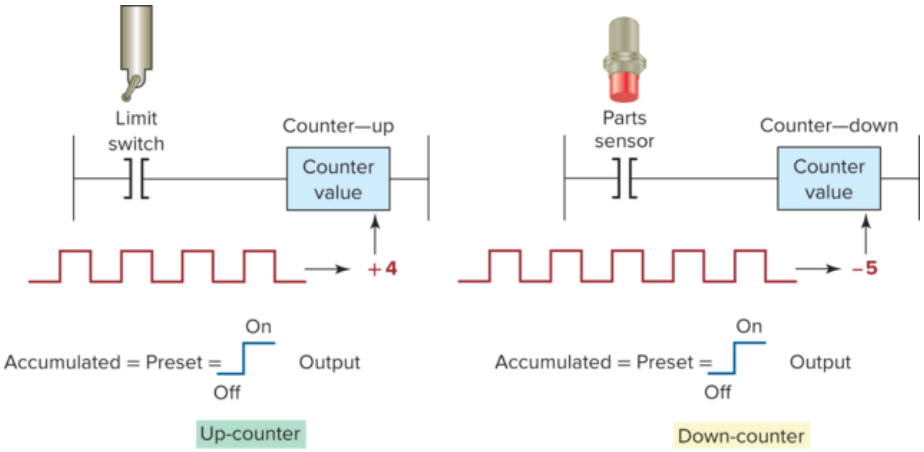


Figure 8-4 Counter counting sequence.

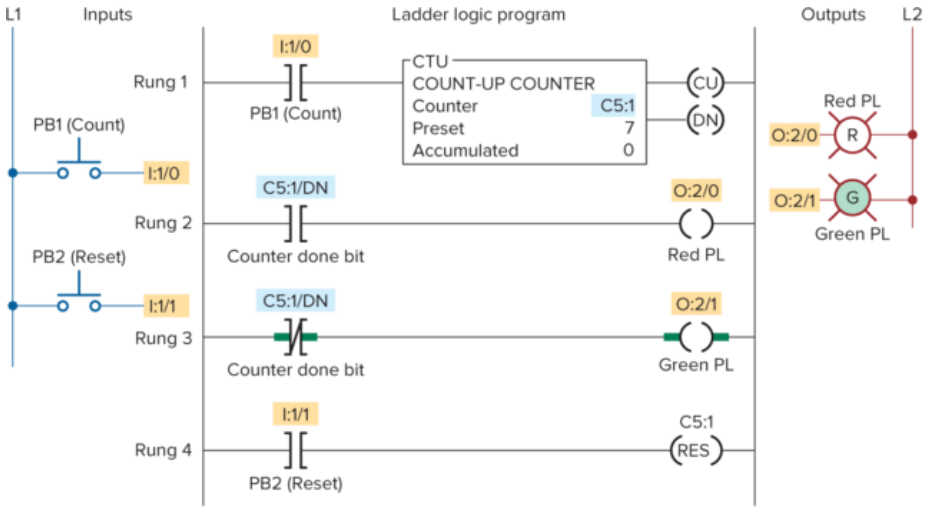
OceanofPDF.com

8.2 Up-Counter

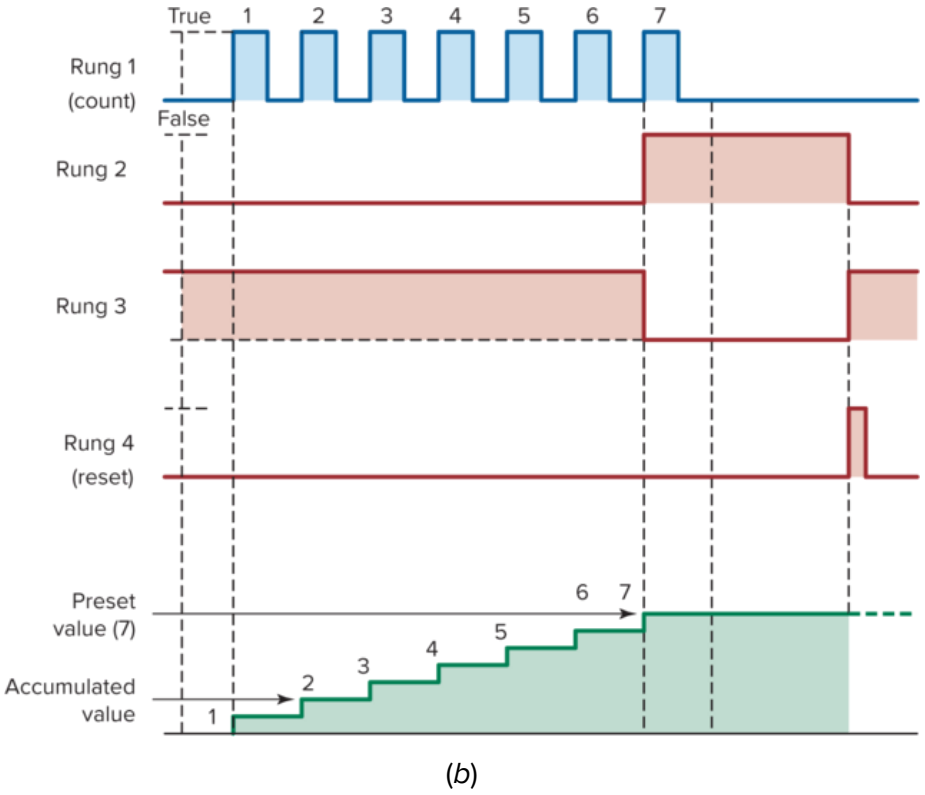
The up-counter is an output instruction whose function is to increment its accumulated value on false-to-true transitions of its instruction. It thus can be used to count false-to-true transitions of an input instruction and then trigger an event after a required number of counts or transitions. The up-counter output instruction will increment by 1 each time the counted event occurs.

Figure 8-5 shows the program and timing diagram for an RSLogix 500 **Count-Up Counter.** This control application is designed to turn the red pilot light on and the green pilot light off after an accumulated count of 7. The operation of the program can be summarized as follows:

- Operating pushbutton PB1 provides the off-to-on transition pulses that are counted by the counter.
- The preset value of the counter is set for 7.
- Each false-to-true transition of rung 1 increases the counter's accumulated value by 1.
- Output O:2/1 is energized as long as the accumulated value is less than 7.
- After 7 pulses, or counts, when the preset counter value equals the accumulated counter value, output DN is energized.
- As a result, rung 2 becomes true and energizes output O:2/0 to switch the red pilot light on.
- At the same time, rung 3 becomes false and de-energizes output O:2/1 to switch the green pilot light off.
- The counter is reset by closing pushbutton PB2, which makes rung 4 true and resets the accumulated count to zero.
- Counting can resume when rung 4 goes false again.



(*a*)



(*b*)

Figure 8-5 Simple up-counter program. (a) Program. (b) Timing diagram.

The RSLogix 500 counter file is file 5 (**Figure 8-6**). Each counter is composed of three 16-bit words, collectively called a counter element. These three data words are the control word, preset word, and accumulated word. Each of the three data words shares the same base address, which is the address of the counter itself. There can be up to 256 counter elements. Addresses for counter file 5, counter element 3 (C5:3), are listed below.

C5 = counter file 5

:3 = counter element 3 (0–255 counter elements per file)

C5:3/DN is the address for the done bit of the counter.

C5:3/CU is the address for the count-up enable bit of the counter.

C5:3/CD is the address for the count-down enable bit of the counter.

C5:3/OV is the address for the overflow bit of the counter.

C5:3/UN is the address for the underflow bit of the counter.

C5:3/UA is the address for the update accumulator bit of the counter. This instruction is only used with the High-Speed Counter (HSC) instruction.

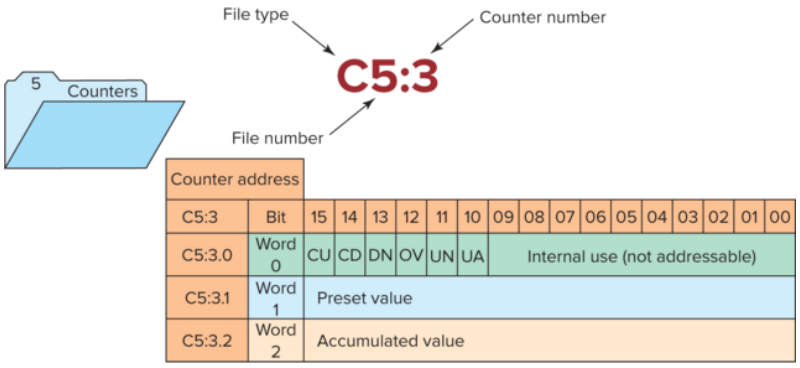


Figure 8-6 RSLogix 500 counter file.

Figure 8-7 shows the counter table for the RSLogix 500 controller. The ***control word*** uses status control bits consisting of the following:

Count-Up (CU) Enable Bit—The count-up enable bit is used with the count-up counter and is true whenever the count-up counter instruction is true. If the count-up counter instruction is false, the CU bit is false.

Count-Down (CD) Enable Bit—The count-down enable bit is used with the count-down counter and is true whenever the count-down counter instruction is true. If the count-down counter instruction is false, the CD bit is false.

Done (DN) Bit—The done bit is true whenever the accumulated value is equal to or greater than the preset value of the counter, for either the count-up or the count-down counter.

Overflow (OV) Bit—The overflow bit is true whenever the counter counts past its maximum value, which is 32,767. On the next count, the counter will wrap around to $-32{,}768$ and will continue counting from there toward 0 on successive false-to-true transitions of the count-up counter.

Underflow (UN) Bit—The underflow bit will go true when the counter counts below $-32{,}768$. The counter will wrap around to $+32{,}767$ and continue counting down toward 0 on successive false-to-true rung transitions of the count-down counter.

Update Accumulator (UA) Bit—The update accumulator bit is used only in conjunction with an external HSC (high-speed counter).

Counter Number—This number must come from the counter file. In the example shown, the counter number is C5:0, which represents counter file 5, counter 0 in that file. The address for this counter should not be used for any other count-up counter.

Preset Value—The preset value can range from $-32{,}768$ to $+32{,}767$. In the example shown, the preset value is 10.

Accumulated Value—The accumulated value can also range from $-32{,}768$ to $+32{,}767$. Typically, as in this example, the value entered in the accumulated word is 0. Regardless of what value is entered, the reset instruction will reset the accumulated value to 0.

Counter Table

	/CU	/CD	/DN	/OV	/UN	/UA	.PRE	.ACC
C5:0	0	0	0	0	0	0	0	0
C5:1	0	0	0	0	0	0	0	0
C5:2	0	0	0	0	0	0	0	0
C5:3	0	0	0	0	0	0	50	0
C5:4	0	0	0	0	0	0	0	0
C5:5	0	0	0	0	0	0	0	0

Address C5:3 Table: C5: Counter

Figure 8-7 RSLogix 500 counter table.

The ***preset value (PRE)*** *word* specifies the value that the counter must count to before it changes the state of the done bit. The preset value is the set point of the counter and ranges from $-32{,}768$ to $+32{,}767$. The number is stored in binary form, with any negative numbers being stored in 2's complement binary.

The ***accumulated value (ACC)*** *word* is the current count based on the number of times the rung goes from false to true. The accumulated value either increments with a false-to-true transition of the count-up counter instruction or decrements with a false-to-true transition of the count-down counter instruction. It has the same range as the preset: $-32{,}768$ to $+32{,}767$. The accumulated value will continue to count past the preset value instead of stopping at the preset like a timer does.

Figure 8-8 shows an example of the count-up counter and its status bits used in the RSLogix 500 instruction set. The address for counters begins at C5:0 and continues through C5:255.

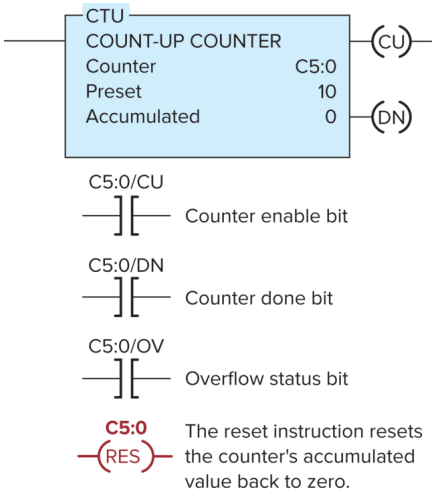


Figure 8-8 Count-up counter instruction.

Figure 8-9 shows the timer/counter menu tab from the RSLogix toolbar. Several timer and counter instructions appear when this tab is selected. The first three are timer instructions that are covered in **Chapter 7**. The next two instructions from the left are the up-counter (CTU) and down-counter

(CTD) instructions. To the right of the CTU and CTD instructions is the reset (RES) instruction, which is used by both counters and timers. The counter commands can be summarized as follows:

CTU (Count-Up)—Increments the accumulated value at each false-to-true transition and retains the accumulated value when an off/on power cycle occurs.

CTD (Count-Down)—Decrements the accumulated value at each false-to-true transition and retains the accumulated value when an on/off power cycle occurs.

HSC (High-Speed Counter)—Counts high-speed pulses from a high-speed input.

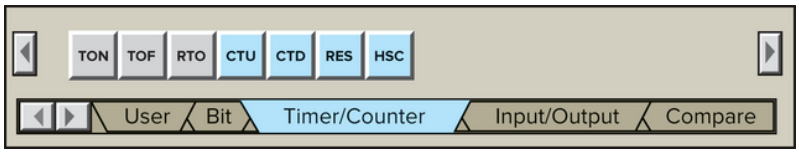


Figure 8-9 Counter selection toolbar.

Figure 8-10 shows a PLC counter program used to stop a motor from running after 10 operations. The operation of the program can be summarized as follows:

- Up-counter C5:0 counts the number of on/off times the motor starts.
- The preset value of the counter is set to 10.
- A counter done bit examine-off instruction is programmed in series with the motor output instruction.
- A motor output examine-on instruction is used to increment the accumulated value of the counter for each off/on operation.

Page 151

- After the count of 10 is reached, the counter done bit examine-off instruction goes false preventing the motor from being started.
- Closure of the reset pushbutton resets the accumulated count to zero.

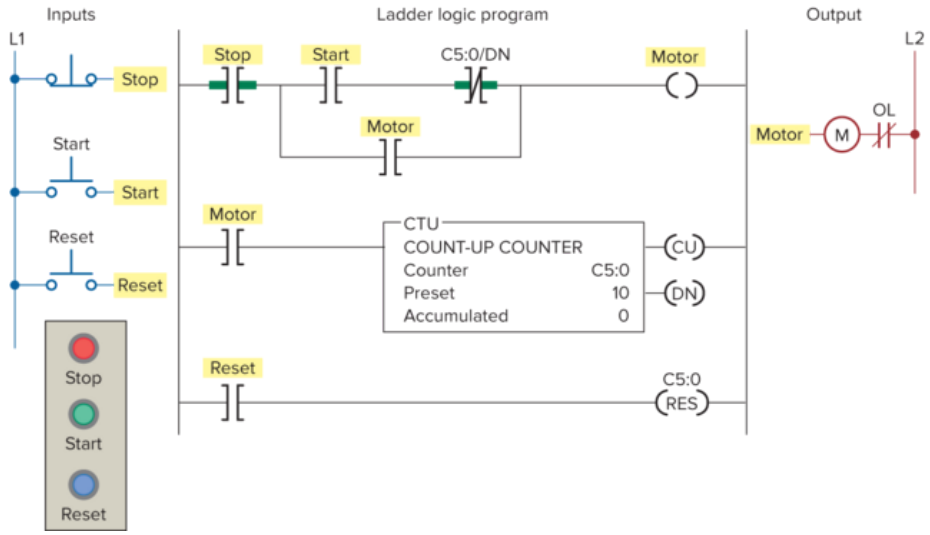


Figure 8-10 PLC counter program used to stop a motor from running after 10 operations.

Figure 8-11 shows a PLC can-counting program that uses three up-counters. The operation of the program can be summarized as follows:

- Counter C5:2 counts the total number of cans coming off an assembly line for final packaging.
- Each package must contain 10 parts.
- When 10 cans are detected, counter C5:1 sets bit B3:0/1 to initiate the box closing sequence.
- Counter C5:3 counts the total number of packages filled in a day. (The maximum number of packages per day is 300.)
- A pushbutton is used to restart the total part and package count from zero daily.

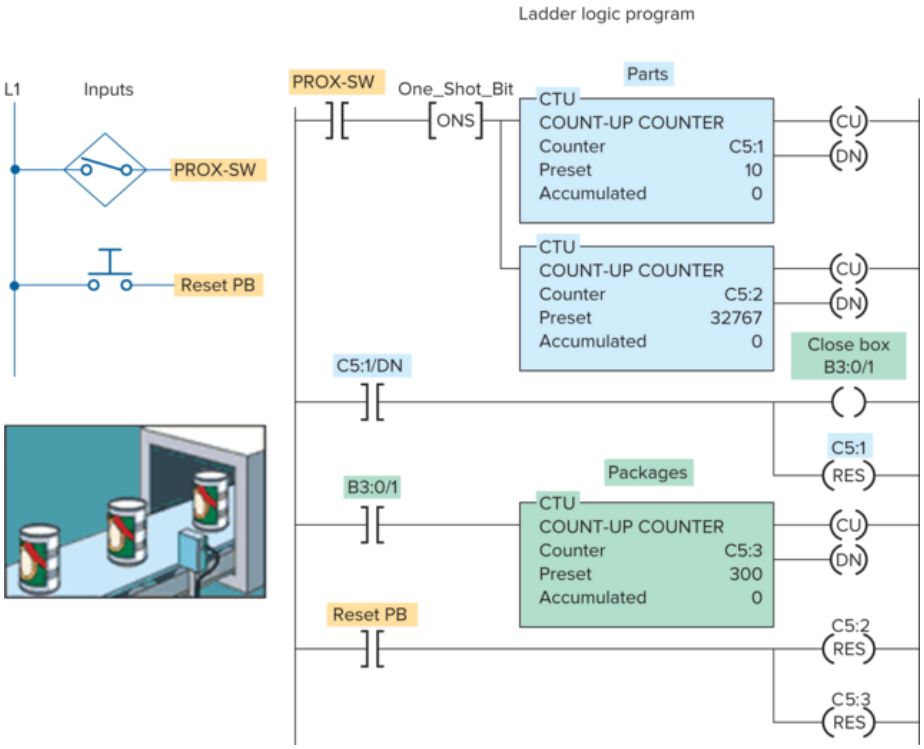
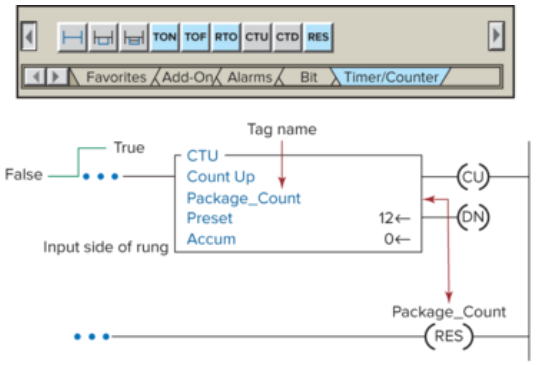


Figure 8-11 Can-counting program.

The counter instructions for the RSLogix 500 and RSLogix 5000 processors operate in exactly the same manner. Figure 8-12 shows the counter selection toolbar, CTU counter instruction, and program tags dialog box for RSLogix 5000 controller.

- Logix 5000 processors use a tag name, such as Package_Count, instead of a counter number.
- This descriptive tag name makes it easier to know what function the counter serves in the control system.
- The associated counter data (PRE, ACC, CU, CD, DN, OV, UN) are found within the program tags dialog box and together make up the counter structure.



Tag Name	Alias For	Base Tag	Type	Style	Description
Package_count			COUNTER	Decimal	12 can counter
Package_count.PRE			DINT	Decimal	
Package_count.ACC			DINT	Decimal	
Package_count.CU			BOOL	Decimal	
Package_count.CD			BOOL	Decimal	
Package_count.DN			BOOL	Decimal	
Package_count.OV			BOOL	Decimal	
Package_count.UN			BOOL	Decimal	

Figure 8-12 ControlLogix counter instruction.

One-Shot Instruction

Figure 8-13 shows the program for a ***one-shot,* or transitional,** *contact circuit* that is often used to automatically clear or reset a counter. The program is designed to generate an output pulse that, when triggered, goes on for the duration of one program scan and then goes off. The one-shot can be triggered from a momentary signal or from a signal that comes on and stays on for some time. Whichever signal is used, the one-shot is triggered by the leading-edge (off-to-on) transition of the input signal. It stays on for one scan and goes off. It stays off until the trigger goes off, and then comes on again. The one-shot is perfect for resetting both counters and timers since it stays on for one scan only.

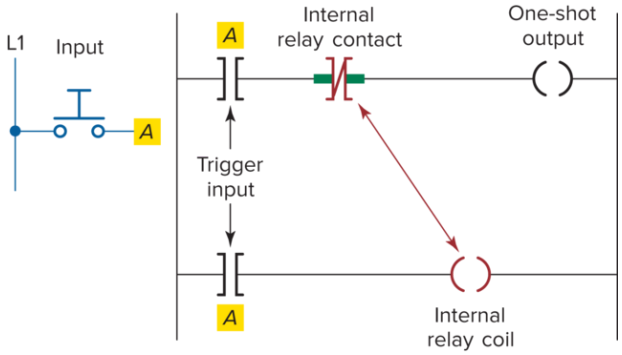


Figure 8-13 One-shot, or transitional, contact program.

Some PLCs provide transitional contacts or one-shot instructions in addition to the standard NO and NC contact instructions. The *off-to-on* transitional contact instruction, shown in **Figure 8-14a**, is programmed to provide a one-shot pulse when the referenced trigger signal makes a positive (off-to-on) transition. This contact will close for exactly one program scan whenever the trigger signal goes from off to on. The contact will allow logic continuity for one scan and then open, even though the triggering signal may stay on. The *on-to-off* transitional contact, shown in **Figure 8-14b**, provides the same operation as the off-to-on transitional contact instruction, except that it allows logic continuity for a single scan whenever the trigger signal goes from an on to an off state.

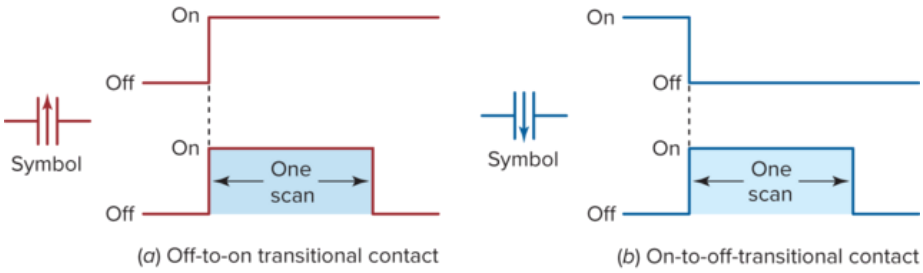


Figure 8-14 Transitional contact instructions.

The conveyor motor PLC program of **Figure 8-15** illustrates the application of an up-counter along with a programmed one-shot (OSR) transitional contact instruction. The counter counts the number of cases coming off the conveyor. When the total number of cases reaches 50, the conveyor motor stops automatically. The trucks being loaded will take a total of only 50 cases of this particular product; however, the count can be changed for different product lines. The operation of the program can be summarized as follows:

- The momentary start button is pressed to start the conveyor motor M1.
- The passage of cases is sensed by the proximity switch.
- Cases move past the proximity switch and increment the counter's accumulated value with each false-to-true transition of the switch.
- The retentive OSR instruction is true for only one scan and then false again, even if the triggering proximity switch signal stays true. This may be required for the count pulse to operate properly.
- After a count of 50, the done bit of the counter changes state to stop the conveyor motor automatically and reset the counter's accumulated value to zero.
- The conveyor motor can be stopped and started manually at any time without loss of the accumulated count.
- The accumulated count of the counter can be reset manually at any time by means of the count reset button.

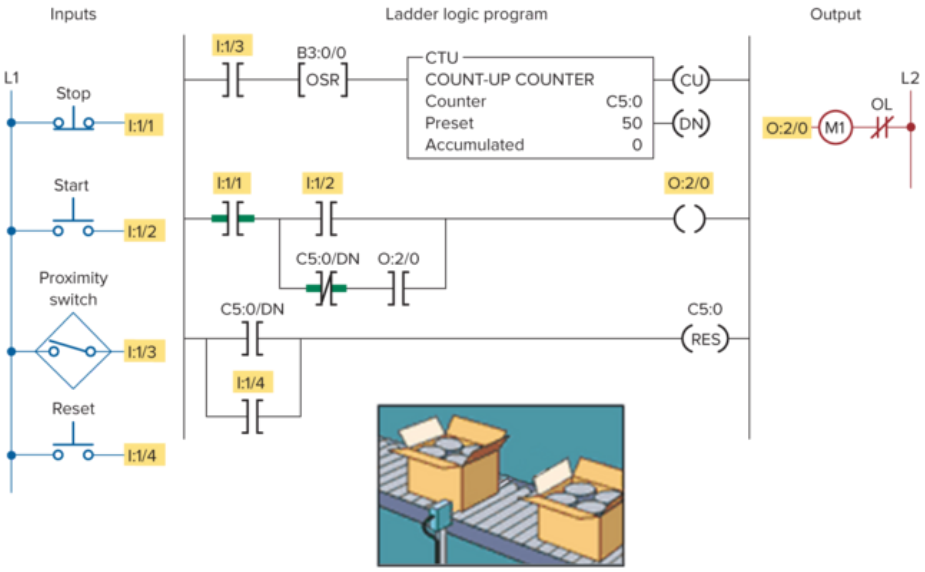


Figure 8-15 Case-counting program.

The Allen-Bradley RSLogix 500 **one-shot rising (OSR)** instruction is an input instruction that triggers an event to occur one time. The OSR instruction is placed in the ladder logic before the output instruction. When the rung conditions preceding the OSR instructions go from false-to-true, the OSR instruction goes true also but for only one scan. **Figure 8-16** illustrates the operation of an OSR rung which can be summarized as follows:

- The OSR, one-shot rising instruction is used to make the counter reset instruction (RES) true for one scan when limit switch input LS1 goes from false to true.
- The OSR is assigned a Boolean bit (B3:0/0) that is not used anywhere else in the program.
- The OSR instruction must immediately precede the output instruction.
- When the limit switch closes the LS1 and OSR, input instructions go from false to true. The OSR instruction conditions the rung so that the counter C5:1 reset output instruction goes true for one program scan.
- The output reset instruction goes false and remains false for successive scans until the input makes another false-to-true transition.
- The OSR bit is set to 1 as long as the limit switch remains closed.
- The OSR bit is reset to 0 when the limit switch is opened.

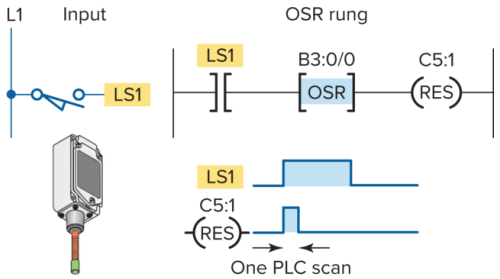


Figure 8-16 One-shot rising (OSR) instruction.

Applications for the OSR instruction include freezing rapidly displayed LED values. Figure 8-17 shows a one-shot instruction used to send data to an output LED display. The one-shot allows the rapidly changing accumulated time from the timer to be frozen to ensure a readable, stable display. The operation of the program is summarized as follows:

- The accumulated value of timer T4:1 is converted to Binary Coded Decimal (BCD) and moved to output word O:6 where an LED display is connected.
- When the timer is running, SW (I:1/1) closed, the accumulated value changes rapidly.
- Closing the momentary pushbutton PB (I:1/0) will freeze and display the value at that point in time.

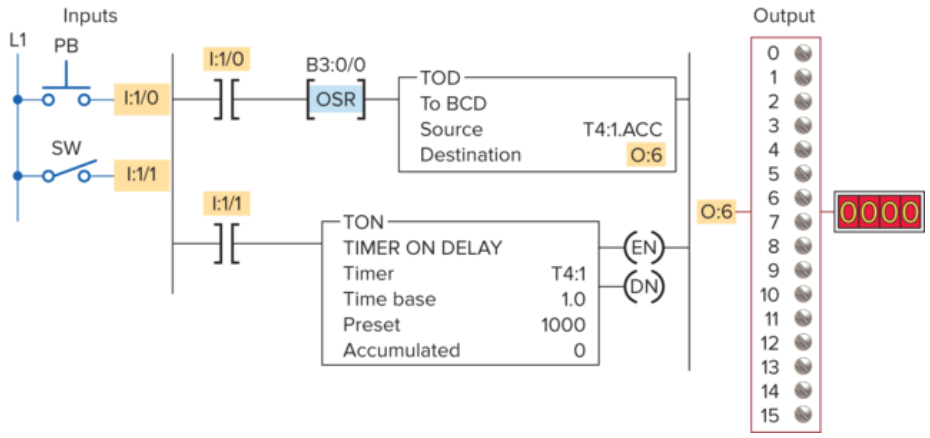


Figure 8-17 OSR instruction used to freeze rapidly displayed LED values.

The alarm monitor PLC program of Figure 8-18 illustrates the application of an up-counter used in conjunction with the programmed timed oscillator circuit studied in Chapter 7. The operation of the program can be summarized as follows:

- The alarm is triggered by the closing of float switch FS.
- The light will flash whenever the alarm condition is triggered and has not been acknowledged, even if the alarm condition clears in the meantime.
- The alarm is acknowledged by closing selector switch SS.
- The light will operate in the steady on mode when the alarm trigger condition still exists but has been acknowledged.

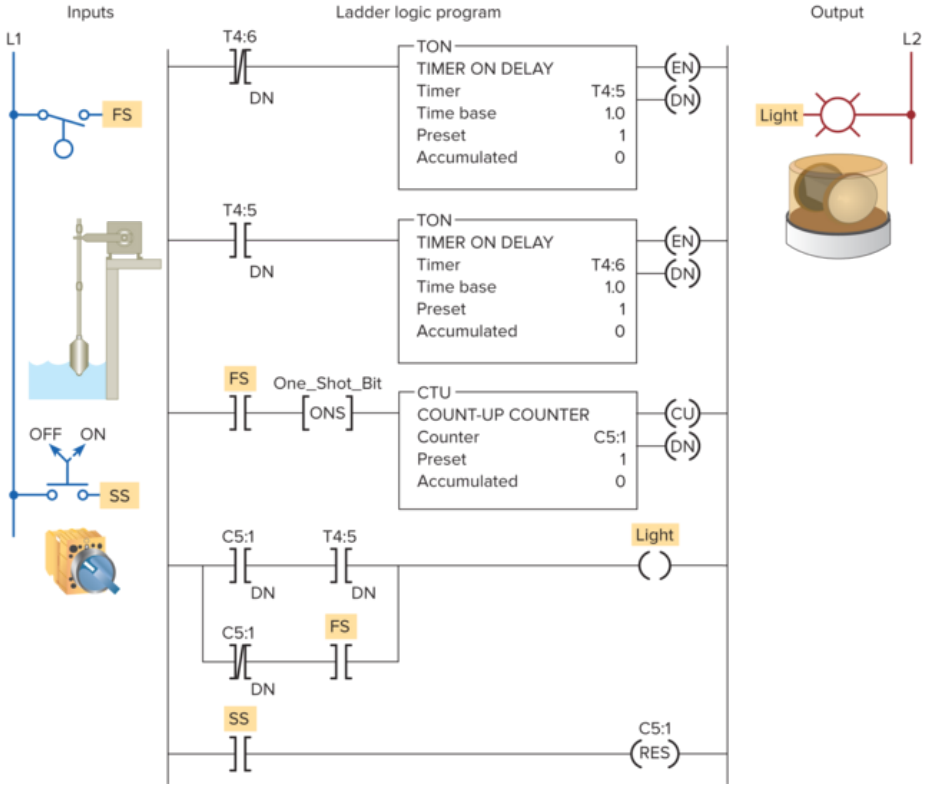


Figure 8-18 Alarm monitor program.

OceanofPDF.com

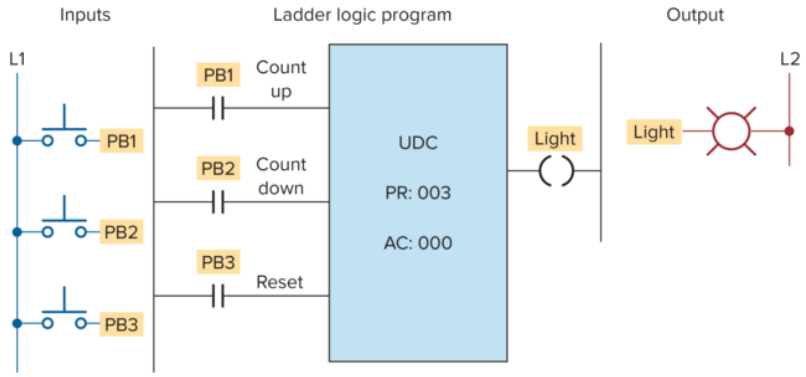
8.3 Down-Counter

The down-counter instruction will count down or decrement by 1 each time the counted event occurs. Each time the down-count event occurs, the accumulated value is decremented. Normally the down-counter is used in conjunction with the up-counter to form an up/down-counter.

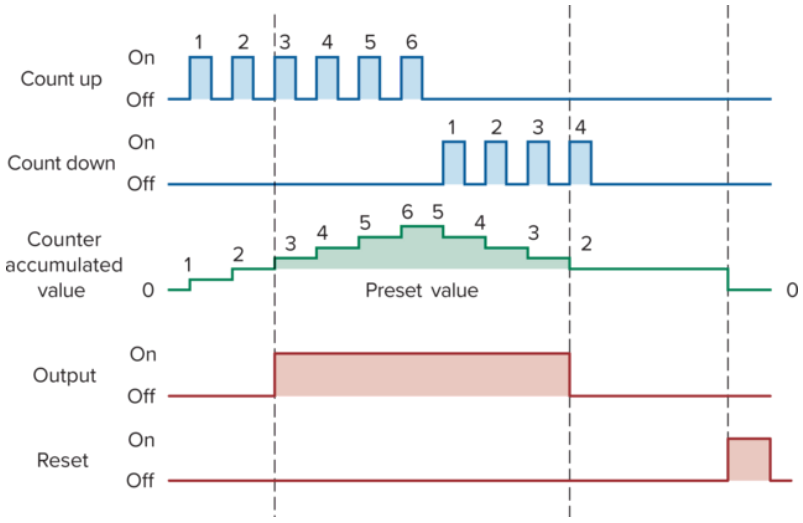
Figure 8-19 shows the program and timing diagram for a generic, block-formatted up/down-counter. The operation of the program can be summarized as follows:

- Separate count-up and count-down inputs are provided.
- Assuming the preset value of the counter is 3 and the accumulated count is 0, pulsing the count-up input (PB1) three times will switch the output light from off to on.
- This particular PLC counter keeps track of the number of counts received above the preset value. As a result, three additional pulses of the count-up input (PB1) produce an accumulated value of 6 but no change in the output.

- If the count-down input (PB2) is now pulsed four times, the accumulated count is reduced to 2 $(6 - 4)$. As a result, the accumulated count drops below the preset count and the output light switches from on to off.
- Pulsing the reset input (PB3) at any time will reset the accumulated count to 0 and turn the output light off.



(*a*)



(*b*)

Figure 8-19 Generic up/down-counter program. (a) Program. (b) Counting diagram.

Not all counter instructions count in the same manner. Some up-counters count only to their preset values, and additional counts are ignored. Other up-counters keep track of the number of counts received above the counter's preset value. Conversely, some down-counters will simply count down to zero and no further. Other down-counters may count below zero and begin counting down from the largest preset value that can be set for the PLC's counter instruction. For example, a PLC up/down-counter that has a maximum counter preset limit of 999 may count up as follows: 997, 998, 999, 000, 001, 002, and so on. The same counter would count down in the following manner: 002, 001, 000, 999, 998, 997, and so on.

One application for an up/down-counter is to keep count of the cars that enter and leave a parking garage. **Figure 8-20** shows a typical PLC program that could be used to implement this. The operation of the program can be summarized as follows:

- As a car enters, the enter switch triggers the up-counter output instruction and increments the accumulated count by 1.
- As a car leaves, the exit switch triggers the down-counter output instruction and decrements the accumulated count by 1.
- Because both the up- and down-counters have the same address, C5:1, the accumulated value will be the same in both instructions as well as the preset.
- Whenever the accumulated value of 150 equals the preset value of 150, the counter output is energized by the done bit to light up the Lot Full sign.
- A reset button has been provided to reset the accumulated count.

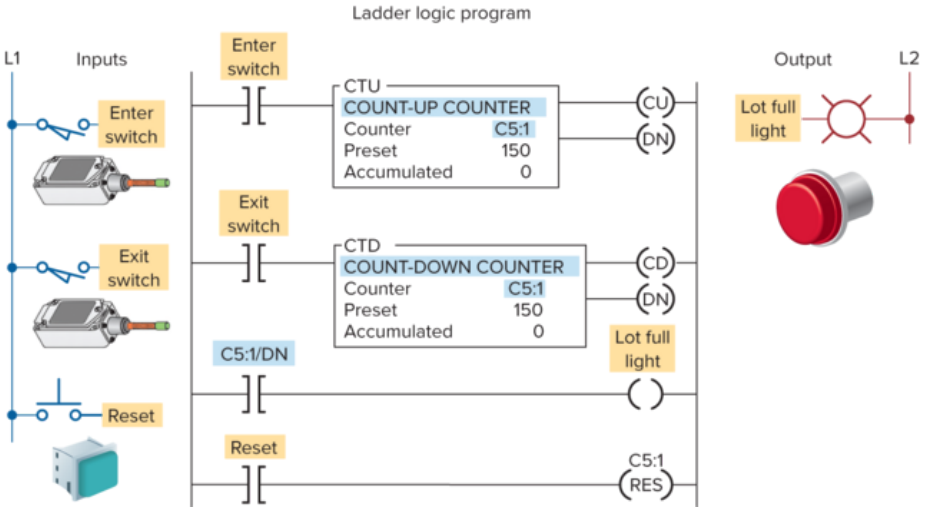


Figure 8-20 Parking garage counter.

Figure 8-21 shows an example of the count-down counter instruction used as part of the RSLogix 500 controller instruction set. The information to be entered into the instruction is the same as for the count-up counter instruction.

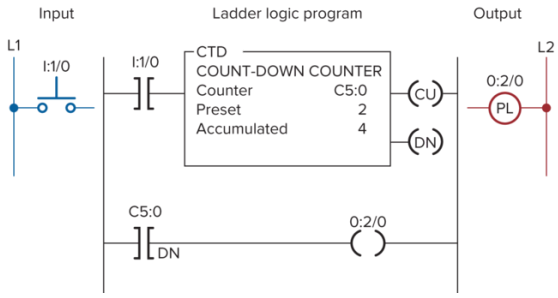


Figure 8-21 Count-down counter instruction.

The CTD instruction decrements its accumulated value by 1 every time it is transitioned. It sets its done bit when the accumulated value is equal to or greater than the preset value. The program of **Figure 8-21** contains a count-down counter instruction, the operation of which can be summarized as follows:

- With the program in the state shown, the CTD done bit will be set (1) and output 0:2/0 will be energized because the accumulated value of 4 is greater than the preset value of 2. Page 157
- When the CTD instruction rung makes a false-to-true transition, the accumulated value decreases by one count to 3.
- When the input rung condition makes another false-to-true transition, the accumulated value will decrease to 2.
- When the input makes one more false-to true transition, the accumulated value will drop to 1.

- At this point the accumulated value of 1 is less than the preset value of 2 so the done bit will be reset (0) de-energizing output O:2/0.

Figure 8-22 shows an up/down-counter program that will increase the counter's accumulated value when pushbutton PB1 is pressed and will decrease the counter's accumulated value when pushbutton PB2 is pressed. Note that the same address is given to the ***up-counter*** instruction, the **down-counter** instruction, and the **reset** instruction. All three instructions will be looking at the ***same address*** in the counter file. When input *A* goes from false to true, one count is added to the accumulated value. When input *B* goes from false to true, one count is subtracted from the accumulated value. The operation of the program can be summarized as follows:

- When the CTU instruction is true, C5:2/CU will be true, causing output *A* to be true.
- When the CTD instruction is true, C5:2/CD will be true, causing output *B* to be true.
- When the accumulated value is greater than or equal to the preset value, C5:2/DN will be true, causing output *C* to be true.
- Input *C* going true will cause both counter instructions to reset. When *reset* by the **RES instruction,** the accumulated value will be reset to 0 and the done bit will be reset.

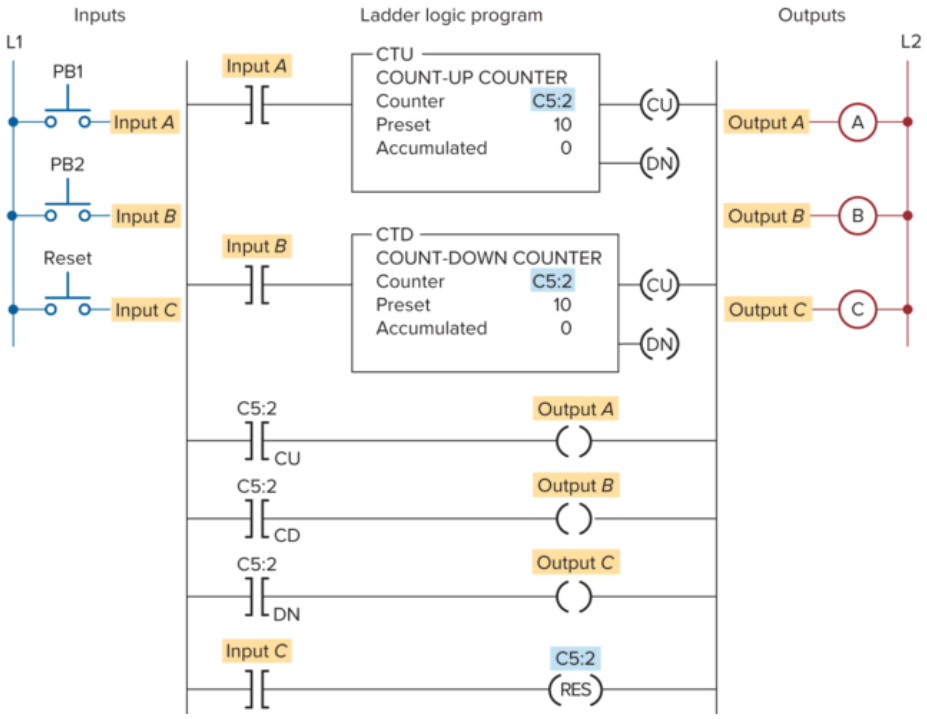


Figure 8-22 Up/down-counter program.

Figure 8-23 illustrates the operation of the up/down-counter program used to provide continuous monitoring of items in process. An in-feed photoelectric sensor counts raw parts going into the system, and an out-feed photoelectric sensor counts finished parts leaving the machine. The number of parts between the in-feed and out-feed is indicated by the accumulated count of the counter. Counts applied

to the up-input are added, and counts applied to the down-input are subtracted. The operation of the program can be summarized as follows:

- Before start-up, the system is completely empty of parts, and the counter is reset manually to 0. Page 158
- When the operation begins, raw parts move through the in-feed sensor, with each part generating an up count.
- After processing, finished parts appearing at the out-feed sensor generate down counts, so the accumulated count of the counter continuously indicates the number of in-process parts.
- The counter preset value is irrelevant in this application. It does not matter whether the counter outputs are on or off. The output on-off logic is not used. We have arbitrarily set the counter's preset values to 50.

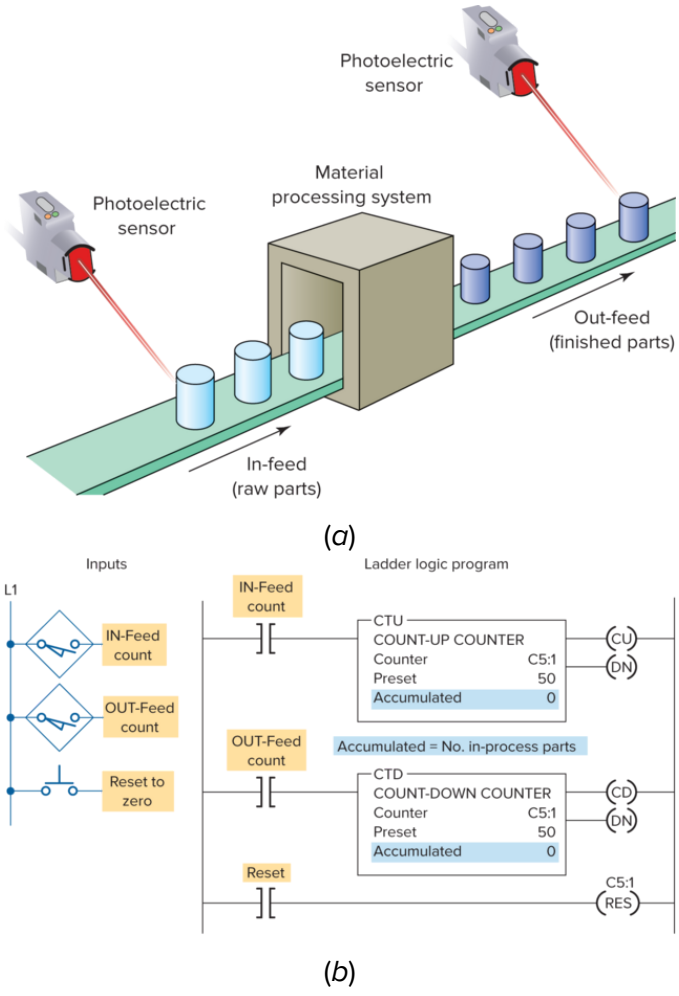


Figure 8-23 In-process monitoring program. (a) Process. (b) Program.

The maximum speed of transitions that you can count is determined by your program's scan time. For a reliable count, your counter input signal must be fixed for one scan time. If the input changes faster

than one scan period, the count value will become unreliable because counts will be missed. When this situation occurs, you need to use a high-speed counter input or a separate counter I/O module designed for high-speed applications.

OceanofPDF.com

8.4 Cascading Counters

Depending on the application, it may be necessary to count events that exceed the maximum number allowable per counter instruction. One way of accomplishing this count is by interconnecting, or **cascading**, two counters. The program of **Figure 8-24** illustrates the application of the technique. For this scenario, both counter presets have been set to 32000. The operation of the program can be summarized as follows:

- The output of the second counter is programmed into the input of the first counter.
- When the accumulated value of the second counter is equal to its preset, the DN bit will be true, which allows the first counter to count.
- The status bits of both counters are programmed in series to produce an output.
- These two counters allow twice as many counts to be measured.
- A CTU instruction that is reset while the counter logic remains true will result in an accumulated value of 1 instead of 0. Using the OSR instruction in the counter enabling logic prevents this from happening.

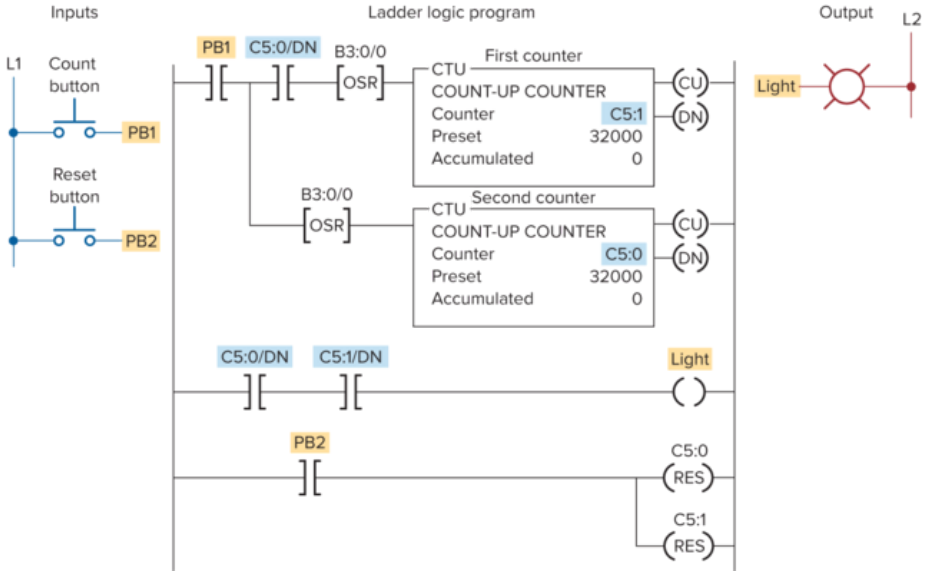


Figure 8-24 Counting beyond the maximum count.

Another method of cascading counters is sometimes used when an extremely large number of counts must be stored. For example, if you require a counter to count up to 250,000, it is possible to achieve this by using only two counters. **Figure 8-25** shows how the two counters would be programmed for this purpose. The operation of the program can be summarized as follows:

- Counter C5:1 has a preset value of 500 and counter C5:2 has a preset value of 500.
- Whenever counter C5:1 reaches 500, its done bit resets counter C5:1 and increments counter C5:2 by 1.

- When the done bit of counter C5:1 has turned on and off 500 times, the output light becomes energized. Therefore, the output light turns on after 500×500, or 250,000, transitions of the count input.

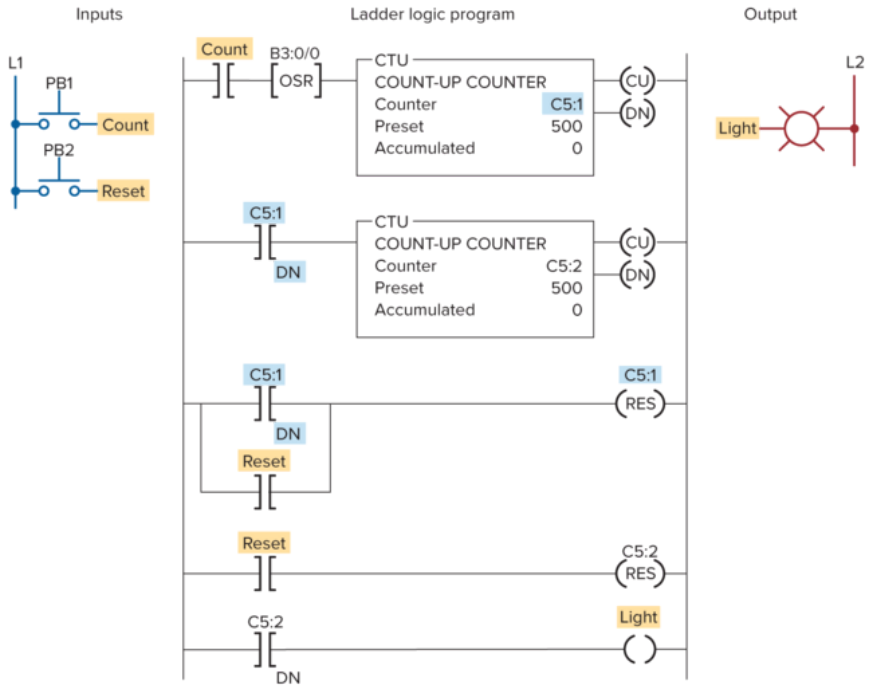


Figure 8-25 Cascading counters for extremely large counts.

Some PLCs include a real-time clock as part of their instruction set. A real-time clock allows you to display the time of day or to log data pertaining to the operation of the process. The logic used to implement a clock as part of a PLC's program is straightforward and simple to accomplish. A single timer instruction and counter instructions are all you need.

Figure 8-26 illustrates a timer-counter program that produces a time-of-day clock measuring time in hours and minutes. The operation of the program is summarized as follows:

- An RTO timer instruction (T4:0) is programmed first with a preset value of 60 seconds.
- The T4:0 timer times for a 60-second period, after which its done bit is set.
- This, in turn, causes the up-counter (C5:0) of rung 001 to increment 1 count.
- On the next processor scan, the timer is reset and begins timing again.
- The C5:0 counter is preset to 60 counts, and each time the timer completes its time-delay period, its count is incremented.
- When the C5:0 counter reaches its preset value of 60, its done bit is set.
- This, in turn, causes the up-counter (C5:1) of rung 002, which is preset for 24 counts, to increment 1 count.
- Whenever the C5:1 counter reaches its preset value of 24, its done bit is set to reset itself.

- The time of day is generated by examining the current, or accumulated, count or time for each counter and the timer.
- Counter C5:1 indicates the hour of the day in 24-h military format, while the current minutes are represented by the accumulated count value of counter C5:0.
- The timer displays the seconds of a minute as its current, or accumulated, time value.

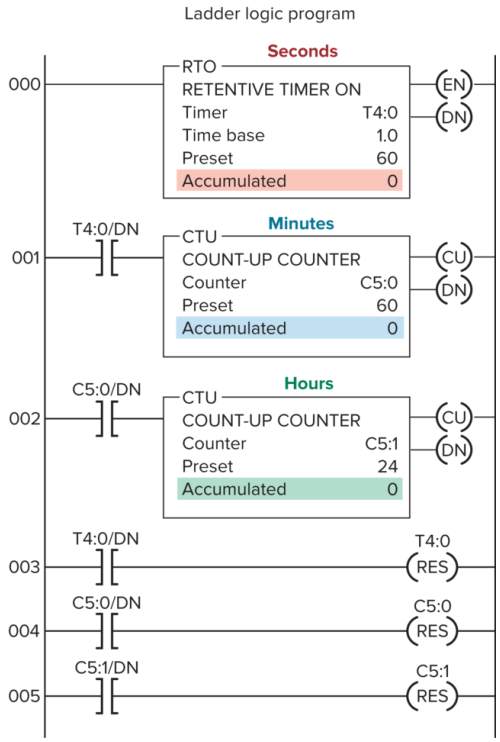


Figure 8-26 24-hour clock program.

The 24-hour clock can be used to record the time of an event. **Figure 8-27** illustrates the principle of this technique. In this application the time of the opening of a pressure switch is to be recorded. The operation of the program can be summarized as follows:

- The circuit is set into operation by pressing the reset button and setting the clock for the time of day.
- This starts the 24-hour clock and switches the set indicating light on.
- Should the pressure switch open at any time, the clock will automatically stop and the trip indicating light will switch on.
- The clock can then be read to determine the time of opening of the pressure switch.

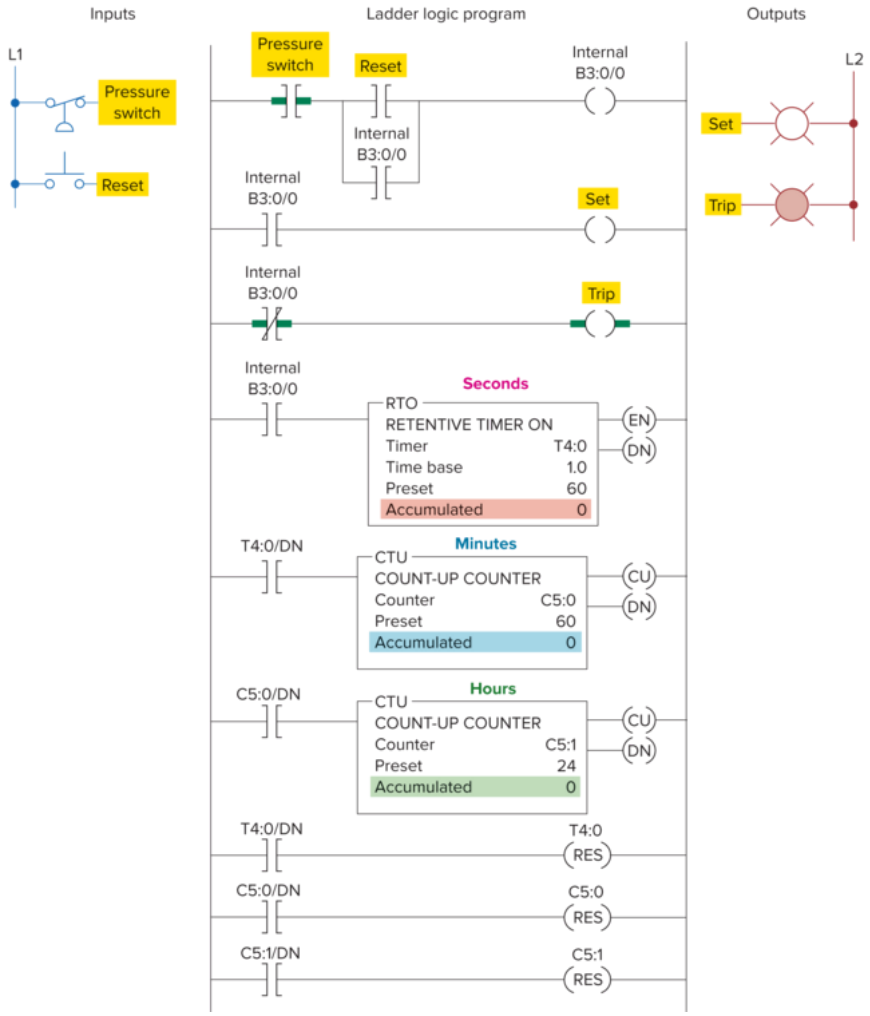


Figure 8-27 Monitoring the time of an event.

OceanofPDF.com

8.5 Incremental Encoder-Counter Applications

Incremental encoders are used to track motion. They provide a specific number of equally spaced pulses per revolution or per inch or millimeter of linear motion. Incremental encoders output pulses each time and only when the shaft is turned. The **incremental optical encoder** shown in Figure 8-28 creates a series of square waves as its shaft is rotated. The encoder disk interrupts the light as the encoder shaft is rotated to produce the square wave output waveform.

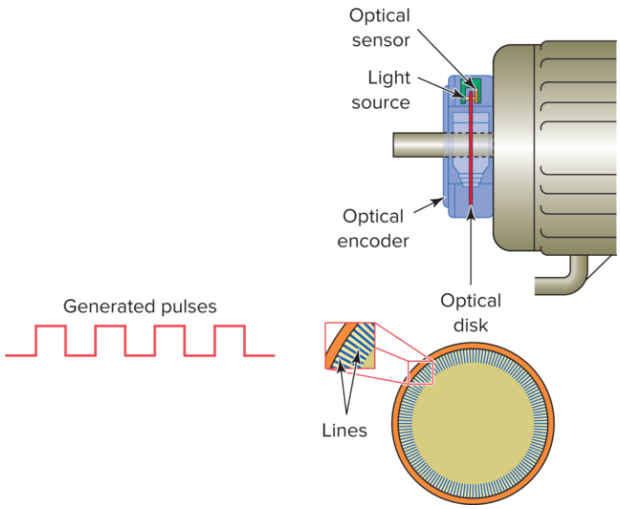


Figure 8-28 Optical incremental encoder.

The number of square waves obtained from the output of the encoder can be made to correspond to the mechanical movement required. For example, to divide a shaft revolution into 100 parts, an encoder could be selected to supply 100 square wave-cycles per revolution. By using a counter to count those cycles, we could tell how far the shaft had rotated.

Figure 8-29 illustrates an example of cutting objects to a specified length. The object is advanced for a specified distance and measured by encoder pulses to determine the correct length for cutting.

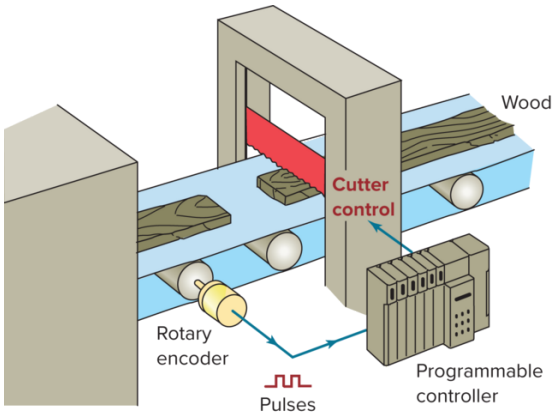


Figure 8-29 Cutting objects to a specified length.

Figure 8-30 shows a counter RSLogix 5000 program used for length measurement. This system accumulates the total length of random pieces of bar stock moved on a conveyor. The operation of the program can be summarized as follows:

Page 162

- Count input pulses are generated by the magnetic sensor, which detects passing teeth on a conveyor drive sprocket.
- If 10 teeth per foot of conveyor motion pass the sensor, the accumulated count of the counter would indicate feet in tenths.
- The photoelectric sensor monitors a reference point on the conveyor. When activated, it prevents the unit from counting, thus permitting the counter to accumulate counts only when bar stock is moving.
- The counter is reset by closing the reset button.

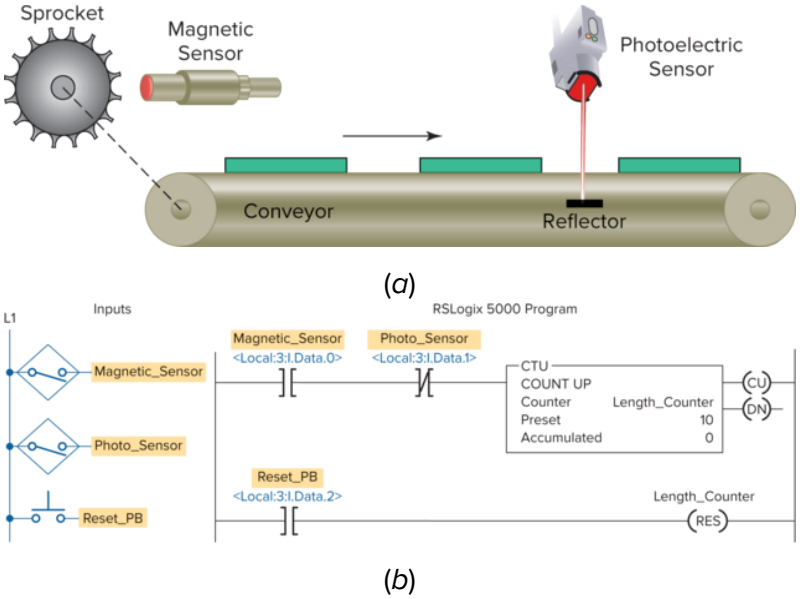


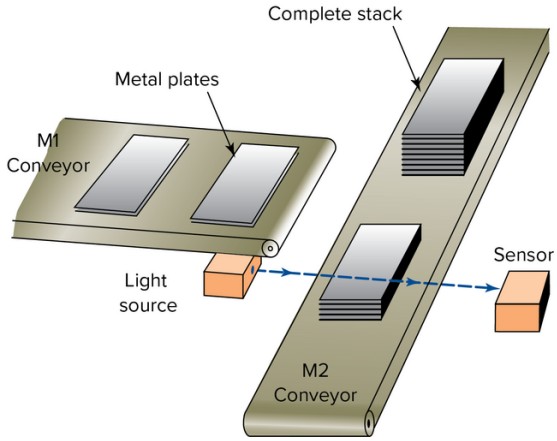
Figure 8-30 Counter used for length measurement. (a) Process. (b) Program.

OceanofPDF.com

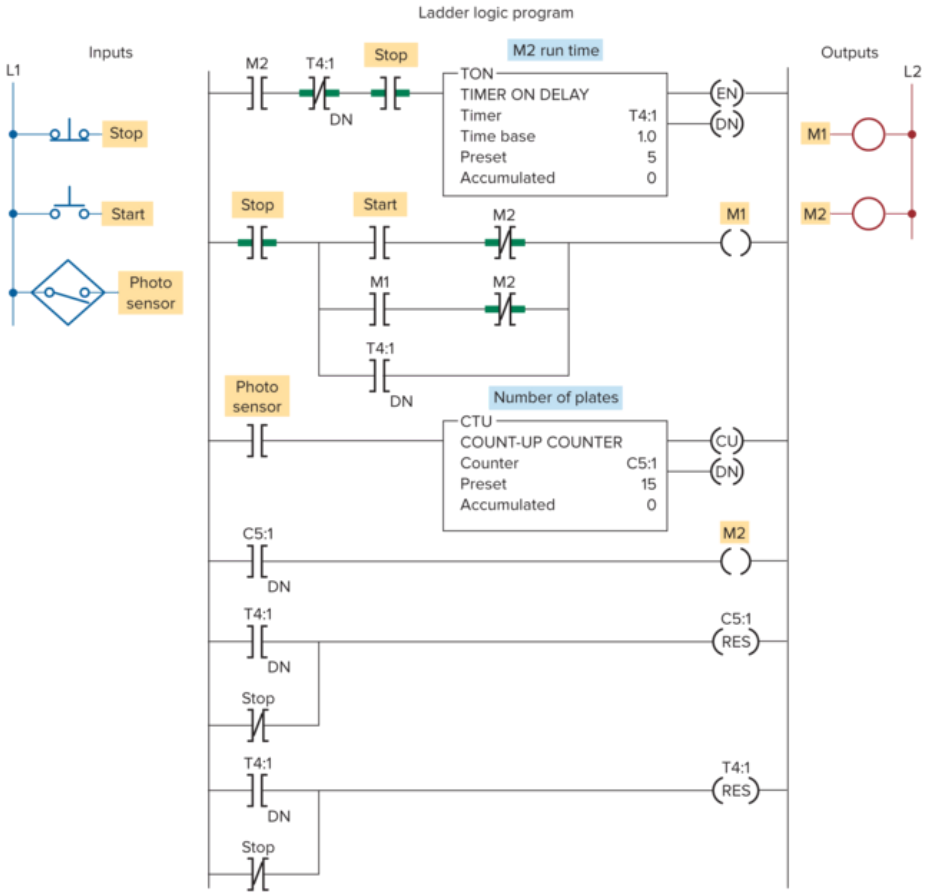
8.6 Combining Counter and Timer Functions

Many PLC applications use both the counter function and the timer function. **Figure 8-31** illustrates an automatic stacking program that requires both a timer and counter. In this process, conveyor M1 is used to stack metal plates onto conveyor M2. The photoelectric sensor provides an input pulse to the PLC counter each time a metal plate drops from conveyor M1 to M2. When 15 plates have been stacked, conveyor M2 is activated for 5 s by the PLC timer. The operation of the program can be summarized as follows:

- When the start button is pressed, conveyor M1 begins running.
- After 15 plates have been stacked, conveyor M1 stops and conveyor M2 begins running.
- After conveyor M2 has been operated for 5 s, it stops and the sequence is repeated automatically.
- The done bit of the timer resets the timer and the counter and provides a momentary pulse to automatically restart conveyor M1.



(*a*)



(*b*)

Figure 8-31 Automatic stacking program. (a) Process. (b) Program.

Page 164

Figure 8-32 shows a motor lock-out program. This program is designed to prevent a machine operator from starting a motor that has tripped off more than 5 times in an hour. The operation of the program can be summarized as follows:

- The normally open overload (OL) relay contact momentarily closes each time an overload current is sensed.
- Every time the motor stops due to an overload condition, the motor start circuit is locked out for 5 min.

Page 165

- If the motor trips off more than 5 times in an hour, the motor start circuit is permanently locked out and cannot be started until the reset button is actuated.
- The lock-out pilot light is switched on whenever a permanent lock-out condition exists.

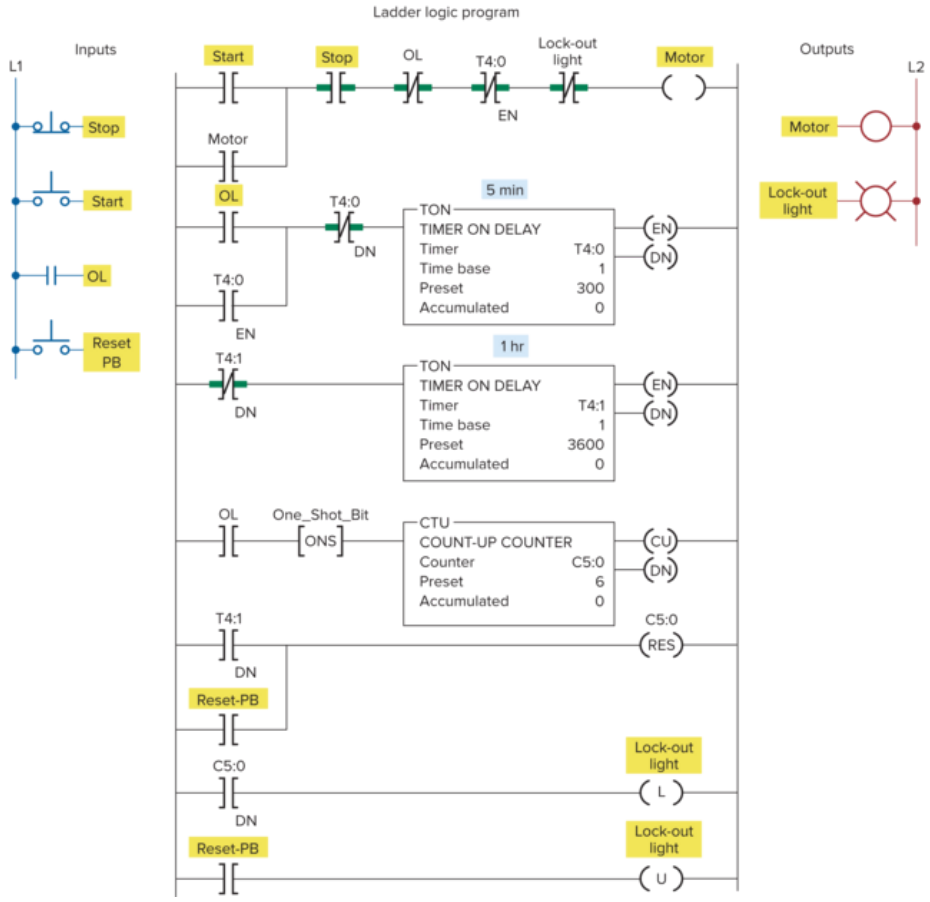


Figure 8-32 Motor lock-out program.

Figure 8-33 shows a product part flow rate program. This program is designed to indicate how many parts pass a given process point per minute. The operation of the program can be summarized as follows:

- When the start switch is closed, both the timer and counter are enabled.
- The counter is pulsed for each part that passes the parts sensor.
- The counting begins and the timer starts timing through its 1-minute time interval.
- At the end of 1 minute, the timer done bit causes the counter rung to go false.
- Sensor pulses continue but do not affect the PLC counter.
- The number of parts for the past minute is represented by the accumulated value of the counter.
- The sequence is reset by momentarily opening and closing the start switch.

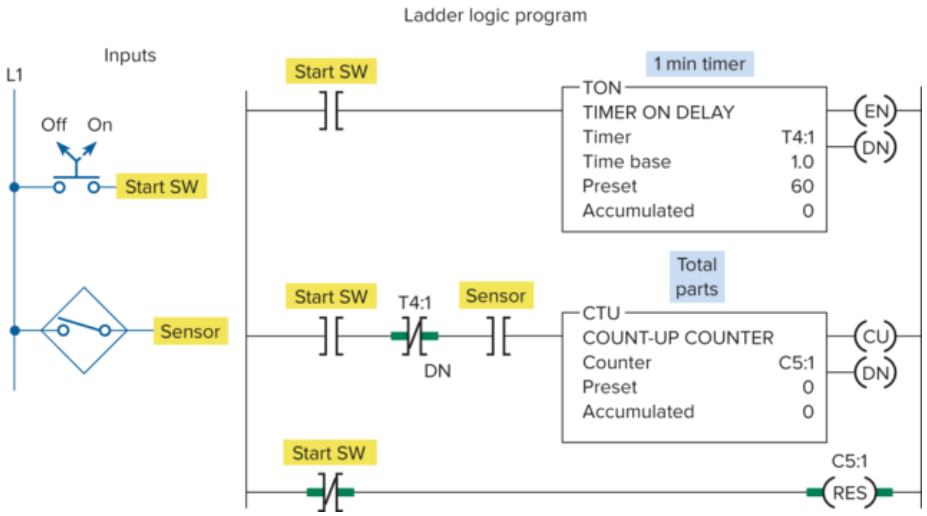


Figure 8-33 Product flow rate program.

A timer is sometimes used to drive a counter when an extremely long time-delay period is required. For example, if you require a timer to time to 1,000,000 s, you can achieve this by using a single timer and counter. **Figure 8-34** shows how the timer and counter would be programmed for such a purpose. The operation of the program can be summarized as follows:

- Timer T4:0 has a preset value of 10,000, and counter C5:0 has a preset value of 100.
- Each time the timer T4:0 input contact closes for 10,000 s, its done bit resets timer T4:0 and increments counter C5:0 by 1.
- When the done bit of timer T4:0 has turned on and off 100 times, the output light becomes energized.
- Therefore, the output light turns on after $10{,}000 \times 100$, or 1,000,000, seconds after the timer input contact closes.

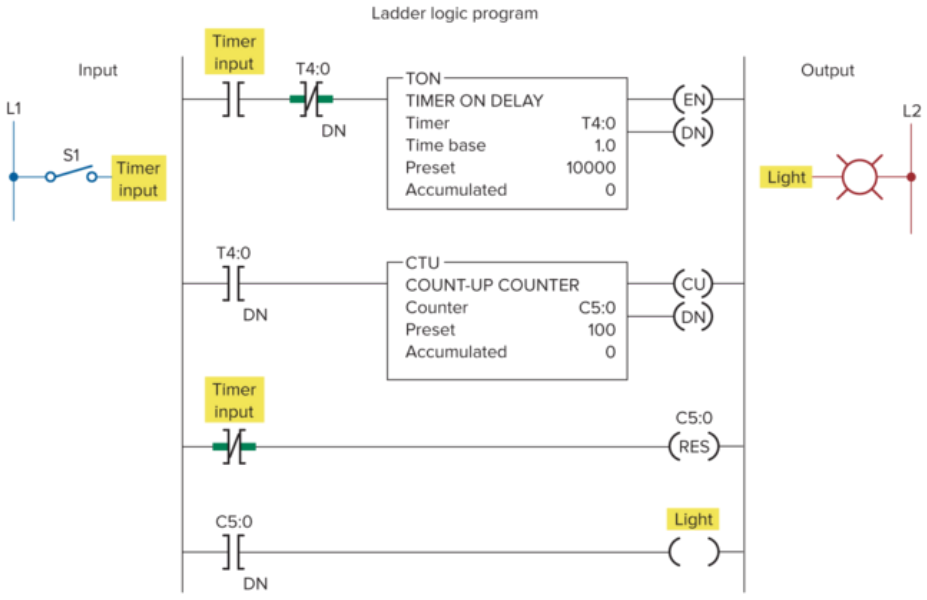


Figure 8-34 Timer driving a counter to produce an extremely long time-delay period.

OceanofPDF.com

8.7 High-Speed Counters

The maximum counting frequency of a traditional PLC's counter is limited by the scan time of the processor. When the frequency of the input signal is higher than that of the scan time, it is necessary to utilize **a high-speed counter (HSC),** to avoid errors. For example, using an incremental encoder in a length-measuring application generally requires the use of a high-speed counter. The HSC instruction may be embedded in the CPU, or fixed hardware, or a separate module.

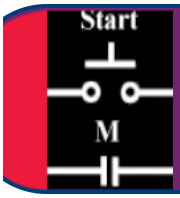
OceanofPDF.com



CHAPTER 8 REVIEW QUESTIONS

1. Name the three types of PLC counter instructions, and explain the basic operation of each.
2. State four program parameters associated with a PLC counter instruction.
3. In a PLC counter instruction, what rule applies to the addressing of the counter and reset instructions?
4. When is the output of a PLC counter energized?
5. When does the PLC counter instruction increment or decrement its current count?
6. The counter instructions of PLCs are normally retentive. Explain what this means.
7. **a.** Compare the operation of a standard Examine-on contact instruction with that of an off-to-on one-shot transitional contact.
   b. What is the normal function of a transitional contact used in conjunction with a counter?
8. Explain how an OSR (one-shot rising) instruction can be used to freeze rapidly changing data.
9. Identify the type of counter you would choose for each of the following situations:
   a. Count the total number of parts made during each shift.
   b. Keep track of the current number of parts in a stage of a process as they enter and exit.
   c. There are 10 parts in a full hopper. As parts leave, keep track of the number of parts remaining in the hopper
10. Describe the basic programming process involved in the cascading of two counters.
11. **a.** When is the overflow bit of an up-counter set?
    b. When is the underflow bit of a down-counter set?
12. Describe two common applications for counters.
13. What determines the maximum speed of transitions that a PLC counter can count? Why?

OceanofPDF.com



CHAPTER 8 PROBLEMS

1. Study the ladder logic program in **Figure 8-35**, and answer the questions that follow:

a. What type of counter has been programmed?

b. When would output O:2/0 be energized?

c. When would output O:2/1 be energized?

d. Suppose your accumulated value is 24 and you lose ac line power to the controller. When power is restored to your controller, what will your accumulated value be?

e. Rung 4 goes true and, while it is true, rung 1 goes through five false-to-true transitions of rung conditions. What is the accumulated value of the counter after this sequence of events?

f. When will the count be incremented?

g. When will the count be reset?

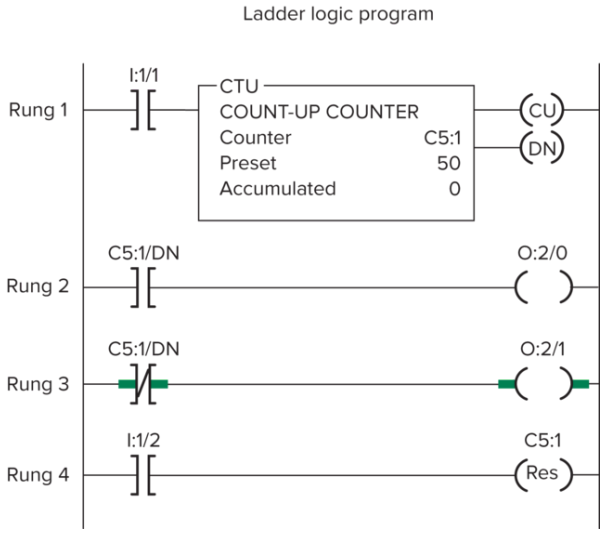


Figure 8-35 Program for Problem 1.

2. Study the ladder logic program in **Figure 8-36**, and answer the questions that follow:

a. Suppose the input pushbutton is actuated from off to on and remains held on. How will the status of output B3:0/9 be affected?

b. Suppose the input pushbutton is now released to the normally off position and remains off. How will the status of output B3:0/9 be affected?

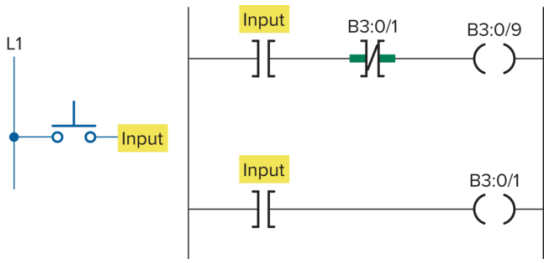


Figure 8-36 Program for Problem 2.

3. Study the ladder logic program in **Figure 8-37**, and answer the questions that follow:

a. What type of counter has been programmed?

b. What input will cause the counter to increment?

c. What input address will cause the counter to decrement?

d. What input address will reset the counter to a count of zero?

e. When would output O:6/2 be energized?

f. Suppose the counter is first reset, and then input I:2/6 is actuated 15 times and input I:3/8 is actuated 5 times. What is the accumulated count value?

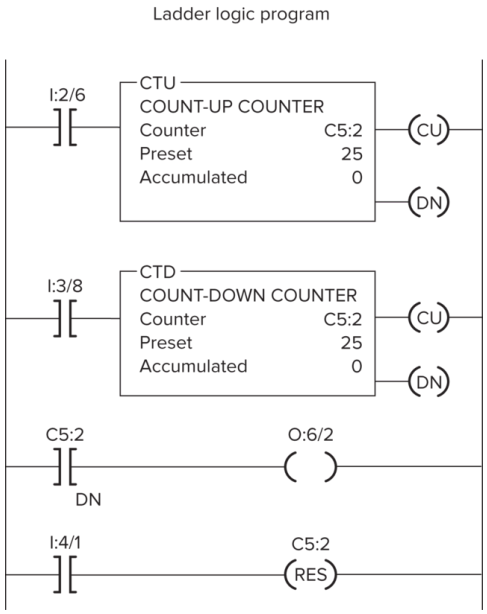


Figure 8-37 Program for Problem 3.

4. Design a PLC program and prepare a typical I/O connection diagram and ladder logic program for the following counter specifications:

- Counts the number of times a pushbutton is closed.
- Decrements the accumulated value of the counter each time a second pushbutton is closed.

- Turns on a light anytime the accumulated value of the counter is less than 20.
- Turns on a second light when the accumulated value of the counter is equal to or greater than 20.
- Resets the counter to 0 when a selector switch is closed.

5. Design a PLC program and prepare a typical I/O connection diagram and ladder logic program that will execute the following control circuit correctly:

- Turns on a nonretentive timer when a switch is closed (preset value of timer is 10 s).
- Resets timer automatically through a programmed transitional contact when it times out.
- Counts the number of times the timer goes to 10 s.
- Resets counter automatically through a second programmed transitional contact at a count of 5.
- Latches on a light at the count of 5.
- Resets light to off and counter to 0 when a selector switch is closed.

6. Design a PLC program and prepare a typical I/O connection diagram and ladder logic program that will correctly execute the industrial control process in **Figure 8-38**. The sequence of operation is as follows:

- Product in position (limit switch LS1 contacts close).
- The start button is pressed and the conveyor motor starts to move the product forward toward position *A* (limit switch LS1 contacts open when the actuating arm returns to its normal position).
- The conveyor moves the product forward to position *A* and stops (position detected by 8 off-to-on output pulses from the encoder, which are counted by an up-counter).
- A time delay of 10 s occurs, after which the conveyor starts to move the product to limit switch LS2 and stops (LS2 contacts close when the actuating arm is hit by the product).
- An emergency stop button is used to stop the process at any time.
- If the sequence is interrupted by an emergency stop, counter and timer are reset automatically.

Page 169

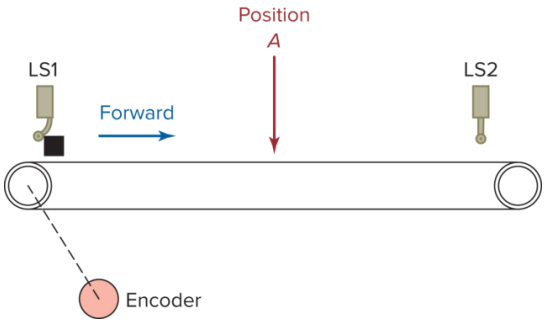


Figure 8-38 Control process for Problem 6.

7. Answer the following questions with reference to the up/down-counter program shown in **Figure 8-39**. Assume that the following sequence of events occurs:

- Input *C* is momentarily closed.

- 20 on/off transitions of input *A* occur.
- 5 on/off transitions of input *B* occur.

As a result:

a. What is the accumulated count of counter CTU?

b. What is the accumulated count of counter CTD?

c. What is the state of output *A*?

d. What is the state of output *B*?

e. What is the state of output *C*?

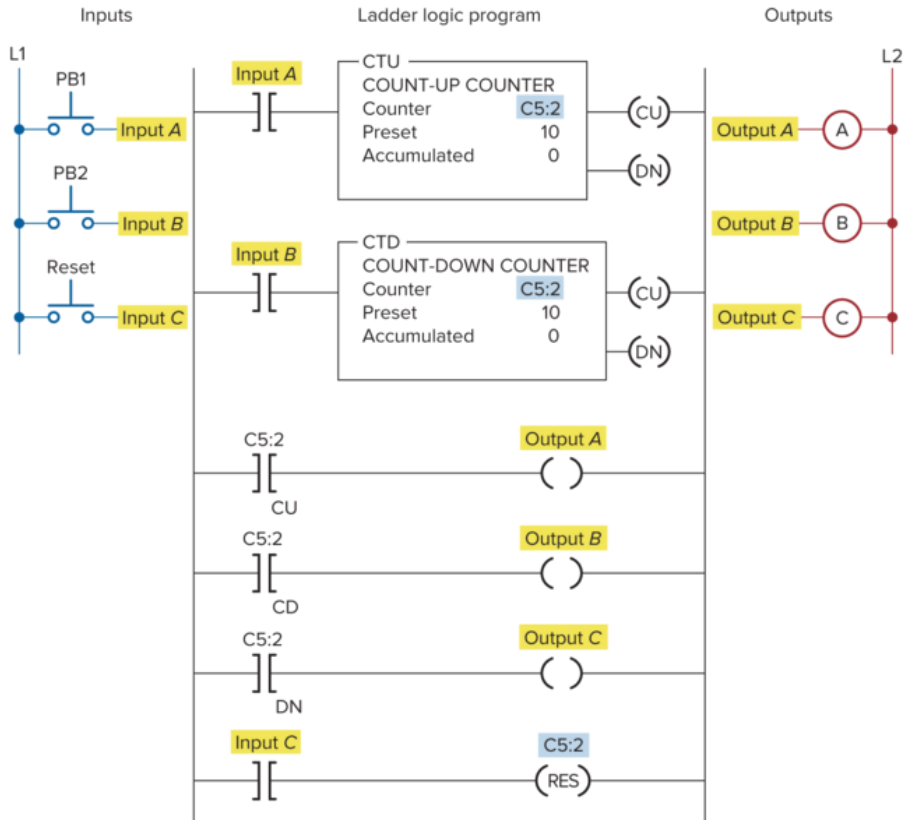


Figure 8-39 Program for Problem 7.

8. Write a program to implement the process illustrated in **Figure 8-40**. An up-counter must be programmed as part of a batch-counting operation to sort parts automatically for quality control. The counter is installed to divert 1 part out of every 1000 for quality control or inspection purposes. The circuit operates as follows:

- A start/stop pushbutton station is used to turn the conveyor motor on and off.
- A proximity sensor counts the parts as they pass by on the conveyor.
- When a count of 1000 is reached, the counter's output activates the gate solenoid, diverting the part to the inspection line.

- The gate solenoid is energized for 2 s, which allows enough time for the part to continue to the quality control line.
- The gate returns to its normal position when the 2-s time period ends.
- The counter resets to 0 and continues to accumulate counts.
- A reset pushbutton is provided to reset the counter manually.

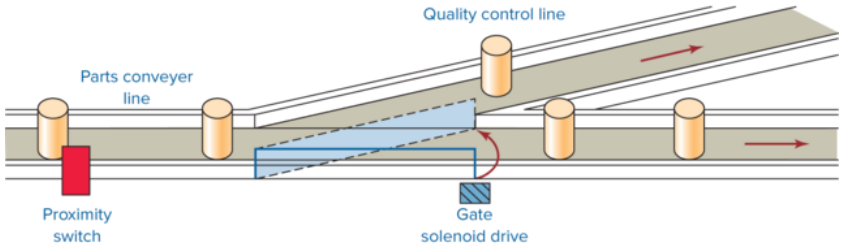


Figure 8-40 Control process for Problem 8.

9. Write a program that will increment a counter's accumulated value 1 count every 60 s. A second counter's accumulated value will increment 1 count every time the first counter's accumulated value reaches 60. The first counter will reset when its accumulated value reaches 60, and the second counter will reset when its accumulated value reaches 12.

10. Write a program to implement the process illustrated in **Figure 8-41**. A company that makes electronic assembly kits needs a counter to count and control the number of resistors placed into each kit. The controller must stop the take-up spool at a predetermined amount of resistors (100). A worker on the floor will then cut the resistor strip and place it in the kit. The circuit operates as follows:

- A start/stop pushbutton station is used to turn the spool motor drive on and off manually.
- A through-beam sensor counts the resistors as they pass by.
- A counter preset for 100 (the amount of resistors in each kit) will automatically stop the take-up spool when the accumulated count reaches 100.
- A second counter is provided to count the grand total used.
- Manual reset buttons are provided for each counter.

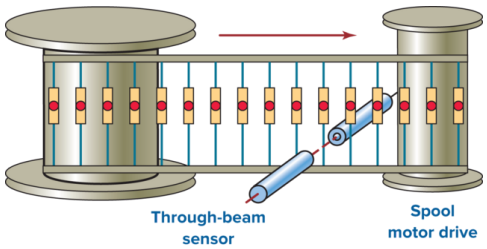


Figure 8-41 Control process for Problem 10.

11. Write a program that will latch on a light 20 s after an input switch has been turned on. The timer will continue to cycle up to 20 s and reset itself until the input switch has been turned off. After the

third time the timer has timed to 20 s, the light will be unlatched.

12. Write a program that will turn a light on when a count reaches 20. The light is then to go off when a count of 30 is reached.

13. Write a program to implement the box-stacking process illustrated in **Figure 8-42**. This application requires the control of a conveyor belt that feeds a mechanical stacker. The stacker can stack various numbers of cartons of ceiling tile onto each pallet (depending on the pallet size and the preset value of the counter). When the required number of cartons has been stacked, the conveyor is stopped until the loaded pallet is removed and an empty pallet is placed onto the loading area. A photoelectric sensor will be used to provide count pulses to the counter after each carton passes by. In addition to a conveyor motor start/stop station, a remote reset button is provided to allow the operator to reset the system from the forklift after an empty pallet is placed onto the loading area. The operation of this system can be summarized as follows:

Page 171

- The conveyor is started by pressing the start button.
- As each box passes the photoelectric sensor, a count is registered.
- When the preset value is reached (in this case 12), the conveyor belt turns off.
- The forklift operator removes the loaded pallet.
- After the empty pallet is in position, the forklift operator presses the remote reset button, which then starts the whole cycle over again.

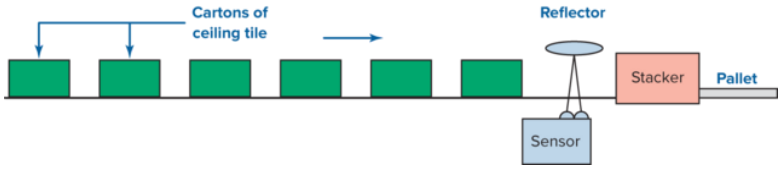


Figure 8-42 Control process for Problem 13.

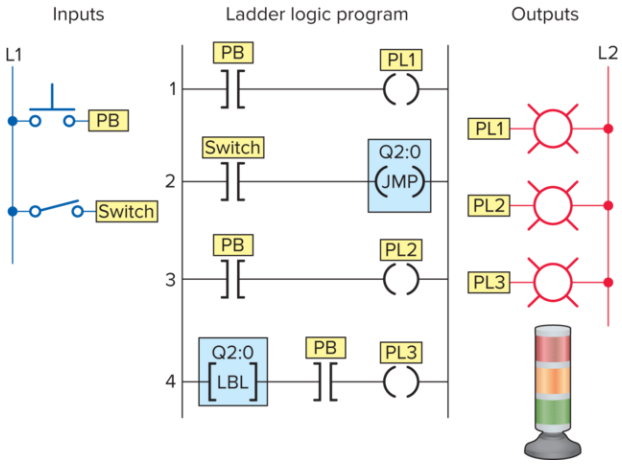
14. Write a program to operate a light according to the following sequence:

- A momentary pushbutton is pressed to start the sequence.
- The light is switched on and remains on for 2 s.
- The light is then switched off and remains off for 2 s.
- A counter is incremented by 1 after this sequence.
- The sequence then repeats for a total of 4 counts.
- After the fourth count, the sequence will stop and the counter will be reset to zero.

OceanofPDF.com

9

Program Control Instructions



Chapter Objectives

After completing this chapter, you will be able to:

- State the purpose of program control instructions
- Describe the operation of the master control reset instruction and develop an elementary program illustrating its use
- Describe the operation of the jump instruction and the label instruction
- Explain the function of subroutines
- Describe the immediate input and output instructions function
- Describe the forcing capability of the PLC
- Describe safety considerations built into PLCs and programmed into a PLC installation
- Explain the differences between standard and safety PLCs
- Describe the function of the fault routine files
- Explain how the temporary end instruction can be used to troubleshoot a program

The program control instructions covered in this chapter are used to alter the program scan from its normal sequence. The use of program control instructions can shorten the time required to complete a program scan. Portions of the program not being utilized at any particular time can be

jumped over, and outputs in specific zones in the program can be left in their desired states. Typical industrial program control applications are explained.

OceanofPDF.com

9.1 Program Control

Several output-type instructions, which are often referred to as ***override*** instructions, provide a means of executing sections of the control logic if certain conditions are met. These program control instructions allow for greater program flexibility and greater efficiency in the program scan. Portions of the program not being utilized at any particular time can be jumped over, and outputs in specific zones in the program can be left in their desired states.

Program control instructions are used to enable or disable a block of logic program or to move execution of a program from one place to another place. The Program Control menu tab of Figure 9-1 shows a partial listing of common program control instructions. The function of each command is summarized as follows:

JMP (Jump to Label)—Jump forward/backward to a corresponding label instruction.

LBL (Label)—Specifies label location.

JSR (Jump to Subroutine)—Jump to a designated subroutine instruction.

RET (Return from Subroutine)—Exits current subroutine and returns to previous condition.

SBR (Subroutine)—Identifies the subroutine program.

TND (Temporary End)—Makes a temporary end that halts program execution.

MCR (Master Control Reset)—Clears all set nonretentive output rungs between the paired MCR instructions.

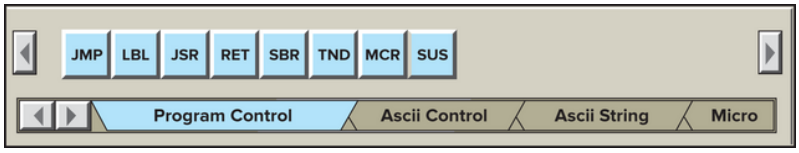


Figure 9-1 Program Control menu tab.

OceanofPDF.com

9.2 Master Control Reset Instruction

A fail-safe is a device or system that is designed to remain safe in the event of a failure. Fail-safes are not designed to prevent failure but to prevent unsafe consequences when a failure does occur. Hardwired master control relays are used in relay control circuitry to provide input/output power shutdown of an entire circuit. **Figure 9-2** shows a typical **hardwired master control relay circuit.** In this circuit, unless the master control relay coil is energized, there is no power flow to the load side of the MCR contacts.

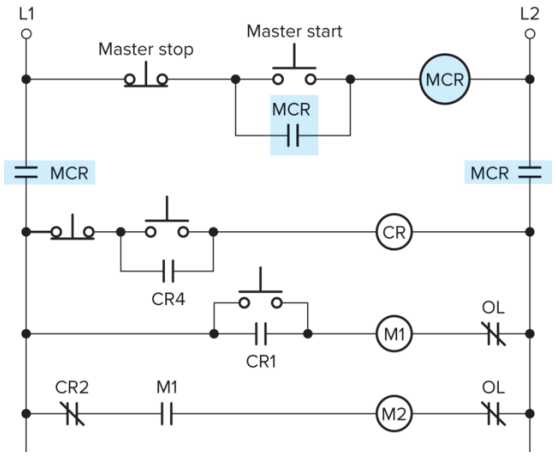


Figure 9-2 Hardwired master control relay.

The equivalent PLC instruction to a Master Control Relay is the **Master Control Reset (MCR)** instruction. This instruction functions in a similar manner to the hardwired master control relay; that is, when the instruction is true, the circuit functions normally, and when the instruction is false, nonretentive outputs are switched off. The programmed MCR instruction is not a substitute for a hardwired Master Control Relay. It is highly recommended that all PLC systems include a hardwired MCR and Emergency Stop switches to provide safe, effective shutdown of I/O power.

Page 174

A *Master Control Reset (MCR)* instruction is an output coil instruction that functions like a master control relay. MCR coil instructions are used in pairs and can be programmed to control an entire circuit or to control only selected rungs of a circuit. In the program of **Figure 9-3**, the MCR is programmed to control an entire circuit. The operation of the program can be summarized as follows:

- The section or zone being controlled begins with the first MCR instruction and ends with the second MCR.
- When the first MCR instruction is false, or disabled, all nonretentive rungs below it, in this case, outputs M and PL1, will be de-energized even if the programmed logic for each rung is true.
- All retentive rungs, in this case SOL, will remain in their last state.
- Assume the motor M is running and the MCR instruction becomes disabled. The motor will immediately become de-energized and stop operating. When the MCR instruction then becomes

enabled, the motor will not revert back to its previous running state but will have to be restarted via the start pushbutton.

- Assume the level switch is closed and the MCR instruction becomes disabled. Pilot light PL1 will immediately become de-energized even though the level switch instruction is true and the rung appears to have logic continuity. When the MCR instruction then becomes enabled, PL1 will automatically be energized, provided the level switch has remained closed.
- Assume solenoid SOL has been latched energized, both limit switches LS1 and LS2 are open, and the MCR instruction becomes disabled. Solenoid SOL will remain energized. When the MCR instruction then becomes enabled, the SOL will remain energized, provided both LS1 and LS2 remained open.
- Assume solenoid SOL has been latched de-energized, both limit switches LS1 and LS2 are open, and the MCR instruction becomes disabled. Solenoid SOL will remain de-energized. When the MCR instruction then becomes enabled, the SOL will remain de-energized, provided both LS1 and LS2 remained open.
- Retentive instructions should not normally be placed within an MCR zone because the MCR zone maintains retentive instructions in the state last active when the instruction disabled.

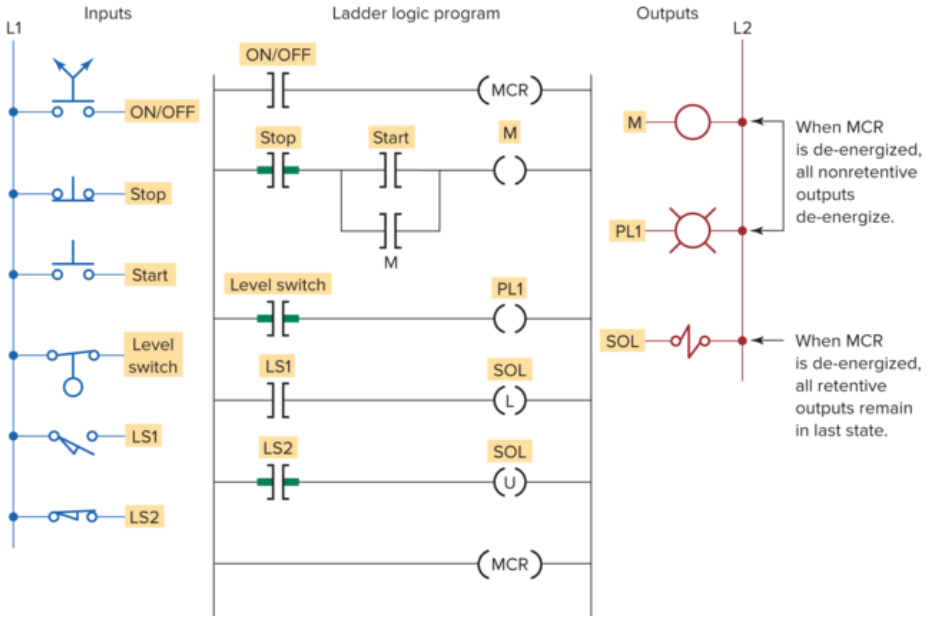


Figure 9-3 Master Control Reset (MCR) instruction.

PLC controllers use the master control reset instruction to set up single or multiple zones within a program. The MCR instruction is used in pairs to disable or enable a zone within a ladder program, and it has ***no address.*** **Figure 9-4** shows the programming of an MCR fenced zone with the zone active or true. The operation of the program can be summarized as follows:

- The MCR zone is enclosed by a ***start fence,*** which is a rung with a conditional MCR, and an ***end fence,*** which is a rung with an unconditional MCR.

- Input *A* of the start rung is true so all outputs act according to their rung logic as if the zone did not exist.

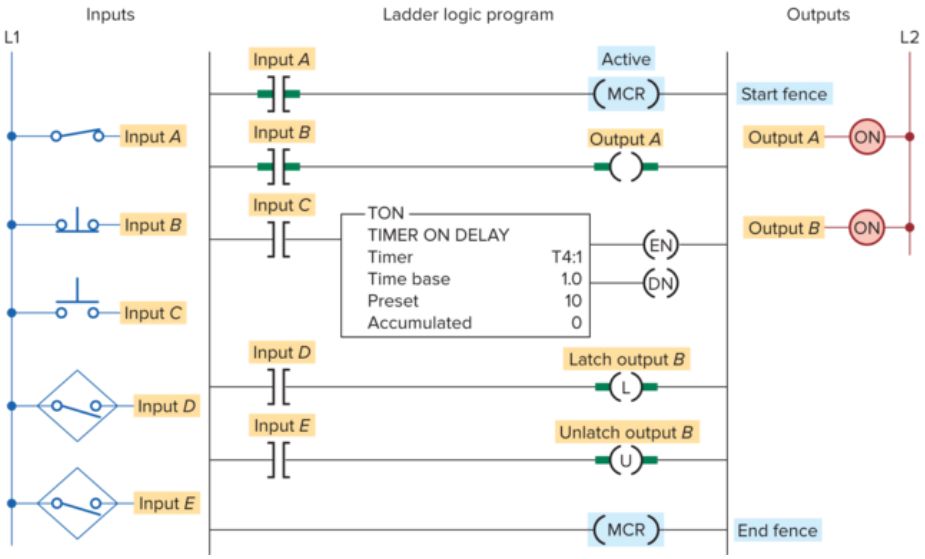


Figure 9-4 MCR fenced zone with the zone true.

Figure 9-5 shows the programmed MCR fenced zone with the zone not activated or false. The operation of the program can be summarized as follows:

- When the MCR in the start fence is false, all rungs within the zone are treated as false. The scan ignores the inputs and de-energizes all nonretentive outputs. Page 176
- All retentive devices, such as latches, retentive timers, and counters, remain in their last state. TOF timers will start timing when the MCR goes false.
- Input *A* of the start rung is false so output *A* and T4:1 will be false and output *B* will remain in its last state.
- The input conditions in each rung will have no effect on the output conditions.

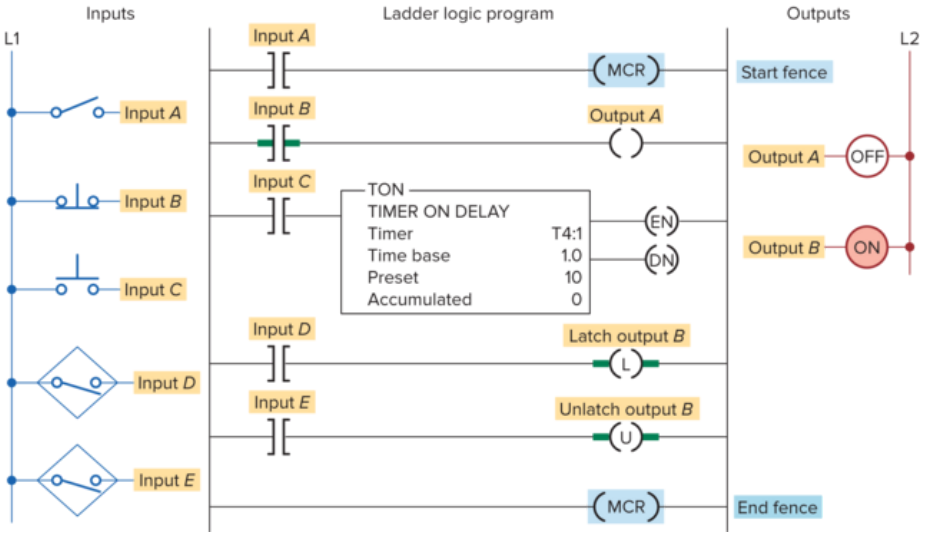


Figure 9-5 MCR fenced zone with the zone false.



A common application of an MCR zone control involves examining one or more fault bits as part of the start fence and enclosing the portion of the program you want de-energized in case of a fault in the MCR zone. In case of a detected fault condition, the outputs in that zone would be de-energized automatically.

If you start instructions such as timers or counters in an MCR zone, instruction operation ceases when the zone is disabled. The TOF timer will activate when placed inside a false MCR zone. When troubleshooting a program that contains an MCR zone, you need to be aware of which rungs are within zones in order to correctly edit the circuit.

MCR-controlled areas must contain only two MCR instructions—one to define the start and one to define the end. Never overlap or nest MCR zones. Any additional MCR instructions, or a jump instruction programmed to jump to an MCR zone, could produce unexpected and damaging results to your program and to machine operation.

In addition to controlling power to an entire system, MCRs are also used when only a portion of a program is required to be isolated. For example:

- Inhibiting zones of the program while loading recipes.
- Monitoring emergency stops.
- Establishing preconditions to synchronize a machine on start-up.

OceanofPDF.com

9.3 Jump and Label Instructions

In PLC programming it is sometimes desirable to be able to jump over certain program instructions when certain conditions exist. Used in combination the Jump (JMP) and Label (LBL) instructions allow for skipping over portions of the program. When the jump instruction is used, the PLC will not execute the instructions of a rung that is jumped. The jump instruction is often used to jump over instructions not pertinent to the machine's operation at that instant. In addition, sections of a program may be programmed to be jumped should a production fault occur.

Some manufacturers provide a skip instruction, which is essentially the same as the jump instruction.

The program of **Figure 9-6** illustrates the use of a jump instruction. In this example, Addresses Q2:0 through Q2:255 are the addresses used for the jump (JMP) instructions. The Q2 is internal and provided by the software as you program the JMP instruction. The Q2 simply identifies this as ladder file 2. A JMP instruction in ladder file 3 would be Q3. The ***label (LBL)*** instruction is a target for the jump instruction.

- The jump instruction with its associated label instruction (LBL) must have the same address.
- The area of the program that the processor jumps over is defined by the locations of the jump and label instructions in the program.
- When the jump instruction is true, all logic between the jump and label instructions is bypassed and the processor continues scanning after the LBL instruction.
- The label instruction must be programmed as the first instruction on the rung where it resides.
- The label instruction is always true, and the remaining instructions on the rung must make up a verifiable rung.
- The instructions to the right of the LBL on the label rung are outside the jump zone and as such are not affected by the jump.

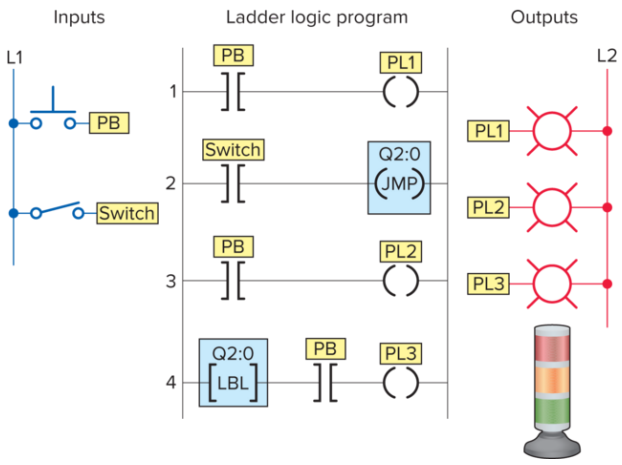


Figure 9-6 Jump (JMP) operation.

The operation of the program can be summarized as follows:

- When the switch is open the jump instruction is not activated. Page 177
- With the switch open, closing PB turns on all three pilot lights.
- When the switch is closed the jump (JMP) instruction will activate.
- With the switch closed, pressing PB turns on pilot lights PL1 and PL3 only.
- Rung 3 is skipped over during the PLC program scan so PL2 will remain in its last state before the JMP was enabled.

Figure 9-7 illustrates the effect on input and output instructions of jumped rungs in a program. The label instruction is used to identify the ladder rung that is the target destination but does not contribute to logic continuity. For practical purposes the label instruction is always considered to be logically true. The operation of the program can be summarized as follows:

- Rungs 1, 2, 3, 8, 9, 10 are programmed outside of the jumped section and will always be executed as normal rungs.
- If rung 4, which contains the JMP instruction, is **false,** the Jump instruction is false and the jump is not executed.
- Rungs 5, 6, and 7 are executed as normal and the label instruction on rung 8 is transparent. Page 178
- When rung 4, containing the JMP instruction, is **true,** the processor is instructed to jump to the LBL target in rung 8 and continue to execute the main program from that point.
- Instructions to the right of the LBL are out of the jump zone and are executed as a normal rung.
- Jumped rungs 5, 6, and 7 are not scanned by the processor.
- Input conditions for the jumped rungs are not examined and outputs controlled by these rungs remain in their last state.
- Any timers or counters programmed within the jump area cease to function and will not update themselves during this period. For this reason they are usually programmed outside the jumped section in the main program zone.
- This is called a forward jump, as we are jumping forward in the program.

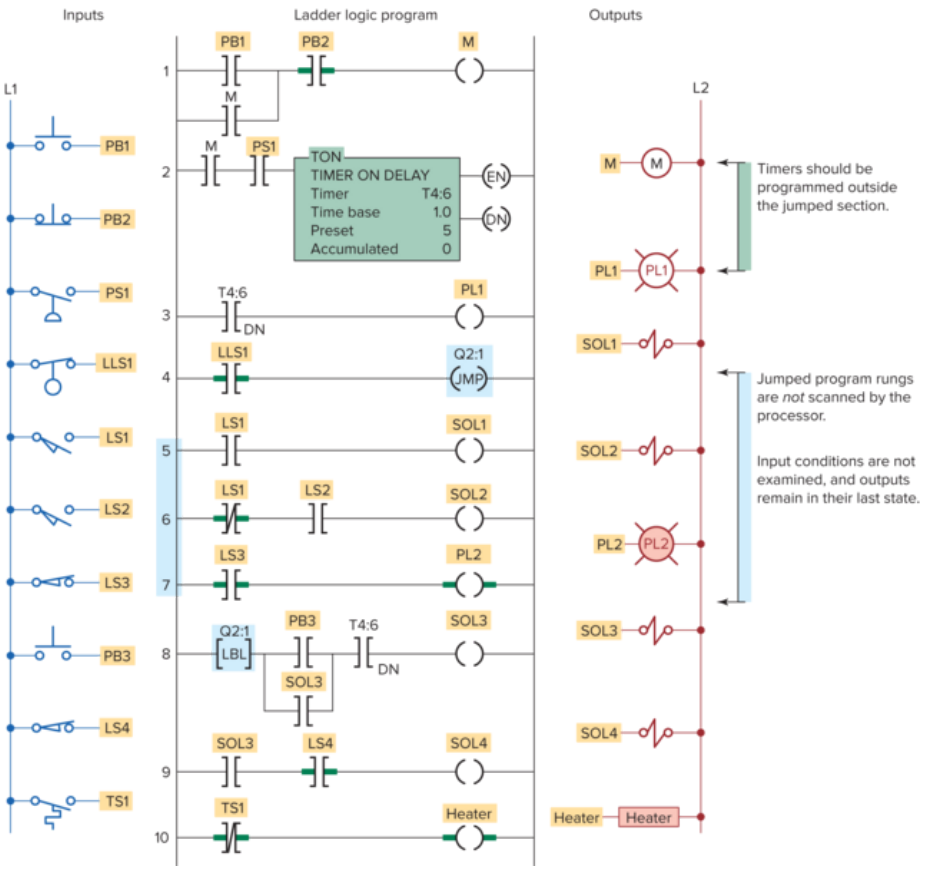


Figure 9-7 Effect on input and output instructions of jumped rungs.

You can jump to the same label from multiple jump locations, as illustrated in the program of **Figure 9-8**. In this example, there are two jump instructions addressed Q2:5. There is a single label instruction addressed Q2:5. The scan can then jump from either jump instruction to label Q2:5, depending on whether input *A* or input *D* is true.

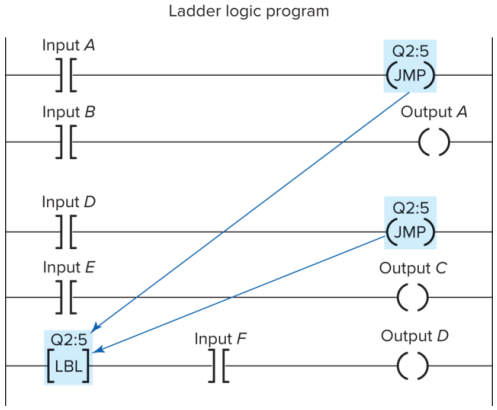


Figure 9-8 Jump-to-label from two locations.

It is possible to jump backward in the program, but this should not be done an excessive number of times. Care must be taken that the scan does not remain in a loop too long. The processor has a watchdog timer that sets the maximum allowable time for a total program scan. If this time is exceeded, the processor will indicate a fault and shut down.

The forward jump is similar to an MCR instruction in that both permit an input logic condition to skip over a block of PLC ladder logic. The main difference between the two is in how the outputs are handled when the instructions are executed. The MCR instruction sets all nonretentive outputs to the false state and keeps the retentive outputs in their last state. The JMP instruction leaves all outputs in their last state. You should never jump into a Master Control Reset zone. If you do, instructions that are programmed within the MCR zone starting at the LBL instruction and ending at the end MCR instruction will always be evaluated as though the MCR zone is true, without consideration to the state of the start MCR instruction.

OceanofPDF.com

9.4 Subroutine Functions

In addition to the main ladder logic program, PLC programs may also contain additional program files known as *subroutines.* A subroutine is a short program that is used by the main program to perform a specific function. Large programs are often broken into subroutine program files, which are called and executed from the main program.

Use of subroutines is a valuable tool in PLC programming. At times it is better to construct programs that consist of several subroutines than a lengthy single program. When programs are written with subroutines, each subroutine can be tested individually for functionality. These subroutines can then be called from the main program as illustrated in **Figure 9-9**.

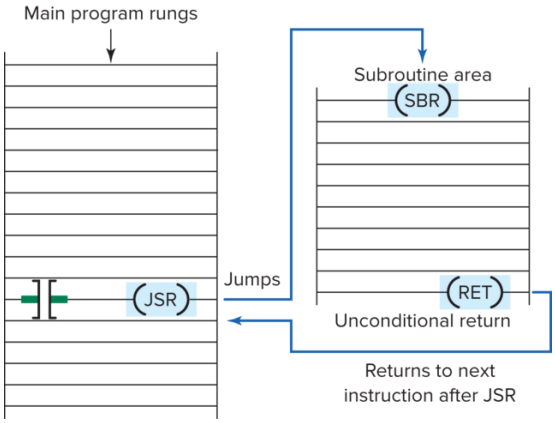


Figure 9-9 Main program with a call from a subroutine.

When a subroutine is called from the main program, the program is able to escape from the main program and *go to* a program *subroutine* to perform certain functions and *then return* to the main program. In situations in which a machine has a portion of its cycle that must be repeated several times during one machine cycle, the subroutine can save a great deal of duplicate programming. The sequence of rungs could be programmed one time into a subroutine and just called when needed.

The subroutine concept is the same for all programmable controllers, but the method used to call and return from a subroutine uses different commands, depending on the PLC manufacturer. The subroutine-related instructions used in the Allen-Bradley PLCs shown in **Figure 9-10** are the jump to subroutine (JSR) output instruction, the subroutine (SBR) input instruction, and the return (RET) output instruction.

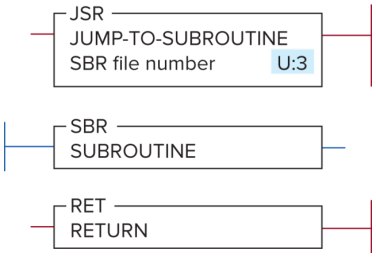


Figure 9-10 Allen-Bradley subroutine-related instructions.

The subroutine instructions can be summarized as follows:

Jump to Subroutine (JSR)—The JSR instruction redirects logic execution from the current ladder file to the specific subroutine file. When rung conditions are true for this output instruction, it causes the processor to jump to the targeted subroutine file.

Subroutine (SBR)—The SBR instruction is the first input instruction on the first rung in the subroutine file. It serves as an identifier that the program file is a subroutine. The subroutine can have a file number or a descriptive name in the JSR instruction to identify the target to which the program should jump depending on the programming software being used. It is always true, and although its use is optional, it is still recommended.

Return (RET)—The RET instruction is an output instruction that marks the end of the subroutine file. It causes the scan to return to the main program at the instruction following the JSR instruction where it exited the program. The scan returns from the end of the file if there is no RET instruction. The rung containing the RET instruction may be conditional if this rung precedes the end of the subroutine. In this way, the processor omits the balance of a subroutine only if its rung condition is true.

The jump to subroutine (JSR), subroutine (SBR), and return (RET) instructions are used to direct the controller to execute a subroutine file. **Figure 9-11** shows a materials conveyor control process with a flashing pilot light as a subroutine. The operation of the program can be summarized as follows:

- If the weight on the conveyor exceeds a preset value, the solenoid is de-energized and pilot light PL1 will begin flashing.
- When the weight sensor switch closes, the JSR is activated and directs the processor scan to jump to the subroutine U:3.
- The subroutine program is scanned and pilot light PL1 begins flashing.
- When the weight sensor switch opens, the processor will no longer scan the subroutine area and pilot light PL1 will return to its normal on state.

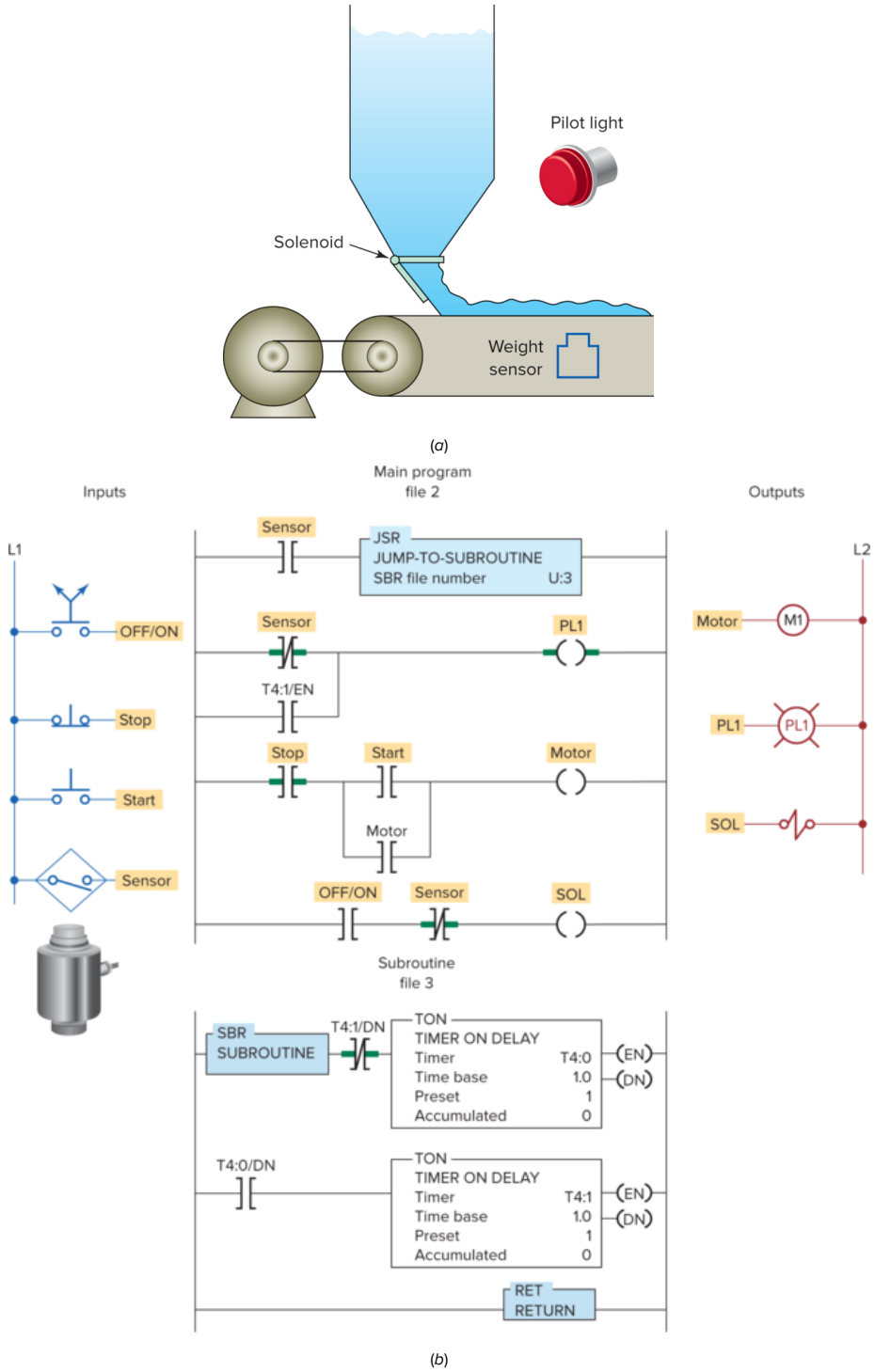


Figure 9-11 Flashing pilot light subroutine. (a) Process. (b) Program.

The RSLogix 500 main program is located in program file 2 whereas, subroutines are assigned to program file numbers 3 to 255. Each subroutine must be programmed in its own program file by

assigning it a unique file number. Figure 9-12 illustrates the procedure for setting up a subroutine and can be summarized as follows:

- Note each ladder location where a subroutine should be called.
- Create a subroutine file for each location. Each subroutine file should begin with an SBR instruction.
- At each ladder location where a subroutine is called, program a JSR instruction specifying the subroutine file number.
- The RET instruction is optional.
  - The end of a subroutine program will cause a return to the main program.
  - If you want to end a subroutine program before it executes to the end of program file, a conditional return (RET) instruction may be used.

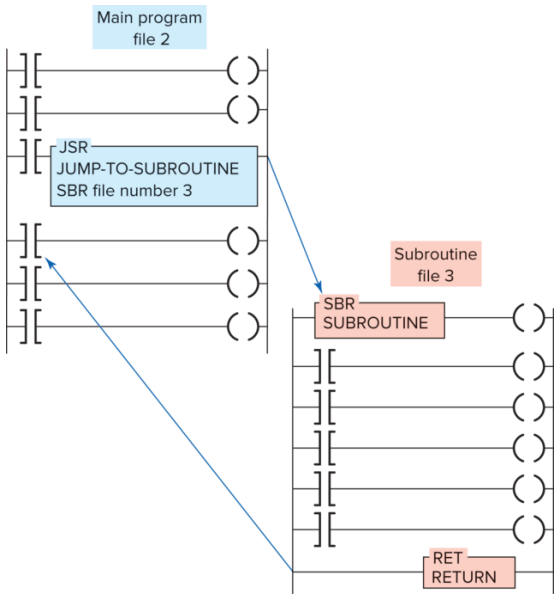


Figure 9-12 Setting up a subroutine file.

Nesting subroutines allows you to direct program flow from the main program to a subroutine and then to another subroutine, as illustrated in Figure 9-13. Nested subroutines make complex programming easier and program operation faster because the programmer does not have to continually return from one subroutine to enter another. Programming nested subroutines may cause scan time problems because while the subroutine is being scanned, the main program is not. Excessive delays in scanning the main program may cause the outputs to operate later than required. This situation may be avoided by updating critical I/O using immediate input and/or immediate output instructions.

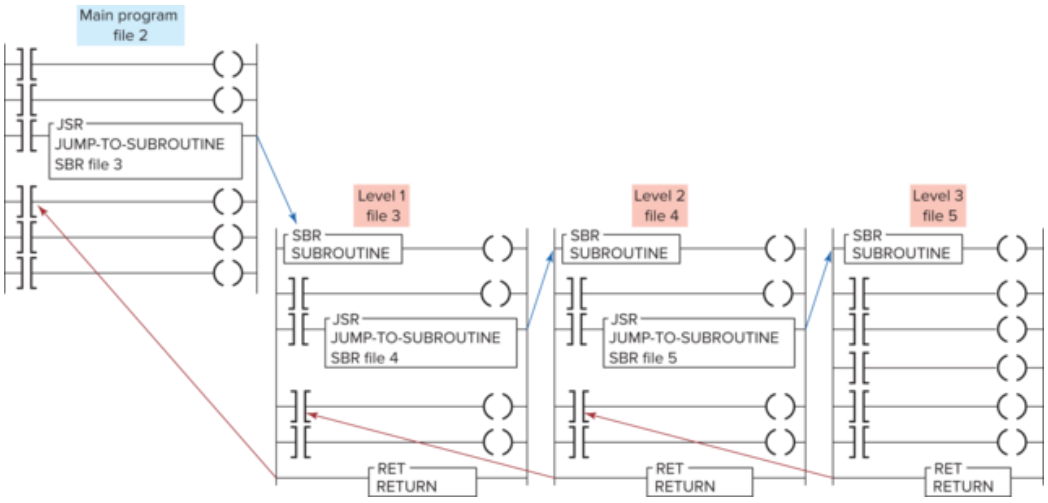


Figure 9-13 Nested subroutines.

OceanofPDF.com

9.5 Immediate Input and Immediate Output Instructions

The PLC input scan normally records the inputs before the program scan, and the output scan normally updates the outputs after the program scan. **Immediate I/O** instructions allow you to update data prior to the normal input scan as illustrated in **Figure 9-14**.

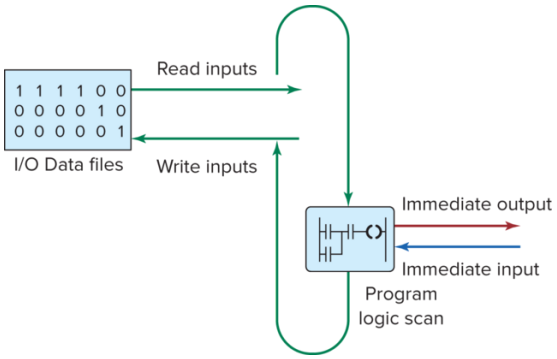


Figure 9-14 Immediate I/O instructions.

Immediate I/O instructions interrupt the normal program scan to update the input image table file with current input data or to update an output module group with the current output image table file data. Immediate I/O instructions for RSLogix 500 are called **immediate input with mask (IIM)** and **immediate output with mask (IOM).**

- Masking is a means of selectively screening out data.
- Masking allows the programmer to specify which of the 16 bits are to be copied from an input module to the input image data table (or from the output image table to an output module).
- The other bits in the input image table or output module are not affected by these instructions.

The immediate input with mask (IIM) instruction is shown in **Figure 9-15**. The **IIM instruction** operates on the inputs assigned to a particular word of a slot. When the IIM rung is true, the program scan is interrupted, and data from a specific input slot are transferred through the mask to the input data file. These data are then available to the commands in the ladder following the IIM instruction. The following parameters are entered in the instruction:

Slot Specifies the slot and word that contain the data to be updated. For example, I:3.0 means the input of slot 3, word 0.

Mask Specifies either a hex constant or a register address. For the mask, a 1 in the bit position passes data from the source to the destination. A 0 inhibits or blocks bits from passing from the source to the destination.

Length Used to transfer more than one word per slot.

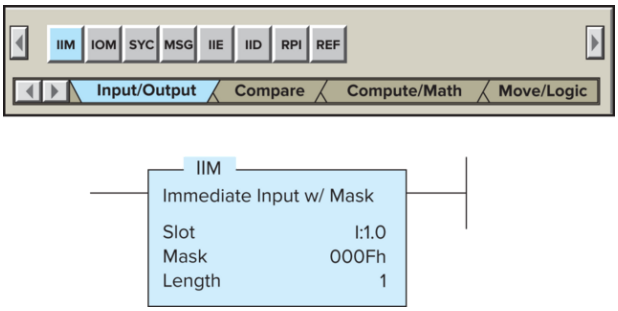


Figure 9-15 Immediate input with mask (IIM) instruction.

The program operation of the instruction is summarized as follows:

- The IIM instruction retrieves data from I:1.0 and passes it through the mask.
- The mask permits only the four least significant bits to be moved to the input register I:1.0.
- This allows the programmer to update only sections of the inputs to be used throughout the rest of the program.

The immediate output with mask **(IOM) instruction** is shown in **Figure 9-16**. The IOM operates on the physical outputs assigned to a particular word of a slot. When the IOM rung is true, the program scan is interrupted to update output data to the module located in the slot specified in the instruction. These data are then available to the commands in the ladder following the IOM instruction. The parameters entered are basically the same as those entered for the IIM instruction.

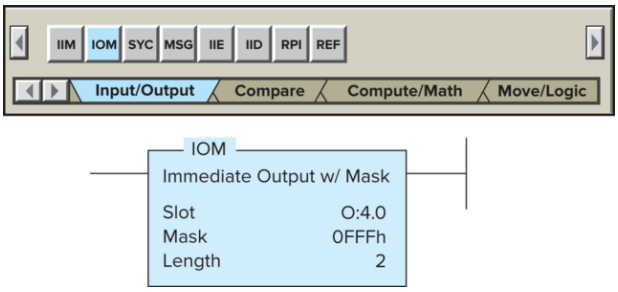


Figure 9-16 Immediate output with mask (IOM) instruction.

Processor communication with the local chassis is many times faster than communication with the remote chassis. This is due to the fact that local I/O scan is synchronous with the program scan and communication is in *parallel* with the processor, whereas the remote I/O scan is asynchronous with the program scan and communication with remote I/O is *serial.* For this reason, fast-acting devices should be wired into the local chassis.

RSLogix 5000 PLCs have no immediate input instruction as they use asynchronous I/O control compared to the RSLogix 500 controllers which use synchronous I/O control. RSLogix 5000 controllers do have an immediate output (IOT) instruction, which operates the same as the immediate output instruction for RSLogix 500. **Figure 9-17** shows an example of the IOT instruction. In this example, when the IOT instruction executes, it immediately updates the entire

output module Local:3:0. When you use the instruction to update the output card, address the entire card (Local:3:0), and not the individual outputs (Local:3:0.Data 0).

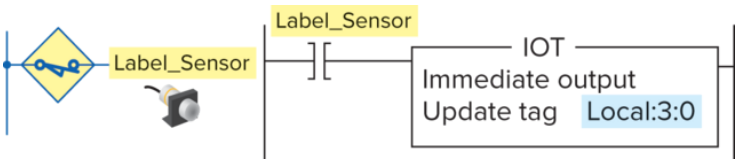


Figure 9-17 ControlLogix immediate output instruction.

OceanofPDF.com

9.6 Forcing External I/O Addresses

The force function is used to override the state of a physical input or output address. I/O can be forced on or off regardless of the state of the actual field device or program execution. The forcing capability allows a machine or process to continue operation until a faulty field device can be repaired. It is also valuable during start-up and troubleshooting of a machine or process to simulate the action of portions of the program that have not yet been implemented.

Forcing inputs manipulates the input image table file bits and thus affects *all* areas of the program that use those bits. The forcing of inputs is done just after the input scan. When we force an input address, we are forcing the status bit of the instruction at the I/O address to an on or off state. **Figure 9-18** illustrates how an input is forced on. The operation of the program can be summarized as follows:

- The processor ignores the actual state of input limit switch I:1/3.
- Although limit switch I:1/3 is off (0 or false) the processor considers it as being in the on (1 or true) state.
- The program scan records this, and the program is executed with this forced status.
- In other words, the program is executed as if the limit switch were actually closed.

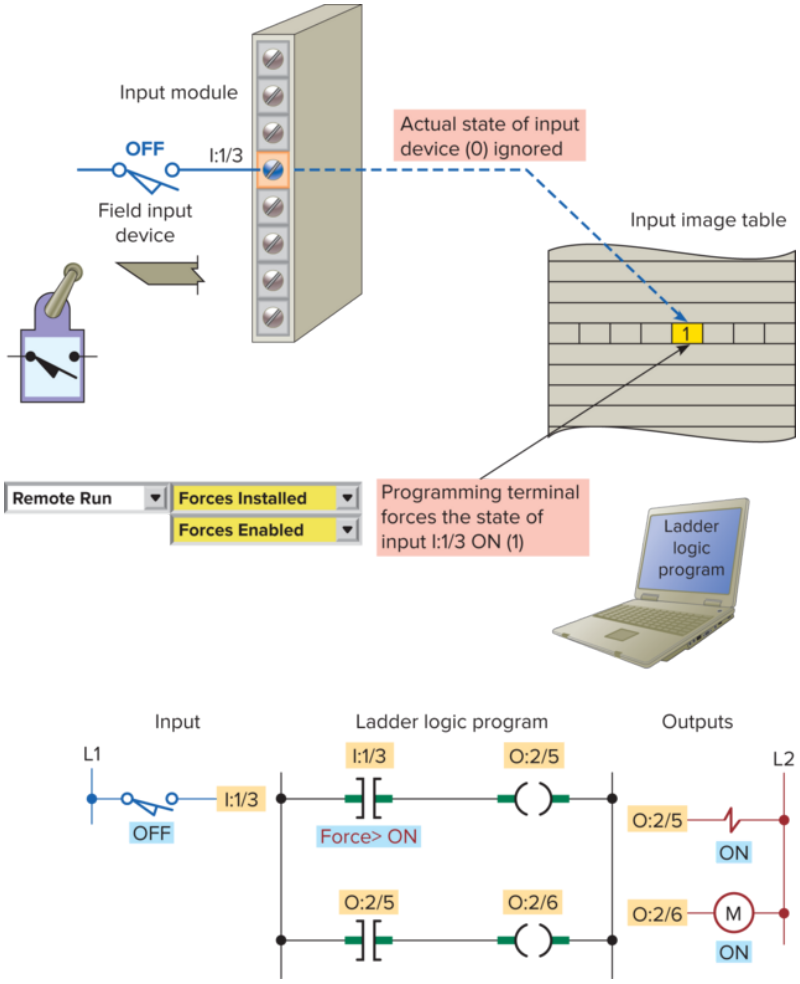


Figure 9-18 Forcing an input on.

Forcing outputs affects *only* the addressed output terminal. Therefore, since the output image table file bits are unaffected, your program will be unaffected. When we force an output address, we are forcing only the output terminal to an on or off state. The status bit of the output instruction at the address is usually not affected. **Figure 9-19** illustrates how an output is forced on. The operation of the program can be summarized as follows:

- The processor ignores the actual state of solenoid output O:2/5.
- The programming device sets the force state in the output force data file and the PLC implements the force to turn solenoid output O:2/5 on even though the output image table file indicates that the user logic is setting the point to off.
- M output O:2/6 remains off because the status bit of output O:2/5 is not affected by the force instruction.

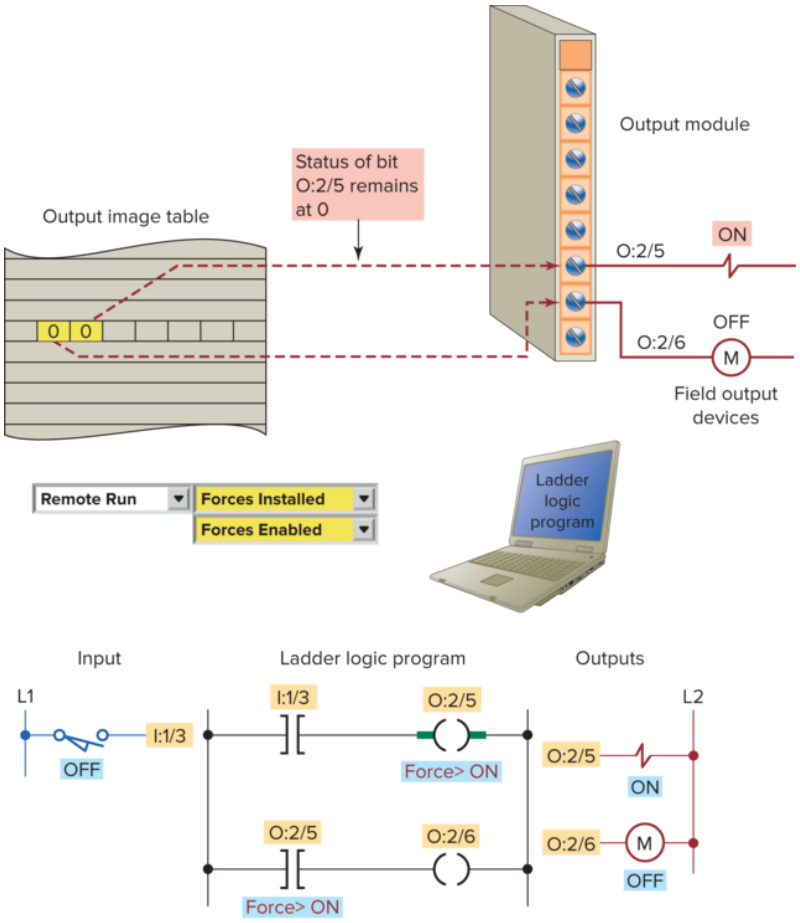


Figure 9-19 Forcing an output on.

Overriding of physical inputs on conventional relay control systems can be accomplished by installing hardwire jumpers. With PLC control, hardwire jumpers are not necessary because the input data table values can be forced to an on or off state. The force function allows you to override the actual status of external input circuits by forcing external data bits on or off. Similarly, you can override the processor logic and status of output data file bits by forcing output bits on or off. By forcing outputs off, you can prevent the controller from energizing those outputs even though the ladder logic, which normally controls them, may be true. In other instances, outputs may be forced on even though logic for the rungs controlling those outputs may be false.

It is possible to enter and enable or disable forces while you are monitoring your file offline, or in any processor mode while monitoring your file online. With RSLogix 500 software, the steps are as follows:

1. Open the program file in which you want to force the logic on or off.
2. With the right mouse button, click the I/O bit you want to force.
3. From the menu that appears, select Go to Data Table or select Force On or Force Off.
4. From the associated data table that appears, click on the Forces button.

5. The Forces version of the data table appears with the selected bit highlighted. Click on this bit with the right mouse button.
6. From the menu that appears, you can force the selected bit on or off.

Exercise care when you use forcing functions. **If used incorrectly, force functions can cause injuries to persons working around a system, and/or equipment damage.** For this reason, forcing functions should be used only by personnel who completely understand the circuit and the process machinery or driven equipment. You must understand the potential effect that forcing given inputs or outputs will have on machine operation in order to avoid possible personal injury and equipment damage. Before using a force function, check whether the force acts on the I/O point only or whether it acts on the user logic as well as on the I/O point. Most programming terminals and PLC CPUs provide some visible means of alerting the user that a force is in effect.

In situations in which rotating equipment is involved, the force instruction can be extremely dangerous. For example, if maintenance personnel are performing routine maintenance on a de-energized motor, the machine may suddenly become energized by someone forcing the motor to turn on. This is why a hardwired master control circuit is required for the I/O rack. The hardwired circuit will provide a method of physically removing power to the I/O system, thereby ensuring that it is impossible to energize any inputs or outputs when the master control is off.

OceanofPDF.com

9.7 Safety Circuitry

Sufficient emergency circuits must be provided to stop either partially or totally the operation of the controller or the controlled machine or process. These circuits should be **hardwired outside the controller** so that in the event of total controller failure, independent and rapid shutdown is available.

Figure 9-20 shows typical safety wiring requirements for a PLC installation. The safety requirements of this installation can be summarized as follows:

Page 186

- A main disconnect switch is installed on the incoming power lines as a means of removing power from the entire programmable controller system.
- The main power disconnect switch should be located where operators and maintenance personnel have quick and easy access to it. Ideally, the disconnect switch is mounted on the outside of the PLC enclosure so that it can be accessed without opening the enclosure.
- In addition to disconnecting electrical power, you should de-energize, lock out, and tag all other sources of power (pneumatic and hydraulic) before you work on a machine or process controlled by the controller.
- An isolation transformer is used to isolate the controller from the main power distribution system and step the voltage down to 120 VAC.
- A hardwired master control relay is included to provide a convenient means for emergency controller shutdown. Because the master control relay allows the placement of several emergency-stop switches in different locations, its installation is important from a safety standpoint.
- Overtravel limit switches or mushroom head emergency stop pushbuttons are wired in series so that when one of them opens, the master control is de-energized.
- This removes power to input and output device circuits. Power continues to be supplied to the controller power supply so that any diagnostic indicators on the processor module can still be observed.
- Note that the master control relay is not a substitute for a disconnect switch. When you are replacing any module, replacing output fuses, or working on equipment, the main disconnect switch should be pulled and locked out.

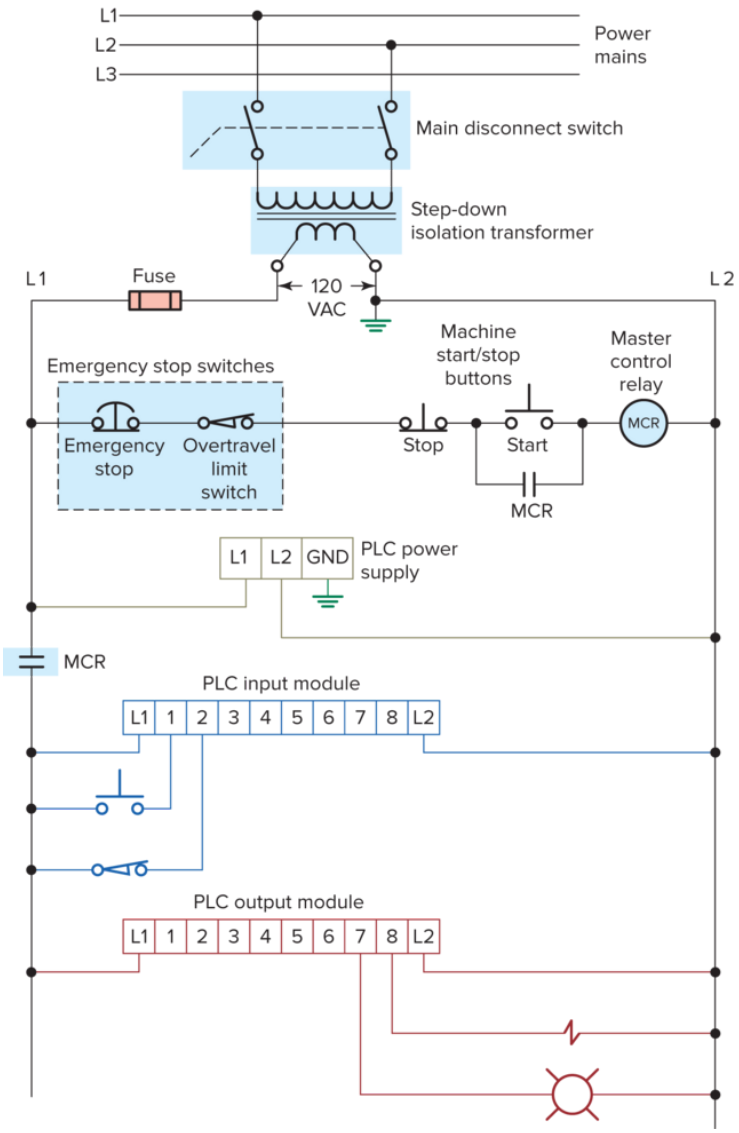
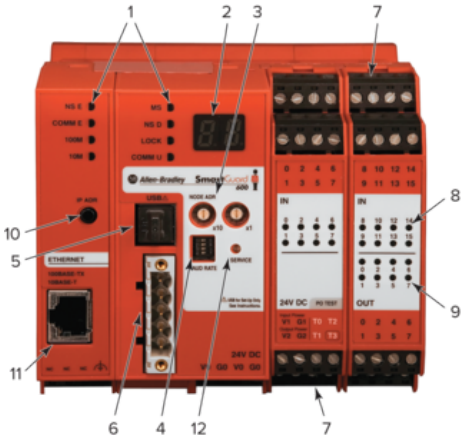


Figure 9-20 Safety wiring requirements for a PLC installation.

The ***master control relay*** must be able to inhibit all machine motion by removing power to the machine I/O devices when the relay is de-energized. This hardwired electromechanical component must not be dependent on electronic components (hardware or software). Any part can fail, including the switches in a master control relay circuit. The failure of one of these switches would most likely cause an open circuit, which would be a safe power-off failure. However, if one of these switches shorts out, it no longer provides any safety protection. These switches should be tested periodically to ensure that they will stop machine motion when needed. Never alter these circuits to defeat their function. Serious injury or machine damage could result.

Safety PLCs, such as the one shown in **Figure 9-21**, are now available for applications that require more advanced safety functionality. A safety PLC is typically certified by third parties to meet rigid safety and reliability requirements of international standards. Both standard and safety PLCs have the

ability to perform control functions but a standard PLC was not initially designed to be fault tolerant and fail-safe. That is the fundamental difference.



Number	Feature
1	Module status indicators
2	Alphanumeric display
3	Node address switches
4	Baud rate switches
5	USB port
6	DeviceNet communication connector
7	Terminal connectors
8	Input status indicators
9	Output status indicators
10	IP address display switch
11	Ethernet connector
12	Service switch

Figure 9-21 Safety PLC.

Source: Image Courtesy of Rockwell Automation, Inc.

Some of the differences between standard and safety PLCs include the following:

- A standard PLC has one microprocessor that executes the program, Flash memory area that stores the program, RAM for making calculations, ports for communications, and I/O for detection and control of the machine. In contrast, a safety PLC has redundant microprocessors, Flash and RAM that are continuously monitored by a watchdog circuit, and a synchronous detection circuit. *Redundancy* is duplication. The probability of hazards arising from one malfunction in an electrical circuit can be minimized by creating partial or complete redundancy (duplication).

Page 187

- Standard PLC inputs provide no internal means for testing the functionality of the input circuitry. By contrast, safety PLCs have an internal output circuit associated with each input for the purpose of testing the input circuitry. Inputs are driven both high and low for very short cycles during runtime to verify their functionality.
- Safety PLCs use power supplies designed specifically for use in safety control systems and redundant backplane circuitry between the controller and I/O modules.

Safety considerations should be developed as part of the PLC program. A PLC program for any application will be only as safe as the time and thought spent on both personnel and hardware considerations make it. One such consideration involves the use of a motor starter ***auxiliary seal-in contact,*** shown in **Figure 9-22**, in place of the programmed contact referenced to the output coil instruction. The use of the field-generated starter auxiliary contact status in the program is more costly in terms of field wiring and hardware, but it is *safer* because it provides positive feedback to the processor about the exact status of the motor. Assume, for example, that the OL contact of the starter opens under an overload condition. The motor, of course, would stop operating because power would be lost to the starter coil. If the program was written using an examine-on contact instruction referenced to the output coil instruction as the seal-in for the circuit, the processor would never know that power had been lost to the motor. When the OL was reset, the motor would restart instantly, creating a potentially unsafe operating condition.

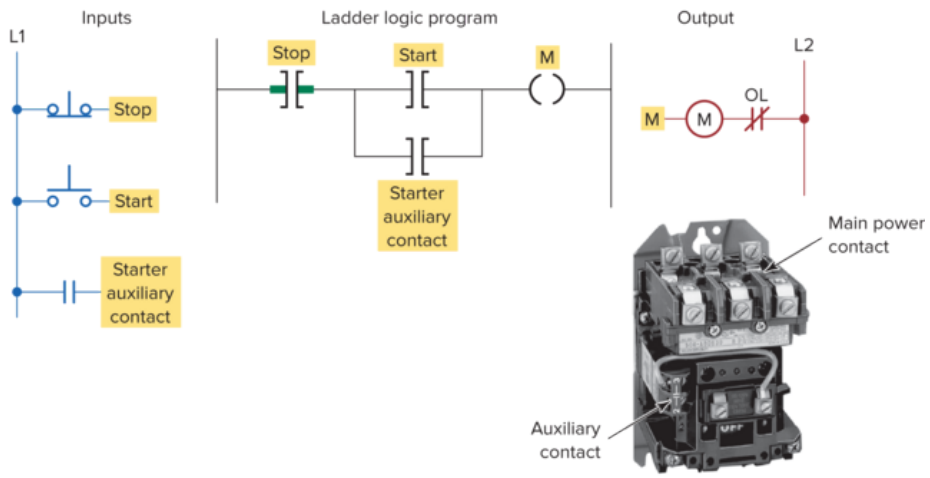


Figure 9-22 Motor starter programmed using the starter auxiliary seal-in contact.

Source: Image Courtesy of Rockwell Automation, Inc.

Another safety consideration concerns the ***wiring of stop buttons.*** A stop button is generally considered a safety function as well as an operating function. As such, ***all stop buttons should be wired using a normally closed contact programmed to examine for an on condition*** (Figure 9-23). Using a normally open contact programmed to examine for an off condition will produce the same logic but is not considered to be as safe. Assume that the latter configuration is used. If, by some chain of events, the circuit between the button and the input point were to be broken, the stop button could be depressed forever, but the PLC logic could never react to the stop command because the input would never be true. The same holds true if power were lost to the stop button control circuit. If the normally closed wiring configuration is used, the input point receives power continuously unless the stop function is desired. Any faults occurring with the stop circuit wiring, or a loss of circuit power, would effectively be equivalent to an intentional stop.

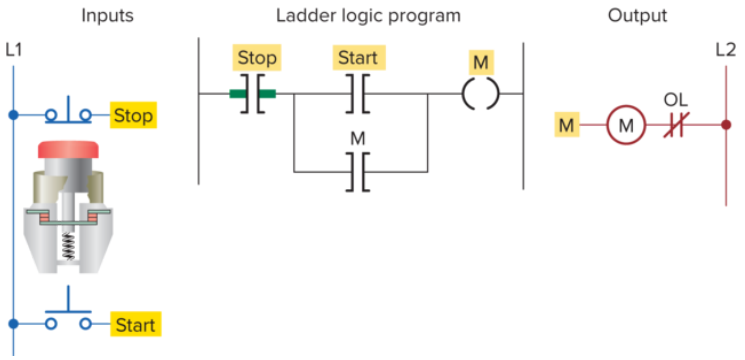


Figure 9-23 Wiring of stop buttons.

OceanofPDF.com

9.8 Fault Routine

Fault routines are fault handling instructions that are used to execute subroutines which respond to errors. PLC fault handling capability is measured in terms of how the PLC system informs the user of such conditions and what types of alternatives are available for dealing with these conditions.

There are two kinds of major faults that result in a processor fault: recoverable and nonrecoverable faults. When the processor detects a major fault, it looks for a fault routine. If a fault routine exists, it is executed; if one does not exist, the processor shuts down. When there is a fault routine, and the fault is *recoverable,* the fault routine is executed. If the fault is *nonrecoverable,* the fault routine is scanned once and shuts down. Either way, the fault routine allows for an orderly shutdown.

OceanofPDF.com

9.9 Temporary End Instruction

The ***temporary end (TND)*** instruction is an output instruction used to progressively debug a program or conditionally omit the balance of your current program file or subroutines. When rung conditions are true, this instruction stops the program scan, updates the I/O, and resumes scanning at rung 0 of the main program file.

Figure 9-24 illustrates the use of the TND instruction in troubleshooting a program. The TND instruction lets your program run only up to this instruction. You can move it progressively through your program as you debug each new section. You can program the TND instruction unconditionally, or you can condition its rung according to your debugging needs.

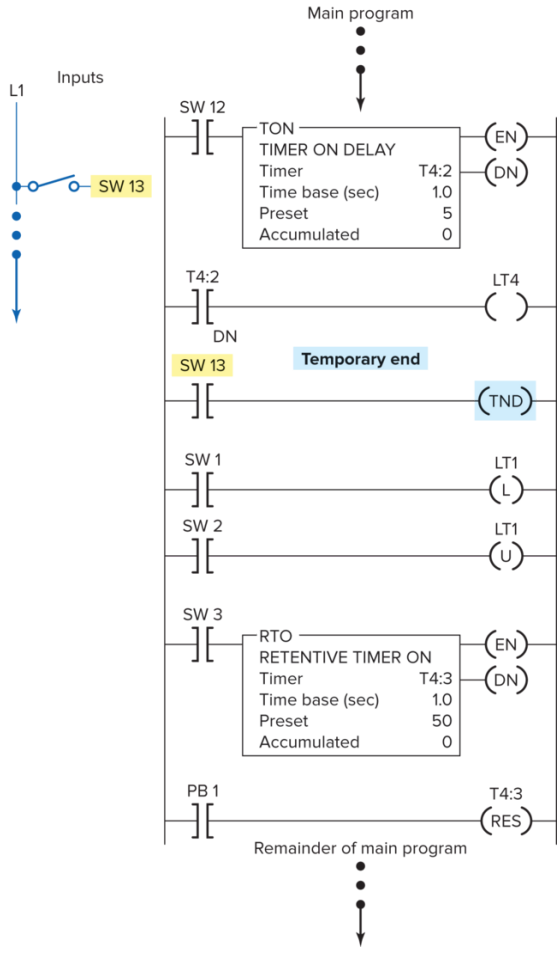


Figure 9-24 Temporary end (TND) instruction.

OceanofPDF.com



CHAPTER 9 REVIEW QUESTIONS

1. **a.** Two MCR output instructions are to be programmed to control a section of a program. Explain the programming procedure to be followed.
   b. State how the status of the output devices within the fenced zone will be affected when the MCR instruction makes a false-to-true transition.
   c. State how the status of the output devices within the fenced zone will be affected when the MCR instruction makes a true-to-false transition.
2. What is the main advantage of the jump instruction?
3. What types of instructions are not normally included inside the jumped section of a program? Why?
4. **a.** What is the purpose of the label instruction in the jump-to-label instruction pair?
   b. When the jump-to-label instruction is executed, in what way are the jumped rungs affected?
5. **a.** Explain what the jump-to-subroutine instruction allows the program to do.
   b. In what type of machine operation can this instruction save a great deal of duplicate programming?
6. What advantage is there to the nesting of subroutines?
7. **a.** When are the immediate input and immediate output instructions used?
   b. Why is it of little benefit to program an immediate input or immediate output instruction near the beginning of a program?
8. **a.** What does the forcing capability of a PLC allow the user to do?
   b. Outline two practical uses for forcing functions.
   c. Why should extreme care be exercised when using forcing functions?
9. Why should emergency stop circuits be hardwired instead of programmed?
10. State the function of each of the following in the basic safety wiring for a PLC installation:
    a. Main disconnect switch
    b. Isolation transformer
    c. Emergency stops
    d. Master control relay
11. Compare standard and safety PLCs with regard to:
    a. Processors
    b. Input circuitry
    c. Output circuitry

d. Power supplies

12. When programming a motor starter circuit, why is it safer to use the starter seal-in auxiliary contact in place of a programmed contact referenced to the output coil instruction?

13. When programming stop buttons, why is it safer to use an NC pushbutton programmed to examine for an on condition than an NO pushbutton programmed to examine for an off condition?

14. Explain the function of the fault routine file.

15. How is the temporary end instruction used to troubleshoot a program?

OceanofPDF.com



CHAPTER 9 PROBLEMS

1. Answer the questions, in sequence, for the MCR program in **Figure 9-25**, assuming the program has just been entered and the PLC is placed in the RUN mode with all switches turned off.

a. Switches S2 and S3 are turned on. Will outputs PL1 and PL2 come on? Why?

b. With switches S2 and S3 still on, switch S1 is turned on. Will output PL1 or PL2 or both come on? Why?

c. With switches S2 and S3 still on, switch S1 is turned off. Will both outputs PL1 and PL2 de-energize? Why?

d. With all other switches off, switch S6 is turned on. Will the timer time? Why?

e. With switch S6 still on, switch S5 is turned on. Will the timer time? Why?

f. With switch S6 still on, switch S5 is turned off. What happens to the timer? If the timer was an RTO type instead of a TON, what would happen to the accumulated value?

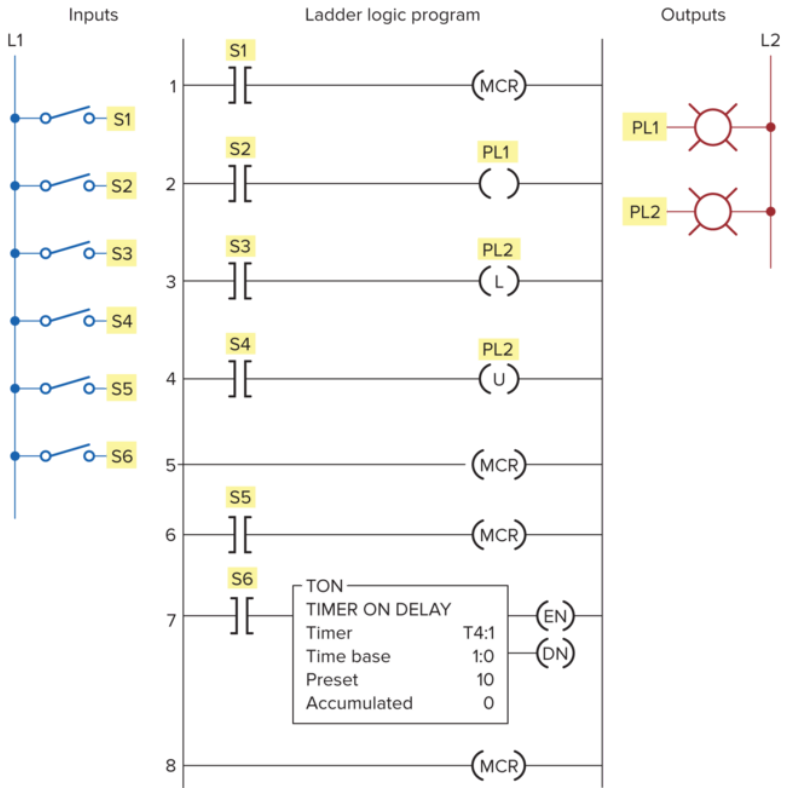


Figure 9-25 Program for Problem 1.

2. Answer the questions, in sequence, for the jump-to-label program in **Figure 9-26**. Assume all switches are turned *off after each operation.*

   a. Switch S3 is turned on. Will output PL1 be energized? Why?

   b. Switch S2 is turned on *first,* then switch S5 is turned on. Will output PL4 be energized? Why?

   c. Switch S3 is turned on and output PL1 is energized. Next, switch S2 is turned on. Will output PL1 be energized or de-energized after turning on switch S2? Why?

   d. All switches are turned on in order according to the following sequence: S1, S2, S3, S5, S4. Which pilot lights will turn on?

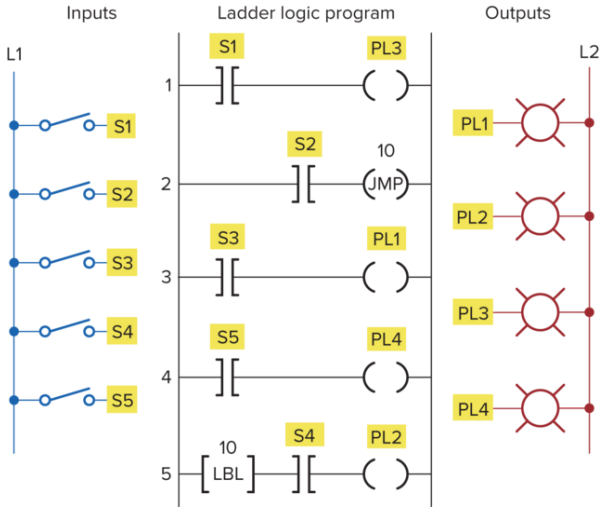


Figure 9-26 Program for Problem 2.

3. Answer the questions, in sequence, for the jump-to-subroutine and return program in **Figure 9-27**. Assume all switches are *turned off after each operation.*

   a. Switches S1, S3, S4, and S5 are all turned on. Which pilot light will *not* be turned on? Why?

   b. Switch S2 is turned on and then switch S4 is turned on. Will output PL3 be energized? Why?

   c. To what rung does the RET instruction return the program scan?

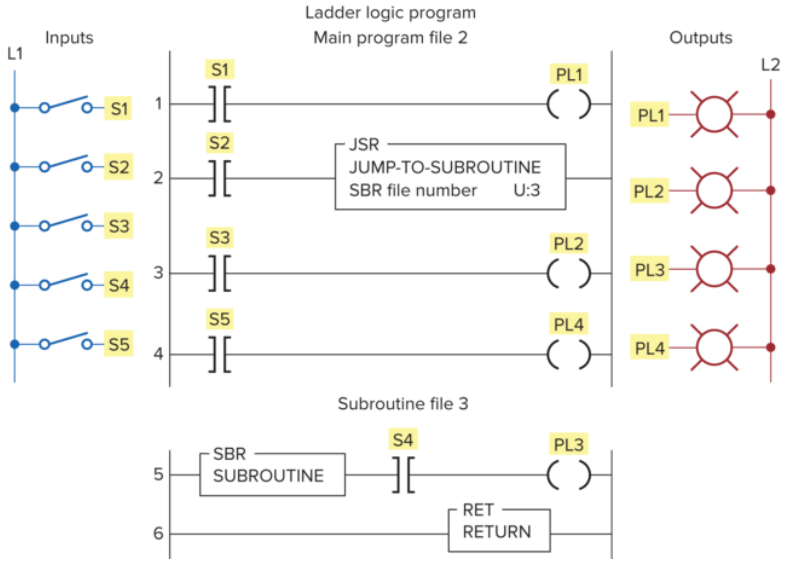


Figure 9-27 Program for Problem 3.

4. Answer the questions, in sequence, for **Figure 9-28**. Assume all switches are *turned off after each operation.*

   a. Switches S2, S12, and S5 are turned on in order. Will output PL5 be energized? Why?

   b. All switches except S7 are turned off. Will RTO start timing? Why?

   c. Switches S3 and S8 are turned on in order. Will pilot light PL2 come on? Why?

   d. When will timer TON function?

   e. Assume all switches are turned on. In what order will the rungs be scanned?

   f. Assume all switches are turned off. In what order will the rungs be scanned?

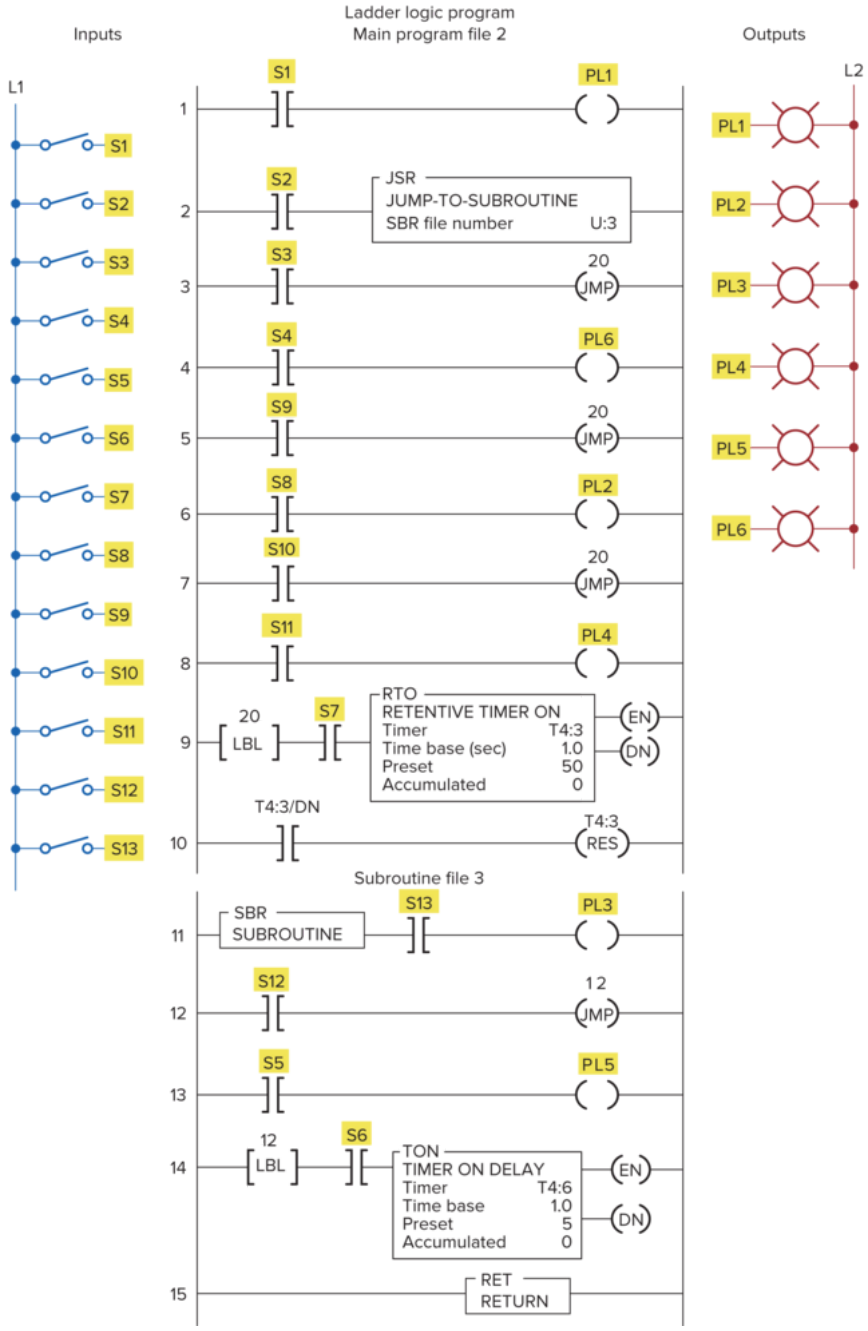
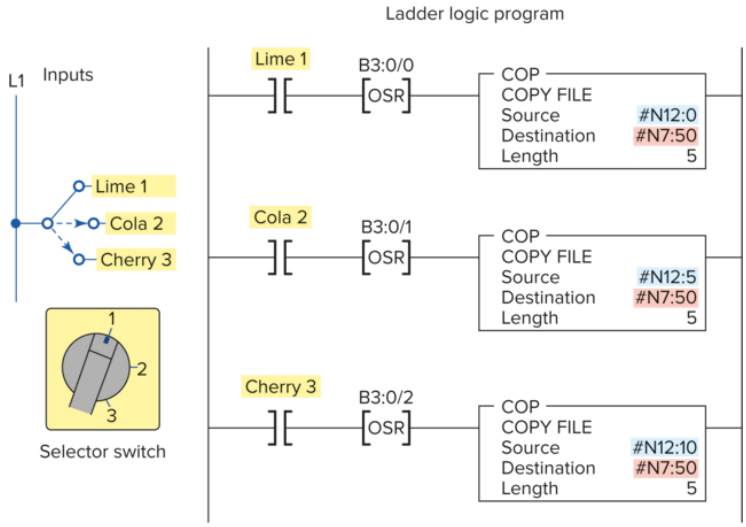


Figure 9-28 Program for Problem 4.

OceanofPDF.com

10

Data Manipulation Instructions



Chapter Objectives

After completing this chapter, you will be able to:

- Execute data transfer of word and file level instructions from one memory location to another
- Interpret data transfer and data compare instructions as they apply to a PLC program
- Compare the operation of discrete I/Os with that of multibit and analog types
- Understand the basic operation of PLC closed-loop control systems

Data manipulation involves transferring data and operating on data with math functions, data conversions, data comparison, and logical operations. This chapter covers both data manipulation instructions that operate on word data and those that operate on file data, which involve multiple words. Data manipulations are performed internally in a manner similar to that used in microcomputers. Examples of processes that need these operations on a fast and continuous basis are studied.

OceanofPDF.com

10.1 Data Manipulation

Data manipulation instructions allow numerical data stored in the controller's memory to be operated on within the control program. It includes operations involving moving or transferring numeric information stored in one memory word location to another word in a different location, and carrying out simple operations such as converting from one data format to another.

The use of data manipulation extends a controller's capability from that of simple on/off control based on binary logic, to quantitative decision making involving data comparisons, arithmetic, and conversions—which in turn can be applied to analog and positioning control.

There are two basic classes of instructions to accomplish data manipulation: instructions that operate on word data and those that operate on file, or block, data, which involve multiple words.

Each data manipulation instruction requires words of data memory for operation. The words of data memory in singular form may be referred to either as ***registers*** or as *words,* depending on the manufacturer. The terms *table* or *file* are generally used when a *consecutive* group of related data memory words is referenced. The data contained in files and words will be in the form of binary *bits* represented as series of 1s and 0s. A group of consecutive elements or words in an RSLogix 500 are referred to as a file.

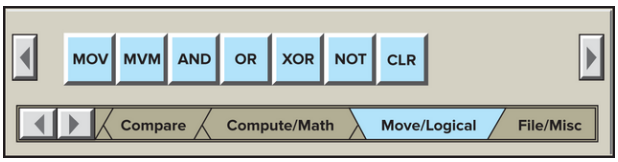


Figure 10-1 Move/Logical menu tab.

The data manipulation instructions allow the movement, manipulation, or storage of data in either single- or multiple-word groups from one data memory area of the PLC to another. Use of these PLC instructions in applications that require the generation and manipulation of large quantities of data greatly reduces the complexity and quantity of the programming required. Data manipulation can be placed in two broad categories: *data transfer* and *data comparison.*

The manipulation of entire words is an important feature of a programmable controller. This feature enables PLCs to handle inputs and outputs containing multiple bit configurations such as analog inputs and outputs. Arithmetic functions also require data within the programmable controller to be handled in word or register format.

Figure 10-1 shows the **Move/Logical** menu tab for the RSLogix software. The commands can be summarized as follows:

MOV (Move)—Moves the source value to the destination.

MVM (Masked Move)—Moves data from a source location to a selected portion of the destination.

AND (And)—Performs a bitwise AND operation.

OR (Or)—Performs a bitwise OR operation.

XOR (Exclusive Or)—Performs a bitwise XOR operation.

NOT (Not)—Performs a bitwise NOT operation.

BTD (Bit Field Distribute)—Copies the specified bits from the source, shifts the bits to the appropriate position, and writes the bits into the destination.

CLR (Clear)—Sets all bits of a word to zero.

OceanofPDF.com

10.2 Data Transfer Operations

Data transfer instructions simply involve the transfer of the contents from one word or register to another. **Figure 10-2*a*** and **Figure 10-2*b*** illustrate the concept of moving numerical binary data from one memory location to another. **Figure 10-2*a*** shows the original data are in register N7:30 and N7:20. **Figure 10-2*b*** shows that after the data transfer has occurred register N7:20 now holds a duplicate of the information that is in register N7:30. The previously existing data stored in register N7:20 have been replaced with those of N7:30. This process is referred to as ***writing over the existing data.***

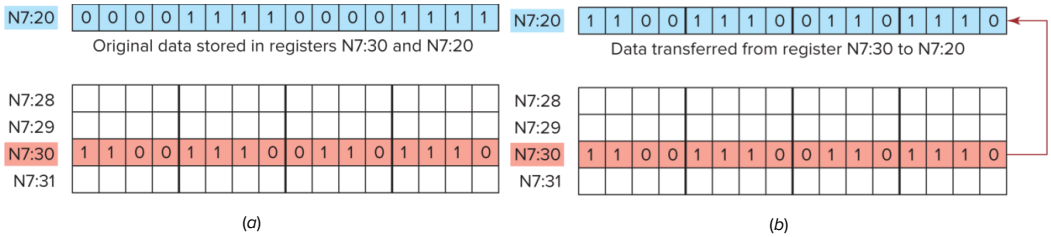


Figure 10-2 Data transfer concept.

Data transfer instructions can address almost any location in the memory. Prestored values can be automatically retrieved and placed in any new location. That location may be the preset register for a timer or counter or even an output register that controls a seven-segment display.

The MOV instruction is used to copy the value in one register or word to another. This instruction copies data from a *source* register to a *destination* register. **Figure 10-3** shows an example of the MOV instruction. The operation of the program can be summarized as follows:

- When the rung is true, input switch *A* closed, the value stored at the source address, N7:30, is copied into the destination address, N7:20.
- When the rung goes false, input switch *A* opened, the destination address will retain the value unless it is changed elsewhere in the program.
- The source value remains unchanged and no data conversion occurs.
- The instruction may be programmed with input conditions preceding it, or it may be programmed unconditionally.

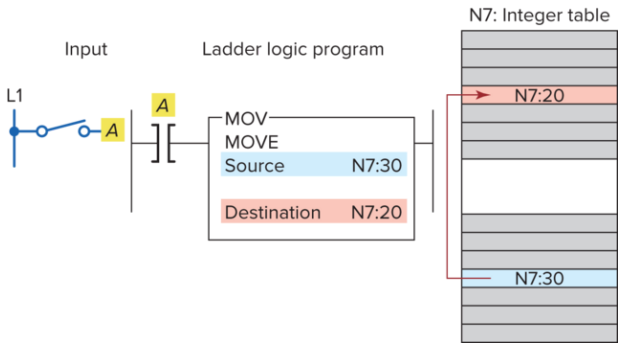


Figure 10-3 SLC 500 block-formatted move instruction.

A **Masked Move (MVM)** instruction functions like a standard move (MOV) instruction, except that a masked move allows data to be filtered out that is not required to be moved. The mask parameter specified in the instruction block is what performs this filtering process. This mask parameter can be either a word address location or a numerical constant. The operation of the mask parameter can be summarized as follows:

- The pattern of characters in the mask determines which source bits will be passed through to the destination address.
- The bits in the mask that are set to zero (0) do not pass data.
- Only the bits in the mask that are set to one (1) will pass the source data through to the destination.
- Bits in the destination are not affected when the corresponding bits in the mask are zero.
- The MVM instruction is used to copy the desired part of a 16-bit word by masking the rest of the value.

Figure 10-4 shows an example of a **mask move (MVM)** instruction. This instruction transfers data through the mask from the source address, B3:0, to the destination address, B3:4. The operation of the program can be summarized as follows:

- The mask may be entered as an address or in hexadecimal format, and its value will be displayed in hexadecimal.
- Where there is a 1 in the mask, data will pass from the source to the destination.
- Where there is a 0 in the mask, data in the destination will remain in their last state.
- Status in bits 4–7 are unchanged due to zeroes in the mask (remained in their last state).
- Status in bits 0–3 and 8–15 were copied from the source to destination when the MVM instruction went true.
- The mask must be the same word size as the source and destination.

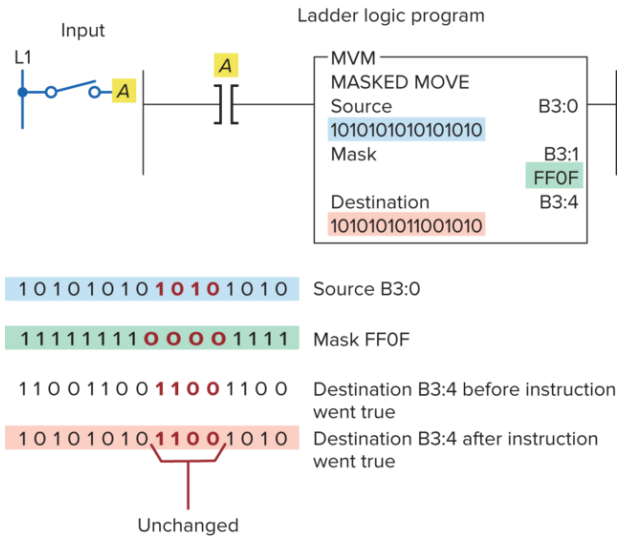


Figure 10-4 Masked move (MVM) instruction.

The program of **Figure 10-5** illustrates how the move (MOV) instruction can be used to create variable preset timer values. A two-position selector switch is operated to select one of two preset timer values. Operation of the program can be summarized as follows:

- When the selector switch is in the open 10 s position, rung 2 has logic continuity and rung 3 does not. Page 196
- As a result, the value 10 stored at the source address, N7:1, is copied into the destination address, T4:1.PRE.
- Therefore, the preset value of timer T4:1 will change from 0 to 10.
- When pushbutton PB1 is closed, there will be a 10 s delay period before the pilot light is energized.
- When the selector switch is in the closed 5 s position, rung 3 has logic continuity and rung 2 does not.
- As a result, the value 5 stored at the source address, N7:2, is copied into the destination address, T4:1.PRE.
- Closing pushbutton PB1 will now result in a 5 s time-delay period before the pilot light is energized.

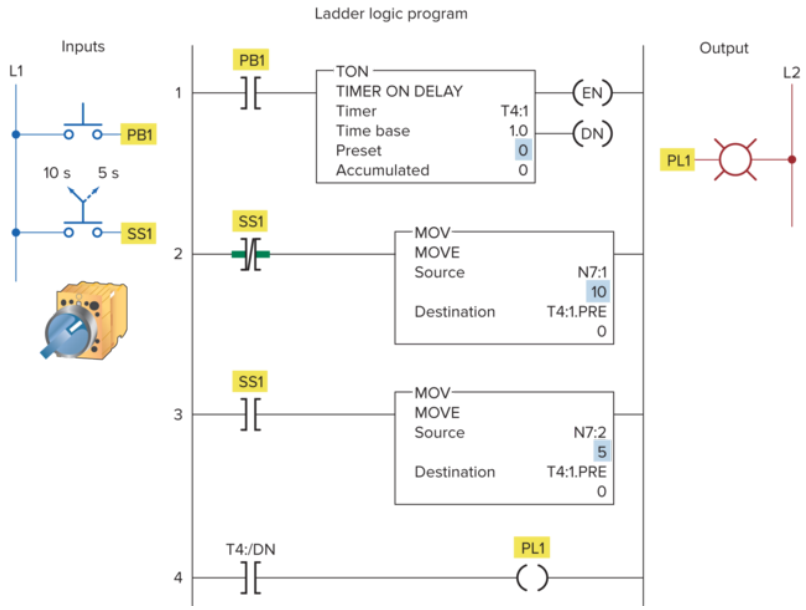


Figure 10-5 Move instruction used to change the preset time of a timer.

The program of **Figure 10-6** illustrates how the move (MOV) instruction can be used to create variable preset counter values. The operation of the program can be summarized as follows:

- Limit switch LSI is programmed to the input of up-counter C5:1 and counts the number of parts coming off a conveyor line onto a storage rack.
- Three different types of products are run on this line.
- The storage rack has room for only 300 boxes of product *A* or 175 boxes of product *B* or 50 boxes of product *C*.
- Three momentary switches are used to select the desired preset counter value depending on the product line (*A, B,* or *C*) being manufactured.
- A reset button is provided to reset the accumulated count to 0.
- A pilot lamp is switched on to indicate when the storage rack is full.
- The program has been constructed so that normally only one of the three switches will be closed at any one time. If more than one of the preset counter switches is closed, the *last* value is selected.

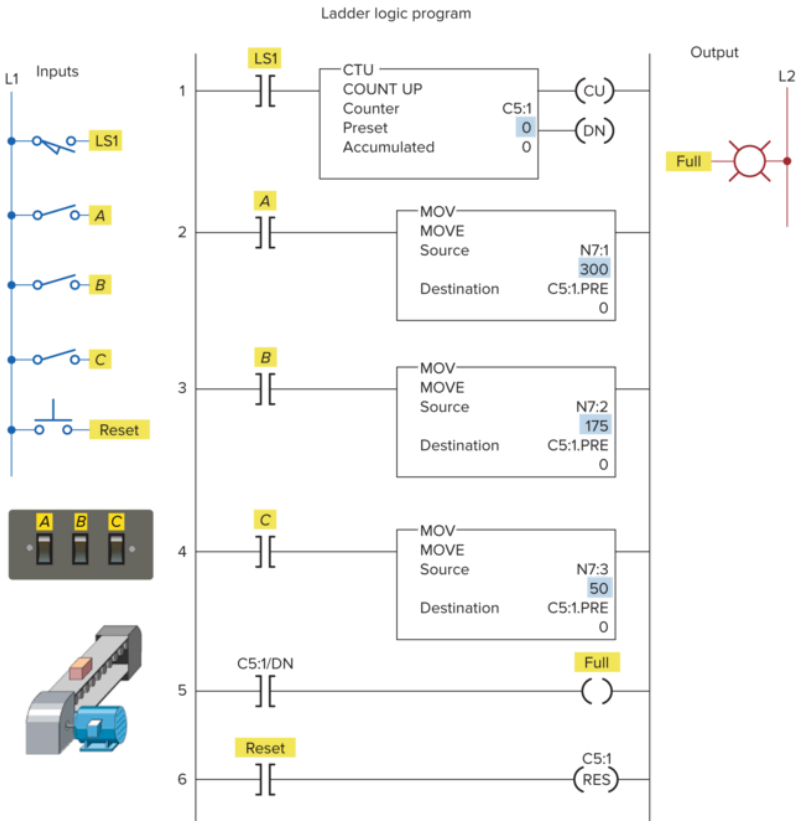


Figure 10-6 Move instruction used to change the preset count of a counter.

A ***file*** is a group of related consecutive words in the data table that have a defined start and end and are used to store information. For example, a batch process program may contain several separate recipes in different files that can be selected by an operator.

Page 197

In some instances it may be necessary to shift complete files from one location to another within the programmable controller memory. Such data shifts are termed ***file-to-file shifts.*** File-to-file shifts are used when the data in one file represent a set of conditions that must interact with the programmable controller program several times and, therefore, must remain *intact* after each operation. Because the data within this file must also be changed by the program action, a second file is used to handle the data changes, and the information within that file is allowed to be altered by the program. The data in the first file, however, remain constant and therefore can be used many times. Other types of data manipulation used with file instructions include word-to-file and file-to-word moves, as illustrated in **Figure 10-7**.

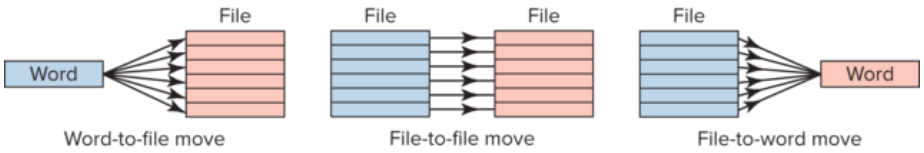


Figure 10-7 Moving data using file instructions.

Files allow large amounts of data to be scanned quickly and are useful in programs requiring the transfer, comparison, or conversion of data. Most PLC manufacturers display file instructions in block format on the programming terminal screen. **Figure 10-8** compares the RSLogix 500 controller word and file addressing. The addressing formats can be summarized as follows:

- The address that defines the beginning of a file or group of words starts with the pound sign #. Page 198
- The # prefix is omitted in a single word or element address.
- Address N7:30 is a word address that represents a single word: word number 30 in integer file 7.
- Address #N7:30 represents the starting address of a group of consecutive words in integer file 7. The length is eight words, which is determined by the instruction where the file address is used.

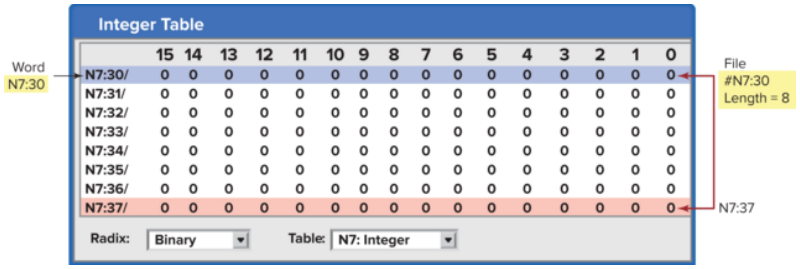


Figure 10-8 RSLogix 500 word and file address.

The ***file arithmetic and logic (FAL)*** instruction is used to copy data from one file to another and to do file math and file logic. This instruction is available on RSLogix 5000 platforms. An example of the FAL instruction is shown in **Figure 10-9**.

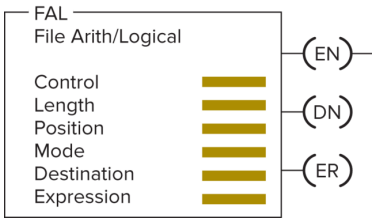


Figure 10-9 File Arithmetic/Logical (FAL) instruction.

The basic operation of the FAL instruction is similar in all functions and requires the following parameters and addresses to be entered in the instruction:

Control

- Is the first entry and the address of the control structure in the control area (R) of processor memory.
- The processor uses this information to run the instruction.
- The default file for the control file is data file 6.
- The control element for the FAL instruction must be unique for that instruction and may not be

used to control any other instruction.

- The control element is made up of three words.
- The control word uses four control bits: bit 15 (enable bit), bit 13 (done bit), bit 11 (error bit), and bit 10 (unload bit).
- For RSLogix 5000 the control address would be a tag such as control_1 with a data type of control.

Length

- Is the second entry and represents the file length.
- This entry will be in words, except for the floating-point file, for which the length is in elements. (A floating-point element consists of two words.)
- The maximum length possible is 1000 elements. Enter any decimal number from 1 to 1000.
- For RSLogix 5000 the number would be a double integer (DINT).

Position

- Is the third entry and represents the current location in the data block that the processor is accessing.
- It points to the word being operated on.
- The position starts with 0 and indexes to 1 less than the file length.
- You generally enter a 0 to start at the beginning of a file. You may also enter another position at which you want the FAL to start its operation.
- When the instruction resets, however, it will reset the position to 0.
- You can manipulate the position from the program.

Mode

- Is the fourth entry and represents the number of file elements operated on per program scan. There are three choices: all mode, numeric mode, and incremental mode.

All Mode

- For this mode you enter the letter *A*.
- In the all mode, the instruction will transfer the complete file of data in *one* scan.
- The enable (EN) bit will go true when the instruction goes true and will follow the rung condition.
- When all of the data have been transferred, the done (DN) bit will go true. This change will occur on the same scan during which the instruction goes true.
- If the instruction does not go to completion due to an error in the transfer of data (such as trying to store too large or too small a number for the data-table type), the instruction will stop at that point and set the error (ER) bit. The scan will continue, but the instruction will not continue until the error bit is reset.
- If the instruction goes to completion, the enable bit and the done bit will remain set until the instruction goes false, at which point the position, the enable bit, and the done bit will all be reset to 0.

Numeric Mode

- For this mode you enter a decimal number $(1-1000)$.
- In the numeric mode, the file operation is distributed over a number of program scans.
- The value you enter sets the number of elements to be transferred per scan.
- The numeric mode can decrease the time it takes to complete a program scan. Instead of waiting for the total file length to be transferred in one scan, the numeric mode breaks up the transfer of the file data into multiple scans, thereby cutting down on the instruction execution time per scan.

Incremental mode

- For this mode you enter the letter I.
- In the incremental mode, one element of data is operated on for every false-to-true transition of the instruction.
- The first time the instruction sees a false-to-true transition and the position is at 0, the data in the first element of the file are operated on. The position will remain at 0 and the UL bit will be set. The EN bit will follow the instruction's condition.
- On the second false-to-true transition, the position will index to 1, and data in the second word of the file will be operated on.
- The UL bit controls whether the instruction will operate just on data in the current position, or whether it will index the position and then transfer data. If the UL bit is reset, the instruction—on a false-to-true transition of the instruction—will operate on the data in the current position and set the UL bit. If the UL bit is set, the instruction—on a false-to-true transition of the instruction—will index the position by 1 and operate on the data in their new position.

Destination

- Is the fifth entry and is the address at which the processor stores the result of the operation.
- The instruction converts to the data type specified by the destination address.
- It may be either a file address or an element address.

Expression

- Is the last entry and contains addresses, program constants, and operators that specify the source of data and the operations to be performed.
- The expression entered determines the function of the FAL instruction.
- The expression may consist of file addresses, element addresses, or a constant and may contain only one function because the FAL instruction may perform only one function.

Figure 10-10 shows an example of a file-to-file copy function using the FAL instruction. The operation of the program can be summarized as follows:

- When input *A* goes true, data from the expression file #N7:20 will be copied into the destination file #N7:50.
- The length of the two files is set by the value entered in the control element word R6:1.LEN.
- In this instruction, we have also used the ALL mode, which means all of the data will be transferred in the first scan in which the FAL instruction sees a false-to-true transition.

- The DN bit will also come on in that scan unless an error occurs in the transfer of data, in which case the ER bit will be set, the instruction will stop operation at that position, and then the scan will continue at the next instruction.

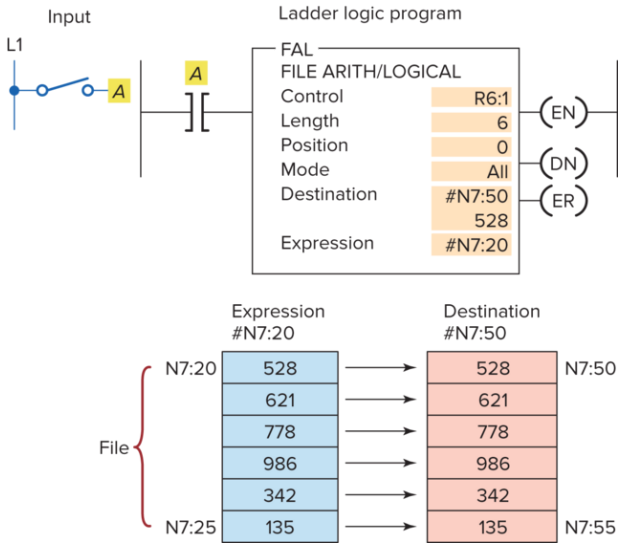


Figure 10-10 File-to-file copy function using the FAL instruction.

Figure 10-11 shows an example of a file-to-word copy function using the FAL instruction. The operation of the program can be summarized as follows:

- With each false-to-true rung transition of input *A,* the processor reads one word of integer file N29.
- The processor starts reading at word 0, and writes the image into word 5 of integer file N29.
- The instruction writes over any data in the destination.

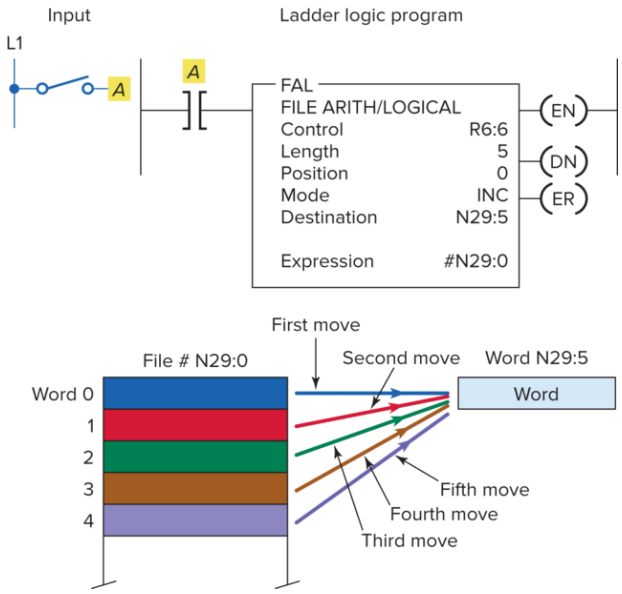


Figure 10-11 File-to-word copy function using the FAL instruction.

Figure 10-12 shows an example of a word-to-file copy function using the FAL instruction. It is similar to the file-to-word copy function except that the instruction copies data from a word address into a file. The operation of the program can be summarized as follows:

- The expression is a word address (N7:100) and the destination is a file address (#N7:101).
- If we start with position 0, the data from N7:100 will be copied into N7:101 on the first false-to-true transition of input *A*.
- The second false-to-true transition of input *A* will copy the data from N7:100 into N7:102.
- On successive false-to-true transitions of the instruction, the data will be copied into the next position in the file until the end of the file, N7:106, is reached.

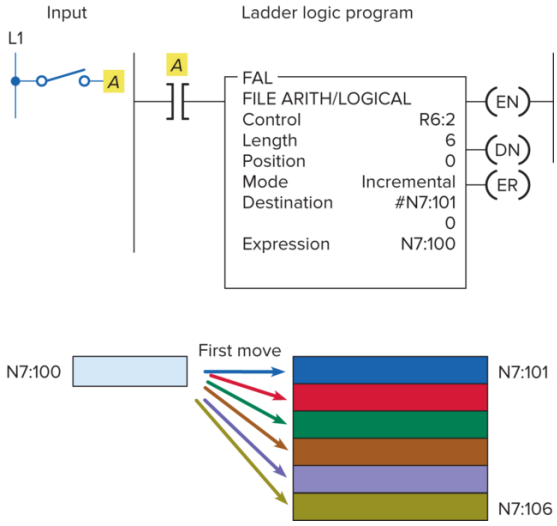


Figure 10-12 Word-to-file copy function using the FAL instruction.

The exceptions to the rule that file addresses must take consecutive words in the data table are in the *timer, counter,* and *control data* files for the FAL instruction. In these three data files, if you designate a file address, the FAL instruction will take every third word in that file and make a file of preset, accumulated, length, or position data within the corresponding file type. This might be done, for example, so that recipes storing values for timer presets can be moved into the timer presets, as illustrated in **Figure 10-13**.

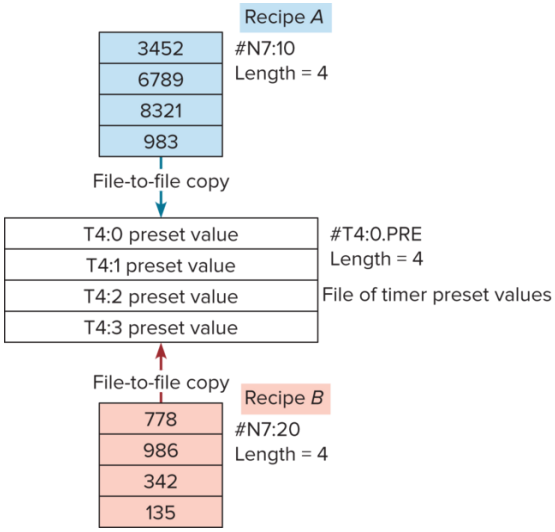


Figure 10-13 Copying recipes and storing values for timer presets.

The ***file copy (COP)*** instruction and the ***fill file (FLL)*** instruction are high-speed instructions that operate more quickly than the same operation with the FAL instruction. Unlike the FAL instruction, there is no control element to monitor or manipulate. Data conversion does not take place, so the source and destination should be the same file types. An example of the file COP instruction is shown in **Figure 10-14**. The operation of the program can be summarized as follows:

- Both the source and destination are file addresses.
- When input *A* goes true, the values in file N40 are copied to file N20.
- The instruction copies the entire file length for each scan during which the instruction is true.

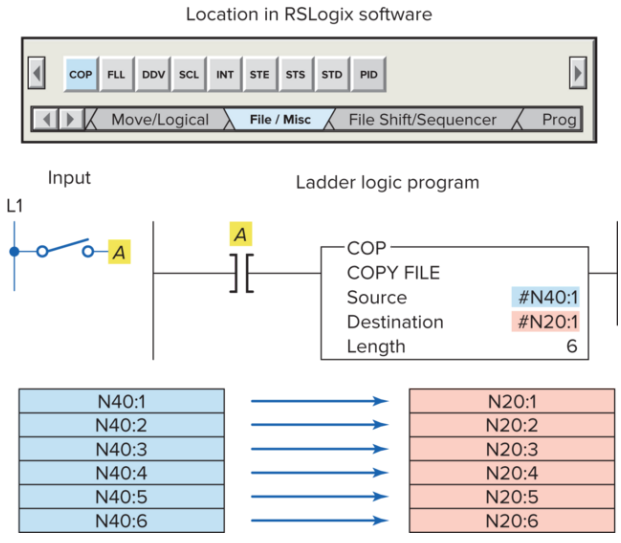


Figure 10-14 File copy (COP) instruction.

An example of the fill file (FLL) instruction is shown in **Figure 10-15**. It operates in a manner similar to the FAL instruction that performs the word-to-file copy in the ALL mode. The operation of the program can be summarized as follows:

- When input *A* goes true, the value in N15:5 is copied into N20:1 through N20:6.
- Because the instruction transfers to the end of the file, the file will be filled with the same data value in each word.

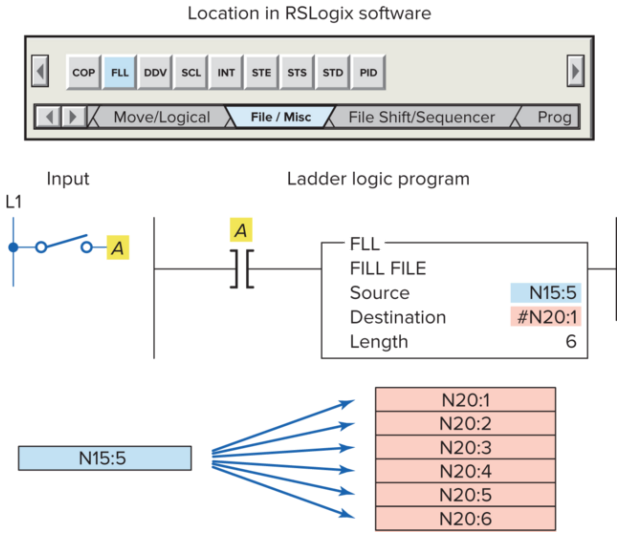


Figure 10-15 Fill file (FLL) instruction.

The FLL instruction is frequently used to zero all of the data in a file, as illustrated in the program of **Figure 10-16**. The operation of the program can be summarized as follows:

- Momentarily pressing pushbutton PB1 copies the contents of file #N10:0 into file #N12:0.
- Momentarily pressing pushbutton PB2 then clears file #N12:0.
- Note that 0 is entered for the source value.

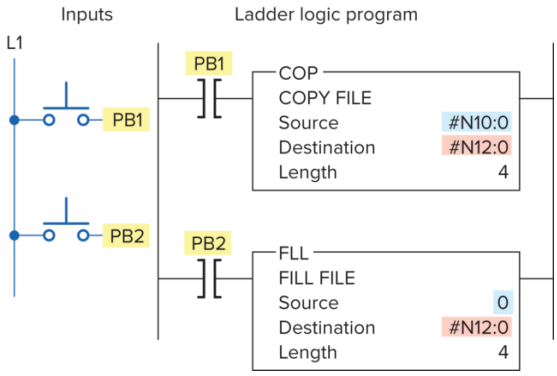


Figure 10-16 Using the FLL instruction to change all the data in a file to zero.

Figure 10-17 is an example of the copy (COP) instruction used as part of a PLC drink-manufacturing program. The operation of the program can be summarized as follows:

- A three-position selector switch is used for drink selection.
- Each selector switch position is electrically isolated so that only one input circuit can be energized at any one time.
- Each of the three selector switch inputs is wired to its corresponding input module address.
- Each recipe uses 5 memory words.

- Depending on the type of drink selected, the recipe is copied to the common working register #N7:50.
- The OSR instruction ensures that the copy instruction is executed only once on a false-to-true transition of the selector switch. In this way, if making the same recipe over a long period of time, the recipe needs to be copied only one time, not at every scan.

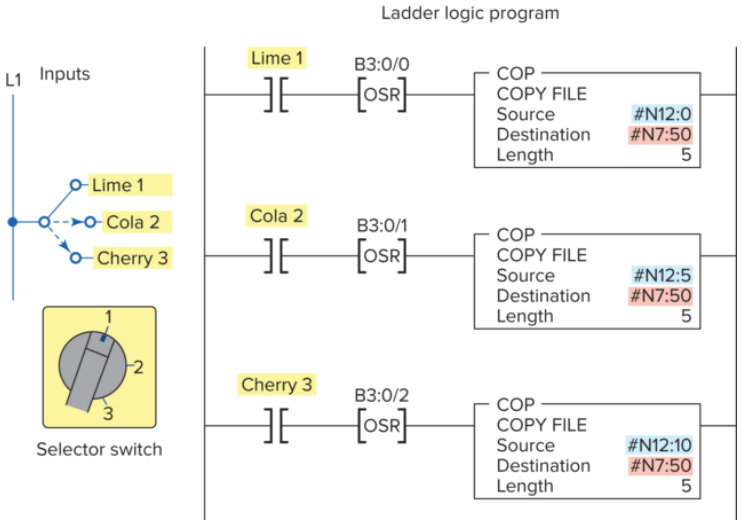


Figure 10-17 The copy (COP) instruction used as part of a PLC drink-manufacturing program.

OceanofPDF.com

10.3 Data Compare Instructions

Data transfer operations are all **output** instructions, whereas ***data compare*** instructions are ***input*** instructions. Data compare instructions are used to compare numerical values. These instructions compare the data stored in two or more words (or registers) and make decisions based on the program instructions. Numeric values in two words of memory can be compared for each of the basic data compare instructions shown in **Figure 10-18**, depending on the PLC.

Name	Symbol
Equal to	(=)
Not equal to	(≠)
Less than	(<)
Greater than	(>)
Less than or equal to	(≤)
Greater than or equal to	(≥)

Figure 10-18 Basic PLC data compare instructions.

Data comparison concepts have already been used with the timer and counter instructions. In both these instructions, an output was turned on or off when the accumulated value of the timer or counter equaled its preset value. What actually occurred was that the accumulated numeric data in one memory word was compared to the preset value of another memory word on each scan of the processor. When the processor saw that the accumulated value was equal to the preset value, it switched the output on or off.

Comparison instructions are used to test pairs of values to determine if a rung is true. **Figure 10-19** shows the Compare menu tab for the RSLogix 500 software. The compare instructions can be summarized as follows:

LIM (Limit test)—Tests whether one value is within the limit range of two other values.

MEQ (Masked Comparison for Equal)—Tests portions of two values to see whether they are equal. Compares 16-bit data of a source address to 16-bit data at a reference address through a mask.

EQU (Equal)—Tests whether the value of Source A is equal to the value of Source B

NEQ (Not Equal)—Tests whether the value of Source A is not equal to the value of Source B

LES (Less Than)—Tests whether the value of Source A is less than the value of Source B

GRT (Greater Than)—Tests whether the value of Source A is greater than the value of Source B

LEQ (Less Than or Equal)—Tests whether the value of Source A is less than or equal to the value of Source B

GEQ (Greater Than or Equal)—Tests whether the value of Source A is greater than or equal to the value of Source B

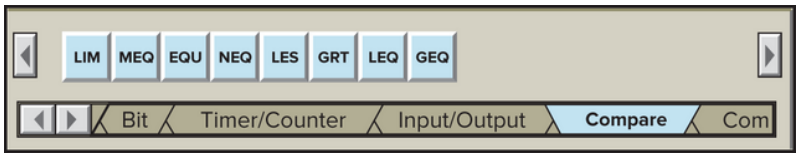


Figure 10-19 Compare menu tab.

The ***equal (EQU)*** instruction is an input instruction that compares source *A* to source *B:* when source *A* is equal to source *B,* the instruction is logically true; otherwise it is logically false. **Figure 10-20** shows an example of an EQU logic rung. The operation of the rung can be summarized as follows:

- When the accumulated value of counter T4:0 stored in source *A*'s address equals the value in source *B*'s address, N7:40, the instruction is true and the output is energized.
- Source *A* may be a word address or a floating-point address.
- Source *B* may be a word address, a floating-point address, or a constant value.
- With the equal instruction, the floating-point data is not recommended because of the exactness required. One of the other comparison instructions, such as the limit test, is preferred.

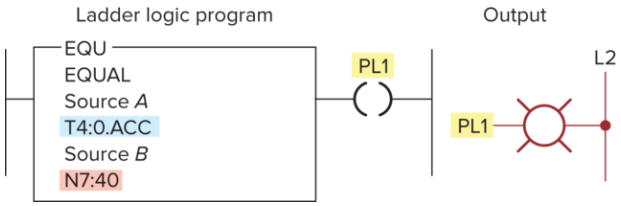


Figure 10-20 EQU logic rung.

The ***not equal (NEQ)*** instruction is an input instruction that compares source *A* to source *B:* when source *A* is not equal to source *B,* the instruction is logically true; otherwise it is logically false. **Figure 10-21** shows an example of an NEQ logic rung. The operation of the rung can be summarized as follows:

- When the value stored at source *A*'s address, N7:5, is not equal to 25, the output will be true; otherwise, the output will be false.
- The value stored at Source *A* is 30.
- The value stored at Source *B* is 25.
- Since the two values are not the same the output will be true or on.
- In all input-comparison instructions, Source *A* must be an address and Source *B* can be an address or a constant.

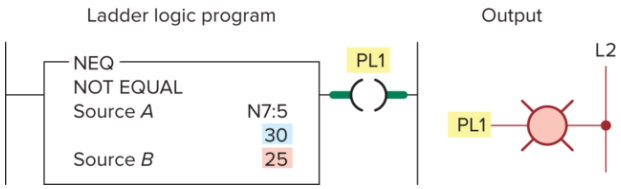


Figure 10-21 NEQ logic rung.

The ***greater than (GRT)*** instruction is an input instruction that compares source *A* to source *B:* when source *A* is greater than source *B,* the instruction is logically true; otherwise it is logically false.

Figure 10-22 shows an example of a GRT logic rung. The operation of the rung can be summarized as follows:

- The instruction is either true or false, depending on the values being compared.
- When the accumulated value of timer T4:10, stored at the address of source A, is greater than the constant 200 of source B, the output will be on; otherwise the output will be off.

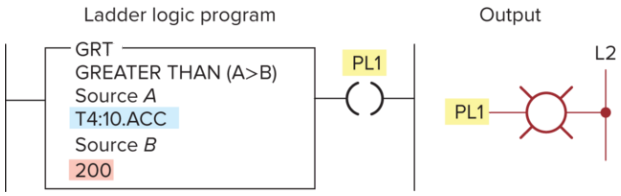


Figure 10-22 GRT logic rung.

The ***less than (LES)*** instruction is an input instruction that compares source *A* to source *B:* when source *A* is less than source *B,* the instruction is logically true; otherwise it is logically false.

Figure 10-23 shows an example of an LES logic rung. The operation of the rung can be summarized as follows:

- The instruction is either true or false, depending on the values being compared.
- When the accumulated value of counter C5:10, stored at the address of source *A,* is less than the constant 350 of source *B,* the output will be on; otherwise, it will be off.

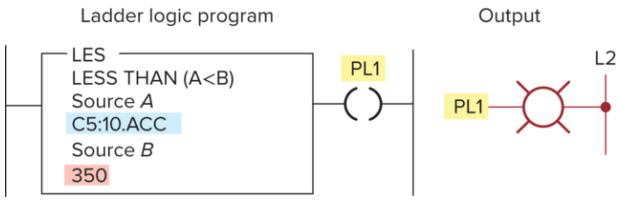


Figure 10-23 LES logic rung.

The ***greater than or equal (GEQ)*** instruction is an input instruction that compares source *A* to source *B:* when source *A* is greater than or equal to source *B,* the instruction is logically true; otherwise it is logically false. **Figure 10-24** shows an example of a GEQ logic rung. The operation of the rung can be summarized as follows:

- When the value stored at the address of source *A,* N7:55, is greater than or equal to the value stored at the address of source *B,* N7:12, the output will be true; otherwise, it will be false.
- The value stored at source *A* is 100.
- The value stored at source *B* is 23.

- Therefore the output will be true or on.

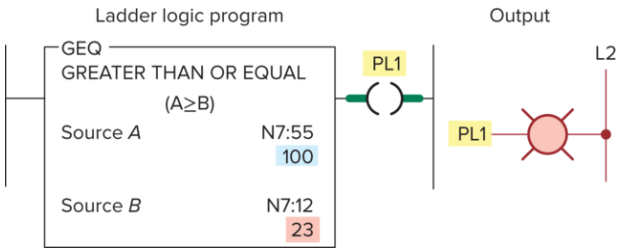


Figure 10-24 GEQ logic rung.

The ***less than or equal (LEQ)*** instruction is an input instruction that compares source *A* to source *B:* when source *A* is less than or equal to source *B,* the instruction is logically true; otherwise it is logically false. **Figure 10-25** shows an example of an LEQ logic rung. The operation of the rung can be summarized as follows:

- When the accumulated count of counter C5:1 is less than or equal to 457, the pilot light will turn on.
- The accumulated value of the counter is less than 457.
- Therefore the output will be true or on.

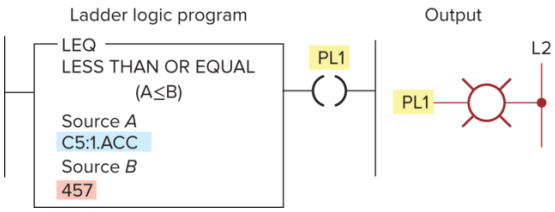


Figure 10-25 LEQ logic rung.

Figure 10-26 shows an example of an up-counter used in conjunction with the LES, EQU, and GRT compare instructions to trigger outputs based on different values of the counter's accumulated count. The operation of the program can be summarized as follows:

- A Less Than (LES) 10 comparison is made with the counter's accumulative value. As a result, the LESS Output will be energized anytime the accumulated count is 9 or less.
- An Equal (EQU) to 20 comparison is made with the counter's accumulative value. As a result, the EQU Output will be energized only when the accumulated count is 20.
- A Greater Than (GRT) 30 comparison is made with the counter's accumulative value. As a result, the GRT Output will be energized anytime the accumulated count is 31 or more.

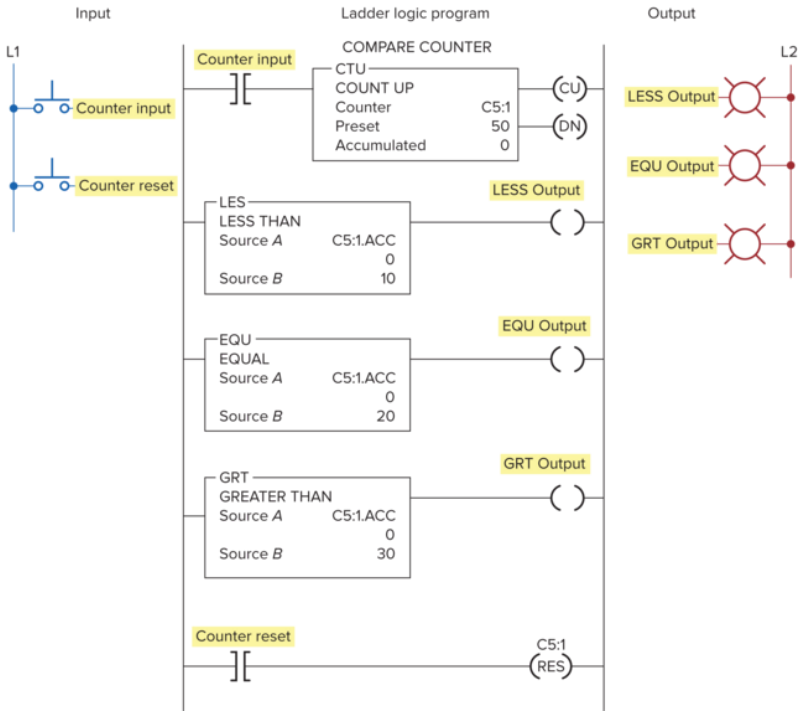


Figure 10-26 Triggering outputs based on the accumulated value of a counter.

The ***limit test (LIM)*** instruction is used to test whether values are within or outside the specified range. Applications in which the limit test instruction is used include allowing a process to operate as long as the temperature is within or outside a specified range.

Programming the LIM instruction consists of entering three parameters: low limit, test, and high limit. The limit test instruction functions in the following two ways:

- ***The instruction is true if***–The lower limit is equal to or less than the higher limit, and the test parameter value is equal to or inside the limits. Otherwise the instruction is false.
- ***The instruction is true if***–The lower limit has a value greater than the higher limit, and the instruction is equal to or outside the limits. Otherwise the instruction is false.

The limit test instruction is said to be circular because it can function in either of two ways.

Figure 10-27 shows an example of an LIM instruction where the low limit value is less than the high limit value. The operation of the logic rung can be summarized as follows:

- The high limit has a value of 50, and the low limit has a value of 25.
- Instruction is true for values of the test from 25 through 50.
- Instruction is false for test values less than 25 or greater than 50.
- Instruction is true because the test value is 48.

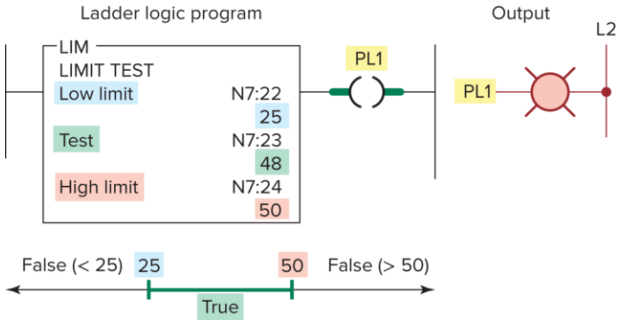


Figure 10-27 LIM instruction where the low limit value is less than the high limit value.

Figure 10-28 shows an example of an LIM instruction where the low limit value is greater than the high limit value. The operation of the logic rung can be summarized as follows:

- The high limit has a value of 50, and the low limit has a value of 100.
- Instruction is true for test values of 50 and less than 50 and for test values of 100 and greater than 100.
- Instruction is false for test values greater than 50 and less than 100.
- Instruction is true because the test value is 125.

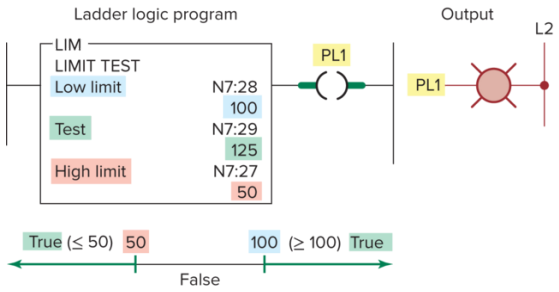


Figure 10-28 LIM instruction where the low limit value is greater than the high limit value.

The program of **Figure 10-29** shows a practical example of the ControlLogix program with the Limit Test (LIM) instruction. In this program, the LIM instruction will energize the Count_Within_Range output when the counter is within the range of 6 to 12 counts. Note that the range includes the values set as the low and high limits.

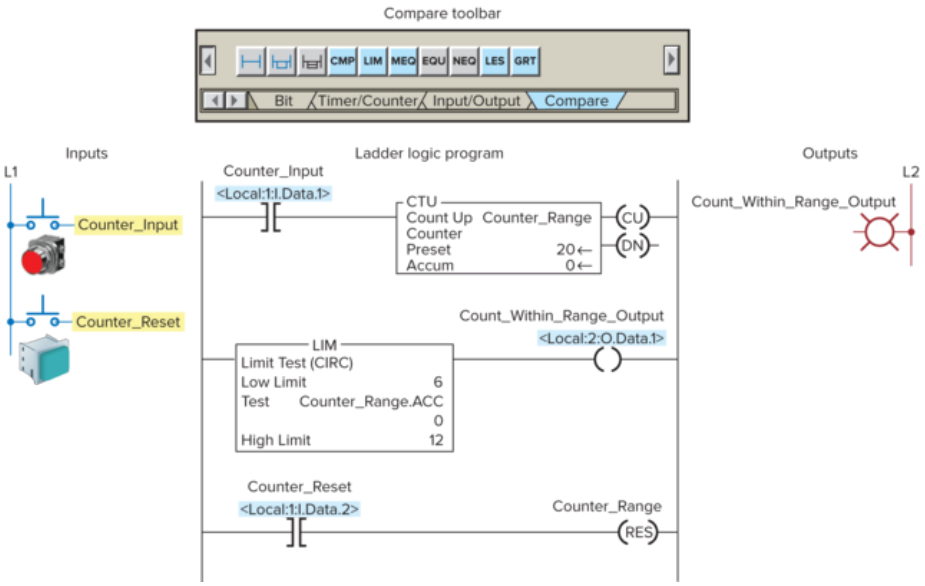


Figure 10-29 ControlLogix program with the Limit Test (LIM) instruction.

The *masked comparison for equal (MEQ)* instruction compares a value from a source address with data at a compare address and allows portions of the data to be masked. One application for the MEQ instruction is to compare the correct position of up to 16 limit switches when the source contains the limit switch address and the compare stores their desired states. The mask can block out the switches you don't want to compare (**Figure 10-30**).

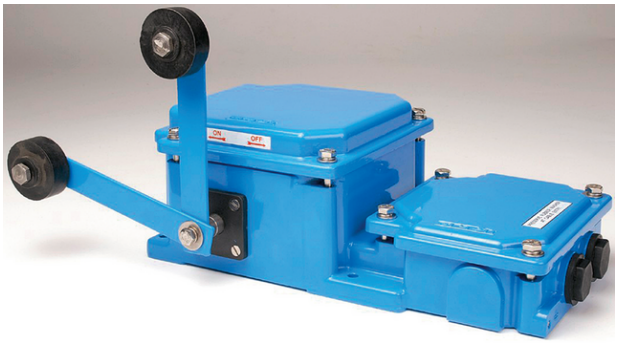


Figure 10-30 MEQ instruction can be used to monitor the state of limit switches.

Source: Courtesy Jayashree Electrodevices.

Figure 10-31 shows an example of an MEQ instruction. The operation of the logic rung can be summarized as follows:

- When the data at the source address match the data at the compare address bit-by-bit (less masked bits), the instruction is true.
- The instruction goes false as soon as it detects a mismatch.

- A mask passes data when the mask bits are set (1); a mask blocks data when the mask bits are reset (0).
- The mask must be the same element size (16 bits) as the source and compare addresses.
- You must set mask bits to 1 to compare data. Bits in the compare address that correspond to 0s in the mask are not compared.
- If you want the ladder program to change mask value, store the mask at a data address. Otherwise, enter a hexadecimal value for a constant mask value.

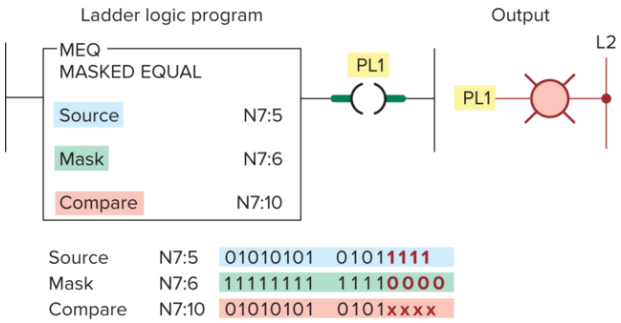


Figure 10-31 Masked comparison for equal (MEQ) logic rung.

OceanofPDF.com

10.4 Data Manipulation Programs

Data manipulation instructions give dimension and flexibility to the programming of control circuits. For example, consider the hardwired relay-operated, time-delay circuit in **Figure 10-32**. This circuit uses three electromechanical time-delay relays to control four solenoid valves. The operation of the hardwired circuit can be summarized as follows:

- When the momentary start pushbutton is pressed solenoid *A* is energized immediately.

Page 207

- Solenoid *B* is energized 5 s later than solenoid *A*.
- Solenoid *C* is energized 10 s later than solenoid *A*.
- Solenoid *D* is energized 15 s later than solenoid *A*.

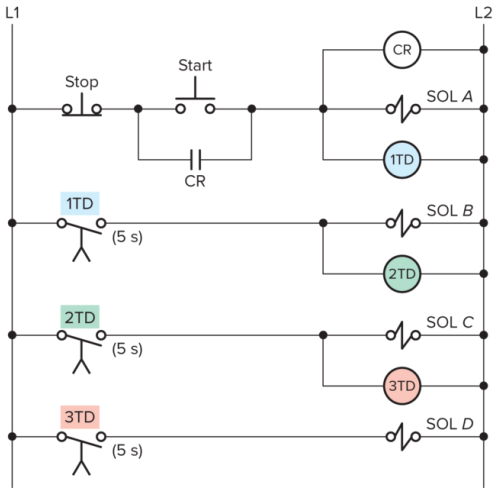


Figure 10-32 Three electromechanical time-delay relays used to control four solenoid valves.

The hardwired time-delay circuit could be implemented using a conventional PLC program and three internal timers. However, the same circuit can be programmed using only *one* internal timer along with data compare instructions. **Figure 10-33** shows the program required to implement the circuit using only one internal timer. The operation of the program can be summarized as follows:

- The momentary stop button is closed.
- When the momentary start button is pressed, SOL *A* output energizes immediately to switch on solenoid *A*.

Page 208

- SOL *A* examine-on contact becomes true to seal in output SOL *A* and to start on-delay timer T4:1 timing.
- The timer preset time is set to 15 seconds.
- Output SOL *B* will energize after a total time delay of 5 seconds, when the accumulated time becomes equal to and then greater than 5 seconds. This, in turn, will energize solenoid *B*.

- Output SOL *C* will energize after a total time delay of 10 seconds, when the accumulated time becomes equal to and then greater than 10 seconds. This, in turn, will energize solenoid *C*.
- Output SOL *D* will energize (through the timer done bit T4:1/DN) after a total time delay of 15 seconds to energize solenoid *D*.

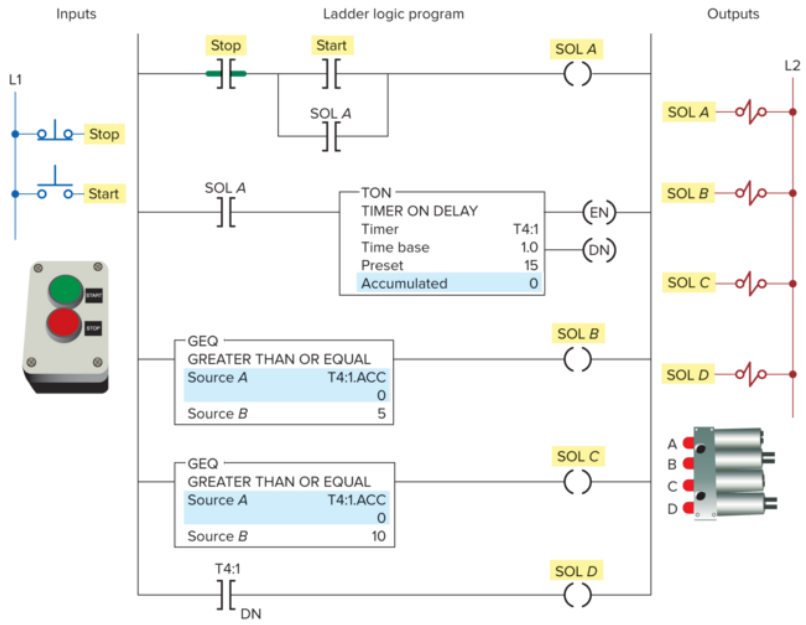


Figure 10-33 Controlling multiple loads using one timer and the GEQ instruction.

Figure 10-34 shows an application of an on-delay timer program implemented using the EQU instruction. The operation of the program can be summarized as follows:

- When the switch (S1) is closed, timer T4:1 will begin timing.
- Both EQU instructions' source *A*s are addressed to get the accumulated value from the timer while it is running.
- The EQU instruction of rung 2 has the value of 5 stored in source *B*.
- When the accumulated value of the timer reaches 5, the EQU instruction of rung 2 will become logic true for 1 second.
- As a result, the latch output will energize to switch the pilot light PL1 on.
- When the accumulated value of the timer reaches 15, the EQU instruction of rung 3 will be true for 1 second.
- As a result, the unlatch output will energize to switch the pilot light PL1 off.
- Therefore, when the switch is closed, the pilot light will come on after 5 seconds, stay on for 10 seconds, and then turn off.

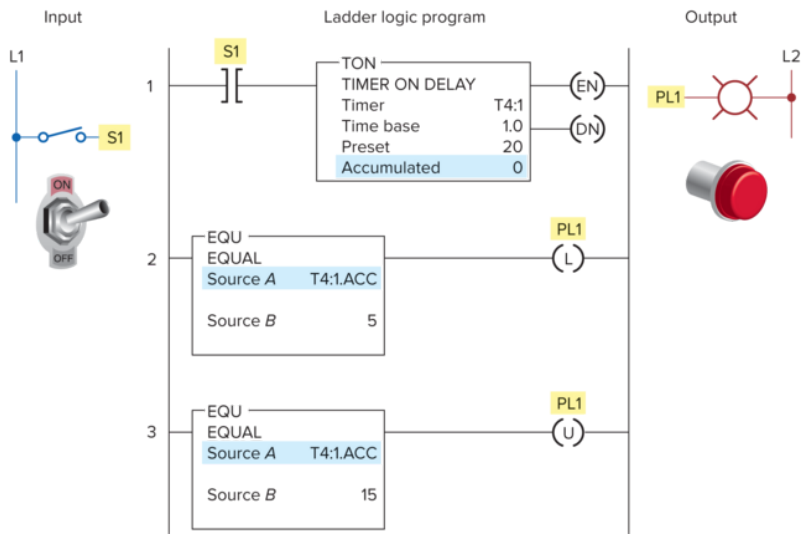


Figure 10-34 Timer program implemented using the EQU instruction.

Figure 10-35 shows an application of an up-counter program implemented using the LES instruction. The operation of the program can be summarized as follows:

- Up-counter C5:1 will increment by 1 for every false-to-true transition of the proximity sensor switch.
- Source *A* of the LES instruction is addressed to the accumulated value of the counter and source *B* has a constant value of 20.
- The LES instruction will be true as the long as the value contained in source *A* is *less than* that of source *B*.
- Therefore, output solenoid SOL will be energized when the accumulated value of the counter is between 0 and 19.
- When the counter's accumulated value reaches 20, the LES instruction will go false, de-energizing output solenoid SOL.
- When the counter's accumulated value reaches its preset value of 50, the counter reset will be energized through the counter done bit (C5:1/DN) to reset the accumulated count to 1.

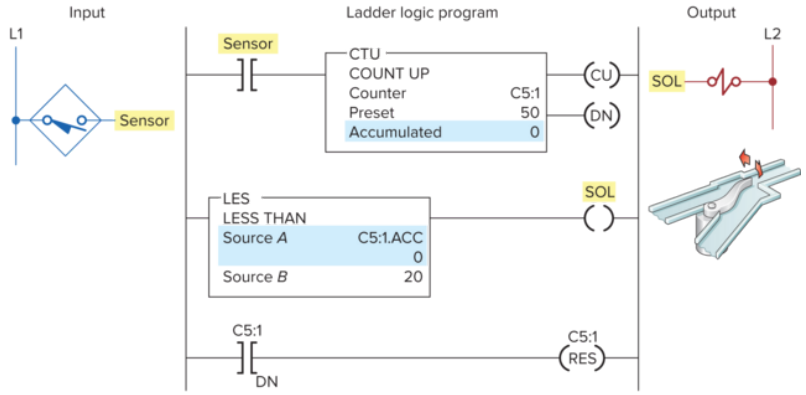


Figure 10-35 Counter program implemented using the LES instruction.

The use of comparison instructions is generally straightforward. However, one precaution involves the use of these instructions in PLC programs used to control the flow in vessel filling operations. This control scenario can be summarized as follows:

- The receiving vessel has its weight monitored continuously by the PLC program as it fills.
- When the weight reaches a preset value, the flow is cut off.
- While the vessel fills, the PLC performs a comparison between the vessel's current weight and a predetermined constant programmed in the processor.
- If the programmer uses only the equal instruction, problems may result.
- As the vessel fills, the comparison for equality will be false. At the instant the vessel weight reaches the desired preset value of the equal instruction, the instruction becomes true and the flow is stopped.
- However, should the supply system leak additional material into the vessel, the total weight of the material could rise *above* the preset value, causing the instruction to go false and the vessel to overfill.
- The simplest solution to this problem is to program the comparison instruction as a greater than or equal to instruction. This way, any excess material entering the vessel will not affect the filling operation.
- It may be necessary, however, to include additional programming to indicate a serious overfill condition.

OceanofPDF.com

10.5 Numerical Data I/O Interfaces

The expanding data manipulation processing capabilities of PLCs led to the development of I/O interfaces known as numerical data I/O interfaces. In general, numerical data I/O interfaces can be divided into two groups: those that provide interface to *multibit digital* devices and those that provide interface to *analog* devices.

The multibit digital devices are like the discrete I/O because processed signals are discrete (on/off). The difference is that, with the discrete I/O, only a *single* bit is required to read an input or control an output. Multibit interfaces allow a group of bits to be input or output *as a unit.* They can be used to accommodate devices that require BCD inputs or outputs.

The *thumbwheel switches (TWS),* shown in **Figure 10-36**, are typical BCD input devices. Each one of the four switches provides four binary digits at its output that correspond to the decimal number selected on the switch. The conversion from a single decimal digit to four binary digits is performed by the TWS device. The BCD input module allows the processor to accept the 4-bit digital codes and input their data into specific register or word locations in memory to be used by the control program. Data manipulation instructions can be used to access the data from the input module allowing a person to change set points, timer, or counter presets *externally* without modifying the control program.

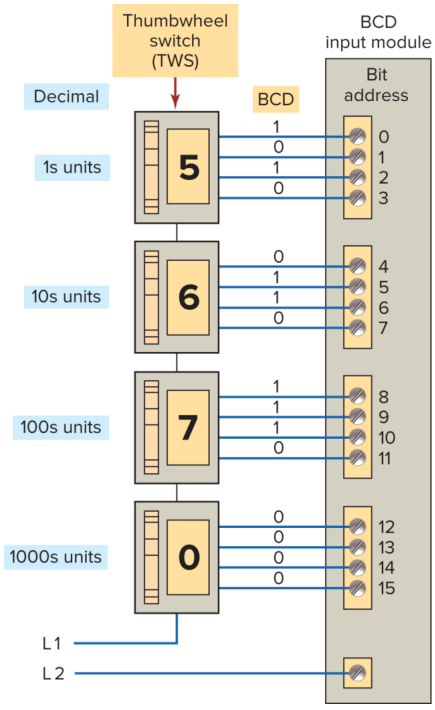


Figure 10-36 BCD input interface module connected to a thumbwheel switch.

The *seven-segment LED* display board, shown in **Figure 10-37**, is a typical Binary Coded Decimal (BCD) output device. It displays a decimal number that corresponds to the BCD value it receives at its input. Conversion of the four binary bits to a single decimal digit on the display is performed by the LED display device. The BCD output module is used to output data from a specific register or word location in memory. This type of output module enables a PLC to operate devices that require BCD coded signals.

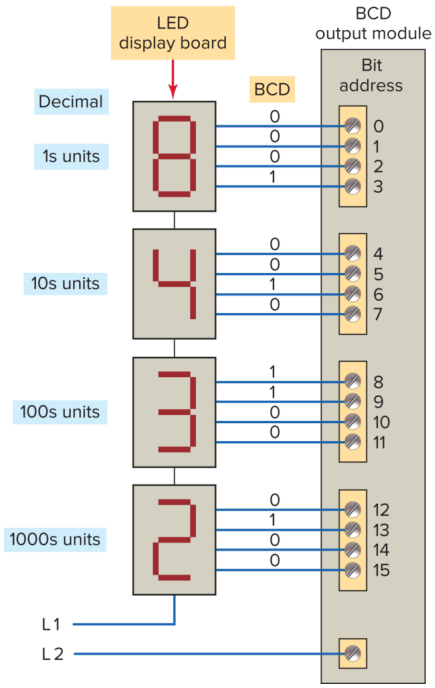


Figure 10-37 BCD output interface module connected to a seven-segment LED display board.

RSLogix 5000 PLCs do not require the use of special BCD input and output modules. Standard digital I/O modules may be used with BCD conversion instructions. The PLC does not recognize a BCD number. Instead, it views it as a simple binary number that has a different set of weight positions. For example:

- Assume 79 is displayed on the thumbwheels.
- The BCD value would be:

8	4	2	1	8	4	2	1	= 79
0	1	1	1	1	0	0	1	

- The PLC views it as:

64	32	16	8				1	= 121
0	1	1	1	1	0	0	1	

- The TOD (convert to BCD) instruction is used to convert a decimal value to a BCD value.
- The FRD (convert to integer) instruction is used to convert a BCD value to a decimal value.

Figure 10-38 shows a PLC program designed to monitor the setting of a thumbwheel switch and turn on a pilot light anytime a value of 79 is displayed. The operation of the program is summarized as follows:

- Data are directly exchanged between the TWS and LED display as both operate on BCD data.
- The FRD (From BCD) instruction is used to take the BCD thumbwheel source value and convert it to its decimal equivalent.
- Setting of the thumbwheel switch is compared to the reference number 79 entered in source B of the EQU instruction.
- Pilot light output PL is energized any time the input switch S1 is true (closed) and the value of the thumbwheel switch is equal to 79.

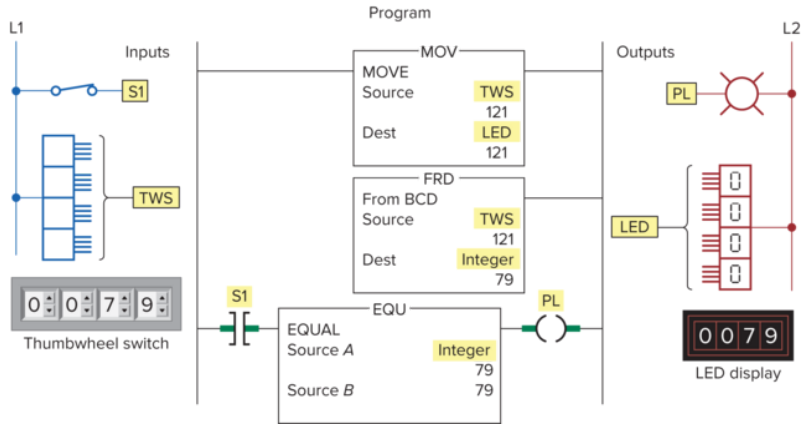


Figure 10-38 Monitoring the setting of a thumbwheel switch.

Analog modules convert analog signals to digital signals (input) or digital signals to analog values (output). The digital signal is a sampled signal, obtained by sampling the analog signal at discrete points in time. These points are usually evenly spaced in time, with the time between being referred to as the sampling interval. The sampling rate is the frequency expressed in Hertz (Hz) at which the ADC samples the analog signal. Analog I/O will allow monitoring and control of analog voltages and currents (e.g., 4–20 mA or 0–10 V). Bit resolution refers to the quantity of analog signal that is represented by 1 bit in the controller. The more binary digits the digital representation uses, the better the resolution will be. Figure 10-39 illustrates how an analog input interface operates. The operation of this input module can be summarized as follows:

- The analog input module contains the circuitry necessary to accept analog voltage or current signals from field devices.
- The input signal is converted from an analog to a digital value by an analog-to-digital (A/D) converter circuit.
- The conversion value, which is proportional to the analog signal, is passed through the controller's data bus and stored in a specific register or word location in memory for later use by the control program.

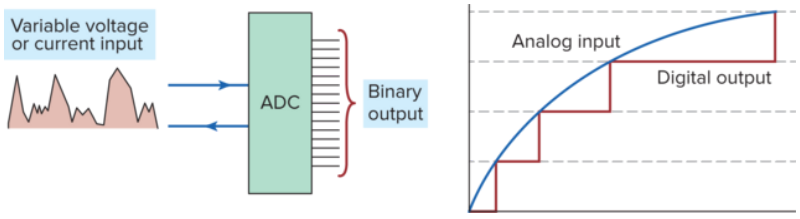


Figure 10-39 Analog input module analog-to-digital converter.

An analog output interface module receives numerical data from the processor; these data are then translated into a proportional voltage or current to control an analog field device. **Figure 10-40** illustrates how an analog output interface operates. The operation of this output module can be summarized as follows:

- The function of the analog output module is to accept a range of numeric values output from the PLC program and to produce a varying current or voltage signal required to control a connected analog output device.
- Data from a specific register or word location in the CPU memory are passed through the controller's data bus to the digital-to-analog (D/A) converter.
- The analog output from the D/A converter is then used to control the analog output device.
- The level of the analog signal output is based on the digital value of the data word supplied by the CPU and manipulated by the control program.
- These output interfaces normally require an external power supply that meets certain current and voltage requirements.

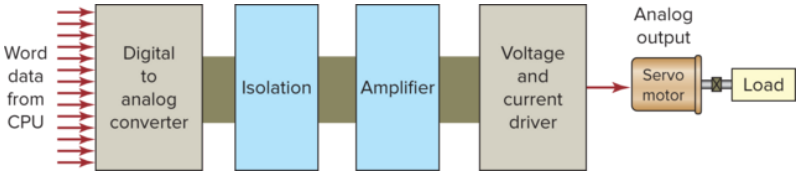


Figure 10-40 Analog output interface module.

OceanofPDF.com

10.6 Closed-Loop Control

In open-loop control, no feedback loop is employed and system variations which cause the output to deviate from the desired value are not detected or corrected. A closed-loop system utilizes feedback to measure the actual system operating parameter being controlled such as temperature, pressure, flow, level, or speed. This feedback signal is sent back to the PLC where it is compared with the desired system set-point. The controller develops an error signal that initiates corrective action and drives the final output device to the desired value.

PLC *set-point control* in its simplest form compares an input value, such as analog or thumbwheel inputs, to a set-point value. A discrete output signal is provided if the input value is less than, equal to, or greater than the set-point value. The temperature control program of **Figure 10-41** is one example of set-point control. In this application, a PLC is to provide for simple off/on control of the electric heating elements of an oven. The operation of the program can be summarized as follows:

- Oven is to maintain an average set-point temperature of $600^{\circ}\mathrm{F}$ with a variation of about 1 percent between the off and on cycles.
- The electric heaters are turned on when the temperature of the oven is $597^{\circ}\mathrm{F}$ or less and will stay on until the temperature rises to $603^{\circ}\mathrm{F}$ or more.
- The electric heaters stay off until the temperature drops to $597^{\circ}\mathrm{F}$, at which time the cycle repeats itself.
- Whenever the less than or equal (LEQ) instruction is true, a low-temperature condition exists and the program switches on the heater.
- Whenever the greater than or equal (GEQ) instruction is true, a high-temperature condition exists and the program switches off the heater.

- For the program as shown the temperature is $595^{\circ}\mathrm{F}$ so the LEQ instruction and B3:0/1 will both be true and the heater output will be switched on and sealed-in through the heater examine-on instruction.
- Once the temperature increases to $598^{\circ}\mathrm{F}$ the LEQ instruction goes false but the heater output remains on until the temperature rises to $603^{\circ}\mathrm{F}$.
- At the $603^{\circ}\mathrm{F}$ point the GEQ instruction and B3:0/2 will both be true and the heater will be switched off.

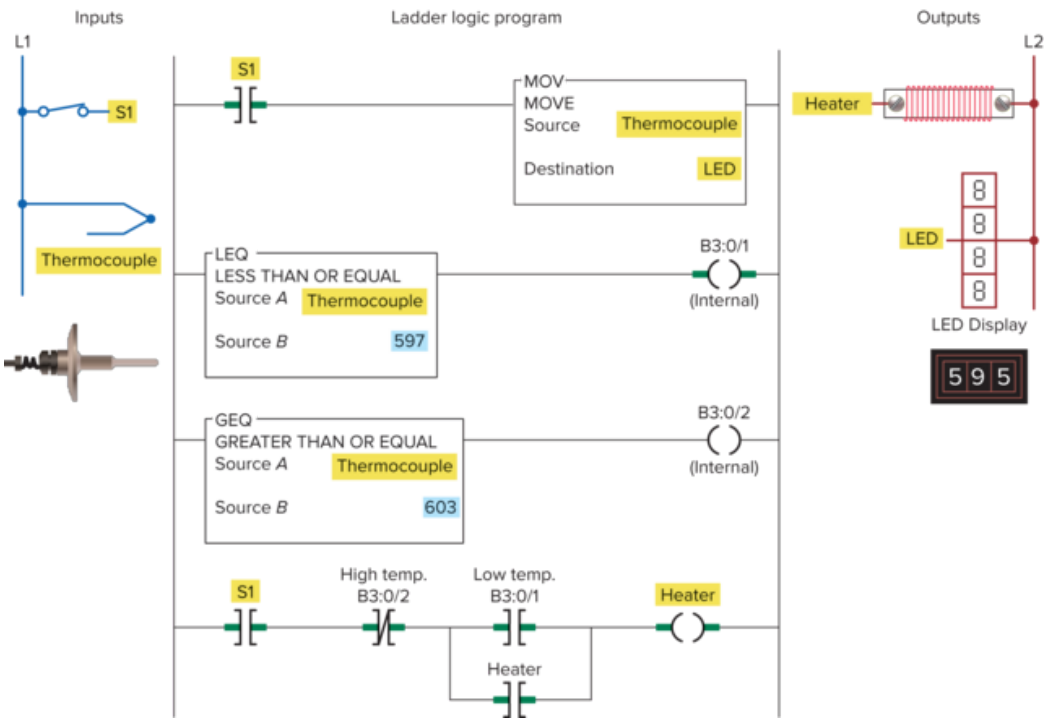


Figure 10-41 Set-point control program.

Several set-point control schemes can be performed by different PLC models. These include on/off control, proportional (P) control, proportional-integral (PI) control, and proportional-integral-derivative (PID) control. Each involves the use of some form of closed-loop control to maintain a process characteristic, such as a temperature, pressure, flow, or level, at a desired value. When a control system is designed such that it receives operating information from the machine and makes adjustments to the machine based on this operating information, the system is said to be a closed-loop system.

The block diagram of a closed-loop control system is shown in **Figure 10-42**. A measurement is made of the variable to be controlled. This measurement is then compared to a reference point, or set-point. If a difference (error) exists between the actual and desired levels, the PLC control program will take the necessary corrective action. Adjustments are made continuously by the PLC until the difference between the desired and actual output is as small as is practical.

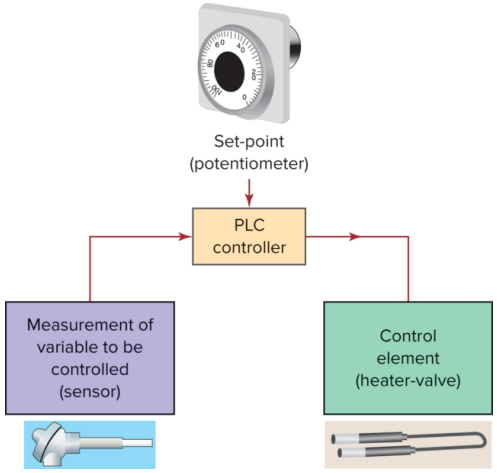


Figure 10-42 Closed-loop control system.

With on/off PLC control (also known as *two-position* and *bang-bang control*), the output or final control element is either on or off—one for the occasion when the value of the measured variable is above the set-point and the other for the occasion when the value is below the set-point. The controller will never keep the final control element in an intermediate position. Most residential thermostats are on/off type controllers.

On/off control is inexpensive but not accurate enough for most process and machine control applications. On/off control almost always means overshoot and resultant system cycling. For this reason a *deadband* usually exists around the set-point. The deadband or hysteresis of the control loop is the difference between the on and off operating points.

Proportional controls are designed to eliminate the hunting or cycling associated with on/off control. They allow the final control element to take intermediate positions between on and off. This permits *analog control* of the final control element to vary the amount of energy to the process, depending on how much the value of the measured variable has shifted from the desired value.

The process illustrated in **Figure 10-43** is an example of a proportional control process. The PLC analog output module controls the amount of fluid placed in the holding tank by adjusting the percentage of valve opening. The valve is initially open 100 percent. As the fluid level in the tank approaches the preset point, the processor modifies the output to degrade closing the valve by different percentages, adjusting the valve to maintain a set-point.

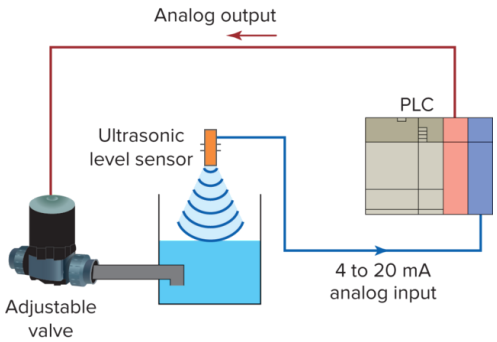


Figure 10-43 Proportional control process.

Proportional-integral-derivative (PID) control is the most sophisticated and widely used type of process control. PID operations are more complex and are mathematically based. PID controllers produce outputs that depend on the *magnitude, duration,* and *rate of change* of the system error signal. Sudden system disturbances are met with an aggressive attempt to correct the condition. A PID controller can reduce the system error to 0 faster than any other controller.

A typical PID control loop is illustrated in **Figure 10-44**. The loop measures the process, compares it to a set-point, and then manipulates the output in the direction which should move the process toward the set-point. The terminology used in conjunction with a PID loop can be summarized as follows:

- Operating information that the controller receives from the machine is called the *process variable (PV)* or *feedback*.
- Input from the operator that tells the controller the desired operating point is called the *set-point (SP)*.
- When operating, the controller determines whether the machine needs adjustment by comparing (by subtraction) the set-point and the process variable to produce a difference (the difference is called the *error*).
- Output from the loop is called the *control variable (CV),* which is connected to the controlling part of the process.
- The PID loop takes appropriate action to modify the process operating point until the control variable and the set-point are very nearly equal.

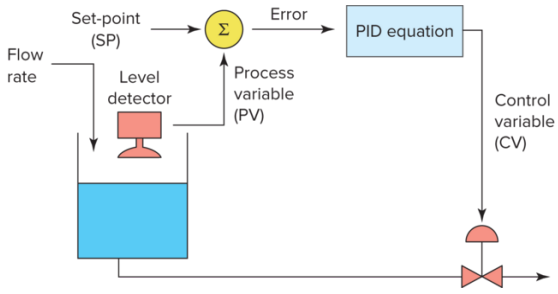


Figure 10-44 Typical PID control loop.

Programmable controllers are either equipped with PID I/O modules that produce PID control or have sufficient mathematical functions of their own to allow PID control to be carried out.

Figure 10-45 shows an RSLogix 500 PID instruction with typical addresses for the parameters entered. The PID instruction normally controls a closed loop using inputs from an analog input module and provides an output to an analog output module. Explanation of the PID instruction parameters can be summarized as follows:

- Control Block is the file that stores the data required to operate the instruction.
- Process Variable (PV) is an element address that stores the process input value.

- Control Variable (CV) is an element address that stores the output of the PID instruction.

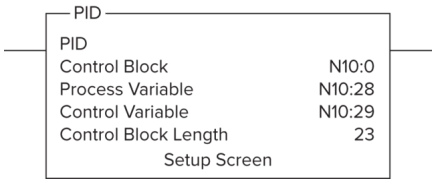


Figure 10-45 RSLogix 500 PID instruction.

OceanofPDF.com

CHAPTER 10 REVIEW QUESTIONS

1. In general, what do data manipulation instructions allow the PLC to do?
2. Explain the difference between a register or word and a table or file.
3. Into what two broad categories can data manipulation instructions be placed?
4. What takes place with regard to a data transfer instruction?
5. The MOV instruction is to be used to copy the information stored in word N7:20 to N7:35. What address is entered into the source and the destination?
6. What is the purpose of the mask word in the MVM instruction?
7. List three types of data shifts used with file instructions.
8. List the six parameters that must be entered into the file arithmetic and logic (FAL) instruction.
9. Assume the ALL mode has been entered as part of a FAL instruction. How will this affect the transfer of data?
10. What is the advantage of using the file copy (COP) or fill file (FLL) instruction rather than the FAL instruction for the transfer of data?
11. What are data compare instructions used for?
12. Name the six different types of data compare instructions.
13. Explain what each of the logic rungs in **Figure 10-46** is instructing the processor to do.

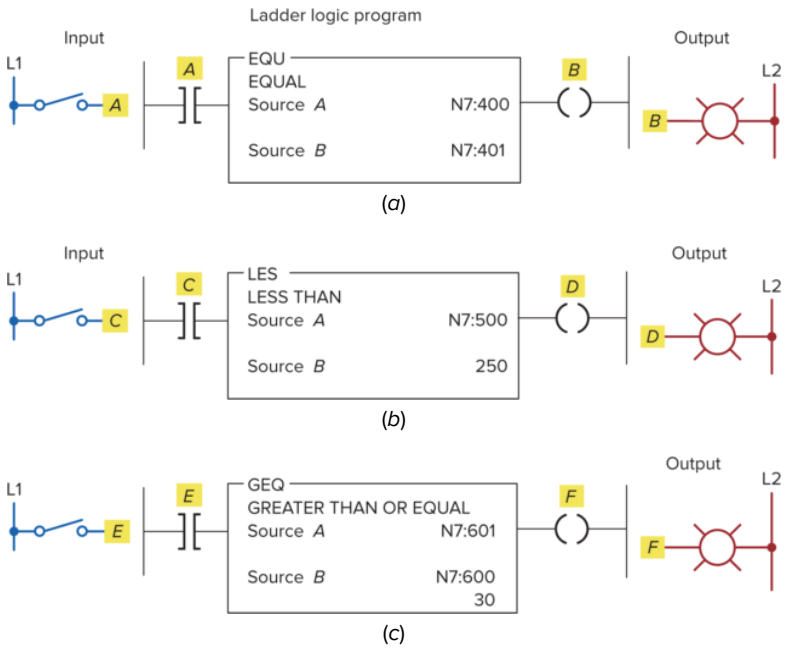


Figure 10-46 Logic rungs for Question 13.

14. What does the limit test (LIM) instruction test values for?

15. How are multibit I/O interfaces different from the discrete type?

16. Assume that a thumbwheel switch is set for the decimal number 3286.

a. What is the equivalent BCD value for this setting?

b. What is the equivalent binary value for this setting?

17. Assume that a thermocouple is connected to an analog input module. Explain how the temperature of the thermocouple is communicated to the processor.

18. Outline the process by which an analog output module operates the field device connected to it.

19. Compare the operation of open-loop and closed-loop PLC systems.

20. Outline the control process involved with simple PLC set-point control.

21. Compare the operation of the final control element in on/off and proportional control systems.

22. Explain the meaning of the following terms as they apply to a PID control:

a. Process variable

b. Set-point

c. Error

d. Control variable

OceanofPDF.com



CHAPTER 10 PROBLEMS

1. Study the data transfer program of **Figure 10-47** and answer the following questions:

a. When S1 is open, what decimal number will be stored in integer word address N7:13 of the MOV instruction?

b. When S1 is on, what decimal number will be stored in integer word address N7:112 of the MOV instruction?

c. When S1 is on, what decimal number will appear in the LED display?

d. What is required for the decimal number 216 to appear in the LED display?

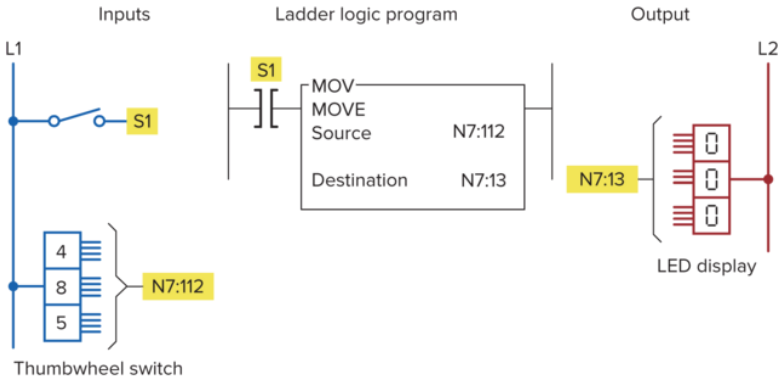


Figure 10-47 Program for Problem 1.

2. Study the data transfer counter program of **Figure 10-48** and answer the following questions:

a. What determines the preset value of the counter?

b. Outline the steps to follow to operate the program so that the PL1 output is energized after 25 off-to-on transitions of the count PB input.

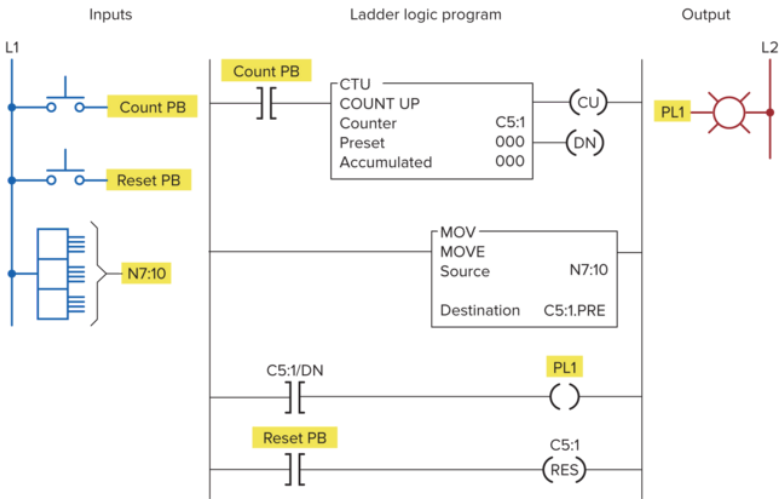


Figure 10-48 Program for Problem 2.

3. Construct a nonretentive timer program that will turn on a pilot light after a time-delay period. Use a thumbwheel switch to vary the preset time-delay value of the timer.

4. Study the data compare program of **Figure 10-49** and answer the following questions:

   a. Will the pilot light PL1 come on whenever switch S1 is closed? Why?

   b. Must switch S1 be closed to change the number stored in source *A* of the EQU instruction?

   c. What number or numbers need to be set on the thumbwheel in order to turn on the pilot light?

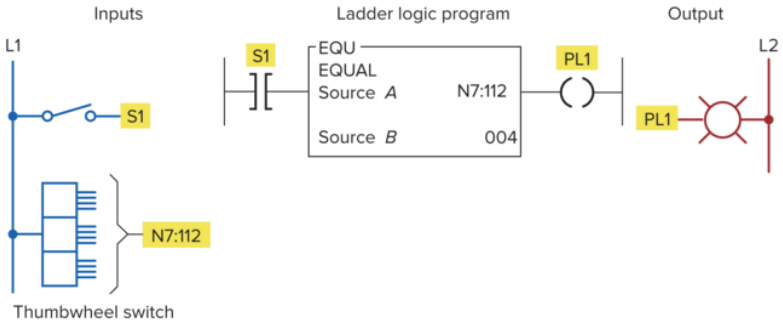


Figure 10-49 Program for Problem 4.

5. Study the data compare program in **Figure 10-50** and answer the following questions:

   a. List the values for the thumbwheel switch that would allow the pilot light to turn on.

   b. If the value in the word N7:112 is 003 and switch S1 is open, will the pilot light turn on? Why?

   c. Assume that source *B* is addressed to the accumulated count of an up-counter. With S1 closed, what setting of the thumbwheel switch would be required to turn the pilot light off when the accumulated count reaches 150?

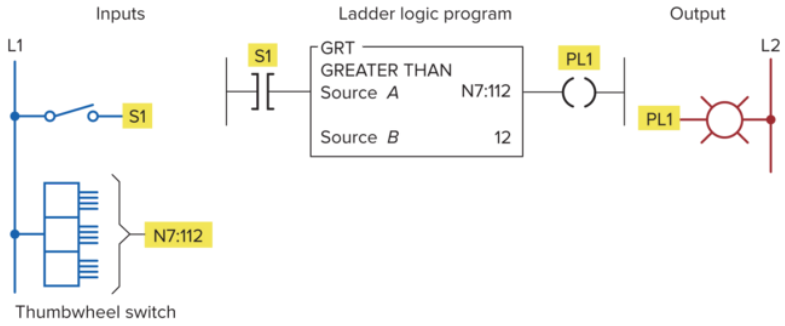


Figure 10-50 Program for Problem 5.

6. Write a program to perform the following:
   - **a.** Turn on pilot light 1 (PL1) if the thumbwheel switch value is less than 4.
   - **b.** Turn on pilot light 2 (PL2) if the thumbwheel switch value is equal to 4.
   - **c.** Turn on pilot light 3 (PL3) if the thumbwheel switch value is greater than 4.
   - **d.** Turn on pilot light 4 (PL4) if the thumbwheel switch value is less than or equal to 4.
   - **e.** Turn on pilot light 5 (PL5) if the thumbwheel switch value is greater than or equal to 4.
7. Write a program that will copy the value stored at address N7:56 into address N7:60.

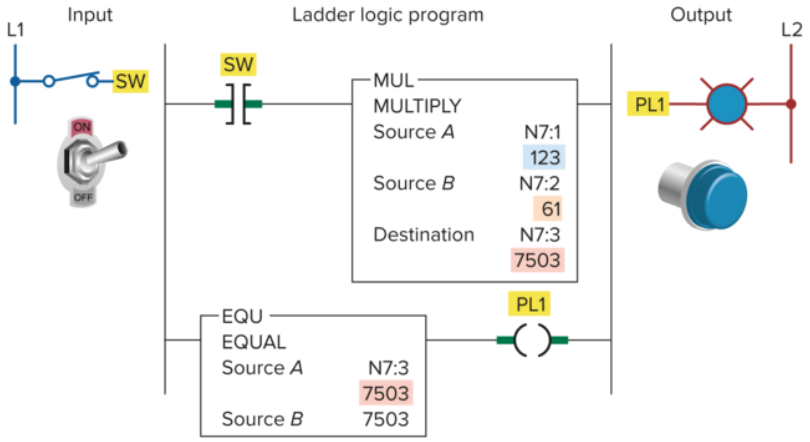
Page 218

8. Write a program that uses the mask move instruction to move only the upper 8 bits of the value stored at address I:2.0 to address O:2.1 and to ignore the lower 8 bits.
9. Write a program that uses the FAL instruction to copy 20 words of data from the integer data file, starting with N7:40, into the integer data file, starting with N7:80.
10. Write a program that uses the COP instruction to copy 128 bits of data from the memory area, starting at B3:0, to the memory area, starting at B3:8.
11. Write a program that will cause a light to come on only if a PLC counter has a value of 6 or 10.
12. Write a program that will cause a light to come on if a PLC counter value is less than 10 or more than 30.
13. Write a program for the following: The temperature reading from a thermocouple is to be read and stored in a memory location every 5 minutes for 4 hours. The temperature reading is brought in continuously and stored in address N7:150. File #7:200 is to contain the data from the last full 4-hour period.

OceanofPDF.com

11

Math Instructions



Chapter Objectives

After completing this chapter, you will be able to:

- Analyze and interpret math instructions as they apply to a PLC program
- Create PLC programs involving math instructions
- Apply combinations of PLC arithmetic functions to processes

Most PLCs have arithmetic function capabilities. Basic PLC math instructions include add, subtract, multiply, and divide to calculate the sum, difference, product, and quotient of the content of word registers. The PLC is capable of doing many arithmetic operations per scan period for fast updating of data. This chapter covers the basic mathematical instructions performed by PLCs and their applications.

OceanofPDF.com

11.1 Math Instructions

The PLC's math functions capability allows it to perform arithmetic functions on values stored in memory arrays, words or registers. For example, assume you are using a counter to keep track of the number of parts manufactured, and you would like to display how many more parts must be produced in order to reach a certain quota. This display would require the data in the accumulated value of the counter to be subtracted from the quota required. Other applications include combining parts counted, subtracting detected defects, and calculating run rates.

Depending on what type of processor is used, various math instructions can be programmed. The basic four mathematical functions performed by PLCs are:

Addition—The capability to add one piece of data to another.

Subtraction—The capability to subtract one piece of data from another.

Multiplication—The capability to multiply one piece of data by another.

Division—The capability to divide one piece of data by another.

Math instructions use the contents of two memory locations and perform the desired function. The PLC instructions for data manipulation (data transfer and data compare) are used with the math symbols to perform math functions. Math instructions are all output instructions. These instructions can be conditional or unconditional. With conditional arithmetic instructions the input logic determines when the instruction executes. Unconditional arithmetic instructions execute with each scan.

Figure 11-1 shows the Compute/Math menu tab for the RSLogix 500 software. The commands can be summarized as follows:

CPT (Compute)—Evaluates an expression and stores the result in the destination.

ADD (Add)—Adds source *A* to source *B* and stores the result in the destination.

SUB (Subtract)—Subtracts source *B* from source *A* and stores the result in the destination.

MUL (Multiply)—Multiplies source *A* by source *B* and stores the result in the destination.

DIV (Divide)—Divides source *A* by source *B* and stores the result in the math register and destination.

SQR (Square Root)—Calculates the square root of the source and places the integer result in the destination.

NEG (Negate)—Changes the sign of the source and places it in the destination.

TOD (To BCD)—Converts a 16-bit integer source value to BCD and stores it in the destination.

FRD (From BCD)—Converts a BCD value in the math register or the source to an integer and stores it in the destination.

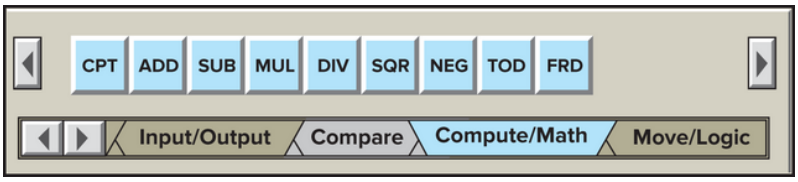


Figure 11-1 Compute/Math menu tab.

The basic math instructions are ADD, SUB, MUL, and DIV. Each of these instructions has three parameter fields. Namely, Source A, Source B, and Destination fields.

- The **Source A** and **Source B** fields can be an input rack location, file address, instruction field, or a fixed value. For example:

Input Location	I:1
File Address	N7:5
Instruction Field	C5:2.ACC
Fixed Value	30

- The **Destination** fields can be an output location, file address, or an instruction field. For example:

Output location	O:2
File Address	N7:8
Instruction Field	T4:1.PRE

Figure 11-2 shows the ***CPT (compute)*** instruction used with RSLogix 500 controllers. When CPT instruction is executed, then copy, arithmetic, logical, or conversion operation residing in the expression field of this instruction is performed and the result is sent to the destination. The execution time of a CPT instruction is longer than that of a single arithmetic operation and uses more instruction words.

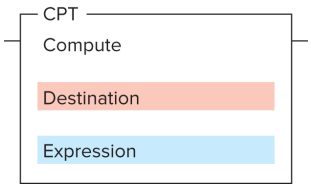


Figure 11-2 RSLogix 500 CPT (compute) instruction.

The main advantage of the compute instruction is that it allows you to enter quite complex expressions in one instruction. **Figure 11-3** shows a ladder rung used to convert a Fahrenheit temperature to a Celsius temperature using a single RSLogix 5000 compute instruction. The CPT instruction for the RSLogix 500 and 5000 processors operate in the same manner. This operation of the rung can be summarized as follows:

- The arithmetic operation $[^{\circ}\mathrm{C} = (^{\circ}\mathrm{F} - 32) \times (5/9)]$ is defined in the Expression.

- The compute operation is performed whenever the Temp_Convert input tag is true.
- When the CPT instruction is executed the result of the equation is put into the DEST tag name Result.
- In this example a temperature of $140^{\circ}\mathrm{F}$ is programmed into N7:5 of the Expression and the computed value of $60^{\circ}\mathrm{C}$ appears in the Result.
- The CPT has its own instruction set consisting of commands, or operators, that can be embedded in a mathematical expression. These commands range from add and subtract to BCD conversion and absolute values.
- The operations you write into the expression are performed by the instruction in a prescribed order, not necessarily the order you write them. The first order is parentheses so you can override the order of operation by grouping terms within parentheses. This forces the instruction to perform an operation within the parentheses ahead of other operations. In this case parentheses were used to assure that the subtraction was done before the multiplication and division.

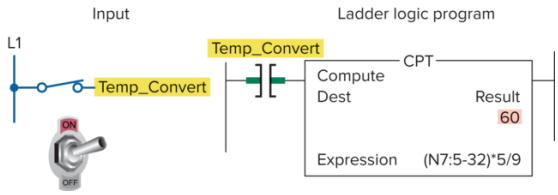


Figure 11-3 Compute instruction used to convert from Fahrenheit to Celsius.

OceanofPDF.com

11.2 Addition Instruction

Math instructions take two input values, perform the specified arithmetic function, and output the result to an assigned memory location. For example, the ***ADD*** instruction performs the addition of two values stored in the referenced memory locations. How these values are accessed depends on the controller. **Figure 11-4** shows the ADD instruction used with the RSLogix 500 controllers. The operation of the logic rung can be summarized as follows:

- When input switch SW is closed the rung will be true.
- The value stored at the source *A* address, N7:0 (25), is added to the value stored at the source *B* address, N7:1 (50).
- The answer (75) is stored at the destination address N7:2.
- Source *A* and source *B* can be either values or addresses that contain values, but *A* and *B* cannot both be constants.

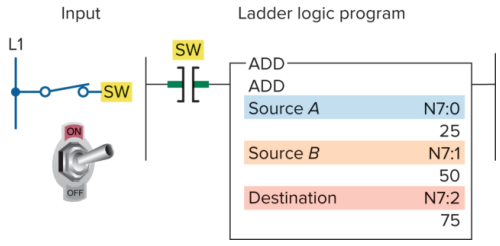


Figure 11-4 RSLogix 500 ADD instruction.

The program of **Figure 11-5** illustrates how the ADD instruction can be used to add the accumulated counts of two up-counters. This application requires a pilot light to come on when the sum of the counts from the two counters is equal to or greater than 350. The operation of the program can be summarized as follows:

- Source *A* of the ADD instruction is addressed to the accumulated value of counter C5:0.
- Source *B* of the ADD instruction is addressed to the accumulated value of counter C5:1.
- The value at source *A* is added to the value at source *B,* and the result (answer) is stored at destination address N7:1.
- Source *A* of the GEQ (greater than or equal) instruction is addressed to the value of the destination address N7:1.
- Source *B* of the GEQ instruction contains the constant value of 350.
- The GEQ instruction and PL1 output will be true whenever the accumulated sum of the values in the two counters is equal to or greater than the constant value 350.
- A reset button is provided to reset the accumulated count of both counters to zero.

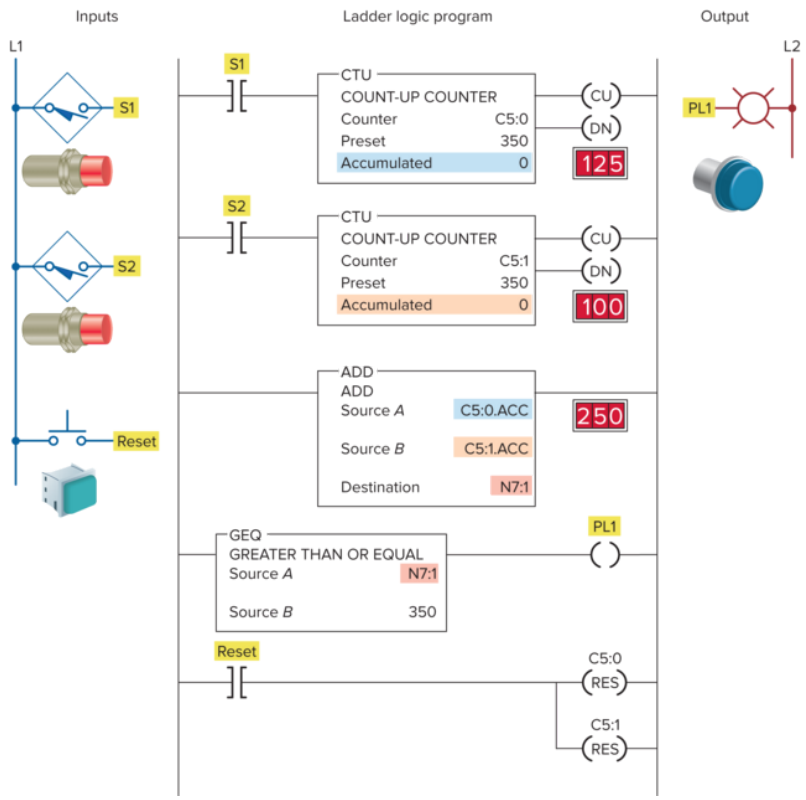


Figure 11-5 Counter program that uses the ADD instruction.

When performing math functions, care must be taken to ensure that values remain in the range that the data table or file can store; otherwise, the overflow bit will be set. The arithmetic status bits for RSLogix 500 controller are found in word 0, bits 0 to 3 of the processor status file S2 (**Figure 11-6**). After an instruction is executed, the arithmetic status bits in the status file are updated. The description of each bit can be summarized as follows:

Carry (C)—Address S2:0/0, is set to 1 when there is a carry in the ADD instruction or a borrow in the SUB instruction.

Overflow (O)—Address S2:0/1, is set to 1 when the result is too large to fit in the destination register.

Zero Bit (Z)—Address S2:0/2, is set to 1 when the result of the subtract instruction is zero.

Sign Bit (S)—Address S2:0/3, is set to 1 when the result is a negative number.

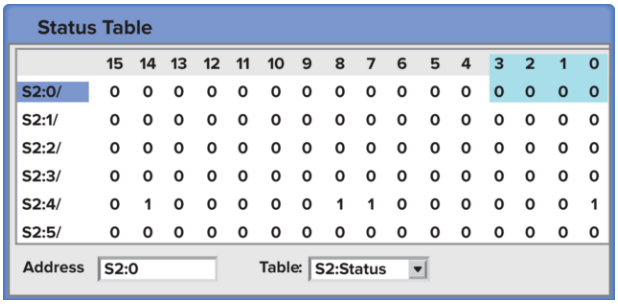


Figure 11-6 Processor status file S2.

OceanofPDF.com

11.3 Subtraction Instruction

The ***SUB (subtract)*** instruction is an output instruction that subtracts one value from another and stores the result in the destination address. When rung conditions are true, the subtract instruction subtracts source *B* from source *A* and stores the result in the destination. **Figure 11-7** shows the SUB instruction used with the RSLogix 500 controllers. The operation of the logic rung can be summarized as follows:

- When input switch SW is closed the rung will be true.
- The value stored at the source *B* address, N7:05 (322), is subtracted from the value stored at the source *A* address, N7:10 (520).
- The answer (198) is stored at the destination address, N7:20.
- Source *A* and source *B* can be either values or addresses that contain values, but *A* and *B* cannot both be constants.

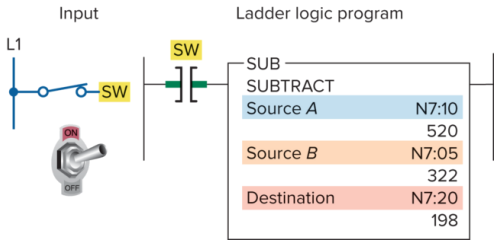


Figure 11-7 RSLogix 500 SUB (subtract) instruction.

The program of **Figure 11-8** shows how the SUB function can be used to indicate a vessel overfill condition. This application requires an alarm to sound when a supply system leaks 5 lb or more of raw material into the vessel after a preset weight of 500 lb has been reached. The operation of the program can be summarized as follows:

- When the start button is pressed, the fill solenoid (rung 1) and filling indicating light (rung 2) are turned on and raw material is allowed to flow into the vessel.
- The vessel has its weight monitored continuously by the PLC program (rung 3) as it fills.
- When the weight reaches 500 lb, the fill solenoid is de-energized and the flow is cut off.
- At the same time, the filling pilot light indicator is turned off and the full pilot light indicator (rung 3) is turned on.
- Should the fill solenoid leak 5 lb or more of raw material into the vessel, the alarm (rung 5) will energize and stay energized until the overflow level is reduced below the 5-lb overflow limit.

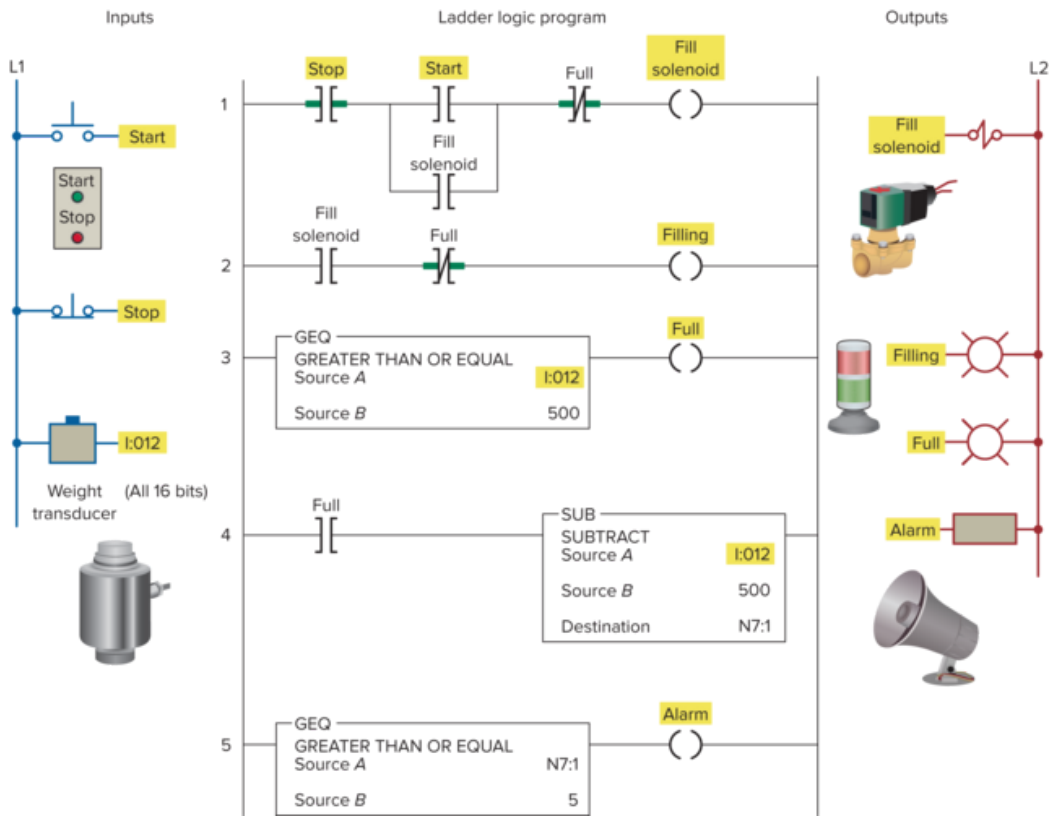


Figure 11-8 Vessel overfill alarm program.

OceanofPDF.com

11.4 Multiplication Instruction

The ***multiply (MUL)*** instruction is an output instruction that multiplies two values and stores the result in the destination address. **Figure 11-9** shows the MUL instruction used with the RSLogix 500 controllers. The operation of the logic rung can be summarized as follows:

- When input switch SW is closed the rung will be true.
- The data in source *A* (constant 20) will be multiplied by the data in source *B* (accumulated value of counter C5:10).
- The resultant answer is placed in the destination N7:2.
- Similar to previous math instructions, source *A* and *B* in multiplication instructions can be values (constants) or addresses that contain values, but *A* and *B* cannot both be constants.

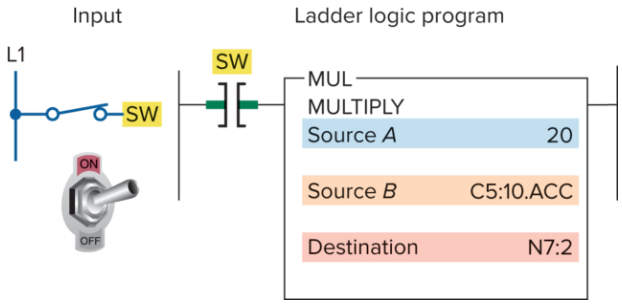


Figure 11-9 RSLogix 500 MUL (multiply) instruction.

The program of **Figure 11-10** is an example of how MUL instruction calculates the product of two sources. The operation of the program can be summarized as follows:

- When input switch SW is closed the MUL instruction is executed.
- The value stored in source *A,* address N7:1 (123), is then multiplied by the value stored in source *B,* address N7:2 (61).
- The product (7503) is placed into destination word N7:3.
- As a result, the equal instruction becomes true, turning output PL1 on.

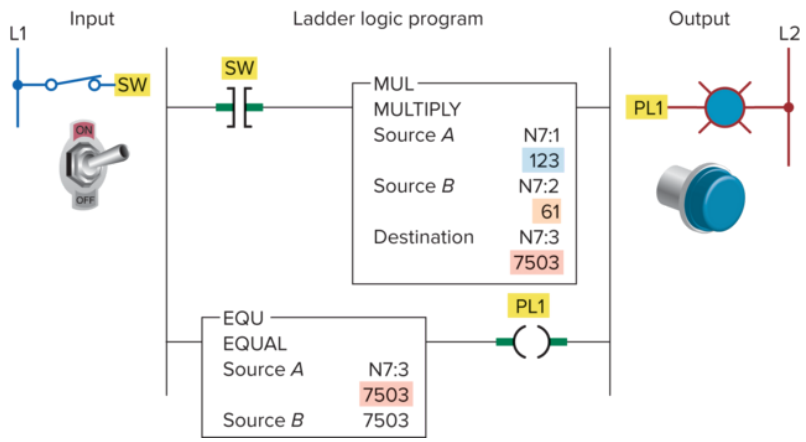


Figure 11-10 MUL instruction used to calculate the product of two sources.

The program of **Figure 11-11** is an example of how the MUL instruction is used as part of an oven temperature control program. The operation of the program can be summarized as follows:

- The PLC calculates the upper and lower deadband, or off/on limits, about the set-point.
- Upper and lower temperature limits are set automatically at ± 1 percent regardless of the set-point value.
- Set-point temperature is adjusted by means of the thumbwheel switch.
- The analog thermocouple interface module is used to monitor the current temperature of the oven.
- In this example, the set-point temperature is $400^{\circ}\mathrm{F}$.

- Therefore, the electric heaters will be turned on when the temperature of the oven drops to less than $396^{\circ}\mathrm{F}$ and stay on until the temperature rises above $404^{\circ}\mathrm{F}$.
- If the set-point is changed to $100^{\circ}\mathrm{F}$, the deadband remains at ± 1 percent, with the lower limit being $99^{\circ}\mathrm{F}$ and the upper limit being $101^{\circ}\mathrm{F}$.
- The number stored in word N7:1 represents the upper temperature limit, and the number stored in word N7:2 represents the lower limit.

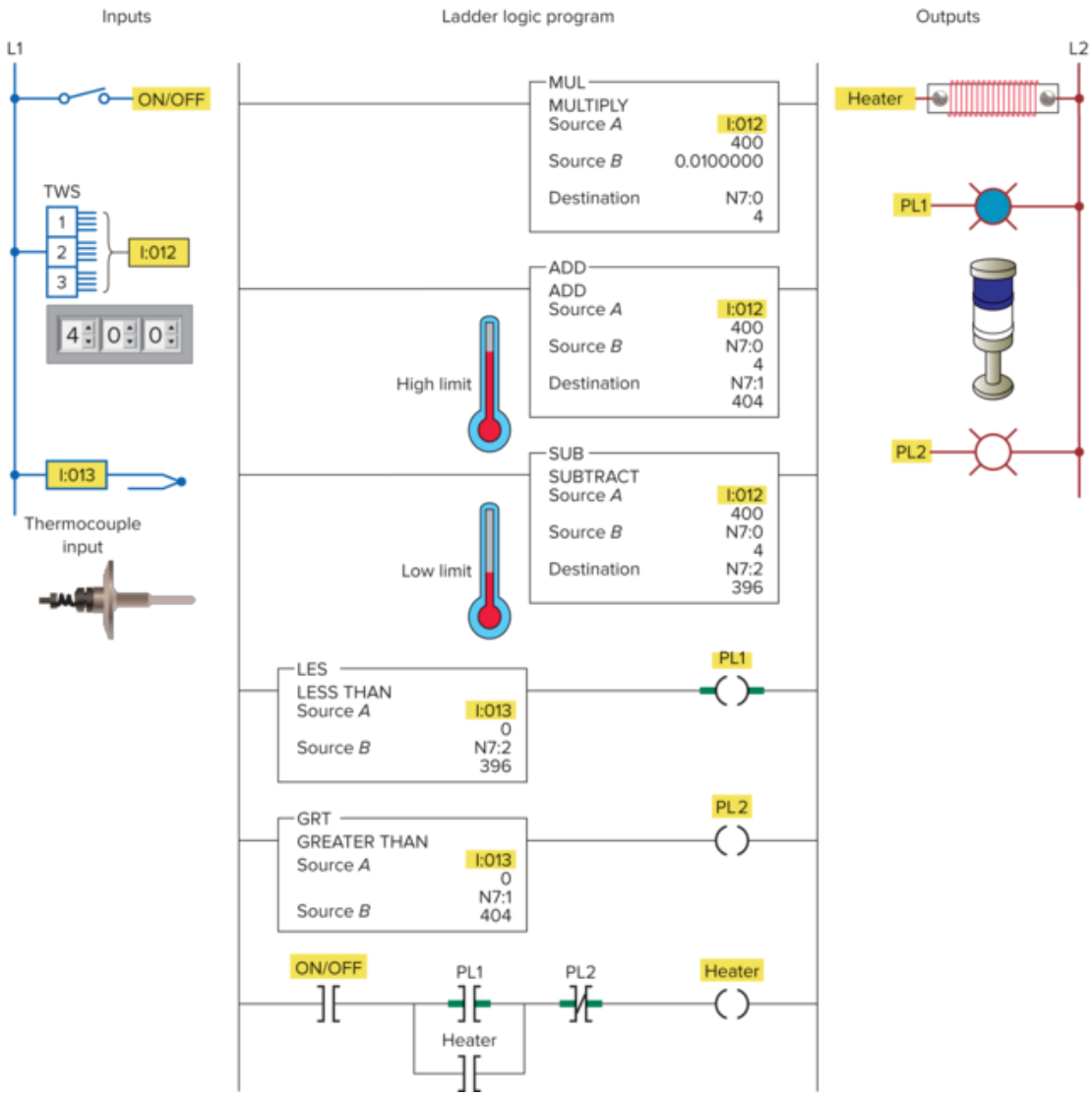


Figure 11-11 The MUL instruction used as part of a temperature control program.

OceanofPDF.com

11.5 Division Instruction

The ***divide (DIV)*** instruction divides the value in source *A* by the value in source *B* and stores the result in the destination and math register. **Figure 11-12** shows an example of the DIV instruction. The operation of the logic rung can be summarized as follows:

- When input switch SW is closed the rung will be true.
- The data in source *A* (the accumulated value of counter C5:10) is then divided by the data in source *B* (the constant 2).
- The result is placed in the destination N7:3.
- If the remainder is 0.5 or greater, a roundup occurs in the integer destination.
- The value stored in the math register consists of the unrounded quotient (placed in the most significant word) and the remainder (placed in the least significant word).
- Most PLCs support the use of floating point (real) numbers as well as integer (whole number) values. As an example, 10 divided by 3 may be expressed as 3.333333 (floating-point notation) or 3 with a remainder of 1.
- A minor fault bit is set upon detection of a division by zero.

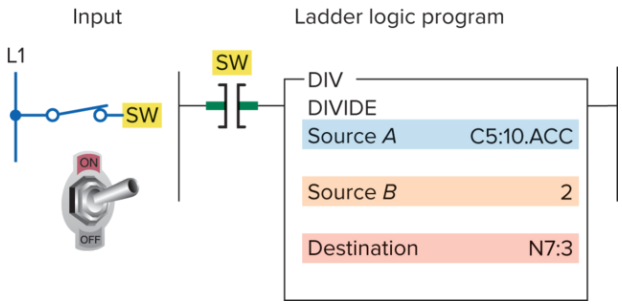


Figure 11-12 RSLogix 500 DIV (divide) instruction.

The program of **Figure 11-13** is an example of how the DIV instruction calculates the integer value that results from dividing source *A* by source *B*. The operation of the program can be summarized as follows:

- When input switch SW is closed the DIV instruction is executed.
- The value stored in source *A,* address N7:0 (120), is then divided by the value stored in source *B,* address N7:1 (4).
- The answer, 30, is placed in the destination address N7:5.
- As a result, the equal instruction becomes true, turning output PL1 on.

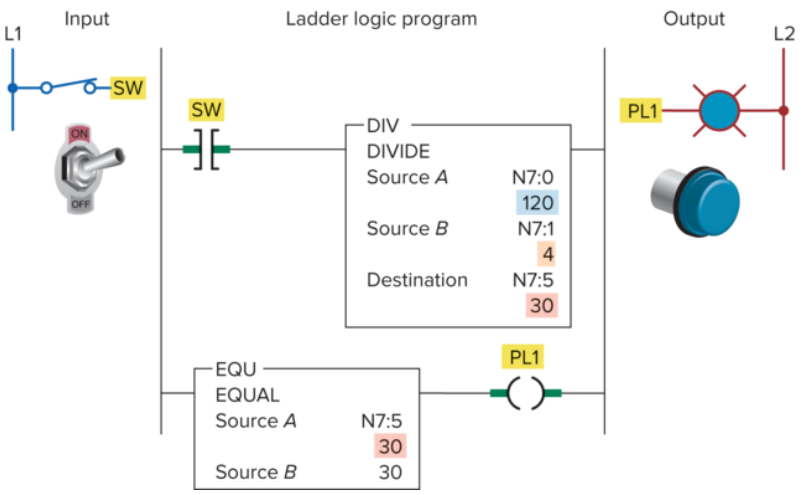


Figure 11-13 DIV instruction used to calculate the value that results from dividing source *A* by source *B*.

The program of **Figure 11-14** is an example of how the DIV function is used as part of a program to convert Celsius temperature to Fahrenheit. The operation of the program can be summarized as follows:

- The thumbwheel switch connected to the input module indicates Celsius temperature.
- The program is designed to convert the recorded Celsius temperature in the data table to Fahrenheit values for display.
- The following conversion formula forms the basis for the program:

$$F = \left(\frac{9}{5} \times C\right) + 32$$

- In this example, a current temperature reading of $60^{\circ}\mathrm{C}$ is assumed.
- The PLC processor carries out its internal operations using binary numbers and the FRD instruction is used to convert the BCD thumbwheel switch setting to its equivalent binary value.
- The MUL instruction multiplies the temperature $(60^{\circ}\mathrm{C})$ by 9 and stores the product (540) in address N7:0.
- The DIV instruction divides the value stored at N7:0 (540) by a constant (5) and stores the answer (108) in address N7:1.
- The ADD instruction adds 32 to the value of 108 and stores the sum (140) in address B3:1.
- Finally the TOD instruction is used to convert the binary value to its BCD equivalent to interface with the LED display.
- Thus $60^{\circ}\mathrm{C}$ is displayed as $140^{\circ}\mathrm{F}$.

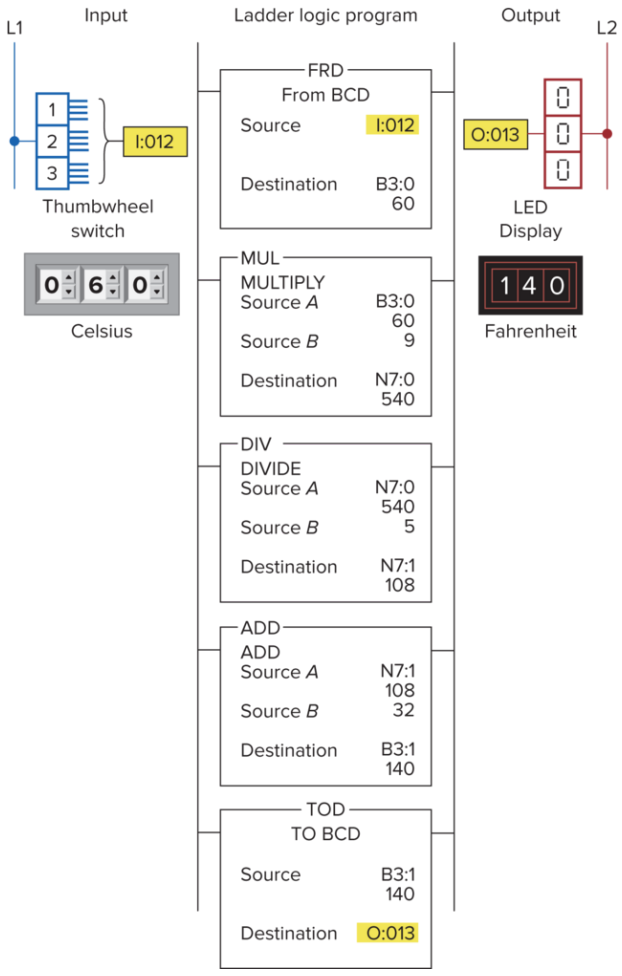


Figure 11-14 Program for converting Celsius temperature to Fahrenheit.

The **Modulo (MOD)** instruction, shown in Figure 11-15, is part of the RSLogix 5000 Compute/Math instruction set. This instruction is used to calculate the **remainder** after the value stored in Source A is divided by the value stored in Source B. In this unconditional rung example:

$$\frac{Source\ A}{Source\ B} = Dest(remainder)$$

$$\frac{Thumb_Value}{2} = Actual_Value$$

$$Thumb_Value = 7$$

$$\frac{7}{2} = 3 \text{ remainder } 1$$

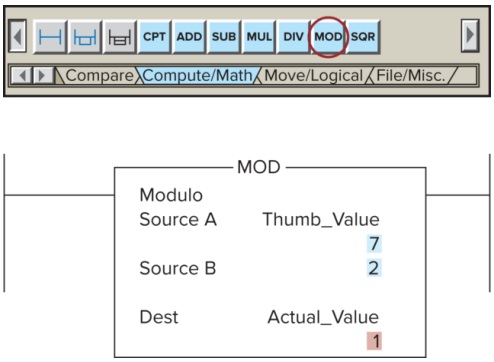


Figure 11-15 RSLogix 5000 Modulo (MOD) instruction.

OceanofPDF.com

11.6 Other Word-Level Math Instructions

The program of Figure 11-16 is an example of the ***square root (SQR)*** instruction. The operation of the logic rung can be summarized as follows:

- When input switch SW is closed the SQR instruction is executed.
- The number whose square root we want to determine (144) is placed in the source.
- The function calculates the square root and places it (12) in the destination.
- If the value of the source is negative, the instruction will store the square root of the absolute (positive) value of the source at the destination.

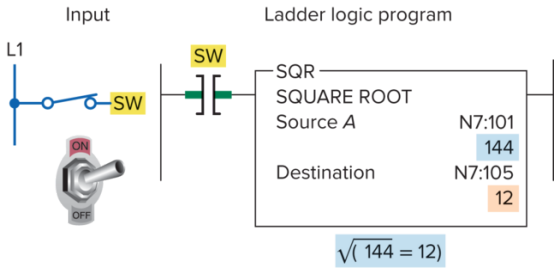


Figure 11-16 RSLogix 500 SQR (square root) instruction.

The program of Figure 11-17 is an example of the ***negate (NEG)*** instruction. This math function changes the sign of the source value from positive to negative. The operation of the logic rung can be summarized as follows:

- When input switch SW is closed the NEG instruction is executed.
- The positive value 101 stored at the source address N7:52 is negated to −101 and stored in destination address N7:53.
- Positive numbers will be stored in straight binary format, and negative numbers will be stored as 2's complement.

Page 228

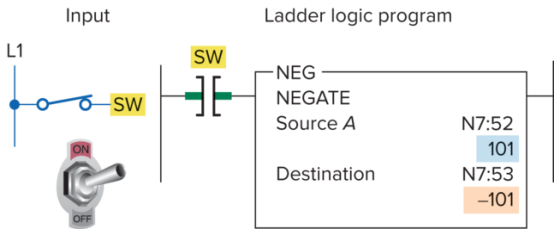


Figure 11-17 RSLogix 500 NEG (negate) instruction.

The program of Figure 11-18 is an example of the ***clear (CLR)*** instruction. The operation of the logic rung can be summarized as follows:

- When input switch SW is closed the CLR instruction is executed.
- Upon execution it sets all bits of a word to zero.
- In this example it changes the value of all bits stored in the destination address N7:22 to 0.

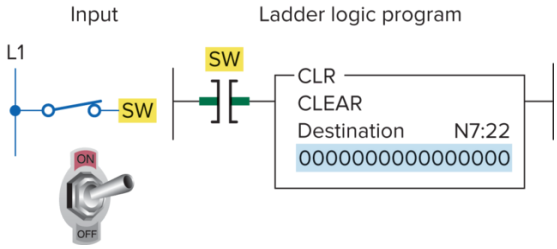


Figure 11-18 RSLogix 500 CLR (clear) instruction.

The ***convert to BCD (TOD)*** instruction is used to convert 16-bit integers into *binary-coded decimal (BCD)* values. This instruction could be used when transferring data from the processor (which stores data in binary format) to an external device, such as an LED display, that functions in BCD format. The program of **Figure 11-19** is an example of the TOD instruction. The operation of the logic rung can be summarized as follows:

- When input switch SW is closed the TOD instruction is executed.
- The binary bit pattern at the source address N7:23 is converted into a BCD bit pattern of the same decimal value at the destination address O:20.
- The source displays the value 10, which is the correct decimal value; however, the destination displays the value 16.
- The processor interprets all bit patterns as binary; therefore the value 16 is the binary interpretation of the BCD bit pattern.
- The bit pattern for 10 BCD is the same as the bit pattern for 16 binary.

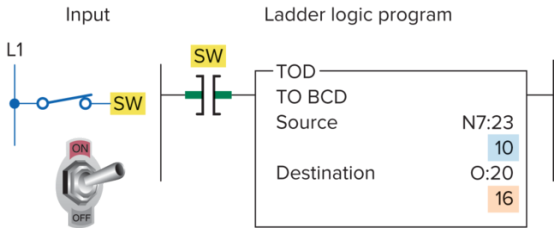


Figure 11-19 RSLogix 500 TOD (convert to BCD) instruction.

The ***convert from BCD (FRD)*** instruction is used to convert binary-coded decimal (BCD) values to integer values. This instruction could be used to convert data from a BCD external source, such as a BCD thumbwheel switch, to the binary format in which the processor operates. The program of **Figure 11-20** is an example of the FRD instruction. The operation of the logic rung can be summarized as follows:

- When input switch SW is closed the FDR instruction is executed.

- The BCD bit pattern stored at the source address I:30 is converted into a binary bit pattern of the same decimal value at the destination address, N7:24.

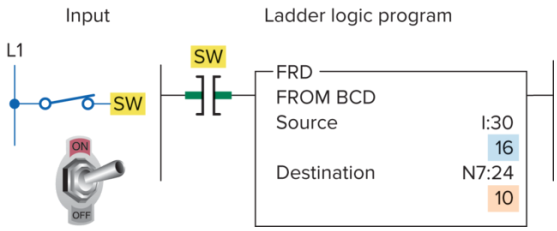


Figure 11-20 RSLogix 500 FRD (convert from BCD) instruction.

At times it is necessary to make conversions to analog input and output values to ensure correct interpretation and processing. As a result, data must be **scaled**, or resized, before it can be used by a PLC control algorithm or output to a field device. The RSLogix 500 Scale data (**SCL**) and Scale with Parameters (**SCP**) instructions are used to perform this task. Both instructions use the same formula to perform the scaling function, which is:

$$\mathbf{y = mx + b}$$

Where: y is the output
m is the scaling rate
x is the input
b is the offset

$$\text{Scaling rate (m)} = (\text{scaled Max} - \text{scaled Min})/(\text{input Max} - \text{input Min})$$

$$\text{Offset (b)} = (\text{scaled Min}) - (\text{input Min} \times \text{m})$$

The ladder rung of **Figure 11-21** is an example of the use of the SCL instruction. When rung conditions are true, this instruction multiplies the source by a specified rate. The rounded result is then added to an offset value and placed in the destination. The execution of the instruction can be summarized as follows:

- When input switch SW is closed the SCL instruction is executed.
- The number 100 stored at the source address, N7:0, is multiplied by the rate 25,000, divided by 10,000, and added to 127.
- The result, 377, is placed in the destination address, N7:1.

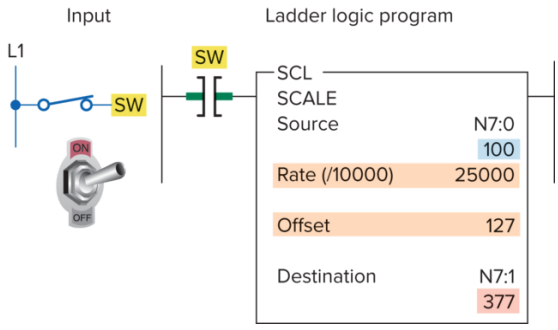


Figure 11-21 RSLogix 500 SCL (scale) instruction.

Figure 11-22 shows an example of an analog input to a PLC, and the SCP instruction used to scale its data. The execution of the instruction can be summarized as follows:

- A strain-gage pressure transducer is connected to input I:1.0.
- The gage measures pressure from 0 to 1000 psi and provides an analog output of 0 to 10V.
- The unscaled range is 0 to 32,767, and the output is loaded into N7:20.
- When executed, the SCP instruction places a number between 0 and 1000 into N7:20 (destination) based on the input signal (0 to 10V) coming from the transducer into the analog input module.

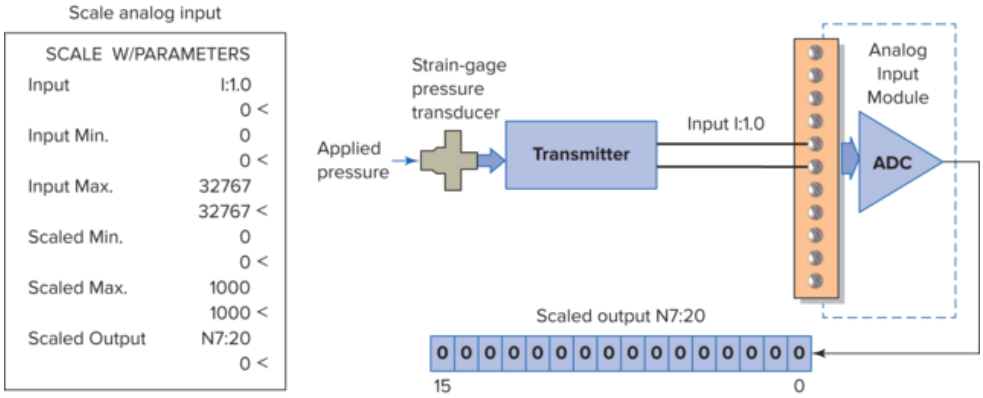


Figure 11-22 Scale analog input using the SCP instruction.

Figure 11-23 shows an example of an analog output from a PLC, and the SCP instruction used to scale its data. The execution of the instruction can be summarized as follows:

- A proportional control valve is connected to the PLC output O:1.0.
- A 4 to 20 mA signal operates the valve from closed to 100% open.
- The percent open is in location N7:21.
- The PLC analog module provides a 4 to 20 mA output signal for a number between from 6,242 to 31,208.
- The SCP directs analog output O:1.0 to provide a 4 to 20 mA signal, which is scaled to the valve position based on a number between 0 and 100.

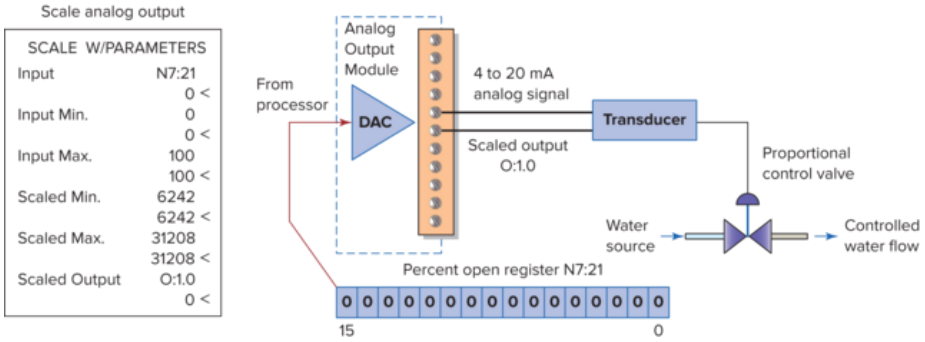


Figure 11-23 Scale analog output using the SCP instruction.

OceanofPDF.com

11.7 File Arithmetic Operations

File arithmetic functions include file add, file subtract, file multiply, file divide, file square root, file convert from BCD, and file convert to BCD. The *file arithmetic and logic (FAL)* instruction can combine an arithmetic operation with file transfer. The arithmetic operations that can be implemented with the FAL are ADD, SUB, MULT, DIV, and SQR.

The *file add* function of the FAL instruction can be used to perform addition operations on multiple words. The program of **Figure 11-24** is an example of the file add function of the FAL instruction. The operation of the logic rung can be summarized as follows:

- When input switch SW is closed the rung goes true and the expression tells the processor to add the data in file address N7:25 to the data stored in file address N7:50 and store the result in file address N7:100.
- The rate per scan is set at All, so the instruction goes to completion in one scan.

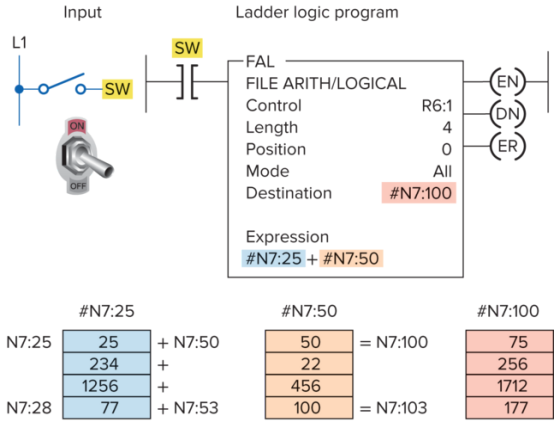


Figure 11-24 RSLogix 500 file add function of the FAL instruction.

The program of **Figure 11-25** is an example of the *file subtract* function of the FAL instruction. The operation of the logic rung can be summarized as follows:

- When input switch SW is closed the rung goes true and the processor subtracts a program constant (255) from each word of file address N10:0 and stores the result at the destination file address, N7:255.
- The rate per scan is set at 2, so it will take 2 scans from the moment the instruction goes true to complete its operation.

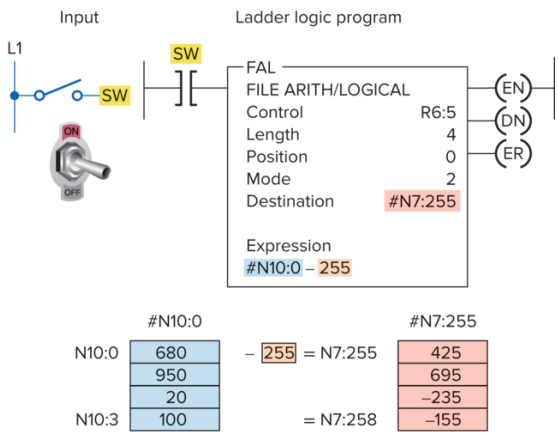


Figure 11-25 RSLogix 500 file subtract function of the FAL instruction.

The program of Figure 11-26 is an example of the ***file multiply*** function of the FAL instruction. The operation of the logic rung can be summarized as follows:

- When input switch SW is closed the rung goes true and the data in file address N7:330 is multiplied by the data in element address N7:23, with the result stored at the destination file address N7:500.
- The rate per scan is set at All, so the instruction goes to completion in one scan.

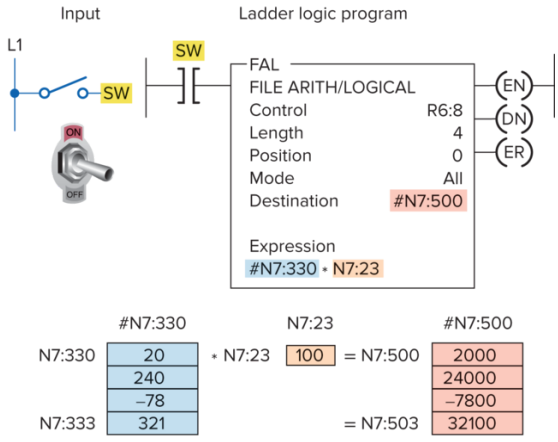


Figure 11-26 RSLogix 500 file multiply function of the FAL instruction.

The program of Figure 11-27 is an example of the ***file divide*** function of the FAL instruction. The operation of the logic rung can be summarized as follows:

- When input switch SW is closed the rung goes true and the data in file address F8:20 is divided by the data in file address F8:100, with the result stored in element address F8:200.
- The mode is Incremental, so the instruction operates on one set of elements for each false-to-true transition of the instruction.

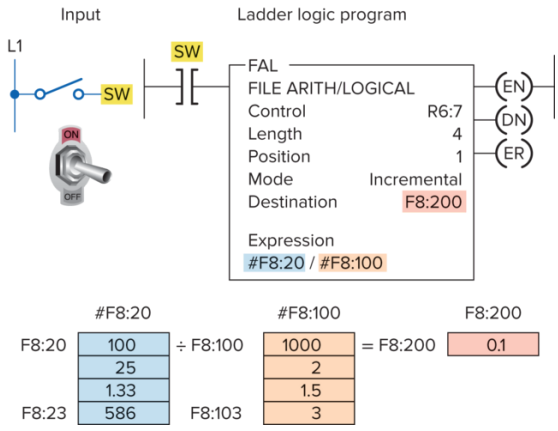


Figure 11-27 RSLogix 500 file divide function of the FAL instruction.

OceanofPDF.com

CHAPTER 11 REVIEW QUESTIONS

1. Explain the function of math instructions as applied to the PLC.
2. Name the four basic math functions performed by PLCs.
3. What standard format is used for PLC math instructions?
4. Would math instructions be classified as input or output instructions?
5. With reference to the instruction of **Figure 11-28**, what is the value of the number stored at source *B* if N7:3 contains a value of 60 and N7:20 contains a value of 80?

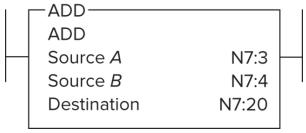


Figure 11-28 Instruction for Question 5.

6. With reference to the instruction of **Figure 11-29**, what is the value of the number stored at the destination if N7:3 contains a value of 500?

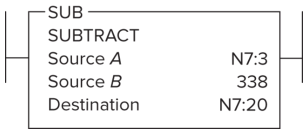


Figure 11-29 Instruction for Question 6.

7. With reference to the instruction of **Figure 11-30**, what is the value of the number stored at the destination if N7:3 contains a value of 40 and N7:4 contains a value of 3?

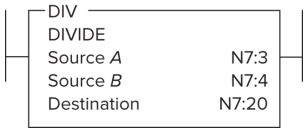


Figure 11-30 Instruction for Question 7.

8. With reference to the instruction of **Figure 11-31**, what is the value of the number stored at the destination if N7:3 contains a value of 15 and N7:4 contains a value of 4?

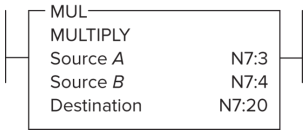


Figure 11-31 Instruction for Question 8.

9. With reference to the instruction of Figure 11-32, what is the value of the number stored at N7:20 if N7:3 contains a value of 2345?

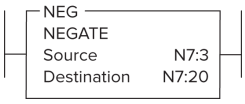


Figure 11-32 Instruction for Question 9.

10. With reference to the instruction of Figure 11-33, what will be the value of each of the bits in word B3:3 when the rung goes true?

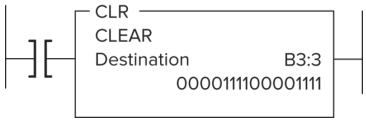


Figure 11-33 Instruction for Question 10.

11. With reference to the instruction of Figure 11-34, what is the value of the number stored at N7:101?

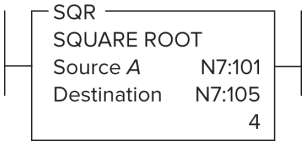
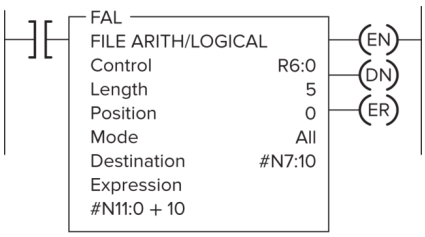


Figure 11-34 Instruction for Question 11.

12. With reference to the instruction of Figure 11-35, list the values that will be stored in file #N7:10 when the rung goes true.



File #N11:0

328
150
10
32
0

Figure 11-35 Instruction for Question 12.

OceanofPDF.com



CHAPTER 11 PROBLEMS

1. Answer each of the following with reference to the counter program shown in **Figure 11-36**.

a. Assume the accumulated count of counters C5:0 and C5:1 to be 148 and 36, respectively. State the value of the number stored in each of the following words at this point:

(1) C5:0.ACC

(2) C5:1.ACC

(3) N7:1

(4) Source *B* of the GEQ instruction

b. Will output PL1 be energized at this point? Why?

c. Assume the accumulated count of counters C5:0 and C5:1 to be 250 and 175, respectively. State the value of the number stored in each of the following words at this point:

(1) C5:0.ACC

(2) C5:1.ACC

(3) N7:1

(4) Source *B* of the GEQ instruction

d. Will output PL1 be energized at this point? Why?

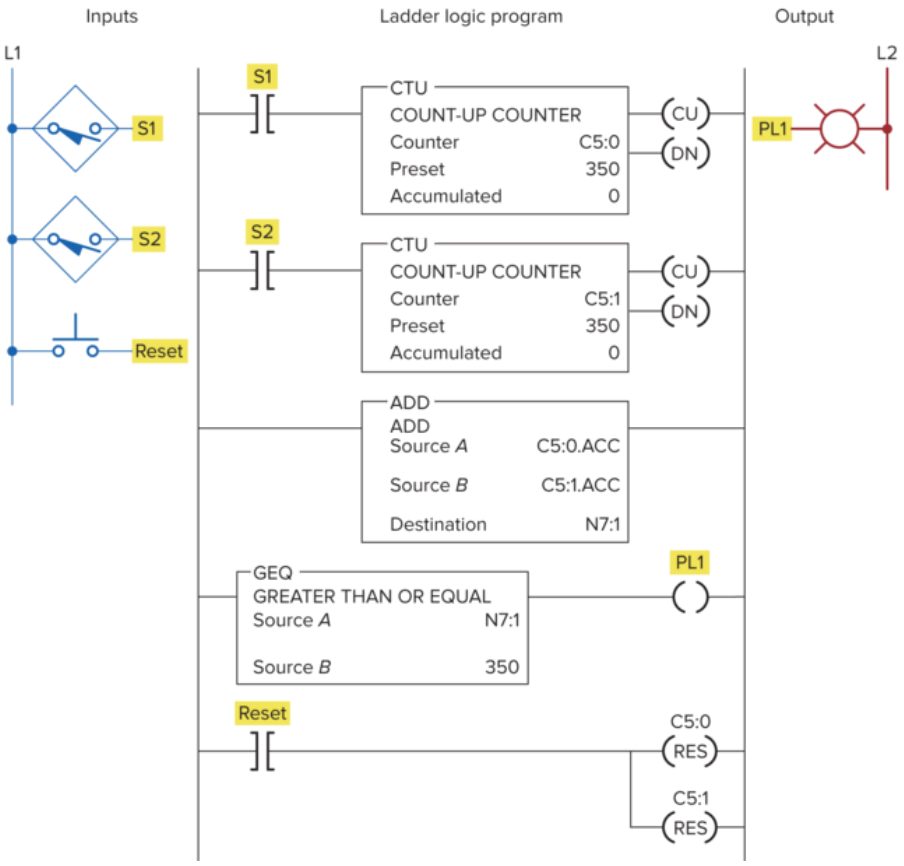


Figure 11-36 Program for Problem 1.

2. Answer each of the following with reference to the overfill alarm program shown in **Figure 11-37**.

a. Assume that the vessel is filling and has reached the 300-lb point. State the status of each of the logic rungs (true or false) at this point.

b. Assume that the vessel is filling and has reached the 480-lb point. State the value of the number stored in each of the following words at this point:

(1) I:012

(2) N7:1

c. Assume that the vessel is filled to a weight of 502 lb. State the status of each of the logic rungs (true or false) for this condition.

d. Assume that the vessel is filled to a weight of 510 lb. State the value of the number stored in each of the following words for this condition:

(1) I:012

(2) N7:1

e. With the vessel filled to a weight of 510 lb, state the status of each of the logic rungs (true or false).

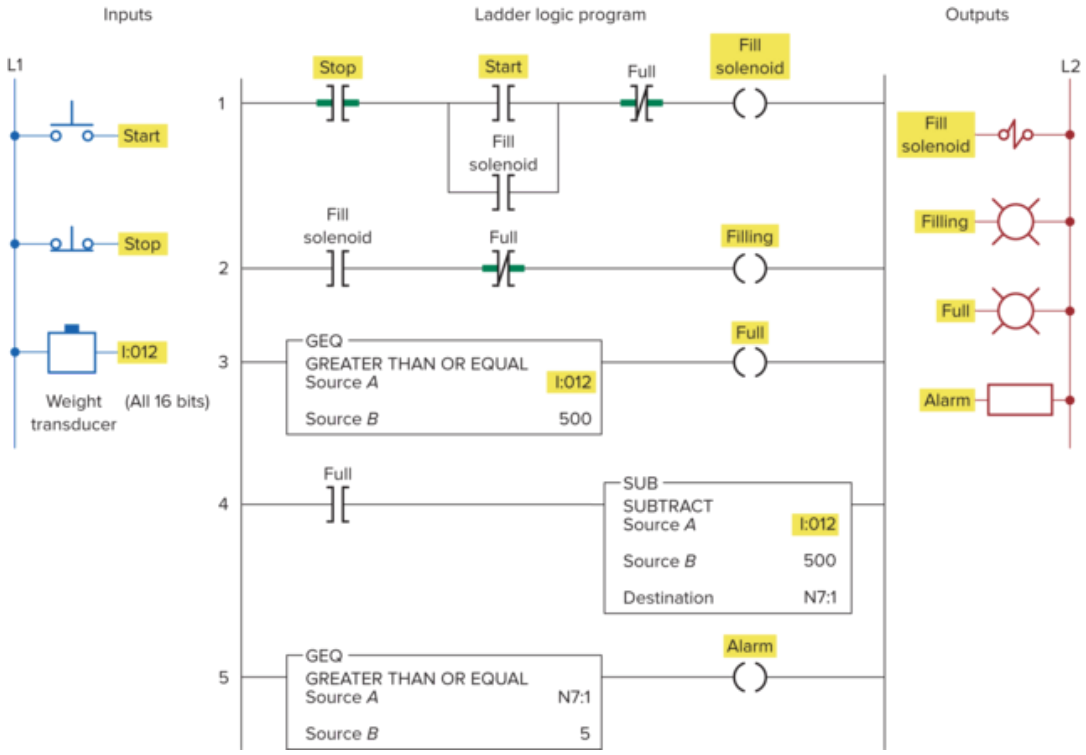


Figure 11-37 Program for Problem 2.

3. Answer the following with reference to the temperature control program shown in **Figure 11-38**.

   a. Assume that the set-point temperature is 600°F. At what temperature will the electric heaters be turned on and off?

   b. Assume that the set-point temperature is 600°F and the thermocouple input module indicates a temperature of 590°F. What is the value of the number stored in each of the following words at this point?

   (1) I:012

   (2) I:013

   (3) N7:0

   (4) N7:1

   (5) N7:2

   c. Assume that the set-point temperature is 600°F and the thermocouple input module indicates a temperature of 608°F. What is the status (energized or not energized) of each of the following outputs?

   (1) PL1

   (2) PL2

   (3) Heater

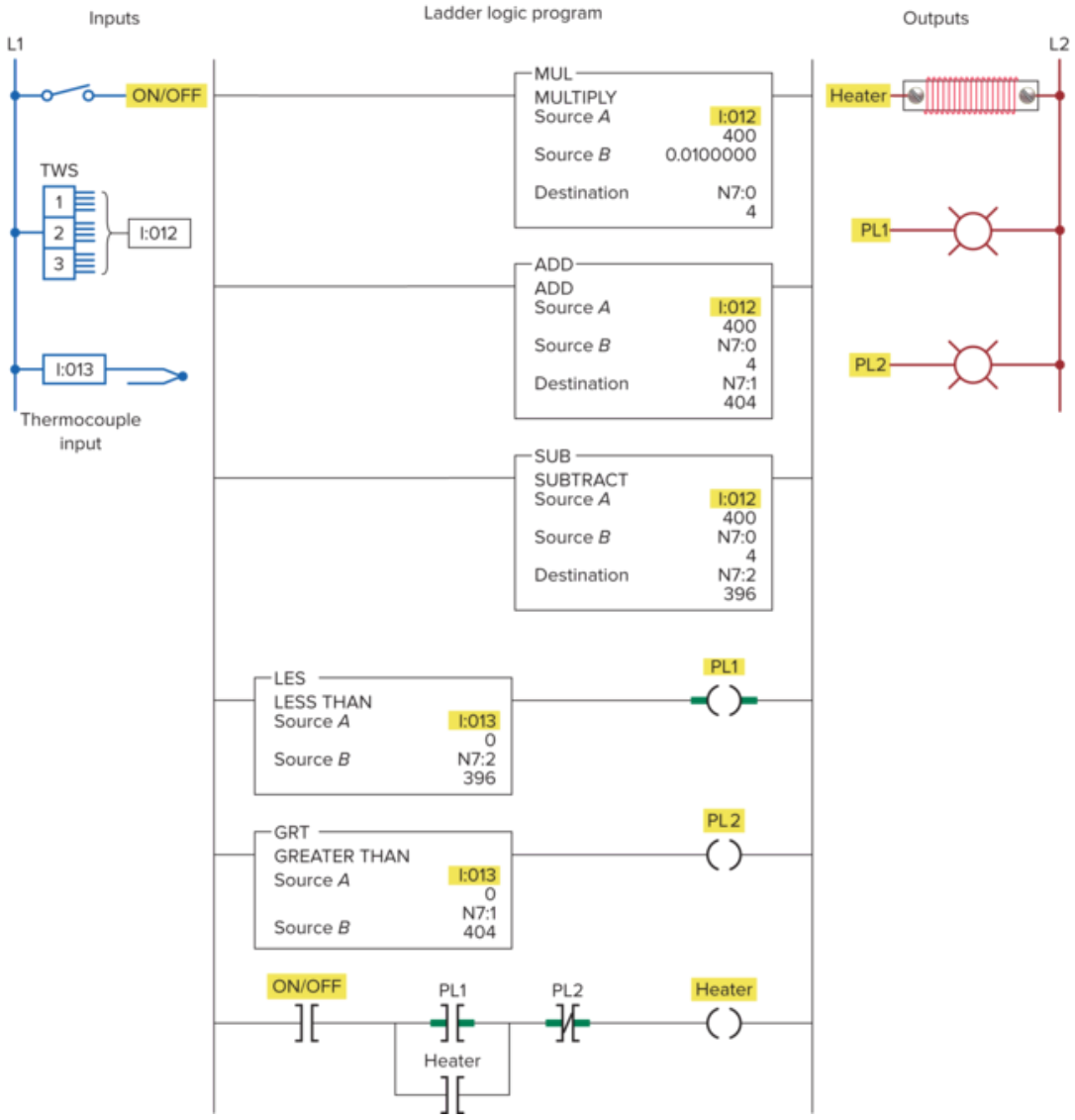


Figure 11-38 Program for Problem 3.

4. With reference to the Celsius to Fahrenheit conversion program shown in **Figure 11-39**, state the value of the number stored in each of the following words for a thumbwheel setting of 035:

a. I:012

b. N7:0

c. N7:1

d. O:013

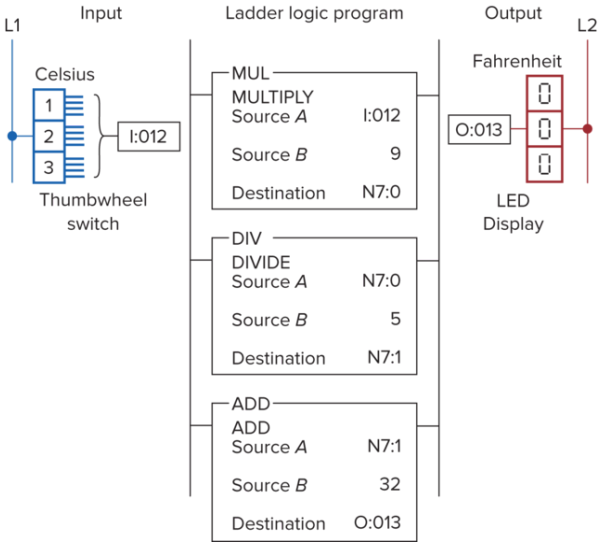


Figure 11-39 Program for Problem 4.

5. Design a program that will add the values stored at N7:23 and N7:24 and store the result in N7:30 whenever input *A* is true, and then, when input *B* is true, will copy the data from N7:30 to N7:31.

6. Design a program that will take the accumulated value from TON timer T4:1 and display it on a 4-digit, BCD format set of LEDs. Use address O:023 for the LEDs. Include the provision to change the preset value of the timer from a set of 4-digit BCD thumbwheels when input *A* is true. Use address I:012 for the thumbwheels.

Page 235

7. Design a program that will implement the following arithmetic operation:

- Use an MOV instruction and place the value 45 in N7:0 and 286 in N7:1.
- Add the values together and store the result in N7:2.
- Subtract the value in N7:2 from 785 and store the result in N7:3.
- Multiply the value in N7:3 by 25 and store the result in N7:4.
- Divide the value in N7:4 by 35 and store the result in F8:0.

8. a. There are three part conveyor lines $(1 - 2 - 3)$ feeding a main conveyor. Each of the three conveyor lines has its own counter. Construct a PLC program to obtain the total count of parts on the main conveyor.

b. Add a timer to the program that will update the total count every 30 s.

9. With reference to math instruction program shown in **Figure 11-40**, when the input goes true, what value will be stored at each of the following?

Page 236

a. N7:3

b. N7:5

c. F8:1

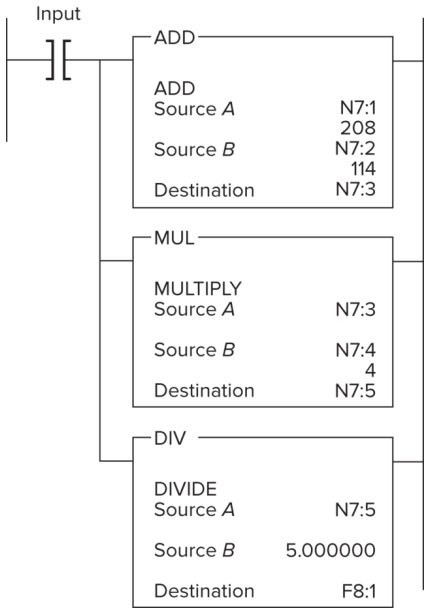


Figure 11-40 Program for Problem 9.

10. With reference to the math instruction program shown in **Figure 11-41**, when the input goes true, what value will be stored at each of the following?

a. N7:3

b. N7:4

c. N7:5

d. N7:6

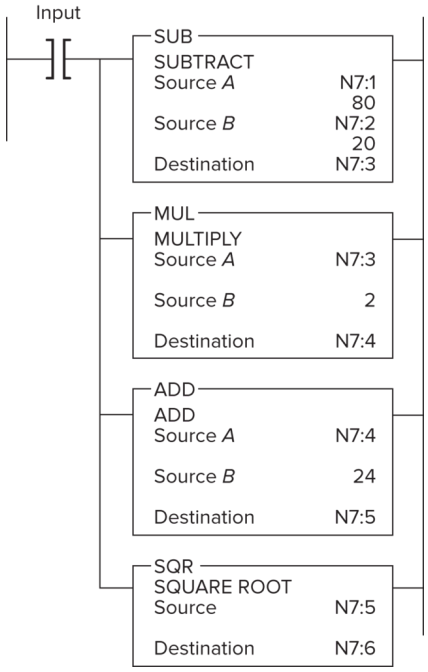


Figure 11-41 Program for Problem 10.

11. Two part conveyor lines, A and B, feed a main conveyor line, M. A third conveyor line, R, removes rejected parts a short distance away from the main conveyor. Conveyors A, B, and R have parts counters connected to them. Construct a PLC program to obtain the total parts output of main conveyor M.

12. A main conveyor has two conveyors, A and B, feeding it. Feeder conveyor A puts six-packs of canned soda on the main conveyor. Feeder conveyor B puts eight-packs of canned soda on the main conveyor. Both feeder conveyors have counters that count the number of *packs* leaving them. Construct a PLC program to give a *total can* count on the main conveyor.

OceanofPDF.com

12

Sequencer and Shift Register Instructions



Image Courtesy Banner Engineering Corp.

Chapter Objectives

After completing this chapter, you will be able to:

- Identify and describe the various forms of mechanical sequencers and explain the basic operation of each
- Interpret and explain information associated with PLC sequencer output, compare, and load instructions
- Compare the operation of an event-driven and a time-driven sequencer
- Describe the operation of bit and word shift registers
- Interpret and develop programs that use shift registers

This chapter explains how the PLC sequencer and shift register functions operate and how they can be applied to control problems. The sequencer instruction evolved from the mechanical drum switch, and it can handle complex sequencing control problems more easily than does the drum switch. Shift registers are often used to track parts on automated manufacturing lines by shifting either status or values through data files.

OceanofPDF.com

12.1 Mechanical Sequencers

Sequencer instructions are designed to operate much like the mechanical rotating cam limit switch shown in Figure 12-1. These mechanical type sequencers are often referred to as drum switches, rotary switches, stepper switches, or cam switches. They are often used to control machinery that has a repetitive cycle of operation.

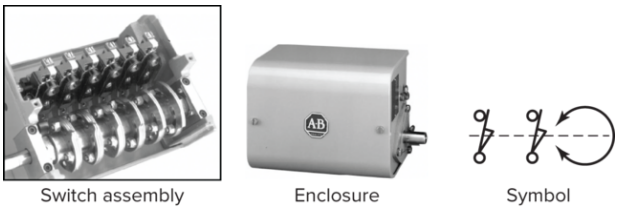


Figure 12-1 Rotating cam limit switch.

Source: Image Courtesy of Rockwell Automation, Inc.

Figure 12-2 illustrates the operation of a cam-operated **sequencer switch.** An electric motor is used to drive the cams. A series of leaf-spring mounted contacts interacts with the cam so that in different degrees of rotation of the cam, various contacts are closed and opened to energize and de-energize various electrical devices. As the cams rotate, load devices connected to the contacts can change from an on to an off state, from an off to an on state, or remain at the same state.

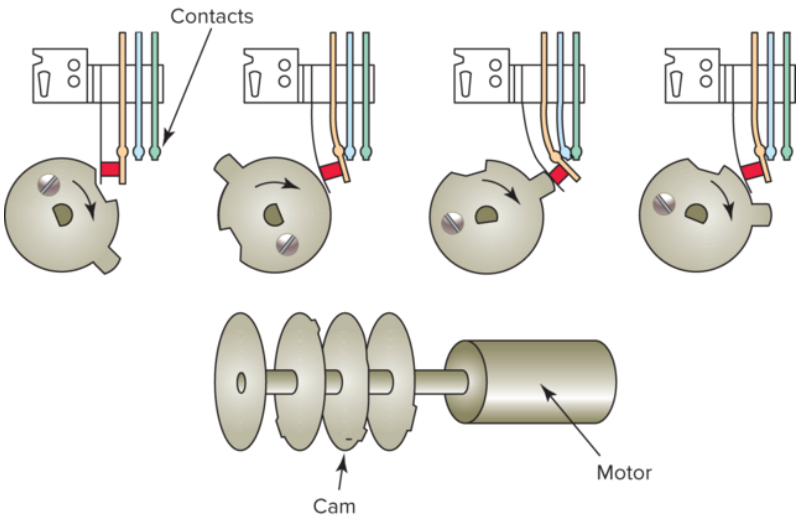


Figure 12-2 Mechanical cam-operated sequencer.

Figure 12-3 illustrates a typical mechanical drum-operated sequencer switch. The switch consists of a series of normally open contact blocks that are operated by pegs located on the motor-driven drum. The operation of this sequencer can be summarized as follows:

- Pegs are placed at specific locations around the circumference of the drum to operate the contact blocks.
- When the drum is rotated, contacts that align with the pegs will close, whereas the contacts where there are no pegs will remain open.
- The presence of a peg can be interpreted as logic 1, or on, and the absence of a peg as logic 0, or off.
- The equivalent sequencer data table illustrates the logic state for the first four steps of the drum cylinder.
- Each location where there was a peg is represented by a 1 (on), and the positions where there were no pegs are each represented by a 0 (off).

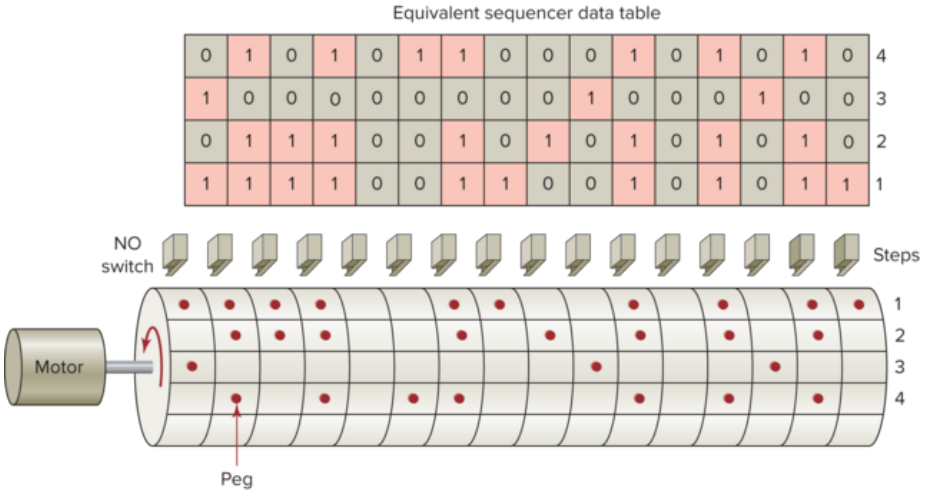


Figure 12-3 Mechanical drum-operated sequencer switch.

Sequencer switches are useful whenever a repeatable operating pattern is required. One example is the timed sequencer switch used in dishwashers to pilot the machinery through a wash cycle (**Figure 12-4**). The cycle is always the same with a fixed routine of actions at each step for a specific time to complete its specified task. The domestic washing machine is another example of the use of a sequencer, as are dryers and similar time-clock controlled devices.

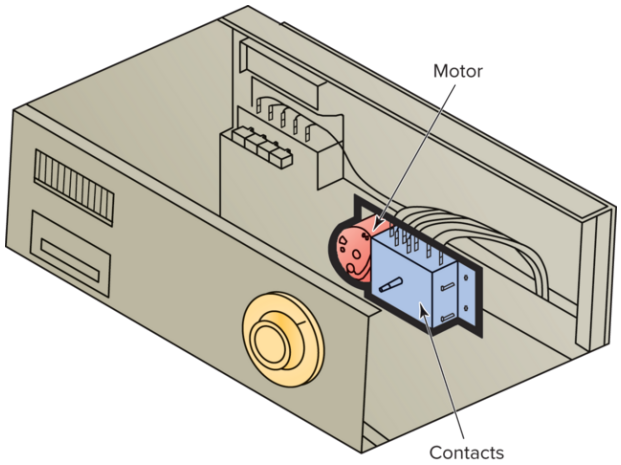
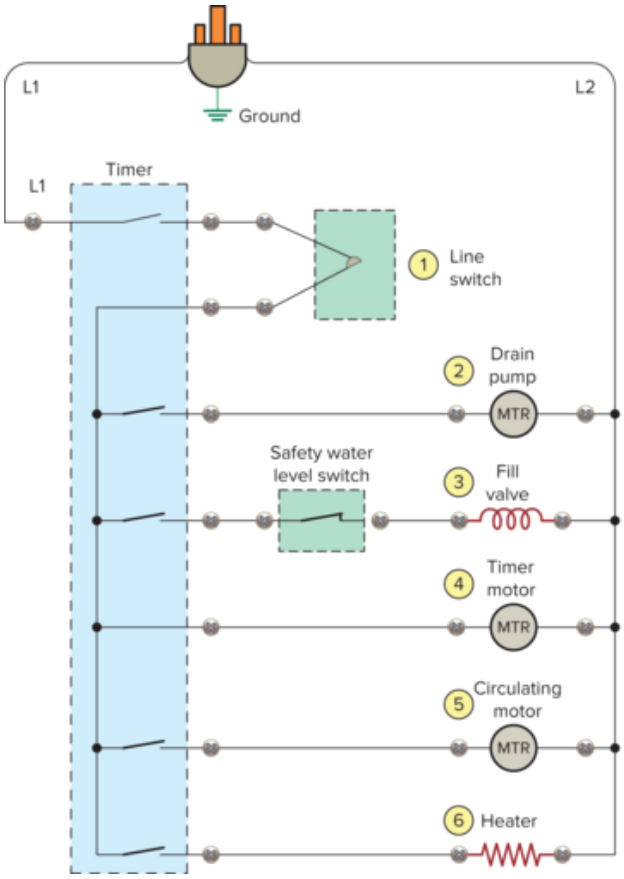


Figure 12-4 Dishwasher timed sequencer switch.

An example of the wiring and timing chart for a dishwasher that uses a cam-operated sequencer, commonly known as the timer, is shown in **Figure 12-5**. A synchronous motor drives a mechanical train that, in turn, drives a series of cam wheels. The operation of this sequencer can be summarized as follows:

- The timer motor operates continuously throughout the cycle of operation.
- The cam advances in time increments of 45 seconds in duration.
- The data timing chart shows the sequence of operation of the timer.
- A total of sixty 45 second steps are used to complete the 45 minute operating cycle.
- Numbers in the active devices column refer to control devices active during each step of the cycle.



Machine function		Timer increment	Active devices
Off		0–1	
First prerinse	Drain	2	1 2 4
	Fill	3	1 3 4 5
	Rinse	4–5	1 4 5 6
	Drain	6	1 2 4 5
Prewash	Fill	7	1 3 4 5
	Wash	8–10	1 4 5 6
	Drain	11	1 2 4 5
Second prerinse	Fill	12	1 3 4 5
	Rinse	13–15	1 4 5 6
	Drain	16	1 2 4
Wash	Fill	17	1 3 4
	Wash	18–30	1 4 5 6
	Drain	31	1 2 4 5
First rinse	Fill	32	1 3 4 5
	Rinse	33–34	1 4 5 6
	Drain	35	1 2 4 5
Second rinse	Fill	36	1 3 4 5
	Rinse	37–41	1 4 5 6
	Drain	42	1 2 4 5
Dry	Dry	43–58	1 4 6
	Drain	59	1 2 4 6
	Dry	60	1 4 6

Figure 12-5 Dishwasher wiring diagram and timing chart.

OceanofPDF.com

12.2 Sequencer Instructions

PLC **sequencer instructions** replace the mechanical drum sequencer that is used to control machines that have a stepped sequence of repeatable operations. Programmed sequencers can perform the same specific on or off patterns of outputs that are continuously repeated with a drum switch, but with much more flexibility. Sequencer instructions simplify your ladder program by allowing you to use a single instruction or pair of instructions to perform complex operations. For example, the on/off operation of 16 discrete outputs can be controlled, using a sequencer instruction, with only one ladder rung. By contrast, the equivalent contact-coil ladder control arrangement would need 16 rungs in the program.

Depending on the PLC manufacturer, various sequencer instructions can be programmed.

Figure 12-6 shows the **Sequencer** menu tab for the RSLogix 500 software. Sequencer commands may include the following:

SQO (Sequencer Output)—Is an output instruction that uses a file to control various output devices.

SQI (Sequencer Input)—Is an input instruction that compares bits from an input file to corresponding bits from a source address. The instruction is true if all pairs of bits are the same.

SQC (Sequencer Compare)—Is an output instruction that compares bits from an input source file to corresponding bits from data words in a sequence file. If all pairs of bits are the same, then a bit in the control register is set to 1.

SQL (Sequencer Load)—Is an output instruction used to capture reference conditions by manually stepping the machine through its operating sequences. It transfers data from the input source module to the sequencer file. The instruction functions much like a word-to-file transfer instruction.

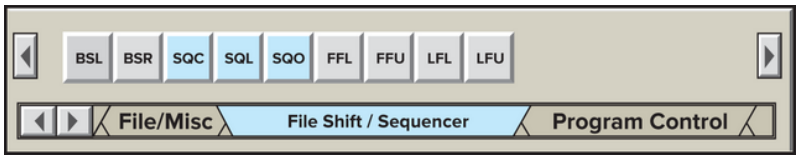


Figure 12-6 Sequencer menu tab.

Figure 12-7 shows an example of an **SQO (Sequencer Output)** instruction. Its execution is summarized as follows:

- The SQO instruction is placed on the right side of the rung as an output.
- Each time the rung makes a false-to-true transition the position is incremented to the next step in the sequencer File.
- Data from the sequencer File are then transferred through the Mask into the specified Destination address.
- The data are updated during each scan where the rung remains true.
- When the last word in the sequencer file is transferred, the done bit is set.
- Then, on the next false-to-true transition, the Destination data are cleared and the Position pointer is reset to step one.

- At start-up, when the processor is switched from program mode to the run mode the operation depends on the rung being true or false on the initial scan.
  - If true, the instruction transfers the value in step zero.
  - If false, the first rung transition from false-to-true transfers the value in step one of the instruction.
- The Mask bits must be set in order to change the value in the Destination word. The bits mask data when reset to 0 and pass data when set to 1.

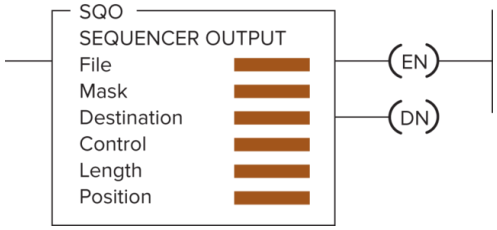


Figure 12-7 SQO (Sequencer Output) instruction.

The Sequencer Output (SQO) instruction is used to control output devices sequentially. The desired sequence operation is stored in a data file or array. As the sequencer advances through its steps, the stored data are transferred one word at a time through a Mask to the Destination. Parameters that may be required to be entered in sequencer instructions can be summarized as follows:

File—Is the starting address for the registers in the sequencer file and you must use the indexed file indicator (#) for this address. The file contains the data that will be transferred to the destination address when the instruction undergoes a false-to-true transition. Each word in the file represents a position, starting with position 0 and continuing to the file length.

Mask—Is the bit pattern through which the sequencer instruction moves source data to the destination address. Recall that in the mask bit pattern, a 1 passes values while a 0 blocks the data flow but the existing bit value remains the same. You use a mask register or file name when you want to change the mask pattern under program control. An ***h*** is placed behind the parameter to indicate that the mask is a hexadecimal number or a ***B*** is placed to indicate binary notation. Decimal notation is entered without any indicator.

Source—Is the address of the input word or file from which the SQC and SQL instruction obtains data for comparison or input to its sequencer file.

Destination—Is the address of the output word or file to which the SQO moves the data from its sequencer file.

Control—Is the address that contains the parameters with control information for the instruction. The control register stores the status byte of the instruction, the length of the sequencer file, and the instantaneous position in the file as follows:

- The **enable bit (EN;** bit 15) is set by a false-to-true rung transition and indicates that the instruction is enabled. It follows the rung condition.
- The **done bit (DN;** bit 13) is set after the last word in the sequencer file is transferred. On the next false-to-true transition of the rung with the done bit set, the position pointer is reset to 1.

- The **error bit (ER;** bit 11) is set when the processor detects a negative position value, or a negative or zero length value.

Length–Is the number of steps of the sequencer file starting at position 1. Position 0 is the start-up position. The instruction resets (wraps) to position 1 at each cycle completion. The actual file length will be 1 plus the file length entered in the instruction.

Position–Indicates the step that is desired to start the sequencer instruction. The position is the word location or step in the sequencer file from which the instruction moves data. Any value up to the file length may be entered, but the instruction will always reset to 1 on the true-to-false transition after the instruction has operated on the last position. Before we start the sequence, we need a starting point at which the sequencer is in a neutral position. The start position is all zeros, representing this neutral position; thus, all outputs will be off in position 0.

To program a sequencer, binary information is first entered into the sequencer file or register made up of a series of consecutive memory words. The sequencer file is typically a bit file that contains one bit file word representing the output action required for each step of the sequence. Data are entered for each sequencer step according to the requirements of the control application. As the sequencer advances through the steps, binary information is transferred from the sequencer file to the output word.

To illustrate the purpose and function of the sequencer file, we will examine the operation of the four-step sequence process shown in **Figure 12-8**. This sequencer is to be used to control traffic in two directions. The operation of the process can be summarized as follows:

- Six outputs are to be energized from one 16-point output module.
- Each light is controlled by one bit address of output word O:2.
- The first 6 bits are programmed to execute the following sequence of light outputs:
  - **Step 1:** Outputs O:2/0 (red) and O:2/5 (green) lights will be energized.
  - **Step 2:** Outputs O:2/0 (red) and O:2/4 (yellow) will be energized.
  - **Step 3:** Outputs O:2/2 (green) and O:2/3 (red) will be energized.
  - **Step 4:** Outputs O:2/1 (yellow) and O:2/3 (red) will be energized.
- Words B3:0, B3:1, B3:2, B3:3, and B3:4 make up the sequencer file.
- Binary information (1s and 0s) that reflects the desired on or off light status for each of the four steps is entered into each word of the sequencer file.
- Before starting the sequence, you need a starting point where the sequencer is in a neutral position. This is provided by the start position which is all zeros.

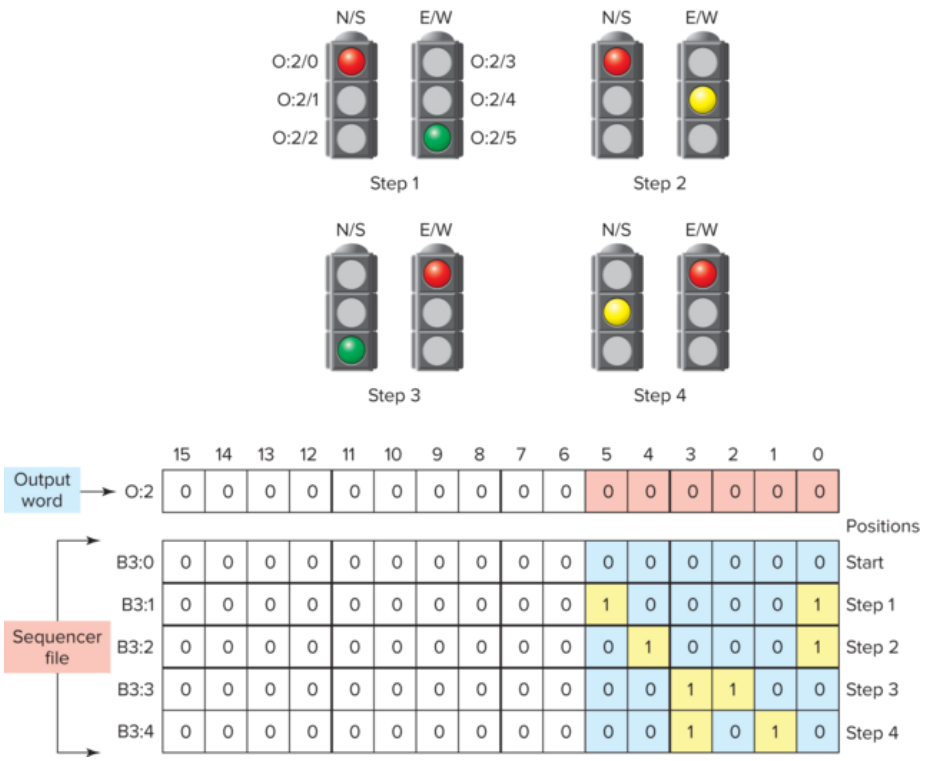


Figure 12-8 Four-step sequencer.

Due to the way in which the sequencer instruction operates, all output points must be on a single output module. When a sequencer operates on an entire output word, there may be outputs associated with the word that do *not* need to be controlled by the sequencer. In our example, bits 6 through 15 of output word O:2 are not used by the sequencer but could be used elsewhere in the program. To prevent the sequencer from controlling these bits of the output word, a mask word is used. The use of a mask word is illustrated in **Figure 12-9**. The operation of the mask can be summarized as follows:

- The **mask word** selectively screens out data from the sequencer word file to the output word.
- The hex number 003Fh is entered as the mask parameter.
- For each bit of output word O:2 that the sequencer is to control, the corresponding bit of the mask word must be set to 1.

- The arrows in the figure indicate the unmasked bits that are passed through the mask and into the destination address.
- The dashes in the bits of the designation address indicate that those bits remain unchanged in the designation location during the sequencing.
- These unchanged bits therefore can be used independently of the sequencer.

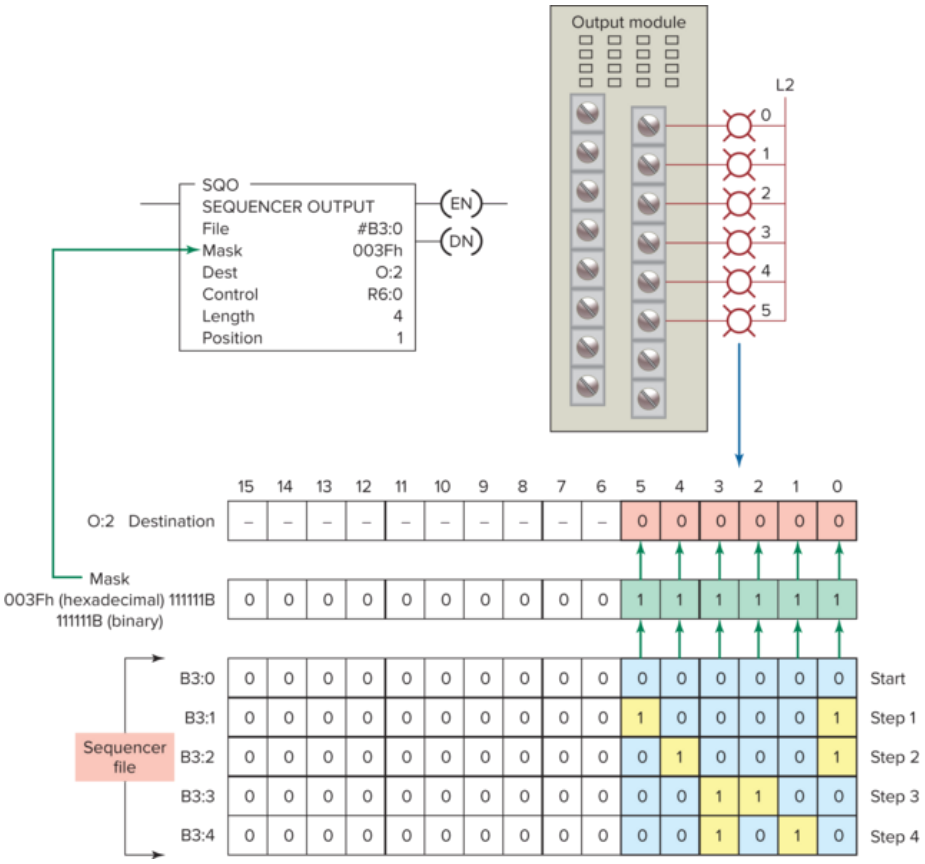


Figure 12-9 Sequencer moving data through a mask word.

The sequencer output instruction requires preceding logic on the rung where it is located. When this logic goes from false to true, it triggers the sequencer to perform its functions. Only when the logic preceding the sequencer instruction makes the transition from false to true will it go through its functions of reading the data file, applying the mask, and transferring the masked data file to the output destination. After this cycle, it waits for another false-to-true occurrence of the preceding logic to increment to the next step.

Figure 12-10 illustrates how the sequencer moves data from a file to an output. The operation of the logic rung can be summarized as follows:

- Pushbutton PB is used to send false-to-true trigger signals to the sequencer output instruction.
- The position of the sequencer instruction is incremented by one for each false-to-true transition of the sequencer rung.
- Whenever PB is momentarily closed, the sequencer is both enabled and advanced to the next position.
- When the sequencer is at step 1, the binary information in word B3:1 (100001) of the sequencer file is transferred into word O:2 of the output.
- As a result output O:2/0 and O:2/5 will be on and all the rest will be off.
- Advancing the sequencer to step 2 will transfer the data from word B3:2 (010001) into word O:2.

- As a result output O:2/0 and O:2/4 will be on and all the rest will be off.
- Advancing the sequencer to step 3 will transfer the data from word B3:3 (001100) into word O:2.
- As a result output O:2/2 and O:2/3 will be on and all the rest will be off.
- Advancing the sequencer to step 4 will transfer the data from word B3:4 (001010) into word O:2.
- As a result output O:2/1 and O:2/3 will be on and all the rest will be off.

Page 243

- When the position parameter reaches 4 (the value in the length parameter), all words would have been moved so the DN (done bit) in the instruction is set to 1.
- On the next false-to-true transition of the rung, with done bit set, the position pointer is automatically reset to 1.

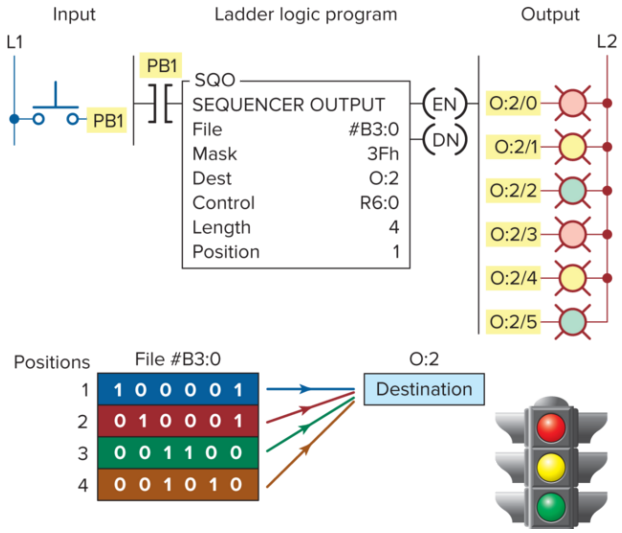


Figure 12-10 Sequencer moving data from a file to an output.

Sequencer instructions are usually retentive, and there can be an upper limit to the number of external outputs and steps that can be operated on by a single instruction. Many sequencer instructions reset the sequencer automatically to step 1 on completion of the last sequence step. Other instructions provide an individual reset control line or a combination of both.

Page 244

The RSLogix 5000 Sequencer Output (SQO) instruction, shown in **Figure 12-11**, operates similarly to that of the RSLogix 500 Sequencer Output instruction. The six parameters entered into this instruction are:

- **Array**—An array tag of the type DINT is the first entry you need to make. This is a word-level tag that defines the starting word of the sequencer data array. The desired output conditions for each step are manually entered into the array in the tag editor. The RSLogix 5000 processor puts the radix and the # sign in front of the value to indicate the radix of the displayed number. For example, 16# (hexadecimal) or 2# (binary).
- **Mask**—The Mask works exactly like the mask in the MVM instruction. It can be a word level tag or a hexadecimal program constant. When the SQO transfers 32-bits of data to an output word, there might be outputs associated with the word that do not need to be controlled by the SQO. By

masking these bits the SQO will not control them and they could be used for other purposes in the program.

- **Dest–**This is the word level data type DINT tag where the data from the instruction will be sent.
- **Control–**The tag of data type Control contains the control structure for the instruction. The Control tag has several bits that can be used: enable (EN), done (DN), error (ER), and so on. The control element also contains the Length of the sequencer (number of steps) and the current Position (step in the sequence).
- **Length–**The length (LEN) parameter stores the value that defines the number of steps the sequencer should make. It also defines the number of words required in the sequencer array. Position 0 is the start-up position. The first time the SQO instruction is enabled it moves from position 0 to position 1 when the instruction is toggled. The instruction resets to position 1 at the end of the last step. The array size must be at least one element larger than the size of the length.
- **Position–**The Position (POS) parameter stores the current step of the sequencer. Steps are numbered starting at zero.

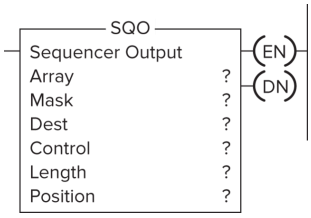
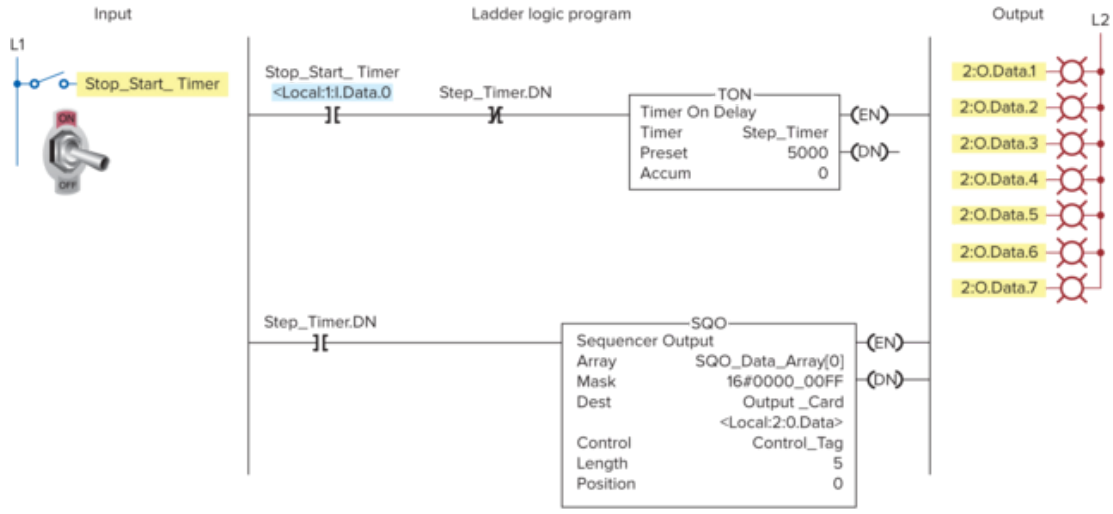


Figure 12-11 RSLogix 5000 Sequencer Output (SQO) instruction.

Figure 12-12 shows an example of an RSLogix 5000 time-driven sequencer output program. The program is designed to execute a sequential process involving five steps which control several outputs. The outputs which are on for each step are shown in the table. The operation of the program can be summarized as follows:

- The array parameter SQO_Data_Array contains the desired output states for each step.
- The output states for each step are entered at the starting location SQO_Data_Array[0].
- In this application none of the outputs are energized in step 0. When the SQO instruction executes, it will be in step 0 on initial start-up.
- When the SQO instruction advances to step 5, it will return to step 1 and continue from there.
- The mask has a constant hexadecimal value of 0000_00FF which is the same as 0000 0000 0000 0000 0000 0000 1111 1111 binary bits. Each bit corresponds to one output in the SQO instruction.
- In this example the two hexadecimal Fs represent 8 binary 1s in memory. A value of 1 in the mask allows the output state from the step to be sent to the Dest.
- The DN (done) bit of the 5-second TON Step_Timer is used to trigger the SQO instruction.
- Every time the timer ACC value reaches 5 seconds, the timer DN bit changes state causing the SQO instruction to increment the Position number of the Control tag and move to the next step.

- Note that the timer DN bit also resets the ACC value of the timer to 0 and the timer starts timing to 5 seconds again.



Output states entered into sequencer data array

[–] SQO_Data_Array	{...}
[+] SQO_Data_Array[0]	2#0000_0000_0000_0000_0000_0000_0000_0000
[+] SQO_Data_Array[1]	2#0000_0000_0000_0000_0000_0000_0000_0010
[+] SQO_Data_Array[2]	2#0000_0000_0000_0000_0000_0000_0000_0110
[+] SQO_Data_Array[3]	2#0000_0000_0000_0000_0000_0000_0001_0000
[+] SQO_Data_Array[4]	2#0000_0000_0000_0000_0000_0000_1100_1000
[+] SQO_Data_Array[5]	2#0000_0000_0000_0000_0000_0000_1111_0000

Outputs which are On for each step

Outputs								Step
7	6	5	4	3	2	1	0	
								0
						On		1
					On	On		2
			On					3
On	On			On				4
On	On	On	On					5

Figure 12-12 RSLogix 5000 time-driven sequencer output program.

OceanofPDF.com

12.3 Sequencer Programs

A sequencer program can be *event-driven* or *time-driven.* An event-driven sequencer operates similarly to a mechanical stepper switch that increments by one step for each pulse applied to it. A **time-driven** sequencer operates similarly to a mechanical drum switch that increments automatically after a preset time period.

A sequencer chart, such as the one shown in Figure 12-13, is a table that lists the sequence of operation of the outputs controlled by the sequencer instruction. These tables use a ***matrix-style*** chart format. A matrix is a two-dimensional, rectangular array of quantities. A time-driven sequencer chart usually indicates outputs on its horizontal axis and the time duration on its vertical axis. An event-driven sequencer indicates outputs on its horizontal axis and the input, or event, on its vertical axis.

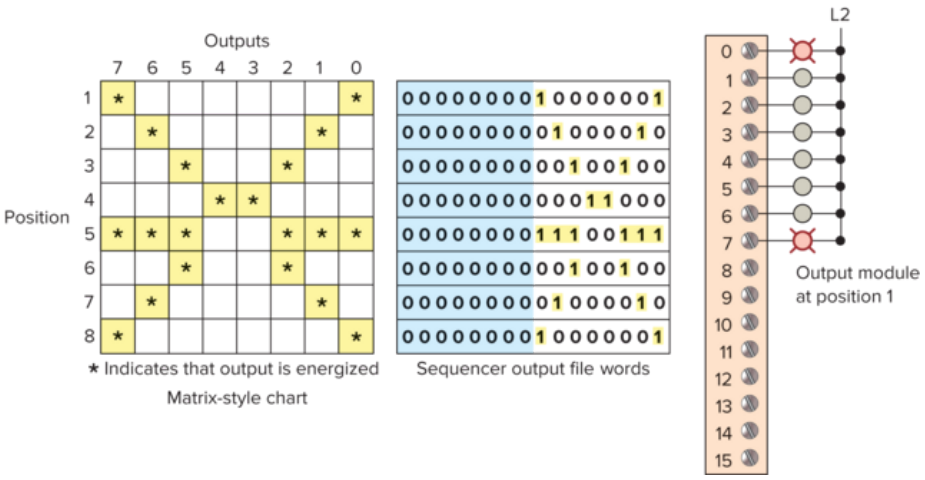


Figure 12-13 Sequencer chart.

An example of a time-driven sequencer with timed steps that are not all the same is shown in Figure 12-14. This sequencer program is used for automatic traffic light control at a four-way intersection. Output lights operate in a sequential fashion with variably timed steps. The system requires two SQO instructions: one for the light outputs and the other for the timed steps. Both SQOs have R6:0 for the control and 4 for the length. The first position is on for 25 seconds, the second for 5 seconds, the third for 25 seconds, and the fourth for 5 seconds.

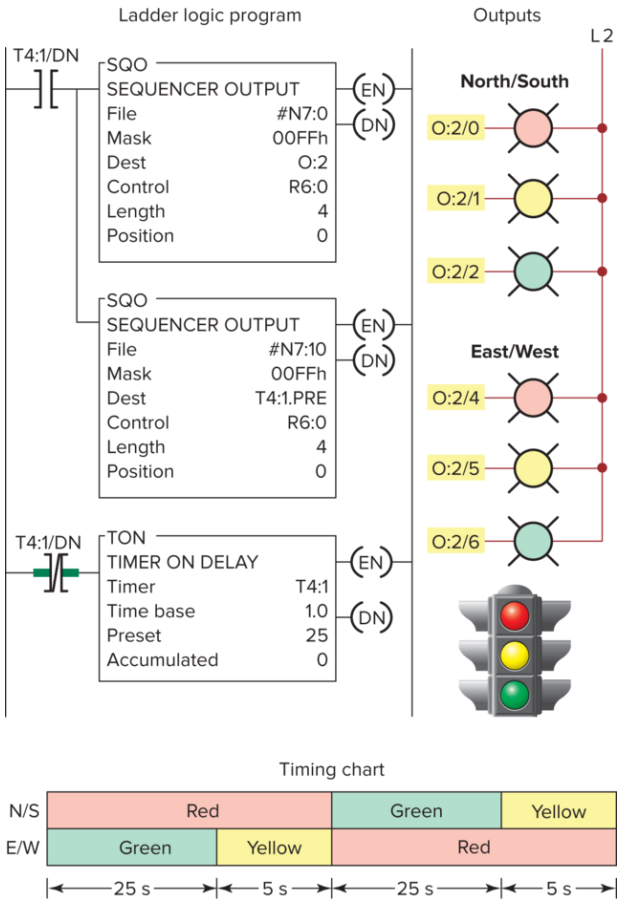


Figure 12-14 Time-driven sequencer output program.

The operation of the time-driven sequencer program can be summarized as follows:

- The bits controlling the traffic light outputs are stored in integer file #N7:0 of the first SQO instruction. The settings for the output bits for each position are entered and stored in binary table format as shown in **Figure 12-15**. Each word of the #N7:0 file is moved from the file by the program to the destination output word O:2 as previously described.
- The second SQO instruction sequencer file, #N7:10, contains the stored preset timer values 25, 5, 25, 5 seconds. These settings are stored in words N7:11, N7:12, N7:13, and N7:14 as illustrated in **Figure 12-16**. Each word of the #N7:10 file is moved by the program to the destination address T4:1.PRE, which is the preset value for the timer. The program moves information from this file to timer T4:1's preset. The mask allows the proper data to pass and blocks the unnecessary data.
- The timer cycles the two SQO instructions through their four states.
- Since both of the SQO instructions have R6:0 for control and 4 for length, they are stepped in unison to provide a sequentially timed output.

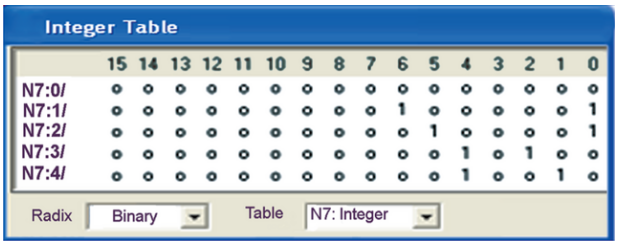


Figure 12-15 Sequencer file #N7:0 light cycle settings.

Source: Courtesy of TheLearningPit.

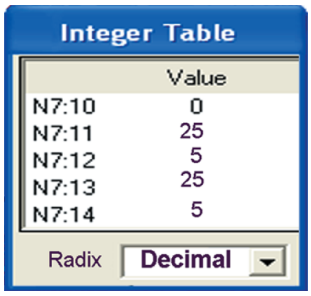
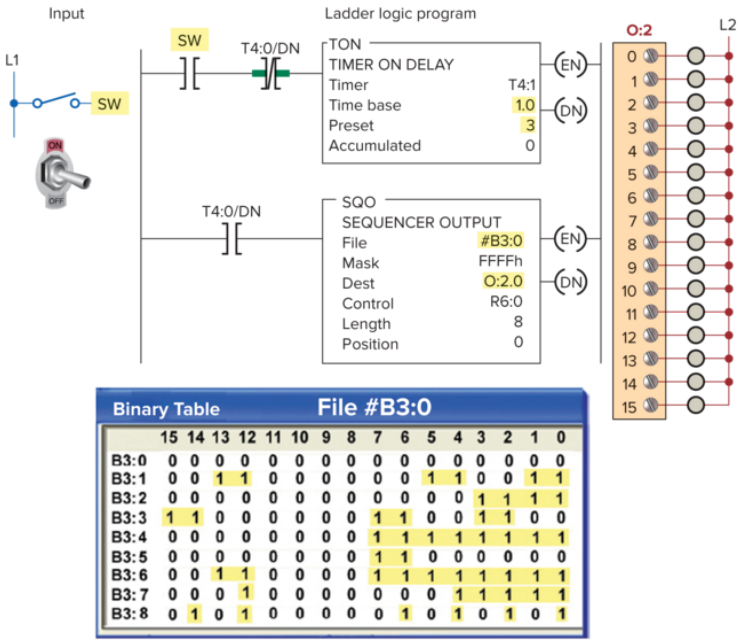


Figure 12-16 Sequencer file #N7:10 timer settings.

Source: Courtesy of TheLearningPit.

An example of a time-driven sequencer program in which the time interval between sequencer steps is always a constant set value is shown in **Figure 12-17**. The operation of the program can be summarized as follows:

- The preset time of timer T4:1 is set for 3 seconds.
- The settings of the output bits for each sequencer position are entered and stored in bit file #B3:0.
- The timer is started by the closing switch SW and 3 seconds later the timer done bit is set to 1.
- As a result the timer done bit increments the SQO instruction to the next position and resets the timer.
- The destination is O:2 and all 16 bits of this word are used for outputs.
- The mask is FFFF hexadecimal or 1111111111111111 binary, which allows all 16 bits to pass through.
- As long as input SW is closed the program continues operating with 3 seconds between sequencer steps.



	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
B3:0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
B3:1	0	0	1	1	0	0	0	0	0	0	1	1	0	0	1	1
B3:2	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1
B3:3	1	1	0	0	0	0	0	0	1	1	0	0	1	1	0	0
B3:4	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
B3:5	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0
B3:6	0	0	1	1	0	0	0	0	1	1	1	1	1	1	1	1
B3:7	0	0	0	1	0	0	0	0	0	0	0	1	1	1	1	1
B3:8	0	1	0	1	0	0	0	0	0	1	0	1	0	1	0	1

Figure 12-17 Time-driven sequencer with constant time interval between steps.

Source: Courtesy of TheLearningPit.

With an event-driven sequencer, the SQO instruction advances to the next step by an external pulsed input event rather than a preset time. An example of an event-driven sequencer is shown in **Figure 12-18**. The operation of the program can be summarized as follows:

- The sequencer SQO instruction uses two OR configured sensor switches (S1 and S2).
- Any one of the two parallel paths can make the SQO rung true.
- As each event occurs, that OR branch makes a false-to-true transition advancing the sequencer position.
- Data are copied from file #B3:0 at the bit locations through mask word, F0FF hex or 1111000011111111 binary, to the destination O:2. Mask bits are set to 1 to pass data and reset to 0 to mask data.
- Once the position reaches the last position on the false-to-true transition of the instruction the position will reset to 1.
- Note that the data in O:2 match the data in position 2 in the file, except for the data in bits 8 through 11.
- Bits 8 through 11 may be controlled from elsewhere in the program; they are not affected by the sequencer instruction because of the 0 in these bit positions in the mask.

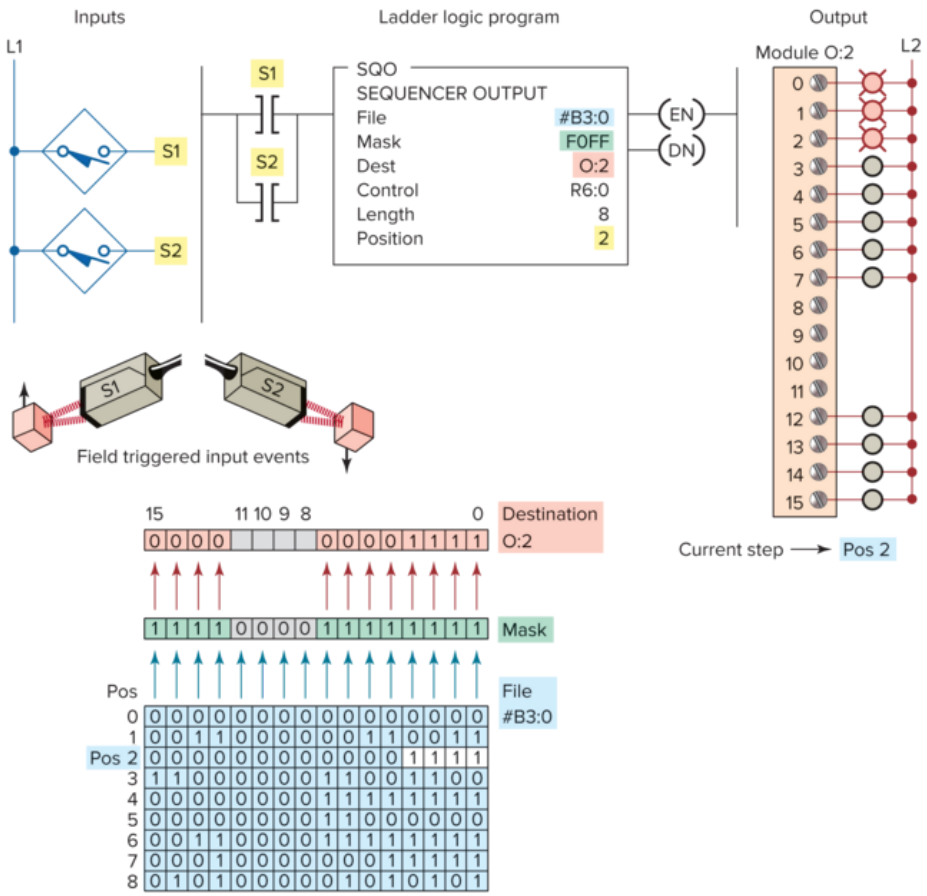


Figure 12-18 Event-driven sequencer output program.

The **sequencer compare (SQC)** instruction is an output instruction used to compare bits from an input source file to corresponding bits from data words in a sequencer file. When the pairs of bits are the same, then the found (FD) bit in the control register is set to 1. This instruction can be used to compare the status of a machine's input devices with what is required for normal operation. When the status of the input devices on the machine (on or off) are identical to the data stored in the sequencer file, the control found bit is set to 1.

Page 248

An example of an RSLogix 500 sequencer compare (SQC) instruction program is shown in **Figure 12-19**. The operation of the program can be summarized as follows:

- The data in the highest 4 bits of the source (I:1) are compared to the data in file #B3:22.
- In this example, the highest 4 bits in I:1 match the status of the highest 4 bits in B3:25 at step position 3.
- If the pushbutton input I:1/0 is true at this point, the found (FD) bit is set, which turns output PL1 on.
- Whenever the combination of opened and closed switches connected to I:1/12, I:1/13, I:1/14, and I:1/15 is equal to the combination of 1s and 0s on a step in the sequencer reference file and the input I:1/0 is true, the PL1 output will become energized.

- The mask (F000h) allows unused bits of the sequencer instruction to be used independently. In this example, unused bit I:1/0 is used for the conditional input of the sequencer compare rung.

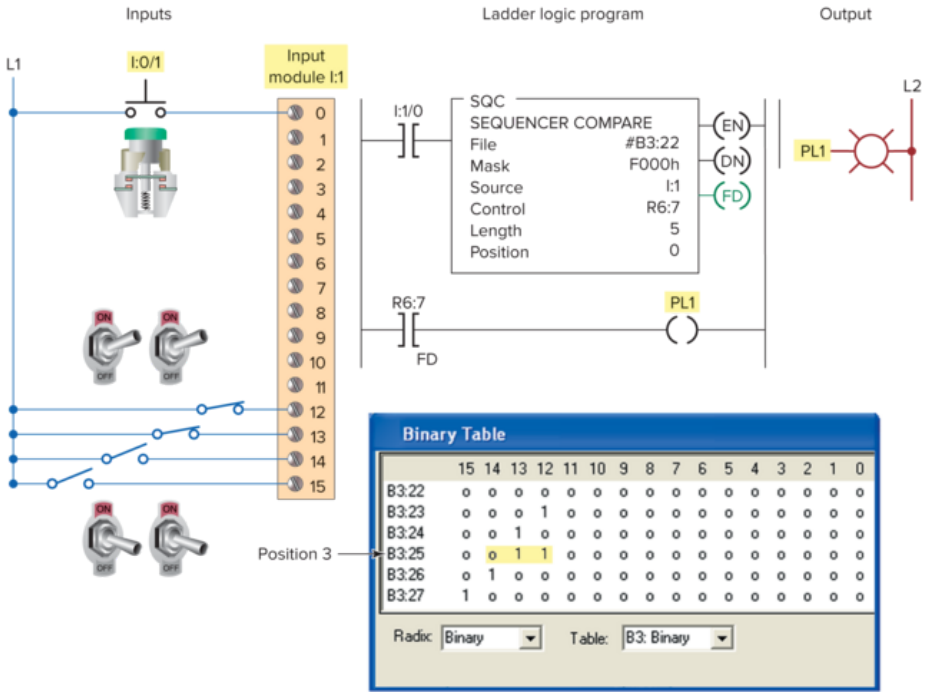


Figure 12-19 Sequencer compare (SQC) instruction program.

Source: Courtesy of TheLearningPit.

The *sequencer load (SQL)* instruction is used to read the PLC input module and store the input data in the sequencer file. Loading input conditions for a large number of process steps is prone to errors. In such instances the sequencer load instruction can be used to load data into a sequencer file one step at a time. For example, a robot may be jogged manually through its sequence of operation, with its input devices read at each step. At each step, the status of the input devices is written to the data file in the sequencer compare instruction. As a result, the file is loaded with the desired input status at each step, and these data are then used for comparison with the input devices when the machine is run in automatic mode.

An example of an RSLogix 500 sequencer load (SQL) instruction program is shown in **Figure 12-20**. The operation of the program can be summarized as follows:

- The sequencer load instruction is used to load the file and *does not* function during the machine's normal operation.
- It replaces the manual loading of data into the file with the programming terminal.
- The sequencer load instruction *does not* use a mask. It copies data directly from the source address to the sequencer file.
- When the instruction goes from false to true, the instruction indexes to the next position and copies the data.

- When the instruction has operated on the last position and has a false-to-true transition, it resets to position 1.
- It transfers data in position 0 only if it is at position 0 and the instruction is true and the processor goes from the program to run mode.
- By manually jogging the machine through its cycle, the switches connected to input I:2 of the source can be read at each position and written into the file by momentarily pressing PB1. Otherwise, the data would have to be entered into the file manually.

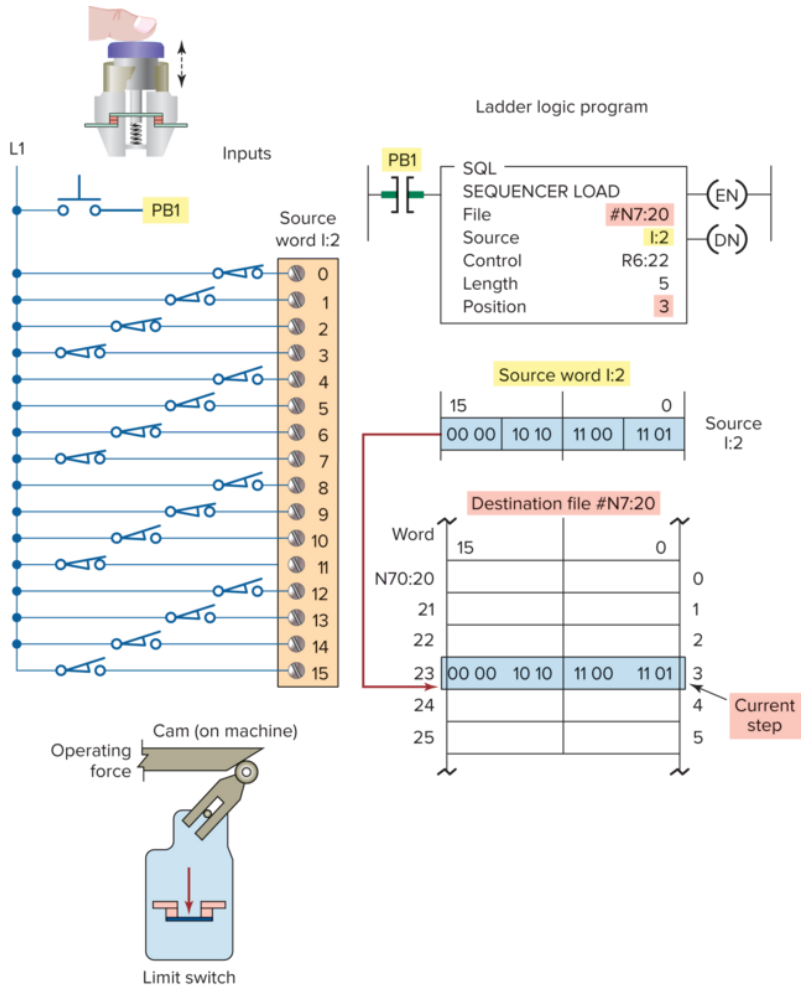


Figure 12-20 Sequencer load (SQL) instruction program.

OceanofPDF.com

12.4 Bit Shift Registers

The PLC not only uses a fixed pattern of register (word) bits, but also can easily manipulate and change individual bits. A bit ***shift register*** is a register that allows the shifting of bits through a single register or group of registers. The bit shift register shifts bits serially (from bit to bit) through an array in an orderly fashion.

A shift register can be used to simulate the movement, or *track* the flow, of parts and information. We use the shift register whenever we need to store the status of an event so that we can act on it at a later time. Shift registers can shift either status or values through data files. Common applications for shift registers include the following:

- Tracking of parts through an assembly line
- Controlling of machine or process operations
- Inventory control
- System diagnostics

Figure 12-21 illustrates the basic concept of a shift register. A shift pulse or clock causes each bit in the shift register to move 1 position to the right. At some point, the number of data bits fed into the shift register will exceed the register's storage capacity. When this happens, the first data bits fed into the shift register by the shift pulse are lost at the end of the shift register. Typically, data in the shift register could represent the following:

- Part types, quality, and size
- The presence or absence of parts
- The order in which events occur
- Identification numbers or locations
- A fault condition that caused a shutdown

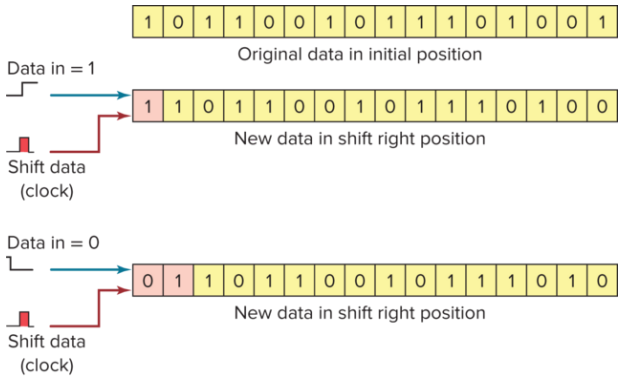


Figure 12-21 Basic concept of a shift register.

You can program a shift register to shift status data either right or left, as illustrated in **Figure 12-22**, by shifting either status or values through data files. When you want to track parts on

a status basis, use bit shift registers. Bit shift instructions will shift bit status from a source bit address, through a data file, and out to an unload bit, one bit at a time. There are two bit shift instructions: ***bit shift left*** *(BSL),* which shifts bit status from a lower address number to a higher address number through a data file, and ***bit shift right (BSR),*** which shifts data from a higher address number to a lower address number through a data file. Some PLCs provide a ***circulating shift register*** function, which allows you to repeat a pattern again and again.

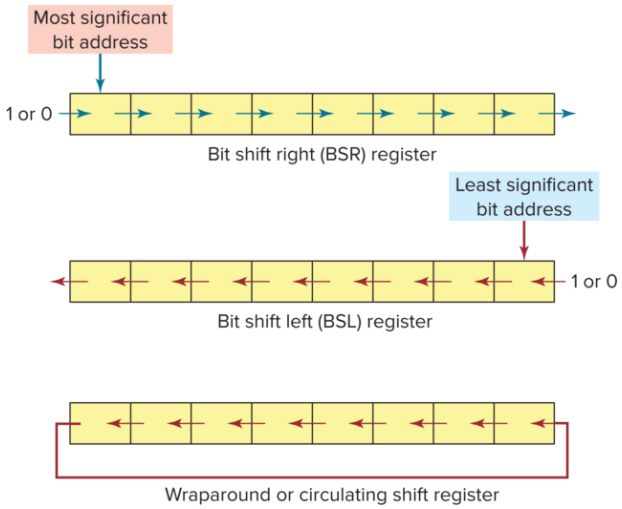


Figure 12-22 Types of shift registers.

When working with a bit shift register, you can identify each bit by its position in the register. Therefore, working with any bit in the register becomes a matter of identifying the position it occupies rather than the conventional word number/bit number addressing scheme.

Figure 12-23 shows the **File Shift** menu tab and BSL and BSR instruction blocks that are part of the instruction set for the Allen-Bradley RSLogix 500 controllers. The commands can be summarized as follows:

BSL (Bit Shift Left)—Loads a bit of data into a bit array, shifts the pattern of data through the array to the left, and unloads the last bit of data in the array.

BSR (Bit Shift Right)—Loads a bit of data into a bit array, shifts the pattern of data through the array to the right, and unloads the last bit of data in the array.

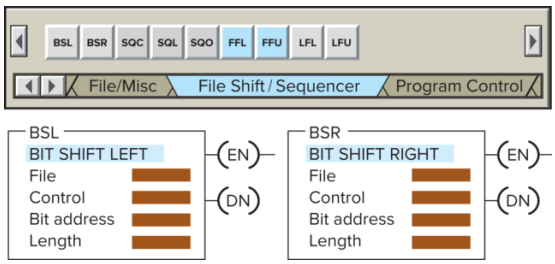


Figure 12-23 Bit shift left and bit shift right instructions.

Shift registers are useful for tracking the status or identification of a part as it moves down an assembly line. The data file used for a shift register usually is the bit file because its data are displayed in binary format, making it easier to read. BSL and BSR are output instructions that load data into a bit array one bit at a time. The data are shifted through the array, then unloaded one bit at a time.

The BSL instruction has the same operands as the BSR instruction. The difference is the direction in which the bits are indexed. A bit shift instruction will execute when its input control logic goes from false to true. To program a bit shift instruction, you need to provide the processor with the following information:

File—The address of the bit array you want to manipulate. The address must start with the # sign and at bit 0 of the first word or element. Any remaining bits in the last word of the array cannot be used elsewhere in the program because the instruction invalidates them.

Control—R data-table type. The address is unique to the instruction and cannot be used to control any other instruction. It is a three-word element that consists of the status word, the length, and the position.

Bit address—Is the address of the source bit. The instruction inserts the status of this bit in either the first (lowest) bit position (for the BSL instruction) or the last (highest) bit position (for the BSR instruction) in the array.

Length—Indicates the number of bits to be shifted, or the file length, in bits. The status bits of the control word are the enable, done, error, and unload bits. Their functions can be summarized as follows:

- **Enable Bit (EN)**—The enable bit follows the instructions status and is set to 1 when the instruction is true.
- **Done Bit (DN)**—The done bit is set to 1 when the instruction has shifted all bits in the file one position. It resets to 0 when the instruction goes false.
- **Error Bit (ER)**—The error bit is set to 1 when the instruction has detected an error, which can happen when a negative number is entered in the length.
- **Unload Bit (UL)**—The unload bit's status is controlled by shifting of the last bit of the file into the unload bit when the instruction is executed. It is the bit location into which the status from the last bit in the file shifts when the instruction goes from false to true. When the next shift occurs, these data are lost, unless additional programming is done to retain the data.

An example of a bit shift left (BSL) instruction program is shown in **Figure 12-24**. The operation of the program can be summarized as follows:

- Momentary de-actuation of limit switch LS causes the BSL instruction to execute.
- When the rung goes from false to true, the enable bit is set and the data block is shifted to the left (to a higher bit number) one bit position.
- The specified bit, at sensor bit address I:1/1, is shifted into the first bit position, B3:10/0.
- The last bit is shifted out of the array and stored in the unload bit, R6:0/UL.
- The status that was previously in the unload bit is lost.
- All the bits in the unused portion of the last word of the file are invalid and should not be used

elsewhere in the program.

- For wraparound operation, set the position of the bit address to the last bit of the array or to the UL bit, whichever applies.

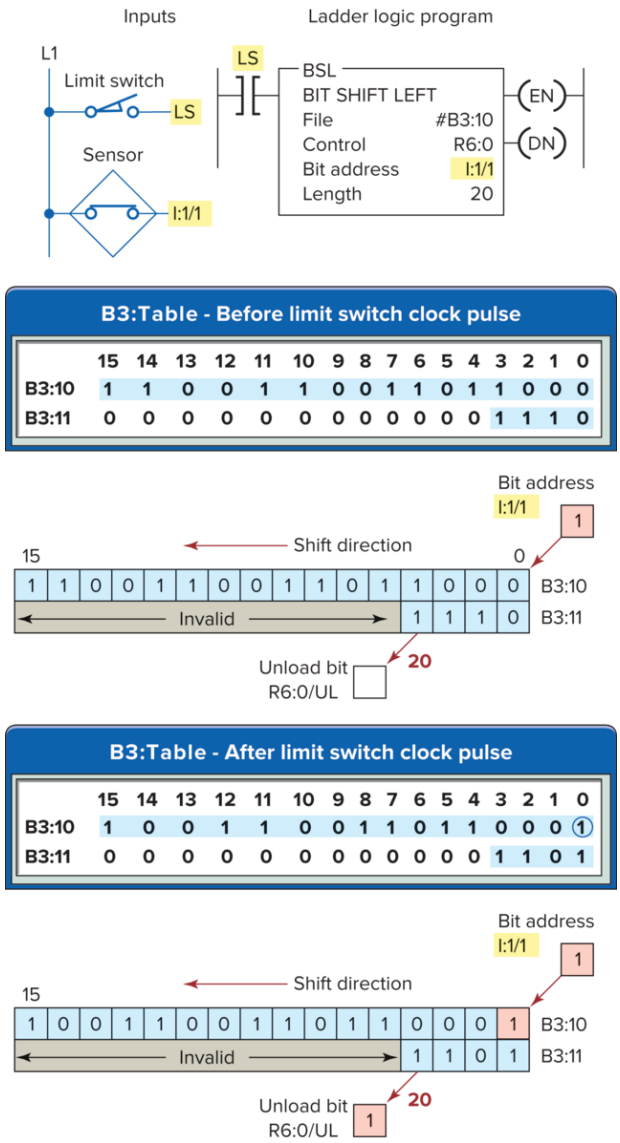


Figure 12-24 Bit shift left (BSL) instruction program.

An example of a bit shift right (BSR) instruction program is shown in **Figure 12-25**. The operation of the program can be summarized as follows:

- Before the rung goes from false to true, the status of bits in words B3:50 and B3:51 is as shown.
- The status of the bit address, I:3/5, is a 0, and the status of the unload bit, R6:1/UL, is a 1.
- When limit switch LS closes, the status of the bit address, I:3/5, is shifted into B3:51/7, which is the 24th bit in the file.
- The status of all the bits in the file is shifted one position to the right, through the length of 24 bits.

- The status of B3:50/0 is shifted to the unload bit, R6:1/UL. The status that was previously in the unload bit is lost.

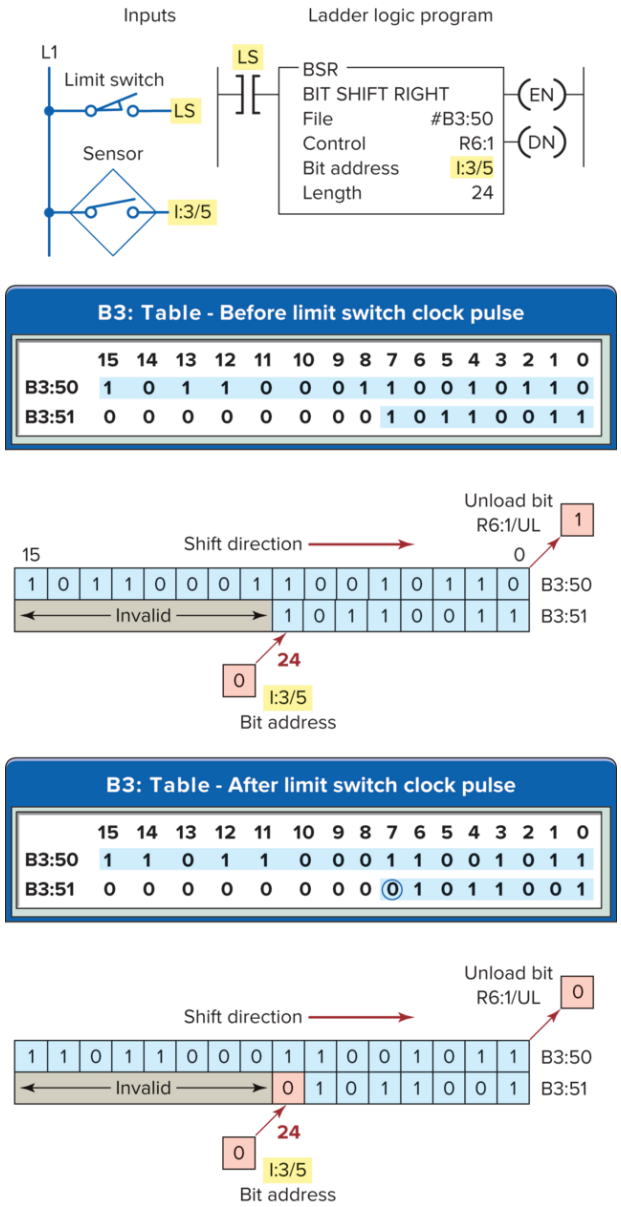


Figure 12-25 Bit shift right (BSR) instruction program.

An example of a bit BSL instruction program with wraparound operation is shown in **Figure 12-26**. The clock pulse input is a fixed regular 3 second pulse-generated on-delay timer T4:0. The operation of the program can be summarized as follows:

- Go to the data tables and set bit addresses B3:0/0, B3:0/1, B3:0/2 to logic 0 and bit address R6:0/UL to logic 1.
- When the PLC is then placed in run, bit B3:0/0 is set to logic 1 causing PL1 to turn on.

- Closing input switch SW starts timer T4:0 timing.
- After 3 seconds, the timer done bit is set to reset the timer accumulated time to zero and shift the logic bit 1 to the left to B3:0/1.
- This causes PL1 to turn off and PL2 to turn on.
- After another 3 seconds, the timer done bit is set once again.
- The BSL instruction shifts the bits to the left once more and causes PL2 to turn off and PL3 to turn on.
- The process continues with each of the pilot lights turned on in sequence for 3 seconds.

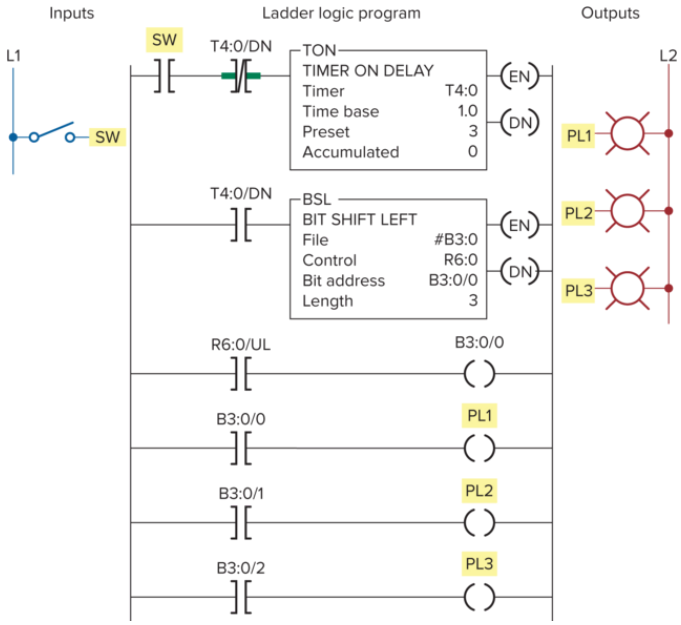


Figure 12-26 BSL instruction with a wraparound operation.

A shift register is often used in material handling processes where some form of binary information must be synchronized with a moving part on a conveyor. The binary information refers to any two conditions that can be assigned to the moving product—for example, the presence or absence of a part. As the part moves along the conveyer, some form of sensing device will determine which of these two categories the passing product falls into. **Figure 12-27** illustrates cartons traveling on a conveyor being detected by a photoelectric sensor. The sensor that drives the data line on a shift register is fixed such that the beam detects the presence or absence of a carton. A logic 1 sensor condition state can indicate the presence of a carton, and a 0 the absence.



Figure 12-27 Cartons traveling on a conveyor being detected by a photoelectric sensor.

Source: Image Courtesy Banner Engineering Corp.

The process of **Figure 12-28** illustrates a spray-painting operation controlled by a shift left register. As the parts pass along the production line, the shift register bit patterns represent the items on the conveyor hangers to be painted. Each file bit location represents a station on the line, and the status of the bit indicates whether or not a part is present at that station.

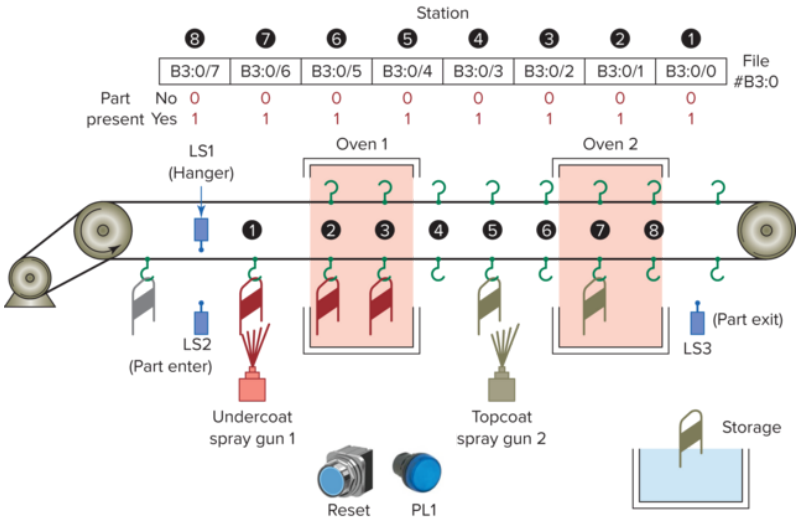


Figure 12-28 Spray-painting operation controlled by a shift left register.

The program for the spray-painting operation is shown in **Figure 12-29**. Its operation can be summarized as follows:

- Limit switch LS1 is used to detect the hanger and limit switch LS2 the part.
- The pulse generated by the hanger-operated limit switch LS1 shifts the status of the data provided by part-detection limit switch LS2.

- The logic of this operation is such that when a part to be painted and a part hanger occur together at station 1 (indicated by simultaneous closing of LS2 and by LS1), logic 1 is input into the shift register at B3:0/0.
- This causes the SOL 1 rung to be true and the undercoat spray gun to energize.
- At station 5, a 1 appears in bit B3:0/5 of the shift register to make the SOL 2 rung true and topcoat spray gun energize.

Page 254

- Logic 0 in the shift register indicates that the conveyor has no parts on it to be sprayed, and it therefore inhibits the operation of the spray guns.
- Counter C5:1 counts the parts as they enter the process and counter C5:2 as they exit.
- The count obtained by the two counters should be equal when no parts are being painted.
- Whenever the two counts are equal in value the equal instruction executes to turn on pilot light PL1. This is an indication that the parts commencing the spray-painting run equal the parts that have completed it.

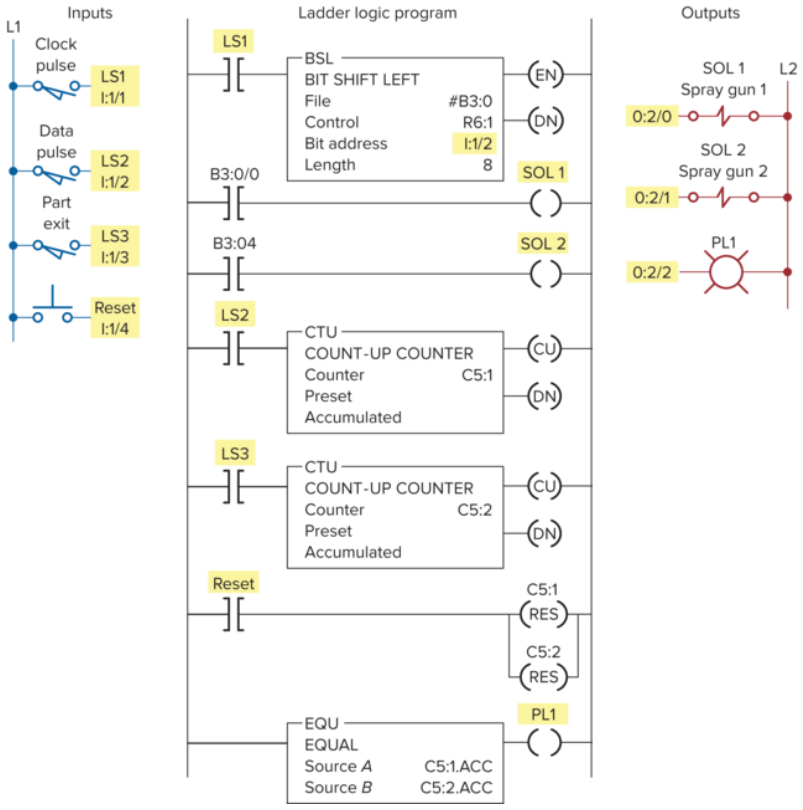


Figure 12-29 Spray-painting operation program.

An example for a bit shift program used to keep track of carriers flowing through a 16-station machine is shown in **Figure 12-30**. The operation of the program can be summarized as follows:

Page 255

- Proximity switch 1 senses a carrier, and proximity switch 2 senses a part on the carrier.
- Clock pulse generated by carrier proximity switch I:1/1 shifts the status of the data provided by part detection proximity switch I:1/2.

- When a part and carrier are sensed together, indicated by simultaneous closing of I:1/2 and I:1/1, logic 1 is input into the shift register at output O:4/0 to energize the pilot light connected to it.
- Remaining pilot lights connected to output module O:4 turn on in sequence as carriers with parts move through each station.
- They turn off or remain off as empty carriers move through.
- Station 5 is an inspection station where parts are examined.
- If the part fails, the inspectors push PB1 as they remove the part from the system, which turns output O:4/4 off.
- Rework parts can be added back into the system at station 7.
- When the operator puts a part on an empty carrier, he or she pushes PB2, turning output O:4/6 on to resume tracking.

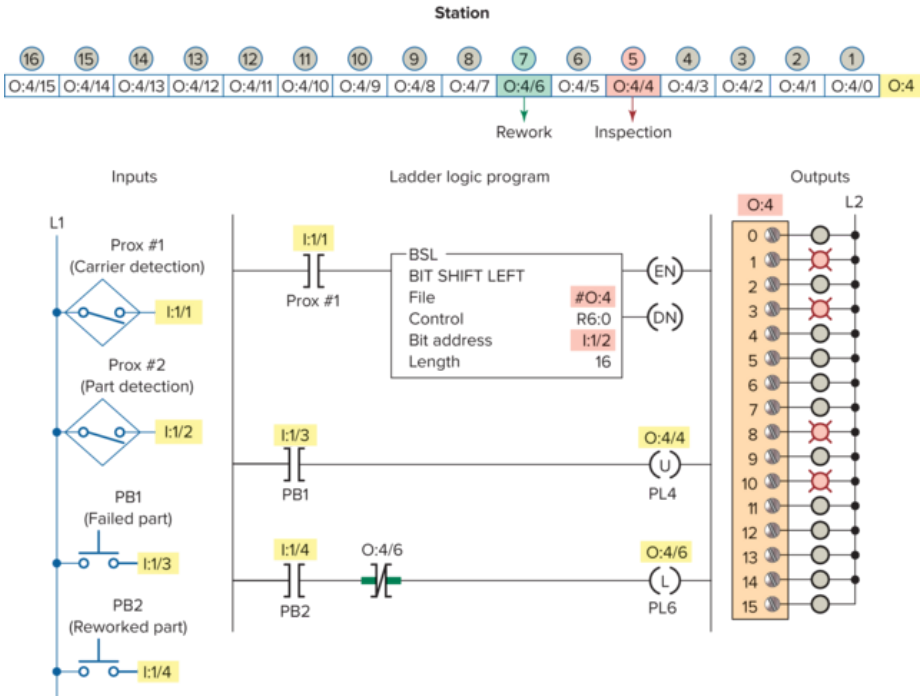


Figure 12-30 Program for tracking of carriers flowing through a 16-station machine.

Allen Bradley RSLogix 5000 has two shift register instructions: Bit Shift Left (BSL) and Bit Shift Right (BSR). The BSL instruction is shown in **Figure 12-31**. Data are loaded into, shifted through, and unloaded from a bit array, one bit at a time when the instruction transitions from false to true. Bit shift instructions contain the following parameters.

Array—A one-dimensional array of type DINT with enough bits to accommodate the length.

Control—A tag of data type Control that stores the status bits and the size (number of bits) of the bit array.

Source Bit—A tag of type BOOL, the status of which the instruction inserts into the first bit position of the bit array on every false-to-true rung transition.

Length—The size (number of bits) of the bit array.

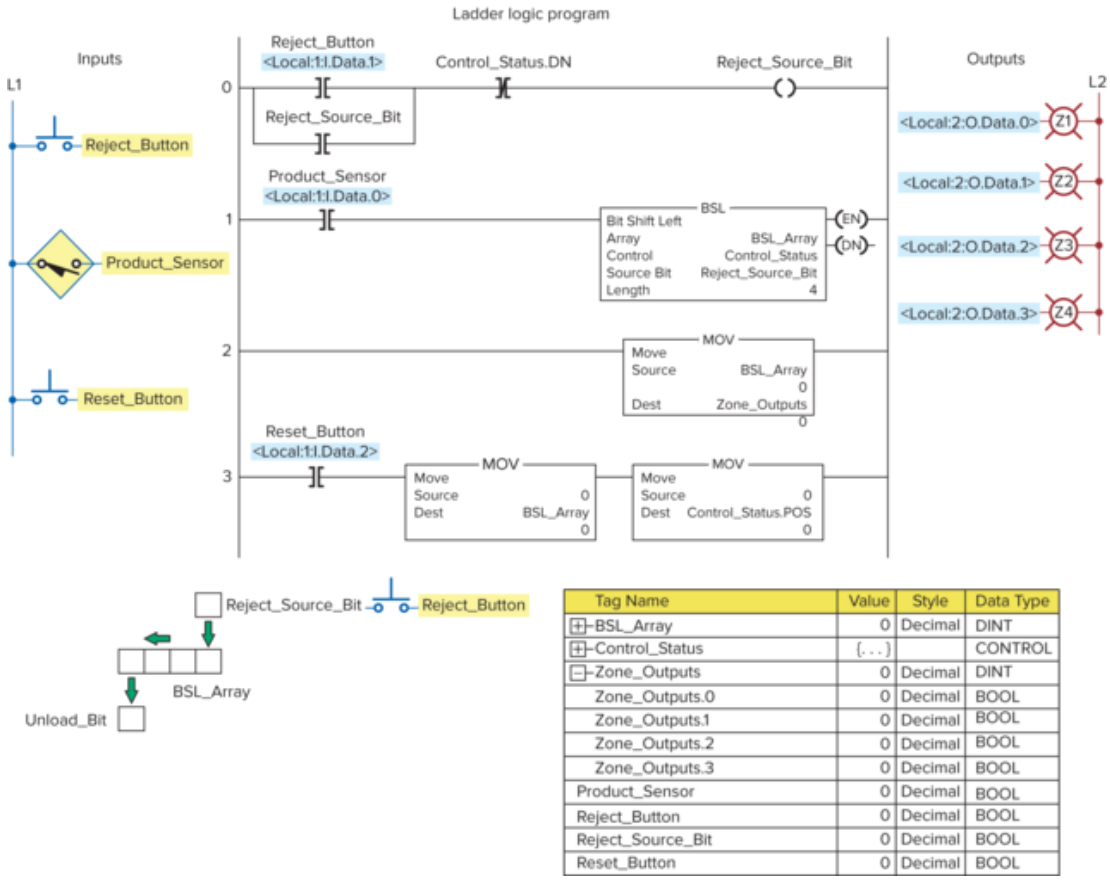


Figure 12-31 RSLogix 5000 Bit Shift Left (BSL) instruction.

Figure 12-32 illustrates an application for the RSLogix 5000 Bit Shift Left (BSL) instruction. A pass/fail inspection station is used in a conveyor system to keep track of defective products as they move down the conveyor at an equal distance. A partially completed product is moved down a conveyor to an inspector before proceeding to the final stages of manufacture. If a product is deemed defective, it is pointless to continue building it, so the part is identified as a reject. Before the part arrives at the reject bin, it must pass through three more zones of manufacture. The operation of the program can be summarized as follows:

- At the zone 1 pass/fail station an inspector examines the product before it proceeds to the final 3 stages of manufacture.
- If the product is deemed defective the inspector actuates the Reject_Button energizing the Reject_Source_Bit through its seal circuit.

- The rejected product passes by the Productor_Sensor and as a result the Reject_Source_Bit (1) is loaded into the first step of the BSL_Array which energizes the Z1 reject pilot light.
- At the same time the Control_Status.DN bit momentarily changes state to open the seal-in circuit to the Reject_Source_Bit, readying it for the next product reject.
- Each of the remaining 3 zones is equipped with a reject pilot light to warn the assembler to ignore the product if the lamp is on.
- Each successive product activating the Product_Sensor will advance the data bit one position to the left in the register, energizing the 3 remaining reject lamps in turn.
- The MOV instruction of rung 2 posts the BSL_Array contents to the four Zone_Outputs.
- Activating the Reset_Button of rung 3 at any time energizes both rung MOV instructions to reset the BSL_Array and Control_Status to 0.



Tag Name	Value	Style	Data Type
⊞ BSL_Array	0	Decimal	DINT
⊞ Control_Status	{...}		CONTROL
⊟ Zone_Outputs	0	Decimal	DINT
Zone_Outputs.0	0	Decimal	BOOL
Zone_Outputs.1	0	Decimal	BOOL
Zone_Outputs.2	0	Decimal	BOOL
Zone_Outputs.3	0	Decimal	BOOL
Product_Sensor	0	Decimal	BOOL
Reject_Button	0	Decimal	BOOL
Reject_Source_Bit	0	Decimal	BOOL
Reset_Button	0	Decimal	BOOL

Figure 12-32 RSLogix 5000 pass/fail inspection program.

OceanofPDF.com

12.5 Word Shift Operations

The ***first in, first out (FIFO)*** instructions are word shift operations that are similar to bit shift operations. Word shifting provides a simpler method of loading and unloading data into a file, usually called the *stack*. It is often used for tracking parts through an assembly line, where parts are represented by values that have a part number or an assembly code. **Figure 12-33** shows a barcode reader used for reading printed barcode data on boxes.

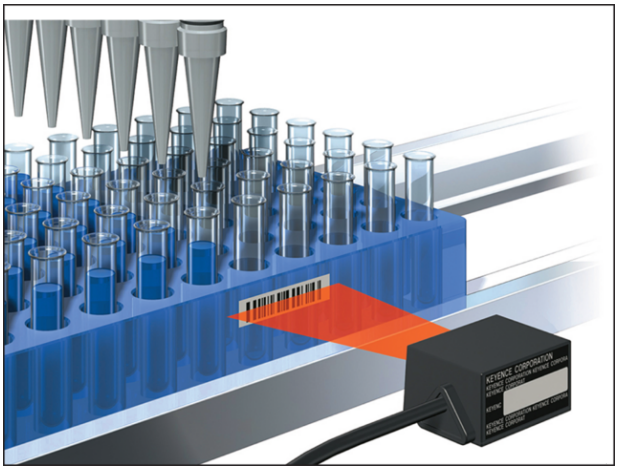


Figure 12-33 Barcode reader.

Source: Courtesy Keyence Canada Inc.

A bit shift register operates ***synchronously;*** because for every bit shifted in, one is shifted out. Data entered in a bit shift register must be shifted the length of the register (one position per shift pulse) before they are available to shift out.

A FIFO function operates ***asynchronously.*** Rather than shifting bits of information within a word it shifts the data from a complete word into a file or stack. Unlike the bit shift register, two separate shift pulses are required: one to shift data into the file (load) and one to shift data out of the file (unload). These two shift pulses operate independently (asynchronously) of each other. Data loaded in a FIFO can be immediately available for unload, regardless of length.

The FFL and FFU instruction are used in pairs. The FFL *loads* logic words into a user-created file called a FIFO stack. The FFU instruction is used to *unload* the words from the FIFO stack, in the same order as the words were entered. The first word entered is the first word out.

The RSLogix 500 FIFO load (FFL) instruction is shown in **Figure 12-34**. The parameters that are required to be entered in the instruction block are summarized as follows:

Source—Word address from which the data are entered into the FIFO file.

FIFO—Address of the file in which the data are entered. The address must start with a # sign.

Control—R data-table type and is the file address of the control structure. The status bits, stack length, and position are stored in this element.

Length—File length in words. Specifies the maximum number of words in the stack.

Position—Is the next available location where the instruction loads data into the stack. The first address in the stack is position 0. As each word is entered into the stack, the position counter, on both the FFL and FFU, will increment up by one. The stack is considered full when the position value equals the length. The status bits of the control word are the enable (EN), the done (DN), and the empty (EM) bits. Their functions can be summarized as follows:

- **Enable Bit (EN)**—The enable bit follows the instructions status and is set to 1 when the instruction is true.
- **Done Bit (DN)**—The done bit is set to 1when the instruction's position equals the length. When the done bit is set, the FIFO is full and does not accept any more data. Also the data in the FIFO file are not overwritten when the instruction goes from false to true.
- **Empty Bit (EM)**—The empty bit is set to 1 when all the data have been unloaded from the FIFO file.

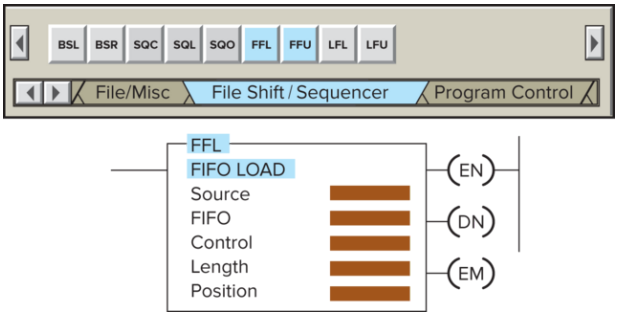


Figure 12-34 RSLogix 500 FIFO load (FFL) instruction.

Figure 12-35 shows the RSLogix 500 FIFO unload (FFU) instruction. The following parameters need to be entered in the RSLogix 500 FFU instruction:

Page 259

FIFO—Address of the file in which the data are entered. The address must start with a # sign. When *paired* with an FFL instruction, this address is the same as the address for the FFL.

Destination—Address to which the FFU unloads data.

Control—R data-table type. It is a three-word element that consists of the status word, the length, and the position. When it is paired with the FFL, the control addresses are the same.

Length—File length in words. Specifies the maximum number of words in the stack.

Position—Next location from which data are unloaded when the instruction goes from false to true.

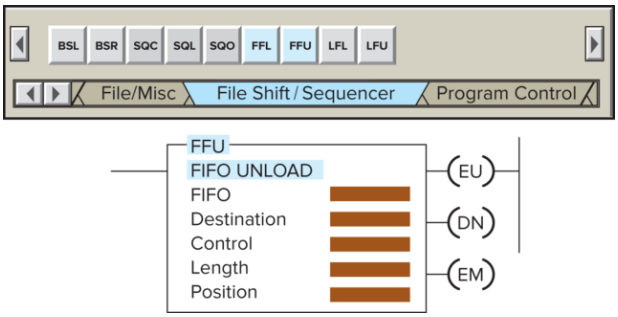


Figure 12-35 RSLogix 500 FIFO unload (FFU) instruction.

The status bits of the control word are the enable (EN), the done (DN), and the empty (EM) bits. The enable bit follows the instruction's status, the done bit is set when the instruction's position equals the length, and the empty bit is set when all the data have been unloaded from the FIFO file.

The program of **Figure 12-36** is an example of how data are indexed in and out of a FIFO file using the FFL and FFU instruction pair. The operation of the program can be summarized as follows:

- The FIFO load and FIFO unload instructions share the same control element, R6:0, which may not be used to control any other instructions.
- FIFO, #N7:12, is the address of the stack. The same address is programmed for the FFL and FFU instructions.
- Data enter the FIFO file from the source address, N7:10, on a false-to-true transition of input *A*.
- Data are placed at the position indicated in the instruction on a false-to-true transition of the FFL instruction, after which the position indicates the current number of data entries in the FIFO file.
- The FIFO file fills from the beginning address of the FIFO file and indexes to one higher address for each false-to-true transition of input *A*.

Page 260

- A false-to-true transition of input *B* causes all data in the FIFO file to shift one position toward the starting address of the file, with the data from the starting address of the file shifting to the destination address, N7:11.

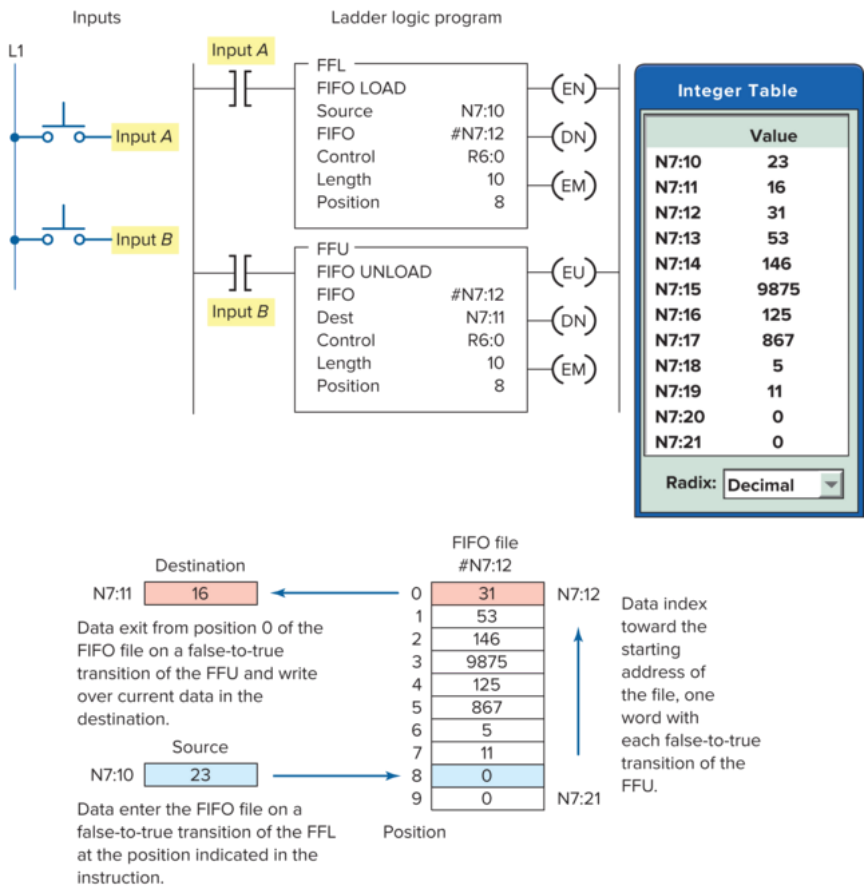


Figure 12-36 How data are indexed in and out of a FIFO file.

The FIFO instruction is often used for inventory control. One example is where different parts need to be removed from inventory to be used in production. Each part is assigned a unique code, which is loaded into a FIFO stack, and parts are removed in the order prescribed by the stack. This type of control ensures that the oldest part in the inventory is used first as the first part entered is the first part removed.

The opposite principle—where the last data to be stored are the first to be retrieved—is known as *LIFO (Last In, First Out).* The LIFO instruction inverts the order of the data it receives by outputting the last data received first and the first data received last. A useful analogy is a pile of work on your desk. As new work arrives you drop it on the top of the stack. If your stack is LIFO, you pick your next job from the top of the pile. If your stack is FIFO, you pick your work from the bottom of the pile.

Figure 12-37 shows how the FIFO and LIFO operations work for container stacking operations.

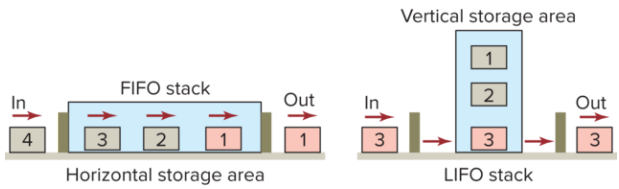


Figure 12-37 FIFO and LIFO container stacking operations.

The difference between FIFO and LIFO stack operation is that the LIFO instruction removes data in the reverse of the order they are loaded (last in, first out). An example of the LIFO instruction pair is shown in **Figure 12-38** and the operation of this function can be summarized as follows:

- The load and unload of the LIFO stack operates similarly to that of the FIFO stack, except that the last word in the LIFO stack is the first word that is unloaded from the stack.
- Words can be added to the LIFO stack without disturbing the words already loaded on the stack.
- Otherwise, LIFO instructions operate the same as FIFO instructions.

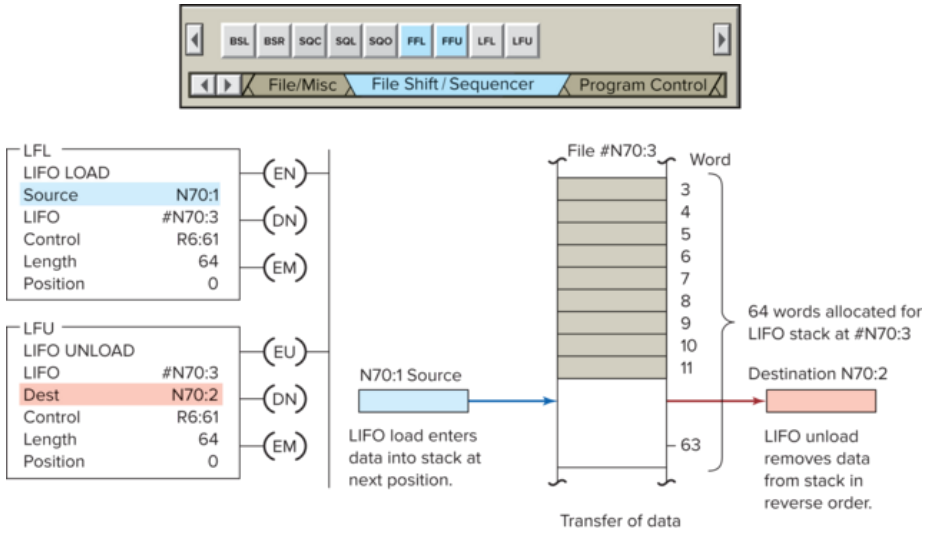


Figure 12-38 LIFO instruction pair.

Allen Bradley RSLogix 5000 programming with FIFO and LIFO functions operates similarly to the RSLogix 500 instructions except that tags and arrays are used in the parameter definitions. The program of **Figure 12-39** is an example of the use of an RSLogix 5000 FIFO instruction pair as part of a data stack operation. The operation of the program can be summarized as follows.

- The thumbwheel switch input is used to set the decimal number.
- The data stack is capable of containing 5 words.
- It takes values from the input thumbwheel switch and stores them in the data stack array.
- The values can be pulled from the stack in a FIFO order and sent to the stack output location.
- The Data_Stack_Load input pushbutton is used to load the decimal numbers into the array.
- The output module is used to display the decimal numbers and represents the destination address.
- The Data_Stack_Unload input pushbutton is used to trigger the FIFO unload operation.

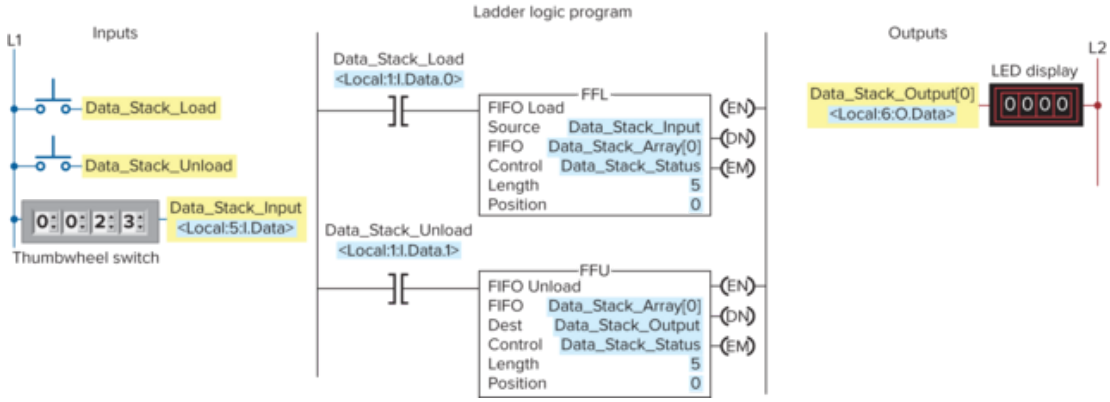


Figure 12-39 RSLogix 5000 FIFO instruction pair.

OceanofPDF.com



CHAPTER 12 REVIEW QUESTIONS

1. Describe the operation of a drum switch.
2. What type of operations are sequencers most suitable for?
3. Why are PLC sequencers easier to program than PLC discrete outputs?
4. Answer the following with regard to a PLC sequencer output instruction:
   a. Where is the information for each sequencer step entered?
   b. What is the function of the Destination?
   c. Explain the transfer of data that occurs as the sequencer is advanced through its various steps.
5. What is the function of the file of a sequencer?
6. What is the function of the Mask in the sequencer instruction?
7. What is the relationship between the length and the position in a sequencer instruction?
8. What output and step programming limits may be placed on sequencer instructions?
9. Sequencer instructions are usually retentive. Explain what this means.
10. Compare the operation of an event-driven and a time-driven sequencer.
11. Explain the function of a sequencer compare instruction.
12. What is the primary application in which an SQL instruction is used?
13. Explain the function of a sequencer load instruction.
14. How does a bit shift register manipulate individual bits?
15. List four common applications for bit shift registers.
16. When using a sensor as the input to the bit address of a BSL instruction, what is its function?
17. Compare the operation of the BSL and BSR bit shift instructions.
18. A bit shift register is said to operate in a synchronous manner. Explain what this means.
19. What is the function of the unload bit in a shift register instruction?
20. A first in, first out word shift register operates in an asynchronous manner. Explain what this means.
21. Why are both FFL and FFU instructions needed to perform a FIFO function?
22. Compare the operation of a FIFO register and a LIFO register.

OceanofPDF.com



CHAPTER 12 PROBLEMS

1. Construct an equivalent sequencer data table for the four steps of the mechanical drum-operated sequencer drawn in **Figure 12-40**.

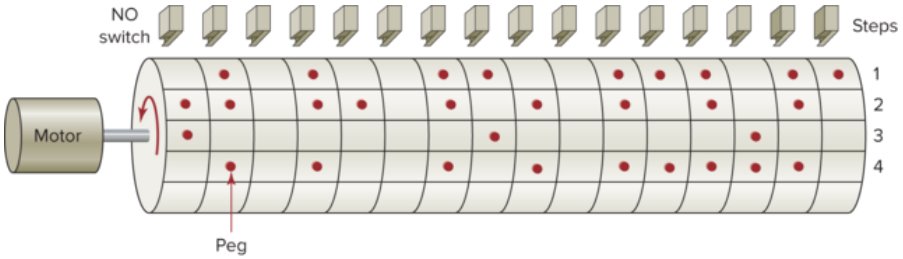


Figure 12-40 Drum-operated sequencer for Problem 1.

2. Answer the following with reference to the sequencer file #B3:0 shown in **Figure 12-41**:

a. Assume that output bit addresses O:2/0 through O:2/15 are controlling associated output pilot lights PL1 through PL16. State the status of each light for steps 1 through 4.

b. Which output bit addresses could be masked and which could not? Why or why not?

c. State the status of each bit of output word O:2 for step 3 of the sequencer cycle.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Output O:2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
																	Positions
	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Start
	1	1	0	1	1	0	1	1	0	1	1	0	0	0	1	1	Step 1
	0	0	1	0	0	1	0	0	1	0	0	1	1	1	0	0	Step 2
	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	Step 3
	1	1	1	1	0	0	0	0	1	1	1	1	0	0	0	0	Step 4

Figure 12-41 Sequencer file for Problem 2.

3. Answer each of the following with reference to the timer-driven sequencer program shown in **Figure 12-42**:

a. How many bit outputs are controlled by this sequencer?

b. Which tag controls the structure for the instruction?

c. Which tag sets the states for the outputs?

d. Which array contains the preset timer values?

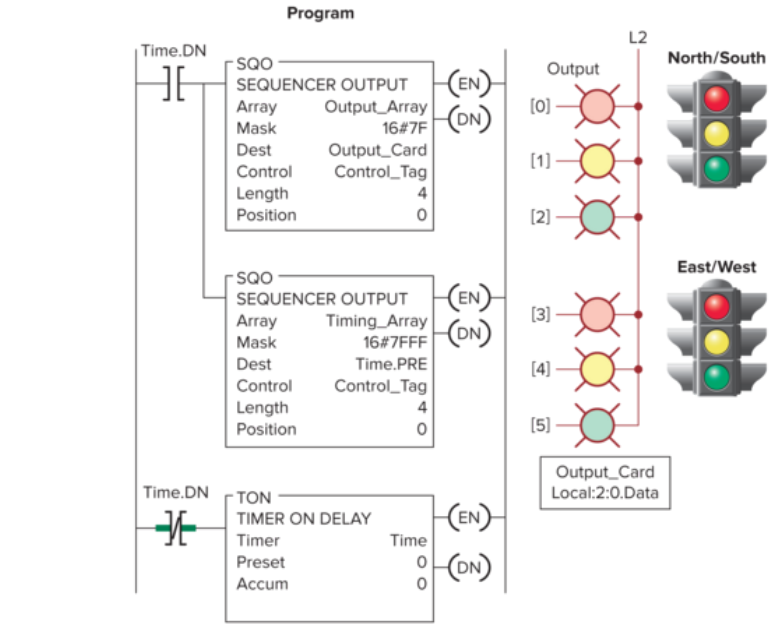
e. For what length of time is the red light programmed to be on?

f. For what length of time is the green light programmed to be on?

g. For what length of time is the yellow light programmed to be on?

h. What is the time required for one complete cycle of the sequencer?

i. Assume that the decimal value stored in Timing_Array[4] is changed to 35. Outline the changes that this new value will have on the timing of the traffic lights.



[–] Timing_Array	Tag Editor / Monitor Tags	Positions
[+] Timing_Array [0]	0	0
[+] Timing_Array [1]	25000	1
[+] Timing_Array [2]	5000	2
[+] Timing_Array [3]	25000	3
[+] Timing_Array [4]	5000	4

[–] Output_Array	Tag Editor / Monitor Tags	Positions
[+] Output_Array [0]	_0000_0000_0000	0
[+] Output_Array [1]	_0000_0010_0001	1
[+] Output_Array [2]	_0000_0001_0001	2
[+] Output_Array [3]	_0000_0000_1100	3
[+] Output_Array [4]	_0000_0000_1010	4

Figure 12-42 Timer-driven sequencer program for Problem 3.

4. Answer each of the following with reference to the event-driven sequencer program shown in **Figure 12-43**:

a. When does the sequencer advance to the next step?

b. Assume that the sequencer is at position 2, as shown; what bit outputs will be on?

c. Assume that the sequencer is stepped to position 8; what bit outputs will be on?

d. Assume that the sequencer is at position 8 and a true-to-false transition of one of the inputs occurs. What happens as a result?

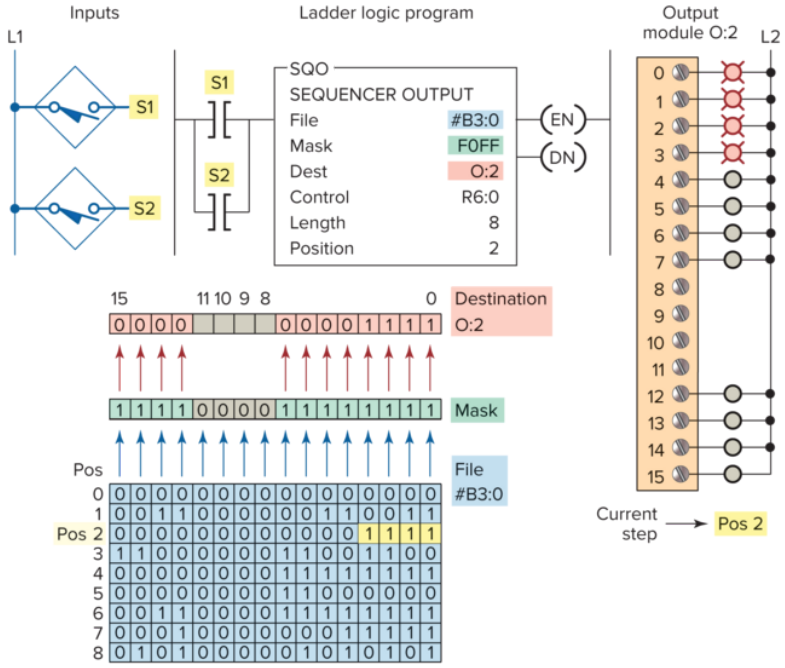


Figure 12-43 Event-driven sequencer program for Problem 4.

5. Using whatever PLC sequencer output instruction you are most familiar with, develop a program that will operate the cylinders in the desired sequence. The time between each step is to be 3 seconds. The desired sequence of operation will be as follows:

- All cylinders to retract.
- Cylinder 1 advance.
- Cylinder 1 retract and cylinder 3 advance.
- Cylinder 2 advance and cylinder 5 advance.
- Cylinder 4 advance and cylinder 2 retract.
- Cylinder 3 retract and cylinder 5 retract.
- Cylinder 6 advance and cylinder 4 retract.
- Cylinder 6 retract.
- Sequence to repeat.

6. Using whatever PLC sequencer output instruction you are most familiar with, develop a program to implement an automatic car-wash process. The process is to be event-driven by the vehicle, which activates various limit switches (LS1 through LS6) as it is pulled by a conveyor chain through the car-wash bay. Design the program to operate the car wash in the following manner:

- The vehicle is connected to the conveyor chain and pulled inside the car-wash bay.
- LS1 turns on the water input valve.
- LS2 turns on the soap release valve, which mixes with the water input valve to provide a wash spray.

- LS3 shuts off the soap valve, and the water input valve remains on to rinse the vehicle.
- LS4 shuts off the water input valve and activates the hot wax valve, if selected.
- LS5 shuts off the hot wax valve and starts the air-blower motor.
- LS6 shuts off the air blower. The vehicle exits the car wash.

7. A product moves continuously down an assembly line that has four stations, as shown in **Figure 12-44**.
   - The product enters the inspection zone, where its presence is sensed by the proximity switch.
   - The inspector examines it and activates a reject button if the product fails inspection.
   - If the product is defective, reject status lights come on at stations 1, 2, and 3 to tell the assembler to ignore the part.
   - When a defective part reaches station 4, a diverter gate is activated to direct that part to a reject bin.
   - Using whatever PLC bit shift register you are most familiar with, develop a program to implement this process.

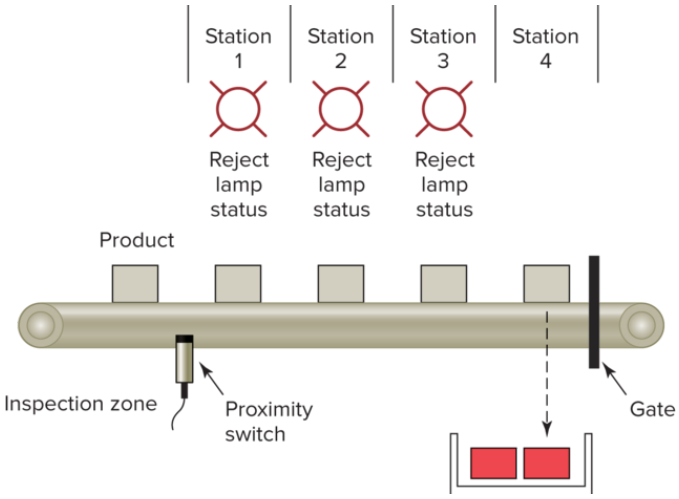
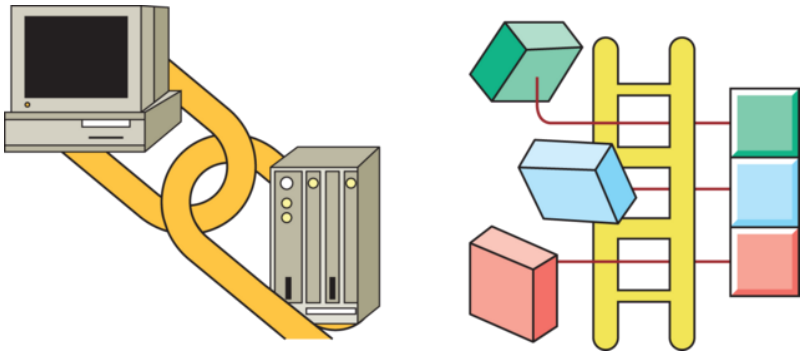


Figure 12-44 Assembly line program for Problem 7.

OceanofPDF.com

13

PLC Installation Practices, Editing, and Troubleshooting



Chapter Objectives

After completing this chapter, you will be able to:

- Outline and describe requirements for a PLC enclosure
- Identify and describe noise reduction techniques
- Describe proper grounding practices and preventive maintenance tasks associated with PLC systems
- List and describe specific PLC troubleshooting procedures

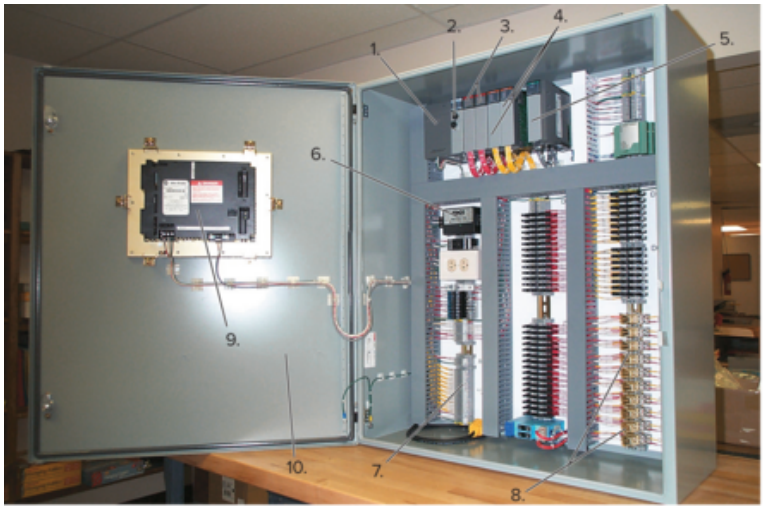
This chapter discusses guidelines for the installation, maintenance, and troubleshooting of a PLC-controlled system. The chapter gives you information on proper grounding that ensures personal safety as well as correct operation of equipment. Unique troubleshooting procedures that apply specifically to PLCs are listed and explained.

OceanofPDF.com

13.1 PLC Enclosures

A PLC system, if installed properly, should give years of trouble-free service. The design of PLCs includes a number of rugged features that allow them to be installed in almost any industrial environment. However, problems can occur if the system is not installed properly.

Programmable logic controllers (PLCs) require protection against temperature extremes, humidity, dust, shock, and vibration or corrosive environments. For these reasons, PLCs are generally mounted within a machine or in a separate ***enclosure*** as shown in Figure 13-1.



1. Power supply
2. PLC (programmable logic controller)
3. Digital input cards
4. Digital output cards
5. Analog input cards
6. Transient surge protectors
7. Circuit breakers
8. Relay switches
9. Operator interface terminal
10. NEMA 12 enclosure

Figure 13-1 Typical PLC control panel enclosure.

Source: Courtesy Aaron Associates

An enclosure is the chief protection from atmospheric conditions. The National Electrical Manufacturers Association (NEMA) has defined enclosure types, based on the degree of protection an enclosure will provide. For most solid-state control devices, a NEMA 12 enclosure is recommended. This type of enclosure is for general-purpose areas and is designed to be dust-tight. Typically, metal enclosures are used because metal enclosures provide shielding that helps minimize the effects of electromagnetic radiation that may be generated by surrounding equipment.

Every PLC installation will dissipate heat from its power supplies, local I/O racks, and processor. This heat accumulates in the enclosure and must be dissipated from it into the surrounding air. Excessive heat can cause erratic operation of the PLC or PLC failure. For many applications, normal convection cooling will keep the controller components within the specified temperature operating range. Proper spacing of components that provides adequate room within the enclosure is usually sufficient for heat dissipation. The temperature inside the enclosure must not exceed the maximum operating temperature of the controller (typically 60°C maximum). Additional cooling provisions, such as a fan or blower, may be required where high internal or ambient temperatures are encountered. PLCs are always mounted horizontally with the name of the manufacturer facing out and right-side up, as illustrated in Figure 13-2. Vertical mounting is not recommended due to thermal considerations.



Figure 13-2 PLCs are always mounted horizontally.

Source: Courtesy Rogers Machinery Company, Inc.

Page 268

A hardwired electromechanical ***master control relay (MCR)*** is normally included as part of the wiring for a PLC system. The master control relay provides a means of de-energizing the entire circuit that *is not* dependent on software. The internally programmed MCR of a PLC is not sufficient to meet safety requirements. The hardwired MCR is connected to interrupt power to the I/O rack in the event of an emergency, but still allow power to be maintained at the processor. **Figure 13-3** shows the typical wiring for an AC power distribution with a safety and master control relay. The operation of the circuit can be summarized as follows:

Page 269

- A power disconnect switch is provided so that, when required, the PLC can be serviced with the power off.
- The step-down transformer provides isolation from the main power distribution system and decreases the voltage to the 120 V required for the controller power supplies and DC power supplies.
- The momentary start button is pressed to energize the master control relay.
- Pressing any one of the emergency-stop switches de-energizes the master control relay and thus de-energizes the I/O devices.
- Power to the processor of the PLC remains on so status LEDs can continue to provide up-to-date information.
- Emergency stop buttons use normally closed contacts wired in series for fail-safe operation. In the event a wire is broken or comes off a terminal, the MCR relay is de-energized and power is removed.

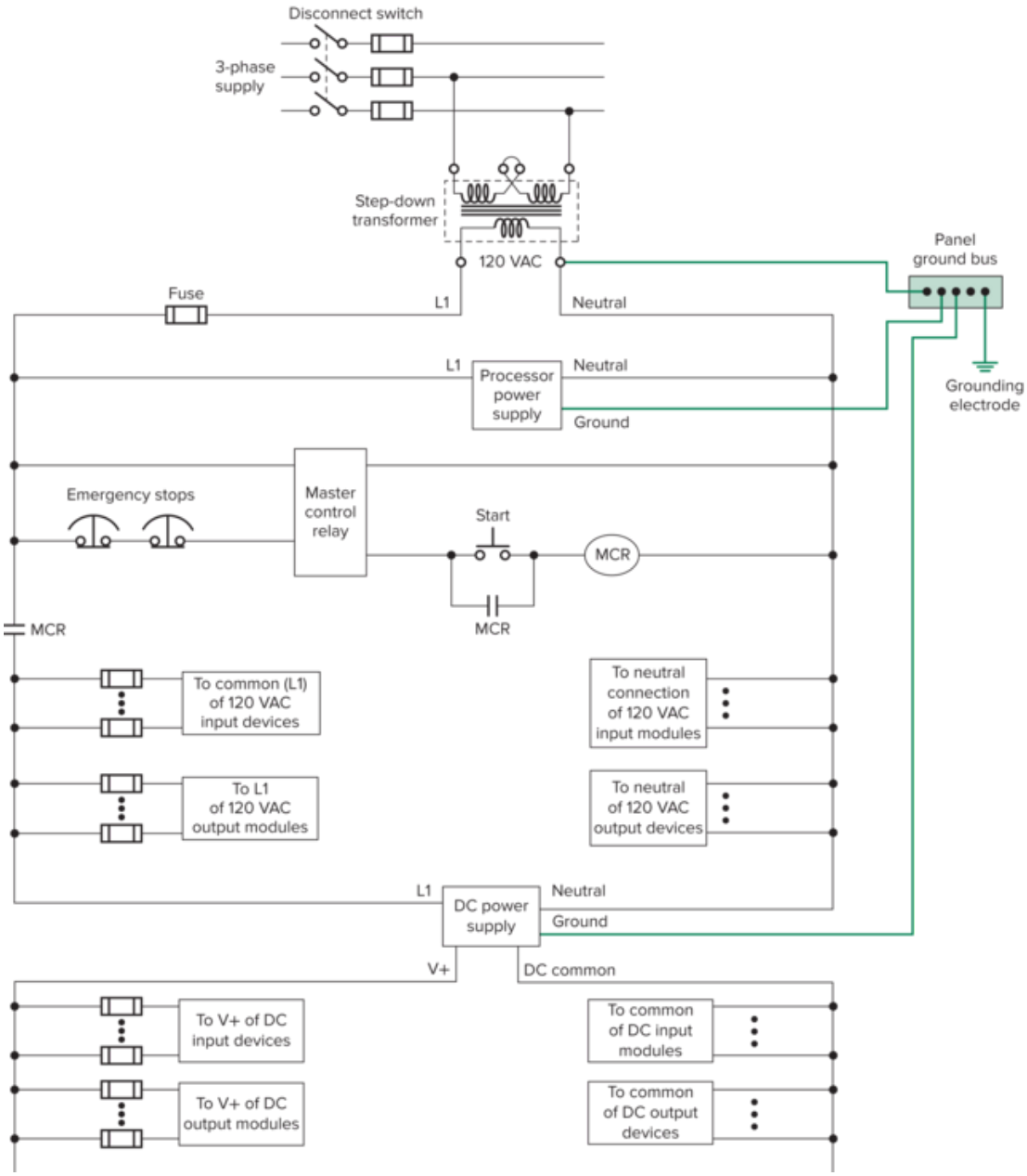


Figure 13-3 Typical wiring for an AC power distribution with a master control relay.

OceanofPDF.com

13.2 Electrical Noise

Electrical noise, also called **electromagnetic interference, or EMI,** is unwanted electrical signals that produce undesirable effects and otherwise disrupt the control system circuits. EMI may be either radiated or conducted. *Radiated* noise originates from a source and travels through the air while *conducted* noise travels on an actual conductor, such as a power line.

When the PLC is operated in a noise-polluted industrial environment, special consideration should be given to possible electrical interference. To increase the operating noise margin, the controller should be located away from noise-generating devices such as large AC motors and high-frequency welders. Malfunctions resulting from noise are temporary occurrences of operating errors that can result in hazardous machine operation in certain applications. Noise usually enters through input, output, and power supply lines. Noise may be coupled into these lines by an electrostatic field or through electromagnetic induction. The following reduce the effect of electrical interference:

- Manufacturer design features
- Proper mounting of the controller within an enclosure
- Proper equipment grounding
- Proper routing of wiring
- Proper suppression added to noise-generating devices

Noise suppression is normally needed for **inductive loads** such as relays, solenoids, and motor starters when operated by hard contact devices such as pushbuttons or selector switches. When inductive loads are switched off, high transient voltages are generated that if not suppressed can reach several thousand volts. **Figure 13-4** illustrates, a typical noise suppression circuit that is used to suppress the high voltage spikes generated when a motor starter coil is de-energized.

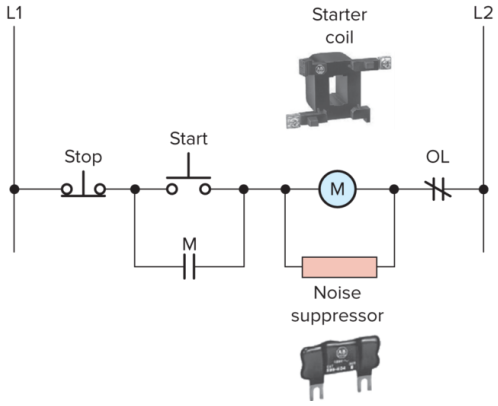


Figure 13-4 Motor starter noise suppression.

Source: Image Courtesy of Rockwell Automation, Inc.

Lack of surge suppression on inductive loads may contribute to processor faults and sporadic operation. RAM can be corrupted (lost), and I/O modules can appear faulty or can reset themselves.

When inductive devices are energized or de-energized, they can cause an electrical pulse to be back-fed into the PLC system. The back-fed pulse, when entering the PLC system, can be mistaken by the PLC for a computer pulse. It takes only one false pulse to create a malfunction of the orderly flow of PLC operational sequences.

Proper routing of field power and signal wiring to the PLC enclosure as well as inside the enclosure helps to cut down on electrical noise (also known as cross-talk interference). The following are some general guidelines for PLC wire routing:

- Use the shortest possible wire runs for I/O signals.
- When possible, conductors that are run from the PLC enclosure to another location should be in a metal conduit as the metal can serve as a shield against EMI.
- *Never* run signal wiring and power wiring in the same conduit.

- Segregate I/O wiring by signal type. Route AC and DC I/O signal wires in separate wireways.
- Low-level signal conductors such as thermocouples and serial communications should be run as shielded twisted pair and routed separately.
- A fiber optic system, which is totally immune to all kinds of electrical interference, can also be used for signal wiring.

An important part of a PLC installation is clearly **identifying** each wire to be connected and the terminal to which it is connected. A reliable labeling method, such as the heat-shrinkable wire identification sleeves shown in **Figure 13-5**, should be used to label each wire. Wiring connectors for input/output modules usually include spaces for labels used for identifying each I/O address and device connected. Proper wire and terminal identification will simplify the installation and aid in troubleshooting and maintenance.

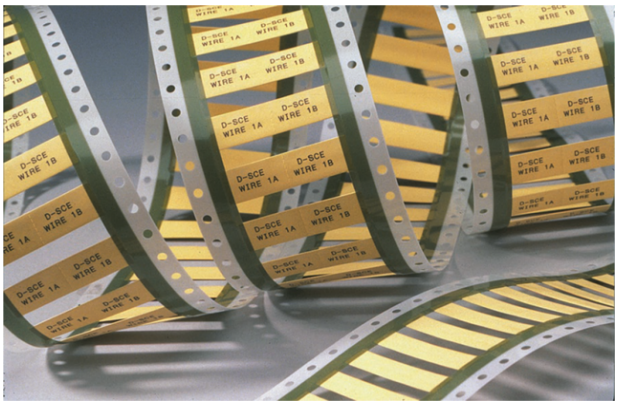


Figure 13-5 Heat-shrinkable wire identification sleeves.

Source: Image Courtesy of Rockwell Automation, Inc.

OceanofPDF.com

13.3 Leaky Inputs and Outputs

Many electronic devices with transistor or triac outputs exhibit a small **leakage current** even when in the off state that may need to be considered when they are connected to PLC input modules. This so-called leakage is typically exhibited by two-wire proximity, photoelectric, and other such sensors. Often, the leaky input will only cause the module's input indicator to flicker. However, a large enough leakage current can activate the input circuit, creating a false input signal.

A common solution to the problem of leaky input current is to connect a **bleeder resistor** across or in parallel with the input, as shown in Figure 13-6. The value of the bleeder resistor is selected to ensure an effective operation of the input circuit's functionality.

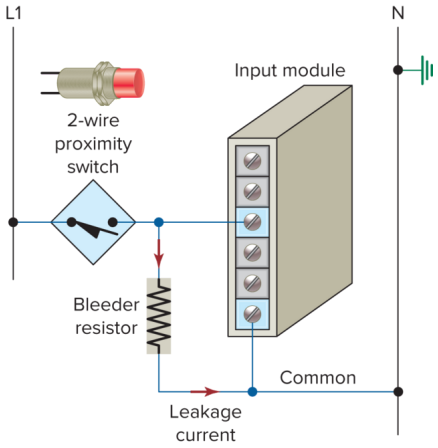


Figure 13-6 Bleeder resistor connection for input sensors.

Leakage current may also occur with the solid-state switch used in many output modules. Problems similar to that encountered with input modules can be created when a high-impedance load device is used with these modules. For example, a PLC output might supply a sound alert device as illustrated in Figure 13-7. In this case the leakage current could cause continuous false or intermittent operation. A resistor can be connected as shown to bleed off this current. An isolation relay could also be used to solve this type of problem.

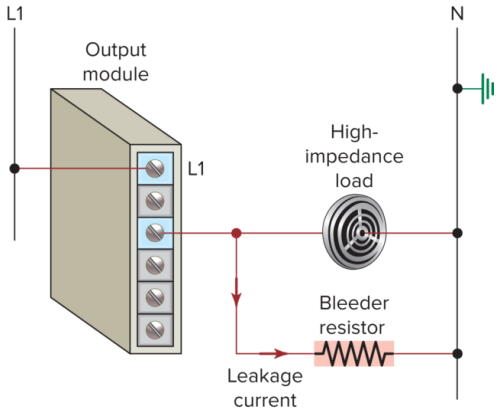


Figure 13-7 Bleeder resistor connection for a high-impedance output.

OceanofPDF.com

13.4 Grounding

Proper **grounding** is an important safety measure in all electrical installations. The authoritative source on grounding requirements for a PLC installation is the **National Electrical Code**. The NEC specifies the types of conductors, color codes, and connections necessary for safe grounding of electrical components. In addition, most manufacturers provide detailed information on the proper grounding methods to use in an enclosure.

Figure 13-8 illustrates a PLC grounding system. A properly installed grounding system will provide a low-impedance path to earth ground. The complete PLC installation, including enclosures, CPU and I/O chassis, and power supplies are all connected to a single low-impedance ground. These connections should exhibit low DC resistance and low high-frequency impedance. A central ground bus bar is provided as a single point of reference inside the enclosure to which all chassis and power supply equipment grounding conductors are connected. The ground bus is then connected to the building's earth ground.

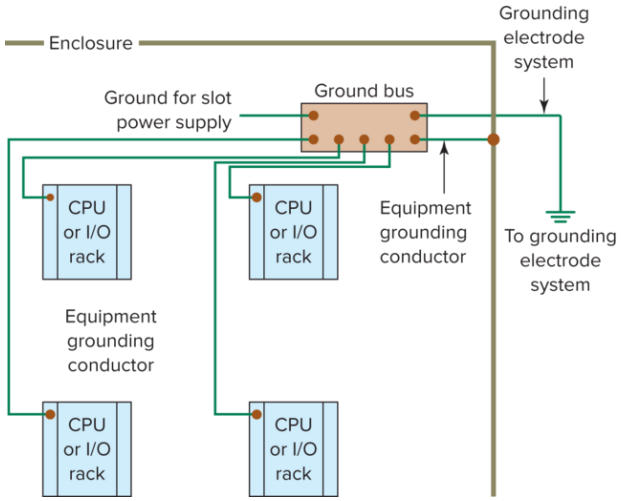


Figure 13-8 PLC grounding system.

In the event of a high value of ground current, the temperature of the conductor could cause the solder to melt, resulting in interruption of the ground connection. Therefore the grounding path must be permanent (no solder), continuous, and able to conduct safely the ground-fault current in the system with minimal impedance. Paint or other nonconductive material should be scraped away from the area where a chassis makes contact with the enclosure. The minimum ground wire size should be No. 12 AWG stranded copper for PLC equipment grounds and No. 8 AWG stranded copper for enclosure backplane grounds. Ground connections should be made with a star washer between the grounding wire and lug and metal enclosure surface, as illustrated in **Figure 13-9**.

- Any protective ground wires must have a resistance value of less than 0.1 Ω.
- The resistance from the system ground to the earth ground must have a value of less than 0.1 Ω.

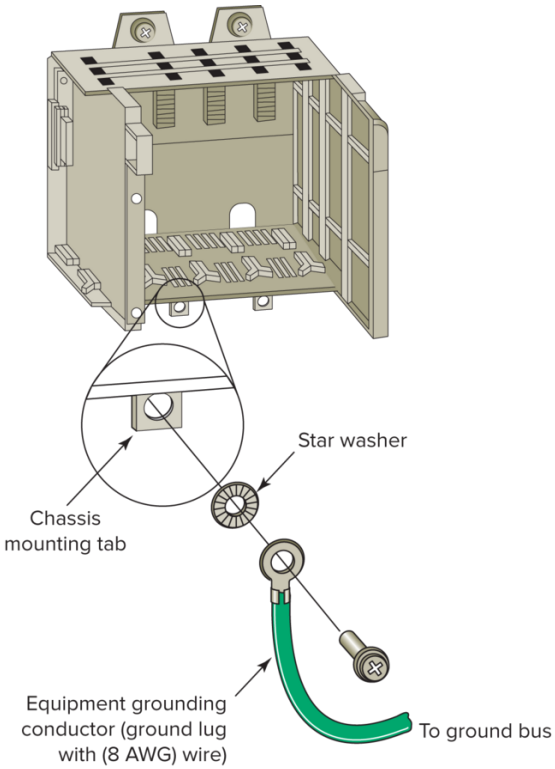


Figure 13-9 Make ground connections using a star washer.

Ground loops can cause problems by adding or subtracting current or voltage from input signal devices. A ground loop circuit can develop when each device's ground is tied to a different earth potential thereby allowing current to flow between the grounds, as illustrated in **Figure 13-10**. If a varying magnetic field passes through one of these ground loops, a voltage is produced and current flows in the loop. The receiving device is unable to differentiate between the wanted and unwanted signals and, thus, can't accurately reflect actual process conditions. Certain connections require shielded cables to help reduce the effects of electrical noise coupling. Each shield should be grounded at one end only, as a shield grounded at both ends forms a ground loop.

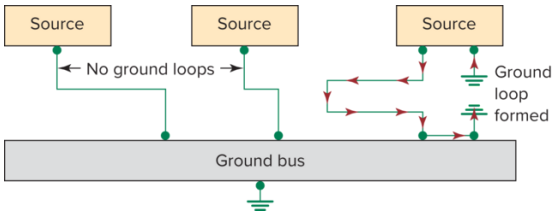


Figure 13-10 Formation of ground loops.

OceanofPDF.com

13.5 Voltage Variations and Surges

The power supply section of the PLC system is built to sustain line fluctuations and still allow the system to function within its operating range. If voltage fluctuations exceed this range, then a system shutdown will occur. In areas where excessive line voltage variation or extended brownouts are anticipated, installing a ***constant voltage (CV)*** transformer may be required to minimize nuisance shutdowns of the PLC.

Isolation transformers are used in some PLC systems to isolate the PLC from electrical disturbances generated by other equipment connected to the distribution system. Although the PLC is designed to operate in harsh environments, other equipment may generate considerable amounts of interference that may result in intermittent disturbances in normal operation. A normal practice is to place the PLC power supply and I/O devices on a separate transformer that may also serve as a step-down transformer to reduce the incoming voltage to the desired level.

When current in an inductive load is interrupted or turned off, a very high voltage spike is generated. This high voltage can be reduced or eliminated through suppression techniques which absorb the inductive induced voltage. Generally, output modules designed to drive inductive loads include suppression networks built in as part of the module circuit.

An additional external **suppression device** is recommended if an output module is used to control devices such as relays, solenoids, motor starters, or motors. The suppression device is wired in parallel (directly across) and as close as possible to the load device. The suppression components must be rated appropriately to suppress the switching transient characteristic of the particular inductive device. **Figure 13-11** illustrates how a diode is connected to suppress DC inductive loads. The operation of the circuit can be summarized as follows:

- The diode is connected in reverse-bias across the solenoid load.
- In normal operation, the electric current can't flow through the diode, so it flows through the solenoid coil.
- When voltage to the solenoid is switched off a voltage opposite in polarity to the original applied voltage is generated by the collapsing magnetic field.
- The induced voltage creates a current flow through the diode bleeding off the high-voltage spike.

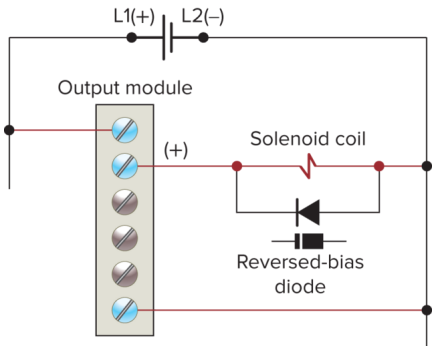


Figure 13-11 Diode connected to suppress DC inductive loads.

Figure 13-12 illustrates how an RC (resistor/capacitor) snubber circuit is connected for suppressing AC load devices. The operation of the circuit can be summarized as follows:

- The voltage peak, which occurs at the instant the current path to the coil is opened, is safely short-circuited by the RC network.
- The resistor and capacitor connected in series slows the rate of rise of the transient voltage.
- The voltage across the capacitor cannot change instantaneously, so a decreasing transient current will flow through it for a small fraction of a second, allowing the voltage to increase more slowly when the circuit is opened.

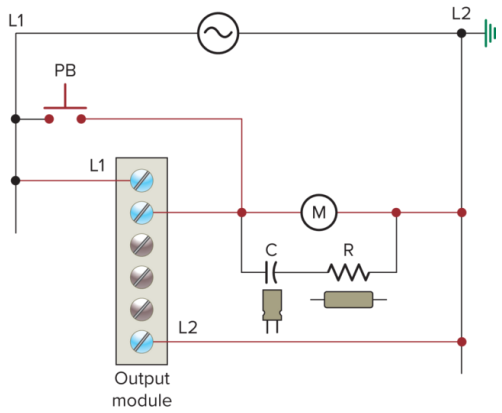


Figure 13-12 RC snubber circuit connected to suppress AC loads.

The ***metal oxide varistor (MOV)*** surge suppressor, shown in **Figure 13-13**, is the most popular surge protection device. It functions in a manner similar to two Zener diodes connected back-to-back. The operation of an MOV can be summarized as follows:

- The device acts as an open circuit until the voltage across it in either direction exceeds its rated value.
- Any greater voltage peak instantly makes the device act like a short circuit that bypasses this voltage away from the rest of the circuit.

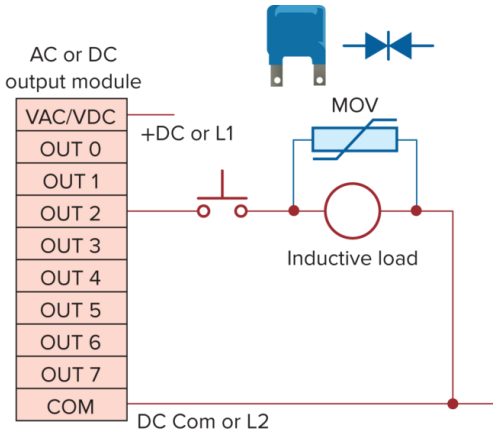


Figure 13-13 Metal oxide varistor (MOV) surge suppressor.

OceanofPDF.com

13.6 Program Editing and Commissioning

After you have entered the rungs for your program, you may need to modify them. ***Editing*** is simply the ability to make changes to an existing program through a variety of editing functions. Using the editing function, instructions and rungs can be added or deleted; addresses, data, and bits can be changed. Again, the editing format varies with different manufacturers and PLC models.

Today, most PLC programming software is Microsoft Windows based, so if you are familiar with Windows and know how to point and click with a mouse, you should have no problem editing a program. In general, both instructions and rungs are selected simply by clicking on them with the left mouse button. Double clicking with the left mouse button allows you to edit an instruction's address, whereas right clicking displays a pop-up menu of related editing commands. If you want to include additional explanation of a symbol or address, you can place an address description on your ladder rung directly above the symbol. To add a page or rung comment, right click on the rung number to which you wish to add the comment and click Edit Rung Comment.

Preparing a control process for start-up, also called ***commissioning,*** involves a series of tests to ensure that the PLC, the ladder logic program, the I/O devices, and all associated wiring operate according to specifications. Before commissioning any control system, you should have a good understanding of how the control system operates and how the various components interact. The following are general steps to be followed when commissioning a PLC system:

- Before applying power to the PLC or the input devices, disconnect or otherwise isolate any output device that could potentially cause damage or injury. Typically this precaution would pertain to outputs that cause movement such as starting a motor or operating a valve.
- Apply power to the PLC and input devices. Measure the voltage to verify that rated voltage is being applied.
- Examine the PLC's status indicator lights. If power is properly applied, the power indicator should be on, and there should be no fault indication. If the PLC does not power up properly, it may be faulty. PLCs rarely fail, but if they do fail, it usually happens immediately upon powering up.
- Verify that you have communication with the PLC via the programming device that is running the PLC programming software.
- Place the PLC in a mode that prevents it from energizing its output circuits. Depending on the make of the PLC, this mode may be called *disable, continuous test,* or *single-scan* mode. This mode will allow you to monitor input devices, execute the program, and update the output image file while keeping the output circuits de-energized.
- Manually activate each input device, one at a time, to verify that the PLC's input status lights turn on and off as expected. Monitor the associated condition instruction to verify that the input device corresponds to the correct program address and that the instruction turns true or false as expected.
- Manually test each output. One way you can do this is by applying power to the terminal where the output device is wired. This test will check the output field device and its associated wiring.

- After verifying all inputs, outputs, and program addresses, verify all preset values for counters, timers, and so on.
- Reconnect any output devices that may have been disconnected and place the PLC in the run mode. Test the operation of all emergency stop buttons and the total system operation.

OceanofPDF.com

13.7 Programming and Monitoring

When you program a PLC, several instruction entry modes are available, depending on the manufacturer and the model of the unit. A personal computer, with appropriate software, is generally used to program and monitor the program in the PLC. Additionally, it makes possible ***offline programming,*** which involves writing and storing the program in the personal computer without its being connected to the PLC and later downloading it to the PLC. **Figure 13-14** illustrates how programs are downloaded and uploaded from and to the computer.

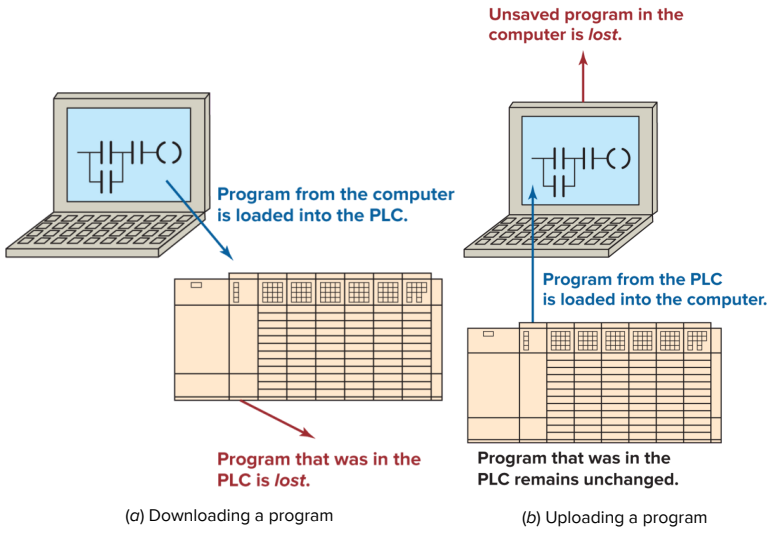


Figure 13-14 Downloading and uploading PLC program.

With ***online programming*** the program can be modified, the modifications can be tested, and finally they can be accepted or rejected while the PLC is running. However, offline programming is the safest manner in which to edit a program because additions, changes, and deletions do not affect the operation of the system until downloaded to the PLC.

Many manufacturers provide a ***continuous test mode*** that causes the processor to operate from the user program without energizing any outputs. This mode allows the control program to be executed and debugged while the outputs are disabled. A check of each rung can be done by monitoring the corresponding output rung on the programming device. A ***single-scan*** test mode may also be available for debugging the control logic. This mode causes the processor to complete a single scan of the user program each time the single-scan key is pressed with no outputs being energized.

An online programming mode permits the user to change the program during machine operation. As the PLC controls its equipment or process, the user can add, change, or delete control instructions and data values as desired. Any modification made is executed immediately on entry of the instruction. Therefore, the user should assess in advance all possible sequences of machine operation that will result from the change. Online programming should be done only by experienced personnel who

understand fully the operation of the PLC they are dealing with and the machinery being controlled. If at all possible, changes should be made offline to provide a safe transition from existing programming to new programming.

Two useful monitoring tools provided with PLC programming packages are data monitor and cross reference. ***Data monitoring*** functions allow you to monitor and/or modify specified program variables. The ***cross reference*** function allows you to search each instance of a particular address.

The data monitor feature allows you to display data from any place in the data table. Depending on the PLC, the data monitor function can be used to do the following:

- View data within an instruction
- Store data or values for an instruction prior to use
- Set or reset values and/or bits during a debug operation for control purposes
- Change the radix or data format

Figure 13-15 shows the data file folder and window for the Allen-Bradley RSLogix 500 PLC and its associated RSLogix software. The data file folder allows the user to determine the status of I/O files as well as the status file (S2), binary file (B3), timer file (T4), counter file (C5), control file (R6), integer file (N7), and the floating-point file (F8). Always be careful when manipulating data using the data monitor function. Changing data could affect the program and turn output devices on or off.

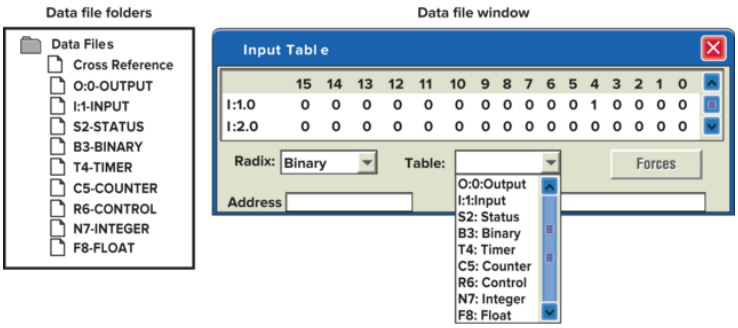


Figure 13-15 Data file folder and window.

Source: Courtesy of TheLearningPit

When troubleshooting a PLC, it may be necessary to locate each instance of a particular address in the ladder program. The cross reference function searches all program files to locate each instance of the selected address. A user can then trace the operation by finding all the places where a particular output coil or contact with the same address is used in the program. **Figure 13-16** shows a sample cross reference report for the Allen-Bradley RSLogix 500 PLC and its associated RSLogix software. Its contents can be summarized as follows:

- The report contains all the addresses used in the program.
- Addresses are displayed in the same order as the data table files.
- The address that the search was performed for (O:2/1) is highlighted.
- The description for each address is displayed.

- Listing includes the instruction type, program file, and rung number for each address.
- Each occurrence of the address is displayed, starting with program file 2 and rung 0.

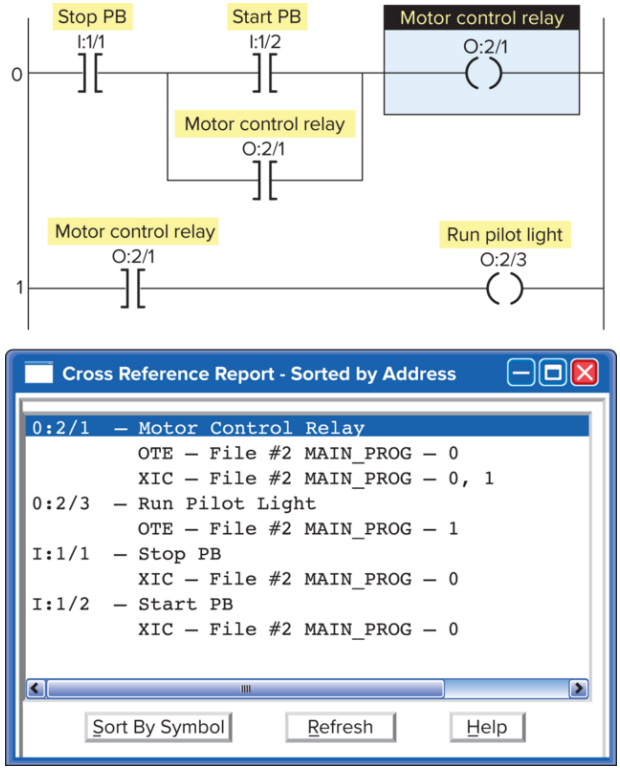


Figure 13-16 Sample cross reference report.

Source: Courtesy of TheLearningPit

The ***contact histogram*** function allows you to view the transition history (the on and off states) of a data table value. The status of the bit(s) (on or off) and the length of time the bit(s) remained on or off (in hours, minutes, seconds, and hundredths of a second) are displayed. In a contact histogram file, the accumulated time indicates the total time that the histogram function was running. The delta time of the contact histogram indicates the elapsed time between the changes in states. Contact histograms are extremely useful for detecting intermittent problems, either hardware- or logic-related. By tracking the status and time between status changes, you can detect different types of problems.

OceanofPDF.com

13.8 Preventive Maintenance

The biggest deterrent to PLC system faults is a proper **preventive maintenance** program. Although PLCs have been designed to minimize maintenance and provide trouble-free operation, there are several preventive measures that should be looked at regularly.

Many control systems operate processes that must be shut down for short periods for product changes. The following preventive maintenance tasks should be carried out during these short shutdown periods:

- Any filters that have been installed in enclosures should be cleaned or replaced to ensure that clear air circulation is present inside the enclosure.
- Dust or dirt accumulated on PLC circuit boards should be cleaned. If dust is allowed to build up on heat sinks and electronic circuitry, an obstruction of heat dissipation could occur and cause circuit malfunction. Furthermore, if conductive dust reaches the electronic boards, a short circuit could result and cause permanent damage to the circuit board. Ensuring that the enclosure door is kept closed will prevent the rapid buildup of these contaminants.
- Connections to the I/O modules should be checked for tightness to ensure that all plugs, sockets, terminal strips, and module connections are making connections and that the module is installed securely. Loose connections may result not only in improper function of the controller but also in damage to the components of the system.
- All field I/O devices should be inspected to ensure that they are adjusted properly. Circuit boards dealing with process control analogs should be calibrated every 6 months. Other devices, such as sensors, should be serviced on a monthly basis. Field devices in the environment, which have to translate mechanical signals into electrical, may gum up, get dirty, crack, or break—and then they will no longer trip at the correct setting.
- Care should be taken to ensure that heavy noise- or heat-generating equipment is not moved too close to the PLC.
- Check the condition of the battery that backs up the RAM memory in the CPU (Figure 13-17). Most CPUs have a status indicator that shows whether the battery's voltage is sufficient to back up the memory stored in the PLC. If a battery module is to be replaced, it must be replaced with exactly the same type of battery module.
- Stock commonly needed spare parts. Input and output modules are the PLC components that fail most often.
- Keep a master copy of operating programs used.

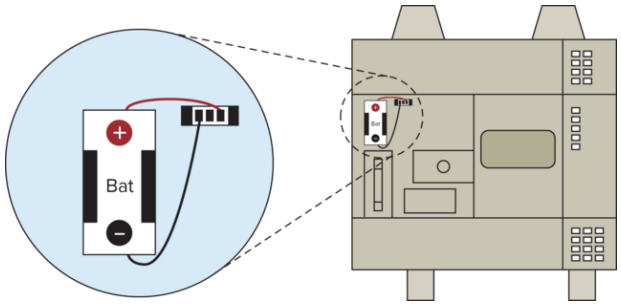


Figure 13-17 CPU backup memory battery.

To avoid injury to personnel and to prevent equipment damage, connections should always be checked with power removed from the system. In addition to disconnecting electrical power, all other sources of power (pneumatic and hydraulic) should be de-energized before someone works on a machine or process controlled by a PLC. Authorized service personnel are required to use **lockout and tagout procedures** to make sure that equipment does not operate while maintenance and repairs are conducted. A personnel protection tag is placed on the power source for the equipment and the PLC, and it can be removed only by the person who originally placed the tag. In addition to the tag, a lock is also attached so that equipment cannot be energized.

OceanofPDF.com

13.9 Troubleshooting

In the event of a PLC fault, you should employ a careful and systematic approach to troubleshoot the system to resolve the problem. PLCs are relatively easy to troubleshoot because the control program can be displayed on a monitor and watched in real time as it executes. If a control system has been operating, you can be fairly confident of the accuracy of the program logic. For a system that has never worked or is just being commissioned, programming errors should be considered.

When a problem occurs, the first step in the troubleshooting procedure is to identify the problem and its source. The source of a problem can generally be narrowed down to the processor module, I/O hardware, wiring, machine inputs or outputs, or ladder logic program. Once a problem is recognized, it is usually quite simple to deal with. The following sections will deal with troubleshooting these potential problem areas.

Processor Module

The processor is responsible for the ***self-detection*** of potential problems. It performs error checks during its operation and sends status information to indicators that are normally located on the front of the processor module. You can diagnose processor faults or obtain more detailed information about the processor by accessing the processor status through programming software. **Figure 13-18** shows sample diagnostics LEDs found on a processor module. What they indicate can be summarized as follows:

RUN (Green)

- On steady indicates that the process is in the RUN mode.
- Flashing during operation indicates that the process is transferring a program from RAM to the memory module.
- Off indicates that processor is in a mode other than RUN.

FLT (Red)

- Flashing at power-up indicates that the processor has not been configured.
- Flashing during operation indicates a major error either in the processor, chassis, or memory.
- On steady indicates that a fatal error is present (no communications).
- Off indicates there are no errors.

BATT (Red)

- On steady indicates the battery voltage has fallen below a threshold level, or the battery is missing or not connected.
- Off indicates that the battery is functional.

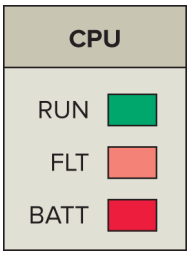


Figure 13-18
Processor diagnostics LEDs.

The processor monitors itself continually for any problems that might cause the controller to execute the user program improperly. If a fault condition occurs that is severe enough, the controller generates a major fault, stops the execution of logic, and switches to Program mode. Depending on your application, you may not want all major faults to shut down your entire system. In those situations, you can use a fault routine to clear a specific fault and let at least some of your system continue to operate.

Most PLCs incorporate a ***watchdog timer*** to monitor the scan process of the system. The watchdog timer is usually a separate timing circuit that must be set and reset by the processor within a predetermined period. The watchdog timer circuit monitors how long it takes the CPU to complete a scan. If the CPU scan takes too long, a watchdog major error will be declared. PLC user manuals will show how to apply this function.

The PLC processor hardware is not likely to fail because today's microprocessors and microcomputer hardware are very reliable when operated within the stated limits of temperature, moisture, and so on. The PLC processor chassis is typically designed to withstand harsh environments.

Input Malfunctions

If the controller is operating in the RUN mode but output devices do not operate as programmed, the faults could be associated with any of the following:

- Input and output wiring between field devices and modules
- Field device or module power supplies
- Input sensing devices
- Output actuators
- PLC I/O modules
- PLC processor

Narrowing down the problem source can usually be accomplished by comparing the actual status of the suspect I/O with controller status indicators. Usually each input or output device has at least two status indicators. One of these indicators is on the I/O module; the other indicator is provided by the programming device monitor.

The circuit of **Figure 13-19** illustrates how to check for discrete input malfunctions. The steps taken can be summarized as follows:

- When input hardware is suspected to be the source of a problem, the first check is to see if the status indicator on the input module illuminates when it is receiving power from its corresponding input device (e.g., pushbutton, limit switch).
- If the status indicator on the input module does *not* illuminate when the input device is on, take a voltage measurement across the input terminal to check for the proper voltage level.
- If the voltage level is correct, then the input module should be replaced.
- If the voltage level is not correct, power supply, wiring, or input device may be faulty.

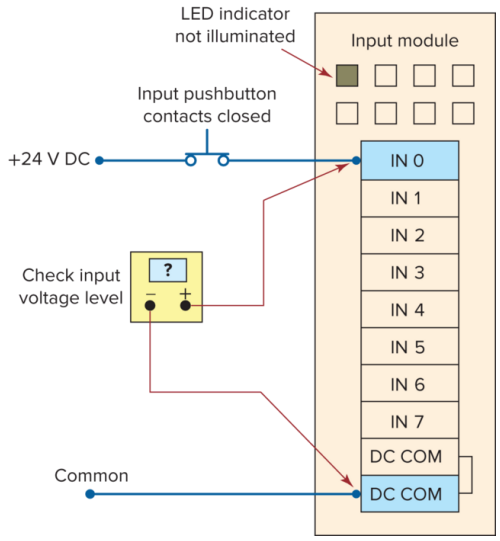


Figure 13-19 Checking for input malfunctions.

If the programming device monitor does not show the correct status indication for a condition instruction, the input module may not be converting the input signal properly to the logic level voltage required by the processor module. In this case, the input module should be replaced. If a replacement module does not eliminate the problem and wiring is assumed to be correct, then the I/O rack, communication cable, or processor should be suspected. **Figure 13-20** shows a typical input device troubleshooting guide. This guide reviews condition instructions and how their true/false status relates to external input devices.

Input device troubleshooting guide				
Input device condition	Input module status indicator	Monitor display status indicator		Possible fault(s)
Closed — ON 24 V DC input	ON	True	False	None - correct indications
Open — OFF 0 V DC input	OFF	False	True	None - correct indications
Closed — ON 24 V DC input	ON	False	True	Sensor condition, input voltage, status indicator are correct. Ladder instructions have incorrect indications. Input module or processor fault.
Closed — ON 0 V DC input	OFF	False	True	Status indicator and instructions agree but not with the sensor condition. Open field device or wiring.
Open — OFF 0 V DC input	OFF	True	False	Sensor condition, input voltage, status indicator are correct. Ladder instructions have incorrect indications. Input module or processor fault.
Open — OFF 24 V DC input	ON	True	False	Input voltage, status indicator, and ladder instructions agree but not with sensor condition. Short circuit in the field device or wiring.

Figure 13-20 Input troubleshooting guide.

Output Malfunctions

In addition to the logic indicator, some output modules incorporate either a blown fuse indicator or a power indicator or both. A blown fuse indicator indicates the status of the protective fuse in the output circuit, while a power indicator shows that power is being applied to the load.

Page 279

Electronic protection, as shown in Figure 13-21, is also used to provide protection for the modules from short-circuit and overload current conditions. The protection is based on a thermal cut-out principle. In the event of a short-circuit or overload current condition on an output channel, that channel will limit current within milliseconds after its thermal cut-out temperature has been reached. All other channels continue to operate as directed by the processor.

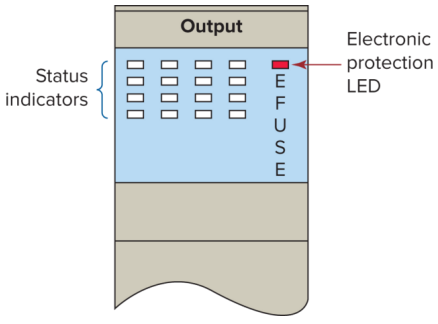


Figure 13-21 Electronic output module protection.

When an output does not energize as expected, first check the output module blown fuse indicator. Many output modules have each output fused. This indicator will normally illuminate only when the output circuit corresponding to the blown fuse is energized. If this indicator is illuminated, correct the cause of the malfunction and replace the blown fuse in the module.

Figure 13-22 shows a typical discrete output module troubleshooting guide. In general, the following items should be noted when troubleshooting discrete output modules:

- If the blown fuse indicator is not illuminated (fuse OK), then check to see if the output device is responding to the LED status indicator.
- An output module's logic status indicator functions similarly to an input module's status indicator. When it is on, the status LED indicates that the module's logic circuitry has recognized a command from the processor to turn on.
- If an output rung is energized, the module status indicator is on, and the output device is not responding, then the wiring to the output device or the output device itself should be suspected.
- If, according to the programming device monitor, an output device is commanded to turn on but the status indicator is off, then the output module or processors may be at fault.
- Check voltage at output; if incorrect, power supply, wiring, or output device may be faulty.

Output device troubleshooting guide			
Output device condition	Output module status indicator	Monitor display status indicator	Fault(s)
Energized — ON	ON	True	None - correct indication
De-energized — OFF	OFF	False	None - correct indication
De-energized — OFF	ON	True	Output instruction and status indicator agree but the field device does not. Open field device or wiring. Module circuit or fuse.
De-energized — OFF	OFF	True	Field device status and status indicator agree but the output condition does not. Module circuit or fuse.

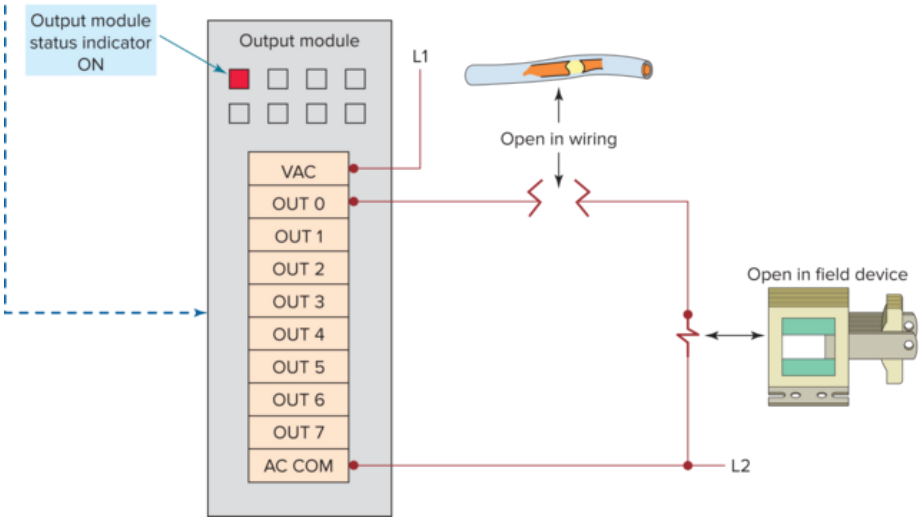


Figure 13-22 Output troubleshooting guide.

Ladder Logic Program

Many PLC software programs offer various software checks used to verify program logic.

Figure 13-23 shows a sample of verifying program errors using RSLogix 500 software. Selecting **edit** then **verify project** will check the program for errors. The sample shows what the error message might look like.

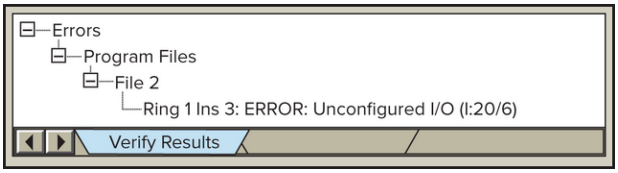


Figure 13-23 Sample of verifying program errors.

The ladder logic program itself is not likely to fail, assuming that the program was at one time working correctly. A hardware fault in the memory IC that holds the ladder logic program could alter the program, but this is a PLC hardware failure. If all other possible sources of trouble have been eliminated, the ladder logic program should be reloaded into the PLC from the master copy of the program. Make sure the master copy of the program is up to date before you download it to the PLC.

Start program troubleshooting by identifying which outputs operate properly and which outputs do not. Then trace back from the output on the nonfunctioning rung and examine the logic to determine what may be preventing the output from energizing. Common logic errors include:

- Programming an examine if closed instruction instead of an examine if open (or vice versa)
- Using an incorrect address in the program

Although the ladder logic program is not likely to fail, the process may be in a state that was unaccounted for in the original program and thus is not controlled properly. In this case, the program needs to be modified to include this new state. A careful examination of the description of the control system and the ladder logic program can help identify this type of fault.

The force on and force off instructions allow you to turn specific bits on or off for testing purposes. **Figure 13-24** illustrates how forces are identified as being enabled or disabled in RSLogix 500 software. Forcing lets you simulate operation or control an output device. For example, forcing a solenoid valve on will tell you immediately whether the solenoid is functional when the program is bypassed. If it is, the problem must be related to the software and not the hardware. If the output fails to respond when forced, either the actual output module is causing the problem or the solenoid itself is malfunctioning. ***Take all necessary precautions to protect personnel and equipment during forcing.***



Figure 13-24 Indication of enabled forces.

Source: Courtesy of TheLearningPit

Certain diagnostic instructions may be included as part of a PLC's instruction set for troubleshooting purposes. The ***temporary end (TND)*** instruction, **Figure 13-25**, is used when you want to change the amount of logic scanned to progressively debug your program. The operation of this output instruction can be summarized as follows:

- The instruction operates only when its rung conditions are true and stops the processor from scanning any logic beyond the TND instruction.

- When the processor encounters a true TND rung, it resets the watchdog timer (to 0), performs an I/O update, and begins running the ladder program at the first instruction in the main program.
- If the TND rung is false, the processor continues the scan until the next TND instruction or the END statement.
- By inserting the TND instruction at different locations in the program you can test parts of the program sequentially until the entire program has been tested.
- Once the troubleshooting process has been completed, any remaining TND instructions are removed from the program.

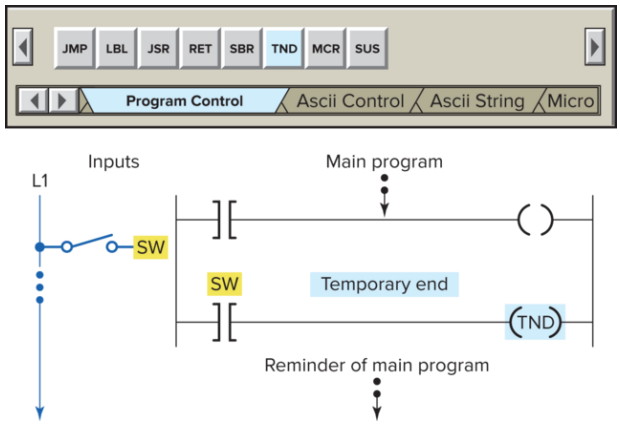


Figure 13-25 TND (temporary end) diagnostic instruction.

The ***suspend (SUS)*** instruction, shown in **Figure 13-26**, is used to trap and identify specific conditions for program debugging and system troubleshooting. The operation of this output instruction can be summarized as follows:

- When the rung is true, this instruction places the controller in the *suspend* or *idle* mode.
- The suspend ID, in this case 100, must be selected by the programmer and entered in the instruction.
- When the SUS instruction executes, the ID number 100 is written in word 7 (S2:7) of the status file.
- If multiple suspend instructions are present, then this will indicate which SUS instruction was active.
- The suspend file (program or subroutine number identifying where the executed SUS instruction resides) is placed in word 8 (S2:8) of the status file.
- All ladder logic outputs are de-energized, but other status files have the data present when the suspend instruction was executed.

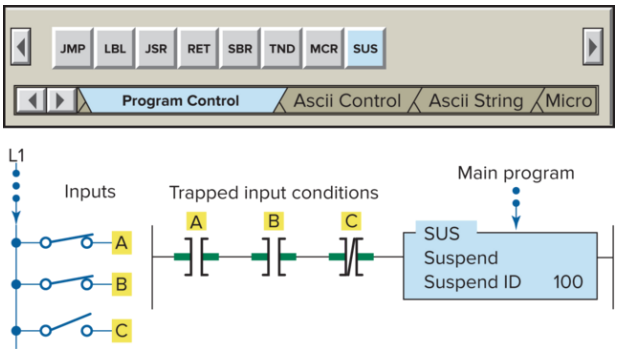


Figure 13-26 SUS (suspend) diagnostic instruction.

Most PLC system faults occur in the field wiring and devices. The wiring between the field devices and the terminals of the I/O modules is a likely place for problems to occur. Faulty wiring and mechanical connection problems can interrupt or short the signals sent to and from the I/O modules.

The sensors and actuators connected to the I/O of the process can also fail. Mechanical switches can wear out or be damaged during normal operation. Motors, heaters, lights, and sensors can also fail. Input and output field devices must be compatible with the I/O module to ensure proper operation.

When an instruction does not seem to be working correctly, the problem may be an addressing conflict caused by the *same address* being used for two or more coil instructions in the same program. As a result, multiple rung conditions can control the same output coil, making troubleshooting more difficult. In the case of duplicate outputs, the monitored rung may be true; but if a rung farther down in the ladder diagram is false, the PLC will keep the output off. The program of **Figure 13-27** illustrates what happens when the same address is used for two coils. The resulting problem scenario can be summarized as follows:

- The problem is turning input switch I:1/1 on *will not* turn on PL output O:2/1 as it appears to be programmed.
- The root of the problem lies in the fact that the PLC scans the program from left to right and top to bottom.
- Whenever input switch I:1/1 is true (closed) and input switch I:1/2 is false (open) output O:2/1 will be off.
- This is because when the PLC updates the outputs it does so based on the status of input I:1/2.
- Regardless of whether input I:1/1 is open or closed the output reacts only to the status of input switch I:1/2.

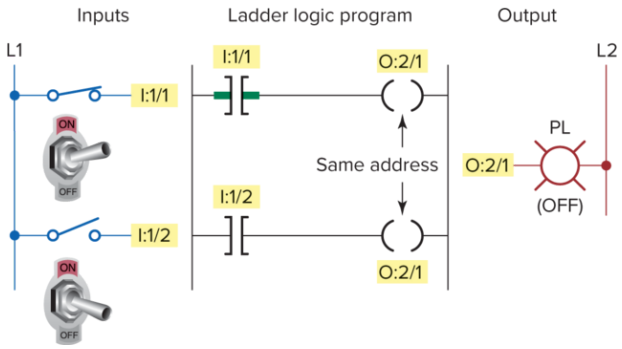


Figure 13-27 Program with the same address used for two coils.

When a problem occurs, the best way to proceed is to try to logically identify the devices or connections that could be causing the problem rather than arbitrarily checking every connection, switch, motor, sensor, I/O module, and so on. First, observe the system in operation and try to describe the problem. Using these observations and the description of the control system, you should identify the possible sources of trouble. Compare the logic status of the hardwired inputs and outputs to their actual state, as illustrated in **Figure 13-28**. Any disagreements indicate malfunctions as well as their approximate location.

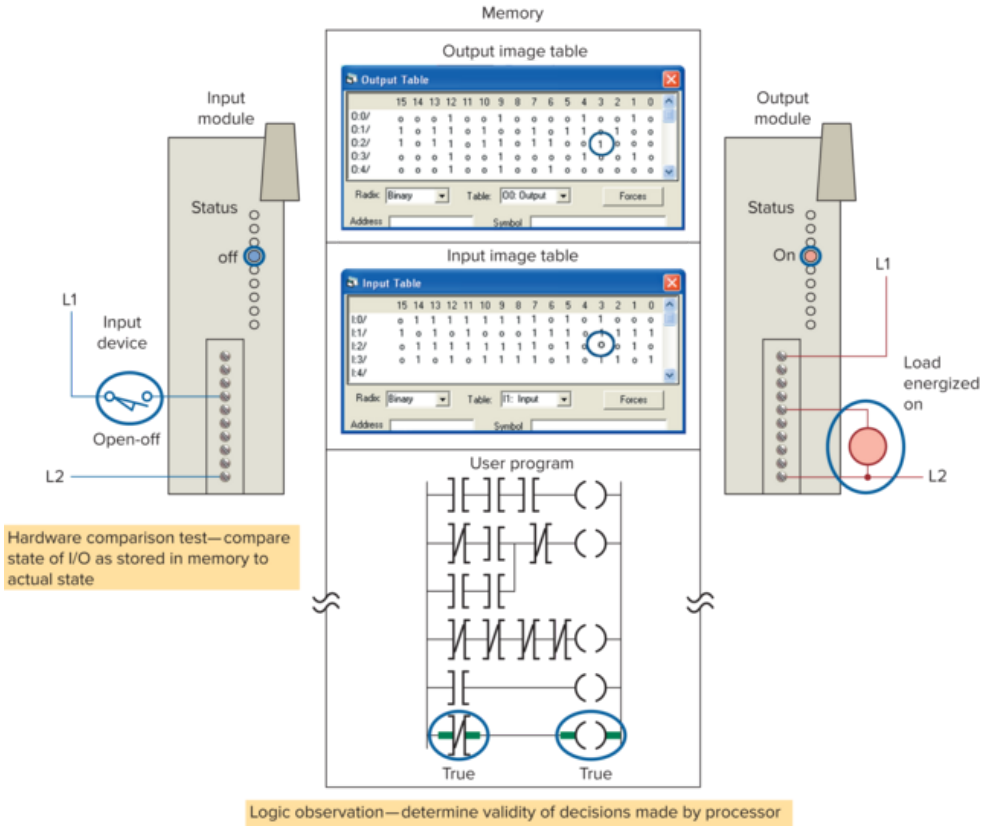


Figure 13-28 General methods of troubleshooting.

Source: Courtesy of TheLearningPit

Some of your troubleshooting can be accomplished by interpreting the status indicators on the I/O modules. The key is to know whether the status indicators are telling you that there is a fault or that the system is normal. Often PLC manufacturers supply a troubleshooting guide, map, or tree that presents a list of observed problems and their possible sources. **Figure 13-29** shows a sample troubleshooting tree for a discrete output module. **Figure 13-30** and **Figure 13-31** are samples of input and output troubleshooting guides.

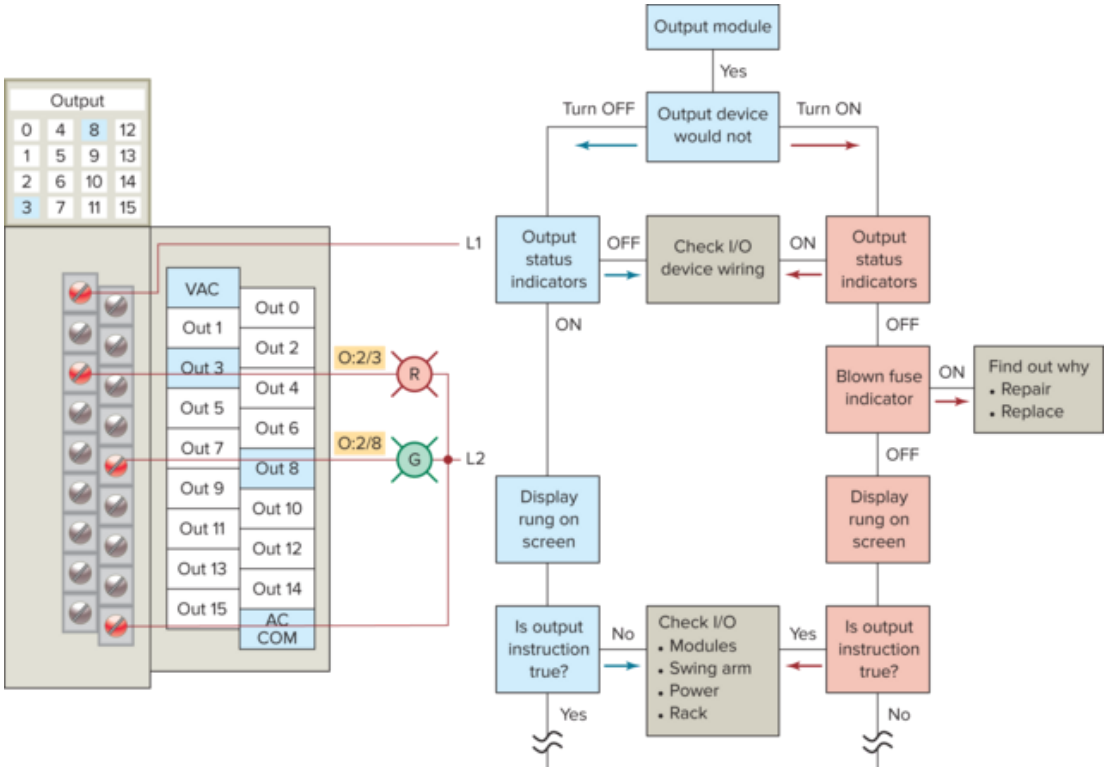


Figure 13-29 Troubleshooting tree for a discrete output module.

If Your Input Circuit LED Is . . .	And Your Input Device Is . . .	And	Probable Cause
ON 	On/Closed/Activated 	Your input device will not turn off.	Device is shorted or damaged.
		Your program operates as though it is off.	Input circuit wiring or module.
			Input is forced off in program.
	Off/Open/Deactivated 	Your program operates as though it is on and/or the input circuit will not turn off.	Input device off-state leakage current exceeds input circuit specification.
			Input device is shorted or damaged.
			Input circuit wiring or module.
OFF 	On/Closed/Activated 	Your program operates as though it is off and/or the input circuit will not turn on.	Input circuit is incompatible.
			Low voltage across the input.
			Input circuit wiring or module.
			Input signal turn-on time too fast for input circuit.
	Off/Open/Deactivated 	Your input device will not turn on.	Input device is shorted or damaged.
		Your program operates as though it is on.	Input is forced on in program.
			Input circuit wiring or module.

Figure 13-30 Input troubleshooting guide.

If Your Output Circuit LED Is . . .	And Your Output Device Is . . .	And	Probable Cause
ON 	On/Energized 	Your program indicates that the output circuit is off or the output circuit will not turn off. 	Programming problem: - Check for duplicate outputs and addresses. - If using subroutines, outputs are left in their last state when not executing subroutines. - Use the force function to force output off. If this does not force the output off, output circuit is damaged. If the output does force off, then check again for logic/programming problem.
			Output is forced on in program.
			Output circuit wiring or module.
	Off/De-energized 	Your output device will not turn on and the program indicates that it is on. 	Low or no voltage across the load.
			Output device is incompatible: check specifications and sink/source compatibility (if dc output).
			Output circuit wiring or module.
OFF 	On/Energized 	Your output device will not turn off and the program indicates that it is off. 	Output device is incompatible.
			Output circuit off-state leakage current may exceed output device specification.
			Output circuit wiring or module.
			Output device is shorted or damaged.
	Off/De-energized 	Your program indicates that the output circuit is on or the output circuit will not turn on. 	Programming problem: - Check for duplicate outputs and addresses. - If using subroutines, outputs are left in their last state when not executing subroutines. - Use the force function to force output on. If this does not force the output on, output circuit is damaged. If the output does force on, then check again for logic/programming problem.
			Output is forced off in program.
			Output circuit wiring or module.

Figure 13-31 Output troubleshooting guide.



OceanofPDF.com

13.10 PLC Programming Software

You must establish a way for your personal computer (PC) software to communicate with the programmable logic controller (PLC) processor. Making this connection is known as ***configuring* the communications.** The method used to configure the communications varies with each brand of controller. In Allen-Bradley controllers, ***RSLogix*** software is required to develop and edit ladder programs. A second software package, ***RSLinx,*** is needed to set up the communication path between the PLC processor and the personal computer (PC). You cannot download multiple projects to the PLC and then run them when required. The PLC will accept only one program at a time, but the program can consist of multiple subroutine files which can be conditionally called from the main program.

Page 285

RSLinx software is available in multiple packages to meet the demand for a variety of cost and functionality requirements. This software package is used as the driver between your PC and PLC processor. A ***driver*** is a computer program that controls a device. For example, you must have the correct printer driver installed in your PC in order to be able to print a word-processing document created on your PC. RSLinx works much like the printer driver for RSLogix software. The RSLinx program must be opened and drivers configured before communications can be established between a PC and a PLC that is using RSLogix software.

RSLinx allows RSLogix to communicate through an interface cable to the PLC processor. The simplest connection between a PC and a PLC is a point-to-point direct connection through the computer serial port, as illustrated in **Figure 13-32**. A serial cable is used to connect to your PC's COM 1 or COM 2 port and to the PLC processor's serial communications port. With RSLinx software you can auto-configure the serial connection and thus automatically find the proper serial port configuration.

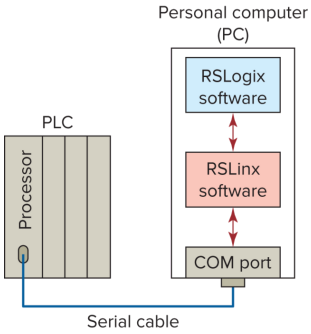


Figure 13-32 Direct PC-to-PLC software connection.

Two important aspects of the communication link must be considered, namely, the RS-232 standard and the communications protocol. The ***RS-232 standard*** specifies a function for each of the wires inside the standard communications cable and their associated pins. ***Communications protocol*** is a

standardized method for transmitting data and/or establishing communications between different devices.

Minimum configuration for two-way communications requires the use of only three connected wires, as shown in Figure 13-33. For ease of connection, the RS-232 standard specifies that computer devices have male connectors and that peripheral equipment have female connectors. Direct communication between two computers, such as a PC and a PLC, does not involve intermediate peripheral equipment. Therefore, a serial null-modem type cable must be used for the connection because both the PC and the PLC processor use pin 2 for data output and pin 3 for data input.

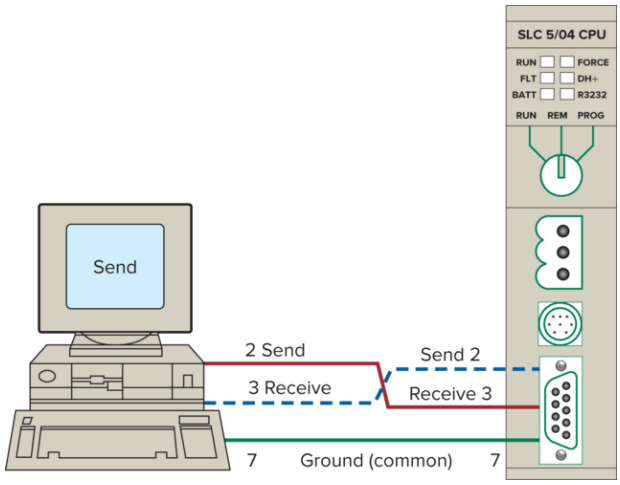


Figure 13-33 Serial wiring connection.

RSLinx is a Windows based communication software package developed by Rockwell Software to interface to all of the Rockwell and A-B industrial control and automation hardware. To setup RSLinx:

- Open RSLinx.
- Click on Communications and select configure drivers from the dialog box.
- Select RS-232 from the available driver types.
- Click on add new and select your communications port from the dialog box.
- Click on auto configure to automatically set the baud rate and parity.
- Minimize the RSLinx window, but leave the program running.

RSWho is RSLinx's network browser interface. The who active or RSWho function, Figure 13-34, shows you what stations are connected on your PLC network. The left pane of RSWho is the tree control, which shows networks and devices. The right pane is the list control, which shows all members of a collection. A collection is a network, or a device that is a bridge. A device that appears with a red X through it indicates a communication status error, such as loss of power to a device or a disconnected communication cable.

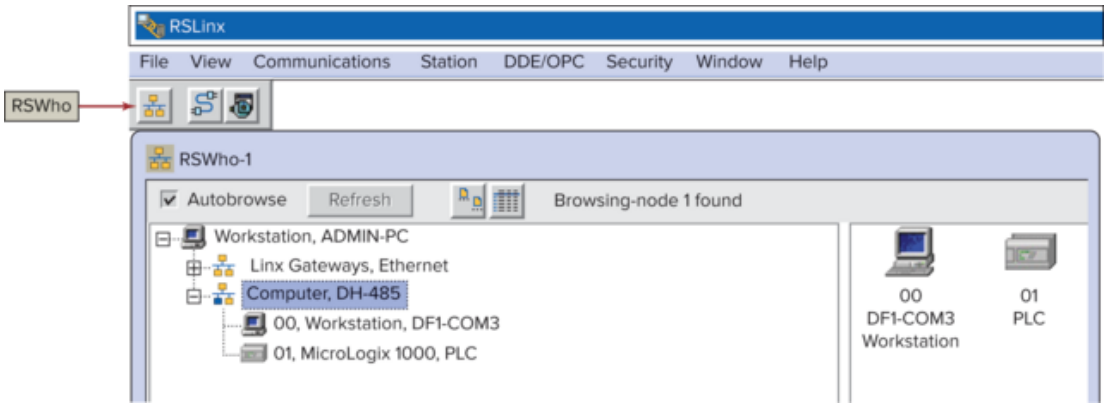


Figure 13-34 The who active or RSWho window.

Source: Image Courtesy of Rockwell Automation, Inc.

OceanofPDF.com



CHAPTER 13 REVIEW QUESTIONS

1. Why are PLCs installed within an enclosure?
2. What methods are used to keep enclosure temperatures within allowable limits?
3. State two ways in which electrical noise may be coupled into a PLC control system.
4. List three potential noise-generating inductive devices.
5. Describe four ways in which careful wire routing can help cut down on electrical noise.
6. **a.** What type of input field devices are most likely to have a small leakage current flow when they are in the off state? Why?
   b. Explain how an input bleeder resistor reduces leakage current.
7. Summarize the basic grounding requirements for a PLC system.
8. Under what condition can a ground loop circuit be developed?
9. When line voltage variations to the PLC power supply are excessive, what can be done to solve the problem?
10. What operating state will cause an inductive load to generate a very high voltage spike?
11. Explain how a diode is connected to suppress a DC inductive load.
12. Explain how an MOV suppresses an AC inductive load.
13. What is the purpose of PLC editing functions?
14. What is involved with commissioning a PLC system?
15. **a.** Compare offline and online programming.
    b. Which method is safer? Why?
16. List four uses for the data monitor function.
17. What information is provided by the cross reference function?
18. What information is provided by the contact histogram function?
19. List five preventive maintenance tasks that should be carried out on the PLC installation regularly.
20. Outline the general procedure followed to lock out and tag a PLC installation.
21. Typically, what does each of the following processor diagnostic light states indicate?
    a. RUN light is off.
    b. Fault light is off.
    c. BATTERY light is on.
22. Explain the function of a watchdog timer circuit.
23. A PLC is operating in the RUN mode but output devices do not operate as programmed. List faults that could be responsible for this condition.

24. What is the verify results function used for?

25. A fast-acting solenoid-operated gate is suspected of not functioning properly when energized and de-energized by the PLC program. Explain how you would use the force function to check its operation.

26. What happens when the processor encounters a temporary end instruction?

27. Explain the function of the suspend instruction.

28. In what negative ways can faulty wiring and connections affect signals sent to and from the I/O modules?

29. The same address is used for two coil instructions within the same PLC program. What will happen as a consequence of this?

30. Compare the uses for RSLogix and RSLinx programming software.

OceanofPDF.com



CHAPTER 13 PROBLEMS

1. The enclosure door of a PLC installation is not kept closed. What potential problem could this create?
2. A fuse is blown in an output module. Suggest two possible reasons why the fuse blew.
3. Whenever a crane located over a PLC installation is started from a standstill, temporary malfunction of the PLC system occurs. What is one likely cause of the problem?

Page 288

4. During the static checkout of a PLC system, a specific output is forced on by the programming device. If an indicator other than the expected one turns on, what is the probable problem?
5. The input device to a module is activated, but the LED status indicator does not come on. A check of the voltage to the input module indicates that no voltage is present. Suggest two possible causes of the problem.
6. An output is forced on. The module logic light comes on, but the field device does not work. A check of the voltage on the output module indicates the proper voltage level. Suggest two possible causes of the problem.
7. A specific output is forced on, but the LED module indicator does not come on. A check of the voltage at the output module indicates a voltage far below the normal on level. What is the first thing to check?
8. An electronic-based input sensor is wired to a high-impedance PLC input and is falsely activating the input. How can this problem be corrected?
9. An LED logic indicator is illuminated, and according to the programming device monitor, the processor is not recognizing the input. If a replacement module does not eliminate the problem, what two other items should be suspected?
10. **a.** A normally open field limit switch examined for an on state normally cycles from on to off five times during one machine cycle. How could you tell by observing the LED status light that the limit switch is functioning properly?
    b. How could you tell by observing the programming device monitor that the limit switch is functioning properly?
    c. How could you tell by observing the LED status light whether the limit switch was stuck open?
    d. How could you tell by observing the programming device monitor whether the limit switch was stuck open?
    e. How could you tell by observing the LED status light if the limit switch was stuck closed?
    f. How could you tell by observing the programming device monitor if the limit switch was stuck closed?

11. Assume that prior to putting a PLC system into operation, you want to verify that each *input device* is connected to the correct input terminal and that the input module or point is functioning properly. Outline a method of carrying out this test.

12. Assume that prior to putting a PLC system into operation, you want to verify that each *output device* is connected to the correct output terminal and that the output module or point is functioning properly. Outline a method of carrying out this test.

13. With reference to the ladder logic program of **Figure 13-35**, add instructions to modify the program to ensure that the second pump_2 does not run while pump_1 is running. If this condition occurs, the program should suspend operation and enter code identification number 100 into S2:7.

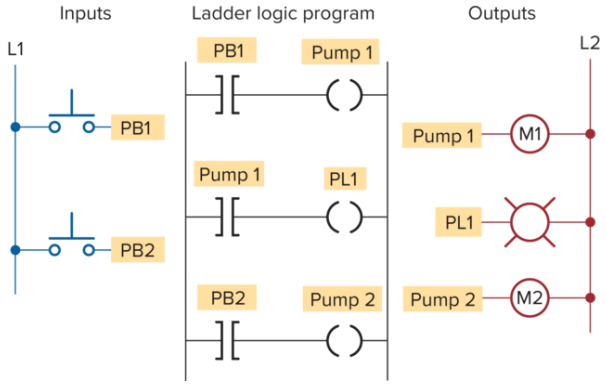


Figure 13-35 Program for Problem 13.

14. The program of **Figure 13-36** is supposed to execute to sequentially turn PL1 off for 5 seconds and on for 10 seconds whenever input *A* is closed.

a. Examine the ladder logic and describe how the circuit would operate as programmed.

b. Troubleshoot the program and identify what needs to be changed to have it operate properly.

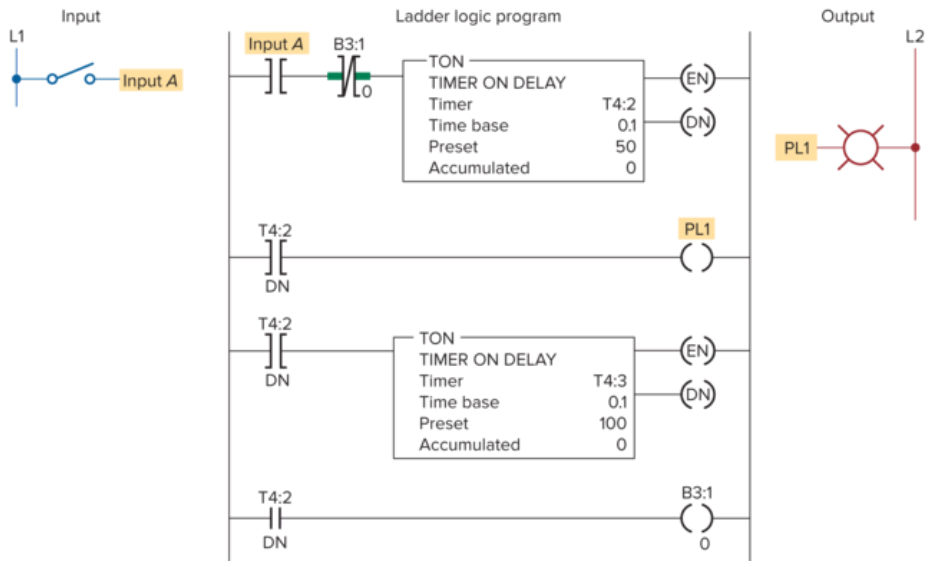
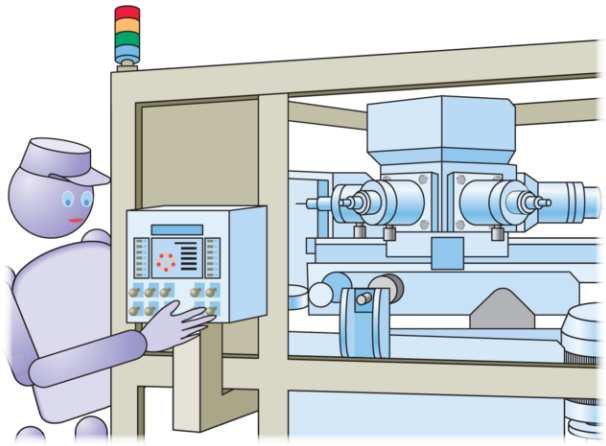


Figure 13-36 Program for Problem 14.

OceanofPDF.com

14

Process Control, Network Systems, and SCADA



Chapter Objectives

After completing this chapter, you will be able to:

- Discuss the operation of continuous process, batch production, and discrete manufacturing processes
- Compare individual, centralized, and distributive control systems
- Explain the functions of the major components of a process control system
- Describe the various functions of electronic HMI screens
- Recognize and explain the functions of the control elements of a closed-loop control system
- Explain how on/off control works
- Explain how PID control works
- Identify the levels of an industrial network
- Discuss the different types of network architecture and protocols
- Describe a typical SCADA application
- Be familiar with the component parts of an HMI terminal
- Explain the process by which HMI graphic objects control a related PLC program
- Be familiar with protocols, severs and drivers associated with a HMI/PLC application

This chapter introduces the kinds of industrial processes that can be PLC controlled. SCADA is such a process. Different types of control systems are used for complex processes. These control systems may be PLCs, but other controllers include robots, data terminals, and computers. For these controllers to work together, they must communicate. This chapter will discuss the different kinds of industrial processes and the means by which they communicate.

OceanofPDF.com

14.1 Types of Processes

Process control is the automated control of a process. Such systems typically deal with analog signals from sensors. The ability of a PLC to perform math functions and utilize analog signals makes it ideally suited for this type of control. Manufacturing is based on a series of processes being applied to raw materials. Typical applications of process control systems include automobile assembly, petrochemical production, oil refining, power generation, and food processing.

A ***continuous process*** is one in which raw materials enter one end of the system and the finished product comes out the other end of the system; the process itself runs continuously. **Figure 14-1** shows a continuous process used in an automotive engine assembly line. Parts are mounted sequentially, in an assembly-line fashion, through a series of stations. Assembly and adjustments are carried out by both automated machine and manual operations.

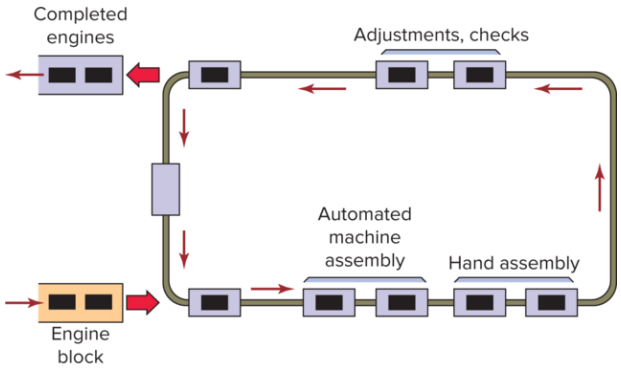


Figure 14-1 Continuous process.

In ***batch processing,*** there is no flow of product material from one section of the process to another. Instead, a set amount of each of the inputs to the process is received in a batch, and then some operation is performed on the batch to produce a product. Products produced using the batch process include food, beverages, pharmaceutical products, paint, and fertilizer. **Figure 14-2** shows an example of a batch process. Three ingredients are mixed together, heated, and then stored. Recipes are the key to producing batches as each batch may have different characteristics by design.

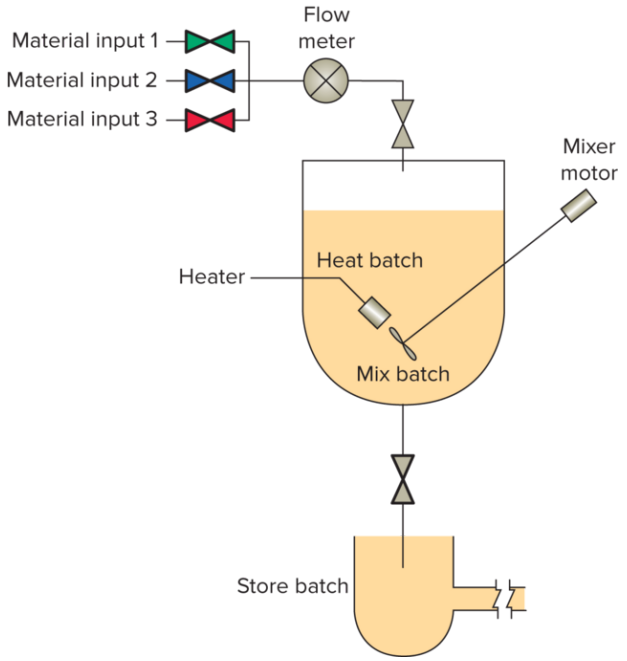


Figure 14-2 Batch process.

Discrete manufacturing is characterized by individual or separate unit production. With this manufacturing process, a series of operations produces a useful output product. Discrete manufacturing systems typically deal with digital inputs to PLCs that cause motors and robotic devices to be activated. The work piece is normally a discrete part that must be handled on an individual basis.

Possible control configurations include individual, centralized, and distributed. ***Individual control*** is used to control a single machine. This type of control does not normally require communication with other controllers. **Figure 14-3** shows an individual control application for a cut to length operation. The operator enters the feed length and batch count via the interface control panel and then presses the start button to initiate the process. Stock lengths vary, so the operator needs to select the length and the number of pieces to be cut.

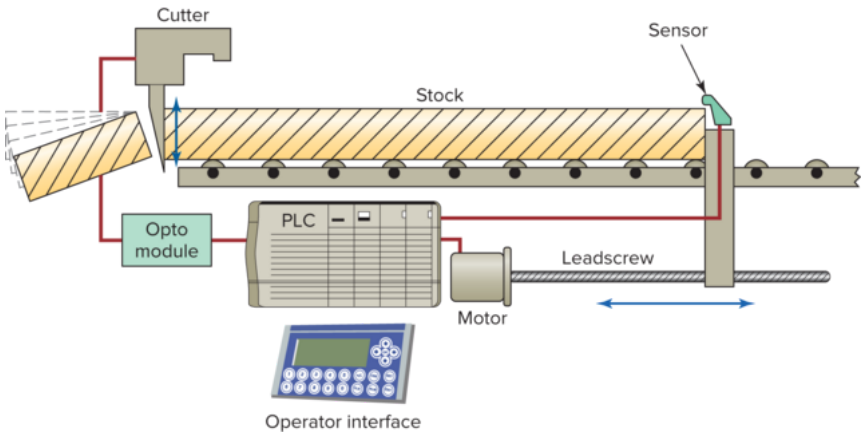


Figure 14-3 Individual control.

Centralized control is used when several machines or processes are controlled by one central controller. The control layout uses a single, large control system to control many diverse manufacturing processes and operations, as illustrated in **Figure 14-4**. The main features of centralized control can be summarized as follows:

- All individual steps in the manufacturing process are handled by a central control system controller.
- No exchange of controller status or data is sent to other controllers.
- If the main controller fails, the whole process stops.

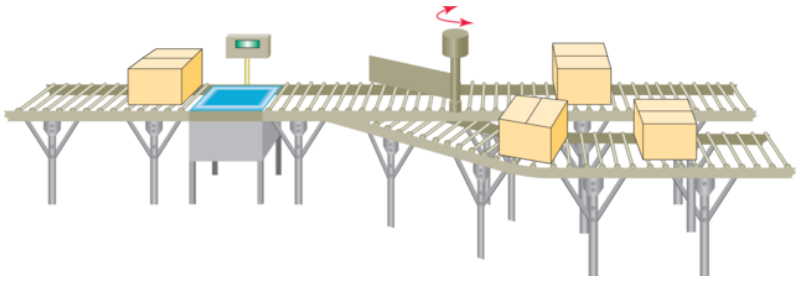


Figure 14-4 Centralized control.

A ***distributive control system (DCS)*** is a network-based system. Distributive control involves two or more PLCs communicating with each other to accomplish the complete control task, as illustrated in **Figure 14-5**. Each PLC controls different processes locally and the PLCs are constantly exchanging information through the communications link and reporting on the status of the process. The main features of a distributive control system can be summarized as follows:

- Distributive control permits the distribution of the processing tasks among several controllers.
- Each PLC controls its associated machine or process.
- High-speed communication among the computers is done through CAT-5 or CAT-6 twisted pair wires, single coaxial cables, fiber optics, or the Ethernet.
- Distributive control drastically reduces field wiring and heightens performance because it places the controller and I/O close to the machine process being controlled.
- Depending on the process, one PLC failure would not necessarily halt the complete process.
- DCS is supervised by a host computer that may perform monitoring/supervising functions such as report generation and storage of data.

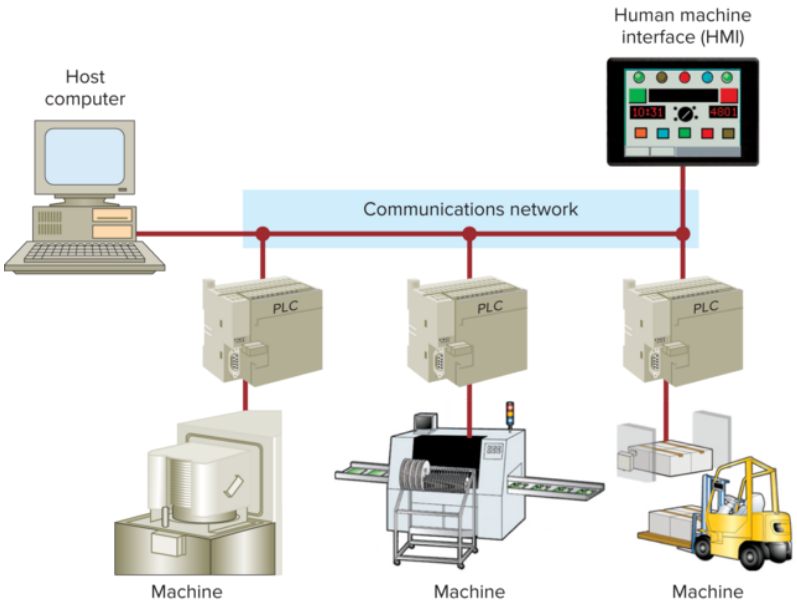


Figure 14-5 Distributive control system (DCS).

OceanofPDF.com

14.2 Structure of Control Systems

Process control normally applies to the manufacturing or processing of products in industry. In the case of a programmable controller, the process or machine is operated and supervised under the control of the user program. The major components of a process control system include the following:

Sensors

- Provide inputs from the process and from the external environment
- Convert physical information such as pressure, temperature, flow rate, and position into electrical signals

Human Machine Interface (HMI)

- Allows human inputs through various types of programmed switches, controls, and keypads to set up the starting conditions or alter the control of a process

Signal Conditioning

- Involves converting input and output signals to a usable form
- May include signal-conditioning techniques such as amplification, attenuation, filtering, scaling, A/D and D/A converters

Actuators

- Convert system output electrical signals into physical action
- Process actuators that include flow control valves, pumps, positioning drives, variable speed drives, clutches, brakes, solenoids, stepping motors, and power relays

Controller

- Makes the system's decisions based on the input signals
- Generates output signals that operate actuators to carry out the decisions

Human machine interface (HMI) equipment provides a control and visualization interface between a human and a process (**Figure 14-6**). HMIs allow operators to control, monitor, diagnose, and manage the application. Depending on the requirements and complexity of the process, the operator may be required to:

- Stop and start the process.
- Operate the controls and make the adjustments required for the process and monitor its progress.
- Detect abnormal situations and undertake corrective action.

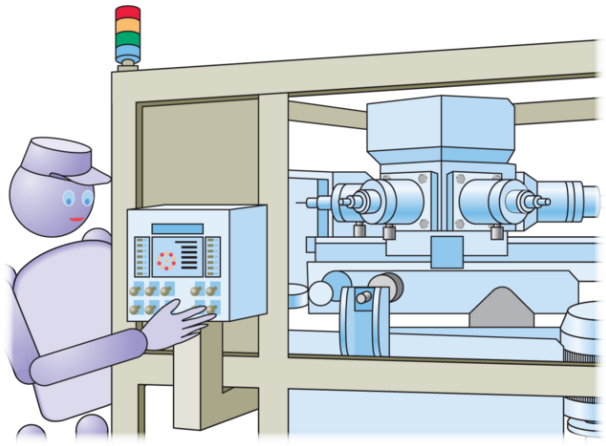


Figure 14-6 Human machine interface (HMI).

Graphic HMI terminals offer electronic interfacing in a wide variety of sizes and configurations. They replace traditional wired panels with a touch screen with graphical representations of switches and indicators. Types of graphical display screens include the following:

Operational Summary—used to monitor the process.

Configuration/Setup—textual in nature, used to detail process parameters.

Alarm Summary—provides a list of time-stamped active alarms.

Event History—presents a time-stamped list of all significant events that have occurred in the process.

Page 294

Trend Values—displays information on process variables, such as flow, temperature, and production rate, over a period of time.

Manual Control—generally available only to maintenance personnel and meant to bypass parts of the automatic control system.

Diagnostics—used by maintenance personnel to diagnose equipment failures.

Graphic terminals come fully packaged with hardware, software, and communications. **Figure 14-7** shows the Allen-Bradley family of graphic terminals. The setup varies with the vendor. In general, the tasks required to develop an HMI application include:

- Establish a communication link with the PLCs.
- Create the tag addresses database.
- Edit and create graphical objects on the screens.
- Animate the objects.



Figure 14-7 PanelView graphic terminals.
Source: Image Courtesy of Rockwell Automation, Inc.

Most control systems are closed loop in that they utilize feedback in which the output of a process affects the input control signal. A closed-loop system measures the actual output of the process and compares it to the desired output. Adjustments are made continuously by the control system until the difference between the desired and actual output falls within a predetermined tolerance.

Figure 14-8 illustrates an example of a closed-loop control system. The actual output is sensed and fed back to be subtracted from the set-point input that indicates what output is desired. If a difference occurs, a signal to the controller causes it to take action to change the actual output until the difference is 0. The operation of the component parts are as follows:

Set-point—The input that determines the desired operating point for the process.

Process Variable—Refers to the feedback signal that contains information about the current process status.

Error Amplifier—Determines whether the process operation matches the set-point. The magnitude and polarity of the error signal will determine how the process will be brought back under control.

Controller—Produces the appropriate corrective output signal based on the error signal input.

Output Actuator—The component that directly affects a process change. Examples are motors, heaters, fans, and solenoids.

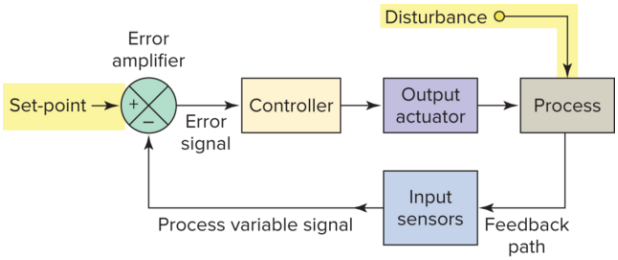


Figure 14-8 Closed-loop control system.

The process shown in **Figure 14-9** is an example of a closed-loop continuous control process used to automatically fill box containers to a specified weight of detergent. An empty box is moved into position and filling begins. The weight of the box and contents is monitored. When the actual weight equals the desired weight, filling is halted.

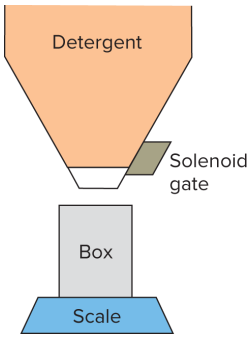


Figure 14-9 Container-filling process.

Operation and block diagrams for the container-filling process are shown in **Figure 14-10**. The operation of the process can be summarized as follows:

- A sensor attached to the scale weighing the container generates the voltage signal or digital code that represents the weight of the container and contents.
- The sensor signal is subtracted from the voltage signal or digital code that has been input to represent the desired weight.
- As long as the difference between the input signal and feedback signal is greater than 0, the controller keeps the solenoid gate open.
- When the difference becomes 0, the controller outputs a signal that closes the gate.

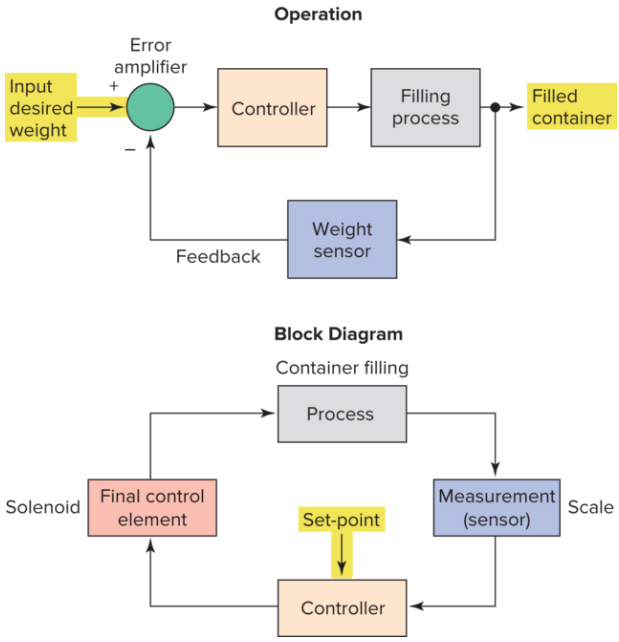


Figure 14-10 Operation and block diagrams for the container-filling process.

Virtually all feedback controllers determine their output by observing the error between the set-point and a measurement of the process variable. Errors can occur when an operator changes the set-point or when a disturbance or a load on the process changes the process variable. The controller's role is to eliminate the error automatically.

OceanofPDF.com

14.3 On/Off Control

With ***on/off controllers*** the final control element is either on or off—one for the occasion when the value of the measured variable is above the set-point and the other for the occasion when the value is below the set-point. The controller will never keep the final control element in an intermediate position. Controlling activity is achieved by the period of on-off cycling action.

Figure 14-11 shows a system using on/off control in which a liquid is heated by steam. The operation of the process can be summarized as follows:

- If the liquid temperature goes below the set-point, the steam valve opens and the steam is turned on.
- When the liquid temperature goes above the set-point, the steam valve closes and the steam is shut off.
- The on/off cycle will continue as long as the system is operating.

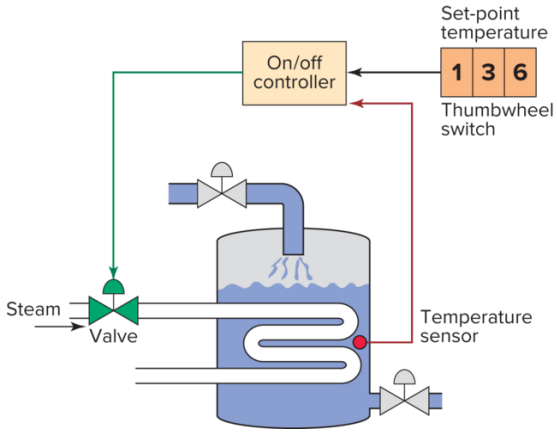


Figure 14-11 On/off controlled liquid heating system.

Figure 14-12 illustrates the control response for an on/off temperature controller. The action of the control response can be summarized as follows:

- The output turns on when the temperature falls below the set-point and turns off when the temperature reaches the set-point.

- Control is simple, but overshoot and cycling about the set-point can be disadvantageous in some processes.
- The measured variable will oscillate around the set-point at an amplitude and frequency that depend on the capacity and time response of the process.
- Oscillations may be reduced in amplitude by increasing the sensitivity of the controller. This increase will cause the controller to turn on and off more often, a possibly undesirable result.
- On/off control is used when a more precise control is unnecessary.

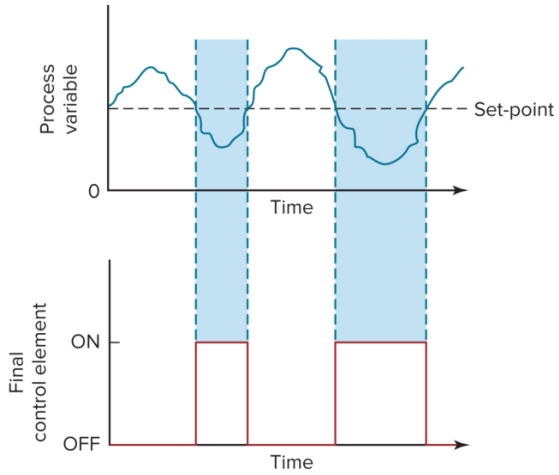


Figure 14-12 On/off control response.

A ***deadband*** is usually established around the set-point. The deadband of the controller is usually a selectable value that determines the error range above and below the set-point that will not produce an output as long as the process variable is within the set limits. The inclusion of deadband eliminates any hunting by the control device around the set-point. **Hunting** occurs when minor adjustments of the controlled position are continually made due to minor fluctuations.

OceanofPDF.com

14.4 PID Control

Proportional controllers are designed to eliminate the hunting or cycling associated with on/off control. They allow the final control element to take intermediate positions between on and off. Proportioning action permits ***analog control*** of the final control element to vary the amount of energy to the process, depending on how much the value of the measured variable has shifted from the desired value.

A proportional controller allows tighter control of the process variable because its output can take on any value between fully on and fully off, depending on the magnitude of the error signal.

Figure 14-13 shows an example of a motor-driven analog proportional control valve used as a final control element. The action of the control valve actuator can be summarized as follows:

- The actuator receives an input current between 4 and 20 mA from the controller.
- In response, it provides linear control of the valve.
- A value of 4 mA corresponds to a minimum value opening (often 0) and 20 mA corresponds to a maximum value opening (full scale).
- The 4 mA lower limit allows the system to detect opens. If the circuit is open, 0 mA would result, and the system can issue an alarm.
- Because the signal is a current, it is unaffected by reasonable variations in connecting wire resistance and is less susceptible to noise pickup from other signals than is a voltage signal.

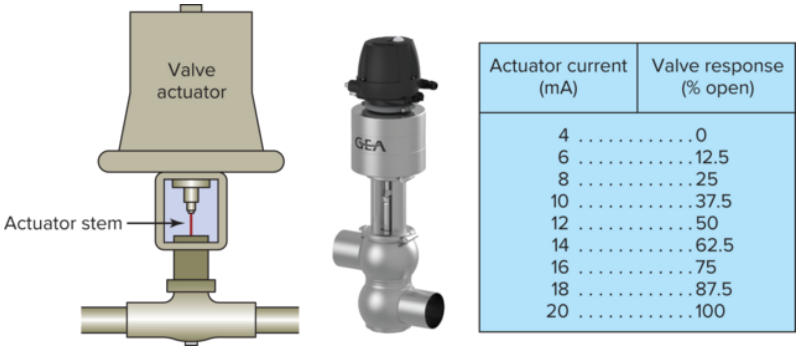


Figure 14-13 Motor-driven analog proportional control valve.

Source: Photo Courtesy Gea Group

Proportioning action can also be accomplished by turning the final control element on and off for short intervals. This ***time proportioning*** (also known as ***pulse width modulation***) varies the ratio of on time to off. **Figure 14-14** shows an example of time proportioning used to produce varying wattage from a 200 watt heater element as follows:

- To produce 100 W the heater must be on 50% of the time.
- To produce 50 W the heater must be on 25% of the time.
- To produce 25 W the heater must be on 12.5% of the time.

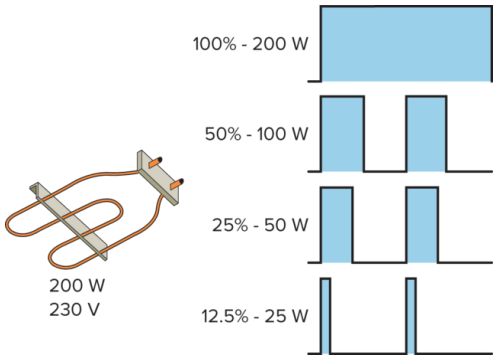


Figure 14-14 Time proportioning of a heater element.

Proportioning action occurs within a proportional band around the set-point. The table of **Figure 14-15** is an example of the proportional band for a heating application with a set-point of $500^{\circ}\mathrm{F}$ and a proportional band of $80^{\circ}\mathrm{F}(\pm 40^{\circ}\mathrm{F})$. Proportioning action can be summarized as follows:

- Proportional controllers have analog input and output values that vary over the range necessary for control of the process.
- The current temperature is called the Process Variable (PV), while the desired temperature is known as the Set-Point (SP).
- The proportional controller changes the controller output in proportion to the difference between the SP and PV. The greater the difference, the greater the corrective action applied.
- At the set-point (the midpoint of the proportional band), the output on:off ratio is 1:1; that is, the on time and off time are equal.
- If the temperature is further from the set-point, the on and off times vary in proportion to the temperature difference.
- If the temperature is below the set-point, the output will be on longer; if the temperature is too high, the output will be off longer.

Time proportional			Temp. (°F)	4–20 mA proportional	
Percent on	On time (seconds)	Off time (seconds)		Output level	Percent output
0.0	0.0	20.0	over 540	4 mA	0.0
0.0	0.0	20.0	540.0	4 mA	0.0
12.5	2.5	17.5	530.0	6 mA	12.5
25.0	5.0	15.0	520.0	8 mA	25.0
37.5	7.5	12.5	510.0	10 mA	37.5
50.0	10.0	10.0	500.0	12 mA	50.0
62.5	12.5	7.5	490.0	14 mA	62.5
75.0	15.0	5.0	480.0	16 mA	75.0
87.5	17.5	2.5	470.0	18 mA	87.5
100.0	20.0	0.0	460.0	20 mA	100.0
100.0	20.0	0.0	under 460	20 mA	100.0

Figure 14-15 Proportional band for a heating application.

In theory, a proportional controller should be all that is needed for process control. Any change in system output is corrected by an appropriate change in controller output. Unfortunately, the operation of a proportional controller leads to a steady-state error known as ***offset,*** or ***droop.*** This steady-state error is the difference between the attained value of the controller and the required value that results in an offset signal that is slightly lower than the set-point value, as illustrated in **Figure 14-16**. Depending on the PLC application, this offset may or may not be acceptable.

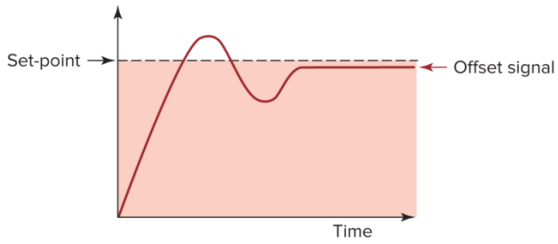


Figure 14-16 Proportional control steady-state error.

Proportional control is often used in conjunction with integral control and/or derivative control.

- The ***integral action,*** sometimes termed reset action, responds to the size and time duration of the error signal. An error signal exists when there is a difference between the process variable and the set-point, so the integral action will cause the output to change and continue to change until the error no longer exists. Integral action eliminates steady-state error. The amount of integral action is measured as minutes per repeat or repeats per minute, which is the relationship between changes and time.
- The ***derivative action*** responds to the speed at which the error signal is changing—that is, the greater the error change, the greater the correcting output. The derivative action is measured in terms of time.

Proportional plus integral (PI) control combines the characteristics of both types of control. A step change in the set-point causes the controller to respond proportionally, followed by the integral response, which is added to the proportional response. Because the integral mode determines the output change as a function of time, the more integral action found in the control, the faster the output changes. This action can be summarized as follows:

- To eliminate the offset error, the controller needs to change its output until the process variable error is zero.
- Reset integral control action changes the controller output by the amount needed to drive the process variable back to the set-point value.

- After the reset integral control action a new equilibrium point is established.
- Since the proportional controller must always operate on its proportional band, the proportional band must be shifted to include the new equilibrium point.
- A controller with reset integral control does this automatically.

Rate action (derivative control) acts on the error signal just like reset does, but rate action is a function of the rate of change rather than the magnitude of error. Rate action is applied as a change in output for a selectable time interval, usually stated in minutes. Rate-induced change in controller output is

calculated from the derivative of the error. Input change, rather than proportional control error change, is used to improve response. Rate action quickly positions the output, whereas proportional action alone would eventually position the output. In effect, rate action puts the brakes on any offset or error by quickly shifting the proportional band. ***Proportional plus derivative (PD) control*** is used in process control systems with errors that change very rapidly. By adding derivative control to proportional control, we obtain a controller output that responds to the error's rate of change as well as to its magnitude.

PID control is a feedback control method that combines proportional, integral, and derivative actions. The proportional action provides smooth control without hunting. The integral action automatically corrects offset. The derivative action responds quickly to large external disturbances. The PID controller is the most widely used type of process controller. When combined into a single control loop the proportional, integral, and derivative modes complement each other to reduce the system error to zero faster than any other controller. **Figure 14-17** shows the block diagram of a PID control loop, the operation of which can be summarized as follows:

- During setup, the set-point, proportional band, reset (integral), rate (derivative), and output limits are specified.
- All these can be changed during operation to tune the process.
- The integral term improves accuracy, and the derivative reduces overshoot for transient upsets.
- The output can be used to control valve positions, temperature, flow metering equipment, and so on.
- PID control allows the output power level to be varied.
- As an example, assume that a furnace is set at $50^{\circ}\mathrm{C}$.
- The heater power will increase as the temperature falls below the $50^{\circ}\mathrm{C}$ set-point.
- The lower the temperature the higher the power.
- PID has the effect of gently turning the power down as the signal gets close to the set-point.

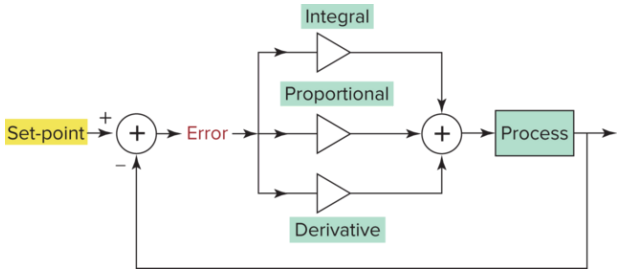


Figure 14-17 PID control loop.

The long-term operation of any system, large or small, requires a mass-energy balance between input and output. If a process were operated at equilibrium at all times, control would be simple. Because change does occur, the critical parameter in process control is time, that is, how long it takes for a change in any input to appear in the output. System time constants can vary from fractions of a second to many hours. The PID controller has the ability to tune its control action to specific process time constants and therefore to deal with process changes over time. PID control changes the amount of

output signal in a *mathematically* specified way that accounts for the amount of error and the rate of signal change.

Either programmable controllers can be fitted with input/output modules that produce PID control, or they will already have sufficient mathematical functions to allow PID control to be carried out. PID is essentially an equation that the controller uses to evaluate the controlled variable. **Figure 14-18** illustrates how a programmable logic controller can be used in the control of a PID loop. The operation of the PID loop can be summarized as follows:

- The process variable (pressure) is measured and feedback is generated.
- The PLC program compares the feedback to the set-point and generates an error signal.
- The error is examined by the PID loop calculation in three ways: with proportional, integral, and derivative methodology.
- The controller then issues an output to correct for any measured error by adjustment of the position of the variable flow outlet valve.

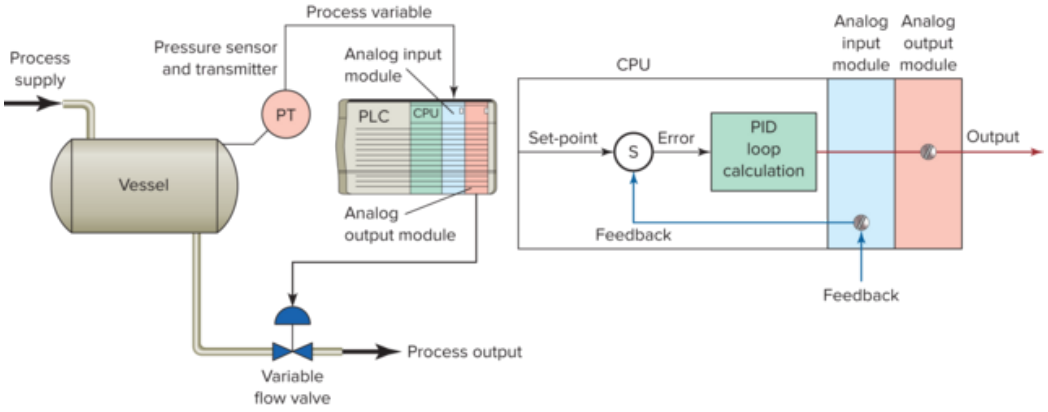


Figure 14-18 PLC control of a PID loop.

The ***response*** of a PID loop is the rate at which it compensates for error by adjusting the output. The PID loop is adjusted or tuned by changing the proportional gain, the integral gain, and/or the derivative gain. A PID loop is normally tested by making an abrupt change to the set-point and observing the controller's response rate. Adjustments can then be made as follows:

- As the proportional gain is increased, the controller responds faster.
- If the proportional gain is too high, the controller may become unstable and oscillate.
- The integral gain acts as a stabilizer.
- Integral gain also provides power, even if the error is zero (e.g., even when an oven reaches its set-point, it still needs power to stay hot).
- Without this base power, the controller will droop and hunt for the set-point.
- The derivative gain acts as an anticipator.
- Derivative gain is used to slow the controller down when change is too fast.

Basically, PID controller tuning consists of determining the appropriate values for the gain (proportional band), rate (derivative), and reset time (integral) tuning parameters (control constants) that will give the control required. Depending on the characteristics of the deviation of the process variable from the set-point, the tuning parameters interact to alter the controller's output and produce changes in the value of the process variable. In general, three methods of controller tuning are used:

Manual

- The operator estimates the tuning parameters required to give the desired controller response.
- The proportional, integral, and derivative terms must be adjusted, or tuned, individually to a particular system using a trial-and-error method.

Semiautomatic or Autotune

- The controller takes care of calculating and setting PID parameters.
  - Measures sensor output
  - Calculates error, sum of error, rate of change of error
  - Calculates desired power with PID equations
  - Updates control output

Fully Automatic or Intelligent

- This method is also known in the industry as fuzzy logic control.
- The controller uses artificial intelligence to readjust PID tuning parameters continually as necessary.
- Rather than calculating an output with a formula, the fuzzy logic controller evaluates rules. The first step is to "fuzzify" the error and change-in-error from continuous variables into linguistic variables, like "negative large" or "positive small." Simple if-then rules are evaluated to develop an output. The resulting output must be de-fuzzified into a continuous variable such as valve position.

The PID programmable controller output instruction uses closed-loop control to automatically control physical properties such as temperature, pressure, liquid level, or flow rate of process loops.

Figure 14-19 shows the PID output instruction and setup screen associated with the RSLogix 500 instruction set. The PID instruction is straightforward: it takes one input and controls one output. Normally, the PID instruction is placed on a rung without conditional logic. The output remains at its last value when the rung goes false. A summary of the basic information that is entered into the instruction is as follows:

Control Block—File that stores the data required to operate the instruction.

Process Variable—The element address that stores the process input value.

Control Variable—The element address that stores the output of the PID instruction.

Setup Screen—Instruction on which you can double-click to bring up a display that prompts you for other parameters you must enter to fully program the PID instruction.

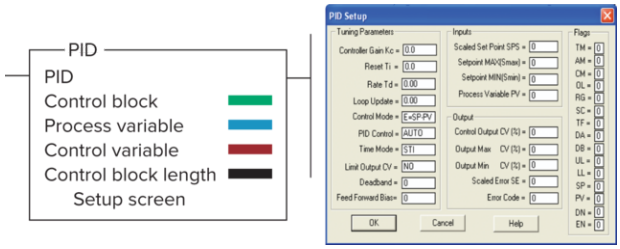


Figure 14-19 PID output instruction and setup screen.

Source: Image Courtesy of Rockwell Automation, Inc.

OceanofPDF.com

14.5 Motion Control

A motion control system provides precise positioning, velocity, and torque control for a wide range of motion applications. PLCs are ideally suited for both linear and rotary motion control applications. ***Pick and Place*** machines are used in the consumer products industry for a wide variety of product transfer applications. The machine takes a product from one point to another. One example is the transfer of a product to a moving conveyor belt as illustrated in **Figure 14-20**.

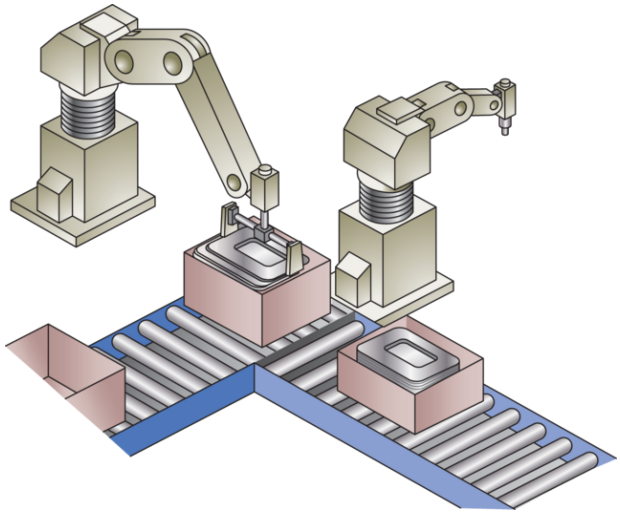


Figure 14-20 Pick and Place machine.

A basic PLC motion control system consists of a controller, a motion module, a servo drive, one or more motors with encoders, and the machinery being controlled. Each motor controlled in the system is referred to as an axis of motion. **Figure 14-21** illustrates a bottle-filling motion control process. This application requires two axes of motion: the motor operating the bottle filler mechanism and the motor controlling the conveyor speed. The role of each control component can be summarized as follows:

Programmable Logic Controller

- The controller stores and executes the user program that controls the process.
- This program includes motion instructions that control axis movements.
- When the controller encounters a motion instruction it calculates the motion commands for the axis.
- A motion command represents the desired position, velocity, or torque of the servo motor at the particular time the calculations take place.

Motion Module

- The motion module receives motion commands from the controller and transforms them into a compatible form the servo drive can understand.

- In addition it updates the controller with motor and drive information used to monitor drive and motor performance.

Servo Drive

- The servo drive receives the signal provided by the motion module and translates this signal into motor drive commands.
- These commands can include motor position, velocity, and/or torque.
- The servo drive provides power to the servo motors in response to the motion commands.
- Motor power is supplied and controlled by the servo drive.
- The servo drive monitors the motor's position and velocity by use of an encoder mounted on the motor shaft. This feedback information is used within the servo drive to ensure accurate motor motion.

Servo Motor

- The servo motors represent the axis being controlled.
- The servo motors receive electrical power from their servo drive which determines the motor shaft velocity and position.
- The filler motor must accelerate the filler mechanism in the direction the bottles are moving, match their speed, and track the bottles.
- After the bottles have been filled, the filler motor has to stop and reverse direction to return the filler mechanism to the starting position to begin the process again.

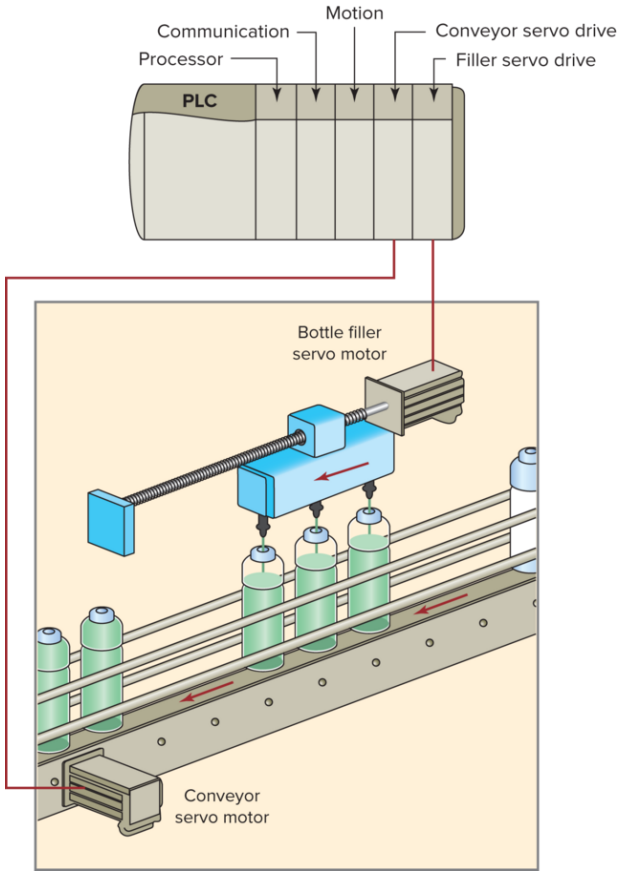


Figure 14-21 Bottle-filling motion control process.

A robot is simply a series of mechanical links driven by servo motors. The basic industrial robot widely used today is an ***arm*** or ***manipulator*** that moves to perform industrial operations. **Figure 14-22** illustrates the motion of a six-axis robot arm. Each axis of the robot arm is fundamentally a closed-loop servo control system. The wrist is the name usually given to the last three joints on the robot's arm. Going out along the arm, these wrist joints are known as the pitch joint, yaw joint, and roll joint. There are two types of controller setups that can be used to control an industrial robot—PLC- and PC-based systems. Depending on the difficulty of the task the robotic system will be performing, you may need a PLC or just a robot controller.

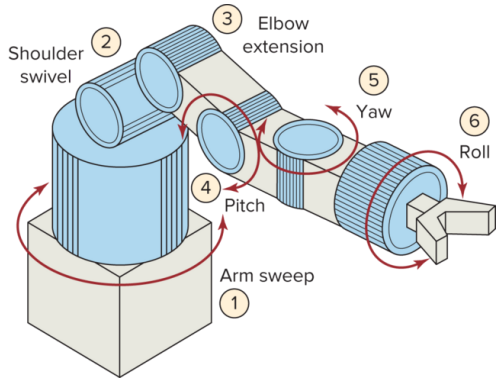


Figure 14-22 Six-axis robot arm.

OceanofPDF.com

14.6 Data Communications

Data communications refers to the different ways that PLC microprocessor-based systems talk to each other and to other devices. The two general types of communications links that can be established between the PLC and other devices are point-to-point links and network links. **Figure 14-23** illustrates a ***point-to-point*** serial communications link. Serial communications is used with devices such as printers, operator workstations, motor drives, bar code readers, computers, or another PLC. Serial communications interfaces are either built into the processor module or come as separate modules. A serial module installed in each controller is normally all that is required for two PLCs of the same manufacturer to establish a point-to-point link.

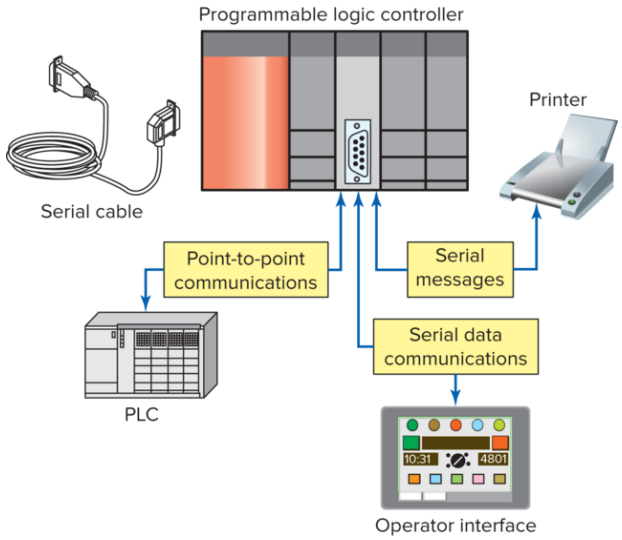


Figure 14-23 Point-to-point serial communications link.

As control systems become more complex, they require more effective communications schemes between the system components. A *local area network* or *LAN* is a system that interconnects data communications components within a limited geographical area, typically no more than one or two miles. Essentially, a LAN is a private, on-site communications system that allows communication between computers, PLCs, robots, and the like. The rate at which characters can be transmitted along a communications line is dependent on the number of bits of binary information that can be sent at a given time. This transfer of information is measured in bits per second, or **baud**. **Figure 14-24** illustrates a LAN communication link. Network communications supports communication among multiple PLCs and other devices. PLC networks allow:

- Sharing of information such as the current state of status bits among PLCs that may determine the action of one another.
- Monitoring of information from a central location.
- Programs to be uploaded or downloaded from a central location.
- Several PLCs to operate in unison to accomplish a common goal.

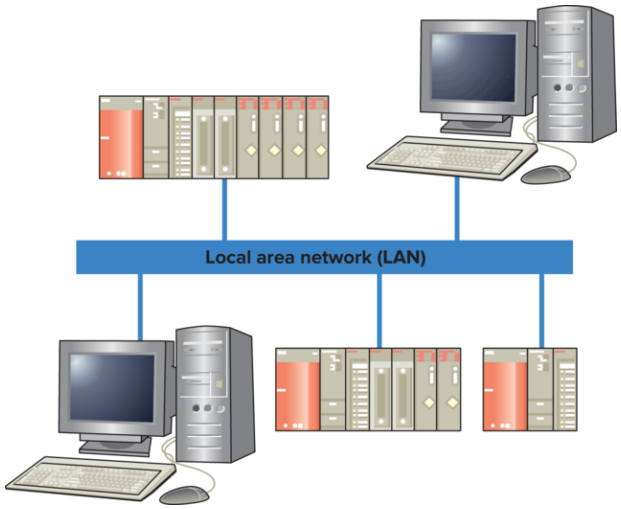


Figure 14-24 Local area network (LAN) communication link.

Transmission media are the cable through which data and control signals flow on a network. The transmission media used in data communications systems include coaxial cable, twisted pair, or fiber optics (**Figure 14-25**). Each cable has different electrical capabilities and may be more or less suitable to a specific environment or network type. Not all networks transmit information through cable. Wireless Wi-Fi Ethernet networks, such as the DF1 Radio Modem, communicate through radio waves, which are transmitted through the air. Fiber optic transmission medium is becoming increasingly popular in PLC based applications. Compared with twisted pairs of wire or coaxial cables, fiber optic cables have advantages such as their small size, no electromagnetic interference, and long transmission ranges. Typically, a fiber optic cable can carry a 1,000-megabaud signal 10 km. These types of cables have very large bandwidths so they are well-suited to high-speed transmission in noisy environments.

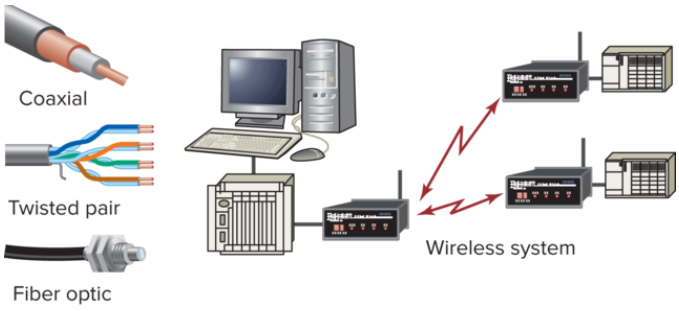


Figure 14-25 Transmission media.

In industrial applications, **LANs** have most often been used as the communication system for distributed control systems (DCS). Recall that a DCS system uses individual controllers to control the subsystems of a machine or process. This approach contrasts with centralized control in which a single controller governs the entire operation. A second major use of local area networks is that of supervisory control and data acquisition (SCADA). A LAN allows data collection and processing for a

group of controllers to be accomplished using one host computer as the central point for collecting data.

There are three general levels of functionality of industrial networks. **Figure 14-26** shows an illustration of the three levels, which can be summarized as follows:

Device Level—The device level involves various sensor and actuator devices of machines and processes. These may include devices such as sensors, switches, drives, motors, and valves.

Control Level—The control level involves the network's industrial controllers. This level may include controllers such as PLCs and robot controllers. Communications on the control level includes sharing I/O and program data between controllers.

Information Level—The information level is a plantwide network typically composed of the company's business networks and computers. This level may include scheduling, sales, management, and corporatewide information.

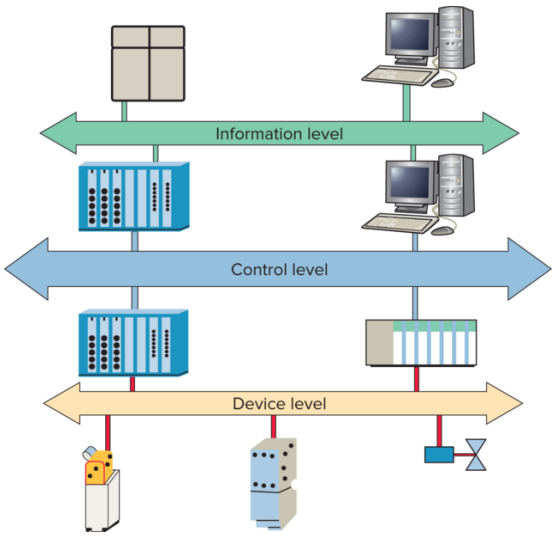


Figure 14-26 Levels of functionality of industrial networks.

Each device connected on a network is known as a ***node*** or ***station***. As signals travel along a network cable, they degrade and become distorted in a process that is called attenuation. If a cable is long enough, the attenuation will finally make a signal unrecognizable. A ***repeater*** is a device that amplifies a signal to its original strength in order to enable its signals to travel further. Different network types will have different specifications for cable length and type without a repeater.

Network topology is the physical layout of devices on a network formed by the network cables when nodes are attached. The ***star topology*** illustrated in **Figure 14-27** and its operation can be summarized as follows:

- A network switch or hub is connected to several PLC network nodes.
- Currently, most Ethernet networks use switches rather than hubs. A switch performs the same basic function as a hub but effectively increases the speed, size, and data handling capacity of the network.

- The configuration allows for bidirectional communication between switch/hub and each PLC.
- PLCs can be added to or removed from the network without disrupting the network.
- One problem with the star topology is that if the switch/hub goes down, the entire LAN is down.
- This type of system works best when information is transmitted primarily between the main controller and remote PLCs. However, if most communication is to occur between PLCs, the operation speed is affected.
- Also, the star system can use substantial amounts of communication conductors to connect all remote PLCs to one central location.

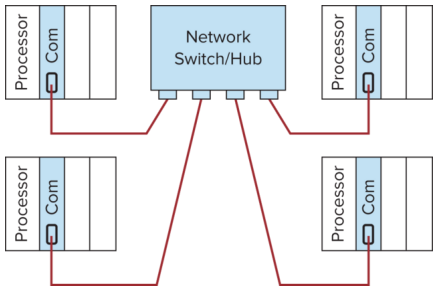


Figure 14-27 Star topology network.

Bus topology, illustrated in **Figure 14-28**, is a network configuration in which all stations are connected in parallel with the communication medium and all stations receive information from every other station on the network. The operation of a bus topology network can be summarized as follows:

Page 304

- Uses a single bus trunk cable to which individual PLC nodes are attached by a cable drop that taps off the main cable.
- Each PLC is interfaced to the bus using a network interface module that is attached using a drop cable or connector.
- Due to the nature of the bus technology, and the way the data are transmitted on the network, each end of the bus must be terminated with a terminating resistor.
- As the data move along the total bus, each PLC node is listening for its own node identification address and accepts only information sent to that address.
- Because of the simple linear layout, bus networks require less cable than all other topologies.
- No single station controls the network and stations can communicate freely to one another.
- Bus networks are very useful in distributive control systems, because each station or node has equal independent control capability and can exchange information at any given time.
- Another advantage of the bus network is that you can add or remove stations from the network with a minimum amount of system reconfiguration.
- This network's main disadvantage is that all the nodes rely on a common bus trunk line, and a break in that common line can affect many nodes.

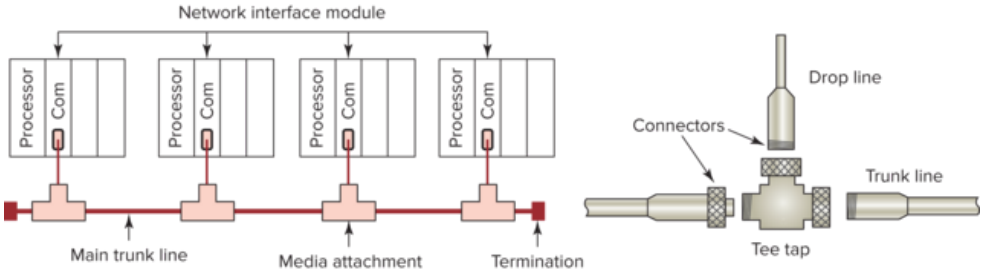


Figure 14-28 Bus topology network.

I/O bus networks can be divided into two categories: device bus networks and process bus networks. ***Device bus networks*** interface with low-level information devices such as pushbuttons and limit switches that primarily transmit data relating to the on/off state of the device and its operational status. Device bus networks can be further classified as bit-wide or byte-wide buses. Device bus networks that include discrete devices as well as small analog devices are called ***byte-wide bus networks.*** These networks can transfer 50 or more bytes of data at a time. Device bus networks that interface only with discrete devices are called ***bit-wide bus networks.*** Bit-wide networks transfer less than 8 bits of information to and from simple discrete devices.

Process bus networks are capable of communicating several hundred bytes of data per transmission. The majority of devices used in process bus networks are analog, whereas most devices used in device bus networks are discrete. Process bus networks connect with high-level information devices such as smart process valves and flowmeters, which are typically used in process control applications. Process buses are slower because of their large data packet size. Most analog control devices are used in controlling such process variables as flow and temperature, which are typically slow to respond.

A ***protocol*** is a set of rules that two or more devices must follow if they are to communicate with each other. Protocols are to computers what language is to humans. This book is in English, and to understand it, you must be able to read English. Similarly, for two devices on a network to successfully communicate, they must both understand the same protocols.

A network protocol defines how data is arranged and coded for transmission on a network. In the past, communications networks were often proprietary systems designed to a specific vendor's standards; users were forced to buy all their control components from a single supplier. This is because of the different communications protocols, command sequences, error-checking schemes, and communications media used by each manufacturer. Today, the trend is toward open network systems based on international standards developed through industry associations.

Most PLCs adhere to the protocols established by the International Organization for Standardization, or ISO. The Open Systems Interconnection (**OSI model**) is a set of seven layers that define the different stages that data must go through to travel from one device to another over a network. The OSI model takes into account the hardware, software, protocols, and network architecture that are needed for any two machines to communicate. The actual connections between machines are physical, while the connections between layers in the model are logical.

Gateways (Figure 14-29) make communication possible between different architectures and protocols. They repackage and convert data going from one network to another network so that the one can understand the other's application data. Gateways can change the format of a message so that it will conform to the application program at the receiving end of the transfer. If network-access translation is their only function, the interfaces are known as ***bridges.*** If the interface also adjusts data formats or performs data transmission control, then it is called a ***gateway.***

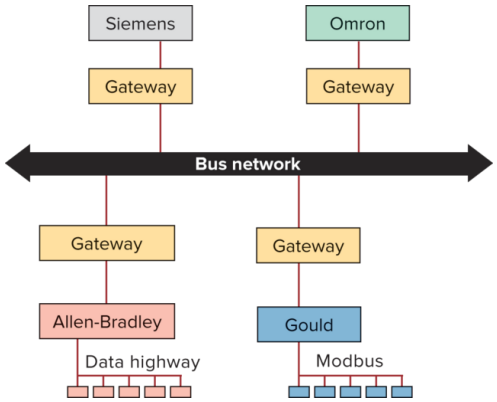


Figure 14-29 Translating from one network-access scheme to another.

A bus topology network requires some method of controlling a particular device's access to the bus. An ***access method*** is the manner in which a PLC accesses the network to transmit information. Network access control ensures that data are transmitted in an organized manner preventing the occurrence of more than one message on the network at a time. Although many access methods exist, the most common are token passing, collision detection, and polling.

In a ***token passing*** network, a node can transmit data on the network only when it has possession of a token. A token is simply a small packet that is passed from node to node as illustrated in Figure 14-30. When a node finishes transmitting messages, it sends a special message to the next node in the sequence, granting it the token. The token passes sequentially from node to node, allowing each an opportunity to transmit without interference. Tokens usually have a time limit to prevent a single node from tying up the token for a long period of time.

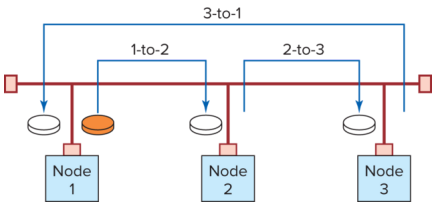


Figure 14-30 Example of token passing.

Ethernet networks use a ***collision detection*** access control scheme. With this access method, nodes listen for activity on the network and transmit only if there are no other messages on the network. On Ethernet networks there is the possibility that nodes will transmit data at the same time. When this

happens a collision is detected. Each node that had sent out a message will wait a random amount of time and will resend its data if it does not detect any network activity.

In the past, certain networked devices were known as master/slave. More recently, these devices have been named server/client to reflect an industry-wide shift to using more inclusive language. In a **server/client network** the server controls all communications originating from other controllers using an access method called polling. This configuration is illustrated in **Figure 14-31** and consists of several clients and one server. Its operation can be summarized as follows:

- The server controller sends data to the client controllers.
- When the server needs data from a client, it will *poll* (address) the client and wait for a response.
- No communication takes place without the server initiating it.
- Direct communication among client devices is not possible.
- Information to be transferred between clients must be sent first to the network server unit, which will, in turn, retransmit the message to the designated client device.
- Server/client networks use two pairs of conductors. One pair of wires is used for the server to transmit data and the client to receive them. On the other pair, the slaves transmit and the server receives.

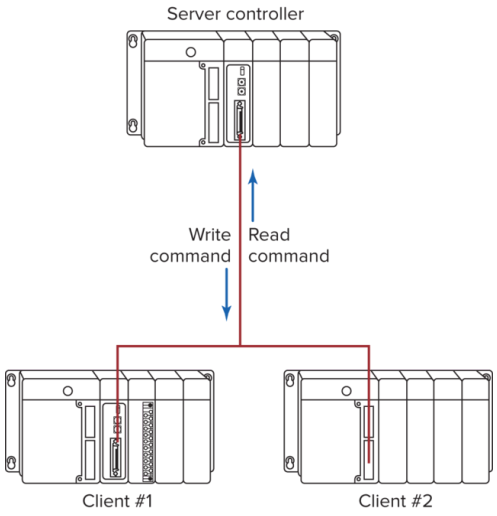


Figure 14-31 Server/client network.

A **peer-to-peer network** has a distributive means of control, as opposed to a server/client network in which one node controls all communications originating from other nodes. The Allen-Bradley Data Highway, shown in **Figure 14-32**, is an example of a peer-to-peer network of programmable controllers and computers linked together to form a data communication system. The operation of the network can be summarized as follows:

- Peer-to-peer networks can use token passing or Ethernet collision detection.
- Each device has the ability to request use of, and then take control of, the network for the purpose of transmitting information to or requesting information from other network devices.
- Each device is identified by an address.

- When the network is operating, the token passes from one device to the next sequentially.
- The device that is transmitting the token also knows the address of the next station that will receive the token.
- Each device receives the packet information and uses it, if needed.
- Any additional information that the node has will be sent in a new packet.

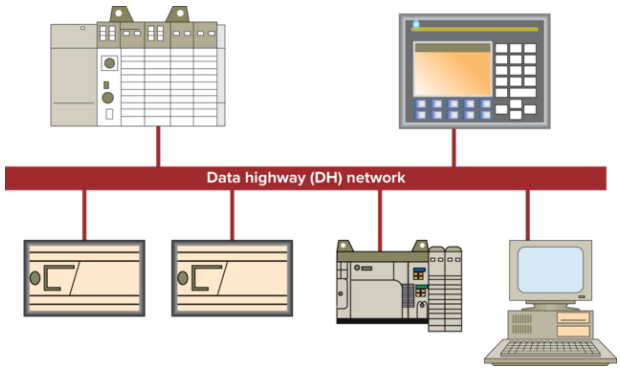


Figure 14-32 Peer-to-peer network.

There are two methods of transmitting PLC digital data: parallel and serial transmission. In ***parallel*** data transmission, all bits of the binary data are transmitted simultaneously, as illustrated in **Figure 14-33**. Parallel transmission of data can be summarized as follows:

- Eight transmission lines are required to transmit the 8-bit binary number.
- Each bit requires its own separate data path and all bits of a word are transmitted at the same time.
- Parallel data transmission is less common but faster than serial transmission.

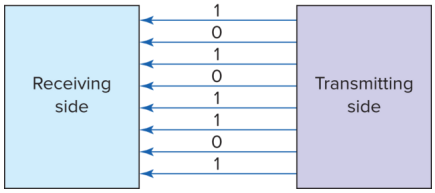


Figure 14-33 Parallel data transmission.

In ***serial*** transmission one bit of the binary data is transferred at a time, as illustrated in **Figure 14-34**. Serial transmission of data can be summarized as follows:

- In serial transmission, bits are sent sequentially on the same channel (wire) which reduces costs for wire but also slows the speed of transmission.
- Serial data can be transmitted effectively over much greater distances than can parallel data.
- Each data word in the serial transmission must be denoted with a known start bit sequence followed by the data bits that contain the intelligence of the data transmission and a stop bit.
- An extra bit, termed a ***parity bit,*** may be used to provide some error-detecting ability.

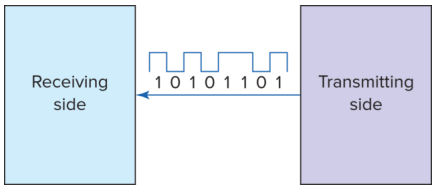


Figure 14-34 Serial data transmission.

A ***duplex*** communication system is a system composed of two connected devices that can communicate with one another in both directions at the same time. A *half-duplex* system provides for communication in both directions, but only one direction at a time (not simultaneously). Half-duplex transmission is used for server/client communications. ***Full-duplex*** transmission allows the transmission of data in both directions simultaneously and can be used for peer-to-peer communications.

The different networking schemes replace traditional point-to-point hardwiring. Network control of systems minimizes the amount of wiring that needs to be done. With traditional wiring multiple wires from each device, fed through control cabinets, often result in large wire bundles running through the system. Due to the sheer volume of wires, installation time is considerable and troubleshooting is complex. If a network is used all devices can be directly connected to a single transmission media cable.

High-speed industrial networking technologies offer a variety of methods for connecting devices. PLC network configurations may be either open or proprietary (vendor-unique). Following is an overview of some of the industrial communication technologies that play a critical role in today's control systems.

Data Highway

The Allen-Bradley Data Highway networks, Data Highway Plus (DH+) and DH-485, are proprietary communications networks. They use peer-to-peer communication implementing token passing. The medium is shielded twisted pair cable. **Figure 14-35** shows the DH+ network connection for an SLC 5/04 controller. The three-pin Phoenix connector is used to form the network transmission media.

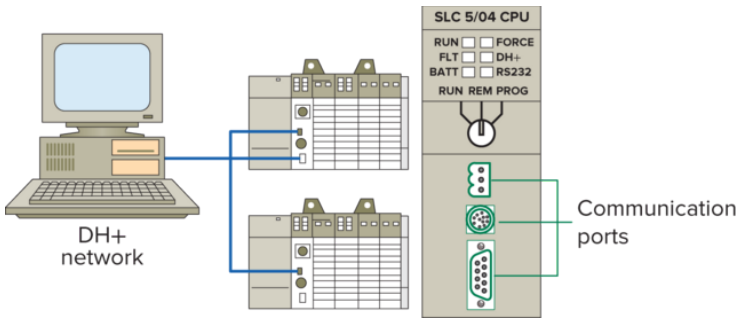


Figure 14-35 Data Highway network connection.

Serial Communication

Serial data communication is implemented using standards such as RS-232, RS-422, and RS-485. The RS in the standard's name means ***recommended standard*** that specifies the electrical, mechanical, and functional characteristics for serial communications. Serial communication interfaces are either built into the processor module or come as a separate communications interface module, as illustrated in **Figure 14-36**. The simplest type of connection is the RS-232 serial port. The RS interfaces are used to connect to devices such as vision systems, barcode readers, and operator terminals that must transfer quantities of data at a reasonably high rate between the remote device and the PLC. The RS-232 type of serial transmission is designed to communicate between one computer and one controller and is usually limited to lengths up to 50 feet. RS-422 and RS-485 serial transmission types are designed to communicate between one computer and multiple controllers, have a high level of noise immunity, and are usually limited to lengths of 650 feet (for RS-485) or 1650 feet (for RS-422).

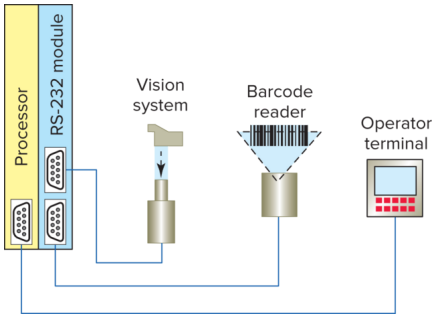


Figure 14-36 Serial communication interface.

DeviceNet

DeviceNet is an open device-level network. It is relatively low speed but efficient at handling the short messages to and from I/O modules. As PLCs have become more powerful, they are being required to control an increasing number of I/O field devices. Therefore, at times it may not be practical to separately wire each sensor and actuator directly into I/O modules. **Figure 14-37** shows a comparison between conventional and DeviceNet I/O systems. Conventional systems have racks of inputs and outputs with each I/O device wired back to the controller. The DeviceNet protocol dramatically reduces costs by integrating all I/O devices on a 4-wire trunk network with data and power conductors in the same cable. This direct connectivity reduces costly and time-consuming wiring.

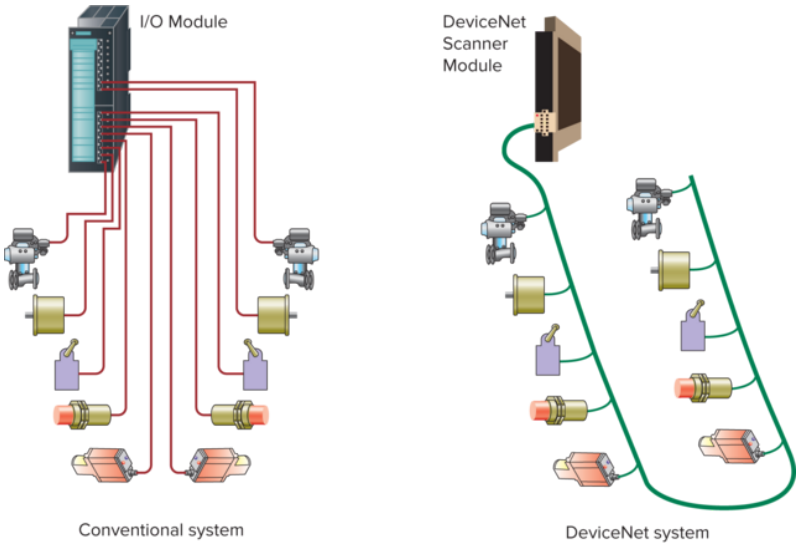


Figure 14-37 Conventional and DeviceNet I/O systems.

The basic function of a DeviceNet I/O bus network is to communicate information with, as well as supply power to, the field devices that are connected to the bus. The PLC drives the field devices directly with the use of a ***network scanner*** instead of I/O modules, as illustrated in **Figure 14-38**. The scanner module communicates with DeviceNet devices over the network to:

- Read inputs from a device.
- Write outputs to a device.
- Download configuration data.
- Monitor a device's operational status.

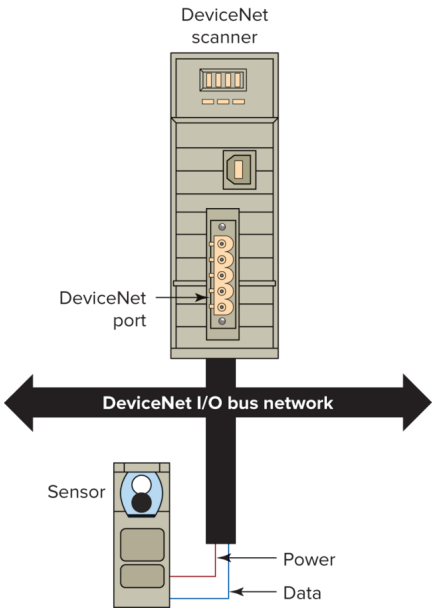


Figure 14-38 DeviceNet network scanner.

The scanner module communicates with the controller to exchange information which includes:

- Device I/O data
- Status information
- Configuration data

DeviceNet also has the unique feature of having power on the network. This allows devices with limited power requirements to be powered directly from the network, further reducing connection points and physical size.

DeviceNet uses the Common Industrial Protocol, called *CIP,* which is strictly object oriented. Each object has attributes (data), services (commands), and behavior (reaction to events). Two different types of objects are defined in the CIP specification: communication objects and application-specific objects. A DeviceNet network can support up to 64 nodes and the network end-to-end distance is variable, based on network speed. **Figure 14-39** shows an example of a typical layout of the trunk wiring for a DeviceNet network. Communications data is carried over two wires with a second pair of wires carrying power.

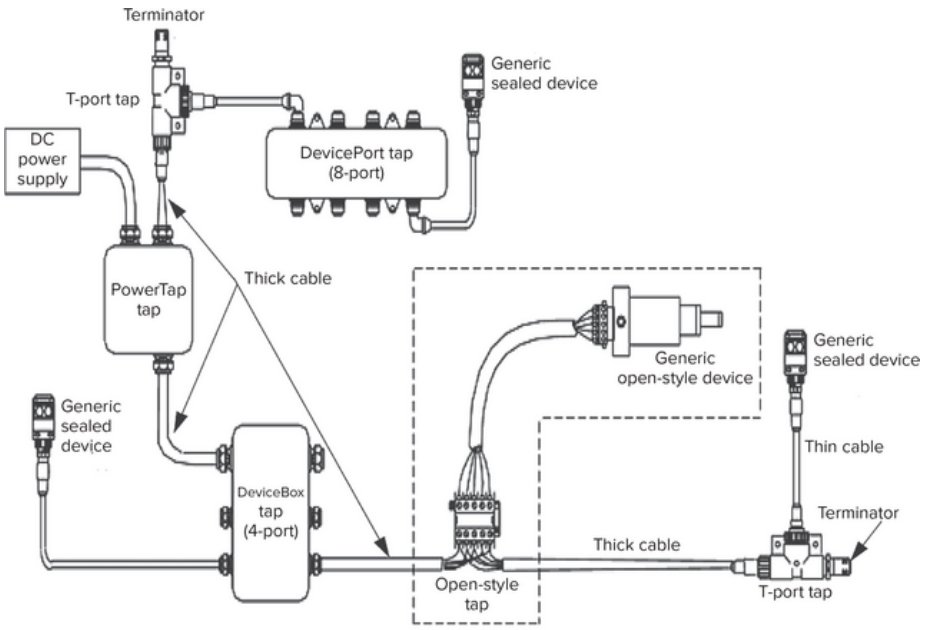




Figure 14-39 Layout of a DeviceNet network.

Source: Image Courtesy of Rockwell Automation, Inc.

The field devices that are connected to the network contain intelligence in the form of microprocessors or other circuits. These devices can communicate not only the on/off status of field devices but also diagnostic information about their operating state. For example, you can detect via the network that a photoelectric sensor is losing reliability because of a dirty lens, and you can correct the situation before the sensor fails to detect an object. A limit switch can report the number of motions it has performed, which may be an indication that it has reached the end of its operating life and thus requires replacement.

ControlNet

ControlNet is positioned one level above DeviceNet. It uses the Common Industrial Protocol (CIP) to combine the functionality of an I/O network and a peer-to-peer network providing high-speed performance for both functions. This open high-speed network is highly deterministic and repeatable. ***Determinism*** is the ability to reliably predict when data will be delivered, and ***repeatability*** ensures that transmit times are constant and unaffected by devices connecting to, or leaving, the network. Electronic device data sheets (EDS-Files) are required for each ControlNet device. During the setup phase the ControlNet scanner must configure each device according to the EDS-Files. The ControlNet layout shown in **Figure 14-40** has a ***redundant media*** option in which two separate cables are installed to guard against failures such as cut cables, loose connectors, or noise.

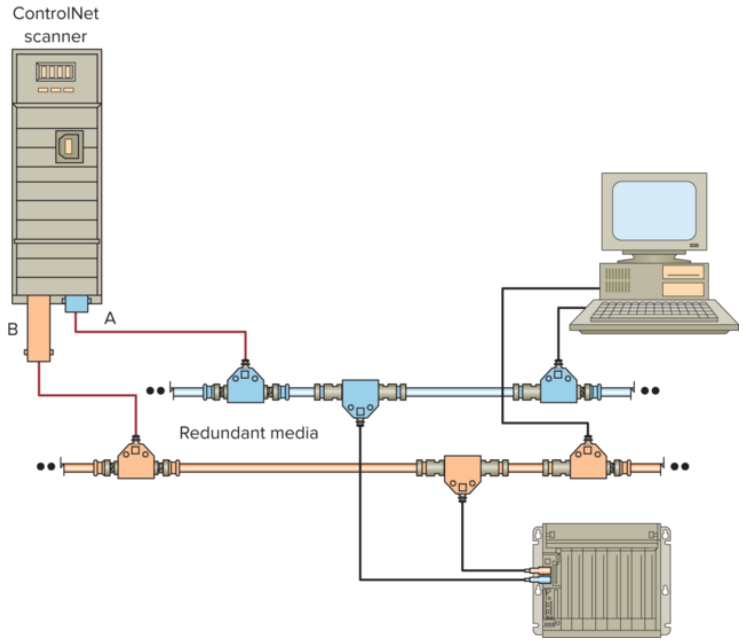


Figure 14-40 ControlNet network with redundant media installed.

EtherNet/IP

EtherNet/IP (Ethernet Industrial Protocol) is an open communications protocol based on the Common Industrial Protocol (CIP) layer used in both DeviceNet and ControlNet. It allows users to link information seamlessly between devices running the EtherNet/IP protocol without custom hardware, as illustrated in **Figure 14-41**.

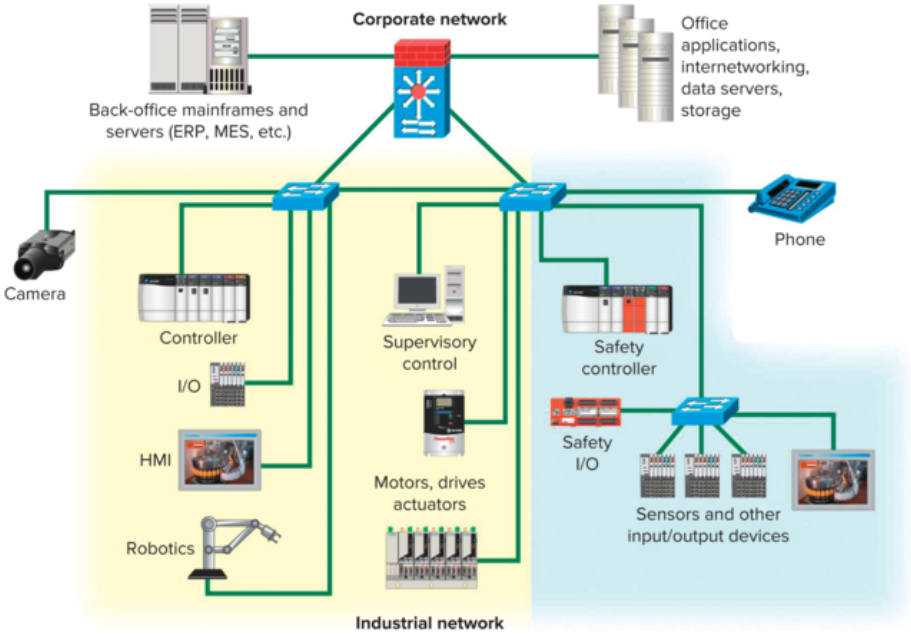


Figure 14-41 EtherNet/IP information links.

Source: Image Courtesy of Rockwell Automation, Inc.

The following are some of the important features of EtherNet/IP:

- Sharing a common application layer between ControlNet, DeviceNet, and Ethernet/IP will make plug-and-play interoperability possible among complex devices from multiple vendors. ***Plug and play*** refers to the ability of a computer system to automatically configure devices. This allows you to plug in a device and play (operate) it without worrying about setting DIP switches, jumpers, and other configuration elements.
- EtherNet/IP provides standardized full-duplex operation which gives a single node, in a peer-to-peer connection, full attention and therefore maximum possible bandwidth. ***Bandwidth*** refers to the data rate supported by a network, commonly expressed in terms of bits per second. The greater the bandwidth the greater the overall performance.
- EtherNet/IP allows interoperability of industrial automation devices and control equipment on the same network used for business applications and browsing the Internet.

Modbus

Modbus is a serial communication protocol originally developed by Modicon for use with its PLCs. Basically, it is a method used for transmitting information over serial lines between electronic devices. The device requesting the information is called the Modbus Server and the devices supplying information are Modbus Clients. Modbus is an open protocol, meaning that it's free for manufacturers to build into their equipment without having to pay royalties. It has become a standard communications protocol in industry, and is one of the most commonly available means of connecting industrial electronic devices.

Fieldbus

Fieldbus is an open, serial, two-way communications system that interconnects measurement and control equipment such as sensors, actuators, and controllers. At the base level in the hierarchy of plant networks, it serves as a network for field devices used in process control applications.

There are several possible topologies for fieldbus networks. **Figure 14-42** illustrates the ***daisy-chain*** topology. With this topology, the fieldbus cable is routed from device to device. Installations using this topology require connectors or wiring practices such that disconnection of a single device is possible without disrupting the continuity of the whole segment.

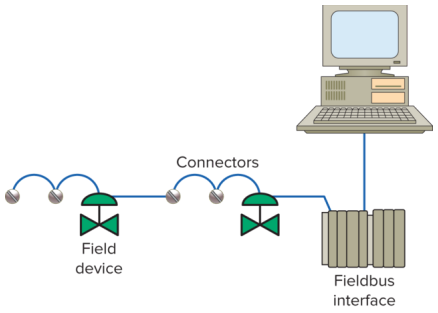


Figure 14-42 Fieldbus implemented using daisy-chain topology.

PROFIBUS-DP

PROFIBUS-DP (where DP stands for Decentralized Periphery) is an open, international fieldbus communication standard that supports both analog and discrete signals. It is functionally comparable to DeviceNet. The physical media are defined via the RS-485 or fiber optic transmission technologies. PROFIBUS-DP communicates at speeds up to 12 Mbps over distances up to 1200 meters.

SERCOS

SERCOS **(Serial Real-time Communications System)** is an internationally approved communication standard for motion control. The SERCOS standard makes it possible to use devices from various manufacturers. This communication network is designed for high-speed serial communication of standardized closed-loop data, in real time, over a noise-immune fiber optic cable. The SERCOS interface modules use a single, digital fiber optic link, which eliminates as many as 18 digital wires per axis.

Figure 14-43 shows a smart belt packaging operation that uses a SERCOS module interface in conjunction with Allen-Bradley Kinetix 6000 multi-axis servo drive. The operation of the process can be summarized as follows:

- This packaging application is used to convert randomly spaced product into evenly spaced product that's properly phased to the in-feed of another machine.

- The SERCOS interface module is linked with fiber optic cable to the Allen-Bradley Kinetix 6000 servo drive.
- The randomly spaced product is fed by a conveyor driven by an induction-type motor.
- The servos that make up the smart belts follow the in-feed of the next conveyor downstream so that the smart belt's speed is matched to that machine's speed.

- The input from the sensor on each smart belt senses the product and makes position corrections based on error calculations performed in the controller.
- The corrections are performed on a percentage of the total error, based on whether the belt is performing a coarse or fine correction.
- The first belts in the process are usually coarse correction belts and typically make a large correction for as much as 70 to 100% of the total phasing error amount.
- The last belts perform fine correcting for 100% of the measured remaining phasing error even though the total error is smaller, relative to what it was at the beginning of the correcting process.

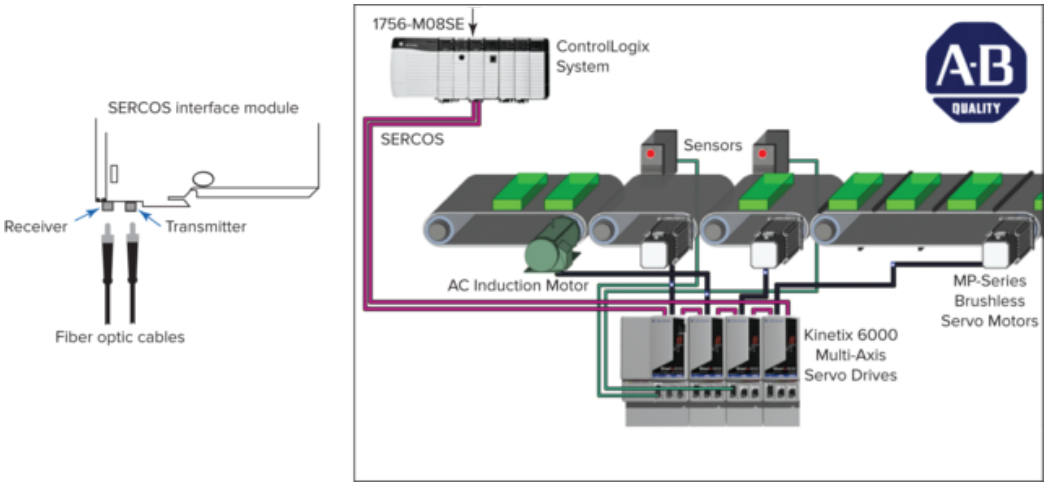


Figure 14-43 Smart belt packaging operation.

Source: Image Courtesy of Rockwell Automation, Inc.

HART

HART is an open server/client communication protocol developed to communicate with smart field devices. Smart field devices contain more information than the traditional 4–20 mA signal. In addition, they carry out some functions that are/were originally programmed within the PLC. HART Protocol allows the simultaneous communication of the continuous 4–20 mA as well as a second digital communication path resting on top of the analog signal, but not interfering. This allows access to the data of the field device and/or exact machine-readable product description of the field devices including their data and functions. More instrumentation devices are available with the HART protocol than with any other digital communications technology.

The HART digital signal is superimposed onto the standard 4–20 mA signal, as illustrated in **Figure 14-44**.

- The digital signal is made up of two frequencies, 1.2 and 2.2 kHz, representing bits 1 and 0 respectively.
- Sine waves of these two frequencies are superimposed onto the analog signal cables to give simultaneous analog and digital communications.
- The DC and low-frequency current signals are modulated by the 1200 and 2200 Hz frequencies; a technique known as frequency-shift keying or FSK.
- As the average value of the FSK signal is always zero there is no effect on the 4– 20 mA analogue signal.
- A minimum loop impedance of 230 ohms is required for communication.

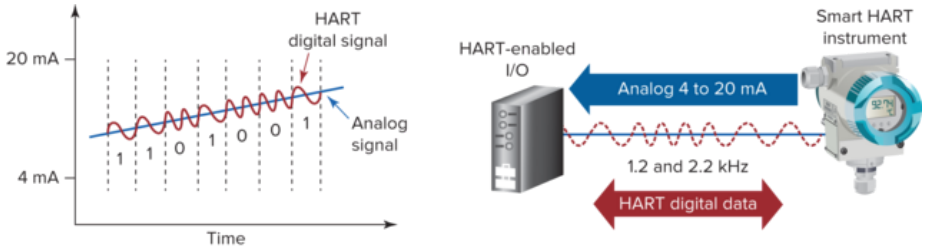


Figure 14-44 HART communication protocol.

OceanofPDF.com

14.7 Supervisory Control and Data Acquisition (SCADA)

In some applications, in addition to its normal control functions, the PLC is responsible for collecting data, performing the necessary processing, and structuring the data for generating reports. As an example, you could have a PLC count parts and automatically send the data to a spreadsheet on your desktop computer.

Data collection is simplified by using a **SCADA (supervisory control and data acquisition)** system, shown in Figure 14-45. Exchanging data from the plant floor to a supervisory computer allows data logging, data display, trending, downloading of recipes, setting of selected parameters, and availability of general production data. The additional supervisory control output capabilities allow you to tweak your processes accurately for maximum efficiency. In general, unlike distributive control systems, a SCADA system usually refers to a system that coordinates but does not control processes in real time.

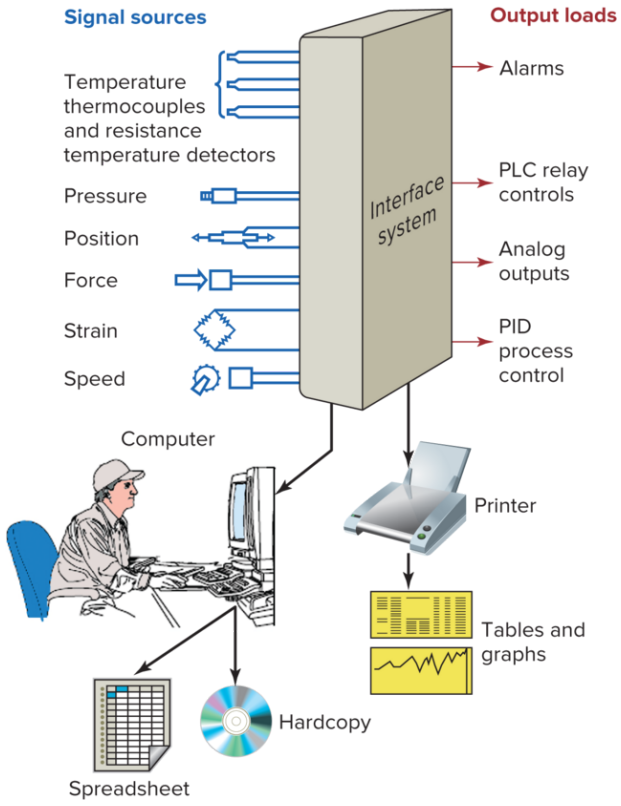


Figure 14-45 Supervisory control and data acquisition (SCADA).

In a typical SCADA system, independent PLCs perform I/O control functions on field devices while being supervised by a SCADA/HMI software package running on a host computer, as illustrated in Figure 14-46. Process control operators monitor PLC operation on the host computer and send

control commands to the PLCs if required. The great advantage of a SCADA system is that data are stored automatically in a form that can be retrieved for later analysis without error or additional work. Measurements are made under processor control and then displayed onscreen and stored to a hard drive. Accurate measurements are easy to obtain, and there are no mechanical limitations to measurement speed.

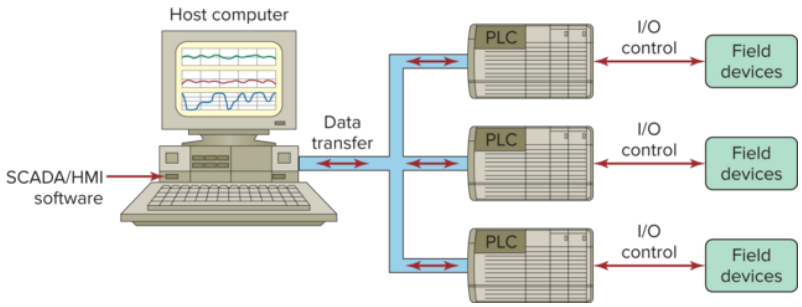


Figure 14-46 Typical SCADA system.

An important part of most SCADA implementations is alarm handling. An **alarm** is an announcement to the operator initiated by a process variable passing a defined limit as it approaches an undesirable or unsafe value. The announcement includes audible sounds, visual indications, and messages. Properly designed alarms will notify the operator of abnormal situations with enough time to successfully manage them. For this reason, alarms often have a several-second delay associated with them to ensure the process being monitored has stabilized before activating an alarm. Alarm management includes being able to distinguish between alarms and alerts. An **alert** provides a warning mechanism, but doesn't necessarily require immediate action.

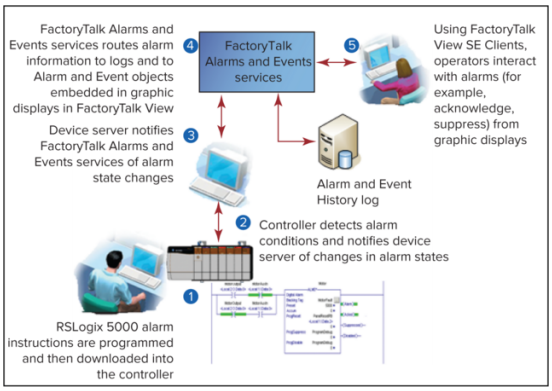
Rockwell's **FactoryTalk** services platform is a suite of software services that include:

- **LiveData—**Provides a data link between clients and servers. For example, the connection between an HMI server and PLC.
- **Directory—**Allows products to share a common address book that finds and provides access to plant-floor resources, such as users, tags, and graphics displays.
- **Security—**Provides a range of security services that are integrated into the FactoryTalk directory.
- **Audit—**Collects messages that document changes done by users during design, management, and product operations.
- **Activation—**Provides a secure, software-based system for activating Rockwell Software products.
- **Alarm and Events—**Provides both server-based and device-based process control alarm server.

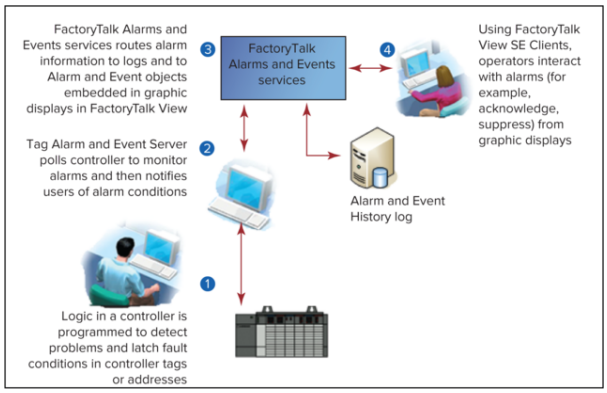
FactoryTalk alarms and events support the following two types of alarm monitoring (**Figure 14-47**):

Device-based—Pre-built alarm instructions that are programmed in a Logix 5000 project and then downloaded into a Logix controller. The controller detects alarm conditions and publishes event information, which is routed through the system for display and logging.

Tag-based—Software-based tag servers monitor data tags for alarm conditions and publish event information for display and logging. Tag-based alarm monitoring is supported by Logix 5000 controllers and Logix 500 controllers.



(*a*) Device-based alarms



(*b*) Tag-based alarms

Figure 14-47 FactoryTalk alarm monitoring.

Source: Image Courtesy of Rockwell Automation, Inc.

OceanofPDF.com

14.8 Variable Frequency Drive (VFD) with PLC Control

A Variable Frequency Drive (VFD) controls AC motor speed by controlling the frequency of the power applied to the motor. The drive's principle of operations is based in the fact that varying the frequency of the AC voltage applied to the motor will result in a corresponding change in the speed of the motor. The use of a VFD will allow for:

- Accurate speed control
- Energy cost savings
- Acceleration/deceleration ramp adjustments
- Monitory motor information
- Displaying faults

It is very common for a VFD to be networked to PLCs. The drive is controlled and monitored through the PLC network via a communication module, such as the DeviceNet Scanner. The procedure for installing the communication module into the PLC is similar to that used for installing input and output modules. Input and output addressing between the two devices is illustrated in **Figure 14-48**.

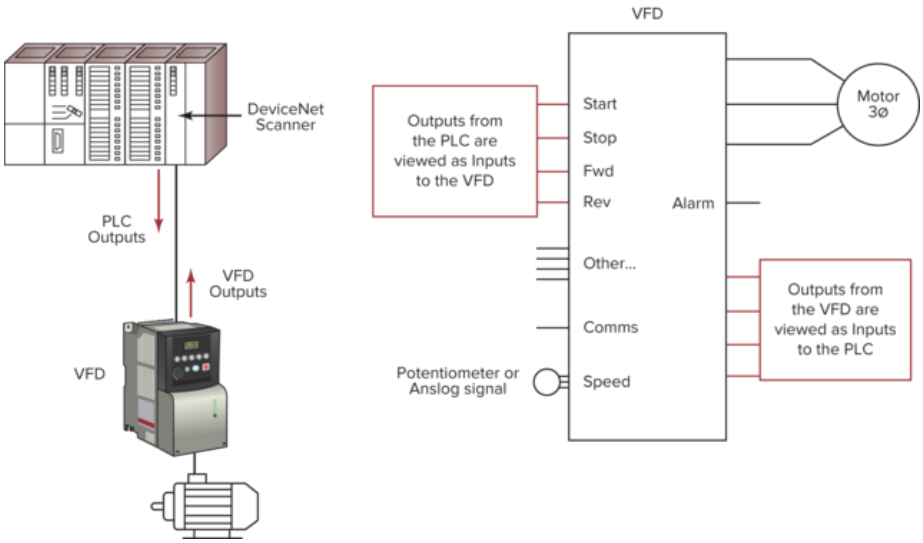


Figure 14-48 Addressing between the PLC and VFD.

- An output from the PLC is viewed as an input into the VFD.
- An output from the VFD is viewed as an input to the PLC.

There is usually a faceplate on the VFD where parameters can be entered into the drive. These include preset speeds, motor faceplate values, 2 wire or 3 wire control, and many other parameters. When the VFD and the PLC are the same brand, it is usually relatively easy to configure all of the drive signals.

Once the DeviceNet module is installed, inputs and outputs tags are created. The input tags are used to monitor the VFD by sending **status** bits to the PLC. The output tags are used to send **commands** to the VFD. In this example, the PowerFlex 40 drive is connected to an RSLogix 5000 controller through DeviceNet using a communication hardware embedded in the drive. Memory for the drive control functions and status is mapped into the DeviceNet Scanner located in slot 6 of the PLC chassis. Input and output words are created when you install the DeviceNet module.

Drive Control

Two elements (words) are used to **send** data to the PowerFlex 40 drive unit. The first element is called the **Logic Command Word,** where individual bits are used to control various functions of the drive. These control bits are used with the Output Energize (OTE) instruction. The six most frequently used commands are shown below.

Logic Command Word—Local:6:O.Data[0]

Bit Address	Command	Description
Local:6:O.Data[0].0	Stop Drive	**1 = Stop 0 = Not Stop**
Local:6:O.Data[0].1	Start Drive	**1 = Start 0 = Not Start**
Local:6:O.Data[0].2	Jog Drive	**1 = Jog 0 = Not Jog**
Local:6:O.Data[0].3	Clear Faults	**1 = Clear Faults 0 = Not Clear Faults**
Local:6:O.Data[0].4	Set Forward Direction	**1 = Forward Request 0 = No Command**
Local:6:O.Data[0].5	Set Reverse Direction	**1 = Reverse Request 0 = No Command**

Page 316

- **Stop Drive** bit zero is used to stop the drive. This command does not need to be on the whole time, all it needs is to be simply pulsed. It is going to stop the drive when this bit is High (1). When it's normal or when it's zero it does not stop the drive. It will only stop it when it is being pulsed.
- **Start Drive** bit one is used to start the drive when there is a one in its memory location. When there is a zero in that memory location, it does not indicate that the drive is going to stop. It simply means it is not being told to start. **Note,** should the Start and Stop bits be set simultaneously, the Stop command will override the Start.
- **Jog Drive** bit two is used for the jog command. The Jog bit is set to 1 for jog operation and 0 for back to normal. Jog allows for running the drive at a different speed/direction than the normal run mode.
- **Clear Faults** bit three is pulsed to 1 to clear all detected fault messages.
- **Set Forward Direction,** bit four and **Set Reverse Direction,** bit five are used if the motor needs to run in both directions.
- An example of a motor **Start/Stop** code is shown in **Figure 14-49**.

Start/Stop Code Example

Bit Address	Command	Description
<Local:6:O.Data[0].0	Stop Drive	1 = Stop 0 = Not Stop
<Local:6:O.Data[0].1>	Stop Drive	1 = Start 0 = Not Start

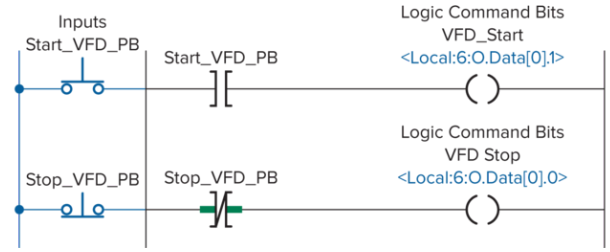


Figure 14-49 Motor Start/Stop code.

The second element is called the **Speed Reference Word,** which tells the drive the **desired** speed and requires the full DINT (32 bits). Speed reference is entered as a decimal value including a digit for the first decimal place. For example to run the drive motor at 35.7 Hz, the value 357 would be placed into memory word **Local:6:O.Data[1]**. This could be done using the move (MOV) instruction.

Drive Status

Two elements (words) are used to **receive** data from the PowerFlex 40 drive unit. The first element is called the **Logic Status Word** where individual bits indicate the status of the drive unit. These bits can be checked for 1 or 0 states by using the XIC of XIO instructions. The nine most frequently used status bits are shown below.

Logic Status Word—Local:6:I.Data[0]

Bit Address	Status	Description	
Local:6:I.Data[0].0	Drive Ready	**1 = Ready**	**0 = Not Ready**
Local:6:I.Data[0].1	Drive Active	**1 = Active**	**0 = Not Ready**
Local:6:I.Data[0].2	Command Direction	**1 = Forward**	**0 = Reverse**
Local:6:I.Data[0].3	Actual Direction	**1 = Forward**	**0 = Reverse**
Local:6:I.Data[0].4	Drive Accelerating	**1 = Accelerating**	**0 = Not Accelerating**
Local:6:I.Data[0].5	Drive Decelerating	**1 = Decelerating**	**0 = Not Decelerating**
Local:6:I.Data[0].6	Alarm	**1 = Alarm**	**0 = No Alarm**
Local:6:I.Data[0].7	Drive Faulted	**1 = Faulted**	**0 = Not Faulted**
Local:6:I.Data[0].8	Drive At Requested Speed	**1 = At Speed**	**0 = Not at Speed**

- The Logic Status Word is an input as you can see by the I in the address.
- This word is broken down into individual bit addresses.
- Bit address 0 indicates whether or not the drive is ready to operate.
- Bit address 1 indicates whether or not the drive is active or operating.

- Bit address 2 is the commanded direction. This is not the direction that the motor is actually going, only the direction the PLC wants it to go.
- Bit address 3 indicates the actual direction the motor is operating in. For example, if the motor is operating full speed in the forward direction and then commanded to operate in reverse, it will take some time for the motor to slow down. During this period of time it will indicate that the motor is still operating in the forward direction even though it has been commanded to operate in reverse.
- Bit addresses 4 and 5 indicate whether the motor is accelerating or decelerating.
- Bit address 6 indicates an alarm condition has been detected.
- Bit address 7 indicates a drive fault has been detected.
- Bit address 8 indicates the drive is at the requested speed.

The second element is called the **Speed Feedback Word,** which indicates the actual speed of the drive. The VFD monitors the rotating field of the motor to determine the actual speed. In this example it is found in memory word **Local:6:I.Data[1]** and requires the full DINT (32 bits). Speed feedback is also received as a decimal value including a digit for the first decimal place. For example, the value of 123 in Local:6:I.Data[1] indicates the drive is rotating at 12.3 Hz.

An example of DeviceNet communication between the drive and PLC to get the motor rotating is shown in **Figure 14-50**. In order to get the motor started:

- The DeviceNet scanner has to be enabled.
- The drive must be powered up and fault free.
- The Start Command has to be issued.
- No stop command can be present.
- Speed reference has to be set.

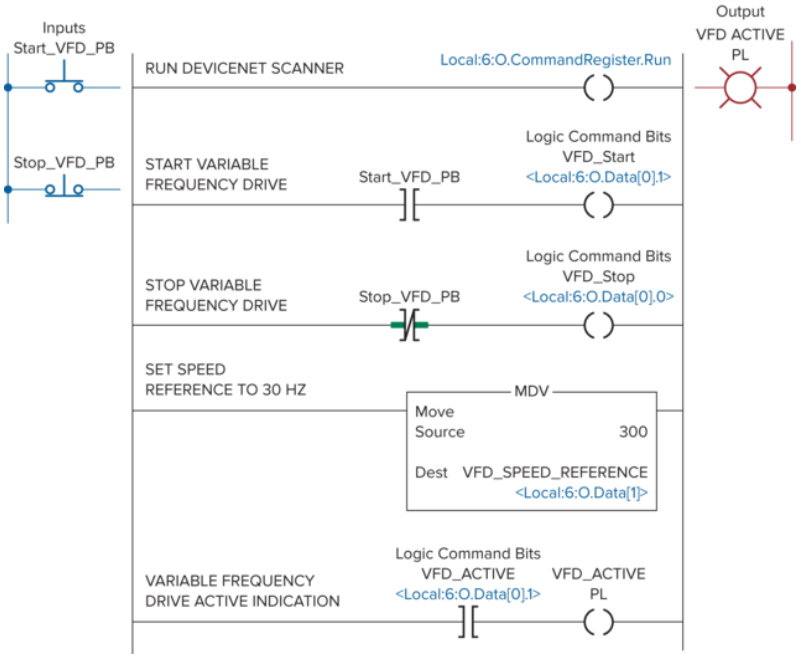


Figure 14-50 DeviceNet communication between the drive and PLC.

OceanofPDF.com

14.9 PLC and HMI Programming

HMI development is paired with PLC programming in order to visualize certain elements of a process controlled by a PLC. Allen Bradley **Studio 5000 and FactoryTalk View** are popular types of software used to collaboratively create PLC instructions and touchscreen applications for use in process control systems. This combination allows an HMI operator to gain more control over the process by issuing directions and receiving feedback information from the process being controlled. The procedure involves setting up communication and data transfer connections between the two through the network. Following is a brief summary of some of the basics associated with navigating and programming Studio 5000 and FactoryTalk View software.

- **Studio 5000**—Refers to the **Logic 5000** programming software used to collaboratively create a PLC/HMI application.
- **HMI Terminal**—The HMI touch screen terminal contains all hardware, software and communication protocols used to access and apply process applications. The Allen Bradley line of **PanelView** graphic terminals are fully packaged hardware, software, and communications HMI units.
- **Explorer Window**—The **Application** tab of the window contains the editors for creating and editing your application, and the **System** folder for setting up FactoryTalk Security. The **Communications** tab contains the communications tree for the development computer.
- **HMI Project**—Refers to the entire structure for the HMI project. It includes all tools needed for development of displays and is saved as an **.Sed file**.
- **Display**—The HMI display touch screen is used to control and monitor a process. Graphic objects on the HMI screen either accept (**read**) data from the PLC or send (**write**) data to the PLC.
- **Protocol**—A protocol is a set of **rules and standards** that define how data is exchanged between devices.
- **EtherNet/IP**—Is an **open-source** industrial communications protocol. It is widely used to connect sensors, controllers, and control systems in the same network.
- **RSLinx Enterprise**—The communications driver used to communicate, via EtherNet/IP, between the PLC Program and its HMI application. This includes the exchange of tags between the two.
- **Graphics Library**—Is used as an aid in creating computer graphics for the HMI display screen and include:
  - **Displays**—The **Main screen** display is used to create a new application or edit an existing one. Most applications require the use of more than the one main display screen to provide user interaction with the process. Navigation commands are used to allow the operator to move between the different screens.
  - **Global Objects**—These are single objects which may be use in many instances in a project. One example would be that of a tank object that could be used to represent different types of tanks.
  - **Symbol Factory**—This source contains a large number of high-quality industrial automation graphics, which can be imported into your displays.

  - **Libraries—**Used to store objects that can be reused and conveniently access to reduce development time.
  - **Images—**Contains the most basic of images such as Pilot Lights and Push Buttons.

- **Graphic Object Properties—**The graphic object States tab is one of several tabs in the Properties dialog box. It is used to set up the **tag value** for each state of a graphic object.
- **OPC UA—**The Open Platform Communications United Architecture is a highly reliable and secure mechanism for transferring information between servers.
- **HMI Tags—**They are used to transfer your PLC program tags over to the HMI. These tags provide additional properties for data manipulation and runtime security. They are created in the Tags editor and stored in an HMI server's tag database. While direct addressing of tags can be used, in most cases, HMI Tags are the preferred way of communicating between the HMI and field devices.
- **Tag Database—**Used for storing, deleting and editing of tags. The Tag Browser allows you to browse the tag database and select tags imported from the PLC database into the project's tag database. While direct addressing can be used, in most cases, HMI Tags are the preferred way of communicating between the HMI and field devices.
- **Servers—**Server store, send, and receives data. A PLC local station contains one HMI server, one Data server, and one Alarm and Event server.

- **Data Server—**The data server provides access to devices on the communication network. Each graphic object on an HMI screen is associated with one of the PLC controller tag names. Data server direct reference tags allow you to browse the controller and link the HMI object directly to the controller tag. This eliminates creating an HMI tag and then linking that back to a controller tag.
- **HMI Server—**The HMI server stores the HMI project components such as graphic displays.
- **Alarm Server and Event Server—**Used to detect alarms, monitor alarms, keep all alarm and event processing in the controller. The difference between alarms and events is that alarms are unexpected and might need corrective action, while events are expected and of importance to the operator.
- **Test Application—**Used to test the application, at any stage, for proper operation. This is done by going to the Application menu and clicking Test Application, or the Test Application tool.
- **Runtime Application—**Before you can run your application, you must create a runtime version. A runtime version compiles all of the necessary application information into a single file **(. mer)** required to run the application.

Figure 14-51 is an example of a Studio 5000 and FactoryTalk View 3-wire motor control application. A summary of the PLC/HMI procedures involved in the development of the application are outlined as follows:

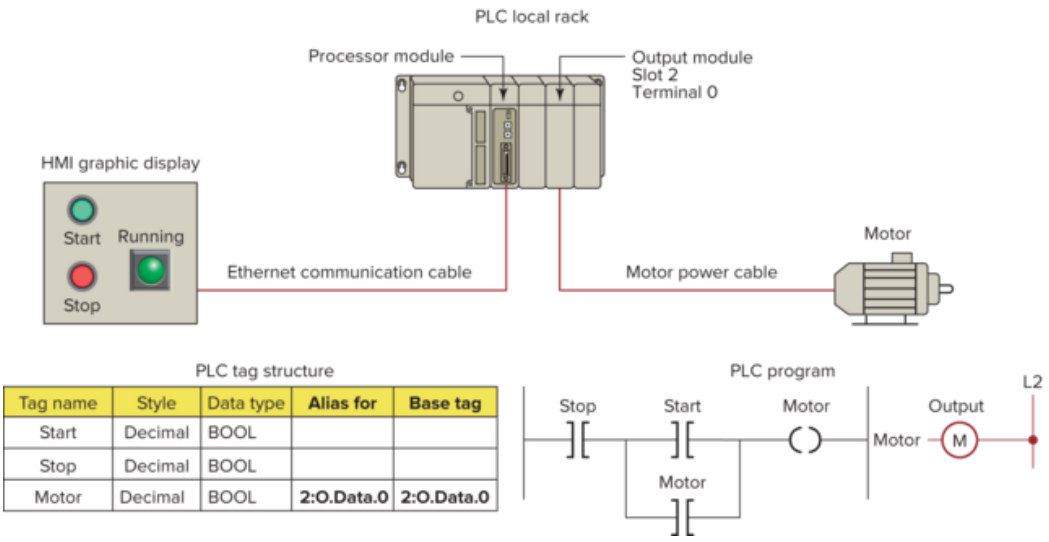


Figure 14-51 PLC/HMI Motor Control Application.

PLC Programming

Studio 5000 is used to create the PLC Program logic rung and Tag Structure.

- The status bit (0 or 1), of the PLC program START instruction, is controlled by the START button on the HMI touch screen.
- Similarly, the status bit (0 or 1), of the PLC program STOP instruction, is controlled by the STOP button on the HMI touch screen.
- The status bit (0 or 1), of the HMI screen RUNNING pilot light, is controlled by the PLC program MOTOR instruction. Whenever this instruction is True it's status bit (0 or 1) will be set to 1, switching the green pilot light ON.
- The PLC program MOTOR output instruction also controls power flow to the field connected motor by way of the rack output module. Whenever this output instruction is True it's the status bit will be set to 1, switching the motor ON.

Graphic Display

Open Displays-New to begin the process.

- Start by creating a virtual graphic object on the display for the START and STOP push buttons, and the RUNNING pilot light indicator.
- From the States tab of the Properties dialog box set the tag value for each state of the virtual graphic object. The tag values are set to match that states of the real field device they emulate.
- The two states of the momentary **normally open** START push button are set with a numerical tag value of 1 when the button is pressed and 0 when it is released.

- The two states of the momentary **normally closed** STOP push button are set with a numerical tag value of 0 when the button is pressed and 1 when it is released.
- The two-state RUNNING pilot light indicator uses **two colors**, Gray and Green, to monitor the operating state of the motor.
- Whenever the numerical tag value of the MOTOR instruction is **0** the light will be set to **Gray**, indicating that the motor is not running.
- Whenever the numerical tag value of the MOTOR instruction is **1** the light will be set to **Green,** indicating that the motor is running.

HMI/PLC Communication

The HMI/PLC communications link allows PLC based tags to be controlled from the HMI application. The connection between the two software packages involves several tasks which are summarized as follows.

- Open Factory Talk Linx - Communications Setup to begin the process.
- Within the Communication Setup, add your PLC to the communications tree.
- Use the online feature in FactoryTalk View Studio to link the PLC program tag structure to the HMI.
- For this application, the tag structure is made up of three BOOL tags named Start, Stop, and Motor.
- The HMI application will toggle these bits and write and read them back for display purposes.
- Add the associated tag from the tag structure to each HMI graphic.
- This involves launching the Properties menu of the graphic object, navigating to the Connections tab, and clicking on the tag Name under the Tag column for the Value modifier.
- At this point, you should have a Tag Browser window appear on your screen. This window allows one to select the appropriate tag for the Value property.

Runtime Application

A runtime version of the application is created in order to transfer the application to the HMI terminal. The procedure involves the following steps:

- Navigate to tool bar Applications.
- Create, name (Motor Control) and save the new runtime application.
- Transfer this runtime application to the HMI terminal by going to tools, Transfer Utility, and selecting your source file (Motor Control) and downloading it.

OceanofPDF.com



CHAPTER 14 REVIEW QUESTIONS

1. Compare continuous and batch processes.
2. Compare centralized and distributive control systems.
3. State the basic function of each of the following as part of a process control system:
   a. Sensors
   b. Human-machine interface
   c. Signal conditioning
   d. Actuators
   e. Controller
4. State the purpose of each of the following types of screens associated with HMIs:
   a. Trend values
   b. Operational summary
   c. Alarm summary
5. What is the main characteristic of a closed-loop control system?
6. State the function of each of the following parts of a closed-loop control system:
   a. Set-point
   b. Process variable
   c. Error amplifier
   d. Controller
   e. Output actuator
7. Explain how on/off control works.
8. How does the proportional controller eliminate the cycling associated with on/off control?
9. Explain how a motor-driven control valve action can provide analog control.
10. How does time proportioning provide analog control?
11. What process error or deviation is produced by a proportional controller?
12. What term of a PID control is designed to eliminate offset?
13. What does the derivative action of a controller respond to?
14. List the three gain adjustments used in tuning the response of a PID control loop.
15. Compare manual, autotune, and intelligent tuning of a PID controller.
16. How many input and output values are normally referenced in a PLC PID instruction?
17. What information is contained in the process variable and control variable elements of a PID instruction?

18. State the function of each of the following elements of a PLC motion control system:

a. Programmable controller

b. Motion module

c. Servo drive

d. Servo motor

19. What does each axis of a robot arm function as?

20. List four types of communication tasks provided by local area networks.

21. Name three common types of transmission media.

22. What are the three general levels of functionality of industrial networks?

23. Define the term *node* as it applies to a network.

24. Explain the physical layout of devices on a network for each of the following network topologies:

a. Star

b. Bus

25. Compare device and process bus networks.

26. Define the term *protocol* as it applies to a network.

27. What is the function of a network gateway?

28. Define the term *access method* as it applies to a network.

29. Summarize the token passing network access method.

30. Summarize the collision detection network access method.

31. Summarize the polling network access method.

32. Compare parallel and serial data transmission.

33. Compare half-duplex and full-duplex data transmission.

34. Explain how networking schemes minimize the amount of wiring required.

35. What type of access control is used with DH+?

36. Compare the transmitting distances of RS-232 and RS-422/485 serial types.

37. What is DeviceNet used for?

38. List three pieces of information obtained from DeviceNet devices by the network scanner.

39. What is ControlNet used for?

40. Explain how redundant media works.

41. Define the term *bandwidth* as it applies to a network.

42. What is Ethernet/IP used for?

43. What type of protocol does Modbus use?

44. What is Fieldbus used for?

45. Summarize the two main functions of a SCADA system.

46. In what way does distributive control differ from the supervisory control of a SCADA system?

47. What are Allen Bradley Studio 5000 and FactoryTalk View software used for?

48. List three components of a typical HMI touch screen terminal.

49. Explain how the HMI display touch screen communicates with the PLC program to control and monitor a process.

50. Which industrial network communications protocol is used by FactoryTalk?

51. Which screen display is used to create a new application?

52. Explain what the graphic object States tab is used for.

53. Explain what HMI tags are used for.

54. Outline how the data server provides access to graphic objects on the HMI screen.

55. Outline how the Test Application and Runtime Application are paired in the development of an HMI application.

OceanofPDF.com



CHAPTER 14 PROBLEMS

1. Distributive control systems have to be network based. Why?

2. Assume an alarm is sounded in a control system with an electronic HMI interface. How would you proceed to identify and solve the problem?

3. How would an on/off controller respond if the deadband were too narrow?

4. In a home heating system with on/off control, what will be the effect of widening the deadband?

5. a. Calculate the proportional band of a temperature controller with a 5% bandwidth and a set-point of $500^{\circ}\mathrm{F}$.

b. Calculate the upper and lower limits beyond which the controller functions as an on/off unit.

6. Explain the advantage of using a $4\text{–}20\ \mathrm{mA}$ current loop as an input signal compared to a $4\text{–}5\ \mathrm{V}$ input signal.

7. What does the term *deterministic* mean, and why is it important in industrial communications?

8. How might a SCADA system be applied to determine the production rate of a bottled product over a two-week period?

OceanofPDF.com

15

ControlLogix Controllers

Programmable logic controllers continue to evolve as new technologies are added to their capabilities. The PLC started out as a replacement for banks of relays used to turn outputs on and off as well as for timing and counting functions. Gradually, various math and logic manipulation functions were added. In order to serve today's expanding industrial control system needs, leading automation companies have created a new class of industrial controllers called ***programmable automation controllers*** or ***PACs*** (**Figure 15-1**). They look like PLCs in their physical appearance but incorporate advanced control of communication, data logging, and signal processing, motion, process control, and machine vision in a single programming environment.



Figure 15-1 Programmable automation controllers (PACs).
Source: Image Courtesy of Rockwell Automation, Inc.

The Allen-Bradley programmable automation controller family includes the ControlLogix system, CompactLogix system, FlexLogix system, SoftLogix 5800 controller, and DriveLogix system. *Software* is the essential difference between PACs and PLCs. Basically, the ladder logic configuration does not change but the addressing of the instructions changes. Application of the software that pertains to the Logix control platform of controllers will be covered in the various sections of this chapter. Knowledge of basic ladder logic instructions and functions (bit, timer, counter, etc.) covered in previous chapters of the text is assumed and is thus not repeated in this chapter.

OceanofPDF.com

Part 1 Memory and Project Organization

Part Objectives

After completing this part, you will be able to:

- Outline project organization
- Define tasks, programs, and routines
- Identify data file types
- Organize and apply the various data file types

Memory Layout

ControlLogix processors provide a flexible memory structure. There are no fixed areas of memory allocated for specific types of data or for I/O. The internal memory organization of a ControlLogix controller is configured by the user when creating a project with RSLogix 5000 software (Figure 15-2). This feature allows the program data to be constructed to meet the needs of your applications rather than requiring your application to fit a particular memory structure. A ControlLogix (CLX) system can consist of anything from a stand-alone controller and I/O modules in a single chassis, to a highly distributed system consisting of multiple chassis and networks working together.



Figure 15-2 RSLogix 5000 screen.

Source: Image Courtesy of Rockwell Automation, Inc.

Configuration

Configuration of a modular CLX system involves establishing a communications link between the controller and the process. The programming software needs to know what CLX hardware is being used

in order to be able to send or receive data. Configuration information includes information about the type of processor and I/O modules used.

LogixDesigner programming software is used to set up or *configure* the memory organization of an Allen-Bradley ControlLogix controller. ***RSLinx*** communication software is used to set up a communications link between RSLogix 5000 programming software and the ControlLogix hardware as illustrated in Figure 15-3. To establish communications with a controller, a driver must be created in RSLinx software. This driver functions as the software interface to a hardware device. The ***RSWho*** is the network browse interface that provides a single window to view all configured network drivers.

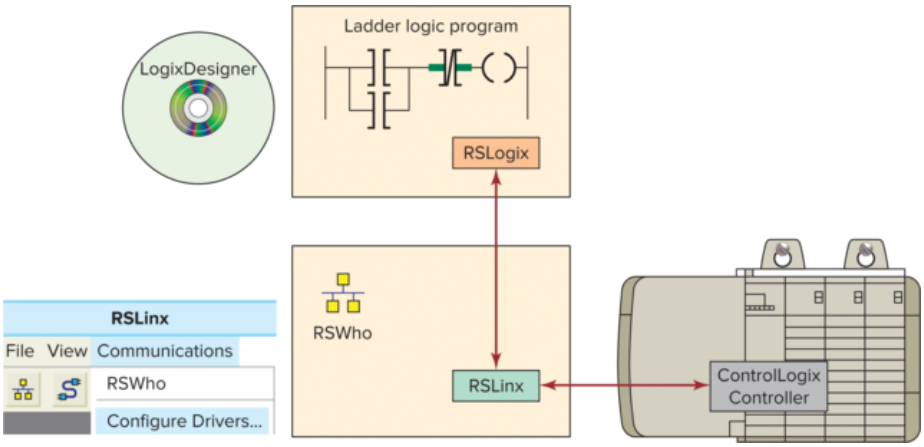


Figure 15-3 RSLinx and LogixDesigner software.

Figure 15-4 shows an example of the ControlLogix's *controllers properties and modules properties* dialog boxes used as part of the configuration process. The parameters shown are typical of what general information is required. After first configuring the controller, the I/O modules are configured using RSLogix 5000 software. Modules will not work unless they have been properly configured. The software contains all the hardware information needed to configure any ControlLogix module.

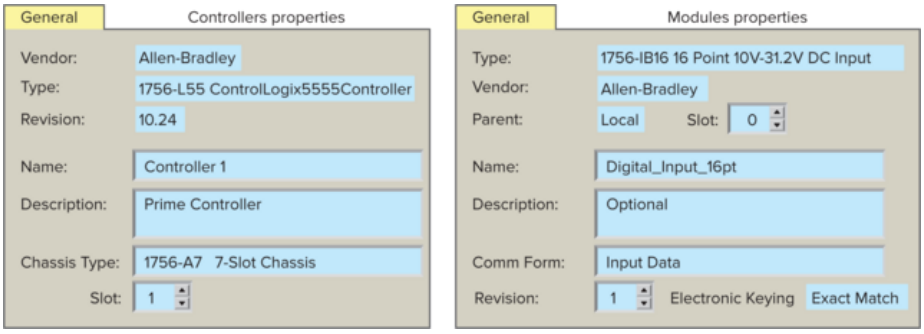


Figure 15-4 Controllers properties and modules properties dialog boxes.

Project

RSLogix software stores a controller's programming and configuration information in a file called a ***project***. The block diagram of the processor's project file is shown in Figure 15-5. A project file contains all information relating to the project. The main components of the project file are tasks, programs, and routines. A controller can hold and execute only one project at a time.

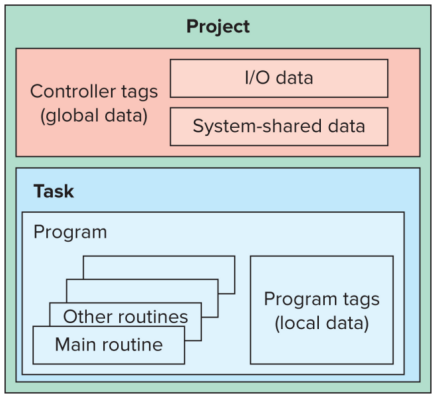


Figure 15-5 ControlLogix processor program file.

The RSLogix 5000 controller organizer (Figure 15-6) displays the project organization in a tree format showing tasks, programs, routines, data types, trends, I/O configuration, and tags. Each folder groups common functions together. This structure simplifies the navigation and the overall view of the whole project.

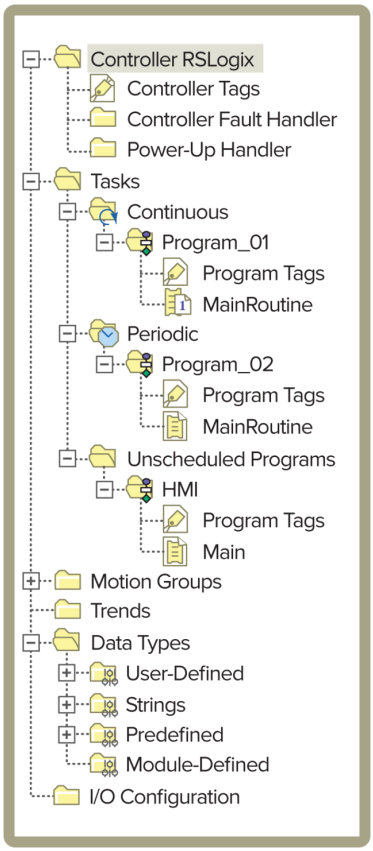


Figure 15-6 Controller organizer tree.

In front of each folder, there is an icon containing a + sign or a − sign. The + sign indicates that the folder is closed. Click it to expand the tree display and display the files in the folder. The − sign indicates that the folder is already open and its contents are visible. Clicking on the right mouse button brings up many different, context-sensitive popup menus. Often, you find that this is a shortcut to access the property window or menu options from the menu bar.

Tasks

Tasks are the first level of scheduling within a project. A task is a collection of scheduled programs. When a task is executed, the associated programs are executed in the order listed. This list of programs is known as the program schedule. Tasks provide scheduling based on specific conditions and do not contain any executable code. Only one task may be executing at any given time. The number of tasks a controller can support depends on the specific controller. The main types of tasks (**Figure 15-7**) include:

- ***Continuous*** tasks execute nonstop but are always interrupted by a periodic task. Continuous tasks have the lowest priority. A ControlLogix continuous task is similar to the File 2 in the RSLogix 500 platform. Here the continuous task is named Main Task.

- ***Periodic*** tasks function as timed interrupts. They interrupt the continuous task and execute for a fixed length of time at specific time intervals.
- ***Event*** tasks also function as interrupts. Rather than being an interrupt on a timed basis, an event task is triggered by an event that happened or failed to happen.

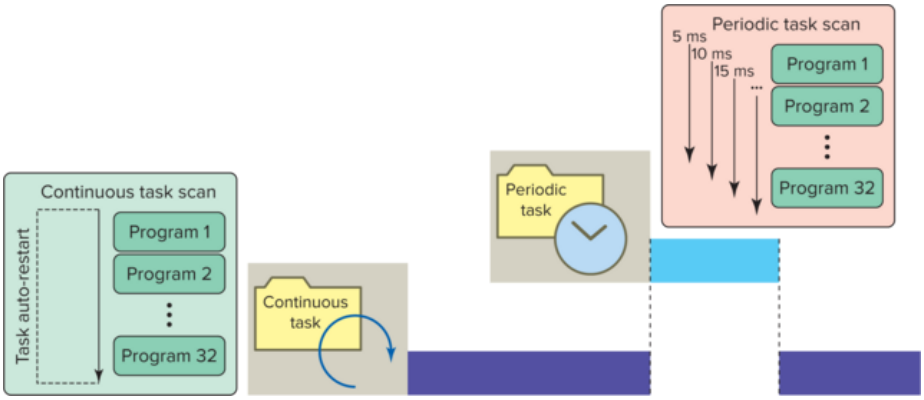


Figure 15-7 Continuous and periodic tasks.

Programs

Programs are the second level of scheduling within a project. The function of the folders under Main Task is to determine and specify the order in which the programs execute. There is no executable code within a program. Routines within programs will execute in the order listed below their associated task in the controller organizer as shown in **Figure 15-8**. In this example, according to the listed order, the Main Program is scheduled to execute first, Program_A second, and Progam_B third. Programs that are not assigned to a task are unscheduled. Unscheduled programs are downloaded to the controller but do not execute. These programs remain unscheduled until needed. Depending on the RSLogix 5000 software version as many as 100 programs could be scheduled within each task.

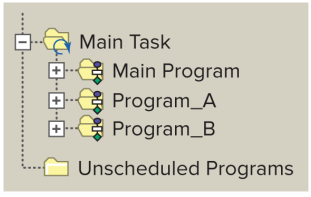


Figure 15-8 Order of execution of programs.

Routines

Routines are the third level of scheduling within a project and provide the executable code for the project. Each routine contains a set of logic elements for a specific programming language. When a

routine is created it is specified as ladder logic, sequential function chart, function block diagram, or structured text (Figure 15-9). Any one routine must be completely in the same language. The number of routines per project is limited only by the amount of controller memory. Libraries of standard routines can be created that can be reused on multiple machines or applications. A routine can be assigned as one of the following types:

- A ***main routine*** is one configured to execute first when the program runs. Each program will have one main routine typically followed by several or many subroutines.
- A ***subroutine*** is one that is called by another routine. Subroutines are used for large or complex programming tasks or tasks that require more than one programming language.
- A ***fault routine*** is one that executes if the controller finds a program fault. Each program can have one fault routine, if desired.

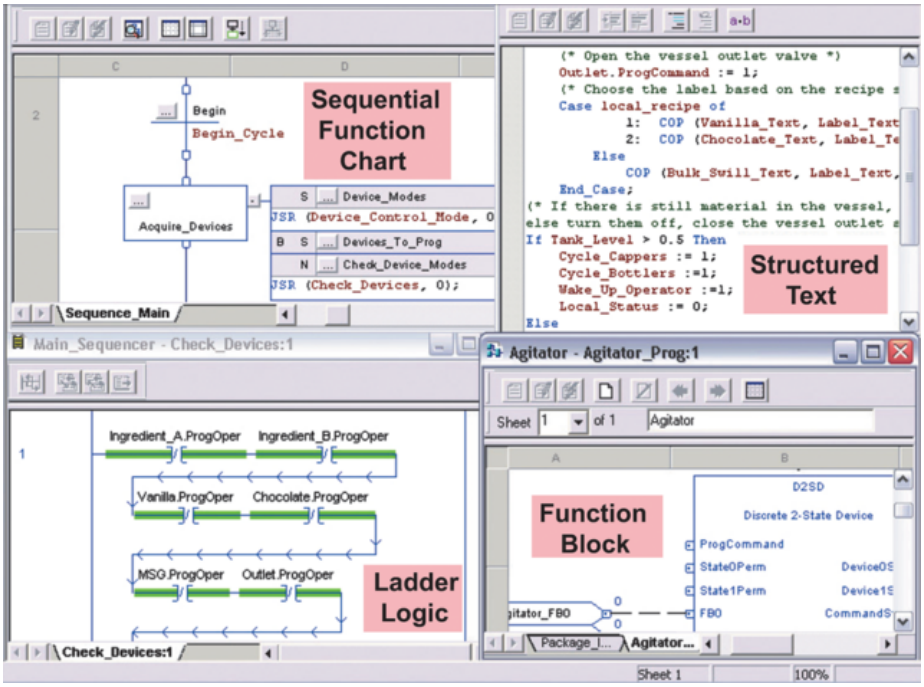


Figure 15-9 Each routine contains a set of logic elements for a specific programming language.

Source: Image Courtesy of Rockwell Automation, Inc.

Tags

Unlike conventional controllers, ControlLogix uses a *tag-based* addressing structure. Tags are meaningful names, descriptive of your application and not merely generic addresses. A tag is created to represent the data and identify areas in the controller's memory where these data are stored. In applications developed using RSLogix 5000 software, there are no predefined data tables such as in RSLogix 500. When you want to use or monitor data in a program you use tag names to refer to the memory locations, as illustrated in Figure 15-10. This functionality allows you to name your data specifically for their functions within the control program while providing self-documented logic. Whenever you wish to group data, you create an array, which is a grouping of tags of similar types.

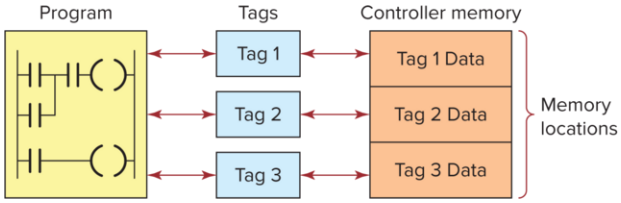


Figure 15-10 Tags used to assign memory locations.

Scope refers to which programs have access to a tag. The scope of a tag must be specified when you create the tag. There are two scopes for tags: program scope and controller scope. A ***program tag*** consists of data that can be accessed only by routines within a specific program (local data). The routines in other programs cannot access program scoped tags of another program. A ***controller tag*** consists of data that are accessible by all routines within a controller (global data). **Figure 15-11** shows two programs, A and B, within a project. Note that each program has program scope tags with identical names (Tag_1, Tag_2, and Tag_3). Because they are program scoped, there is no relationship between them, even though they have the same name. The program scope data are accessible only to the routines within a single program. The same tag name may appear in different programs as local variables because you can select the scope in which to create the tag.

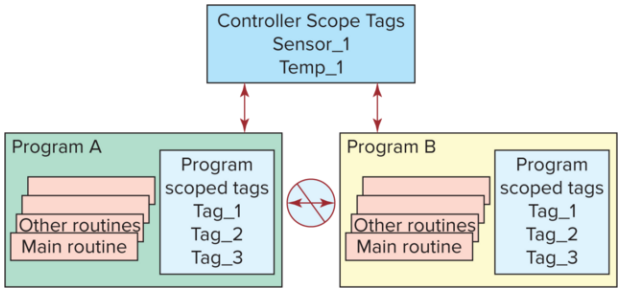


Figure 15-11 Program scoped and controller scoped tags.

The scope of a tag must be declared when you create the tag. **Figure 15-12** shows program and controller scoped tags as listed in the controller organizer under the program they are assigned to. I/O tags are automatically created as controller scoped tags.

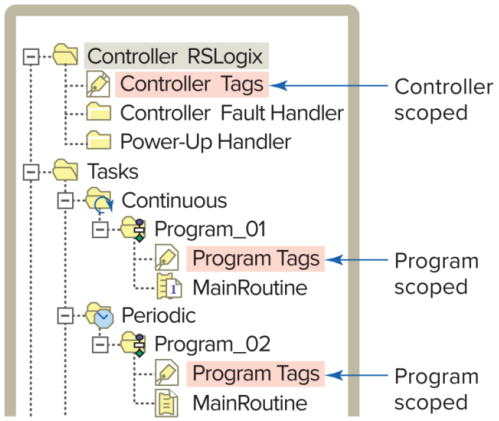


Figure 15-12 Listing of program and controller scoped tags.

There are four different tag types: base, alias, produced, and consumed tags. The tag type defines how the tag operates within the project. A ***base tag*** stores various types of data for use by logic in the project. This tag defines a memory location where data are stored. Base tag memory use depends on the type of data the tag represents. An example of the base tag Local:2:O.Data.4 is shown in **Figure 15-13** and is based on the following format:

Location	Network location LOCAL = same chassis as the controller
Slot	Slot number of I/O module in its chassis
Type	Type of data I = input O = output C = configuration S = status
Member	Specifies the type of data that the module can store. Digital (discrete) I/O modules use a DATA member. Analog I/O modules use a Channel Member (CH#)
SubMember	Specific data related to a Member.
Bit	Specific point on a digital I/O module; depends on the size of the I/O module (0-31 for a 32-point module)

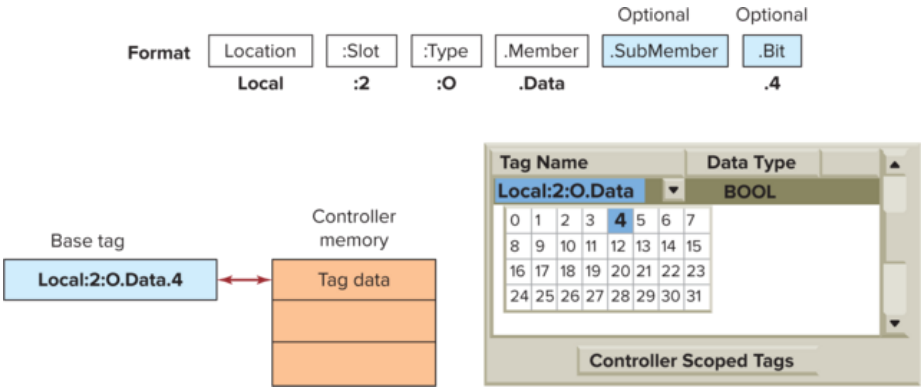


Figure 15-13 Base tag.

An ***alias tag*** is used to create an alternate name (alias) for a tag. The alias tag is simply another name for an already named memory location. An alias tag can refer to a base, alias, consumed, or produced tag. The alias tag is often used to create a tag name to represent a real-word input or output. **Figure 15-14** shows an example of the use of an alias tag. The alias tag (Fan_Motor) is linked to the base tag (<local:2:O.Data.5>) so that any action to the base also happens to the alias and vice versa. The alias name is easier to understand and easier to relate to the application, while the base tag contains the physical location of the output point in the ControlLogix chassis.

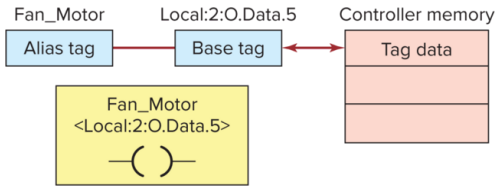


Figure 15-14 Alias tag linked to a base tag.

Produced/consumed tags are used to share tag information over a network between two or more devices. A produced tag sends data while a consumed tag receives data. Produced tags are always controller scoped. **Figure 15-15** shows an example of how a controller can produce data and send them over the network to two controllers that use or consume the data. The producing controller will have a tag that is of the produced type, whereas the consuming controllers will have a tag with the exact same name that is of the consumed type.

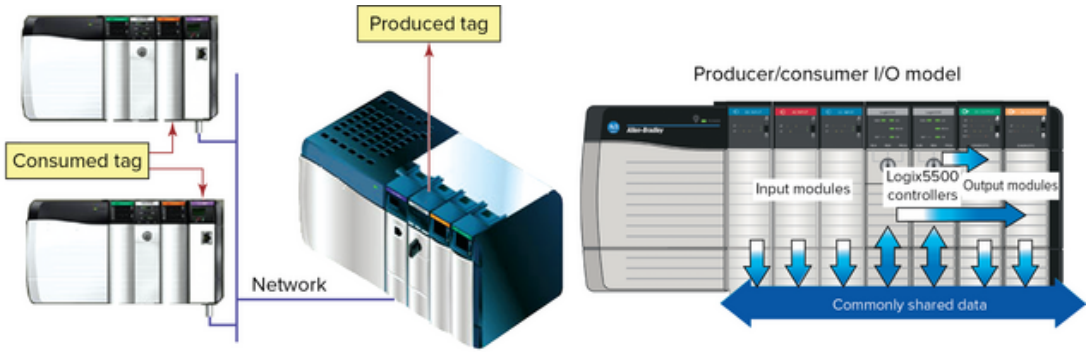


Figure 15-15 Produced/consumed tags used to share information.

Source: Image Courtesy of Rockwell Automation, Inc.

When you design your application, you configure it to both produce globally to other controllers in the system via the backplane and to consume tags from other controllers. This feature allows you to be selective about which data are sent and received by any controller. Likewise, multiple controllers can connect to any data being produced, thereby preventing the need to send multiple messages containing the same data.

Logix controllers are based on 32-bit operations. The types of data that can be a base tag are BOOL, SINT, INT, DINT, and REAL, as illustrated in **Figure 15-16** and listed below. The controller stores all data in a minimum of 4 bytes or 32 bits of data.

- A **BOOL** or Boolean base tag is 1 bit of data stored in bit 0 of a 4 byte memory location. The other bits, 1 to 31, are unused. BOOLs have a range of 0 to 1, off or on respectively. Page 330
- A **SINT** or Short Integer base tag uses 8 bits of memory and stores the data in bits 0 to 7. These bits are sometimes called the low byte. The other 3 bytes, bits 8 to 31, are unused. SINTs have a range of −128 (negative values) to 127 (positive values).
- An **INT** or Integer base tag is 16 bits, bits 0 to 15, sometimes called the lower bytes. Bits 16 to 31 are unused. INTs have a range between −32,768 and 32,767.
- A **DINT** or Double Integer base tag uses 32 bits, or all 4 bytes, and has the following range: -2^{31} to $2^{31} - 1$ (−2,147,483,648 to 2,147,483,647).
- A **REAL** base tag also uses 32 bits of a memory location and has a range of values based on the IEEE Standard for Floating-Point Arithmetic.

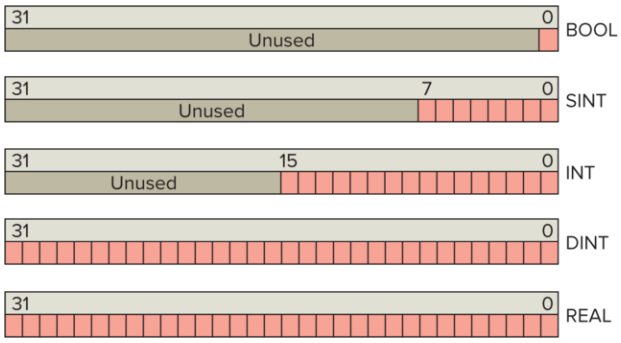


Figure 15-16 Types of base tag data.

Structures

There is another class of data types called structures. A **structure-type tag** is a grouping of different data types that function as a single unit and serve a specific purpose. An example of an RSLogix structure is shown in **Figure 15-17**. Each element of a structure is referred to as a member and each member of a structure can be a different data type.

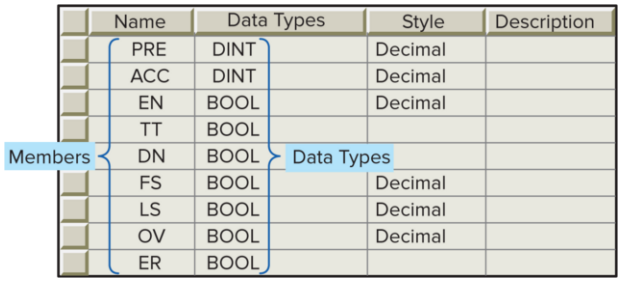


Figure 15-17 Structure-type tag.

There are three different types of structures in a ControlLogix controller: *predefined, module-defined,* and *user-defined.* The controller creates ***predefined* structures** for you that include timers, counters, messages and PID types. An example of a predefined counter instruction structure is shown in **Figure 15-18**. It is made up of the preset value, the accumulated value, and the instruction's status bits.

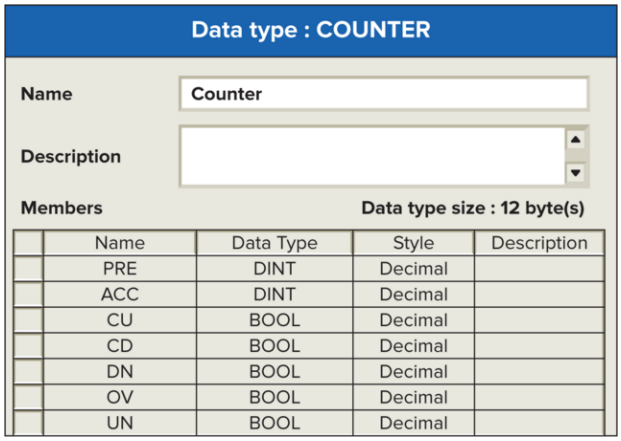


Figure 15-18 Predefined structure.

***Module-defined* structures** are automatically created when the I/O modules are configured for the system. When you add input or output modules a number of defined tags are automatically added to the controller tags. **Figure 15-19** shows the two tags (Local:1:C and Local:1:I) created after a digital input module has been added. Tags of these types are created to store input, output, and configuration data for the module. Input tags labeled Data contain the actual input bits from the module. Configuration tags determine the characteristics and operation of the module. The name Local indicates that these tags are in the same rack as the processor. The 1 indicates that the module occupies slot 1 in the chassis. The letters I and C indicate whether the data are input data or configuration data.

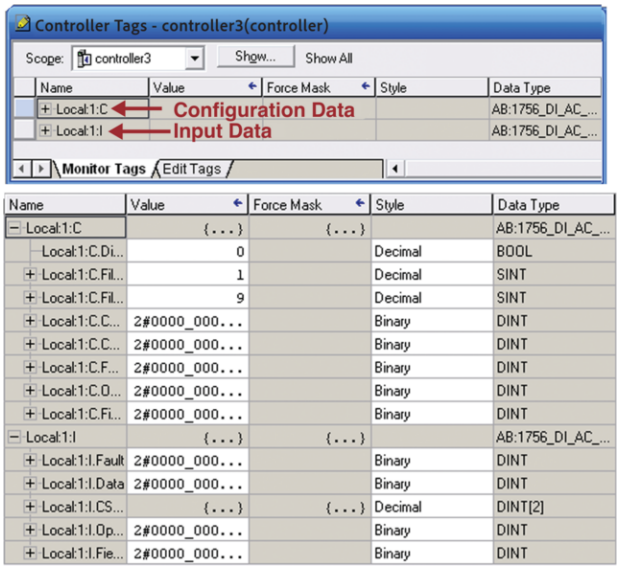
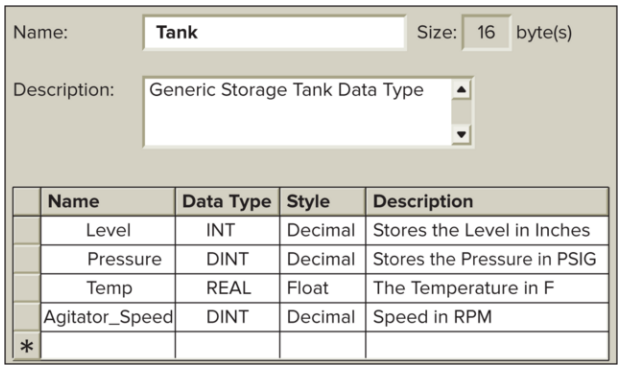


Figure 15-19 Module-defined structure for a digital input module.

Source: Image Courtesy of Rockwell Automation, Inc.

A ***user-defined*** **structure** supplements the predefined structures by providing the ability to create custom-defined structures to store and handle data as a group. **Figure 15-20** illustrates a user-defined structure that contains data for a storage tank. All data relative to the tank are stored together. In the design stage the programmer creates a generic user-defined memory structure that contains all the different aspects of the storage tank. Each member has a meaningful name and is created in the appropriate data type and style like REAL (floating point) for temperature and DINT (decimal) for agitator speed in rpm. Installation and maintenance personnel can easily locate all data associated with the operation of the tank since all the information is stored together.



	Name	Data Type	Style	Description
	Level	INT	Decimal	Stores the Level in Inches
	Pressure	DINT	Decimal	Stores the Pressure in PSIG
	Temp	REAL	Float	The Temperature in F
	Agitator_Speed	DINT	Decimal	Speed in RPM
*				

Figure 15-20 User-defined storage tank structure.

Creating Tags

There is more than one way to create tags. You may create tags in the tag editor before your program is entered, enter tag names as you program, or use question marks [?] in place of tag names and assign the tags later. Figure 15-21 shows an example of a controller scope base tag created in the new tag dialog box. When defining tags, the following information has to be specified:

- A *tag name,* which must begin with an alphabetic character or an underscore (_). Names can contain only alphabetic characters, numeric characters, or underscores and may be up to 40 characters in length. They may not have consecutive or trailing underscore characters, are not case sensitive and cannot have spaces in the tag name.
- An optional *tag description,* which may be up to 120 characters in length.
- The *tag type:* base, alias, or consumed.
- The *data type,* which is obtained from the list of predefined or user-defined data types.
- The scope in which to create the tag. Your options are the controller scope or any one of the existing program scopes.
- The *display style* to be used when monitoring the tag in the programming software. The software will display the choices of available styles.
- Whether or not you want to make this tag available to other controllers and the number of other controllers that can consume the tag.

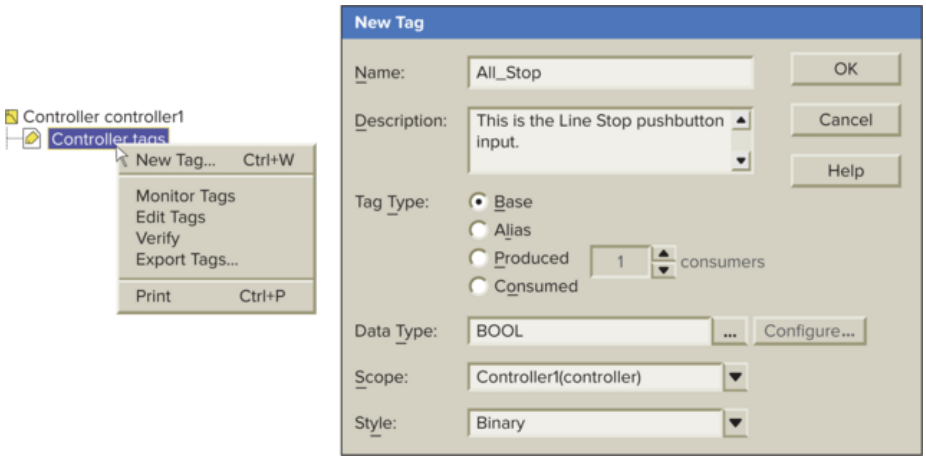


Figure 15-21 Controller scope base tag.

Source: Image Courtesy of Rockwell Automation, Inc.

Monitoring and Editing Tags

After tags have been created they can be monitored using the Monitor Tags window displayed in Figure 15-22. When ***Monitor Tags*** is selected the actual value(s) for the tags will be shown. The Force Mask column is used to force inputs and outputs when troubleshooting. You can also create new tags or edit existing tags using the Edit Tags window displayed in Figure 15-23. When ***Edit Tags*** is selected new tags may be created, and existing tag properties may be modified.

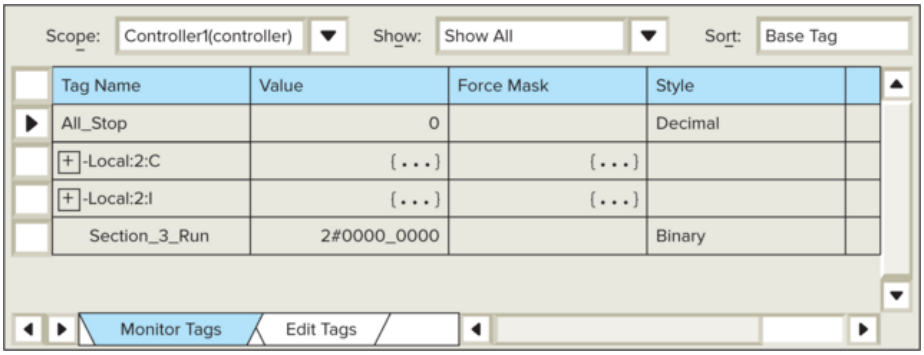


Figure 15-22 Monitor Tags window.

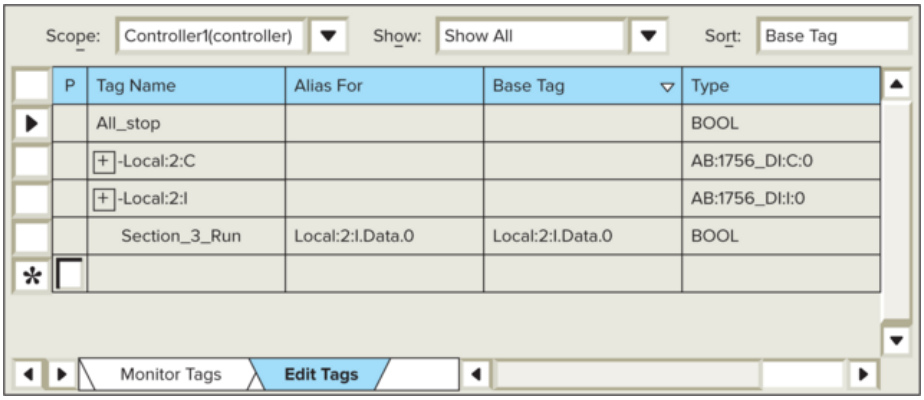


Figure 15-23 Edit Tags window.

Array

Many control programs require the ability to store blocks of information in memory in the form of tables that can be accessed at runtime. An ***array*** is a tag type that contains a block of multiple pieces of data. Each element of an array must be of the *same data type* (e.g., BOOL, SINT, or INT). An array occupies a contiguous block of controller memory. Arrays are similar to tables of values. The use of arrayed data types offers the fastest data throughput (output) from a ControlLogix processor. Because arrays are numerically sequenced tags of the same data type that occupy a contiguous memory location, large amounts of data can be retrieved efficiently. Arrays can be built using 1, 2, or 3 dimensions, as illustrated in **Figure 15-24**, to represent the data they are intended to contain.

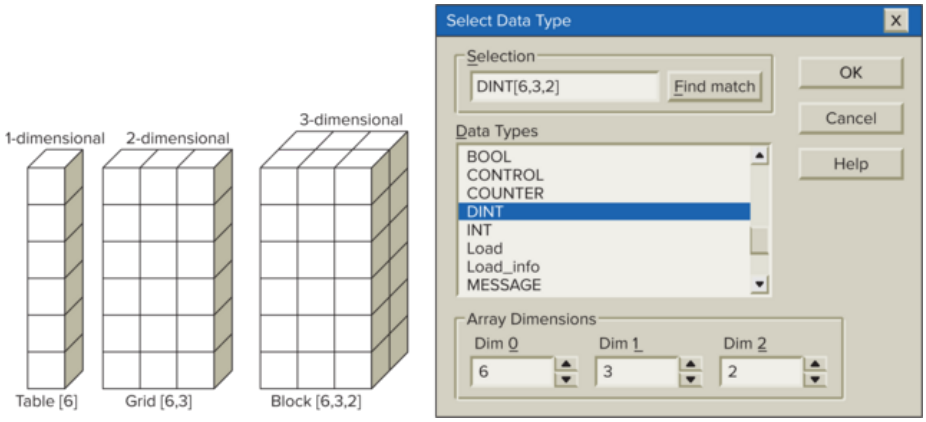


Figure 15-24 Types of arrays.

Source: Image Courtesy of Rockwell Automation, Inc.

A single tag within the array is one element. The element may be a basic data type or a structure. The elements start with 0 and extend to the number of elements minus 1. **Figure 15-25** is an example of the memory layout for a 1-dimensional (one column of values) array created to hold five temperatures. The tag name is Temp and the array consists of 5 elements numbered 0 through 4.

Array - Temp

Data Type - INT[5]

Temp[0]	297
Temp[1]	200
Temp[2]	180
Temp[3]	120
Temp[4]	100

Figure 15-25 Memory layout for a one-dimensional array.

PART 1 REVIEW QUESTIONS

1. Compare the memory configuration of a Logix 5000 controller with that of an RSLogix 500 controller.
2. What does a project contain?
3. List four programming functions that can be carried out using the program organizer.
4. Explain the function of tasks within the project.
5. State the three main types of tasks.
6. What type of tasks function as timed interrupts?
7. Explain the function of programs within the project.
8. Explain the function of routines within the project.
9. Which routine is configured to execute first?
10. Name the four types of programming languages that can be used to program Logix 5000 controllers.
11. What are tags used for?
12. Compare the accessibility of program scope and controller scope tags.
13. Name the tag type used for each of the following:
    - **a.** Create an alternate name for a tag.
    - **b.** Share information over a network.
    - **c.** Store various types of data.
14. What is the difference between a produced tag and a consumed tag?
15. List the five types of base tag data.
16. State the data type used for each of the following:
    - **a.** 32-bit memory storage
    - **b.** On/Off toggle switch
    - **c.** 16-bit memory storage
    - **d.** 8-bit memory storage
17. Describe the make-up of a predefined structure.
18. Describe the make-up of a module-defined structure.
19. Describe the make-up of a user-defined structure.
20. Explain two ways of creating tags.
21. When defining tags what limitations are placed on the entering of a tag name?
22. What is meant by the tag display style?
23. Write an example of an array tag used to hold 4 speeds.

OceanofPDF.com

Part 2 Bit-Level Programming

Part Objectives

After completing this part, you will be able to:

- Know what happens during the program scan
- Demonstrate an understanding of input, output, and internal relay addressing format for a tag-based Logix controller
- Develop ladder logic programs with input instructions and output coil combinations
- Develop ladder logic programs with latched outputs
- Determine the state of output devices for different states of input devices

Program Scan

When a CLX controller executes a program, it must know—in real time—when external devices controlling a process are changing. During each operating cycle, the processor reads all the inputs, takes these values, and energizes or de-energizes the outputs according to the user program. This process is known as the ***program scan***.

Figure 15-26 illustrates the signal flow into and out of a Logix controller during a controller's operating cycle when ladder logic is executing. During the program scan, the controller reads rungs and branches from left to right and top to bottom as follows:

- Only one rung at a time is scanned.
- As the program is scanned, the status of inputs are checked for True (1 or ON) or False (0 or OFF) conditions.
- The status signals from the inputs are sent to the input tags where they are stored.
- As the program is scanned by the processor, inputs are checked for True or False conditions and the ladder logic is evaluated based on these values.
- The resulting ON or OFF action, as a result of evaluating each rung, is then sent to the output tags for storage.
- During the output update portion of the scan, corresponding output values are sent to the process or machine by way of the output module.
- I/O updates occur asynchronously to the scan of the logic. With a ControlLogix processor two separate 32-bit unsynchronized processes go on simultaneously—that is, asynchronously. This means that the module can update the input tag from the field and write the output tag to the field at

any point (or at several points) during the processor's execution of the ladder rungs. The result is more efficiency and control over when the input field device data are updated in the input tag and when the output data resulting from the solved logic are sent to the output modules and their respective field devices.

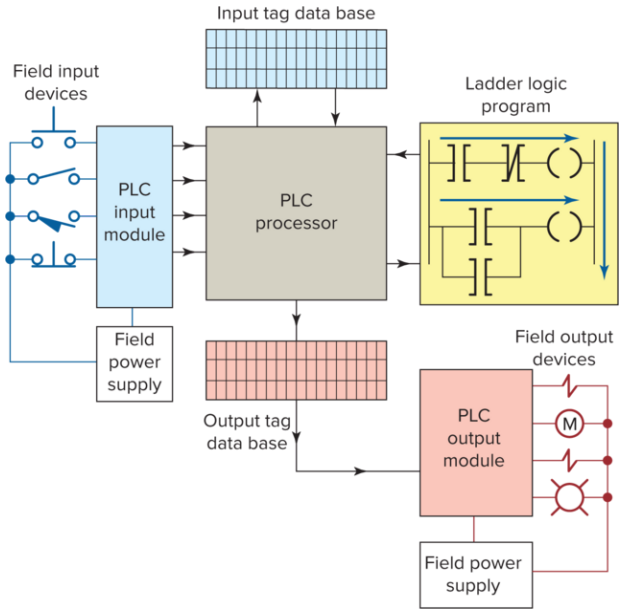


Figure 15-26 Logix controller operating cycle.

Creating Ladder Logic

Although other programming languages are available, ladder logic is the most common programming language for PLCs. The instructions in ladder logic programming can be divided into two broad categories: input and output instructions. The most common input instruction is equivalent to a relay contact and the most common output instruction is the equivalent of a relay coil (**Figure 15-27**). When creating ladder I/O bit instructions, the following rules apply:

- All input instructions must be to the left of an output instruction.
- A rung cannot begin with an output instruction if it also contains an input instruction. This is because the controller tests all inputs for true or false before deciding what value the output instruction should be.
- A rung does not need to contain any input instructions, but it must contain at least one output instruction.
- When a rung has only one output instruction it will always be true.
- The last instruction on a rung must always be an output instruction.
- The XIC, or Examine If Closed contact instruction, checks to see if the input has a value of one. If the input is one, the XIC instruction returns a true value.

- The XIO, or Examine If Open contact instruction, checks to see if the input has a value of zero. If the input is zero, the XIO instruction returns a true value.
- The OTE or Output Energize coil instruction sets the tag associated with it to true or one when the rung has logic continuity. When true it can be used to energize an output device or simply set a value in memory to one.

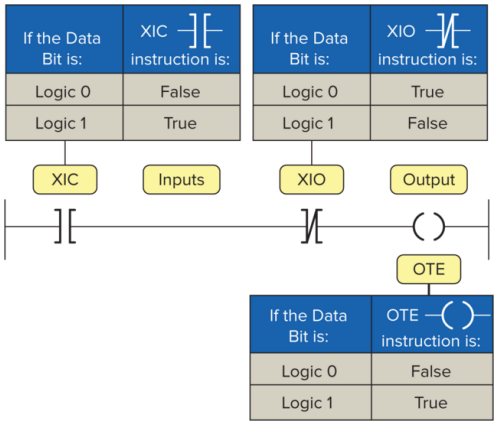


Figure 15-27 Contacts and coil instructions.

ControlLogix PACs support multiple outputs on one rung. CLX controllers allow the use of serial logic that does not conform to traditional electrical hardwired circuits or ladder logic. For example, both of the rungs shown in **Figure 15-28** are valid in RSLogix 5000. However the series connection of outputs would not work if wired that way in an equivalent electrical circuit or programmed that way in RSLogix 500. In both instances in RSLogix 5000, instructions tagA and tagB must be true to energize output tag1 and tag2.

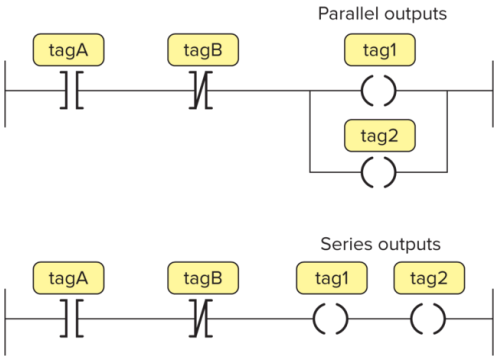


Figure 15-28 Parallel and series outputs.

In ControlLogix output instructions can be placed between input instructions as illustrated in **Figure 15-29**. In this example instructions tagA and tagB must be true to energize output tag1. Instructions tagA and tagB and tagC must all be true before output tag2 is set to energize.

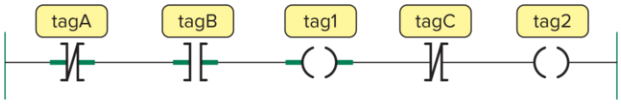
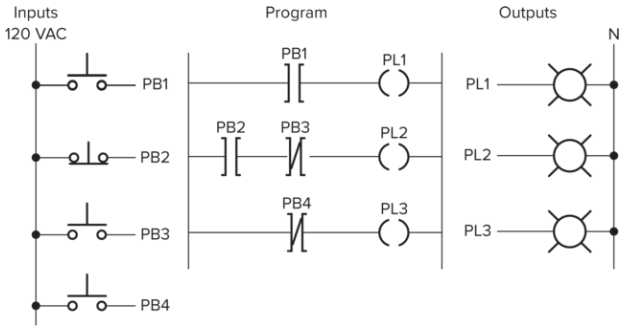


Figure 15-29 Output instruction placed between input instructions.

Figure 15-30 shows a program that uses both the XIC and XIO instructions. The controller evaluates each logic rung base on the rung conditions preceding the output instruction. To determine whether the pilot light is on or off, start by recording the state (0 or 1) of each input device. Keep in mind that the state these devices is completely independent of the program and its state has nothing to do with the program itself. Next, evaluate each rung for logic continuity base on the values (0 or 1) in both the input and output memory address location for each input condition. The truth table shows the resultant outputs you can expect to see.



Truth table

Input device state	Input state				Output state		
	PB4	PB3	PB2	PB1	PL3	PL2	PL1
No PBs pressed	0	0	1	0	1	1	0
PB1 pressed	0	0	1	1	1	1	1
PB3 pressed	0	1	1	0	1	0	0
PB2,3,4 pressed	1	1	0	0	0	0	0

Figure 15-30 Program that uses both the XIC and XIO instructions.

Tag-Based Addressing

Logix 5000 controllers use a tag-based addressing structure. A **tag** is a text-based name for an area of the controller where data is stored. An example of how a tag-based address is implemented using a ControlLogix controller is shown in Figure 15-31. Tag names use a meaningful description of the variable. In this application when the normally closed high limit switch is activated the program will switch the high limit output light on. The addressing format can be summarized as follows:

- The physical address for the tag Limit_switch is Local:1:I.Data.2(C). Local indicates that the module is in the same rack as the processor, 1 indicates that the module is in slot 1 in the rack, I indicates that the module is an input type, Data indicates that it is a digital input, 2 indicates that the limit switch is connected to terminal 2 on the module, and C indicates that it is a controller tag with global access.

- The physical address for the tag High_limit_light is Local:2:O.Data.4(C). Local indicates that the module is in the same rack as the processor, 2 indicates that the module is in slot 2 in the rack, O indicates that the module is an output type, Data indicates that it is a digital output, 4 indicates that the high limit light is connected to terminal 4 on the module, and C indicates that it is a controller tag with global access.

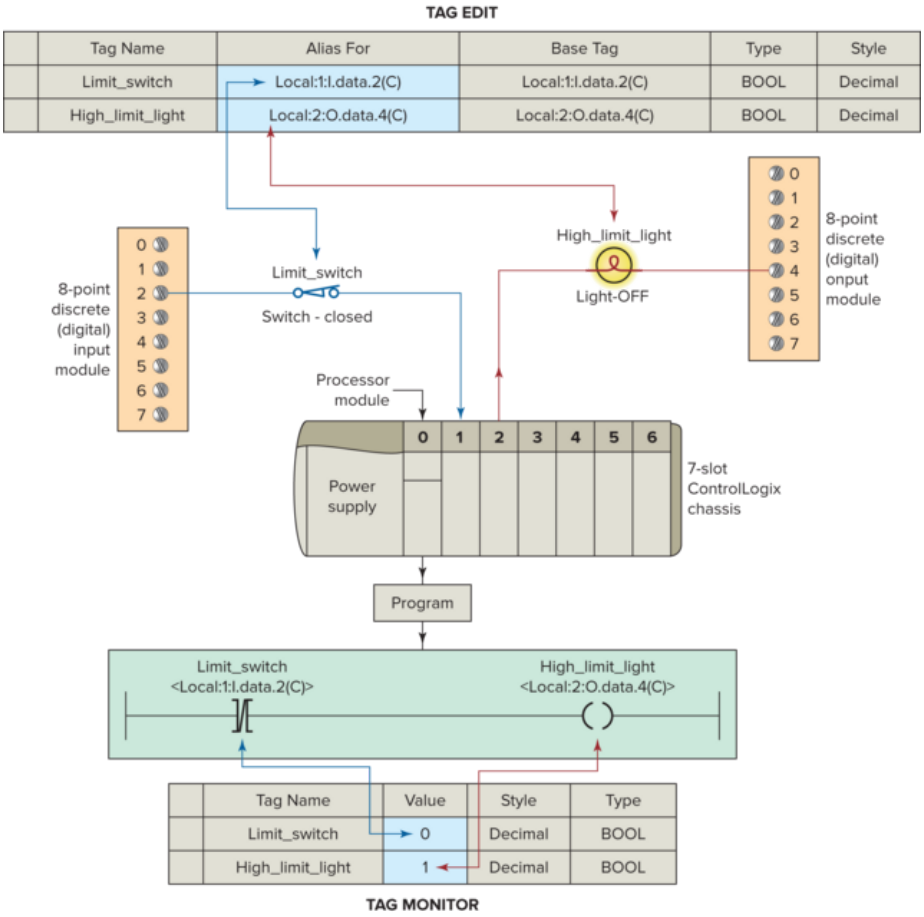


Figure 15-31 Tag-based address implementation.

One advantage of the use of tag-based addressing is that the allocation of variable names for program values is not tied to specific memory locations in the memory structure, as is the case with rack/slot and rack/group type systems. Initially, all program development can proceed with just the tag names and data types assigned. Using tag aliases, programmers can write code independent of electrical connection assignments. At a later date, input and output field devices are easily matched to the pin numbers on the respective module they are connected to.

Adding Ladder Logic to the Main Routine

Figure 15-32 shows the diagram for a hardwired contactor operated motor start/stop control circuit. The normally open start button is momentarily closed to energize the contactor coil and close its main

contacts to start the motor. The seal-in auxiliary contact of the contactor is connected in parallel with the start button to keep the starter coil energized when the start button is released. The normally closed stop button is momentarily opened to de-energize the contactor coil and stop the motor.

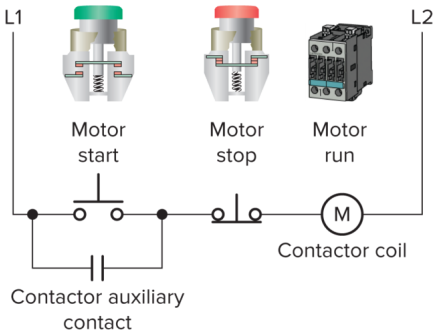


Figure 15-32 Hardwired motor start/stop control circuit.

Figure 15-33 shows the ladder logic program for the motor start/stop control circuit and the RSLogix 5000 toolbar used to create it. Free form editing found in RSLogix 5000 helps speed development in that you do not have to assign addresses to instructions before adding more instructions. In this example we have chosen to use question marks [?] in place of tag names and assign the tags later. Field device wiring for the two pushbutton inputs and the single contactor coil output are as illustrated. The stop button is connected to terminal 4 and the start button to terminal 3 of the DC input module located in slot 1 of the rack. The contactor coil is connected to terminal 4 of the DC output module located in slot 2 of the rack. Both the start and stop buttons are examined for a closed condition (XIC) because both buttons must be closed to cause the motor starter to operate.

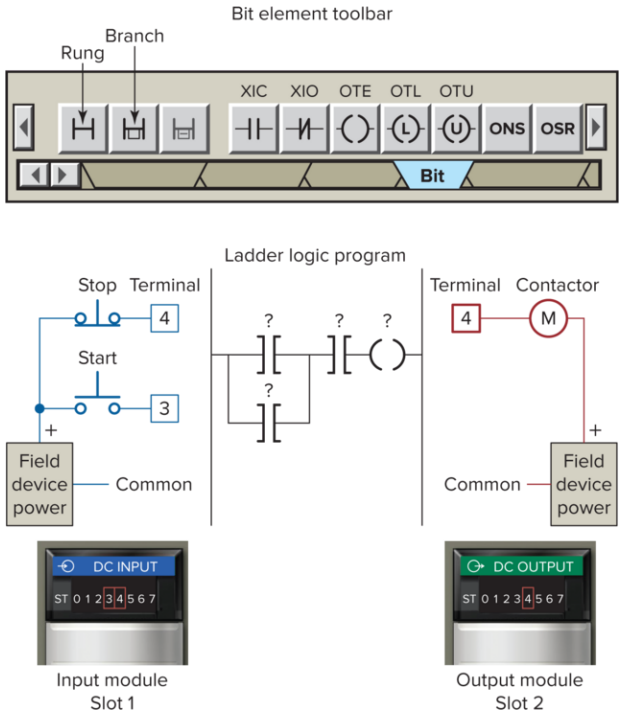


Figure 15-33 Programmed motor start/stop control circuit.

With text-based Logix systems you can use the name of the tag to document your ladder code and organize your data to mirror your application. For the programmed motor start/stop control circuit three tags, Motor_Start, Motor_Stop, and Motor_Run, are created. **Figure 15-34** illustrates how the Motor_Start tag is created in the New Tag window. This window can be accessed by right clicking the ? mark above the XIC instruction in the ladder logic program. Since this tag represents a value from an input field device, a link through the module to the field device must be created. When Local:1:I.Data is selected a dialog box for all of the terminal numbers on the input module appears. The tag name (Motor_Start) used in the program is then linked to input terminal number 3 where the field device represented by the tag name is connected.

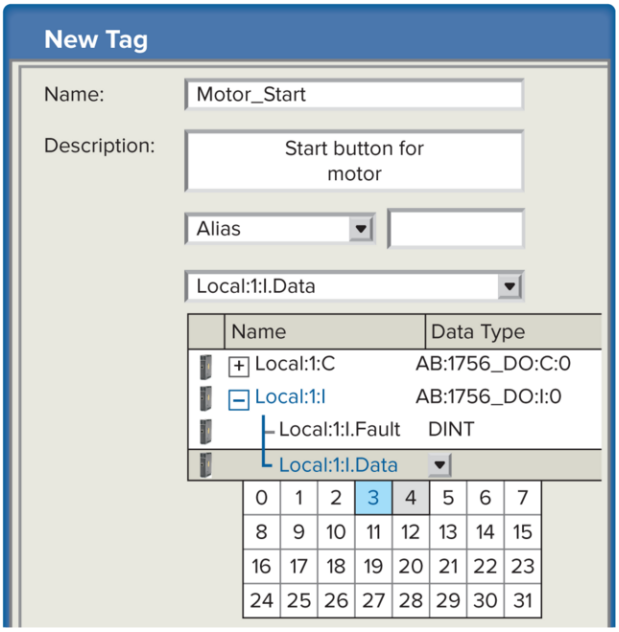


Figure 15-34 Creating the Motor_Start tag.

Source: Image Courtesy of Rockwell Automation, Inc.

Figure 15-35 shows what the ladder logic program would look like after all three tags have been created. Users have the ability to reference data via multiple names using Aliases. This allows the flexibility to name data differently depending on their use. The tag description provides for a more meaningful description of the tag name. Tag names are downloaded and stored in the controller but the description is not, as it is part of the documentation of the project.

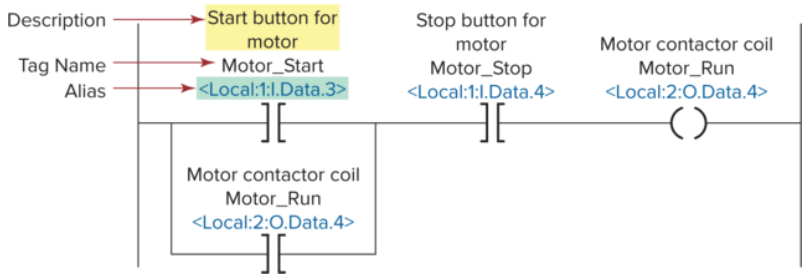
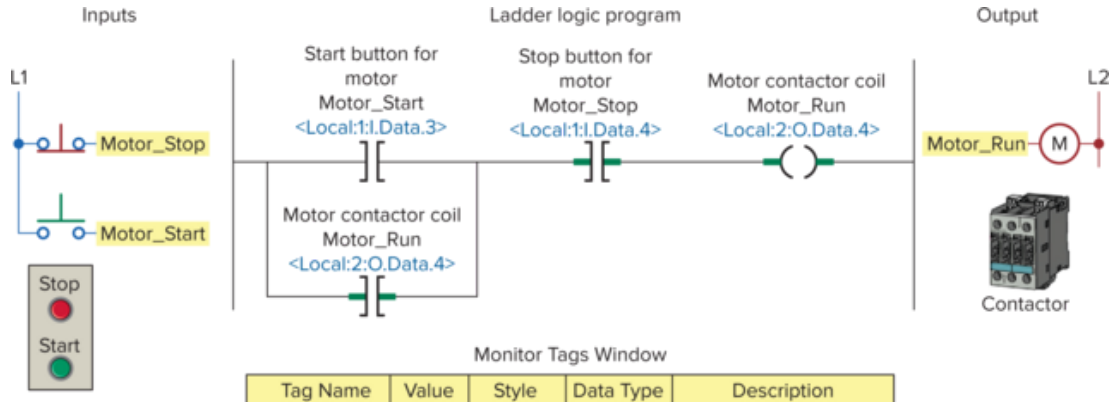


Figure 15-35 Ladder logic program after all tags have been created.

Figure 15-36 shows the state of the tags created for the motor start/stop program as seen in the program and Monitor Tags window, when the motor is operating. When the motor is operating:

- The XIC Motor_Start instruction is false because the NO start button is open; therefore its value is 0.
- The XIC Motor_Stop instruction is true because the NC stop button is closed; therefore its value is 1.
- The OTE Motor_Run instruction is true because the rung has logic continuity; therefore its value is 1.



Monitor Tags Window

Tag Name	Value	Style	Data Type	Description
Motor_Start	0	Decimal	BOOL	Start button for motor
Motor_Stop	1	Decimal	BOOL	Stop button for motor
Motor_Run	1	Decimal	BOOL	Motor contactor coil

Figure 15-36 Ladder logic program and Monitor Tags window with motor operating.

A hardwired reversing conveyor motor control circuit is shown in **Figure 15-37**. The operation of the circuit can be summarized as follows:

- The normally closed contact controlled by the forward contactor coil is connected in series with the reverse coil.
- The normally closed contact controlled by the reverse contactor coil is connected in series with the forward coil.
- When the forward coil is energized, the normally closed contact in series with the reverse coil is opened to prevent the reverse coil from being energized.
- When the reverse coil is energized, the normally closed contact in series with the forward coil opens to prevent the forward coil from being energized.
- To reverse the motor with this control circuit, the operator must first press the stop button to de-energize the respective coil, reclosing the respective normally closed contact. The operator must then press the Reverse pushbutton to energize the reverse coil.
- **Figure 15-38** shows the reversing starter circuit implemented using an RSLogix 5000 controller.

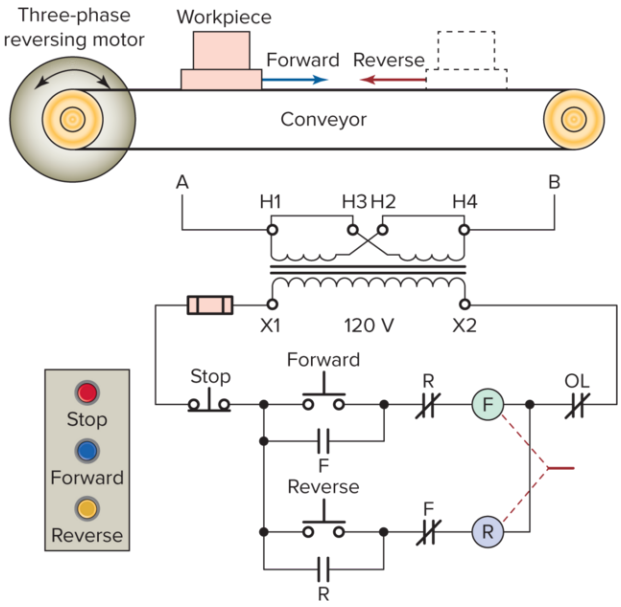


Figure 15-37 Hardwired reversing conveyor motor control circuit.

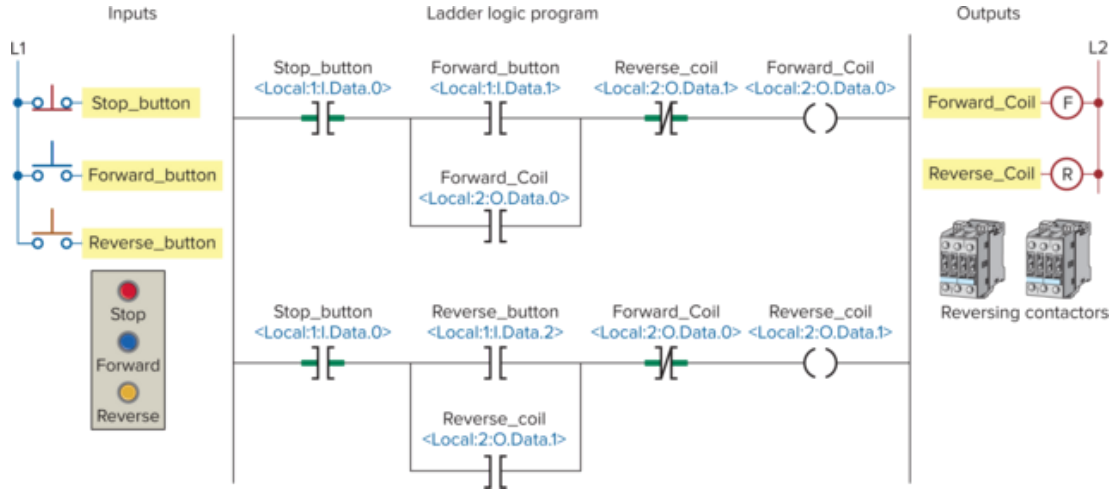


Figure 15-38 Programmed reversing conveyor motor control circuit.

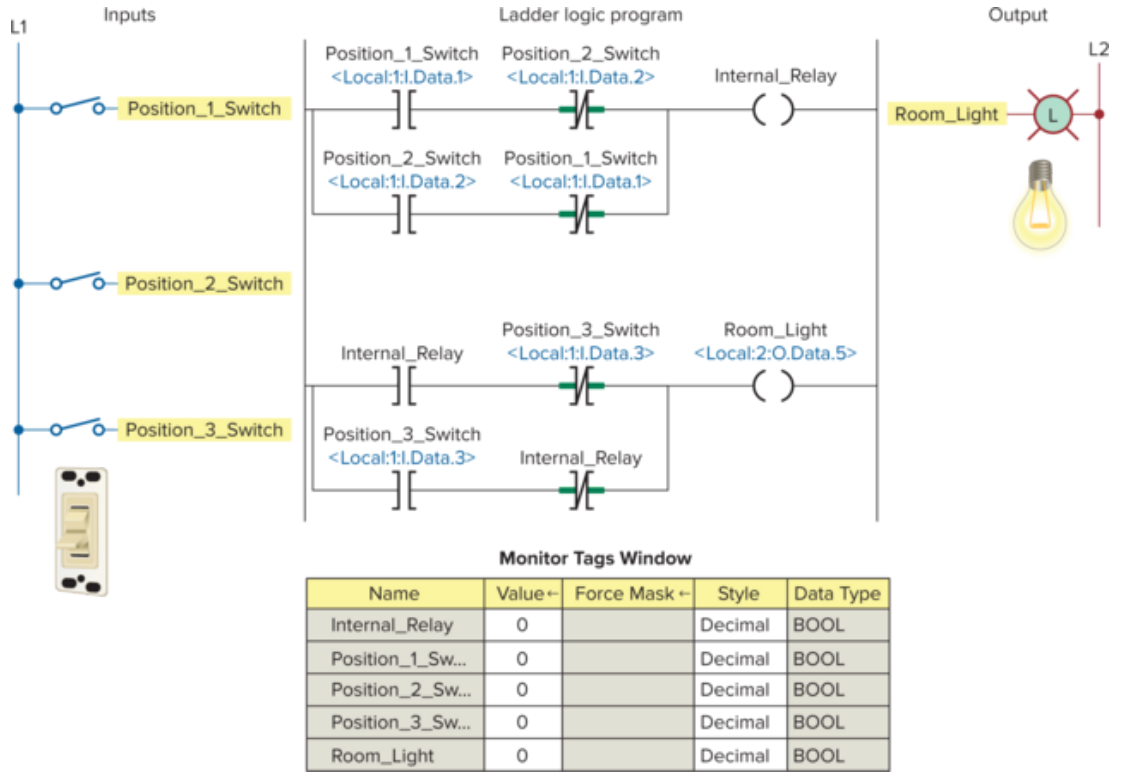
Internal Relay Instructions

Internal relay instructions are used when other than real-world field devices are needed as input or output reference instructions. For example, an internal relay bit is used as an output when the logical resultant of a rung is used to control other internal logic. An internal control relay is programmed in the ControlLogix system by creating a tag (either program or controller type) and assigning a Boolean type to the tag.

Figure 15-39 shows a ControlLogix program that uses an internal relay to implement on/off control of a room light from three different entrances or positions. Three single pole switches are used for inputs in place of the two 3-way and one 4-way switches normally required for an equivalent hardwired control circuit. The operation of the program can be summarized as follows:

- An internal relay is used to execute the logic of the circuit without having to use a real-world output.
- The status value stored in memory for all tags, when all input switches are open, is 0 and so the room light will be off.
- Closing Position_1_Switch changes the status of its XIC instruction from false to true thereby establishing logic continuity for Rung 1.

- As a result, the status of the internal relay coil and its XIC contact change from false to true.
- This establishes logic continuity for Rung 2 and switches the room light on.
- A change in the state of any of input switches will change the current state of the light.



Monitor Tags Window

Name	Value	Force Mask	Style	Data Type
Internal_Relay	0		Decimal	BOOL
Position_1_Sw...	0		Decimal	BOOL
Position_2_Sw...	0		Decimal	BOOL
Position_3_Sw...	0		Decimal	BOOL
Room_Light	0		Decimal	BOOL

Figure 15-39 Internal relay to implement on/off control of a room light from three different entrances.

Internal relay output coils are used in ladder logic programs, but are not accessible at I/O racks. They are used where an output instruction is required in a program, but no physical connection to a real world device is needed. A hardwired pilot light motor control circuit is shown in Figure 15-40. The operation of the circuit can be summarized as follows:

- The Stop/Start pushbutton station controls relay coil CR.
- When CR is de-energized, the green standby pilot light is ON, the red run pilot is OFF, and the motor is not operating.

- When CR is energized, the red run pilot light turns ON, the green standby pilot light turns OFF, and motor M starts operating.

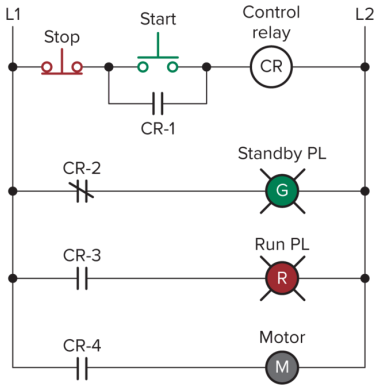


Figure 15-40 Hardwired pilot light motor control circuit

Figure 15-41 shows the pilot light motor control circuit implemented using an RSLogix controller. The control relay CR is represented by tag Internal_Relay_CR. Note that Internal_Relay_CR doesn't have an address and as such is not accessible at the I/O rack.

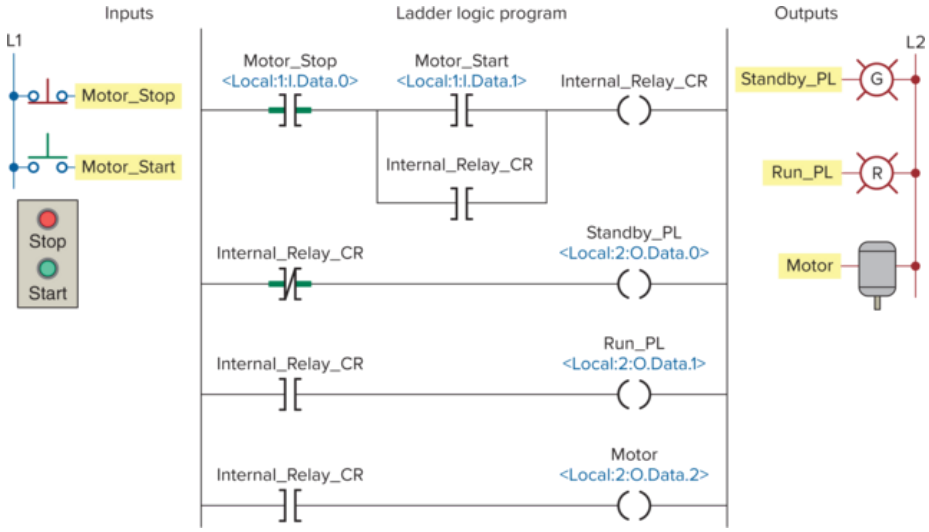


Figure 15-41 Pilot light motor control circuit implemented using an RSLogix controller.

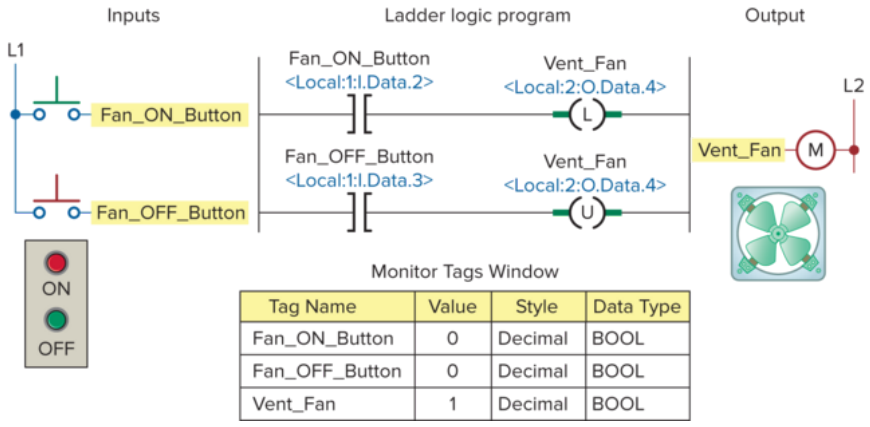
Latch and Unlatch Instructions

The ***output latch (OTL)*** instruction is a retentive output instruction that is used to maintain, or latch, an output. If this output is turned on, it will stay on even if the status of the input logic that caused the output to energize becomes false. The OTL instruction will remain in a latched on condition until an **unlatch instruction (OTU)** with the same referenced tag is energized. The OTL instruction is often used

in programs where the value of a variable must be maintained in instances where there is a shutdown due to a power failure or system fault. Retentive memory permits the system to be restarted with memory locations holding the values that were present when the program execution was halted.

Figure 15-42 shows a ControlLogix program that uses an output latch and unlatch instruction pair to implement the control of a vent fan motor. The operation of the program can be summarized as follows:

- The OTL instruction will write a 1 to its address when true.
- When the OTL goes false, the output address will remain a 1.
- This is true even if the processor powers down and then back up.
- The output address will remain a 1 until reset to 0 by the unlatch instruction.
- If the output address is off, both the latch and unlatch instructions are not intensified, but once the bit is turned on, you will see both the latch and unlatch intensified even though both inputs are shut off.



Tag Name	Value	Style	Data Type
Fan_ON_Button	0	Decimal	BOOL
Fan_OFF_Button	0	Decimal	BOOL
Vent_Fan	1	Decimal	BOOL

Figure 15-42 Output latch and unlatch instructions used to control a vent fan motor.

Figure 15-43 shows an application of the latch/unlatch instruction used as part of a car wash program. The car wash uses a conveyor chain that pulls cars along a track. The operation of the program can be summarized as follows:

- The car wash only washes one car at a time.
- The operator controls the ON/OFF operation of the track motor with the track motor Stop and Start pushbuttons.
- When the Motor_Start button is closed the Track_Motor is energized and performs a seal-in function that keeps the motor operating when the Motor_Start button is released.
- As the car enters the car wash, the Enter_LS contact momentarily closes to energize the Wash_In_Use latch instruction.
- This, in turn, energizes the Do_Not_Enter_Sign to indicate that the car wash is in use.
- When the car exits the car wash, the Exit_LS contact momentarily closes and energizes the Wash_In_Use unlatch instruction.
- This, in turn, de-energizes the Do_Not_Enter_Sign to indicate the car wash is not in use.
- The track motor remains running ready to restart the process once another car enters.

- Once running, the track motor can be stopped at any time by operating the Motor_Stop button to de-energize the Track_Motor.

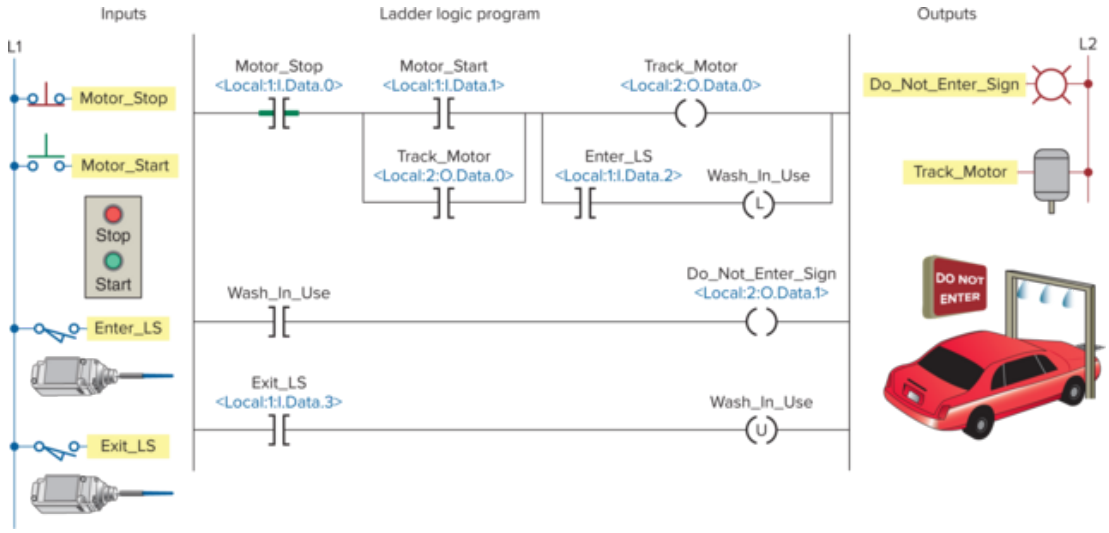


Figure 15-43 Latch/unlatch instruction used as part of a car wash program.

The dual compressor application shown in **Figure 15-44** is designed to alternate the operation of two compressors in order to allow for an extended cooling period between cycles. Each compressor is to take its turn in maintaining pressure in the storage tank. The task is implemented using the latch and unlatch coil instructions. Its program operation is summarized as follows:

- With the program in the initiating state shown, momentarily closing the Start_PB causes the Run_PL rung to become true switching the pilot light on and sealing in the rung true.
- This in turn causes the Motor_M1 rung to become true which in turn:
  - switches Motor_M1 on
  - seals the rung true
  - switches Motor_M1_PL on
  - makes the Internal_Output Latch instruction true which in turn toggles the true/false state of the pair of Internal_Relay instructions.
- Momentarily opening the Stop_PB causes the Run_PL rung to become false, switching Motor_M1 off.
- Momentarily closing the start button for the second time mimics the operation with the Motor_M2 rung and the Output_Unlatch instruction becoming true.

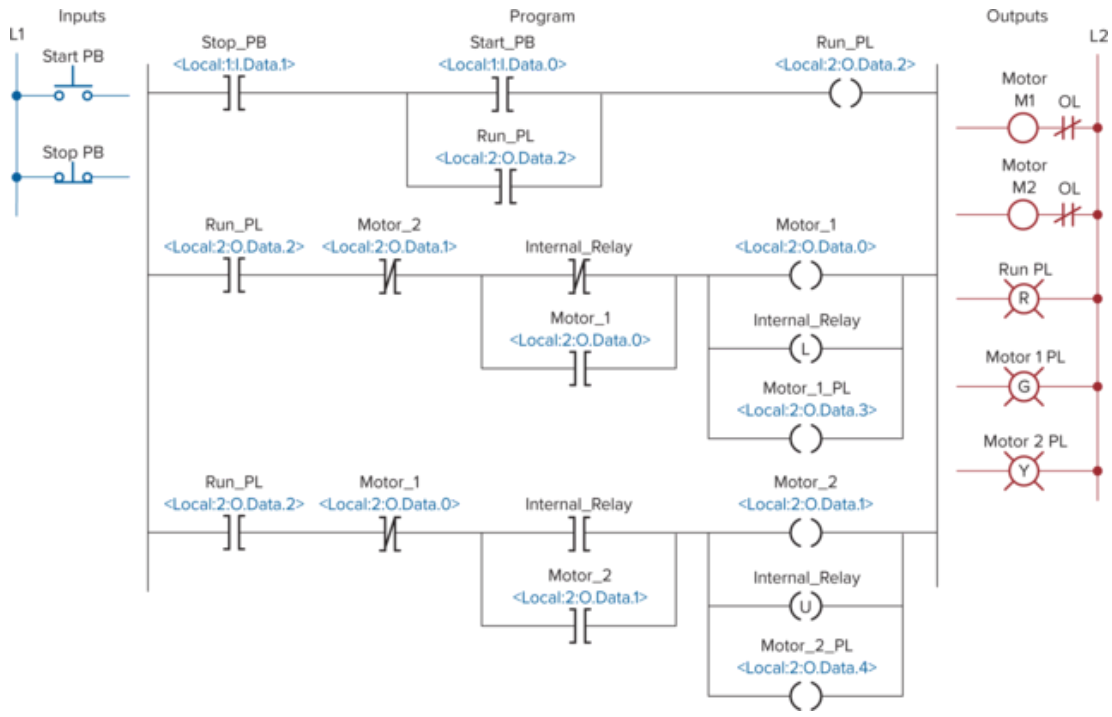


Figure 15-44 Alternating dual compressor program.

One-Shot Instruction

The CLX ***One-Shot (ONS)*** instruction is an input instruction used to turn an output on for one program scan only. The program of **Figure 15-45** uses the ONS instruction with a math instruction to perform a calculation once per closing of the switch. This program is used to execute the ADD math function only once per actuation of the limit switch, no matter how long the limit switch is held closed. The operation of the program can be summarized as follows:

- On any scan for which *limit_switch_1* is cleared or *storage_1* is set, this rung has no effect. Page 346
- On any scan for which *limit_switch_1* is set and *storage_1* is cleared, the ONS instruction sets *storage_1* and the ADD instruction increments *sum* by 1.
- As long as *limit_switch_1* stays set, *sum* stays the same value. The *limit_switch_1* must go from cleared to set again for *sum* to be incremented again.

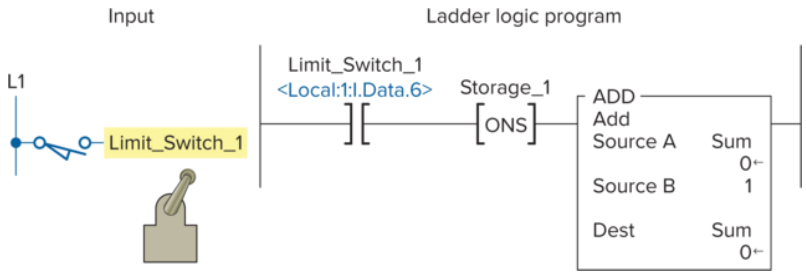


Figure 15-45 ONS instruction used to perform a calculation once per scan.

The **One-Shot Rising (OSR)** instruction shown in Figure 15-46 is an output instruction which produces an output for one program scan when the input rises from zero to one (leading edge of the input pulse). The OSR instruction sets or clears the output bit, depending on the status of the Storage Bit (SB). When enabled and the storage bit is cleared, the OSR instruction sets the output bit. When enabled and the storage bit is set or when disabled, the OSR instruction clears the output bit.

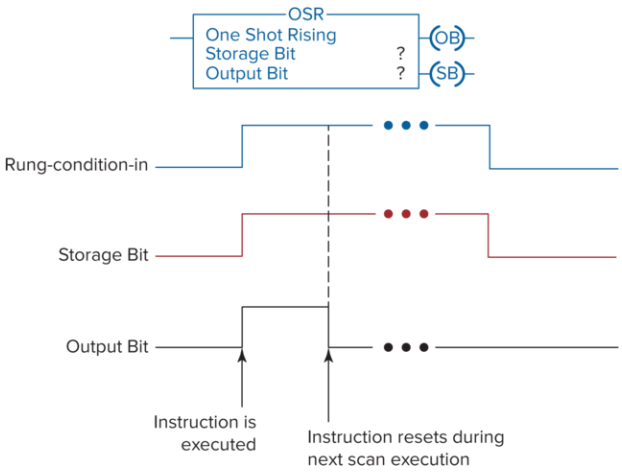


Figure 15-46 One-Shot Rising (OSR) instruction.

The **One-Shot Falling (OSF)** instruction shown in Figure 15-47 is an output instruction which produces an output for one program scan when the input drops from one to zero (falling edge of the input pulse). The OSF instruction sets or clears the output bit depending on the status of the storage bit. When disabled and the storage bit is set, the OSF instruction sets the output bit. When disabled and the storage bit is cleared, or when enabled, the OSF instruction clears the output bit.

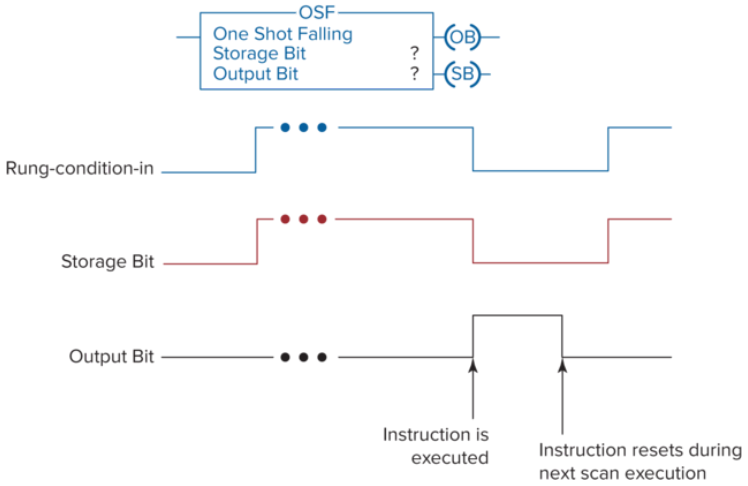


Figure 15-47 One-Shot Falling (OSF) instruction.

One-shot instructions are very useful for math operations. The program of **Figure 15-48** shows an example of a program that makes use of all three one-shot instructions. The function of the program is to add 4, 8, or 12 to a register each time the associated input sensor detects a product. The instructions evaluate the preceding conditions, and when either the ONS, OSR, or OSF input changes state, a shot is triggered. The Clear Sums input pushbutton is used to clear all three instructions back to zero.

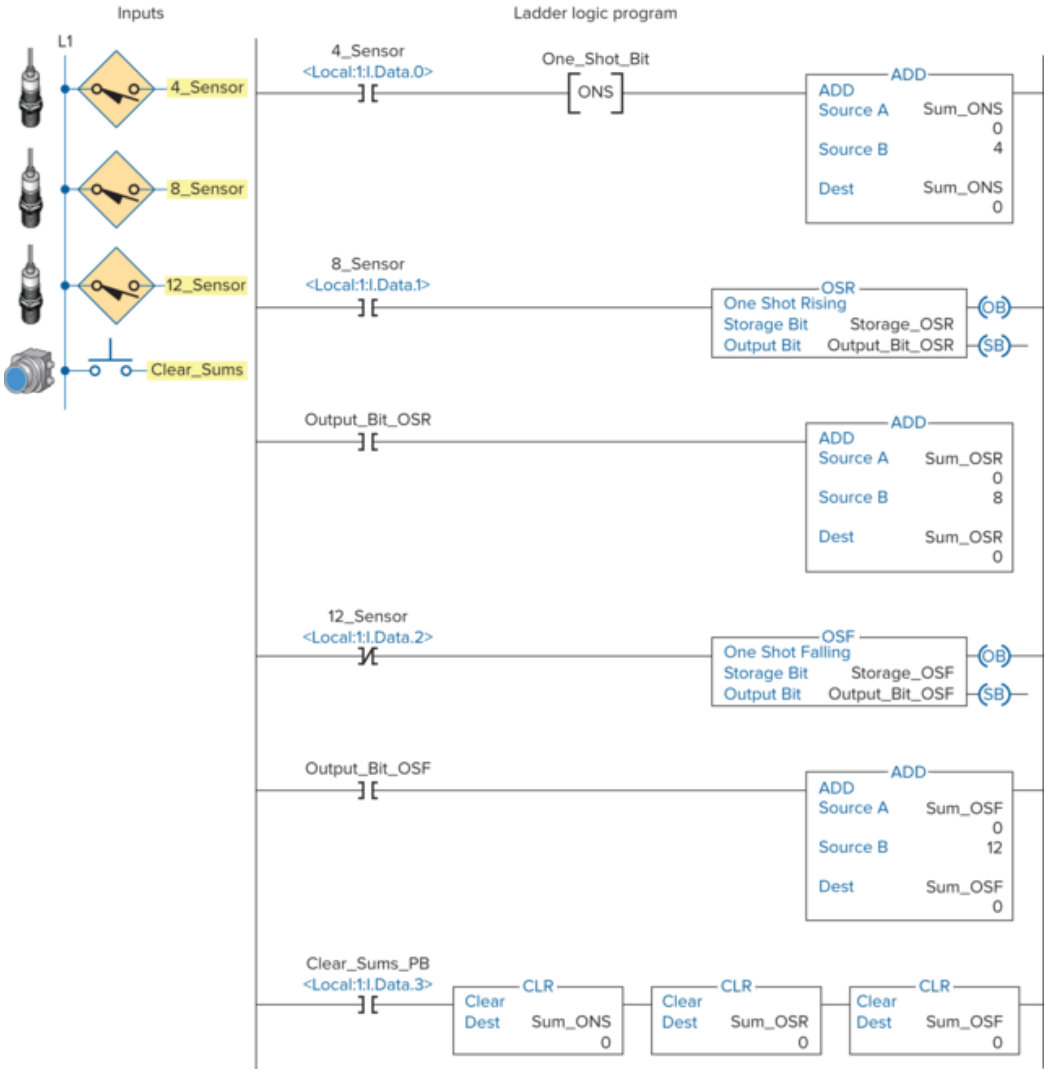
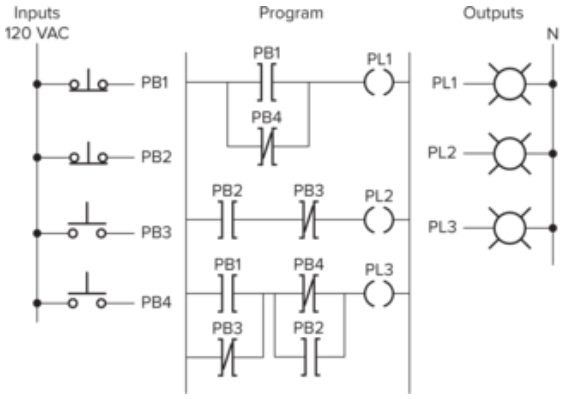


Figure 15-48 One-shot instructions used in conjunction with math operations.

PART 2 REVIEW QUESTIONS

1. What operations are performed by the processor during the program scan?
2. With a ControlLogix processor, I/O updates occur asynchronously. Explain what this means.
3. In ladder logic programming into what two broad categories can instruction types be classified?
4. A field input switch is examined using an XIC instruction.
   a. What is the value (0 or 1) stored in its memory bit when the switch is opened and closed?
   b. What is the state of the instruction (true or false) when the switch is opened and closed?
5. A field input switch is examined using an XIO instruction.
   a. What is the value (0 or 1) stored in its memory bit when the switch is opened and closed?
   b. What is the state of the instruction (true or false) when the switch is opened and closed?
6. The value of an OTE instruction as it appears in the Monitor Tags window is 1. Explain what this means as far as the status of a real-world field output and programmed XIC and XIO instructions associated with this tag are concerned.
7. Define a tag in the ControlLogix system.
8. What advantage do tag-based addressing systems have over rack/slot types?
9. How is an internal relay programmed in the ControlLogix system?
10. The output latch instruction is a retentive output instruction. Explain what retentive means.
11. The ControlLogix ONS instruction is a one-shot instruction. Explain what this means.
12. Complete the Truth Table for the different states of the PLC program input devices.



Truth table

Input device state	Input state				Output state		
	PB4	PB3	PB2	PB1	PL3	PL2	PL1
No PBs pressed							
PB1 pressed							
PB3 pressed							
PB2,3,4 pressed							

Figure 15-49 PLC program and Truth Table for Review question 12.



1. Modify the original ControlLogix start/stop motor control program (Figure 15-35) with a second start and stop button added to the program. The additional start button is to be connected to pin 1 and the stop button to pin 2 of the digital input module.
2. Extend control of the original ControlLogix internal relay program (Figure 15-39) used to control a room light from 3 entrances to 4. The additional single-pole switch is to be connected to pin 4 of the digital input module.
3. Implement the hardwired latching relay alarm circuit of Figure 15-50 in Logix format. The alarm will be latched on anytime:
   - The normally open temperature switch closes.
   - Both normally open float switches 1 and 2 close.
   - Either normally open sensor switch 1 or 2 closes while the normally closed pressure switch is closed.

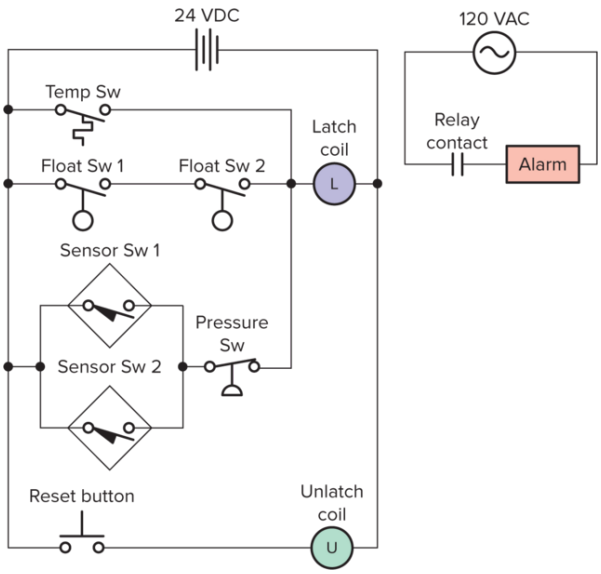


Figure 15-50 Hardwired latching relay alarm circuit for Problem 3.

4. Implement the hardwired tank filling and emptying operation shown in Figure 15-51 in Logix format. The operation of the control circuit can be summarized as follows:
   - Assuming the liquid level of the tank is at or below the empty level mark, momentarily pressing the FILL pushbutton will energize control relay 1CR.

- Contacts $1CR_1$ and $1CR_2$ will both close to seal in the 1CR coil and energize normally closed solenoid valve A to start filling the tank.
- As the tank fills, the normally open empty-level sensor switch closes.
- When the liquid reaches the full level, the normally closed full-level sensor switch opens to open the circuit to the 1CR relay coil and switch solenoid valve A to its de-energized closed state.
- Anytime the liquid level of the tank is above the empty-level mark, momentarily pressing the EMPTY pushbutton will energize control relay 2CR.
- Contacts $2CR_1$ and $2CR_2$ will both close to seal in the 2CR coil and energize normally closed solenoid valve B to start emptying the tank.
- When the liquid reaches the empty level, the normally open empty-level sensor switch opens to open the circuit to the 2CR relay coil and switch solenoid valve B to its de-energized closed state.
- The stop button may be pressed at any time to halt the process.

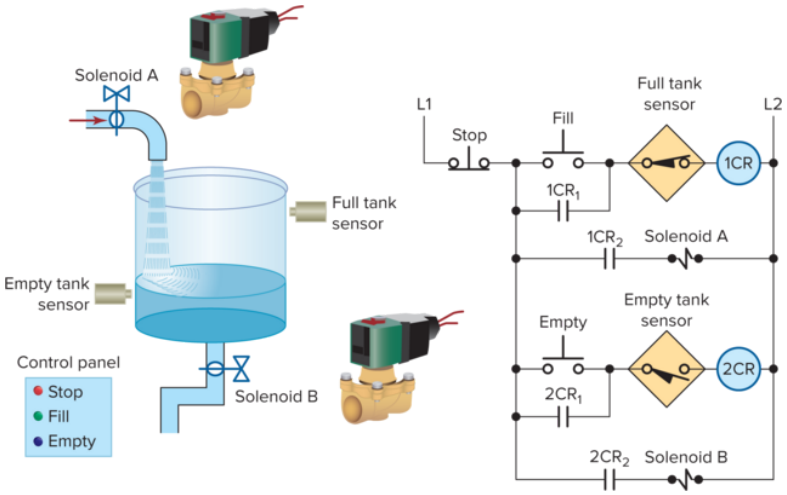


Figure 15-51 Hardwired tank filling and emptying operation for Problem 4.

OceanofPDF.com

Part 3 Programming Timers

Part Objectives

After completing this part, you will be able to:

- Understand ControlLogix timer tags and their members
- Utilize status bits from timers in logic
- Develop ladder logic programs using ControlLogix timers

Timer Predefined Structure

Timers are used to turn outputs on and off after a time delay, turn outputs on or off for a set amount of time, and keep track of the time an output is on or off. The timer address in RSLogix 500 is a data table address or symbol, whereas the timer address in Logix 5000 is a tag and uses a predefined structure of the TIMER data type. The **TIMER structure** is shown in Figure 15-52. Timer parameters and status bits include:

- **Tag Name**—User-friendly tag name for the timer (e.g., Pump_Timer). If you want to use a timer, you must create a tag of type timer.
- **Preset (PRE)**—The number of time increments that the timer must accumulate to reach the desired time delay. Specifies the value (in milliseconds) which the timer must reach before the done bit (DN) changes state. The preset value is stored as a binary number (DINT). The time base is always 1 msec. For example, for a 3 second timer, enter 3000 for the PRE value.
- **Accumulator (ACC)**—The accumulator value is the number of milliseconds the instruction has been enabled. The accumulator value stops changing when $\text{ACC value} = \text{PRE value}$.
- **Enable Bit (EN)**—The enable bit indicates the timing instruction is enabled. The EN bit is true when the rung input logic is true, and false when the rung input logic is false.
- **Timer Timing Bit (TT)**—The timing bit indicates that a timing operation is in process. The TT bit is true only when the accumulator is incrementing. TT remains true until the accumulator reaches the preset value.
- **Done Bit (DN)**—The done bit indicates that accumulated value (ACC) is equal to the preset (PRE) value. The DN bit signals the end of the timing process by changing states from false-to-true or from true-to-false depending on the type of TIMER instruction used. The DN bit is the most commonly used timer status bit.

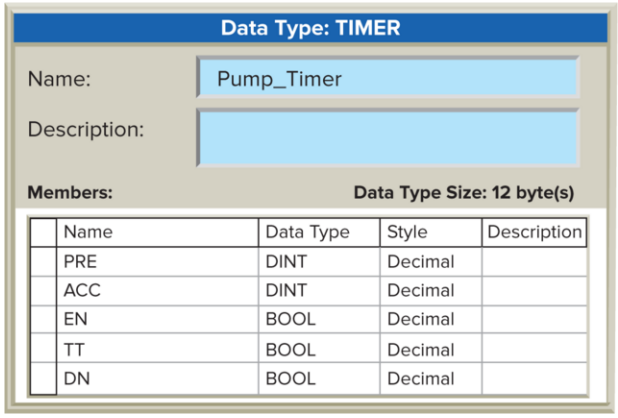


Figure 15-52 TIMER predefined structure.

Source: Image Courtesy of Rockwell Automation, Inc.

On-Delay Timer (TON)

The ***on-delay timer (TON)*** is a nonretentive output instruction used when the application requires an action to occur at some time after the rung conditions for the timer become true. The ControlLogix TON on-delay instruction and timer selection toolbar are shown in **Figure 15-53**. When you want to use a timer, you must create a tag of type TIMER (it is a predefined data type) and enter the preset and the accumulated value. The tag must be defined before the preset and accumulated values can be entered. A value can be entered for the accumulator while programming. When the program is downloaded this value will be in the timer for the first scan. If the TON timer is not enabled the value will be set back to zero. Normally zero will be entered for the accumulator value.

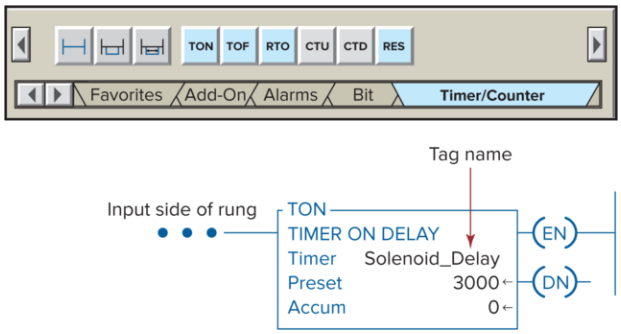


Figure 15-53 TON on-delay instruction.

The timer tag name is declared using the new tag properties dialog box shown in **Figure 15-54**. Tag name, description (optional), tag type, data type, and scope are selected or typed to complete the validation. A descriptive tag name, such as Solenoid_Delay, makes it easier to know what function the timer serves in the control system.

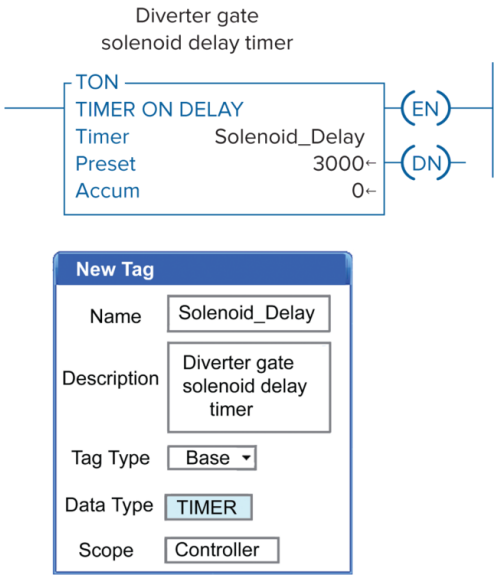
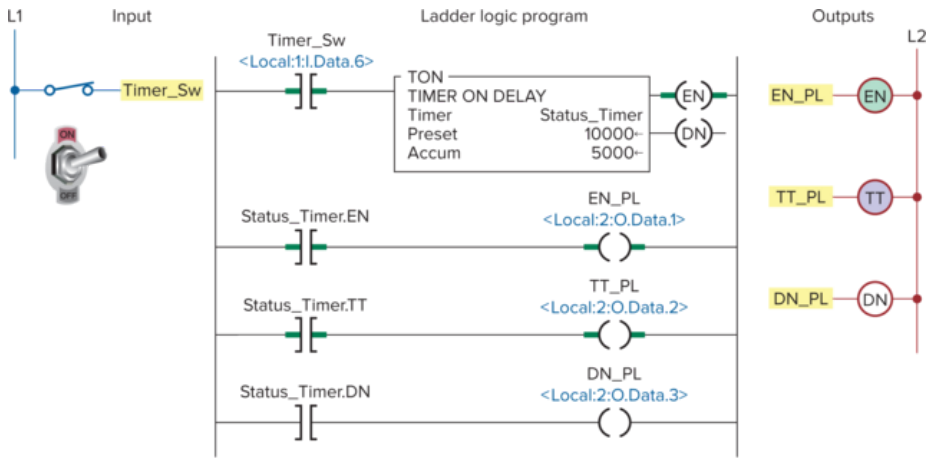


Figure 15-54 Timer tag validation.

The program of **Figure 15-55** is an example of a 10000 ms (10 s) TON timer. Timers generate both word level (DINT) and bit level (BOOL) data and status. The operation of the program can be summarized with reference to the Monitor Tags window.

- The status of all instruction is shown after the timer input switch has been switched from off to on (1) and accumulated 5000 ms (5 s) of time.
- At this halfway point the EN bit is 1 since the rung is true, the TT bit is 1 since the accumulated value is changing, and the DN bit is 0 since the accumulated value does not yet equal the preset value.
- When the ACC equals PRE, the accumulated value stops incrementing, EN stays on for as long as the rung remains true, TT equals 0 since the accumulated value is not changing, and DN equals 1 since $\mathrm{ACC} = \mathrm{PRE}$.
- This will result in the DN pilot light switching on at the same time as the TT pilot light switches off.
- The EN pilot light remains on as long as the input switch is closed.
- Opening the input switch at any time causes the TON instruction to go false, resetting the counter ACC value to 0 and EN, TT, and DN bits to 0. This in turn switches off all output pilot lights.
- The TON instruction is a self-resetting timer. When the rung goes false, the timer is automatically reset. A reset instruction can be used, but usually is not.

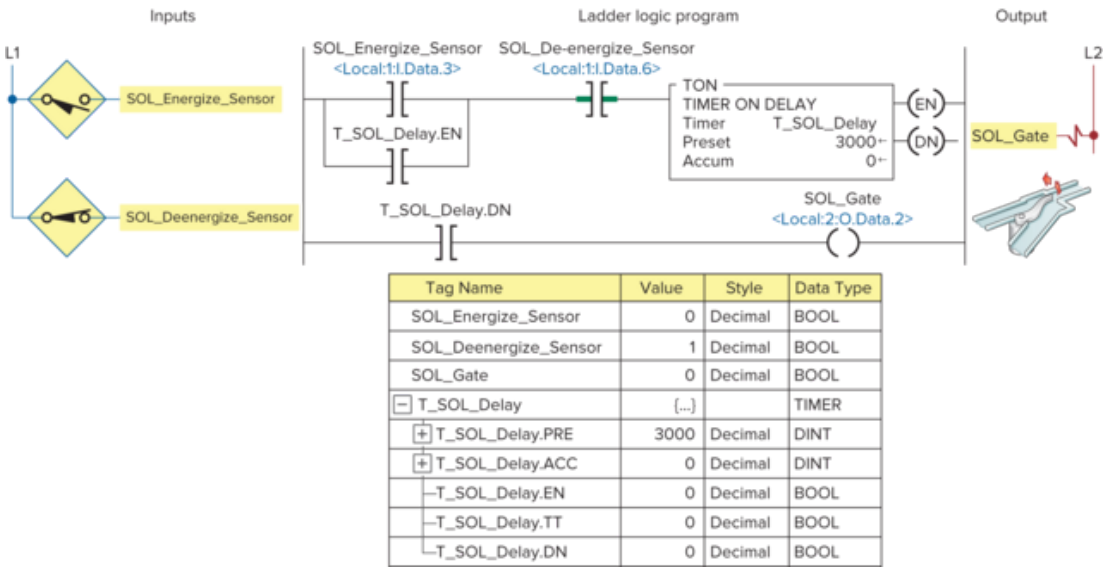


Tag Name	Value	Style	Data Type
Timer_Sw	1	Decimal	BOOL
EN_PL	1	Decimal	BOOL
TT_PL	1	Decimal	BOOL
DN_PL	0	Decimal	BOOL
Status_Timer	{...}		TIMER
Status_Timer.**PRE**	10000	Decimal	DINT
Status_Timer.**ACC**	500	Decimal	DINT
Status_Timer.**EN**	1	Decimal	BOOL
Status_Timer.**TT**	1	Decimal	BOOL
Status_Timer.**DN**	0	Decimal	BOOL

Figure 15-55 Ten-second TON timer program.

Figure 15-56 shows a TON timer used to delay the operation of a diverter gate solenoid for 3 seconds after a target has been sensed by the solenoid energize sensor. The operation of the program can be summarized as follows:

- Detection of the target causes closure of the SOL_Energize_Sensor contacts, making the timer rung true and start timing. Page 352
- With passage of the target, the SOL_Energize_Sensor contacts open but the rung remains true through the EN bit of the TON timer.
- After 3000 ms (3 s) delay time has elapsed, delay timer DN bit is set to 1 to energize the SOL_Gate.
- Momentary detection of the target by the SOL_Deenergize_Sensor causes the opening of its contacts and resets the program to its original state. Page 353



Tag Name	Value	Style	Data Type
SOL_Energize_Sensor	0	Decimal	BOOL
SOL_Deenergize_Sensor	1	Decimal	BOOL
SOL_Gate	0	Decimal	BOOL
T_SOL_Delay	{...}		TIMER
T_SOL_Delay.PRE	3000	Decimal	DINT
T_SOL_Delay.ACC	0	Decimal	DINT
T_SOL_Delay.EN	0	Decimal	BOOL
T_SOL_Delay.TT	0	Decimal	BOOL
T_SOL_Delay.DN	0	Decimal	BOOL

Figure 15-56 TON timer used to delay the operation of a diverter gate solenoid.

Figure 15-57 shows a program that uses a TON timer to illuminate a green pilot light for 20 seconds each time a momentary button is pressed. In addition to the TON timer this program uses multiple outputs on one rung, output latch and unlatch instructions, as well as a timer reset instruction. The operation of the program can be summarized as follows:

- Initially closing the Timer_Button sets (latches) the Green_PL on and enables the Pilot_Light_Timer.
- When the button is then opened the timer rung remains true through the logic path created by the Pilot_Light_Timer.EN bit.
- After 20000 ms (20 s) have elapsed the timer DN bit is set to reset the timer to its original state and unlatch the Green_PL and switch it off.

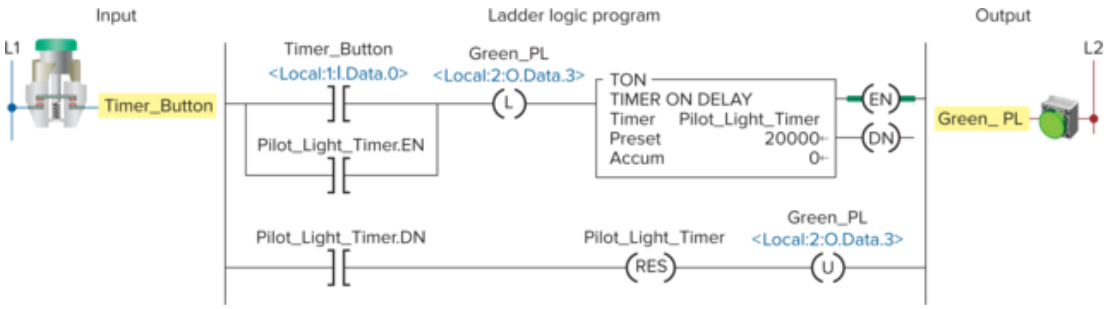


Figure 15-57 Pilot light TON timer.

The ControlLogix program of **Figure 15-58** shows three TON timers cascaded (connected together) for traffic light control. The ladder logic used is the same as that used to program the traffic lights using RSLogix 500. The different tags created to fit the program are shown in **Figure 15-59**. Operation of the program can be summarized as follows:

- Transition from red light to green light to amber light is accomplished by the interconnection of the EN and DN bits of the three TON timer instructions.
- The input to the Red_Light_Timer is controlled by the Amber_Light_Timer.DN bit.
- The input to the Green_Light_Timer is controlled by the Red_Light_Timer.DN bit.
- The input to the Amber_Light_Timer is controlled by the Green_Light_Timer.DN bit.
- The timed sequence of the lights is:
  - Red—30 s on
  - Green—25 s on
  - Amber—5 s on
- The sequence then repeats itself.

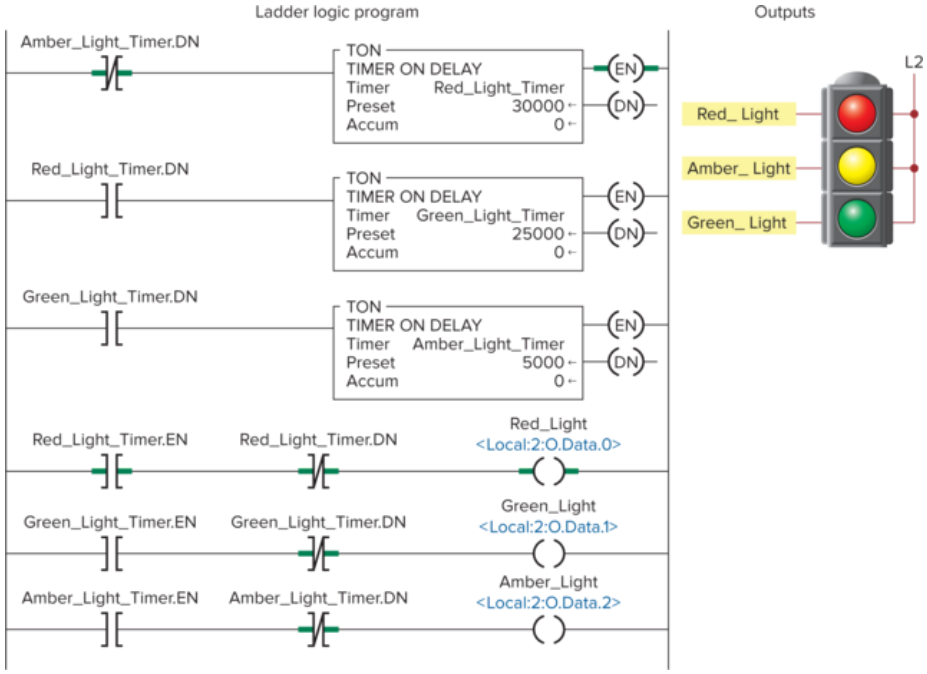


Figure 15-58 ControlLogix traffic control program.

Tag Name	Value	Style	Data Type
⊞-Amber_Light_Timer	{...}		TIMER
⊞-Green_Light_Timer	{...}		TIMER
⊟-Red_Light_Timer	{...}		TIMER
⊞-Red_Light_Timer.PRE	30000	Decimal	DINT
⊞-Red_Light_Timer.ACC	0	Decimal	DINT
—Red_Light_Timer.EN	1	Decimal	BOOL
—Red_Light_Timer.TT	1	Decimal	BOOL
—Red_Light_Timer.DN	0	Decimal	BOOL
Red_Light	1	Decimal	BOOL
Green_Light	0	Decimal	BOOL
Amber_Light	0	Decimal	BOOL

Figure 15-59 Tags created for traffic light program.

Off-Delay Timer (TOF)

The ***off-delay timer (TOF)*** operates in a fashion opposite to the TON on-delay timer. An off-delay timer will turn on immediately when the rung of ladder logic is true, but it will delay before turning off after the rung goes false. The ControlLogix TOF off-delay timer instruction is shown in **Figure 15-60**. The description of the function block fields and tag references are the same as for that of a TON timer.

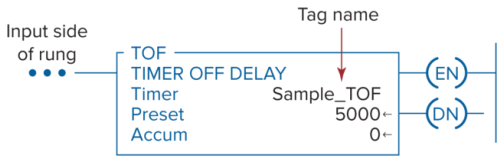


Figure 15-60 ControlLogix TOF off-delay timer instruction.

Figure 15-61 shows a program that uses a TOF timer to illuminate a green pilot light for 20 seconds each time a momentary button is pressed. The program code is simpler than that used to accomplish the same task using a TON timer. The operation of the program can be summarized as follows:

- When the Timer_Button is initially closed the timer rung and instruction and DN bit all become true.
- The DN bit switches on the Green_PL and the program remains in this state as long as the button is held closed.
- When the button is released the Timer_Button instruction goes false and starts the timing cycle.
- The light remains on and the timer begins accumulating time.
- When the accumulator reaches 20000 ms (20 s) the timer DN bit becomes false and the light is switched off.

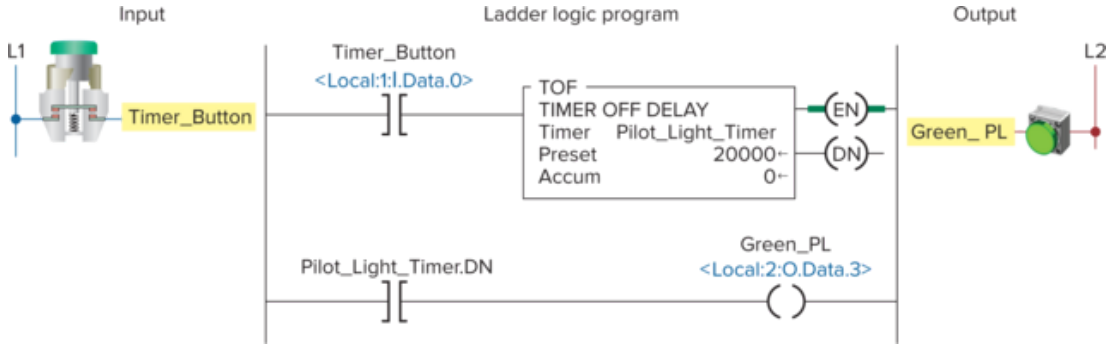


Figure 15-61 Pilot light TOF timer.

The program of **Figure 15-62** uses both on-delay and off-delay timers for control of a heating oven process. The different tags created to fit the program are shown in **Figure 15-63**. Operation of the program can be summarized as follows:

- Pressing the Oven_On_Button energizes the Oven_On_PL output which seals itself in and enables the TON and TOF timer instructions.
- The Timer_Heat.TT bit of the TON timer becomes true which sounds the Warning_Horn to warn that the oven is about to come on.
- The Timer_Cooling.DN bit of the TOF timer becomes true which energizes the Fan_Motor.
- After 10 s (10000 ms) have elapsed the Timer_Heat.TT bit becomes false to turn off the Warning_Horn and the Timer_Heat.DN bit becomes true to energize the Heater_Contactor and turn on the heating coils.
- When the Oven_Off_Button is momentarily actuated the Oven_On_PL output goes false which turns the pilot light off and opens the continuity of its seal-in logic path.
- The Timer_Heat timer instruction and its DN bit instruction become false which de-energizes the Heater_Contactor and turns off the heating coils.
- The Timer_Cooling timer begins accumulating time and the fan continues to operate for the 5 minute (300000 ms) delay period after which the Timer_Cooling.DN bit becomes false to turn the fan off.

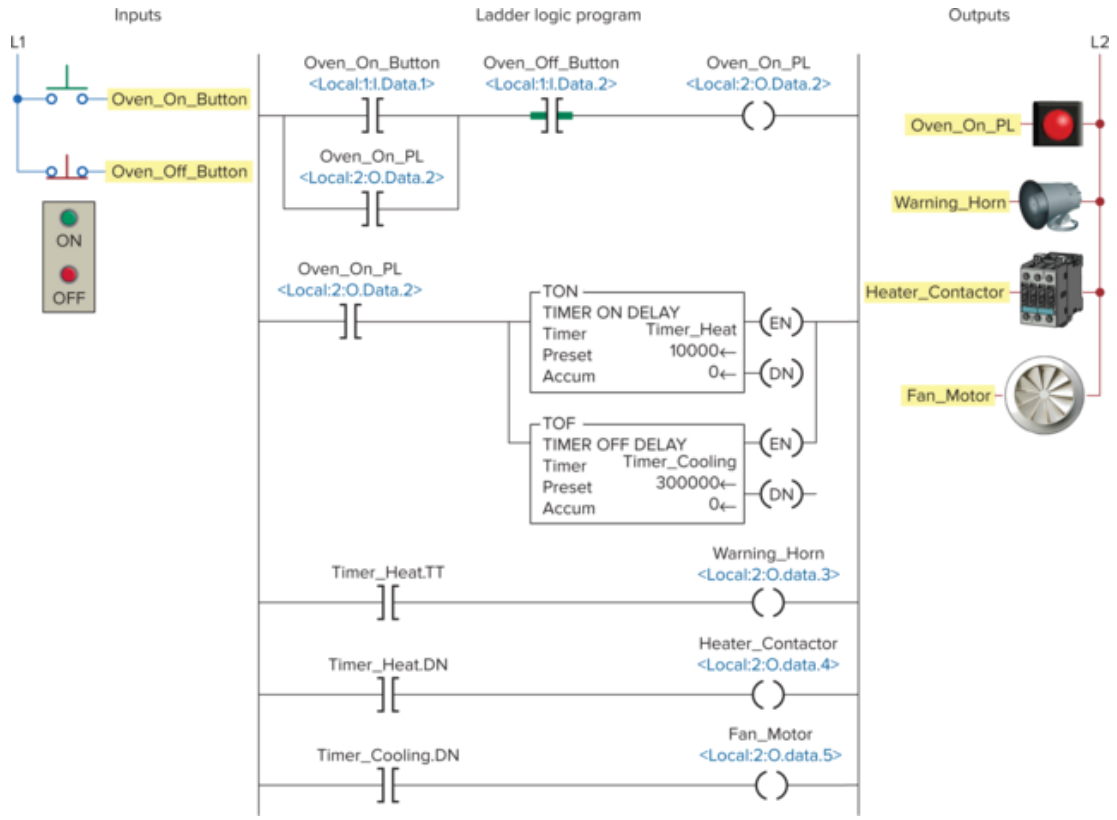


Figure 15-62 Timer control of a heating oven process.

Tag Name	Alias For	Base Tag	Data Type	Style
Warning_Horn	Local:2:O.Data.3	Local:2:O.Data.3	BOOL	Decimal
Heater_Contactor	Local:2:O.Data.4	Local:2:O.Data.4	BOOL	Decimal
Fan_Motor	Local:2:O.Data.5	Local:2:O.Data.5	BOOL	Decimal
Oven_On_PL	Local:2:O.Data.2	Local:2:O.Data.2	BOOL	Decimal
Oven_On_Button	Local:1:I.Data.1	Local:1:I.Data.1	BOOL	Decimal
Oven_Off_Button	Local:1:I.Data.2	Local:1:I.Data.2	BOOL	Decimal
+-Timer_Heat			TIMER	
+-Timer_Cooling			TIMER	

Figure 15-63 Tags created for heating oven process.

Retentive Timer On (RTO)

A ***retentive on-delay timer (RTO)*** operates the same as a TON timer, except that the retentive timer retains (remembers) its ACC value even if:

- The rung goes false.
- The processor is placed in the program mode.
- The processor faults.
- Power to the processor is temporarily interrupted and the processor battery is functioning properly.

The ControlLogix RTO retentive on-delay timer instruction is shown in **Figure 15-64**. The description of the function block fields and tag references are the same as for that of a TON timer; however, a RES reset instruction must be used to reset the accumulated value of a retentive timer. The RES instruction must have the same tag name as the timer you want to reset.

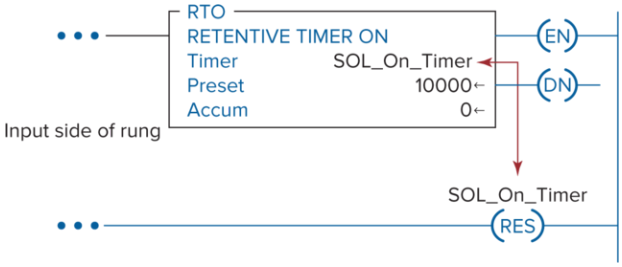


Figure 15-64 RTO retentive on-delay timer instruction.

An example application of a limit switch 2 minute (120000 ms) RTO timer program is shown in **Figure 15-65**. The different tags created to fit the program are shown in **Figure 15-66**. The operation of the program can be summarized as follows:

- The status and value of all instructions, with the timer initially reset, are as shown in the monitor tags window.
- When the Limit_Switch has been closed for 1 minute, the status and value of the instructions would be:

- PRE – 120000
- ACC – 60000
- LS_Timer.EN – 1
- LS_Timer.TT – 1
- LS_Timer.DN – 0
- LS_EN_PL – 1
- LS_TT_PL – 1
- LS_Alarm – 0

- When the Limit_Switch is opened after 1.5 minutes, the status and value of the instructions would be:
  - PRE – 120000
  - ACC – 90000
  - LS_Timer.EN – 0
  - LS_Timer.TT – 0
  - LS_Timer.DN – 0
  - LS_EN_PL – 0
  - LS_TT_PL – 0
  - LS_Alarm – 0
- When the Limit_Switch is closed and stays closed until the timer times out, the status and value of the instructions would be:

  - PRE – 120000
  - ACC –120000
  - LS_Timer.EN – 1
  - LS_Timer.TT – 0
  - LS_Timer.DN – 1
  - LS_EN_PL – 1
  - LS_TT_PL – 0
  - LS_Alarm – 1
- When the Limit_Switch is opened after the timer times out, the status and value of the instructions would be:
  - PRE – 120000
  - ACC –120000
  - LS_Timer.EN – 0
  - LS_Timer.TT – 0
  - LS_Timer.DN – 1
  - LS_EN_PL – 0
  - LS_TT_PL – 0

- LS_Alarm – 1

- When the Reset_LS_Timer is closed, the status and value of the instructions are reset to their original values.

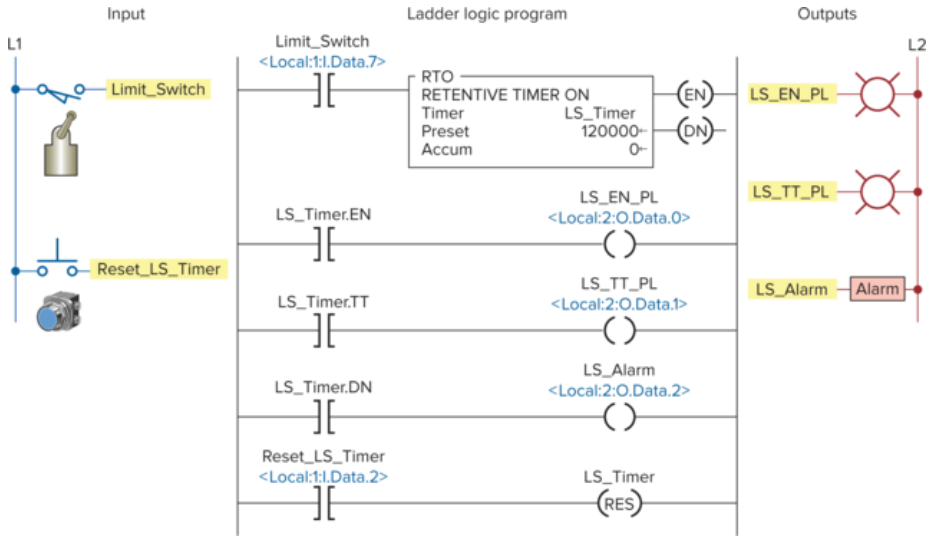


Figure 15-65 Limit switch RTO timer program.

Tag Name	Value	Style	Data Type
-LS_Timer	{...}		TIMER
-LS_Timer.PRE	120000	Decimal	DINT
-LS_Timer.ACC	0	Decimal	DINT
LS_Timer.EN	0	Decimal	BOOL
LS_Timer.TT	0	Decimal	BOOL
LS_Timer.DN	0	Decimal	BOOL
Limit_Switch	0	Decimal	BOOL
LS_EN_PL	0	Decimal	BOOL
LS_TT_PL	0	Decimal	BOOL
LS_Alarm	0	Decimal	BOOL

Figure 15-66 Tags created for the RTO retentive on-delay timer program.

Cascading of Timers

Timers can be linked together or **cascaded** to extend their control capability. The industrial control program of **Figure 15-67** is an example of cascading TON timers for timed event-driven routines. In this program the timers are cascaded in such a manner that one event leads to another. The operation of the program can be summarized as follows:

- The process consists of three distinct timed events or steps.
- Each step must last for a predetermined period of time and must be performed in a specific order.

- The Start_PB, associated internal relay seal-in circuit, is activated to start the sequence.
- Event_1 has a 3 second duration, after which Event_2 begins.
- Event_2 has a 10 second duration, after which Event _3 begins.
- Event_3 has a 5 second duration, after which the entire process is completed.
- The output pilot lights turn on to signify the completion of each step in the sequence.
- Once the final step has been completed, the Process_Complete_PL is turned on.
- Activating the Stop_PB at any time will reset the process.

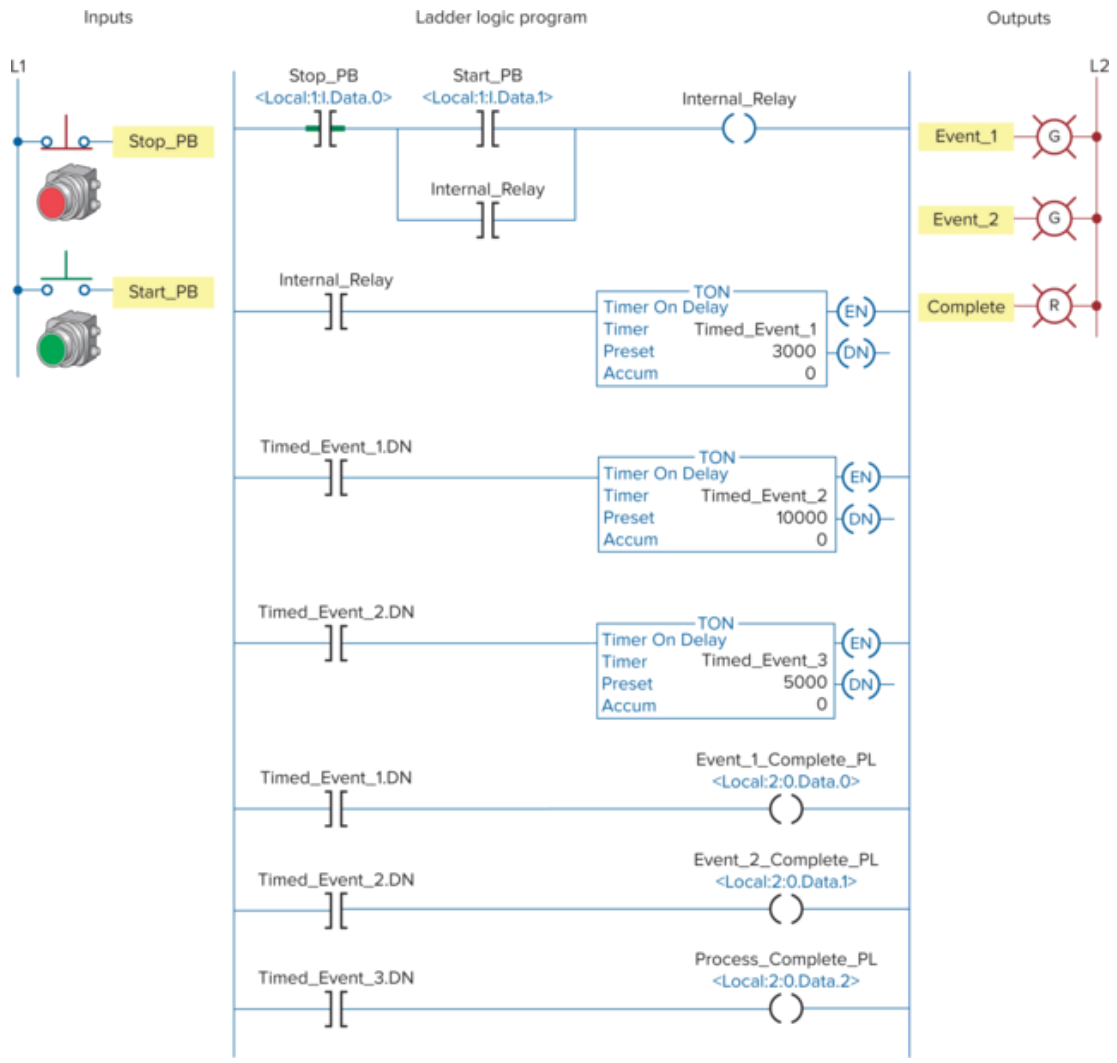


Figure 15-67 Cascading TON timers for timed event-driven routines.

Reciprocating timers are timing functions where the output of one timer is used to reset the input of a second timer, each resetting the other. These types of timers are used in situations where a constant cycling of an output is required. For example, if a flashing light is required in the event of a control system failure, a program with reciprocating timers could be used to create the flashing output function,

as illustrated in the PAC program of Figure 15-68. The operation of the program can be summarized as follows:

- When the contacts of the Limit_Switch close, Timer_Reciprocationg_Off beings its timing cycle.
- After two seconds, the Timer_Reciprocationg_Off.DN bit status changes to true and both the Timer_Reciprocating_On instruction and the flashing Alarm_PL become energized.
- After two more seconds, the Timer_Reciprocating_On.DN bit status changes to reset the sequence.

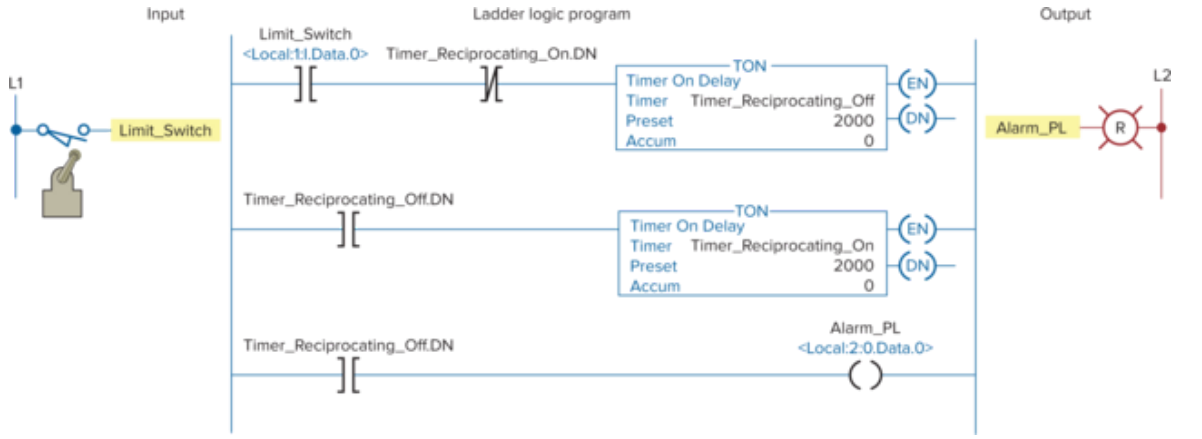


Figure 15-68 Flashing pilot light program.

PART 3 REVIEW QUESTIONS

1. Compare the methods used to address timers in RSLogix 500 and RSLogix 5000.
2. List the five different members of a TIMER structure.
3. What type of timing application may require you to use a TON on-delay timer?
4. What PRE value is used for a timer?
5. To what value is the accumulated value of a timer normally set?
6. What timer status bit is set to 1 when the TON timer times out?
7. The TON instruction is self-resetting. Explain what this means.
8. What number would be entered into the PRE value of a ControlLogix timer for a timing period of 4.5 minutes?
9. Compare the operation a TOF and a TON timer.
10. When does the rung of a TOF timer begin accumulating time?
11. The RTO timer is a retentive timer. Explain what this means.
12. How are the retentive timer and reset instruction related?



PART 3 PROBLEMS

1. Modify the original CLX ten-second TON timer program (Figure 15-55) with an additional rung added to the program that will energize a solenoid whenever the timer is enabled and timing. The solenoid is to be connected to pin 6 of the digital output module.
2. With reference to the ladder logic of the CLX diverter gate program (Figure 15-56), assume the solenoid gate fails to energize as programmed. You suspect the problem is due to an open in the solenoid coil or wiring to it. How might observation of the solenoid output status light help confirm this?
3. You are required to extend the Green light-on time of the CLX traffic control program (Figure 15-58) to 40 seconds. What changes would have to be made to the program?
4. With reference to the CLX heating oven process program (Figure 15-62), assume the oven-on pilot light burns out. In what way would the operation of the program be affected?
5. With reference to the CLX limit switch RTO program (Figure 15-65), in addition to the alarm, you are required to install a warning pilot light to indicate that the timer has timed out. How would you proceed?
6. Implement the hardwired TON alarm circuit of Figure 15-69 in Logix format.

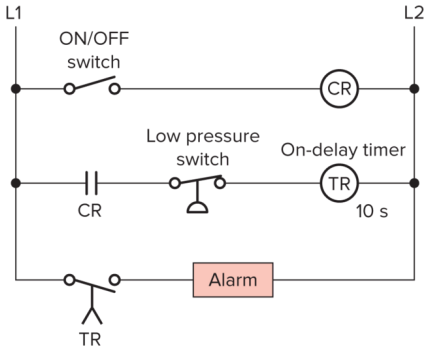


Figure 15-69 Hardwired TON alarm circuit for Problem 6.

OceanofPDF.com

Part 4 Programming Counters

Part Objectives

After completing this part, you will be able to:

- Understand ControlLogix counter tags and their members
- Utilize status bits from counters in logic
- Develop ladder logic programs using ControlLogix counters

Counters

Counters are similar to timers, except that a counter accumulates (counts) the changes in state of an external trigger signal whereas timers increment using an internal clock. PLC counters are generally triggered by a change in an input field device that causes a false-to-true transition of the counter ladder rung. It does not matter how long the rung stays true or false—it is only the transition that counts.

There are two basic counter types: count-up (CTU) and count-down (CTD). The ControlLogix CTU instruction and counter selection toolbar are shown in **Figure 15-70**. When you want to use a counter, you must create a **tag of type COUNTER** (it is a predefined data type) and enter the preset and the accumulated value. When entering the instruction, this tag must be defined before the preset and accumulated values can be entered. A RES reset instruction that has the same tag name as the counter must be used to reset the accumulated value of the counter to zero.

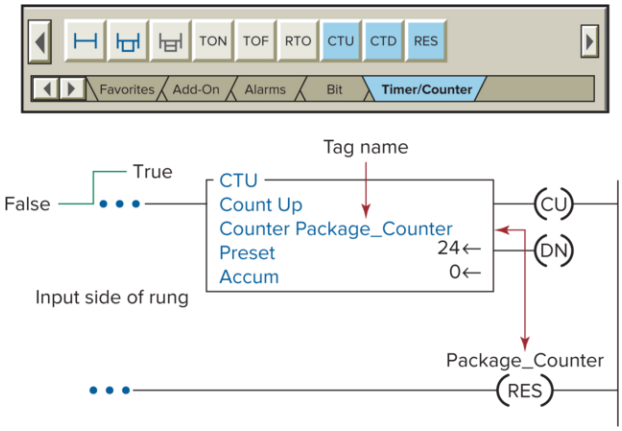


Figure 15-70 CTU count-up counter instruction.

All counters are retentive in that the accumulated value of any counter is retained, even during a power failure, until reset. The on/off status of the counter done, overflow, and underflow bits are retentive as well. ControlLogix counter parameters and status bits are shown in the edit tags window of Figure 15-71 and can be summarized as follows:

- **Preset (PRE) Value**—Specifies the value the counter must reach before the done (DN) bit turns on (1).
- **Accumulated (ACC) Value**—Is the number of false-to-true transitions of the counter rung. ACC is reset to zero when a reset (RES) instruction (of the same counter address) is executed.
- **CU (Count-Up Enable Bit)**—The count-up enable bit indicates the CTU instruction is enabled. Page 361
- **CD (Count-Down Enable Bit)**—The count-down enable bit indicates the CTD instruction is enabled.
- **DN (Count-Up Done Bit)**—The DN bit is set (1) when ACC value is equal to or greater than the PRE value and is reset by the RES instruction.
- **OV (Overflow Bit)**—The overflow bit indicates the counter exceeded the upper limit. The OV bit is set when the ACC value is greater than +2,147,483,647 and reset when the reset instruction is executed. Note that the accumulated value keeps incrementing even after the ACC value equals the PRE value.
- **UN (Underflow Bit)**—Indicates that the counter exceeded the lower limit of −2,147,483,648.

Tag Name	Data Type	Style
-Part_Counter	COUNTER	Decimal
-Part_Counter.PRE	DINT	Decimal
-Part_Counter.ACC	DINT	Decimal
Part_Counter.CU	BOOL	Decimal
Part_Counter.CD	BOOL	Decimal
Part_Counter.DN	BOOL	Decimal
Part_Counter.OV	BOOL	Decimal
Part_Counter.UN	BOOL	Decimal

Figure 15-71 ControlLogix counter parameters and status bits.



The counter tag name is declared using the new tag properties dialog box shown in Figure 15-72. Tag name, description (optional), tag type, data type (base type is used most often), and scope are selected or typed to complete the validation.

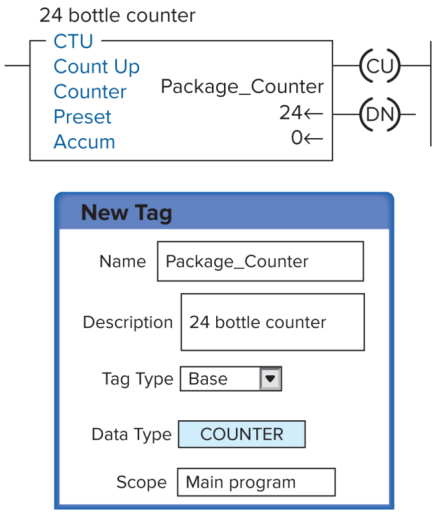


Figure 15-72 Counter tag validation.

Count-Up (CTU) Counter

Count-up (CTU) counters will cause the accumulated count to increase by 1 every time there is a false-to-true transition of the counter ladder rung. An example application of a count-up counter program used to count packets of bottles is shown in **Figure 15-73**. The operation of the program can be summarized as follows:

Page 362

- Each open-to-close transition of the Bottle_Sensor proximity switch causes the counter to increment by 1.
- The Increment_PL controlled by the Package_Counter.CU status bit turns on and off as each bottle passes to show that the counter is incrementing.
- When the accumulated value of the counter is 24 the DN bit of the counter is set and switches on the Preset_Reached_PL.
- The counter is reset by momentarily closing the Reset_Button.

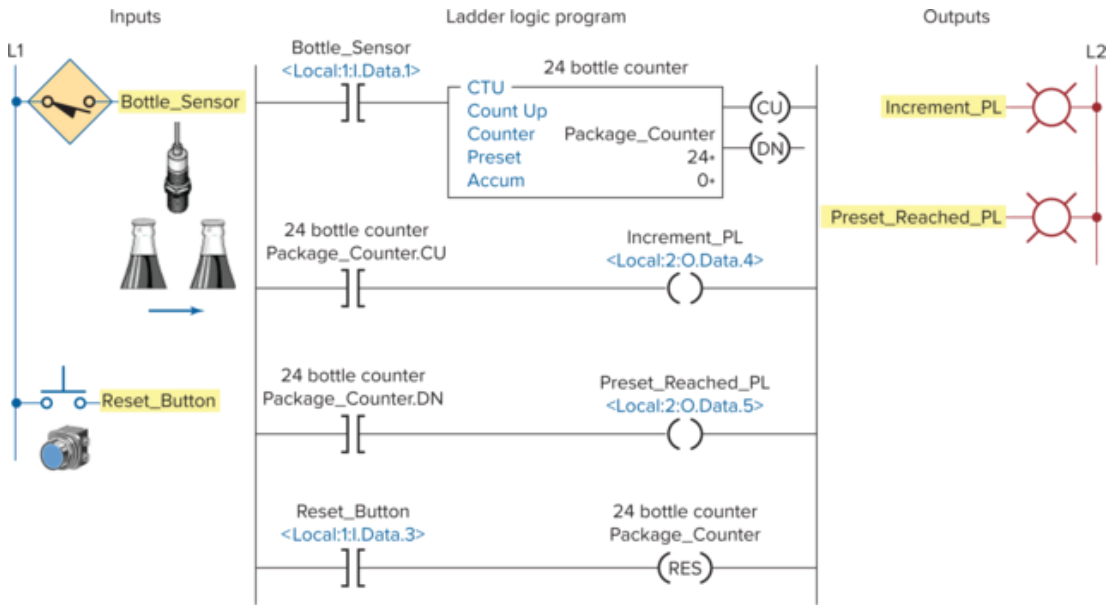


Figure 15-73 Count-up counter program used to count packets of bottles.

The program shown in **Figure 15-74** uses two CTU instructions as part of a program to remove 5 out of every 10 containers from a conveyor line using an electric solenoid. The different tags created to fit the program are shown in **Figure 15-75**. The operation of the program can be summarized as follows:

Page 363

- The preset for the Container_Counter_Counts is set for 6 and that for the Container_Counter_Max is set to 11.
- When the container is detected both counters will increase their accumulated values by 1.
- When the sixth part arrives the Container_Counter_Counts counter will then be done, thereby allowing the solenoid to actuate for any container after the fifth.
- The Container_Counter_Max counter will continue until the eleventh part is detected and then both of the counters will be reset.

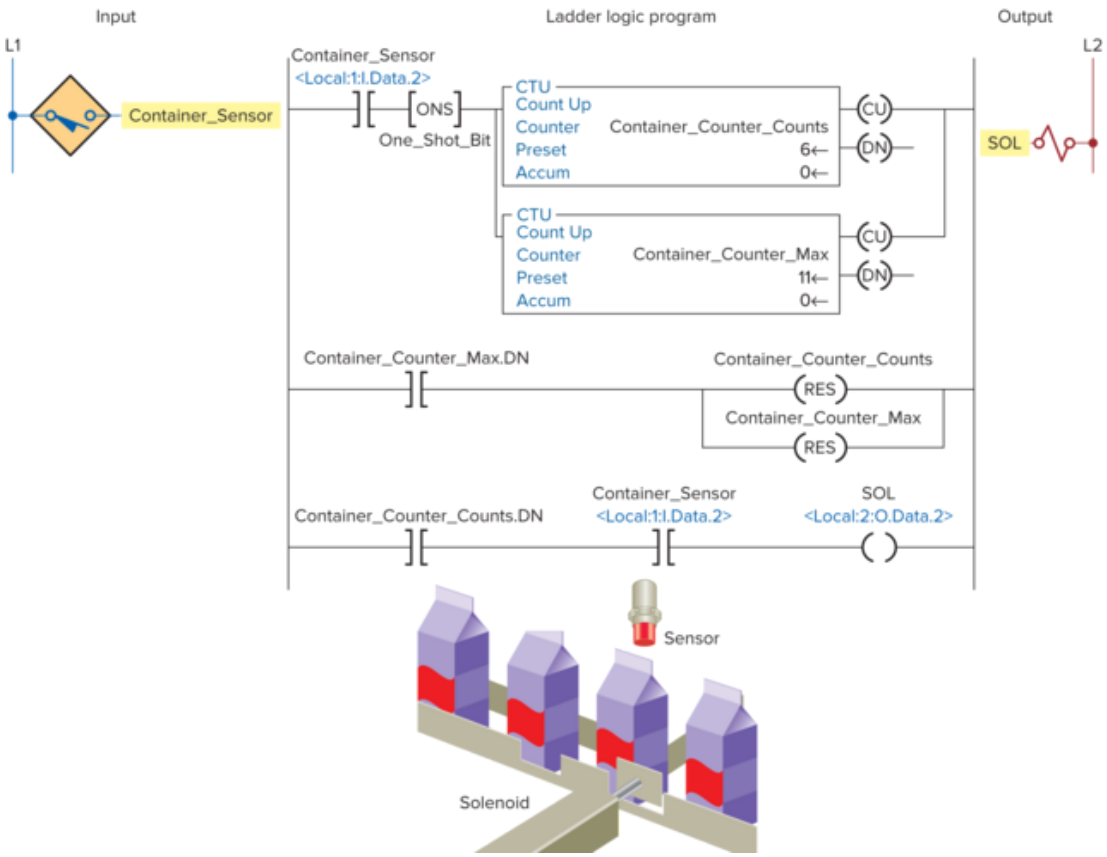


Figure 15-74 CTU program used to remove containers from a conveyor line.

Tag Name	Value	Style	Data Type
Container_Counter_Counts	{...}		COUNTER
Container_Counter_Counts .PRE	6	Decimal	DINT
Container_Counter_Counts .ACC	0	Decimal	DINT
Container_Counter_Counts .CU	0	Decimal	BOOL
Container_Counter_Counts .CD	0	Decimal	BOOL
Container_Counter_Counts .DN	0	Decimal	BOOL
Container_Counter_Counts .OV	0	Decimal	BOOL
Container_Counter_Counts .UN	0	Decimal	BOOL
Container_Counter_Max	{...}		COUNTER
Container_Counter_Max .PRE	11	Decimal	DINT
Container_Counter_Max .ACC	0	Decimal	DINT
Container_Counter_Max .CU	0	Decimal	BOOL
Container_Counter_Max .CD	0	Decimal	BOOL
Container_Counter_Max .DN	0	Decimal	BOOL
Container_Counter_Max .OV	0	Decimal	BOOL
Container_Counter_Max .UN	0	Decimal	BOOL
Container_Sensor	0	Decimal	BOOL
SOL	0	Decimal	BOOL

Figure 15-75 Tags created for the CTU program used to remove containers from a conveyor line.

A timer can used to drive a counter when an extremely long time-delay period is required. The PLC program of **Figure 15-76** illustrates an example of this. The execution of the program is summarized as follows:

- For purposes of simplification the preset value of Base_Timer is set at10 s (10000 ms) and Drive_Counter preset to value of 5.
- Each time the Base_Timer times out after 10 s, its done bit resets the timer and increments Drive_Counter accumulated valve by 1.
- Once the done bit of the Base_Timer has turned on and off 5 times, the done bit of the Drive_Counter is set to switch Ouput_PL on.
- The output light turns on after 50 seconds after the timer input contact closes.
- Therefore, whatever of the timer preset the net time-delay period will be 5 times greater.

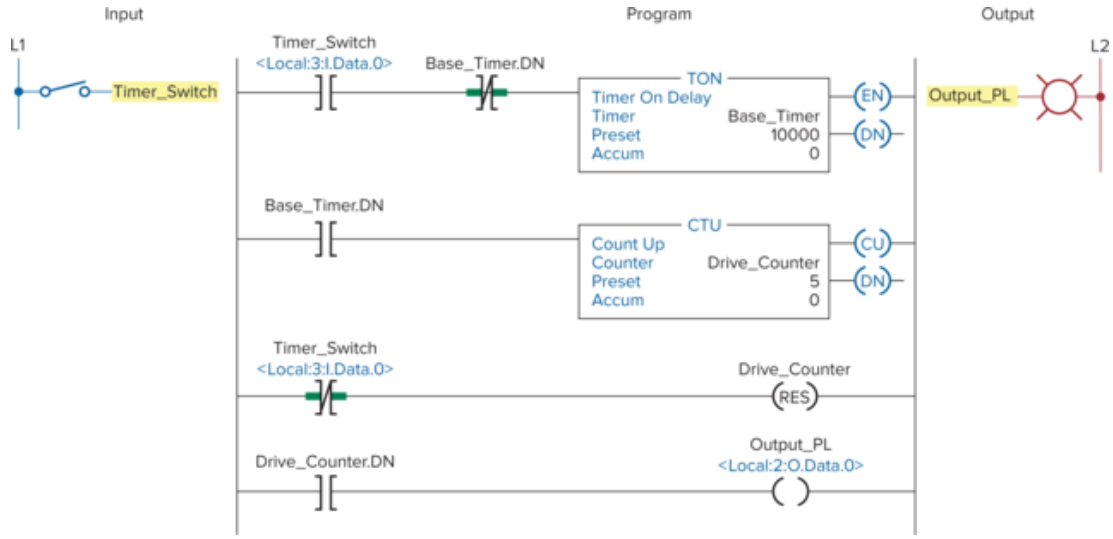


Figure 15-76 Timer used to drive a counter.

Count-Down (CTD) Counter

The ***count-down (CTD) counter*** operates in a fashion opposite to the count-up CTU counter. CTD counters will cause the accumulated count to decrease instead of increase by one every time there is a false-to-true transition of the counter ladder rung. The ControlLogix CTD down-counter instruction is shown in **Figure 15-77**. The descriptions of the function block fields and the tag references are the same as those associated with the CTU function block. The CTD instruction is typically used with a CTU instruction that references the same counter structure.

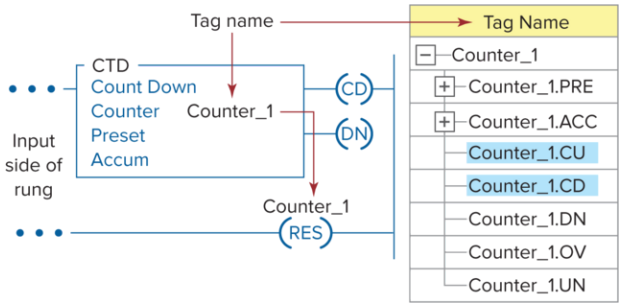


Figure 15-77 Count-down CTD counter instruction.

The application program shown in **Figure 15-78** is used to limit the number of parts that can be stored in the buffer zone to a maximum of 50. A CTU counter and a CTD counter are used together *with the same address* to form an Up/Down counter. This is the most common type of application of the CTD counter. The different tags created to fit the program are shown in **Figure 15-79**. The operation of the program can be summarized as follows:

- The Restart_Button is momentarily actuated at any time to reset the accumulated value of the counter to zero.
- Conveyor brings parts into a buffer zone.
- Each time a part enters the buffer zone, the Enter_Limit_Sw is actuated and Counter_1 increments by 1.
- Each time a part leaves the buffer zone, the Exit_Limit_Sw is actuated and Counter_1 decrements by 1.
- When the number of parts in the buffer zone, at any one time, reaches 50, the Counter_1.DN bit is set.
- As a result the Conveyor_Contactor rung goes false to de-energize the conveyor contactor, automatically stopping the conveyor from bringing in any more parts until the accumulated count drops below 50.

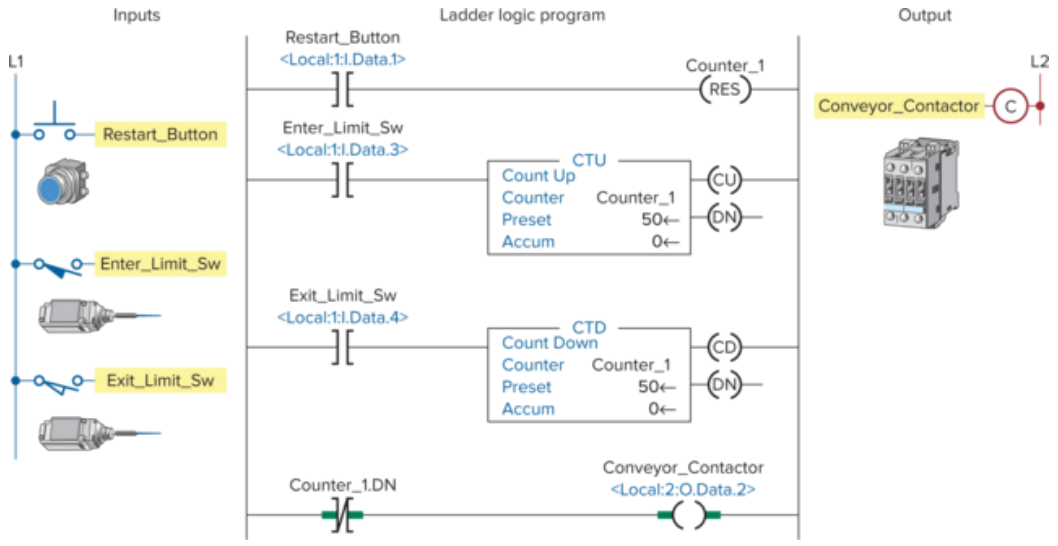


Figure 15-78 CTU counter and CTD counter used together to form an Up/Down counter.

Tag Name	Value	Style	Data Type
Counter_1	{...}		COUNTER
Counter_1.PRE	50	Decimal	DINT
Counter_1.ACC	0	Decimal	DINT
Counter_1.CU	0	Decimal	BOOL
Counter_1.CD	0	Decimal	BOOL
Counter_1.DN	0	Decimal	BOOL
Counter_1.OV	0	Decimal	BOOL
Counter_1.UN	0	Decimal	BOOL
Restart_Button	0	Decimal	BOOL
Enter_Limit_Sw	0	Decimal	BOOL
Exit_Limit_Sw	0	Decimal	BOOL
Conveyor_Contactor	1	Decimal	BOOL

Figure 15-79 Tags created for the Up/Down counter program.

Combining Counter and Timer Functions

Figure 15-80 shows a bottle flow rate program implemented using a Logix controller. This program is designed to indicate how many bottles pass a given process point per minute. The operation of the program can be summarized as follows:

- When the Start_Switch is closed, both the timer and counter are enabled.
- The Total_Bottles counter is pulsed for each bottle that passes the Bottle_Sensor.
- The counting begins and the 1_Min_Timer starts timing through its one minute (60000 milliseconds) time interval.
- At the end of one minute, the timer done bit (DN) causes the counter rung to go false.
- Sensor pulses continue but do not affect the Total_Bottles counter.
- The number of bottles for the past minute is represented by the accumulated value of the Total_Bottles counter.
- The sequence is reset by momentarily opening and closing the Start_Switch.

Page 365

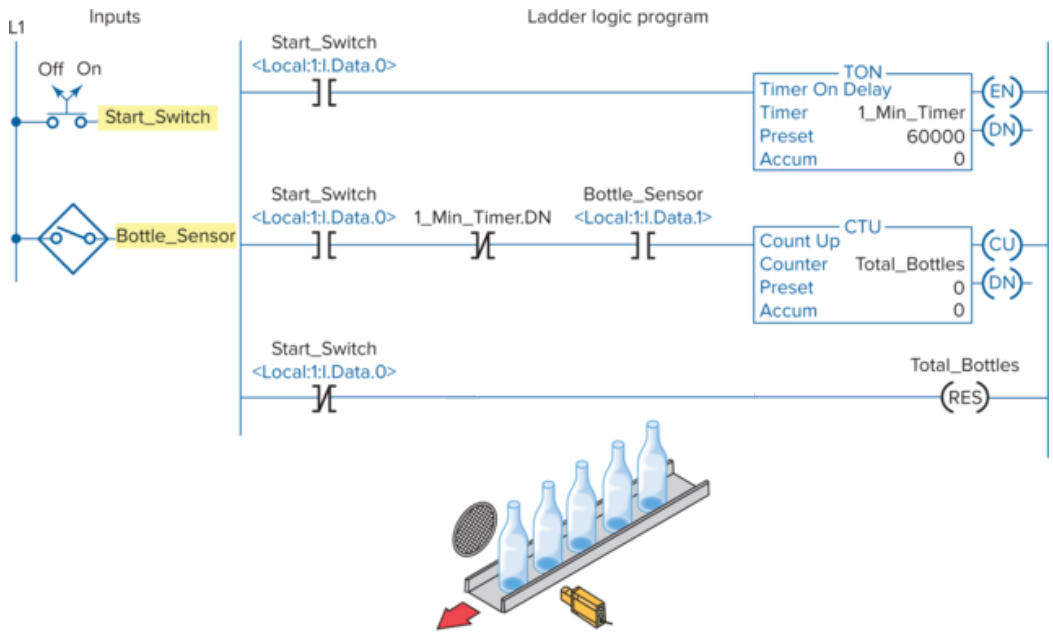


Figure 15-80 Bottle flow rate program.

PART 4 REVIEW QUESTIONS

1. In what way are timers and counters similar?
2. Outline the procedure followed to create a tag when you want to use a counter.
3. All counters are retentive. In what way does this affect their operation?
4. What is specified by the preset value of a counter?
5. When is each of the following counter bits set?
   a. CU
   b. DN
   c. CD
6. Compare the operations of a CTU and a CTD counter.
7. What is an Up/Down counter?
8. Explain how you go about creating tags for an Up/Down counter that uses a CTU and CTD instruction.



PART 4 PROBLEMS

1. With reference to the CTU packets of bottles program (Figure 15-73), what changes to the program would be required to count six bottle packets?
2. With reference to the CTU program used to remove containers from a conveyor line (Figure 15-74), assume the output solenoid coil failed open. In what way would the operation of the program be affected?
3. Modify the original Up/Down counter program (Figure 15-78) to include:
   a. A red pilot light to indicate entry of a part into the buffer zone. Light to be connected to pin 4 of the digital output module.
   b. A green pilot light to indicate exit of a part from the buffer zone. Light to be connected to pin 3 of the digital output module.
4. Write a ControlLogix program, complete with tags, for an Up/Down counter used to keep track of cars entering and exiting a parking lot. The program requirements for this application can be summarized as follows:
   - The parking lot holds 30 vehicles.
   - There is an entrance vehicle sensor and an exit vehicle sensor.
   - When the parking lot is full a Lot Full sign is illuminated.
   - Whenever a car exits the lot, a Caution Buzzer/Light is activated to warn pedestrians.

OceanofPDF.com

Part 5 Math, Comparison, and Move Instructions

Part Objectives

After completing this part, you will be able to:

- Utilize ControlLogix math instructions in programs
- Utilize ControlLogix comparison instructions in programs
- Utilize ControlLogix move instructions in programs
- Develop and follow the operation of programs that use math, comparison, and move instructions
- Develop a program that will display the time lapsed period of a timer
- Utilize ControlLogix mask move instructions in programs
- Utilize ControlLogix bit field instructions in programs

Math Instructions

ControlLogix basic math instructions include addition, subtraction, multiplication, division, and square root. Figure 15-81 shows the Compute/Math toolbar for the ControlLogix controller.

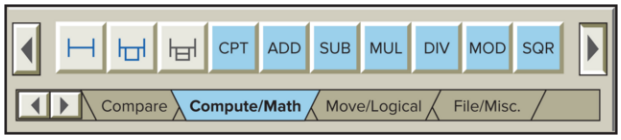


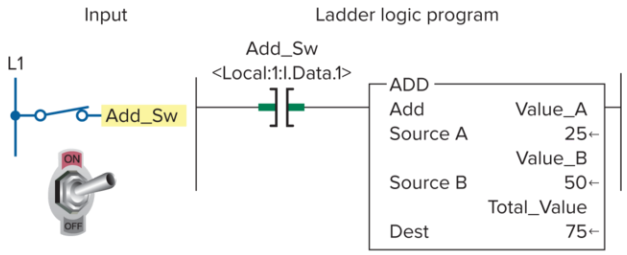
Figure 15-81 Compute/Math toolbar for the ControlLogix controller.

The ***ADD* instruction** is used to add two numbers. This instruction adds these values from Source A and Source B. The source can be a constant value or a tag. The result of the ADD instruction is put in the destination (Dest) tag.

Figure 15-82 shows an example of an ADD instruction rung along with its Monitor Tags window. The operation of the rung can be summarized as follows:

- When the Add_Sw is closed the rung will be true.

- The ADD instruction will execute to add the number from Source A (Value_A) and the value from Source B (Value_B).
- The result will be stored in the Dest tag (Total_Value).
- In this example, the 25 was added to 50 and the result (75) was stored in Total_Value.

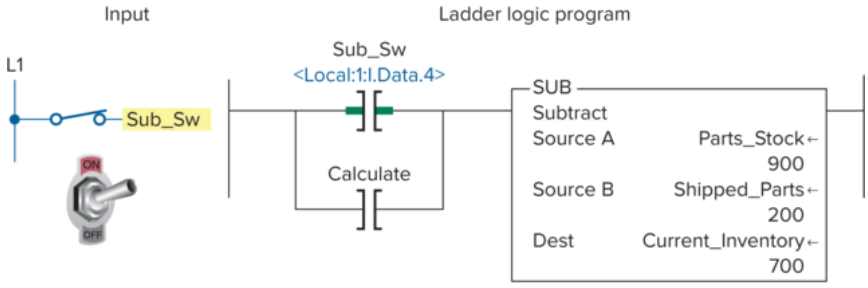


Tag Name	Value	Style	Data Type
Total_Value	75	Decimal	DINT
Value_A	25	Decimal	DINT
Value_B	50	Decimal	DINT
Add_Sw	1	Decimal	BOOL

Figure 15-82 ADD instruction rung and its Monitor Tags window.

The ***SUB* instruction** is used to subtract two numbers. **Figure 15-83** shows an example of a SUB instruction rung along with its Monitor Tags window. The operation of the rung can be summarized as follows:

- When the SUB_Sw or Calculate tag is true the SUB instruction is executed.
- Source B (Shipped_Parts) is subtracted from Source A (Parts_Stock) and the result is stored in the Dest tag named Current_Inventory.
- In this example, the 200 was subtracted from 900 and the result (700) was stored in Current_Inventory.
- Source A and Source B can be constants (numbers) or tags.

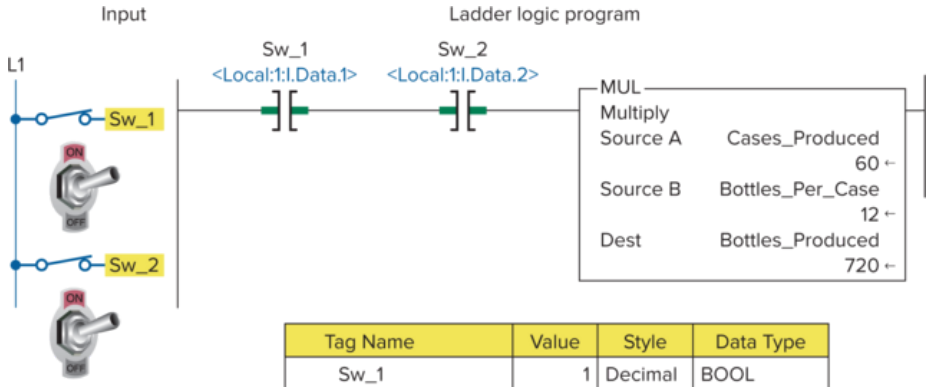


Tag Name	Value	Style	Data Type
Parts_Stock	900	Decimal	DINT
Shipped_Parts	200	Decimal	DINT
Current_Inventory	700	Decimal	DINT
Sub_Sw	1	Decimal	BOOL
Calculate	0	Decimal	BOOL

Figure 15-83 SUB instruction rung and its Monitor Tags window.

The ***MUL* instruction** is used to multiply two numbers. **Figure 15-84** shows an example of a MUL instruction rung along with its Monitor Tags window. When multiple bottles are packed in cases, the number of bottles per case, the number of cases, and the multiply instruction will give you the total number of bottles. The operation of the rung can be summarized as follows:

- When the Sw_1 and Sw_2 are both true the MUL instruction is executed.
- Source A (the value in tag Cases_Produced) is multiplied by Source B (the value in tag Bottles_Per_Case) and the result is stored in the Dest tag Bottles_Produced.
- Source A and Source B can be constants (numbers) or tags.

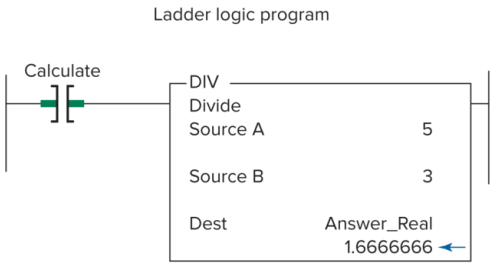


Tag Name	Value	Style	Data Type
Sw_1	1	Decimal	BOOL
Sw_2	1	Decimal	BOOL
Cases_Produced	60	Decimal	DINT
Bottles_Per_Case	12	Decimal	DINT
Bottles_Produced	720	Decimal	DINT

Figure 15-84 MUL instruction rung and its Monitor Tags window.

The ***DIV* instruction** is used to divide two numbers. **Figure 15-85** shows an example of a DIV instruction rung along with its Monitor Tags window. The operation of the rung can be summarized as follows:

- A constant (5) is used for Source A and a constant (3) for Source B. Note that tags could have been used for Source A or Source B.
- When the Calculate tag is true the DIV instruction is executed.
- Source A (5) is divided by Source B (3) and the result (1.6666666) is stored in the Dest tag Answer_Real. Note that in this example a Real-type tag has been used for its destination.

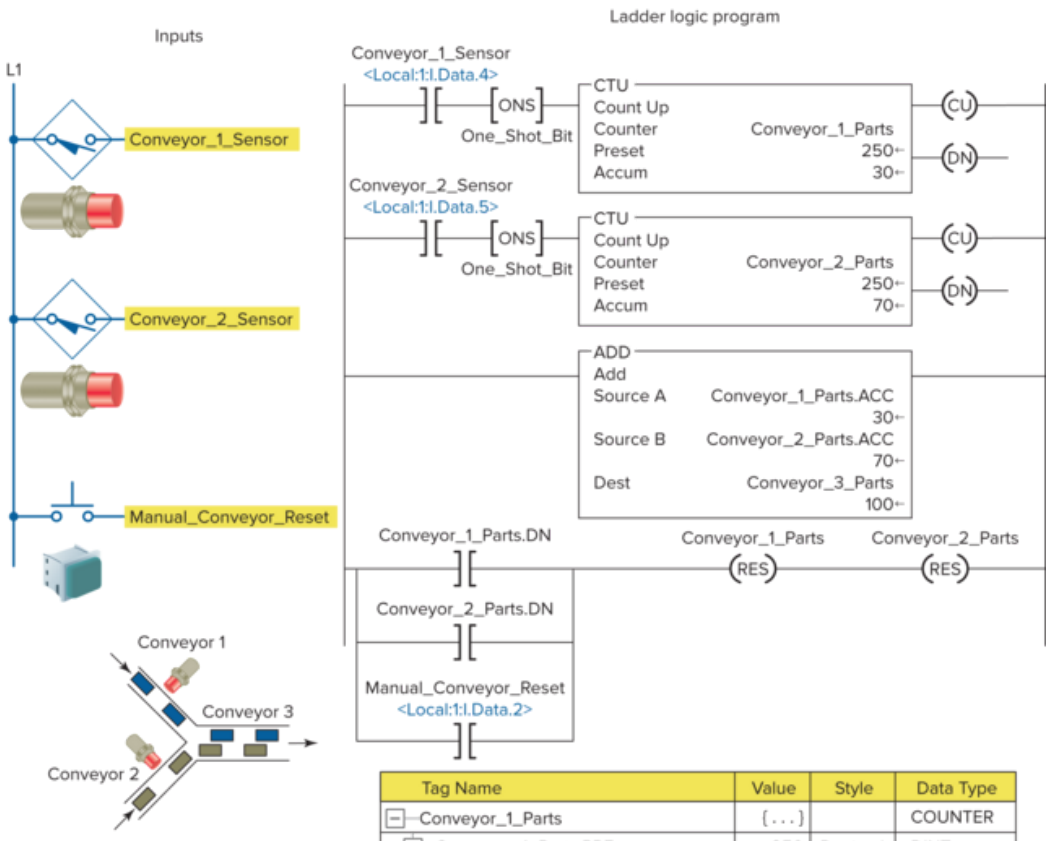


Tag Name	Value	Style	Data Type
Calculate	1	Decimal	BOOL
Answer_Real	1.6666666	Float	REAL

Figure 15-85 DIV instruction rung and its Monitor Tags window.

The program of **Figure 15-86** is used as part of a parts tracking system with three conveyors. The number of parts in conveyor 1 and the number of parts in conveyor 2 are added to get the number of parts on conveyor 3. The operation of the program can be summarized as follows:

- Each time Conveyor_1_Sensor is actuated the accumulated value of Counter_1_Parts is incremented by 1.
- Each time Conveyor_2_Sensor is actuated the accumulated value of Counter_2_Parts is incremented by 1.
- The addition in the ADD instruction places the sum of the accumulated values of the two counters in the Conveyor_3_Parts tag.
- When the accumulated value for either counter is equal to 150 the reset (RES) instructions for both counters are enabled to automatically reset both counter ACC values to zero.
- Both counters can also be reset manually at any time by actuation of the Manual_Conveyor_Reset button.



Tag Name	Value	Style	Data Type
Conveyor_1_Parts	{...}		COUNTER
Conveyor_1_Parts.PRE	250	Decimal	DINT
Conveyor_1_Parts.ACC	30	Decimal	DINT
Conveyor_1_Parts.CU	0	Decimal	BOOL
Conveyor_1_Parts.DN	0	Decimal	BOOL
Conveyor_2_Parts	{...}		COUNTER
Conveyor_2_Parts.PRE	250	Decimal	DINT
Conveyor_2_Parts.ACC	70	Decimal	DINT
Conveyor_2_Parts.CU	0	Decimal	BOOL
Conveyor_2_Parts.DN	0	Decimal	BOOL
Conveyor_1_Sensor	0	Decimal	BOOL
Conveyor_2_Sensor	0	Decimal	BOOL
Conveyor_3_Parts	100	Decimal	DINT
Manual_Conveyor_Reset	0	Decimal	BOOL

Figure 15-86 Program used as part of a parts tracking system.

Comparison Instructions

Compare instructions are used to compare two values. They can be used to see if two values are equal, if one value is greater or less than the other, and so on. In ControlLogix controllers, compare instructions are input instructions that do comparisons by either using an expression or doing the comparison indicated by the specific instruction. **Figure 15-87** shows the Compare toolbar for the ControlLogix controller.

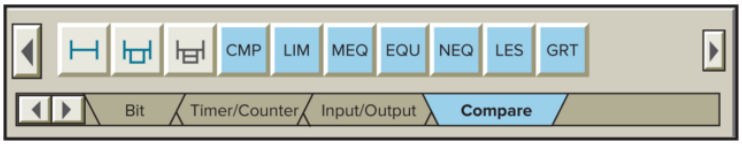
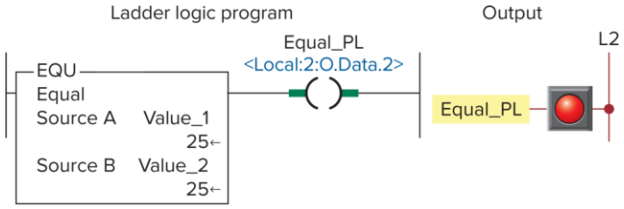


Figure 15-87 Compare toolbar for the ControlLogix controller.

The ***equal (EQU)* instruction** is used to test if two values are equal. Values compared can be actual values or tags that contain values. **Figure 15-88** shows an example of an EQU instruction rung along with its Monitor Tags window. The operation of the rung can be summarized as follows:

- The value stored at Source A is compared to the value stored at Source B.
- If the values are equal, the instruction is logically true.
- If the values are unequal, the instruction is logically false.
- In this example Source A (25) is equal to Source B (25) so the instruction is true and output Equal_PL is on.
- Source A and Source B may be SINT, INT, DINT, or REAL data types.



Tag Name	Value	Style	Data Type
Equal_PL	1	Decimal	BOOL
+Value_1	25	Decimal	DINT
+Value_2	25	Decimal	DINT

Figure 15-88 EQU instruction rung and its Monitor Tags window.

The ***not equal (NEQ)*** instruction is used to test two values for inequality. **Figure 15-89** shows an example of an NEQ instruction rung. When Source A is not equal to Source B, the instruction is logically true; otherwise, it is logically false. In this example the two values are not equal so the Not_Equal_PL is energized.

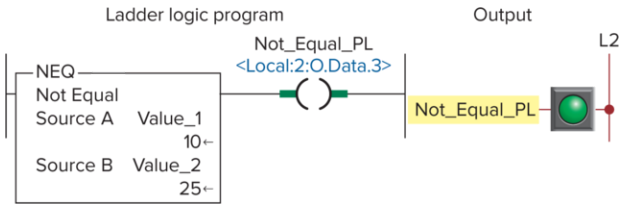


Figure 15-89 NEQ instruction rung.

The ***less than (LES)*** instruction is used to check if a value from one source is less than the value from a second source. **Figure 15-90** shows an example of an LES instruction rung. When Source A is less than Source B, the instruction is logically true; otherwise, it is logically false. In this example Value_1 (100) is less than Value_2 (300) so the Less_Than_PL is energized.

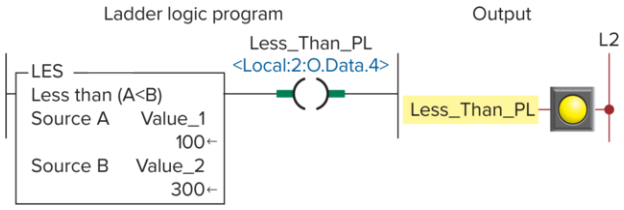


Figure 15-90 LES instruction rung.

The ***greater than (GRT)*** instruction is used to check if a value from one source is greater than the value from a second source. **Figure 15-91** shows an example of a GRT instruction rung. When Source A is greater than Source B, the instruction is logically true; otherwise, it is logically false. In this example Value_1 (1420) is greater than Value_2 (1200) so the Greater_Than_PL is energized.

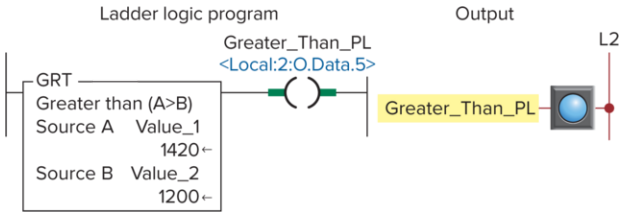
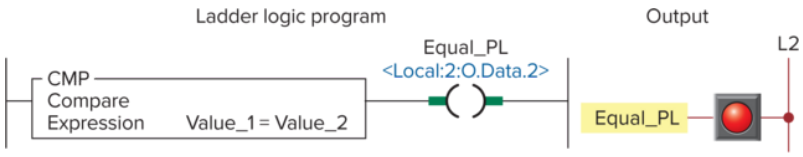


Figure 15-91 GRT instruction rung.

The ***compare (CMP)*** instruction performs a comparison on the arithmetic operations specified by the expression. The expression may contain arithmetic operators, comparison operators, and tags. The execution of a CMP instruction is slightly slower and uses more memory than the execution of the other comparison instructions. The advantage of the CMP instruction is that it allows you to enter complex expressions in one instruction. **Figure 15-92** shows an example of a CMP instruction rung. In this example the comparison operator found in the expression is the equivalent of an EQU instruction. The comparison instruction is true because Value_1 (300) is equal to Value_2 (300).



Tag Name	Value	Style	Data Type
Equal_PL	1	Decimal	BOOL
⊞ Value_1	300	Decimal	DINT
⊞ Value_2	300	Decimal	DINT

Figure 15-92 CMP instruction rung.

The program of Figure 15-93 is an example of the use of comparison instructions used to test the accumulated value of a counter. The operation of the program can be summarized as follows:

- When the accumulated count is between 5 and 10 the GRT and LES instructions will both be logically true so the PL_1 pilot light will be on.
- When the accumulated count is equal to 15, the EQU instruction will be logically true so the PL_2 pilot light will be on.
- The PL_3 pilot light will be on at all times except when the accumulated count is 20 at which time the NEQ instruction is logically false.
- The counter is reset automatically when the accumulated count reaches 25 or manually anytime the Reset_PB is actuated.

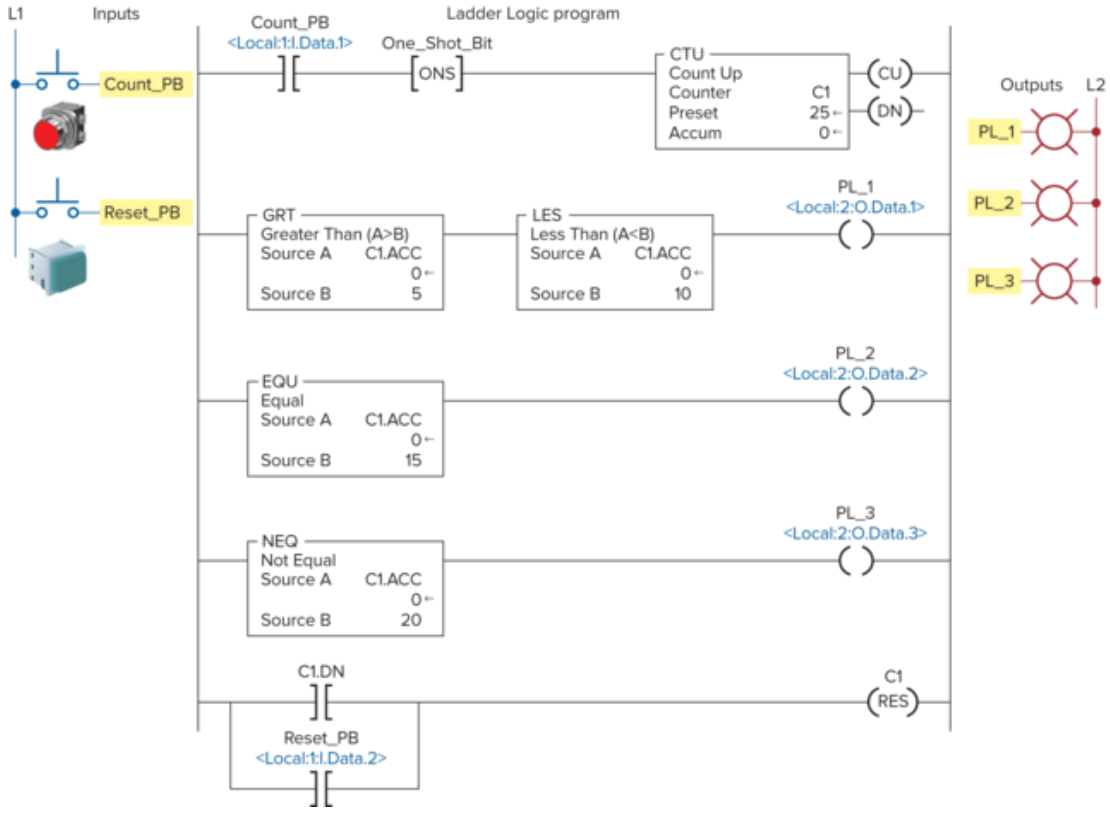


Figure 15-93 Comparison instructions used to test the accumulated value of a counter.

Move Instructions

The ***move (MOV)*** instruction is an output instruction that can move a constant or the contents of one memory location to another location. Figure 15-94 shows the Move toolbar and instruction for the

ControlLogix controller. The MOV instruction is used to copy data from a source to a destination. Both the source and the destination data type of a MOV instruction may be INT, DINT, SINT, or REAL.

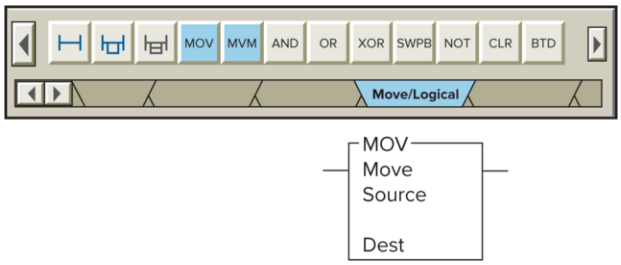


Figure 15-94 Move toolbar for the ControlLogix controller.

The program of **Figure 15-95** is an example of how the MOV instruction can be used to create a variable preset timer. The operation of the program can be summarized as follows:

- Actuating the PB_10s button executes its MOV instruction to transfer 10000 to the timer preset value setting the delay period for 10 seconds.
- Actuating the PB_15s button executes its MOV instruction to transfer 15000 to the timer preset value setting the delay period for 15 seconds.
- Closing the Timer_Start switch starts the timer timing.
- While the timer is timing, the pilot light PL_1 is on for the duration of the timer preset period.
- When the timer times out, PL_1 turns off and PL_2 turns on.

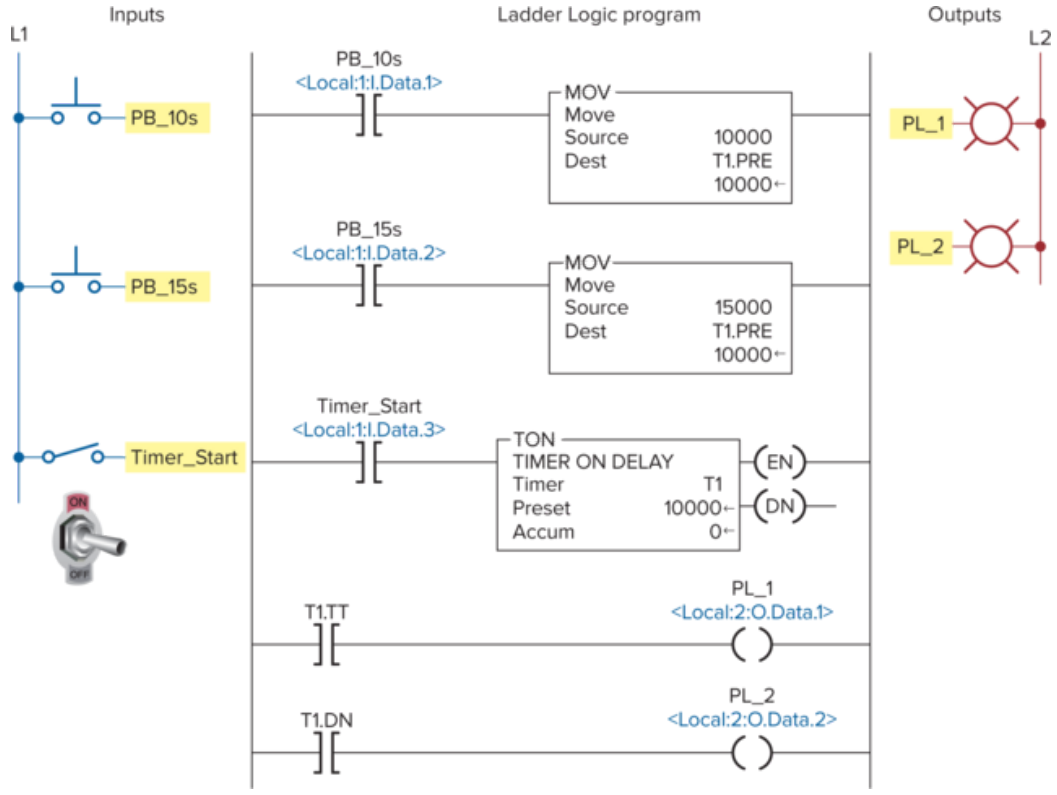


Figure 15-95 MOV instruction used to create a variable preset timer.

Combining Math, Comparison, and Move Instructions

Combining math, comparison, and move instructions provides a PLC with the ability to perform more complex operations. **Figure 15-96** shows a PAC Logix program designed so that the BCD_Display displays the setting of the BCD_TWS (thumbwheel switch). Both the MOV and EQU instructions form part of the program. The operation of the program can be summarized as follows:

- The BCD_Display board monitors the decimal setting of the BCD_ TWS.
- The MOV instruction is used to move the data (in the form of BCD) from the BCD_TWS to the BCD_Display.
- The setting of BCD_TWS is compared to the decimal reference number 20 (BCD 100000) stored in Source B of the EQU instruction.
- The EQUAL_ PL output is energized whenever the PL_Equal_Sw is true (closed) and the value of BCD_TWS is equal to 20 decimal (BCD 100000).

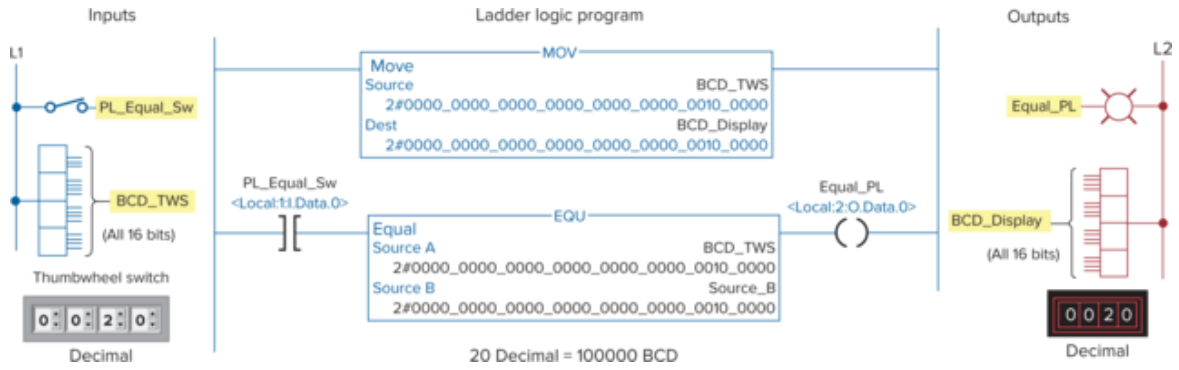


Figure 15-96 Monitoring the setting of a thumbwheel switch.

Figure 15-97 shows the PLC program for speed control of a three-speed motor conveyor system. The system comes equipped with a weight detector and three-speed motor controller. It is designed to move the conveyor belt at a certain speed when a specific value of weight is on the conveyor. If the weight exceeds the preset value, the conveyor speed increases to compensate for the increase in weight. If the weight falls below the preset value, the conveyor speed is reduced to compensate for the decrease in weight. The operation of the program can be summarized as follows:

- When the Conveyor_Start button is pressed, Internal_Relay seals in, and the conveyor begins moving.
- An analog input is used as the Source A location, which stores the data obtained from the weight detector.
- The reference, or Source B value is set at 50.
- The Conveyor_Load_Cell tag stores the data from the weight detector.

- If the data in the Conveyor_Load_Cell is equal to the Source B value, the motor operates at normal speed.
- If the data in the Conveyor_Load_Cell is above the reference value, due to more weight on the conveyor, the Weight_Above_Normal output is energized and a signal is sent to the motor controller to change to a higher speed.
- If the data in the Conveyor_Load_Cell is below the reference value, due to less weight on the conveyor, the Weight_Below_Normal output is energized and a signal is sent to the motor controller to decrease its speed.

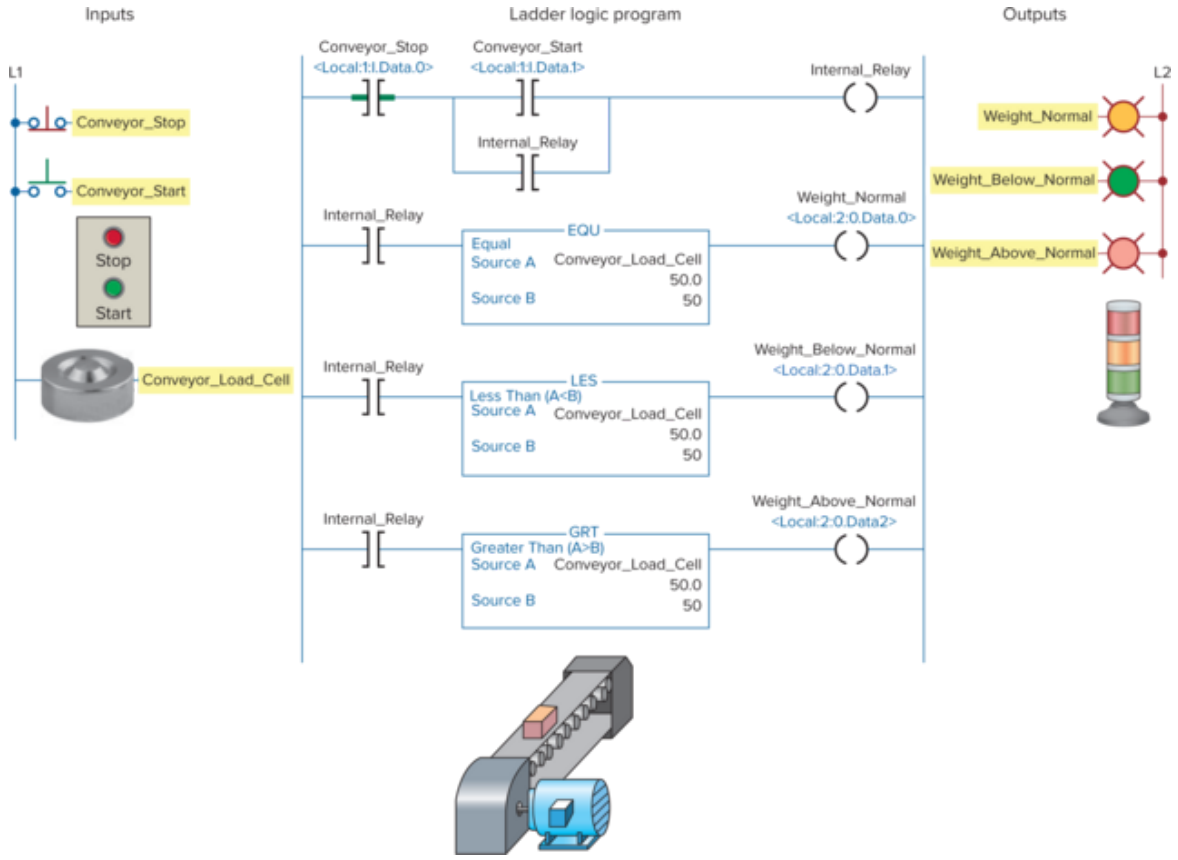


Figure 15-97 PLC program for three-speed control of a conveyor system.

Figure 15-98 shows the parts tracking program for a conveyor system consisting of a main conveyor and rejection conveyor. If a part fails inspection, the diverter gate is energized and the part is routed onto the reject conveyor. Each conveyor has a limit switch used to count parts as they pass by. The operation of the parts tracking program of the process can be summarized as follows:

- The Main_Conveyor_Counter.ACC contains the total number of parts that have been counted.
- The Reject_Conveyor_Counter.ACC contains the number of parts that have been rejected.
- The SUB instruction is used to provide a running count of items that have passed inspection and has as its Destination the Data_Format_Conversion tag.

- BCD_Output_Display is used to show the running count for passed items.
- Reset_PB switch is activated to reset both counters to zero and clear the BCD_Output_Display to zero.

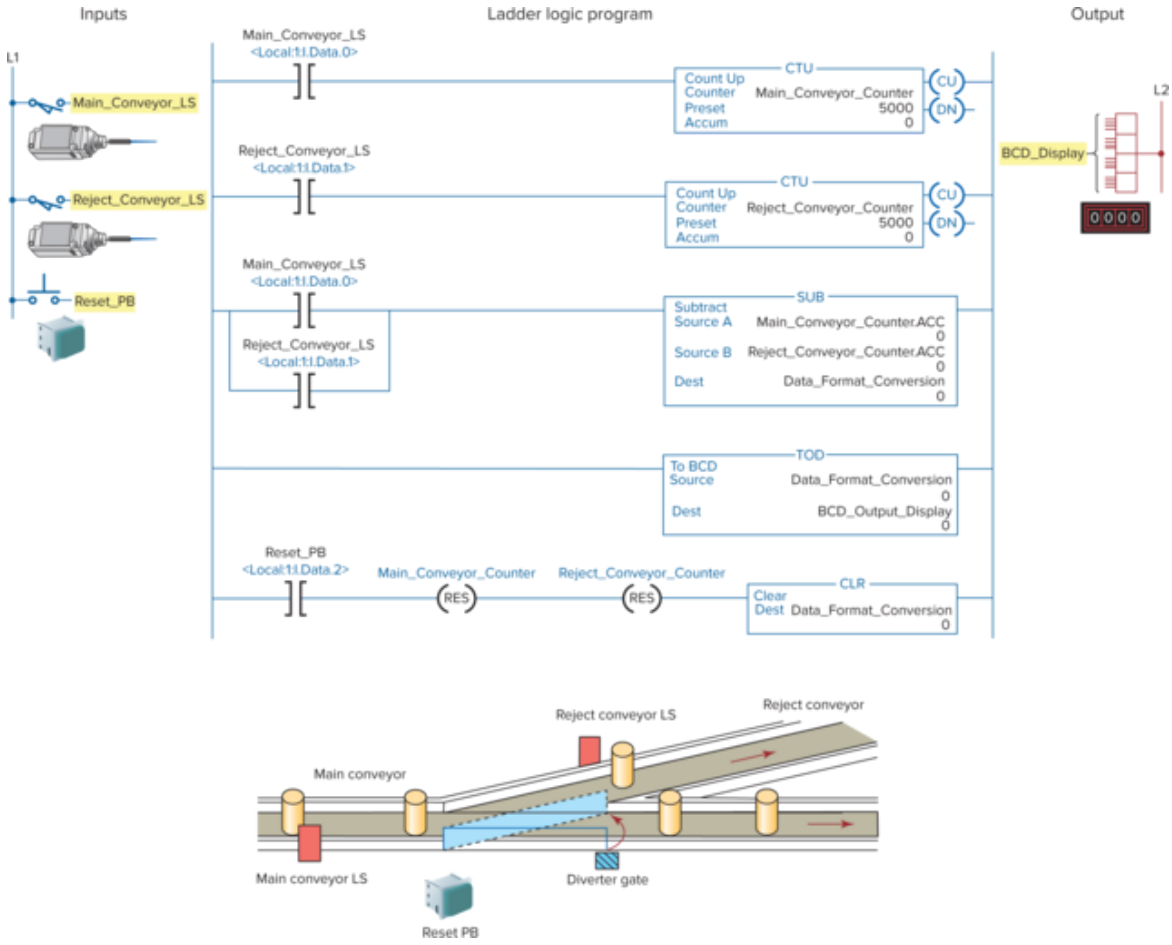


Figure 15-98 Conveyor parts tracking program.

The program of **Figure 15-99** is an example of how the lapsed time period of a timer is displayed using math instructions. The operation of the program is summarized as follows:

- The Time timer preset value is set to 10 seconds (10000ms).
- The SUB instruction subtracts the preset counter value from its accumulated value and sends the result to the Difference tag.
- The DIV instruction divides the Difference value by 1000, to convert it to seconds, and sends the result to the Quotient tag.
- The TOD instruction converts the binary Quotient value to BCD and sends the result to the BCD_Display.
- When the switch is closed the display shows the count down of the timer in seconds.

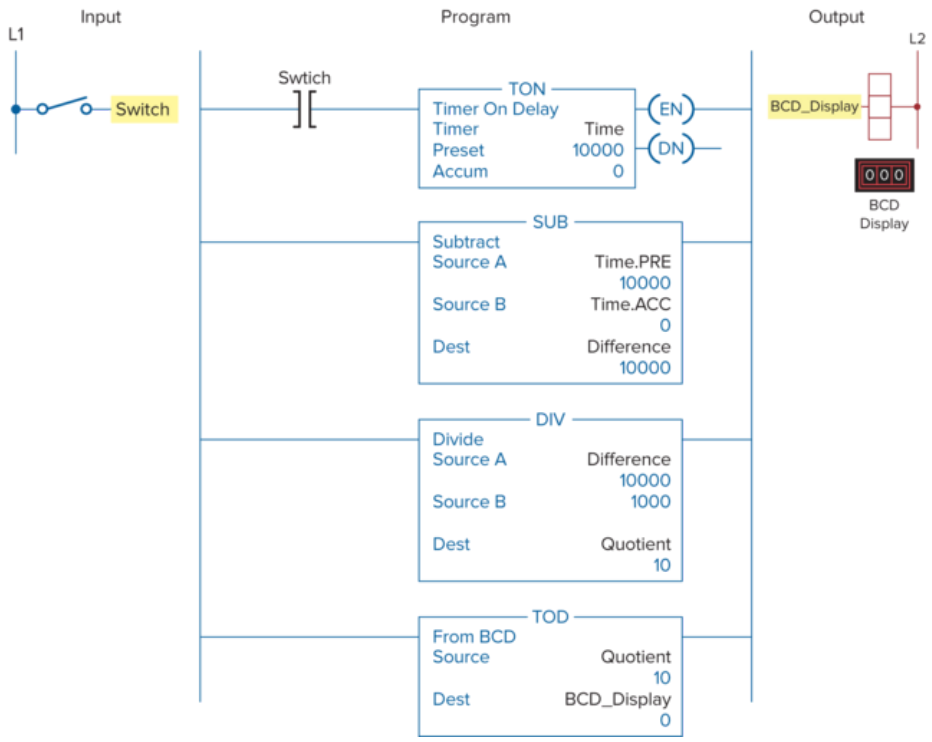


Figure 15-99 Displaying the lapsed time period of a timer program.

Cooling tower systems are used to remove heat from building air conditioning equipment and industrial processes. They operate by using water to extract waste heat from a system and ejecting it into the atmosphere primarily through evaporation. The most common method for controlling the temperature of the water leaving a cooling tower is to cycle the tower fans on and off. **Figure 15-100** shows an example of a PLC two fan cooling tower program. The execution of the program is summarized as follows:

- The MUL instruction is used to configure the temperature transmitter voltage from 0-to-10 Volts to 0-to-100 Volts.
- The GRT instruction sets the FH_Fan latch temperature to 30°C.
- The LES instruction sets the FH_Fan unlatch temperature to 12°C
- The GRT instruction sets the FHH_Fan latch temperature to 45°C.
- The LES instruction sets the FHH_Fan unlatch temperature to 25°C
- As a result, cooling Fan FH will come On @ 30°C and turn Off @ 12°C, while cooling Fan FHH will come On @ 45°C and turn Off @ 25°C.

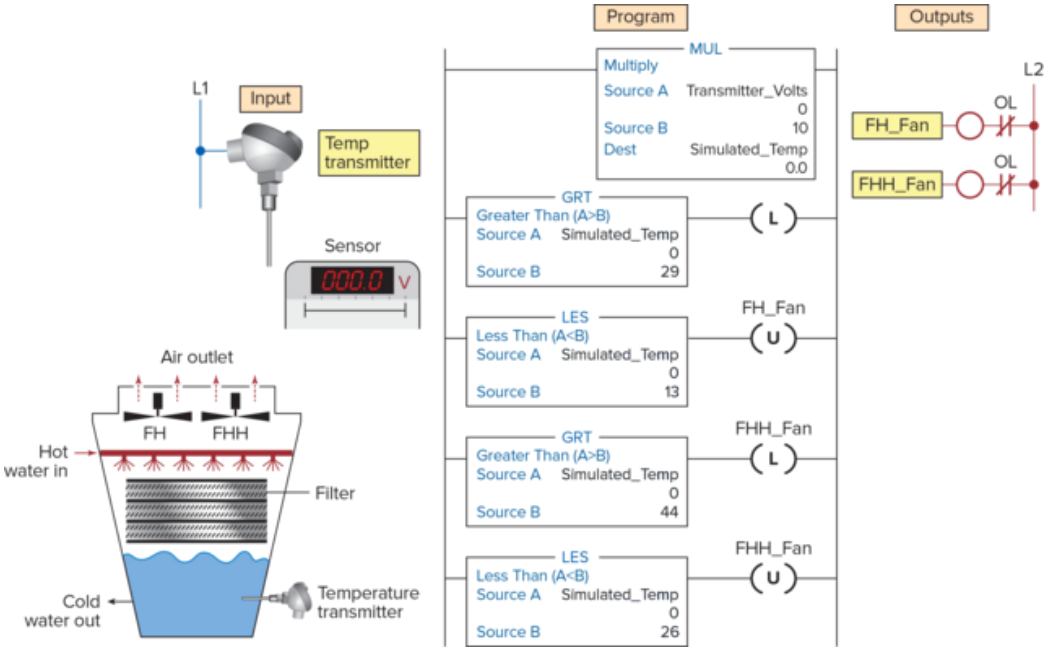


Figure 15-100 PLC two fan cooling tower program.

The **Move (MOV)** instruction shown in Figure 15-101, when enabled, transfers the state of each position of the source data to corresponding bit position of the destination memory location. Only the first 16 bit positions are shown. The execution of the program is summarized as follows:

- The bit pattern from Local:1:I.Data is to be copied to memory location Local:2:O.Data. Page 377
- The source and destination data structures must be either SINT, INT, or DINT.
- The instruction may be programmed with input conditions preceding it, or it may be programmed unconditionally.
- The source of data can be an immediate value or a tag name. An immediate value is a piece of data that is stored as part of the instruction itself instead of being in a memory location. Page 378
- The destination location must be a tag name.
- When the rung is true, input switch closed, the value stored at the source location, Local:1:I.Data, is copied into the destination location, — Local:2:O.Data.
- When the rung goes false, input switch opened, the destination location will retain the value unless it is changed elsewhere in the program.
- The source value remains unchanged and no data conversion occurs.

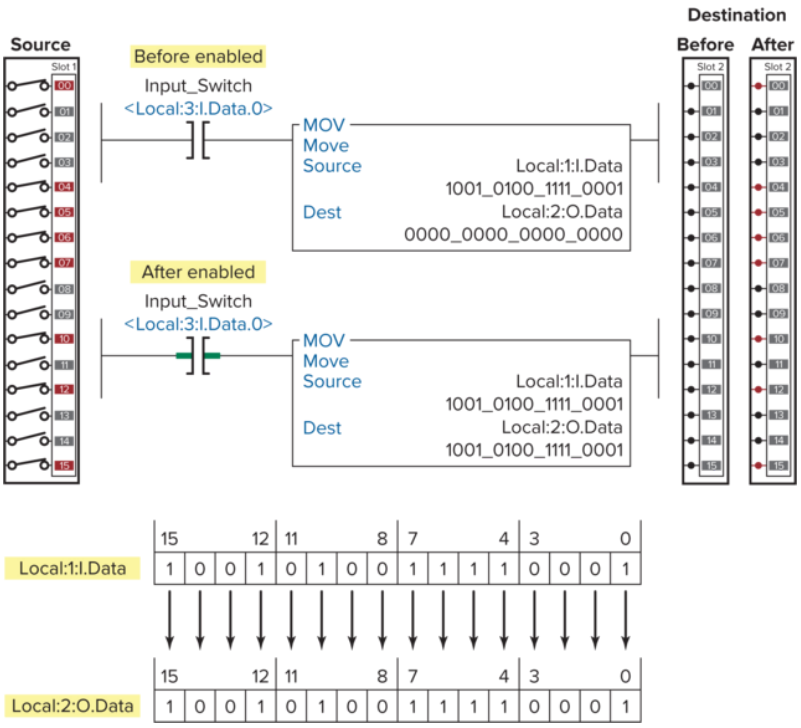


Figure 15-101 Move instruction used to move 32 bits from one memory location to another.

The **Masked Move (MVM)** instruction shown in Figure 15-102 functions like the standard move (MOV) instruction, except that a **mask** is used to allow data that that is not required to be filtered out of the move. This instruction executes to selectively control which bits are used in instructions. For each of the destination word that is to be masked, or not copied, a 0 is used. For each of the destination word that is to be copied, a **1** is used. A MVM instruction program is shown in Figure 15-102 and summarized as follows:

- Source, Mask, and Dest must be entered for the instruction.
- The Source of data, and the Mask can be an immediate value or a tag but the destination address must be a tag name.
- In this example the values for the Source and Des are shown in binary.
- The Mask value is shown in Hex (16#F034).
- When the MVM instruction is executed the bit pattern from Local:1:I.Data is copied to memory location Local:2:O.Data through the filter mask.
- Bit positions 2,4,5 and 12 through 15 are copied to the Dest because a 1 (pass) for each bit position exists in the mask.
- Bit positions 0, 1, 3 and 6 through 11 of Local:1:I.Data did not get copied because a 0 (block) for each bit position was set in the mask. Bit position 0,1, 3, 6 through 11 of the Dest do not change and remain at their previous value before the instruction was executed.

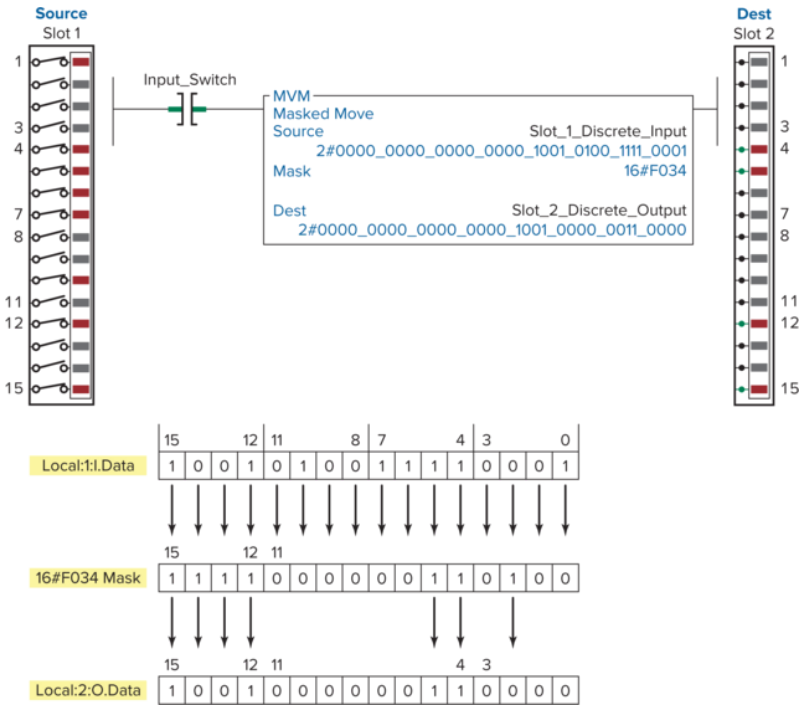


Figure 15-102 Masked Move instruction program.

The **Bit Field Distribute (BTD)** instruction is a very flexible move instruction. In addition to copying data from one memory location to another it allows you to shift the bits in that destination location if needed. An example of the BTD move instruction which shifts bits is shown in **Figure 15-103**. The step-up and execution of the instruction is summarized as follows:

- The enabled bit pattern from Local:1:I.Data, bits 8 through 15 are copied to location Local:2:O.Data, bits 0 through 7, when the Input_Switch is true.
- The Source Bit (8) defines where the copy will start.
- Starting from the Source Bit, the eight higher value bits will be copied as defined by the length (8).
- The eight bits will be copied to destination Local:2:O.Data starting at bit 0 as defined by the Dest Bit (0).
- Bits 0 through 7 of Local:1:I.Data do not get copied.
- Bits 8 through 15 of the destination do not change and remain at their previous value before the BTD instruction was executed.
- The destination address must be a tag name. The source bit (0-31), destination bit (0-31), and length (1-32) variables must be immediate values.

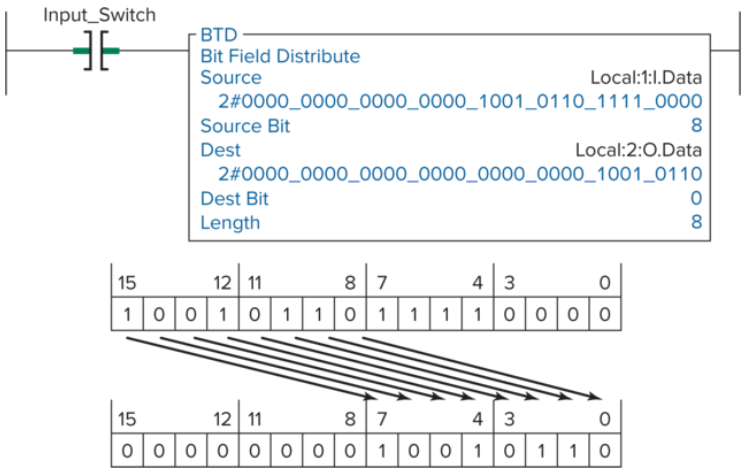


Figure 15-103 Bit Field Distribute instruction program.

PART 5 REVIEW QUESTIONS

1. Construct a ControlLogix ladder rung with a math instruction that executes when a toggle switch is closed to add the tag named Pressure_A (value 680) to the constant of 50 and store the answer in the tag named Result.
2. Construct a ControlLogix ladder rung with a math instruction that executes when two normally open limit switches are closed to subtract the tag named Count_1 (value 60) from the tag named Count_2 (value 460) and store the answer in the tag named Count_Total.
3. Construct a ControlLogix ladder rung with a math instruction that executes when either one of two normally open pushbuttons is closed to multiply the tag named Cases (value 10) by the constant 24 and store the answer in the tag named Cans.
4. Construct a ControlLogix ladder rung with a compare instruction that will energize a pilot light output anytime the value stored at Data_3 is 60.
5. Construct a ControlLogix ladder rung with a compare instruction that will energize a pilot light output anytime the value stored at Data_2 is not the same as that stored at Data_6.
6. Construct a ControlLogix ladder rung with compare instructions that will energize a pilot light output anytime the pressure of a system goes above 300 psi or below 100 psi.
7. Explain how data is transferred using the Move instruction.
8. Explain how data can be transferred using the Masked Move instruction.
9. Explain how data can be transferred using the Bit Field Distribute instruction.



PART 5 PROBLEMS

1. While checking the operation of the parts tracking system (Figure 15-86) with the Monitor Tags window, you note that the value of Conveyor_1_Sensor remains at 1 at all times with parts passing by. What can you surmise from this? Why?
2. Three conveyors are delivering the same parts in different packages. A package can hold 12, 24, or 18 parts. Proximity switches installed on each of the conveyor lines are used to advance the accumulated value of the three counters. Write a ControlLogix program that uses multiply and add instructions to calculate the sum of the parts.
3. A single pole switch is used in place of the two pushbuttons for the variable preset timer program (Figure 15-95). When this switch is closed the timer is to be set to 10 seconds and when open to 15 seconds. Make the necessary changes to the program.

OceanofPDF.com

Part 6 Function Block Programming

Part Objectives

After completing this part, you will be able to:

- Describe the difference between ladder logic and function block diagram programming
- Recognize the basic elements of a function block diagram
- Write and read a function block diagram

Function Block Diagram (FBD)

A ***function block diagram (FBD)*** is a graphical depiction of process flow using simple and complex interconnecting blocks. It is similar to a ladder logic diagram, except that function blocks replace the interconnection of contacts and the coils. In addition, there are no power rails.

A function block circuit is analogous to an electrical circuit where links and wires depict signal paths between components. The workplace is known as a sheet and consists of function blocks joined together with lines called wires. The structure of a function block program, or routine, is shown in **Figure 15-104**. A function block diagram consists of four basic elements: function block, references, wire connectors, and wires. Data flows on a wire from wire connectors or input references, through the function block, and then is passed on to an output reference. The line type of the link between function blocks indicates what type of data is present. A dash line indicates a Boolean signal path (e.g., 0 or 1) and a solid line indicates an integer or real value.

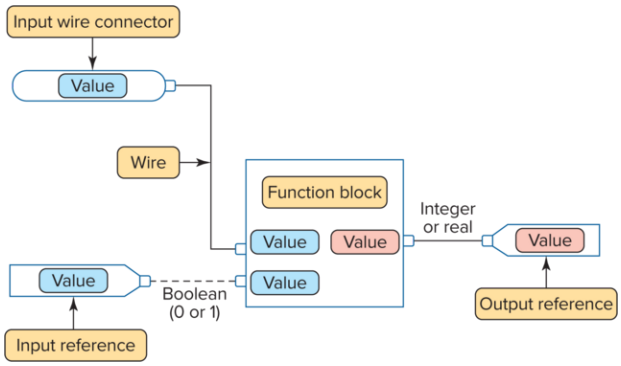


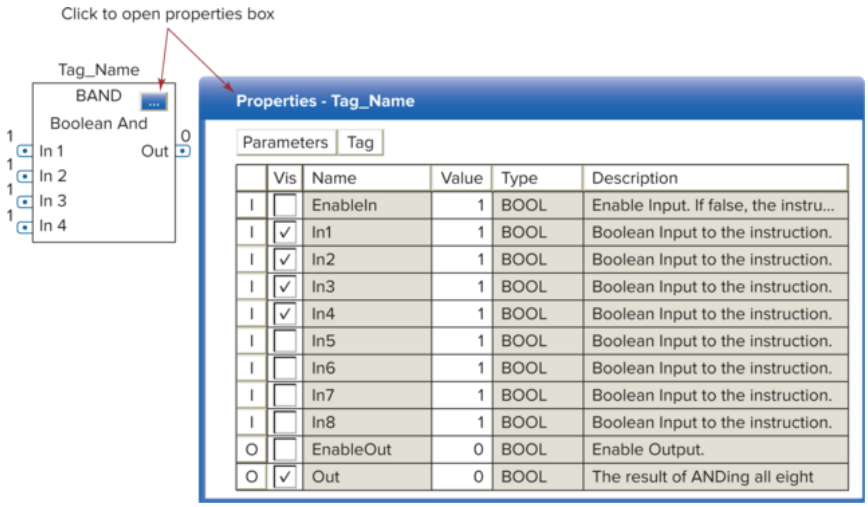
Figure 15-104 Structure of function block or routine.

Function blocks are graphical representations of executable code. A function block can take one or more inputs and make decisions or calculations and then generate one or more outputs. There are many different types of function blocks included in the programming software to perform various common tasks. In addition, customized ***Add-On*** instructions can be created by the programmer for sets of commonly used logic. Once an Add-On instruction is defined in a project, it appears on the instruction toolbar and behaves like the standard instructions.

Figure 15-105 shows an example of a BAND (Boolean AND) function block. The information associated with a function block can be summarized as follows:

- Inputs are shown entering from the left and outputs exiting on the right.
- The function block type is shown within the block.
- A tag name for the block is placed above it.
- The names of the inputs and outputs are shown within the block.

- The default view of the block has some but not all of the input and output parameters visible when the box is placed into the program.
- The properties box, used to set the option of input and output parameters, is displayed by clicking the selection button located at the upper right hand corner of the block.
- The 1 and 0 next to the inputs and outputs identify the logical state of the input and output pins for the instruction.
- The dots on the input and output pins indicate BOOL type data is required.



	Vis	Name	Value	Type	Description
I	☐	EnableIn	1	BOOL	Enable Input. If false, the instru...
I	☑	In1	1	BOOL	Boolean Input to the instruction.
I	☑	In2	1	BOOL	Boolean Input to the instruction.
I	☑	In3	1	BOOL	Boolean Input to the instruction.
I	☑	In4	1	BOOL	Boolean Input to the instruction.
I	☐	In5	1	BOOL	Boolean Input to the instruction.
I	☐	In6	1	BOOL	Boolean Input to the instruction.
I	☐	In7	1	BOOL	Boolean Input to the instruction.
I	☐	In8	1	BOOL	Boolean Input to the instruction.
O	☐	EnableOut	0	BOOL	Enable Output.
O	☑	Out	0	BOOL	The result of ANDing all eight

Figure 15-105 Example of a BAND (Boolean AND) function block.

Source: Image Courtesy of Rockwell Automation, Inc.

References represent tags that are linked to values stored in a controller's memory. The two types of references, input and output, are illustrated in **Figure 15-106**. An input reference, or IREF, is used to receive a value from an input device or tag. An output reference, or OREF, is used to send a value to an output device or tag. When you use an IREF or an OREF you must create a tag or assign an existing tag to the element. You may use any of the data types for an IREF or OREF.

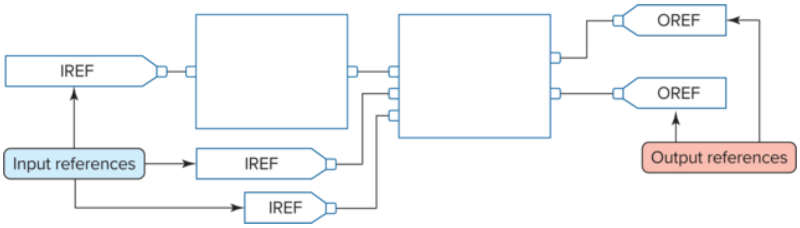


Figure 15-106 Input and output references.

Function blocks can be connected to other function blocks by connecting their outputs to the input of another function block using ***wires and pins*** (Figure 15-107). Wires map a signal's path and show the flow of controller execution. Each element in a function block diagram contains pins. Elements are connected by moving wires from input pins to output pins or vice versa. The pins on the left of a function block are input pins, and those on the right are output pins. To wire two elements together, click the output pin of the first element (A) and then click the input pin of the other element (B). A green dot shows a valid connection point.

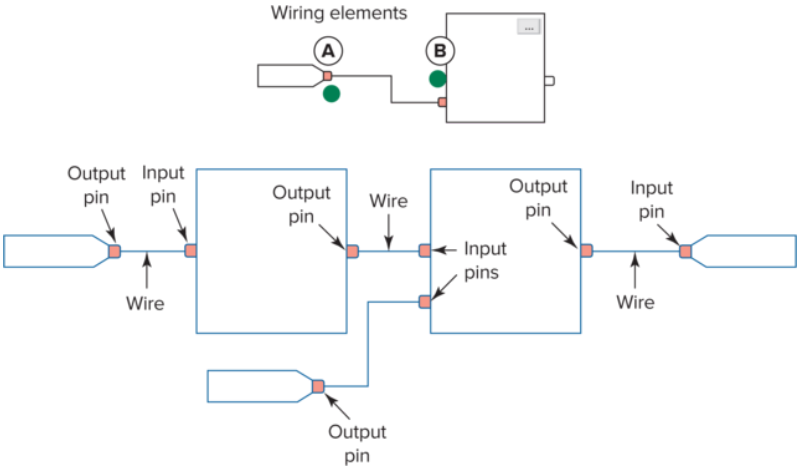


Figure 15-107 Function block diagram wire and pins.

Wire connectors are used to create a path without using a wire. When there are many function blocks on a sheet, or the function blocks are far apart, wire connectors used in place of wires can make the logic easier to read. Wire connectors are also used to connect function blocks that are on a different sheet of the same function block routine, as illustrated in Figure 15-108. The use of wire connectors can be summarized as follows:

- An output wire connector, or **OCON**, sends a value or signal to an input wire connector, or **ICON**.
- Each output wire connector must have at least one corresponding input wire connector.
- Each output wire connector requires a unique tag name and the corresponding input connector must have the same name.
- Multiple input wire connectors can reference the same output wire connector. This lets you share data at several points in your function block diagram.

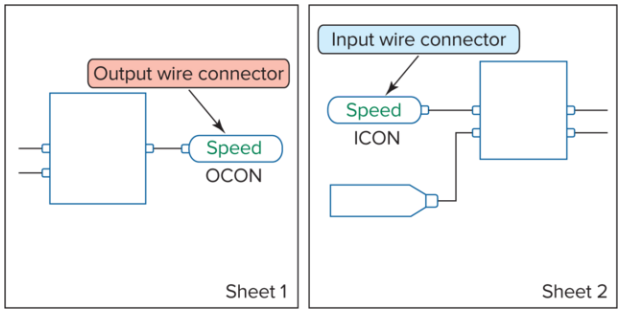


Figure 15-108 OCON and ICON wire connectors.

Figure 15-109 illustrates the signal flow and execution of an FBD program. The operation can be summarized as follows:

- Each program scan sets all the FBD blocks starting on the left side of the signal flow and continues to evaluate all blocks according to the signal flow until the final output is determined.
- The location of a block does not affect the order in which the blocks execute.

- The inputs of a block require data to be available before the controller can execute that block.
- If function blocks are not wired together, it does not matter which block executes first as there is no data flow between the blocks.
- The interconnected line between the blocks indicates what type of signal is present.

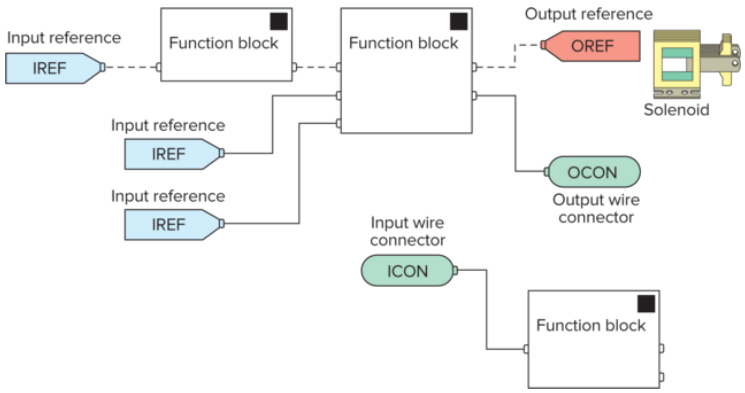


Figure 15-109 Signal flow and execution of an FBD program.

Data latching refers to how the controller verifies that the data present at the input to a function block are valid. If you use an IREF to specify input data for a function block instruction, as illustrated in **Figure 15-110**, the data in that IREF are latched (won't change) for the scan of the function block routine. The IREF latches data from program-scoped and controller-scoped tags. The controller updates all IREF data at the beginning of each scan. A function block routine executes in the following order:

- The controller latches all data values in IREFs.
- The controller executes the other function blocks in order.
- The controller writes outputs in OREFs.

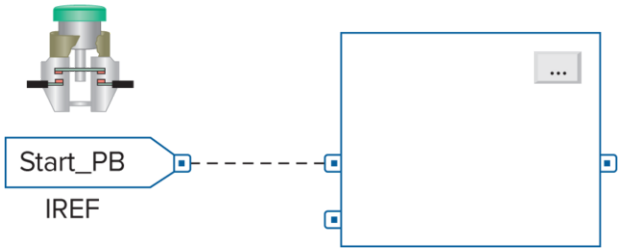


Figure 15-110 IREF is latched for the scan of the function block routine.

When you add a Function Block instruction, the block appears with a set of pins for the default parameters, as illustrated in **Figure 15-111**. The rest of the pins are hidden. You can hide or show a pin by:

- Clicking on the Parameters tab in the Properties dialog box.
- In the Properties dialog box, on the Parameters tab, clear the Vis check box to hide the pin.
- Select the Vis check box to show the pin.
- Click OK.

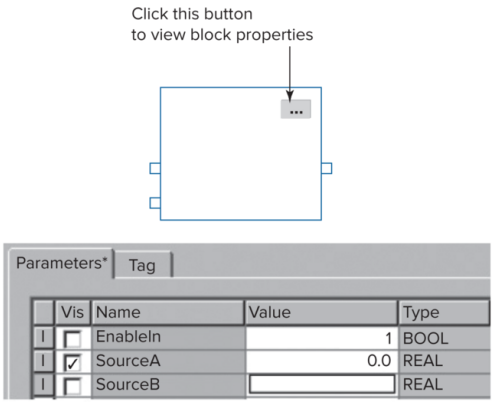


Figure 15-111 Using the Parameters tab to show or hide a pin.

Source: Image Courtesy of Rockwell Automation, Inc.

To create a ***feedback loop*** around a block, wire an output pin of the block to an input pin of the same block. The input pin will receive the value of the output that was produced on the last scan of the function block. The loop contains only a single block, so execution order does not matter.

Figure 15-112 shows an example of a feedback loop used to reset an on-delay timer. When the timer finishes timing its DN bit is used to reset the timer.

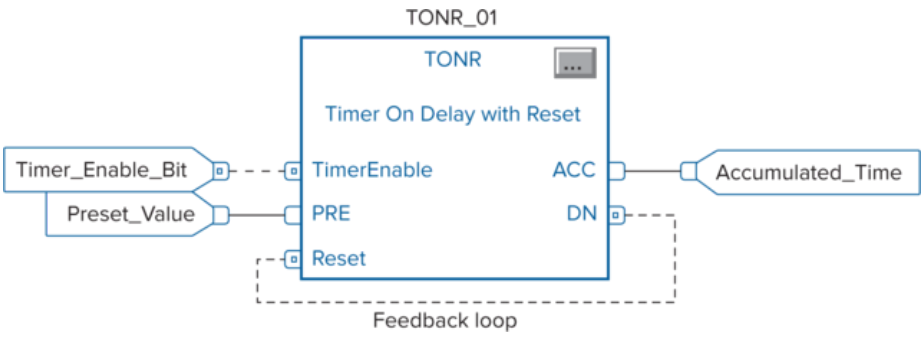


Figure 15-112 Feedback loop used to reset an on-delay timer.

When a group of function blocks are in a feedback loop, the controller cannot determine which block to execute first. This problem is resolved by placing an **Assume Data Available** indicator mark at the input pin of the function block that should be executed first. In the example shown in **Figure 15-113**, the input for block 1 uses the data from block 3 that were produced in the previous scan. To place the indicator, click on the interconnecting wire and select the **Assume Data Available** choice.

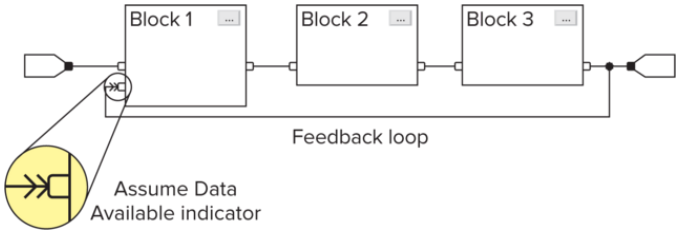


Figure 15-113 Assume Data Available indicator marker.

FBD Programming

Figure 15-114 illustrates the setup procedure for FBD programming. The steps to be followed can be summarized as follows:

- Right click on the MainProgram file and select New Routine from the pop-up menu.
- Select the Function Block diagram entry from the Type window.
- Enter a name for the Routine (e.g., FDB_Sample).
- You will now see the new program (FDB_Sample) listed under MainProgram.
- Left clicking the FBD_Sample twice opens the graphic development window.
- FBD instructions selected from the Language Element toolbar are used in the development of the program.
- Extra sheets can be added when the current sheet is full by clicking the add sheet icon. Movement between sheets is provided by left and right arrows.

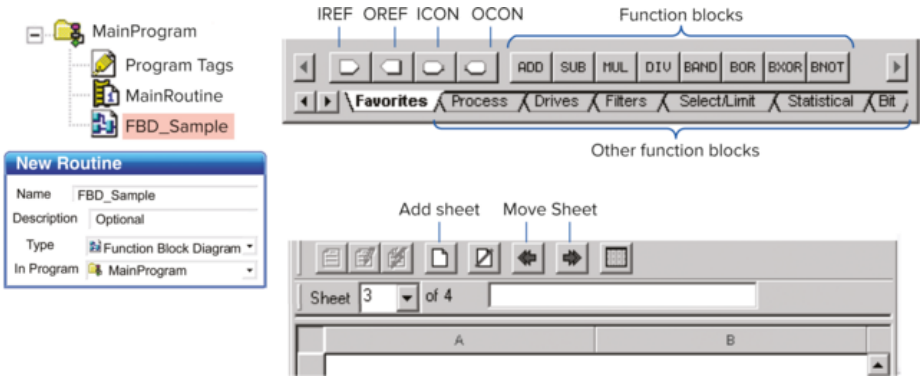


Figure 15-114 Setup procedure for FDB programming.

Source: Image Courtesy of Rockwell Automation, Inc.

The MainRoutine is always a ladder logic program in RSLogix 5000 software, and all other routines are called from the MainRoutine. Therefore, the MainRoutine will have one unconditional rung with a jump to subroutine (JSR) calling FBD_Sample. The FBD program will execute from the JSR instruction. No subroutine or return subroutine instruction in the FBD is necessary.

Function block programs are similar to ladder logic programs, except that the process is visualized in the form of function blocks instead of ladder rungs. **Figure 15-115** shows a comparison between ladder logic and the FBD equivalent for a three-input AND ladder logic rung. The operation of the FBD can be summarized as follows:

- When the inputs represented by Sensor_1, Sensor_2, and Sensor_3 are true (value 1) the BAND (Boolean AND) function block will be true.
- The BAND block executes to set output Caution_PL true and switch the pilot light on.
- The 0 to the right of the input reference and out pin indicates its logic state. A 0 indicates the state of the tag is false, while a 1 signifies it is true.
- The same field input sensors and output pilot light devices and tags can be used with either program.
- The XIC and OTE contact and coil instructions have been replaced by the BAND function block.

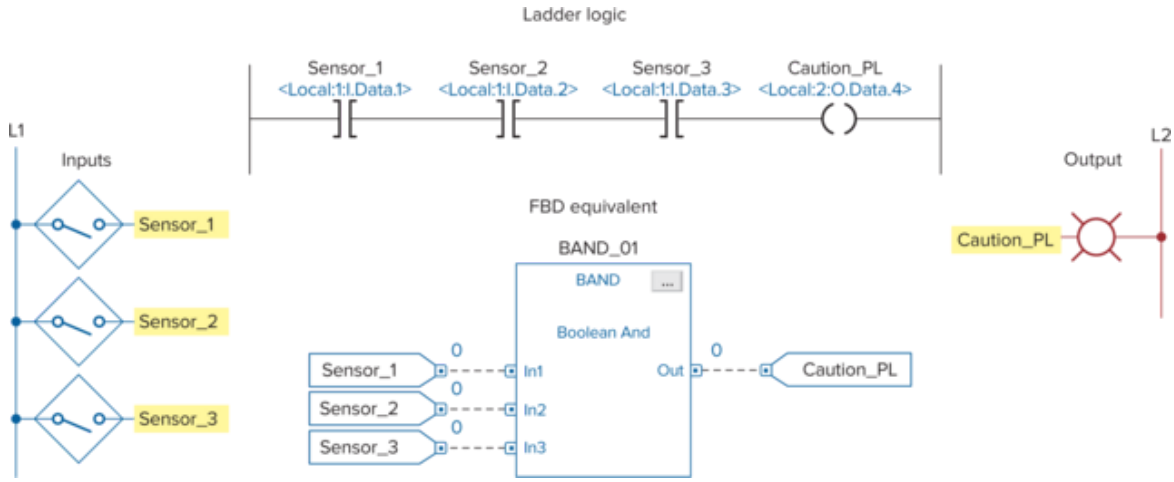


Figure 15-115 Comparison between ladder logic and the FBD equivalent for a three-input AND ladder logic rung.

Figure 15-116 shows a comparison between ladder logic and the FBD equivalent for a two-input OR ladder logic rung. As with ladder OR logic, if any of the two inputs is true the BOR function block will be true. In this example, with the BOR function block true, the output reference tag SOL_1 will be true, energizing the solenoid.

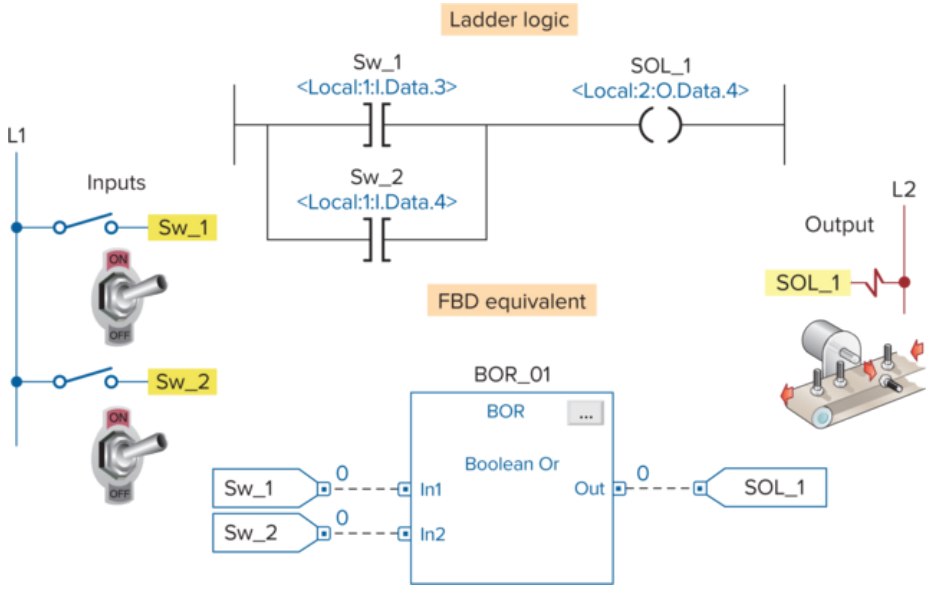


Figure 15-116 Comparison between ladder logic and the FBD equivalent for a two-input OR ladder logic rung.

Figure 15-117 shows a comparison between ladder logic and the FBD equivalent for a combination of multiple inputs. The operation of the FBD can be summarized as follows:

- The alarm will be energized if either input In1 or In2 to the BOR block is true.
- Input In2 of the BOR block will be true only when all three of the sensor switches are closed.
- Input In1 of the BOR block will be true only when the Temp_Sw is closed at the same time as the Press_Sw is open.
- The BNOT function block executes similarly to an XIO ladder logic contact instruction. When In is 0, Out is 1 and vice versa.

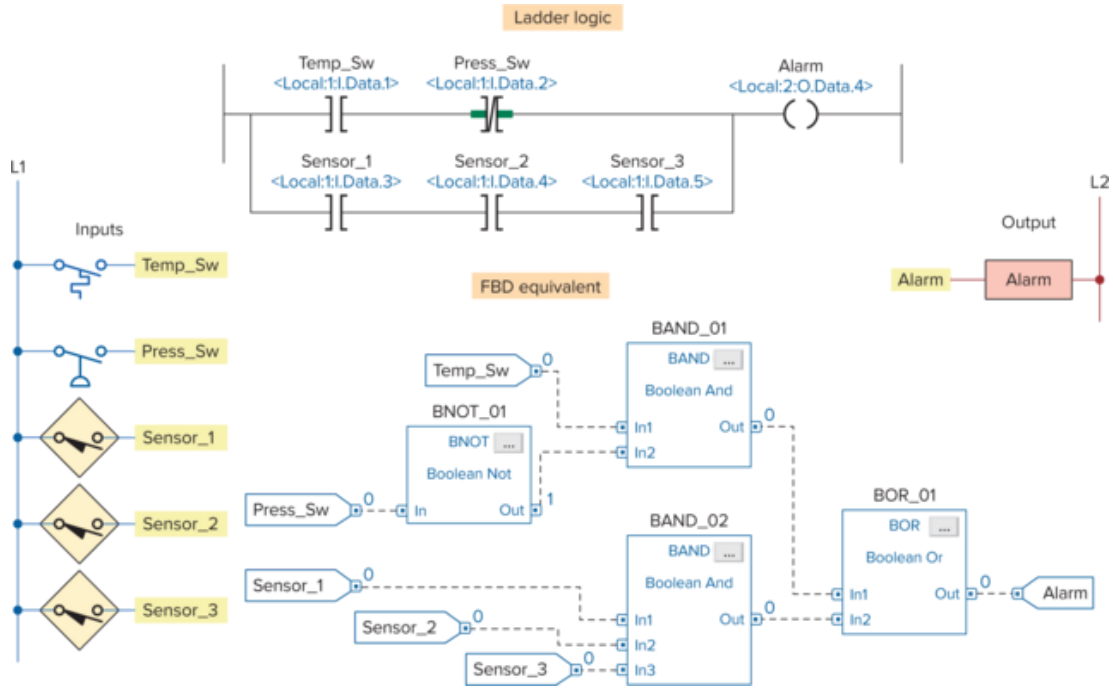


Figure 15-117 Comparison between ladder logic and the FBD equivalent for a combination of multiple inputs.

Figure 15-118 shows a comparison between ladder logic and the FBD equivalent for the motor start/stop control circuit. The logic sequence for starting and stopping the motor can be summarized as follows:

- When Motor_Start button is closed the BOR output will become true making the BAND output true. Page 387
- Motor_Run output energizes the contactor coil, the contacts of which close to start the motor operating.
- When the Motor_Start button is then opened the output of the BOR block remains true due to the 1 status of the feedback signal from the Motor_Run tag. Page 388
- When the Motor_Stop button is opened the output of the BAND block turns false to de-energize the contactor coil and stop the motor.

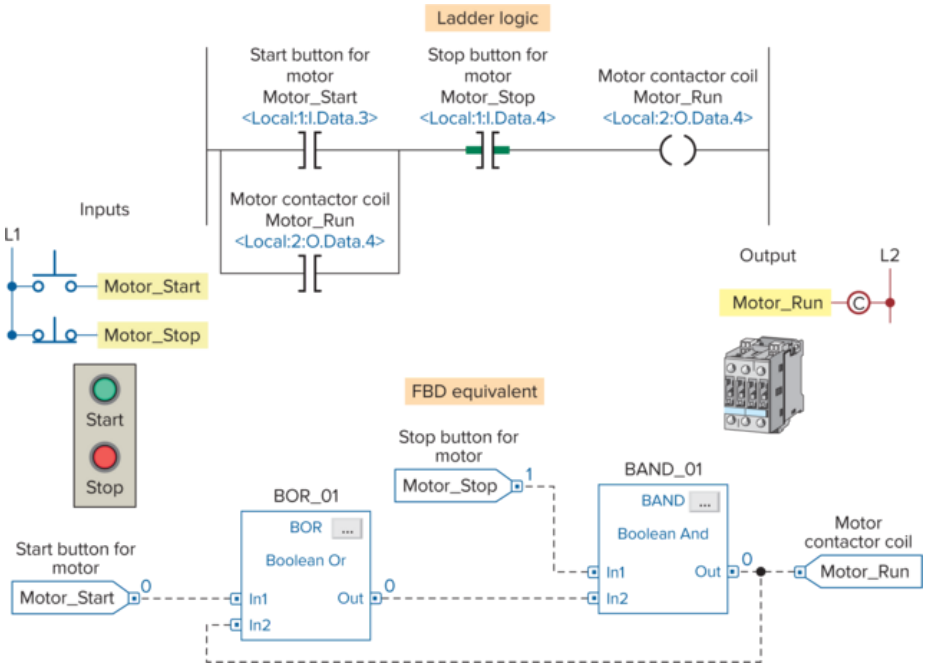


Figure 15-118 Comparison between ladder logic and the FBD equivalent for a motor start/stop control circuit.

Figure 15-119 shows a comparison between ladder logic and the FBD equivalent for the 10 second TON (on-delay timer) and TONR (on-delay with reset). The operation of the FBD can be summarized as follows:

- When the Timer_Sw is closed, the TONR function block timer turns true and starts accumulating time.
- The accumulated time is monitored by the output reference tag named ACC_Value.
- The EN (enable bit) output changes to 1 to turn on the EN_PL.
- The TT (timer timing bit) output changes to 1 to turn on the TT_PL.
- The timer times out after 10 seconds to set the DN (done bit) to 1 and turn on the DN_PL and reset the TT bit to zero and turn off the TT_PL.
- The EN bit and EN_PL remain on as long as the Timer_Sw stays toggled closed.
- Opening the Timer_Sw resets all outputs as well as the accumulated value to zero.
- The timer can also be reset by way of the Reset input.

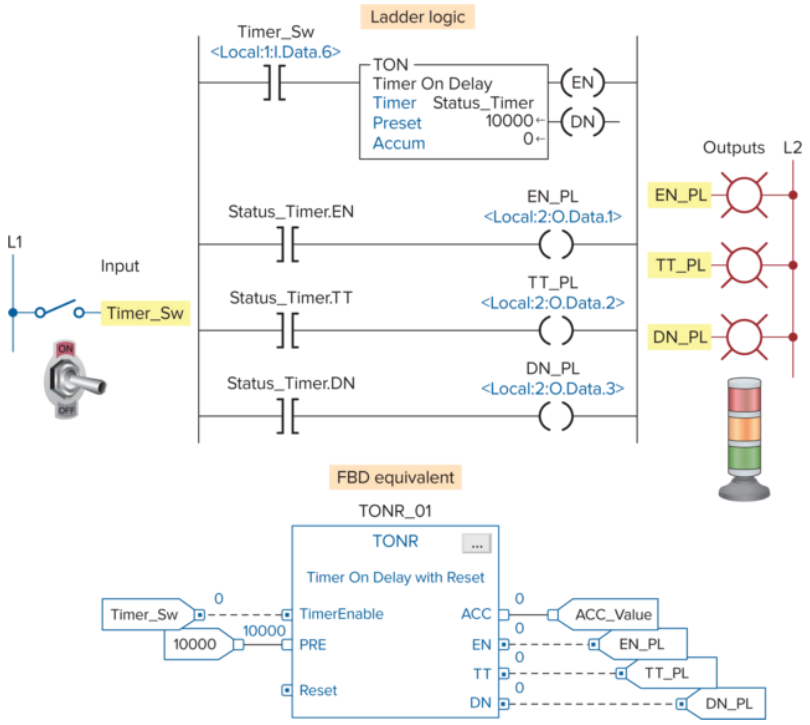


Figure 15-119 Comparison between ladder logic and the FBD equivalent for a 10 second TON and TONR timer.

Figure 15-120 shows a comparison between ladder logic and the FBD equivalent for the Up/Down counter used to limit the number of parts stored in a buffer zone to 50. The operation of the FBD can be summarized as follows:

- The CTUD up/down counter function block accumulated value is initially reset by momentary actuation of the Restart_Button.
- The accumulated count is monitored by the output reference tag named ACC.
- Each time a part enters the buffer zone, the Enter_Limit_Sw is actuated and the CUEnable input turns true to increment the count by 1.
- Each time a part exits the buffer zone, the Exit_Limit_Sw is actuated and the CDEnable input turns true to decrement the count by 1.
- Whenever the number of parts in the buffer zone reaches 50 the DN bit is set to 1 and the output of the BNOT block is reset to zero. This de-energizes the Conveyor_Contactor to stop the conveyor motor from delivering more parts to the buffer zone.

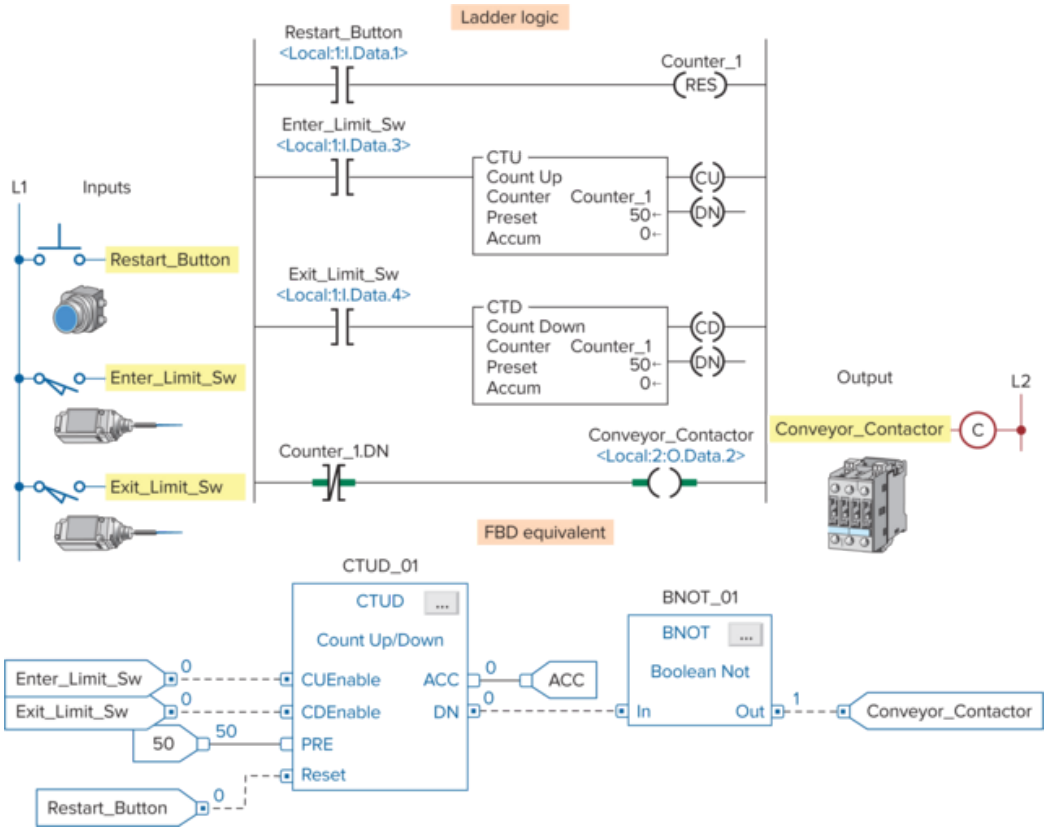


Figure 15-120 Comparison between ladder logic and the FBD equivalent for an Up/Down counter application.

Figure 15-121 shows a comparison between ladder logic and the FBD equivalent for the program used to test the accumulated value of a counter. The operation of the FBD can be summarized as follows:

- The function block routine is broken into four sheets.
- The order of the sheets does not affect the order in which the function blocks execute.
- When a function block routine executes, all sheets execute.
- Using one sheet for each device that is to be programmed helps organize your program and make it easier to understand.
- The use of the OCON and ICON named ACC enables the function blocks to be on different sheets of the same function block routine.
- The numbers and letters under the ACC output indicate the sheet number and location on the sheet where the output is used.

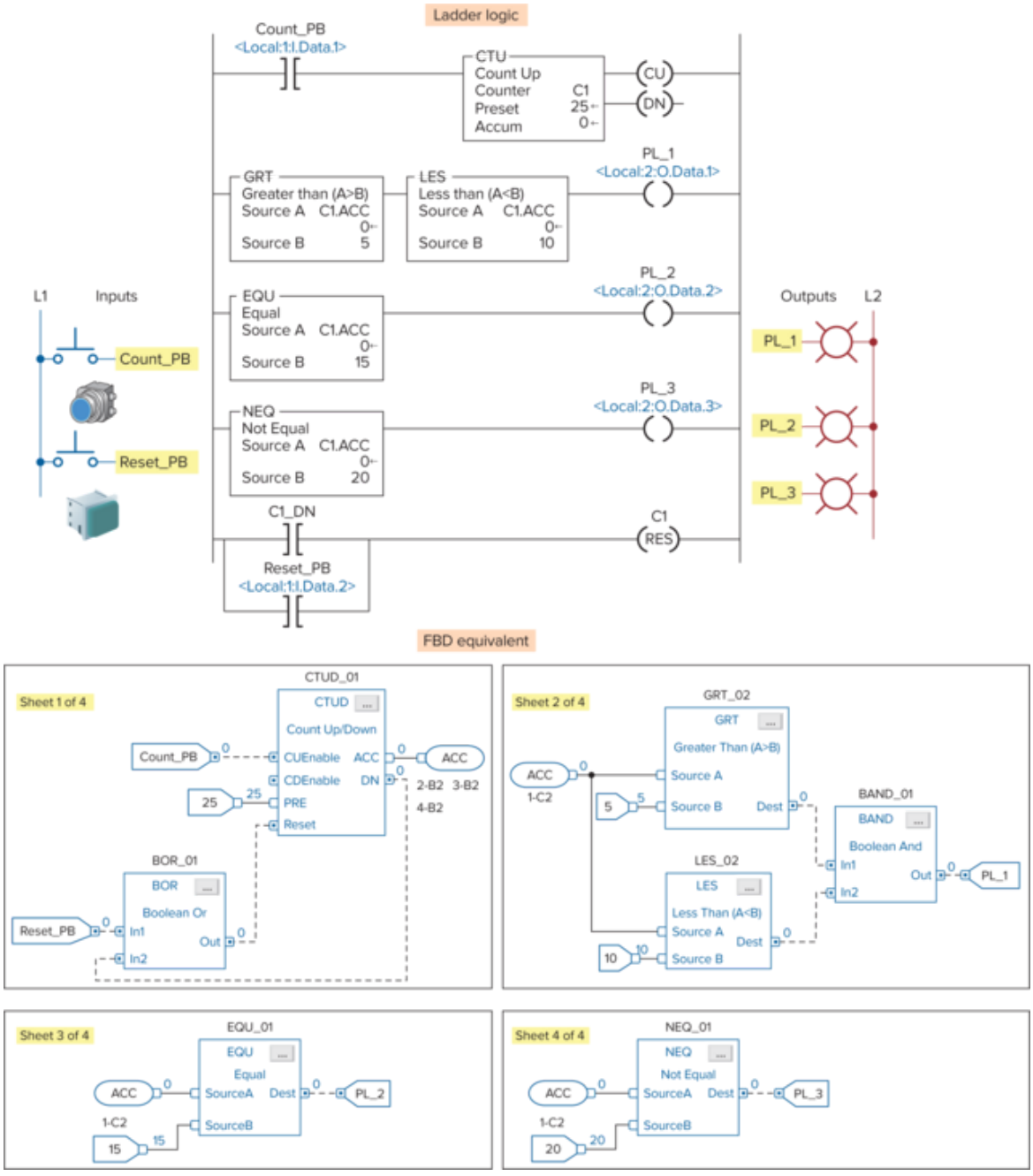


Figure 15-121 Comparison between ladder logic and the FBD equivalent for a program used to test the accumulated value of a counter.

PART 6 REVIEW QUESTIONS

1. Compare the graphical representation of a function block diagram to that of a logic ladder diagram.
2. Name the four basic elements of an FBD.
3. What do the solid and dashed interconnecting lines between FBD function blocks indicate?
4. What is an Add-On instruction?
5. How are the input and output parameter options for a function block set?
6. What does the dot on an input or output pin of a function block indicate?
7. Compare the functions of input and output reference tags.
8. Which pins of a function block are inputs and which are outputs?
9. Explain the role of input and output wire connectors.
10. How does the program scan function for an FBD program?
11. Explain data latching as it applies to function block inputs.
12. How is a function block feedback loop created?
13. What is the Assume Data Available indicator used for?
14. Outline how an FBD program is initiated.



PART 6 PROBLEMS

1. Write an FBD program that will cause the output, solenoid SOL_1, to be energized when pushbutton PB_1 is open and PB_2 is closed, and either limit switch LS_1 is open or limit switch LS_2 is closed. Assume all pushbuttons and limit switches are of the normally open type.
2. Modify the motor start/stop FBD program (**Figure 15-118**) to include a second start/stop pushbutton station.
3. You are required to change the on-delay time of the 10 second timer program (**Figure 15-119**) to 1 minute. What changes would have to be made to the FBD program?
4. Modify the Up/Down counter FBD program (**Figure 15-120**) to include the following pilot lights:
   - PL_1 to come on when a part enters
   - PL_2 to come on when a part exits
   - PL_3 to come on when the buffer zone is full
5. Modify the test accumulated value of a counter FBD program (**Figure 15-121**) as follows:
   - PL_1 to be on for an accumulated count between 0 and 5
   - PL_2 to be on for an accumulated count of 12
   - PL_3 to be on at all times except for when the accumulated count is 15.

OceanofPDF.com

Glossary

1's complement The system used to represent negative numbers in a personal computer and a programmable logic controller.

2's complement A numbering system used to express positive and negative binary numbers.

A

Access To locate data stored in a programmable logic controller system or in computer-related equipment.

Accumulated value The number of elapsed timed intervals or counted events.

Actuator An output device normally connected to an output module. Examples are an air valve and cylinder.

Address A code that indicates the location of data to be used by a program, or the location of additional program instructions.

Algorithm Mathematical procedure for problem solving.

Alias tag References a memory location that has been defined by another tag.

Alphanumeric Term describing character strings consisting of any combination of letters, numerals, and/or special characters (e.g., A15$) used for representing text, commands, numbers, and/or code groups.

Alternating current (AC) input module An input module that converts various alternating current signals originating at user devices to the appropriate logic-level signal for use within the processor.

Alternating current (AC) output module An output module that converts the logic-level signal of the processor to a usable output signal to control a user alternating current device.

Ambient temperature The temperature of the air surrounding a module or system.

American National Standard Code for Information Interchange (ASCII) An 8-bit (7 bits plus parity) code that represents all characters of a standard typewriter keyboard, both uppercase and lowercase, as well as a group of special characters used for control purposes.

American National Standards Institute (ANSI) A clearinghouse and coordinating agency for voluntary standards in the United States.

American wire gauge (AWG) A standard system used for designating the size of electrical conductors. Gauge numbers have an inverse relationship to size; larger numbers have a smaller diameter.

Analog device Apparatus that measures continuous information (e.g., voltage or current). The measured analog signal has an infinite number of possible values. The only limitation on resolution is

the accuracy of the measuring device.

Analog input module An input circuit that employs an analog-to-digital converter to convert an analog value, measured by an analog measuring device, to a digital value that can be used by the processor.

Analog output module An output circuit that employs a digital-to-analog converter to convert a digital value, sent from the processor, to an analog value that will control a connected analog device.

Analog signal Signal having the characteristic of being continuous and changing smoothly over a given range rather than switching suddenly between certain levels, as with discrete signals.

Analog-to-digital (A/D) converter A circuit for converting a varying analog signal to a corresponding representative binary number.

AND (logic) A Boolean operation that yields a logic 1 output if all inputs are 1, and a logic 0 if any input is 0.

Arithmetic capability The ability to do addition, subtraction, multiplication, division, and other advanced math functions with the processor.

Array A group of data, of the same type, under a common name.

ASCII-input module Converts ASCII-code input information from an external peripheral into alphanumeric information a PLC can understand.

ASCII-output module Converts alphanumeric information from the PLC into ASCII code to be sent to an external peripheral.

Asynchronous Recurrent or repeated operations that occur in unrelated patterns over time.

Automatic control A process in which the output is kept at a desired level by using feedback from the output to control the input.

Auxiliary power supply A power supply not associated with the processor. Auxiliary power supplies are usually required to supply logic power to input/output racks and to other processor support hardware and are often referred to as *remote power supplies.*

B

Backplane A printed circuit board, located in the back of a chassis, that contains a data bus, power bus, and mating connectors for modules to be inserted in the chassis.

Base tag A definition of the memory location at which a data element is stored.

BASIC A computer language that uses brief English-language statements to instruct a computer or microprocessor.

Battery indicator A diagnostic aid that provides a visual indication to the user and/or an internal processor software indication that the memory power-fail support battery is in need of replacement.

Baud A unit of signaling speed equal to the number of discrete conditions or signal events per second; often defined as the number of binary digits transmitted per second.

BCD-input module Allows the processor to accept 4-bit BCD digital codes.

BCD-output module Enables a PLC to operate devices that require BCD-coded signals to operate.

Binary A number system using 2 as a base. The binary number system requires only two digits, zero (0) and one (1), to express any alphanumeric quantity desired by the user.

Binary-coded decimal (BCD) A system of numbering that expresses each individual decimal digit (0–9) of a number as a series of 4-bit binary notations. The binary-coded decimal system is often referred to as *8421 code.*

Binary word A related group of 1s and 0s that has meaning assigned by position or by numerical value in the binary system of numbers.

Bit An abbreviated term for the words *binary digit.* The bit is the smallest unit of information in the binary numbering system. It represents a decision between one of two possible and equally likely values or states. It is often used to represent an off or on state as well as a true or false condition.

Bit manipulation instructions A family of programmable logic controller instructions that exchange, alter, move, or otherwise modify the individual bits or groups of processor data memory words.

Bit storage A user-defined data table area in which bits can be set or reset without directly affecting or controlling output devices. However, any storage bit can be monitored as necessary in the user program.

Block diagram A method of representing the major functional subdivisions, conditions, or operations of an overall system, function, or operation.

Block format Format that uses a box shape to display instructions.

Block transfer An instruction that copies the contents of one or more contiguous data memory words to a second contiguous data memory location; an instruction that transfers data between an intelligent input/output module or card and specified processor data memory locations.

BOOL A data type that stores the state of a single bit, where 0 equals off and 1 equals on.

Boolean algebra A mathematical shorthand notation that expresses logic functions, such as AND, OR, EXCLUSIVE OR, NAND, NOR, and NOT.

Branch A parallel logic path within a rung.

Buffer In software terms, a register or group of registers used for temporary storage of data; a buffer is used to compensate for transmission rate differences between the transmitter and receiving device. In hardware terms, a buffer is an isolating circuit used to avoid the reaction of one circuit with another.

Bug A system defect or error that causes a malfunction; can be caused by either software or hardware.

Burn The process by which information is entered into programmable read-only memory.

Bus A group of lines used for data transmission or control; power distribution conductors.

Bus topology A network configuration in which all stations are connected in parallel with the communication medium and all stations can receive information from any other station on the

network.

Byte A group of adjacent bits usually operated on as a unit, such as when moving data to and from memory. There are 8 bits per byte.

C

Cascading In the programming of timers and counters, a technique used to extend the timing or counting range beyond what would normally be available. This technique involves the driving of one timer or counter instruction from the output of another, similar instruction.

Cell controller A specialized computer used to control a work cell through multiple paths to the various cell devices.

Central processing unit (CPU) The electronic circuitry that controls all the data activity of the PLC, performs calculations, and makes decisions with its operation controlled by a sequence of instructions. The central processing unit is also referred to as the *processor* or the *CPU.*

Channel A designated path for a signal.

Character A symbol that is one of a larger group of similar symbols and that is used to represent information on a display device. The letters of the alphabet and the decimal numbers are examples of characters used to convey information.

Chassis A rack that serves as an electrical backplane for a PLC processor and I/O modules.

Chip A tiny piece of semiconductor material on which electronic components are formed. Chips are normally made of silicon and are typically less than 1/4 in. square and 1/100 in. thick.

Clear An instruction or a sequence of instructions that removes all current information from a programmable logic controller's memory.

Clock A circuit that generates timed pulses to synchronize the timing of computer operations.

Clock rate The speed at which the microprocessor system operates.

Closed loop A control system that uses feedback from the process to maintain outputs at a desired level.

Coaxial cable A transmission line constructed such that an outer conductor forms a cylinder around a central conductor. An insulating dielectric separates the inner and outer conductors, and the complete assembly is enclosed in a protective outer sheath. Coaxial cables are not susceptible to external electric and magnetic fields and generate no electric or magnetic fields of their own.

Code A system of communications that uses arbitrary groups of symbols to represent information or instructions. A set of programmed instructions.

Coil Represents the output of a programmable logic controller. In the output devices, it is the electrical coil that, when energized, changes the status of its corresponding contacts.

Coil format Format that uses coils to display instructions.

Comment Text that is included with each PLC ladder rung and is used to help individuals understand how the program operates or how the rung interacts with the rest of the program.

Communication module Allows the user to connect the PLC to high-speed local networks that may differ from the network communication provided with the PLC.

Compare An instruction that compares the contents of two designated data memory locations of a programmable logic controller for equality or inequality.

Compatibility The ability of various specified units to replace one another with little or no reduction in capability; the ability of units to be interconnected and used without modification.

Complement A logical operation that inverts a signal or bit.

Complementary metal-oxide semiconductor (CMOS) A logic base that offers lower power consumption and high-speed operation.

Computer Any electronic device that can accept information, manipulate it according to a set of preprogrammed instructions, and supply the results of the manipulation.

Computer integrated manufacturing (CIM) A manufacturing system controlled by an easily reprogrammable computer for flexibility and speed of changeover.

Computer interface A device designed for data communication between a programmable logic controller and a computer.

Consumed tag References data that come from another controller.

Contact The current-carrying part of an electric relay or switch. The contact engages to permit power flow and disengages to interrupt power flow to a load device.

Contact bounce The uncontrollable making and breaking of a contact during the initial engaging or disengaging of the contact.

Contact histogram An instruction sequence that monitors a designated memory bit or a designated input or output point for a change of state. A listing is generated by the instruction sequence that displays how quickly the monitored point is changing state.

Contactor A special-purpose relay designed to establish and interrupt the power flow of high-current electric circuits.

Contact symbology A set of symbols used to express the control program with conventional relay symbols.

Continuous current per module The maximum current for each module. The sum of the output current for each point should not exceed this value.

Continuous current per point The maximum current each output is designed to supply continuously to a load.

Controlled variable The output variable that the automatic control adjusts to keep the process value at a set-point.

Control logic The control plan for a given system; the program.

Control loop The method of adjusting the control variable in a process control system by analyzing the process variable data and then comparing it to the set-point to determine the amount of error in the system.

ControlNet An open, high-speed, deterministic network that transfers on the same network time-critical I/O updates, controller-to-controller interlocking data, and non-time-critical data such as data monitoring and program uploads and downloads.

Control relay A relay used to control the operation of an event or a sequence of events.

Counter An electromechanical device in relay-based control systems that counts numbers of events for the purpose of controlling other devices based on the current number of counts recorded; a programmable logic controller instruction that performs the functions of its electromechanical counterpart.

Cross reference In ladder diagrams, letters or numbers to the right of coils or functions. The letters or numbers indicate where on other ladder lines contacts of the coil or function are located.

Crosstalk Undesired energy appearing in one signal path as a result of coupling from other signal paths or use of a common return line.

Current The rate of electrical electron movement, measured in amperes.

Current-carrying capacity The maximum amount of current a conductor can carry without heating beyond a predetermined safe limit.

Current sinking Refers to an output device (typically an NPN transistor) that allows current flow from the load through the output to ground.

Current sourcing Output device (typically a PNP transistor) that allows current flow from the output through the load and then to ground.

Cycle A sequence of operations repeated regularly; the time it takes for one such sequence to occur.

D

Data Information encoded in a digital form, which is stored in an assigned address of data memory for later use by the processor.

Data address A location in memory where data can be stored.

Data file A group of data memory words acted on as a group rather than singly.

Data highway A communications network that allows devices such as PLCs to communicate. They are normally proprietary, which means that only like devices of the same brand can communicate over the highway.

Data latching A technique used to read the value of the input data that will be operated on by the instructions with a function block.

Data link The equipment that makes up a data communications network.

Data manipulation The process of exchanging, altering, or moving data within a programmable logic controller or between programmable logic controllers.

Data manipulation instructions A classification of processor instructions that alter, exchange, move, or otherwise modify data memory words.

Data server Used to store, send, and receive data.

Data table The part of processor memory that contains input and output values as well as files where data are monitored, manipulated, and changed for control purposes.

Data transfer The process of moving information from one location to another, in other words, from register to register, from device to device, and so forth.

Data transmission line A medium for transferring signals over a distance.

Debouncing The act of removing intermediate noise states from a mechanical switch.

Debug The process of locating and removing mistakes from a software program or from hardware interconnections.

Decimal number system A number system that uses ten numeral digits (decimal digits): 0, 1, 2, 3, 4, 5, 6, 7, 8, 9. Each digit position has a place value of 1, 10, 100, 1000, and so on, beginning with the least significant (rightmost) digit; base 10.

Decrement The act of reducing the contents of a storage location or value in varying increments.

Determinism The ability to reliably predict when data will be delivered.

DeviceNet An open communication network designed to connect factory-floor devices together without interfacing through an I/O system. Up to 64 intelligent nodes can be connected to one DeviceNet network.

Diagnostic program A user program designed to help isolate hardware malfunctions in the programmable logic controller and the application equipment.

Diagnostics The detection and isolation of an error or malfunction.

Digital device One that processes discrete electric signals.

Digital signals A system of discrete states: high or low, on or off, 1 or 0.

Digital-to-analog converter An electrical circuit that converts binary bits to a representative, continuous, analog signal.

DINT A data type that stores a 32-bit (4-byte) signed integer value.

DIP switch A group of small, in-line on-off switches. From *d*ual *i*n-line *p*ackage.

Direct addressing An addressing mode in which the memory address of the data is supplied with the instruction.

Discrete I/O A group of input and/or output modules that operate with on/off signals, as contrasted to analog modules that operate with continuously variable signals.

Disk drive The device that writes or reads data from a magnetic disk.

Diskette The flat, flexible disk on which a disk drive writes and reads.

Display The image that appears on a screen or on other image projection systems.

Display menu The list of displays from which the user selects specific information for viewing.

Distributed control A method of dividing process control into several subsystems. A PLC oversees the entire operation.

Divide A programmable logic controller instruction that performs a numerical division of one number by another.

Documentation An orderly collection of recorded hardware and software data such as tables, listings, and diagrams to provide reference information for programmable logic controller application operation and maintenance.

Done bit (DN) Bit that is set to 1 when the instruction has completed its task, such as reaching its preset value.

Double precision The system of using two addresses or registers to display a number too large for one address or register; allows the display of more significant figures because twice as many bits are used.

Down-counter A counter that starts from a specified number and increments down to zero.

Download Loading data from a master listing to a readout or another position in a computer system.

Dry-contact-output module Enables a PLC's processor to control output devices by providing a contact isolated electrically from any power source.

E

Edit The act of modifying a programmable logic controller program to eliminate mistakes and/or simplify or change system operation.

Electrically erasable programmable read-only memory (EEPROM) A type of programmable read-only memory that is programmed and erased by electrical pulses.

Electrical optical isolator A device that couples input to output using a semiconductor light source and detector in the same package.

Electromagnetic interference (EMI) A phenomenon responsible for noise in electric circuits.

Element A single instruction of a relay ladder diagram program; an individual piece of data in an array.

Emergency stop relay A relay used to inhibit all electric power to a control system in an emergency or other event requiring that the controlled hardware be brought to an immediate halt.

Enable To permit a particular function or operation to occur under natural or preprogrammed conditions.

Enclosure A steel box with a removable cover or hinged door used to house electric equipment.

Encoder A rotary or linear device that transmits position information; a device that transmits a fixed number of pulses for each revolution.

Energize The physical application of power to a circuit or device to activate it; the act of setting the on, true, or 1 state of a programmable logic controller's relay ladder diagram output device or instruction.

Erasable programmable read-only memory (EPROM) A programmable read-only memory that can be erased with ultraviolet light, then reprogrammed with electrical pulses.

Error signal A signal proportional to the difference between the actual output and the desired output.

Ethernet A network type or protocol that uses the Carrier Sense Multiple Access Collision Detection (CSMA/CD) network access method.

Ethernet/IP An open industrial networking standard that takes advantage of commercial off-the-shelf EtherNet communication devices and physical media; IP refers to industrial protocol.

Even parity When the sum of the number of 1s in a binary word is always even.

Examine if closed (XIC) Refers to a normally open contact instruction in a logic ladder program. An examine if closed instruction is true if its addressed bit is on (1). It is false if the bit is off (0).

Examine if open (XIO) Refers to a normally closed contact instruction in a logic ladder program. An examine if open instruction is true if its addressed bit is off (0). It is false if the bit is on (1).

Exclusive-OR gate A logic device requiring one or the other, but not both, of its inputs to be satisfied before activating its output.

Execution The performance of a specific operation accomplished through processing one instruction, a series of instructions, or a complete program.

Execution time The total time required for the execution of one specific operation.

F

False As related to programmable logic controller instructions, a disabling logic state; a bit that is off or zero.

Fault Any malfunction that interferes with normal operation.

Fault indicator A diagnostic aid that provides a visual indication and/or an internal processor software indication that a fault is present in the system.

Fault-routine file A special subroutine that, if assigned, executes when the processor has a major fault.

Feedback In analog systems, a correcting signal received from the output or an output monitor. The correcting signal is fed to the controller for process correction.

Fiber optic cable Transmits information via light pulses down optical fibers.

Fieldbus An open, all-digital, serial, two-way communications system that interconnects measurement and control equipment such as sensors, actuators, and controllers.

File A formatted block of data treated as a unit.

Fixed I/O Input/output terminals on a programmable logic controller that are built into the unit and are not changeable. A fixed I/O PLC has no removable modules.

Floating-point data file Used to store real numerical values that cannot be stored in an integer file.

Flowchart A graphical representation for the definition, analysis, or solution of a problem. Symbols are used to represent a process or sequence of decisions and events.

Force function A mode of operation or instruction that allows an operator to override the processor to control the state of a device.

Force off function A feature that allows the user to reset an input image table file bit or de-energize an output independently of the programmable logic controller program.

Force on function A feature that allows the user to set an image table file bit or energize an output independently of the programmable logic controller program.

Full duplex A mode of data communications in which data may be transmitted and received simultaneously.

Function block Rectangular block with inputs entering from the left and outputs exiting on the right.

Function block diagram (FBD) Graphical language where the basic programming elements appear as blocks.

Function keys Keys on a personal computer, electronic operator device, or hand-held programmer keyboard that are labeled F1, F2, and so on. The operation of each of these keys is defined on many electronic operator interface devices.

G

Gate A circuit having two or more input terminals and one output terminal, where an output is present when and only when the prescribed inputs are present.

Gateway A device or pair of devices that connects two or more communication networks. This device may act as a host to each network and may transfer messages between the networks by translating their protocols.

Glitch A voltage or current spike of short duration that adversely affects the operation of a PLC.

Graphics library An aid in creating computer graphics for an HMI display screen.

Gray code A binary coding scheme that allows only 1 bit in the data word to change state at each increment of the code sequence.

Gray-encoder module Converts the Gray-code signal from an input device into straight binary.

Ground A conducting connection between an electric circuit or equipment chassis and the earth ground.

Ground loop A condition in which two or more electrical paths exist within a ground line.

Ground potential Zero voltage potential with respect to the ground.

H

Half-duplex A mode of data transmission that communicates in two directions but in only one direction at a time.

Handshaking The method by which two digital machines establish communication.

Hard contacts Any type of physical switch contacts.

Hard copy Any form of a printed document such as a ladder diagram program listing, paper tape, or punched cards.

Hard drive A data storage device with non-removable media used as a computer disk drive.

Hardware The mechanical, electric, and electronic devices that make up a programmable logic controller and its application.

Hardwired The physical interconnection of electric and electronic components with wire.

Hexadecimal A number system having a base of 16. This numbering system requires 16 elements for representation, and thus uses the decimal digits zero (0) through nine (9) and the first six letters of the alphabet, A through F.

High-speed counter encoder module A module that enables you to count and encode faster than you could with a regular control program written on a PLC in which the control program's execution is too slow.

Histogram A graphic representation of the frequency at which an event occurs.

HMI terminal An interactive screen that provides information, data, and control of a given process to a human user operating the HMI screen.

Host computer A main computer that controls other computers, PLCs, or computer peripherals.

Human machine interface (HMI) Graphical display hardware in which machine status, alarms, messages, diagnostics, and data entry are available to the operator in graphical display format.

I

IEC 1131 programming standard The international standard for programmable controller programming languages.

Image table An area in programmable logic controller memory dedicated to input/output data. Ones and zeros (1s and 0s) represent on and off conditions, respectively. During every input/output scan, each input controls a bit in the input image table file; each output is controlled by a bit in the output image table file.

Immediate input instruction A programmable logic controller instruction that temporarily halts the user program scan so that the processor can update the input image table file with the current status of one or more user-specified input points.

Immediate output instruction A programmable logic controller instruction that temporarily halts the user program scan so that the current status of one or more user-specified output points can be updated to current output image table file status by the processor.

Impedance The total resistive and inductive opposition that an electric circuit or device offers to a varying current at a specified frequency. Impedance is measured in Ohms (Ω) and is denoted by the symbol Z.

Increment The act of increasing the contents of a storage location or value in varying amounts.

Index A reference used to specify an element within an array.

Indirect addressing An addressing mode in which the address of the instruction serves as a reference point-instead of the actual address.

Inductance A circuit property that opposes any current change. Inductance is represented by the letter L and the unit for inductance is the Henry (H).

Industrial terminal The device used to enter and monitor the program in a PLC.

Input Information transmitted from a peripheral device to the input module and then to the data table.

Input devices Devices such as limit switches, pressure switches, pushbuttons, and analog and/or digital devices that supply data to a programmable logic controller.

Input/output (I/O) address A unique number assigned to each input and output. The address number is used when programming, monitoring, or modifying a specific input or output.

Input/output (I/O) module A plug-in assembly that contains more than one input or output circuit. A module usually contains two or more identical circuits. Normally, it contains 2, 4, 8, 16, 32, or 64 circuits.

Input/output (I/O) scan time The time required for the processor to monitor inputs and control outputs.

Input/output (I/O) update The continuous process of revising each and every bit in the input and output tables, based on the latest results from reading the inputs and processing the outputs according to the control program.

Input scan One of three parts of the PLC scan. During the input scan, input terminals are read and the input table is updated accordingly.

Instruction A command that causes a programmable logic controller to perform one specific operation. The user enters a combination of instructions into the programmable logic controller's memory to form a unique application program.

Instruction set The set of general-purpose instructions available with a given controller. In general, different machines have different instruction sets.

INT Two-byte integer.

Integer A positive or negative whole number.

Integrated circuit (IC) A circuit in which all components are integrated on a single tiny silicon chip.

Intelligent field devices Microprocessor-based devices used to provide process-variable, performance, and diagnostic information to the PLC processor. These devices are able to execute their assigned control functions with little interaction, except communications, with their host processor.

Intelligent input/output module A microprocessor-based module that performs processing or sophisticated closed-loop application functions.

Interface A circuit that permits communication between the central processing unit and a field input or output device. Different devices require different interfaces.

Interlock A system for preventing one element or device from turning on while another device is on.

Internal coil instruction A relay coil instruction used for internal storage or buffering of an on/off logic state. An internal coil instruction differs from an output coil instruction because the on/off status of the internal coil is not passed to the input/output hardware for control of a field device.

Interrupt The act of redirecting a program's execution to perform a more urgent task.

Inversion Conversion of a high level to a low level, or vice versa.

Inverter The digital circuit that performs inversion.

I/O module A plug-in assembly, containing two or more identical input or output circuits, that contain the connections between a processor and connected devices.

IP address Internet protocol address, a unique 32 bit number consisting of 4 octets, assigned for every Ethernet device on a computer network.

Isolated input module A module that receives dry contacts as inputs, which the processor can recognize and change into two-state digital signals.

Isolated input/output (I/O) circuits Input and output circuits that are electrically isolated from any and all other circuits of a module. Isolated input/output circuits are designed to allow field devices that are powered from different sources to be connected to one module.

J

Jumper A short length of conductor used to make a connection between terminals around a break in a circuit.

Jump instruction An instruction that permits the bypassing of selected portions of the user program. Jump instructions are conditional whenever their operation is determined by a set of preconditions and unconditional whenever they are executed to occur every time they are programmed.

K

K $2^{10} = 1\text{K} = 1024$; used to denote size of memory and can be expressed in bits, bytes, or words; example: $2\text{K} = 2048$.

k Kilo; a prefix used with units of measurement to designate quantities 1000 times as great.

Keyboard The alphanumeric keypad on which the user types instructions to the PLC.

Keying Bands installed on backplane connectors to ensure that only one type of module can be inserted into a keyed connector.

L

Label instruction A programmable logic controller instruction that assigns an alphanumeric designation to a particular location in a program. This location is used as the target of a jump, skip, or jump to subroutine instruction.

Ladder diagram An industry standard for representing relay logic control systems. The diagram resembles a ladder because the vertical supports of the ladder appear as power feed and return buses and the horizontal rungs of the ladder appear as series and/or parallel circuits connected across the power lines.

Ladder diagram programming A method of writing a user program in a format similar to a relay ladder diagram.

Ladder matrix A rectangular array of programmed contacts that defines the number of contacts that can be programmed across a row and the number of parallel branches allowed in a single ladder rung.

Language A set of symbols and rules for representing and communicating information among people or between people and machines; the method used to instruct a programmable device to perform various operations.

Language module Enables the user to write programs in a high-level language. BASIC is the most popular language module. Other language modules available include C, Forth, and PASCAL.

Latching relay A relay that maintains a given position by mechanical or electrical means until released mechanically or electrically.

Latch instruction One-half of an instruction pair (the second instruction of the pair being the unlatch instruction) that emulates the latching action of a latching relay. The latch instruction for a programmable logic controller energizes a specified output point or internal coil until it is de-energized by a corresponding unlatch instruction.

Leakage The small amount of current that flows in a semiconductor device when it is in the off state.

Least significant bit (LSB) The bit that represents the smallest value in a byte or word.

Least significant digit (LSD) The digit that represents the smallest value in a byte or word.

Light-emitting diode (LED) A semiconductor junction that emits light when biased in the forward direction.

Light-emitting diode (LED) display A display device incorporating light-emitting diodes to form the segments of the displayed characters and numbers.

Limit switch An electric switch actuated by some part and/or motion of a machine or equipment.

Limit test A test that determines if a value is inside or outside a specified range.

Line A component part of a system used to link various subsystems located remotely from the processor; the source of power for operation; example: 120 V alternating current line.

Line-powered sensor Normally, three-wire sensors, although four-wire sensors also exist. The line-powered sensor is powered from the power supply. A separate wire (the third) is used for the output line.

Liquid-crystal display (LCD) A display device using reflected light from liquid crystals to form the segments of the displayed characters and numbers.

Load The power used by a machine or apparatus; to place data into an internal register under program control; to place a program from an external storage device into central memory under operator control.

Load-powered sensor A two-wire sensor. A small leakage current flows through the sensor even when the output is off. The current is required to operate the sensor electronics.

Local area network (LAN) A system of hardware and software designed to allow a group of intelligent devices to communicate within a fairly close proximity.

Local input/output (I/O) A programmable logic controller whose input/output distance is physically limited. The PLC must be located near the process; however, the PLC may still be mounted in a separate enclosure.

Local power supply The power supply used to provide power to the processor and a limited number of local input/output modules.

Location In reference to memory, a storage position or register identified by a unique address.

Logic A process of solving complex problems through the repeated use of simple functions that can be either true or false. The three basic logic functions are AND, OR, and NOT.

Logic diagram A diagram that represents the logic elements and their interconnections.

Logic gate A device in which the output state is based on the state of its inputs.

Logic level The voltage magnitude associated with signal pulses representing 1s and 0s in binary computation.

Loop control A control of a process or machine that uses feedback. An output status indicator modifies the input signal effect on the process control.

M

MAC address Media access control address. A unique address, assigned by the manufacturer, that each digital device on a network has.

Machine language A programmable language using the binary form.

Major fault A fault condition that is severe enough for the controller to shut down, unless the condition is cleared.

Manufacturing automation protocol (MAP) Standard developed to make industrial devices communicate more easily.

Masking A means of selectively screening out data. Masking allows unused bits in a specific instruction to be used independently.

Mass storage A means of storing large amounts of data on magnetic tape, floppy disks, and so on.

Master control relay (MCR) A mandatory hardwired relay that can be de-energized by any series-connected emergency stop switch. Whenever the master control relay is de-energized, its contacts open to de-energize all application input and output devices.

Master control relay (MCR) zones User program areas in which all nonretentive outputs can be turned off simultaneously. Each master control relay zone must be delimited and controlled by master control relay fence codes (master control relay instructions).

Matrix A logic network that is an intersection of input and output connection points.

Memory That part of the programmable logic controller in which data and instructions are stored either temporarily or semi-permanently. The control program is stored in memory.

Memory map A diagram showing a system's memory addresses and what programs and data are assigned to each section of memory.

Menu A list of programming selections displayed on a programming terminal.

Metal-oxide semiconductor (MOS) A semiconductor device in which an electric field controls the conductance of a channel under a metal electrode called a *gate.*

Metal-oxide varistor (MOV) Used for suppressing electrical power surges.

Microprocessor A central processing unit manufactured on a single integrated-circuit chip (or several chips) by utilizing large-scale integration technology.

Microsecond One millionth of a $\text{second} = 1 \times 10^{-6}\,\text{second} = 0.000001$ second.

Millisecond One thousandth of a $\text{second} = 1 \times 10^{-3}\,\text{second} = 0.001$ second.

Mnemonic A term, usually an abbreviation, that is easy to remember and pronounce.

Mnemonic code A code in which information is represented by symbols or characters.

Modbus A network that uses a server/client communication technique.

Mode A term used to refer to the selected operating method, such as automatic, manual, TEST, PROGRAM, or diagnostic.

Module An interchangeable, plug-in item containing electronic components.

Module addressing A method of identifying the input/output modules installed in a chassis.

Most significant bit (MSB) The bit representing the greatest value of a byte or word.

Most significant digit (MSD) The digit representing the greatest value of a byte or word.

Motor controller A device or group of devices that serve to govern, in a predetermined manner, the electric power delivered to a motor.

Motor starter A special relay designed to provide power to motors; it has both a contactor relay and an overload relay connected in series and prewired so that, if the overload operates, the contactor is de-energized.

Move instruction A programmable logic controller instruction that moves data from one location to another. Although a move instruction typically places the data in a new location, the original data still reside in their original location.

Multiplexing The time-shared scanning of a number of data lines into a single channel, and only one data line is enabled at any time; the incorporation of two or more signals into a single wave from which the individual signals can be recovered.

Multiply instruction A programmable logic controller instruction that provides for the mathematical multiplication of two numbers.

Multiprocessing A method of applying more than one microprocessor to a specific function to speed up operation time and reduce the possibility of system failure.

N

National Electrical Code (NEC) A set of regulations developed by the National Fire Protection Association that governs the construction and installation of electric wiring and electric devices. The National Electrical Code is recognized by many governmental bodies, and compliance is mandatory in much of the United States.

National Electrical Manufacturers Association (NEMA) An organization of electric device and product manufacturers. The National Electrical Manufacturers Association issues standards relating to the design and construction of electric devices and products.

NEMA Type 12 enclosure A category of industrial enclosures intended for indoor use and designed to provide a degree of protection against dust, falling dirt, and dripping noncorrosive liquids. They do not provide protection against conditions such as internal condensation.

Nested branches A branch that begins or ends within another branch.

Network A series of stations or devices connected by some type of communications medium.

Network access control The method of accessing the network media (cable) to ensure that data are transmitted in an organized manner in order to reduce the possibilities of data corruption.

Node In hardware, a connection point on the network; in programming, the smallest possible increment in a ladder diagram.

Noise Random, unwanted electric signals, normally caused by radio waves or electric or magnetic fields generated by one conductor and picked up by another.

Noise filter or **noise suppressor** An electronic filter network used to reduce and/or eliminate any noise that may be present on the leads to an electric or electronic device.

Noise immunity A measure of insensitivity of an electronic system to noise.

Noise spike A short burst of electric noise with more magnitude than the background noise level.

Nonretentive output An output controlled continuously by a program rung. Whenever the rung changes state (true or false), the output turns on or off; contrasted with a retentive output, which remains in its last state (on or off) depending on which of its two rungs, latch or unlatch, was last true.

Nonvolatile memory A memory designed to retain its data while its power supply is turned off.

NOR The logic gate that results in zero unless both inputs are zero.

Normally closed contact (NC) A contact that is conductive when its operating coil is not energized.

Normally open contact (NO) A contact that is nonconductive when its operating coil is not energized.

NOT A logical operation that yields a logic 1 at the output if a logic 0 is entered at the input, and a logic 0 at the output if a logic 1 is entered at the input. The NOT, also called the *inverter,* is normally used in conjunction with the AND and OR functions.

O

Octal number system A base eight numbering system that uses numbers 0–7, 10–17, 20–27, and so on. There are no 8s or 9s in the octal number system.

Odd parity Condition when the sum of the number of 1s in a binary word is always odd.

Off-delay timer An electromechanical relay with contacts that change state a predetermined time period after power is removed from its coil; on re-energization of the coil, the contacts return to their shelf state immediately; also, a programmable logic controller instruction that emulates the operation of the electromechanical off-delay relay.

Offline programming and/or **offline editing** A method of programmable logic controller programming and/or editing in which the operation of the processor is stopped and all output devices are switched off. Offline programming is the safest way to develop or edit a programmable logic controller program since the entry of instructions does not affect operating hardware until the program can be verified for accuracy of entry.

On-delay timer An electromechanical relay with contacts that change state a predetermined time period after the coil is energized; the contacts return to their shelf state immediately on de-energization of the coil; also, a programmable logic controller instruction that emulates the operation of the electromechanical on-delay timer.

One-shot A programmed technique that sets a storage bit or output for only one program scan.

Online data change Allows the user to change various data table values using a peripheral device while the application is operating normally.

Online programming and/or **online editing** The ability of a processor and programming terminal to make joint user-directed additions, deletions, or changes to a user program while the processor is actively solving and executing the commands of the existing user program. Extreme care should be exercised when performing online programming to ensure that erroneous system operation does not result.

OPC UA An open platform communications united architecture used for transferring information between servers.

Open loop A system that has no feedback or auto correction.

Operand A number used in an arithmetic operation as an input.

Operational amplifier (op-amp) A high-gain DC amplifier used to increase signal strength for devices such as analog input modules.

Optical coupler A device that couples signals from one circuit to another by means of electromagnetic radiation, usually infrared or visible light. A typical optical coupler uses a light-emitting diode to convert the electric signal of the primary circuit into light and uses a phototransistor in the secondary circuit to reconvert the light back into an electric signal; sometimes referred to as *optical isolation*.

Optical isolation Electrical separation of two circuits with the use of an optical coupler.

OR A logical operation that yields a logic 1 output if one of any number of inputs is 1, and a logic 0 if all inputs are 0.

Output Information sent from the processor to a connected device via some interface. The information could be in the form of control data that will signal some device such as a motor to switch on or off or to vary the speed of a drive.

Output device Any connected equipment that will receive information or instructions from the central processing unit, such as control devices (e.g., motors, solenoids, alarms) or peripheral devices (e.g., line printers, disk drives, displays). Each type of output device has a unique interface to the processor.

Output image table file A portion of a processor's data memory reserved for the storage of output device statuses. A 1, on, or true state in an output image table file storage location is used to switch on the corresponding output point.

Output instruction The term applied to any programmable logic controller instruction capable of controlling the discrete or analog status of an output device connected to the programmable logic controller.

Output register or **output word** A particular word in a processor's output image table file in which numerical data are placed for transmission to a field output device.

Output scan One of three parts of the PLC scan. During the output scan, data associated with the output status table are transferred to the output terminals.

Overflow Exceeding the numerical capacity of a device such as a timer or counter. The overflow can be either a positive or negative value.

Overload A load greater than the one that a component or system is designed to handle.

Overload relay A special-purpose relay designed so that its contacts transfer whenever its current exceeds a predetermined value. Overload relays are used with electric motors to prevent motor burnout due to mechanical overload.

P

Parallel circuit A circuit in which two or more of the connected components or contact symbols in a ladder program are connected to the same pair of terminals so that current may flow through all the branches; contrasted with a series connection, in which the parts are connected end to end so that current flow has only one path.

Parallel instruction A programmable logic controller instruction used to begin and/or end a parallel branch of instructions programmed on a programming terminal.

Parallel operation A type of information transfer in which all bits, bytes, or words are handled simultaneously.

Parallel transmission A computer operation in which two or more bits of information are transmitted simultaneously.

Parity The use of a self-checking code that employs binary digits in which the total number of 1s is always even or odd.

PC Personal computer.

Peer-to-peer network A network in which nodes are given an equal chance of initiating and controlling communications.

Peripheral equipment Units that communicate with the programmable logic controller but are not part of the programmable logic controller; example: a programming device or computer.

PID Proportional-integral-derivative closed-loop control that lets the user hold a process variable at a desired set-point.

Pilot-type device Used in a circuit as a control apparatus to carry electric signals for directing performance. This device does not carry primary current.

PLC processor A computer designed specifically for programmable controllers. It supervises the action of the modules attached to it.

Polarity The directional indication of electrical flow in a circuit; the indication of charge as either positive or negative, or the indication of a magnetic pole as either north or south.

Polling A network access method where a server manages the communication process by interrogating each client under it to determine whether the client has any information to send.

Port A connector or terminal strip used to access a system or circuit. Generally, ports are used for the connection of peripheral equipment.

Power supply The unit that supplies the necessary voltage and current to a system's circuitry.

Preset value (PRE) In a counter instruction, the number of time intervals or events to be counted.

Pressure switch A switch activated at a specified pressure.

Process A continuous manufacturing operation.

Program A sequence of instructions to be executed by the processor to control a machine or process.

Program files The area of processor memory in which the ladder logic programming is stored.

Programmable controller A computer that has been hardened to work in an industrial environment and is equipped with special I/O and a control programming language.

Programmable read-only memory (PROM) A retentive memory used to store data. This type of memory device can be programmed only once and cannot be altered afterward.

Programming terminal A combination of keyboard and monitor used to insert, modify, and observe programs stored in a PLC.

Program scan One of three parts of the PLC scan. During the program scan, the CPU scans each rung of the user program.

Project file Contains all data associated with the PLC project. A project comprises five major pieces: help folder, controller folder, ladder folder, data folder, and data base folder.

Proportional-integral-derivative (PID) A mathematical formula that provides a closed-loop control of a process. Inputs and outputs are continuously variable and typically will be analog signals.

Protocol A formal definition of criteria for receiving and transmitting data through communications channels.

Proximity switch An input device that senses the presence or absence of a target without physical contact.

Pulse A short change in the value of a voltage or current level. A pulse has a definite rise and fall time and a finite duration.

R

Rack A housing or framework used to hold assemblies; a plastic and/or metal assembly that supports input/output modules and provides a means of supplying power and signals to each input/output module or card.

Random-access memory (RAM) A memory system that permits the random accessing of any storage location for the purpose of either storing (writing) or retrieving (reading) information. Random-access memory systems allow the data to be retrieved and stored at speeds independent of the storage locations being accessed.

Read The accessing of information from a memory system or data storage device; the gathering of information from an input device or devices or a peripheral device.

Read-only memory (ROM) A permanent memory structure in which data are placed at time of fabrication or by the user at a speed much slower than it will be read. Information entered in a read-only memory is usually not changed once it is entered.

Read/write memory Memory in which data can be stored (write mode) or accessed (read mode). The write mode replaces previously stored data with current data; the read mode does not alter stored data.

Real numbers Numbers that have both integer and fractional parts.

Real-time clock (RTC) A device that continually measures time in a system without respect to what tasks the system is performing.

Rectifier A solid-state device that converts alternating current to pulsed direct current.

Register A memory word or area for the temporary storage of data used within mathematical, logical, or transfer functions.

Relay An electrically operated device that mechanically switches electric circuits.

Relay contacts The contacts of a relay that are either opened or closed according to the condition of the relay coil. Relay contacts are designated as either normally open or normally closed in design.

Relay logic A representation of the program or other logic in a form normally used for relays.

Remote input/output (I/O) system Any input/output system that permits communication between the processor and input/output hardware over a coaxial or twin axial cable. Remote input/output systems permit the placement of input/output hardware at any distance from the processor.

Resolution The smallest distinguishable increment into which a quantity is divided.

Response time The amount of time required for a device to react to a change in its input signal or to a request.

Retentive instruction Any programmable logic controller instruction that does not need to be continuously controlled for operation. Loss of power to the instruction does not halt execution or operation of the instruction.

Retentive timer An electromechanical relay that accumulates time whenever the device receives power and maintains the current time should power be removed from the device. Loss of power to the device after reaching its preset value does not affect the state of the contacts.

Retentive timer instruction A programmable logic controller instruction that emulates the timing operation of the electromechanical retentive timer.

Retentive timer reset instruction A programmable logic controller instruction that emulates the reset operation of the electromechanical retentive timer.

Ring topology A network topology that that forms a data path in a ring.

Routine A series of instructions that perform a specific function or task.

RS-232 An Electronic Industries Association (EIA) standard for data transfer and communication for serial binary communication circuits.

RSLinx A communications driver used to communicate, via EtherNet/IP, between the PLC Program and its HMI application.

Run The single, continuous execution of a program by a programmable logic controller.

Rung A group of programmable logic controller instructions that controls an output or storage bit, or performs other control functions such as file moves, arithmetic, and/or sequencer instructions. A rung is represented as one section of a ladder logic diagram.

Runtime application Compiles all of the necessary application information into a single file required to run the application.

S

SCADA An acronym for supervisory control and data acquisition.

Scan time The time required to read all inputs, execute the control program, and update local and remote input and output statuses. Scan time is, in effect, the time required to activate an output controlled by programmed logic.

Schematic A diagram of graphic symbols representing the electrical scheme of a circuit.

SD Card Secure digital card. A removable solid state data storage device.

Search function Allows the user to display quickly any instruction in the programmable logic controller program.

Self-diagnostic The hardware and firmware within a controller that monitors its own operation and indicates any fault it can detect.

Sensor A device used to gather information by the conversion of a physical occurrence to an electric signal.

Sequencer A mechanical, electric, or electronic device that can be programmed so that a predetermined set of events occurs repeatedly.

Sequence table A table or chart indicating the sequence of operation of output devices.

Sequential control A process that dictates the correct order of events and allows one event to occur only after the completion of another.

Sequential Function Chart (SFC) Graphical language whose basic language elements are steps or states with associated actions and transitions with associated conditions used to move from the current state to the next.

Serial communication A type of information transfer in which the bits are handled sequentially; contrasted with parallel communication.

Series circuit A circuit in which the components or contact symbols are connected end to end, and all must be closed to permit current flow.

Servo module The device whose feedback is used to accomplish closed-loop control. Though programmed through a PLC, once programmed it can control a device independently without interfering with the PLC's normal operation.

Set-point The value that the process value is to be held to by the automatic control function.

Shield A barrier, usually conductive, that substantially reduces the effect of electric and/or magnetic fields.

Shift To move binary data within a shift register or other storage device.

Shift register A PLC function capable of storing and shifting binary data.

Short circuit An undesirable path of very low resistance in a circuit between two points.

Short-circuit protection Any fuse, circuit breaker, or electronic hardware used to protect a circuit or device from severe overcurrent conditions or short circuits.

Signal The event or electrical quantity that conveys information from one point to another.

Significant digit A digit that contributes to the precision of a number. The number of significant digits is counted beginning with the digit contributing the most value, called the *most significant digit* (leftmost), and ending with the digit contributing the least value, called the *least significant digit* (rightmost).

Silicon-controlled rectifier (SCR) A semiconductor device that functions as an electronic switch.

Single-scan function A supervisory instruction that causes the control program to be executed for one scan, including input/output update. This troubleshooting function allows step-by-step inspection of occurrences while the machine is stopped.

Sink mode output A mode of operation of solid-state devices in which the device controls the current from the load. For example, when the output is energized, it connects the load to the negative pole of the supply.

SINT A data type that stores an 8-bit (1-byte) signed integer value.

Snubber A circuit generally used to suppress inductive loads; it consists of a resistor in series with a capacitor (RC snubber) and/or a MOV placed across the alternating current load.

Software Programs that control the processing of data in a system, as contrasted to the physical equipment itself (hardware).

Solid-state switch Any electronic device incorporating a transistor, silicon-controlled rectifier, or triac semiconductor switch to control the on/off flow of electric power.

Source mode output A mode of operation of solid-state output devices in which the device controls the current to the load. For example, when the output is energized, it connects the load to the positive pole of the supply.

Star topology A network architecture in which all network nodes are connected to a central device that routes the nodes' messages.

State The logic 0 or 1 condition in programmable logic controller memory or at a circuit input or output.

Station Any programmable logic controller, computer, or data terminal connected to, and communicating by means of, a data highway.

Status indicators LEDs that indicate the on-off status of an input or output point and are visible on the outside of the PLC.

Stepper-motor module Provides pulse trains to a stepper-motor translator that enables control of a stepper motor.

STI An acronym for selectable time interrupt, a subroutine that executes on a time basis rather than an event basis.

Storage bit A bit in a data table word that can be set or reset but that is not associated with a physical input or output terminal point.

Structure A data type that is composed from multiple data types.

Structured Text (ST) High-level, text-based language with commands that support a highly structured program development and the ability to evaluate complex mathematical expressions.

Subroutines Program files that are scanned only when called on by logic and can be used to break the program into smaller segments.

Subtract A programmable logic controller instruction that performs the mathematical subtraction of one number from another.

Suppression device A unit that attenuates the magnitude of electrical noise.

Surge A transient wave of current or power.

Synchronous shift register A shift register in which only one change of state occurs per control pulse.

Synchronous transmission A type of serial transmission that maintains a constant time interval between successive events.

T

Tag A text-based name for an area of the controller's memory where data are stored.

Tag database Used for storing, deleting, and editing of tags.

Tap A device that provides mechanical and electrical connections to a trunk cable. A tap allows the signals on the trunk to be passed to a station and the signals transmitted by the stations to be passed to the trunk.

Task It holds the information necessary to schedule the program's execution and sets the execution priority for one or more programs.

Terminal address The alphanumeric address assigned to a particular input or output point. It is also related directly to a specific image table bit address.

Thermocouple A temperature-measuring device that utilizes two dissimilar metals for temperature measurement. As the junction of the two dissimilar metals is heated, a proportional voltage difference, which can be measured, is generated.

Thumbwheel switch A rotating switch used to input numeric information into a controller.

Time base A unit of time generated by a microprocessor's clock circuit and used by PLC timer instructions. Typical time bases are 0.001, 0.01, 0.1, and 1.0 second.

Timed contact A normally open and/or normally closed contact that is actuated at the end of a timer's time-delay period.

Timer In relay-panel hardware, an electromechanical device that can be wired and preset to control the operating interval of other devices. In a programmable logic controller, a timer is internal to the *processor;* that is, it is controlled by a user-programmed instruction.

Toggle switch A panel-mounted switch with an extended lever; normally used for on/off switching.

Token The logical right to initiate communications in a communication network.

Token passing A technique in which tokens are circulated among nodes in a communication network.

Topology The structure of a communications network; examples are bus, ring, and star.

Transducer A device used to convert physical parameters such as temperature, pressure, and weight into electric signals.

Transformer An electric device that converts a circuit's electrical energy into a circuit or circuits with different voltages and current ratings.

Transistor A three-terminal active semiconductor device composed of silicon or germanium that is capable of switching or amplifying an electric current.

Transistor-transistor logic (TTL) A semiconductor logic family in which the basic logic element is a multiple-emitter transistor. This family of devices is characterized by high speed and medium power dissipation.

Transitional contact A contact that, depending on how it is programmed, will be on for one program scan every 0 to 1 transition, or every 1 to 0 transition, of the referenced coil.

Transmission line A system of one or more electric conductors used to transmit electric signals or power from one place to another.

Triac A solid-state component capable of switching alternating current.

True As related to programmable logic controller instructions, an on, enabled, or 1 state.

Truth table A table listing that shows the state of a given output as a function of all possible input combinations.

TTL-input module Enables devices that produce TTL-level signals to communicate with a PLC's processor.

TTL-output module Enables a PLC to operate devices requiring TTL-level signals to operate.

Twisted pair cable A pair of wires that can transmit data; the wires are twisted to provide protection against crosstalk.

U

UDT User data type. A data structure that is created by the programmer.

Unlatch instruction One-half of a programmable logic controller instruction pair that emulates the unlatching action of a latching relay. The unlatch instruction de-energizes a specified output point or internal coil until re-energized by a latch instruction. The output point or internal coil remains de-energized regardless of whether or not the unlatch instruction is energized.

Up-counter An event that starts from 0 and increments up to the preset value.

V

Variable A factor that can be altered, measured, or controlled.

Variable data Numerical information that can be changed during application operation. It includes timer and counter accumulated values, thumbwheel settings, and arithmetic results.

Volatile memory A memory structure that loses its information whenever power is removed. Volatile memories require a battery backup to ensure memory retention during power outages.

W

Watchdog timer Monitors logic circuits controlling the processor. If the watchdog timer, which is reset every scan, ever times out, the processor is assumed to be faulty and is disconnected from the process.

Word A grouping or a number of bits in a sequence treated as a unit.

Word length The total number of bits that make up a word. Most programmable logic controllers use either 8, 16, or 32 bits to form a word.

Work cell A group of machines that work together to manufacture a product; normally includes one or more robots. The machines are programmed to work together in appropriate sequences. Work cells are often controlled by one or more PLCs.

Write Refers to the process of loading information into memory; can also refer to block transfer, that is, a transfer of data from the processor data table to an intelligent input/output module.

Z

Zone The portion of a PLC ladder program that can be enabled or disabled by a control function.

OceanofPDF.com

Index

Note: Page numbers followed by f and t represent figures and tables respectively.

A

Access method, 305
Accumulated time, 124
Accumulated value, 124
  timer, 124, 126
  up-counter, 149–150
Accumulated value (ACC) word, 125
  up-counter, 149
Activation, SCADA, 314
Active tab, 83
Actuators, 103
  in control system, 293
A/D (analog-to-digital) converter, 24, 85
Addition, 47
Addition instruction, 220–223, 221f, 222f
Add-On instructions, 381
Addressing, 15, 79, 79f
  bit level, 16, 17f
  I/O connection diagram, 79f
  rack/slot-based, 15, 15f
  tag-based, 15, 17f
    CLX system, 337, 338f
  word level, 16, 17f
Addressing formats, 197–198, 198f
Alarm monitor program, 154–155, 154f
Alarm server, 319
Alarm system, SCADA, 313–314, 314f
Alias tag, 16, 328–329, 329f
Allen-Bradley
  controllers, 45. *See also* specific types

ControlLogix platform from, 41
memory structures, 68
Micro-Logic 4-channel analog thermocouple input module, 24
MicroLogix controller, 71, 71f
RSLogix 500. *See* RSLogix 500 controllers
RSLogix 5000. *See* RSLogix 5000 controllers
RSLogix software packages, 83
subroutine-related instructions, 179, 179f
Allen-Bradley Data Highway, 305, 306f, 307, 307f
Allen-Bradley Pico controller, 8
installation, 8–9
All Mode, FAL instruction, 199
Alternating current (AC) supply, 5
Ambient temperature rating, discrete I/O module, 28
American Standard Code for Information Interchange (ASCII), 46
ANALOG COMMON terminal, 25, 25f
Analog control, 296
Analog devices, 23, 85–86, 209
Analog I/O modules, 23–26, 24–26f
specifications, 29
channels per module, 29
common-mode rejection, 29
input current/voltage range, 29
input impedance and capacitance, 29
input protection, 29
output current/voltage range, 29
resolution, 29
Analog output interface module, 26, 211–212, 211f
Analog signals, 85–86, 86f
Analog-to-digital (A/D) converter, 24, 85
AND function, 54–55, 54f, 57f, 194
AND gate, 54, 54f
Annunciator flasher program, 137, 137f
Architecture, 4
Arithmetic
binary, 47–49

floating point, 49–50, 49–50f
Array, CLX system, 332–333, 333f
ASCII (American Standard Code for Information Interchange), 46
ASCII code, 46–47, 46t
Associative law, 59
Assume Data Available indicator marker, FBD, 384, 385f
Audit, SCADA, 314
Automatic control process, 111–112, 112f
Automatic mode, water level control process in storage tank, 109–110
Automatic sequential control systems on-delay timer (TON), 129–130, 129f
Automatic stacking program, 162–164, 163f
Auxiliary seal-in contact, 187, 187f

B

Backplane current draw, discrete I/O module, 28
Backplane power, 19
BAND (Boolean AND) function block, 75, 381–382, 382f
Bandwidth, 311
Bang-bang control, 213
Barcode reader, 258, 258f
Bar code scanners, 101, 101f
Base of 2, 40. *See also* Binary system
Base of 10, 40. *See also* Decimal system
Base tag, 16, 328, 329f
data, 329–330, 330f
BASIC, 74
Batch processing, 291, 291f
Battery backup, 276, 276f
BCD. *See* Binary Coded Decimal (BCD)
BCD-to-binary conversion, 45
Bearing lubrication program, 135f
Binary arithmetic, 47–49
Binary Coded Decimal (BCD) output device, 209–210, 210f
Binary coded decimal (BCD) system, 43–45, 44t, 45f
numeric values, 44t
Binary information, 269
Binary number(s)

complement of, 42, 42t
conversion to decimal number, 40, 41f
decimal number conversion to, 41, 41f
hexadecimal number conversion to, 43, 43f
octal number conversion to, 43f
signed, 42t
Binary principle, 54
Boolean algebra and, 57–58
Binary system, 40–41, 40–41f, 40t
binary principle, 54
Boolean algebra and, 57–58
numeric values, 44t
Binary-to-BCD conversion, 45
Bit, 41, 47
Bit Field Distribute (BTD), 194, 379
Bit file, 68
Bit level addressing, 16, 17f
Bit-level logic instructions, 76–78
Examine If Closed (XIC), 76, 76f
Examine If Open (XIO), 76–77, 76f
Output Energize (OTE), 77–78, 77f
separating action of field device, 78, 78f
symbolic, 76
Bit-level programming, CLX system, 335–347
internal relay instructions, 341–342, 342–343f
ladder logic program, 336–337, 336f
addition to main routine, 338–341, 338–341f
latch and unlatch instructions, 343–345, 343–345f
one-shot instruction, 345–346, 345–347f
program scan, 335–336, 335f
tag-based addressing, 337, 338f
Bit-oriented I/O, 19
Bit shift instructions, 250, 251f
bit address, 251
carriers tracking flowing through 16-station machine, 254, 256, 256f

control, 251
file, 251
length, 252
Bit shift left (BSL) instruction, 250, 251f
Allen Bradley RSLogix 5000, 256–257, 257f
with wraparound operation, 252–253, 254f
Bit shift registers, 249–257
bit shift instructions. *See* Bit shift instructions
File Shift menu tab, 251, 251f
Bit shift right (BSR) instruction, 250, 251f, 252, 253f
Allen Bradley RSLogix 5000, 256
Bit-wide bus networks, 304
Bleeder resistor, 270, 270f
connected to proximity sensor, 98, 98f
Block-formatted counter, 146, 146f
Block format timer instruction, 124, 124f
BOOL (base tag), 329
Boolean algebra, 57
associative law, 59
commutative law, 58
distributive law, 59
logic operators, 58f
Boolean equations, 57
for logic gate circuits, 59–60, 60f
Boolean expressions, logic gate circuits from, 59, 59f
Boolean instructions, 31
Bottle-filling motion control process, 300, 300f
motion module, 301
programmable logic controller, 300
servo drive, 301
servo motor, 301
Branch instructions, 79–81, 80f
input, 79, 80f
matrix limitation diagram, 80–81
nested, 80, 80f

original circuit, 81f
output, 80, 80f
reprogrammed circuit, 81f
vertical contact, 80–81, 81f
Break-before-make pushbutton, 94
Bridges, 305
Bus topology network, 303–304, 303f
Byte, 41
Byte-wide bus networks, 304

C

Calibration, instrument, 117
Cam-operated sequencer switch, 238, 238f, 239
Cam switches, 238
Can-counting program, 151, 151f
Capacitive proximity sensors, 98, 98f
Carry bit, 222
Cascading counters, 158–161, 160f
Cascading timers, 136–138, 138f
CLX system, 357, 358f
Centralized control, 291, 292f
Central processing unit (CPU), 4–6, 29–31, 29f, 30f. *See also* Processor module
backup memory battery, 276, 276f
power supply, 29, 30f
Channels per module, analog I/O modules, 29
Circuits. *See also* specific types
electrical interlocking, 106–108, 108f
seal-in, 106
Circulating shift register function, 250
Clear Faults bit, 316
Clear (CLR) instruction, 194, 228
Closed architecture, 4
Closed-loop control system, 212–214, 294, 294f
block diagram of, 213, 213f
Closed-loop servo motor system, 105, 106f
CLX system. *See* ControlLogix (CLX) system

CMOS-RAM chips, 32
Coaxial cable, 302, 302f
Codes. *See* Number systems and codes
Cold junction compensating (CJC) thermistor, 24
Collision detection access control scheme, 305
Combination control process, 111, 111f
Commands, to VFD, 315
Commissioning, 273
Common Industrial Protocol (CIP), 308
Common-mode rejection, analog I/O modules, 29
Communication module, 15, 27
Communications, by PLC CPU, 30
Communications capability, 3
Communications protocol, 285
Commutative law, 58
CompactLogix system, 323
Comparison instruction, CLX system, 369–371, 370–372f, 373–379, 373–379f
Compute (CPT) instruction, 220–221, 220–221f
Compute/Math menu tab, 220, 220f
Computer memory, 41
Computers, *vs.* PLCs, 10–11, 10–11f
Configuration
  CLX system, 324–325, 325f
  communications, 284
Consecutive group, 194
Constant voltage (CV) transformer, 272
Contact histogram function, 275–276
Contact instruction, 78
Contactors, 92, 93f
Contacts
  control relays, 92
  instantaneous, 122
  normally closed (NC), 91, 92f
  normally open (NO), 91, 92f
Container-filling process, 294f
Continuously variable, 23

Continuous process, 291, 291f
Continuous-scan test mode, 85
Continuous tasks, 326
Continuous test mode, 274
Control circuit, 92
Control devices, output, 103–105, 104–105f
Control, FAL instruction, 198
Control file, 68
Controllers, 293–295. *See also* specific types
  deadband, 296
  proportional, 296–300, 296f
Controller tag, 328
ControlLogix (CLX) system, 323–391
  array, 332–333, 333f
  bit-level programming, 335–347
    internal relay instructions, 341–342, 342–343f
    ladder logic addition to main routine, 338–341, 338–341f
    ladder logic program, 336–337, 336f
    latch and unlatch instructions, 343–345, 343–345f
    one-shot instruction, 345–346, 345–347f
    program scan, 335–336, 335f
    tag-based addressing, 337, 338f
  comparison instructions, 369–371, 370–372f
  configuration, 324–325, 325f
  counters, 360–361, 360f
    combined with timer functions, 364–365, 365f
    count-down (CTD) counter, 363–364, 363–364f
    count-up (CTU) counter, 361–362f, 361–363
    overview, 360–361
  FBD programming, 385–390, 385–390f
  function block diagram (FBD), 381–390f, 381–391
    Add-On instructions, 381
    Assume Data Available indicator marker, 384, 385f
    BAND (Boolean AND) function block, 381–382, 382f
    data latching, 384, 384f
    feedback loop, 384, 384f

function blocks, 381
references, 382, 382f
signal flow and execution, 383–384, 383f
wire connectors, 382–383, 383f
wires and pins, 382, 383f
math, comparison, and move instructions, combining, 373–379, 373–379f
math instructions, 367–369, 367–369f
memory layout, 324, 324f
move instructions, 372–379, 373–379f
platform, 41
program, 205, 206f, 326–327, 327f
project, 325–326, 325–326f
routines, 327, 327f
structures, 330–331, 330–331f
tags, 327–330, 328–330f
creating, 331, 331f
edit tags, 332, 332f
monitor tags, 332, 332f
tasks, 326, 326f
timers, 350–358
cascading, 357, 358f
off-delay timer (TOF), 354–355, 354–355f
on-delay timer (TON), 351–354, 351–354f
predefined structure, 350–351, 350f
reciprocating, 357
retentive on-delay timer (RTO), 356–357, 356–357f
Control management PLC application, 11, 11f
ControlNet, 308–309, 310f
Control panel
PLC-based, 2, 2f
relay-based, 2, 2f
Control process
automatic, 111–112, 112f
combination, 111, 111f
sequential, 111, 111f, 112f

flow diagram, 111, 112f
Control relays (CR)
coils, 92
contacts, 92
electromagnetic, 91–92, 91–92f
internal, 81
operation, 91, 91f
pilot lights control operation, 91–92, 92f
symbol, 91, 92f
Control systems. *See also* Process(es); Process control
closed-loop, 294, 294f
distributive, 291, 292f, 293
HMI, 293–294, 293f
motion control, 300–301
on/off control, 295–296, 295f
PID control, 296–300
sensors, 293
signal conditioning, 293
structure of, 293–295
Control variable (CV), 214
Convert from BCD (FRD) instruction, 220, 228, 228f
Convert-to-BCD (TOD) instruction, 220, 228, 228f
Conveyor warning signal circuit, 128f
COP. *See* File copy (COP) instruction
Cost, 3
Count-down (CTD) counter, 156–157, 156f
CLX system, 363–364, 363–364f
Count-down (CD) enable bit, 149
Counter file, 68
Counter function, timer function and, 162–165
Counter number, 149
Counters, 74, 145–166
accumulated value, 146f, 147
application, 146f
block-formatted, 146, 146f

cascading counters, 158–161, 160f
CLX system, 360–361, 360f
combined with timer functions, 364–365, 365f
count-down (CTD) counter, 363–364, 363–364f
count-up (CTU) counter, 361–362f, 361–363
overview, 360–361
combining with timer functions, 162–165, 163f
decrement, 146
down-counter, 155–158, 155f
electronic, 146
high-speed, 165, 166f
increment, 146
incremental encoder-counter applications, 161–162, 162f
input conditions, 146
instructions, 146
mechanical, 146, 146f
selection toolbar, 150f
up-counter, 147–155, 148f
Counting, high-speed, 3f
Count-up (CTU) counter CLX system, 361–362f, 361–363
Count-up (CU) enable bit, 149
CPU section, processor module, 29
Cross reference function, 274, 275, 275f
Current sensing, 23–24
CV. *See* Control variable (CV)

D

D/A (digital-to-analog) converter, 26, 85
Daisy-chain topology, 311, 311f
Data communications, 301–313. *See also* Network communications
Allen-Bradley Data Highway, 305, 306f, 307, 307f
ControlNet, 308–309, 310f
defined, 301
DeviceNet, 307–308, 308f, 309f
EtherNet/IP, 309, 310f, 311
Fieldbus, 311, 311f
HART, 312–313, 312f

industrial networks, functionality levels, 302, 303f
LAN, 302, 302f
Modbus, 311
parallel data transmission, 306, 306f
point-to-point serial communications link, 301, 301f
PROFIBUS-DP, 311–313, 312f
SERCOS, 311–312, 312f
serial, 307, 307f
serial transmission, 306, 306f
transmission media, 302, 302f
Data comparison, 194, 202–206. *See also* Data manipulation
data transfer *vs.,* 202
EQU instruction, 201, 202, 202f, 204–205, 204f
GEQ instruction, 202–204, 204f
GRT instruction, 202, 203, 203f, 204–205, 204f
LEQ instruction, 202–204, 204f
LES instruction, 202, 203, 203f, 204–205, 204f
LIM instruction, 202, 205, 205f, 206f
MEQ instruction, 202
NEQ instruction, 202, 203, 203f
use, 202
Data files, 68–71, 69–71f
Data File screens, RSLogix 500 software, 84, 84f
Data flow, scan process, 72, 72f
Data Highway, 305, 306f, 307, 307f
Data latching, FBD, 384, 384f
Data manipulation, 194
closed-loop control, 212–214
data comparison, 194, 202–206
data transfer, 194–202
defined, 194
FAL instruction. *See* File arithmetic and logic (FAL) instruction
numerical data I/O interfaces, 209–212
PID controllers and, 213–214, 214f
programs, 206–209, 207–209f
use of, 194

Data monitoring functions, 274–275, 275f
Data server, 319
Data transfer. *See also* Data manipulation
concept, 194, 194f
data comparison *vs.,* 202
defined, 194
DCS. *See* Distributive control system (DCS)
Deadband, 213, 296
Decimal number(s)
binary number conversion to, 40, 41f
conversion to binary number, 41, 41f
hexadecimal number conversion to, 43, 43f
octal number conversion to, 43f
Decimal system, 40, 40f, 40t
numeric values, 44t
Decrement counter, 146–147
Derivative action, proportional control and, 297
Destination, FAL instruction, 199
Destination register, 194
Determinism, data communication and, 308
Device bus networks, 304
DeviceNet, 307–308, 308f, 309f, 315, 317, 317f
DF1 Radio Modem, 302
Difference, 47
Differential connections, 29
Digital devices, 85
Digital signal waveform, 40f
Digital-to-analog (D/A) converter, 26, 85
DINT (base tag), 330
DINT (data as a double integer), 31
DIP (dual in-line package) switches, 95, 95f
Direct current (DC) supply, 5
Directory, SCADA, 314
Direct scan technique, 100, 100f
Discrete I/O modules, 19–23, 19–23f, 19t
specifications, 27–28

ambient temperature rating, 28
backplane current draw, 28
electrical isolation, 28
input on/off delay, 28
input threshold voltages, 28
inrush current, 28
leakage current, 28
nominal current per input, 28
nominal input voltage, 27–28
output voltage, 29
points per module, 28
short circuit protection, 28
Discrete manufacturing, 291, 291f
Dishwasher
timed sequencer switch, 238–239, 239f
wiring diagram and timing chart, 239f
Distributive control system (DCS), 291, 292f, 293, 302
Distributive law, 59
Division, 49
Division instruction, 220, 225–227, 226–227f
Done bit, 241, 252
Done (DN) bit, 125, 149
Double precision, 50
Down-counter, 146–147, 155–158, 155–158f
Drilling process, ladder logic program for (example), 114–116, 114–116f
Drink-manufacturing program, PLC, 201–202, 202f
DriveLogix system, 323
Driver, 285
Drum-operated sequencer switch, 238, 238f
Drum switches, 238
Dual in-line package (DIP) switches, 95, 95f
Duplex communication system, 306
Duration, PID controllers, 213

E

Editing, 273

Edit tags, CLX system, 332, 332f
Electrical continuity, 72
Electrical devices, 85
Electrical interlocking circuits, 106–108, 107f, 108f
Electrical isolation, discrete I/O module, 28
Electrically erasable programmable read-only memory (EEPROM), 32, 33f
Electrical noise, 269–270, 269f
Electromagnetic control relays, 91–92, 91–92f
Electromagnetic interference (EMI), 269–270
Electromagnetic latching relays, 108–110, 108–110f
Electromechanical retentive timer, 133f
Electronic counters, 146
Electronic output module protection, 279, 279f
Electronic timing relays, 122, 122f
Enable (EN) bit, 125, 241, 252
Enclosures
  HMI, 34–35
  NEMA 12, 267–269, 267f
  temperature inside, 267
Encoder, 103, 103f
Encoder-counter module, 27, 27f
EQU (Equal) instruction, 202, 203, 203f, 204–205, 204f
Erasable Programmable Read-Only Memory (EPROM), 32
Error(s), 214
  checking, 277
  verifying program, 280, 280f
Error amplifier, 294
Error bit, 241, 252
EtherNet/IP, 309, 310f, 311, 318
Even parity bit, 47, 47t
Event-driven sequencer program, 244, 245f, 247, 248f
Event history, HMIs and, 35
Event server, 319
Event tasks, 326
Examine If Closed (XIC) instruction, 76, 76f
  interpretation, 77

programming, 82–83, 82f
Examine If Open (XIO) instruction, 76–77, 76f
interpretation, 77
programming, 82–83, 82f
Exclusive-OR (XOR) function, 57, 58f, 194
Explorer window, 318
Exponent, 50
Expression, FAL instruction, 199

F

FactoryTalk services platform, SCADA, 314, 314f
FactoryTalk View software, 318
FAL instruction. *See* File arithmetic and logic (FAL) instruction
Fault routine, 188, 327
Feedback, 214
Feedback loop, FBD, 384, 384f
Fiber optic sensors, 100, 100f
Fiber optic transmission medium, 302, 302f
Fieldbus, 311, 311f
Field devices, 6
Field devices testing, 4
FIFO (first in, first out) instruction, 257–261
LIFO stack operation *vs.,* 260, 260f
operating program, 258–259, 259f
RSLogix 500 FIFO load (FFL) instruction, 258, 258f
RSLogix 500 FIFO unload (FFU) instruction, 258–259, 258f
RSLogix 5000 programming with, 260–261, 261f
File, 194, 196
File add function, 230, 230f
File addressing, 197–198, 198f
File arithmetic and logic (FAL) instruction, 230, 230f
All Mode, 199
Control, 198
Destination, 199
exceptions to rule, 200
Expression, 199

file-to-file copy function and, 199–200, 199f
file-to-word moves, 200, 200f
Incremental mode, 199
Length, 198
Mode, 198
Numeric Mode, 199
overview, 198, 198f
Position, 198
File arithmetic operations, 230–231, 230–231f
File copy (COP) instruction, 201, 201f
PLC drink-manufacturing program, 201–202, 202f
File divide function, 231, 231f
File multiply function, 231, 231f
File Shift menu tab, 251, 251f
File subtract function, 230, 230f
File-to-file shifts, 196–197, 197f
FAL instruction and, 199–200, 199f
File-to-word moves, 197
FAL instruction and, 200, 200f
Fill file (FLL) instruction, 201, 201f
First in, first out (FIFO) instructions, 257–261
LIFO stack operation *vs.,* 260, 260f
operating program, 258–259, 259f
RSLogix 500 FIFO load (FFL) instruction, 258, 258f
RSLogix 500 FIFO unload (FFU) instruction, 258–259, 258f
RSLogix 5000 programming with, 260–261, 261f
Fixed I/O, 4, 4f
advantage, 4
disadvantage, 4
Flash EEPROMs, 32–33, 33f
Flashing pilot light subroutine, 179, 180f
Flexibility, 3
FlexLogix system, 323
FLL instruction. *See* Fill file (FLL) instruction
Floating point arithmetic, 49–50

components of, 49–50, 49f
example of, 49, 49f
features of, 49
Floating point file, 68
Flow measurement, 102–103, 103f
Flowmeter, turbine type, 102–103, 103f
Fluid pumping process, 132–133, 132f
Force function, 182–185
Forcing external I/O addresses, 182–185, 183–184f
Full-duplex transmission, 306
Fully automatic/intelligent PID control, 299
Function block diagram (FBD), CLX system, 381–390f, 381–391
Add-On instructions, 381
Assume Data Available indicator marker, 384, 385f
BAND (Boolean AND) function block, 381–382, 382f
data latching, 384, 384f
feedback loop, 384, 384f
function blocks, 381
references, 382, 382f
signal flow and execution, 383–384, 383f
wire connectors, 382–383, 383f
wires and pins, 382, 383f
Function block diagram (FBD) programming, 74, 75
CLX system, 385–390, 385–390f
concept, 74
equivalents to ladder logic contacts, 75f
ladder diagram and, 74, 75f
use of, 74
Function blocks, 381

G

Gateways, 304–305
GEQ instruction. *See* Greater than or equal (GEQ) instruction
Global objects, 318
Graphic HMI terminals, 293
Graphic object properties, 318

Graphics library, 318
Graphics library, HMI, 35, 35f
Gray code, 45, 45t, 46f
Greater than (GRT) instruction, 202, 203, 203f, 204–205, 204f
Greater than or equal (GEQ) instruction, 202–204, 204f
Grounding, 270–271, 271f
Ground loops, 271, 271f
GRT instruction. *See* Greater than (GRT) instruction

H

Hand-held programming devices, 33, 33f
Hardware, 4, 14–35
  analog I/O modules, 23–26, 24–26f
  communication module, 27
  CPU, 29–31, 29f, 30f
  data recording and retrieval, 33
  discrete I/O modules, 19–23, 19–23f, 19t
  encoder-counter module, 27, 27f
  high-speed counter module, 27, 27f
  human machine interfaces, 34–35, 34f, 35f
  I/O specifications
    analog I/O module, 29
    discrete I/O module, 27–28
  memory design, 31–32, 31f
  motion and position control modules, 27, 27f
  programming terminal devices, 33, 33f
  stepper-motor module, 27
Hardwired logic
  defined, 60
  programmed logic *vs.,* 60–61
Hardwired time-delay circuit, 207
Hardwired TOF relay circuit, 130–132, 131f
  equivalent PLC program, 132–133, 132f
HART, 312–313, 312f
Heating application, proportional band for, 297, 297f
Heat-shrinkable wire identification sleeves, 270, 270f

Hexadecimal (hex) numbering system, 40t, 43, 43f, 44t
conversion to binary number, 43, 43f
conversion to decimal number, 43, 43f
numeric values, 44t
High-density module, 18
High-speed counter (HSC), 150, 165, 166f
High-speed counter module, 27, 27f
High-speed counting, 3f
HMI. *See* Human machine interface (HMI)
Holding circuits, 106, 106f
Hold-up time, 29
Horizontal scan, 73
Human machine interface (HMI), 10, 10f, 34–35, 34f, 35f, 293–294, 293f
communication, 320
design, 34–35
event history, 35
graphics library, 35, 35f
graphic terminals, 293
monitor and enclosure, 34–35
motor control application, 319f
PLC programming, 318–320
project, 318
server, 319
structure, 34, 34f
tag, 318
tasks in development of, 34
terminal, 318
trend monitoring, 35, 35f
Hunting, 296
Hysteresis, 97–98

IEC 61131 standard programming language, 73–74, 74f
IEEE 754 Standard, 50
Immediate input with mask (IIM) instruction, 181–182, 182f
Immediate I/O instructions, 181–182, 182f
Immediate output with mask (IOM) instruction, 181–182, 182f

Incremental encoder-counter applications, 161–162, 162f
Incremental mode, FAL instruction, 199
Incremental optical encoder, 161, 162f
Increment counter, 146
Individual control, 291, 292f
Inductive loads, noise suppression for, 269, 269f
Inductive proximity sensors, 97, 97f
Inductive proximity switches, 98–99
Industrial networks
  control level, 302, 303f
  device level, 302
  functionality levels, 302, 303f
  information level, 302, 303f
Input branching, 79, 80f
  nested, 80, 80f
Input conditions, 61
Input current/voltage range, 29
Input devices, 6
  and outputs, relationships between, 2, 3f
Input file, 68
Input image table file, 69–70, 70f
Input impedance and capacitance, analog I/O modules, 29
Input instructions, 202. *See also* Data comparison
Input malfunctions, troubleshooting, 277–278, 278f
Input on/off delay, discrete I/O module, 28
Input/output (I/O) section, 4, 6, 6f, 15–18, 15f
  addressing, 15, 17f
  field/real-world devices, 6
  fixed, 4, 4f
  modular, 5, 5f
  rack-based, 15, 15f
  remote I/O rack, 15, 15f
  wiring connections, 7f, 8
Input protection, 29
Input threshold voltages, discrete I/O module, 28
Input troubleshooting guide, 283f

Inrush current, discrete I/O module, 28
Instantaneous contacts, 122
Instruction addressing, 79, 79f
Instruction list (IL) programming language, 74
vs. ladder diagram programming, 74, 74f
Instruction palette, 84
Instructions. *See also* Program control instructions; Sequencer instructions
counter, 146–147
override, 173
timer, 123–124, 124f
block format, 124, 124f
off-delay timer, 130–132f, 130–133
on-delay timer, 124–130, 125–127f
Instruction set, 11, 11t
Instruction toolbar with bit, RSLogix 500 software, 83f
Instrument(s), 117
calibration, 117
smart, 117, 117f
Instrumentation, 117
INT (base tag), 330
Intact, data, 197
Integer file, 68
Integral action, proportional control and, 297
Internal bits, 81
Internal coils, 81
Internal control relays, 81
Internal outputs, 81
Internal relay instructions, 81, 81f
CLX system, 341–342, 342–343f
Internal storage bits, 81
Interposing relay connection, 21, 22f
Inverter, 55
I/O address format for SLC family of PLCs, 69, 70f
I/O bus networks, 304
I/O Configuration screen, RSLogix 500 software, 84, 84f

I/O connection diagram, 79f
I/O count, 11
I/O modules
  analog, 23–26, 24–26f
  communication module, 27
  discrete, 19–23, 19–23f, 19t
  encoder-counter module, 27, 27f
  high-speed counter module, 27, 27f
  motion and position control modules, 27, 27f
  stepper-motor module, 27
I/O (input/output) section. *See* Input/output (I/O) section
I/O specifications
  analog I/O module, 29
    channels per module, 29
    common-mode rejection, 29
    input current/voltage range, 29
    input impedance and capacitance, 29
    input protection, 29
    output current/voltage range, 29
    resolution, 29
  discrete I/O module, 27–28
    ambient temperature rating, 28
    backplane current draw, 28
    electrical isolation, 28
    input on/off delay, 28
    input threshold voltages, 28
    inrush current, 28
    leakage current, 28
    nominal current per input, 28
    nominal input voltage, 27–28
    output voltage, 29
    points per module, 28
    short circuit protection, 28
Isolation transformers, 272

J

Jog circuit with control relay, 113f
Jog Drive bit, 316
Jump (JMP) instruction, 176–178, 176–178f
Jump to subroutine (JSR) output instruction, 179

K

Keyswitch, three-position, 85f

L

Label (LBL) instruction, 176–177
Ladder diagram (LD) language, 74
  entering program, 83–85
    Data File screens, 84, 84f
    instruction toolbar with bit, 83f
    I/O Configuration screen, 84, 84f
    main window, 83–84, 83f
    Select Processor Type screen, 84, 84f
  function block diagram and, 75, 75f
  function of, 78, 78f
  vs. instruction list programming, 74–75, 74f
Ladder logic contacts, function block diagram equivalents to, 75f
Ladder logic program, 6, 7, 8f, 60, 60f
  CLX system, 336–337, 336f
    addition to main routine, 338–341, 338–341f
  diagram for modified process, 9f
  drilling process (example), 114–116, 114–116f
  electrical continuity in, 72
  monitoring, 84f, 85
  parallel instructions, 113, 113f
  relay schematics conversion to, 111–113
  rung conditions, evaluating, 72
  for sequential control process, 111, 112f
  series instructions, 113, 113f
  troubleshooting, 280–282, 280–284f
  writing, from narrative description, 114–116, 114–116f
Ladder rung, 78, 78f

Ladder window, 84
properties, 84f
Language, 6, 60, 73–75, 74f
function block diagram programming, 74, 75, 75f
IEC 61131 standard, 73–74, 74f
instruction list programming language, 74, 74f
ladder diagram language, 74, 74f, 75
sequential function chart (SFC) programming language, 74, 74f, 75
structured text (ST), 74, 75, 75f
Last In, First Out (LIFO) instruction, 260
example, 260, 260f
FIFO stack operation *vs.,* 260, 260f
Latch coil, 108
Latching relays, 108–110, 108–110f
Latch instructions, CLX system, 343–345, 343–345f
Leakage current, 270
Leakage current, discrete I/O module, 28
Least significant bit (LSB), 41
Length, FAL instruction, 198
Less than (LES) instruction, 202, 203, 203f, 204–205, 204f
Less than or equal (LEQ) instruction, 202, 204, 204f
Level switches, 96, 96f
LIFO (last in, first out) instruction, 260
example, 260, 260f
FIFO stack operation *vs.,* 260, 260f
Light sensors, 99–101, 99–101f
bar code scanners, 101, 101f
fiber optic, 100–101, 100f
photoconductive cell, 99, 99f
photoelectric, 99–100, 100f
photovoltaic/solar cell, 99, 99f
scan technique, 100, 100f
LIM instruction. *See* Limit test (LIM) instruction
Limit switch, 95, 95f
Limit test (LIM) instruction, 202, 205, 205f, 206f

LiveData, SCADA, 314
Local area network (LAN), 302, 302f
Lockout and tagout procedures, 276
Logic, 74
Logical continuity, 72, 78f
Logic Command Word, 315
Logic gate, 54, 54f
  truth tables, 54
Logic gate circuits
  Boolean equation for, 59–60, 60f
  from Boolean expression, 59, 59f
Logic operators, 58f
Logic section, input circuit, 19
Logic Status Word, 316–317
Logic wiring error, 2
LogixDesigner, 324, 325f
LSB (least significant bit), 41

M

Magnetic reed switch, 99, 99f
Magnitude, PID controllers, 213
Main ladder program (file 2), 68
Main routine, CLX system, 327
  ladder logic program and, 338–341, 338–341f
Main screen display, 318
Maintenance, preventive, 276, 276f
Main window, RSLogix 500 software, 83–84, 83f
Mantissa, 50
Manually operated switches, 94–95, 95f
  dual in-line package (DIP) switches, 95, 95f
  pushbutton switches, 94, 95f
  selector switch, 95, 95f
Manual mode, water level control process in storage tank, 109
Manual PID control, 299
Masked Comparison for Equal (MEQ) instruction, 202, 205–206, 206f
Masked move (MVM) instruction, 194, 195, 195f, 378–379

Master control relay (MCR), 186, 268–269, 268f
hardwired electromechanical, 268–269, 268f
Master control reset (MCR) instruction, 173–175f, 173–176
Math instructions, 219–231
addition, 220–223, 221f, 222f
clear, 228
CLX system, 367–369, 367–369f, 373–379, 373–379f
Compute/Math menu tab, 220, 220f
convert from BCD (FRD), 220, 228, 228f
convert-to-BCD (TOD), 220, 228, 228f
division, 220, 225–227, 226–227f
file arithmetic operations, 230–231, 230–231f
multiplication, 220, 224–225, 224–225f
negate, 227–228, 228f
RSLogix 500 CPT (compute) instruction, 220–221, 220f
scale, 228–229, 229–230f
square root, 227, 227f
subtraction, 220, 223–224, 223f
word-level, 227–229
Matrix-style sequencer chart, 244, 246f
MCR instruction. *See* Master control reset (MCR) instruction
Mechanical counters, 146, 146f
Mechanically operated switches, 95–96, 95f
level switches, 96, 96f
limit switch, 95, 95f
pressure switches, 96, 96f
temperature switch/thermostat, 95–96, 96f
Mechanical sequencers, 238–239, 238f
cam-operated, 238, 238f, 239
dishwasher timed sequencer switch, 238–239, 239f
drum-operated, 238, 238f
operation, 238
Mechanical timing relays, 122–123, 122f
Memory
capacity measurement, 11
CLX system, 324, 324f

design, 31–32, 31f
EEPROM, 32, 33f
EPROM, 32
factors, 11
location, 32
nonvolatile, 32
organization of, 41
RAM, 32, 32f
ROM, 32
single bit, 76
size, 11
sizes, 31, 31f
types, 32–33, 33f
utilization, 32
volatile, 32
Memory map, 68
Memory section, processor module, 29
Memory structures, 68
rack-based, 68
tag-based, 68
Menu bar, 83
MEQ instruction. *See* Masked Comparison for Equal (MEQ) instruction
Metal oxide varistor (MOV) surge suppressor, 272–273, 273f
MicroLogix controller, 71, 71f
addressing, 79
Microprocessor, 29–30
Modbus, 311
Mode, FAL instruction, 198
Modes of operation
program mode, 85
remote mode, 85
run mode, 85
test mode, 85
three-position keyswitch, 85f
Module-defined structures, CLX system, 330–331, 330f
Modules. *See also* I/O modules

high-density, 18
Modulo (MOD) instruction, 227, 227f
Monitor, HMI, 34–35
Monitoring, 274–276
Monitor tags, CLX system, 332, 332f
Most significant bit (MSB), 41, 42
Motion control modules, 27, 27f
Motion control system, 300–301
bottle-filling process, 300, 300f
Motion module, 301
Motor-driven analog proportional control valve, 296f
Motor lock-out program, 164–165, 164f
Motor seal-in circuit, 107f
Motor starters, 93–94, 93f
Motor Start/Stop code, 316, 316f
Move (MOV) instruction, 194, 195, 195f
CLX system, 372–379, 373–379f
variable preset counter values, 196, 197f
variable preset timer values, 195–196, 196f
Move/Logical menu tab, 194, 194f
MSB (most significant bit), 41, 42
Multibit digital devices, 209
Multiple rung program, scan process and, 73f
Multiplication, 49
Multiplication instruction, 220, 224–225, 224–225f
Multitask PLC application, 11
MVM instruction. *See* Masked move (MVM) instruction

N

NAND gate symbol, 56, 56f
National Electrical Code (NEC), 270–271
National Electric Manufacturers Association (NEMA), 94, 267
Negate (NEG) instruction, 227–228, 228f
Negative numbers, 42, 42t
NEMA 12 enclosure, 267
NEQ instruction. *See* Not equal (NEQ) instruction

Nested branching, 80, 80f
contact program, 80f
program to eliminate, 80–81, 80f
Nested contact program, 80f
Nested subroutines, 179, 181f
Network communications. *See also* Data communications
access method, 305
bit-wide bus networks, 304
bridges, 305
bus topology, 303–304, 303f
byte-wide bus networks, 304
ControlNet, 308–309, 310f
Data Highway, 305, 306f, 307, 307f
device bus networks, 304
DeviceNet, 307–308, 308f, 309f
EtherNet/IP, 309, 310f, 311
Fieldbus, 311, 311f
gateways, 304–305
HART, 312–313, 312f
industrial networks, functionality levels, 302, 303f
I/O bus networks, 304
LAN, 302, 302f
Modbus, 311
OSI model, 304
peer-to-peer network, 305–306, 306f
process bus networks, 304
PROFIBUS-DP, 311–313, 312f
SERCOS, 311–312, 312f
server/client network, 305, 305f
star topology, 303, 303f
token passing network, 305, 305f
Network protocol, 304
Network scanner, 307, 308f
Node, 302
Noise, electrical, 269–270, 269f
Noise suppression, 269, 269f

Nominal current per input, discrete I/O module, 28
Nominal input voltage, discrete I/O module, 27–28
Nonretentive on-delay timer (TON), 133
Nonvolatile memory, 32
NOR gate symbol, 56, 56f
Normally closed (NC) contacts, 91, 92f
Normally closed (NC) pushbutton, 94
Normally closed, timed closed (NCTC) contact
  off-delay timer circuit, 123, 123f
Normally closed, timed open (NCTO) contact
  on-delay timer circuit, 122–123, 123f
Normally open (NO) contacts, 91, 92f
Normally open (NO) pushbutton, 94
Normally open, timed closed (NOTC) contact
  on-delay timer circuit, 122, 123f
Normally open, timed open (NOTO) contact
  off-delay timer circuit, 123, 123f
Not equal (NEQ) instruction, 202, 203, 203f
NOT function, 55–56, 56f, 57f, 194
Number systems and codes, 39–50
  ASCII code, 46–47, 46t
  binary arithmetic, 47–49
  binary coded decimal (BCD) system, 43–45, 44t, 45f
  binary system, 40–41, 40–41f, 40t
  comparisons, 40t
  decimal system, 40, 40f, 40t
  floating point arithmetic, 49–50, 49–50f
  Gray code, 45, 45t, 46f
  Hexadecimal (hex) numbering system, 40t, 43, 43f, 44t
  hexadecimal system, 40t, 43, 43f, 44t
  negative numbers, 42, 42t
  octal system, 40t, 43, 43f
  parity bit, 47, 47t
Numerical data I/O interfaces, 209–212
  analog devices, 209

multibit digital devices, 209
Numeric Mode, FAL instruction, 199
Numeric values, comparison, 202, 202f. *See also* Data comparison

O

Octal numbering system, 40t, 43, 43f
conversion to binary number, 43f
conversion to decimal number, 43f
numeric values, 44t
Odd parity bit, 47, 47t
Off-delay timer (TOF), 124
CLX system, 354–355, 354–355f
fluid pumping process, 132–133, 132f
hardwired TOF relay circuit, 130–132, 131f, 132f
instruction, 130–133
operation, 130, 130f
for switching motors off, 130, 131f
Off-delay timer circuit
normally closed, timed closed (NCTC) contact, 123, 123f
normally open, timed open (NOTO) contact, 122–123, 123f
Offline programming, 274
OFF position, water level control process in storage tank, 109
OL relay. *See* Overload (OL) relay
On-delay timer (TON), 123
accumulated value (ACC) word, 126, 129, 129f
application, 127, 128f
CLX system, 351–354, 351–354f
control bits, 127
control word, 125
fluid pumping process, 132–133, 132f
instruction, 124–130
preset value (PRE) word, 125
principle of operation, 124–125, 125f
RSLogix 500 timer file, 125, 125f
RSLogix 500 timer table, 127, 127f
On-delay timer circuit

normally closed, timed open (NCTO) contact, 123, 123f
normally open, timed closed (NOTC) contact, 122, 123f
On-delay timer program, 208, 208f
1-K word memory, 41, 41f
1's complement number, 42, 42t
One-Shot Falling (OSF) instruction, CLX system, 346, 346f
One-Shot (ONS) instruction
CLX system, 345–346, 345–347f
up-counter, 153–154, 154f
One-shot rising (OSR) instruction, 153–154, 153–154f
CLX system, 346, 346f
Online programming, 274
On/off control, 213, 295–296, 295f
Open architecture, 4
Open-loop motor control system, 105, 105f
⊣⊢ symbol, 8
Open Systems Interconnection (OSI model), 304
Optical encoder, 103, 103f
incremental, 161–162, 162f
Optical encoder disk, 45, 46f
Optical isolator, 6
OR function, 55, 55f, 56f, 194
OR gate, 55, 55f
OTL instruction. *See* Output latch (OTL) instruction
OTU instruction. *See* Output unlatch (OTU) instruction
Output actuator, 294
Output branching, 80, 80f
with conditions, 80, 80f
nested, 80, 80f
Output conditions, 61
Output control devices, 103–105, 104–105f
actuator, 103
operation, 103–104
servo motors, 105, 106f
stepper motors, 105, 105f

symbols for, 104f
Output current/voltage range, 29
Output devices, 6
inputs and, relationships between, 2, 3f
Output Energize (OTE) instruction, 77–78, 77f
Output file, 68
Output image table file, 70–71, 70f
Output instructions, 202. *See also* Data transfer
Output latch (OTL) instruction, 108, 109f
CLX system, 343–345, 343–345f
Output malfunctions, troubleshooting, 278–280, 279f
Output troubleshooting guide, 284f
Output unlatch (OTU) instruction, 108, 109f
CLX system, 343–345, 343–345f
Output voltage, discrete I/O module, 29
Overflow bit, 222
Overflow (OV) bit, 149
Overload (OL) relay, 94
function of, 93, 93f
Override instructions, 173

P

PAC. *See* Programmable automation controller (PAC)
PanelView graphic terminals, 294f
Parallel data transmission, 306, 306f
Parity bit, 306
Parking garage counter, 156, 156f
PASCAL, 74
PC. *See* Personal computer (PC)
Peer-to-peer network, 305–306, 306f
Periodic tasks, 326
Personal computer (PC), 6, 6f, 33, 33f
memory. *See* Memory
vs. PLCs, 10–11, 10–11f
Photoconductive cell, 99, 99f
Photoelectric sensors, 99–100, 99f

Photoresistive cell, 99
Photovoltaic cell, 99, 99f
Pick and Place machines, 300, 300f
PI control. *See* Proportional plus integral (PI) control
PID control. *See* Proportional-integral-derivative (PID) control
Pins, FBD, 382, 383f
PLC-based control panel, 2, 2f
PLC drink-manufacturing program, 201–202, 202f
PLC programming language. *See* Language
PLCs. *See* Programmable logic controllers (PLCs)
Plug and play, 311
Pneumatic on-delay timer, 122f
Points per module, discrete I/O module, 28
Point-to-point serial communications link, 301, 301f
Polling, 305
Position and distance measurement, 27
Position control modules, 27, 27f
Position, FAL instruction, 198
Position sensors, 103, 103f
Positive temperature coefficient (PTC), of metals, 102
Power circuit, 92
Power section, input circuit, 19
Power supply, 4, 5, 5f
  alternating current (AC), 5
  CPU, 29, 30f
  direct current (DC), 5
Predefined structures, CLX system, 330, 330f
  programming timers, 350–351, 350f
Preset counter values, MOV instruction and, 196, 197f
Preset time, 124
Preset timer values, MOV instruction and, 195–196, 196f
Preset value, 125
  counter, 149
Preset value (PRE) word
  timers, 125
  up-counter, 149

Pressure switches, 96, 96f
Preventive maintenance program, 276, 276f
Principles of operation, 6–9, 7–9f
  modifying, 9, 9f
Process(es). *See also* Control systems
  batch, 291, 291f
  centralized control, 291, 292f
  continuous, 291, 291f
  control, 291
  discrete manufacturing, 291, 291f
  distributive control system, 291, 292f, 293
Process bus networks, 304
Process control, 291. *See also* Control systems
  centralized, 291, 292f
  distributive control system, 291, 292f, 293
  individual, 291, 292f
Process control problem, 6–7, 7f
  ladder logic program, 7–8, 8f
  relay method for, 7, 7f
  RUN operation, 8
  wiring connections, 7f, 8–9, 9f
Processor (CPU), 4–6
Processor memory organization, 68–71, 69f
  data files, 68–71, 69–71f
  program files, 68, 69f
Processor module, 29f, 30f
  CPU section, 29
  memory section, 29
  PROG position, 30
  REM position, 30
  RUN position, 30
  troubleshooting, 277, 277f
Process variable (PV), 214, 294
Produced/consumed tags, 329, 329f
Product part flow rate program, 165, 165f

PROFIBUS-DP, 311–313, 312f
Program, 6
Program control instructions, 172–188. *See also* Instructions
fault routine, 188
forcing external I/O addresses, 182–185, 183–184f
immediate I/O instructions, 181–182, 182f
jump instruction, 176–178, 176–178f
master control reset, 173–175f, 173–176
safety circuitry, 185–187f, 185–188
subroutine functions, 178–181, 178–181f
temporary end, 188, 188f
Program files, 68, 69f
Programmable automation controller (PAC), 10, 323, 323f
Programmable logic controllers (PLCs), 319
advantages, 2
applications, 2, 11
architecture, 4
benefits, 2–4
closed architecture, 4
communication, 320
communications capability, 3
defined, 2
easier to troubleshoot, 3
faster response time, 3
flexibility, 3
hardware, 4
human machine interface, 318–320
industrial application, 4
instruction set, 11, 11t
I/O system, 4–6, 4f, 5f, 6f
lower cost, 3
modifying operation, 9, 9f
motion control process, 300
open architecture, 4
overview, 1–11
parts of, 4–6, 4f

power supply, 5, 5f
principles of operation, 6–9, 7–9f
processor (CPU), 5–6
program, 6
programming device, 6
programming language, 6
proprietary, 4
as real-time system, 2
relay logic, 2
reliability, 2–3
size, 11
software, 4, 10
status bits to, 315
Variable Frequency Drive with, 314–317, 315f
vs. computers, 10–11, 10–11f
Programmed logic, hardwired logic *vs.,* 60–61
Programming, 274–276
addressing, 79, 79f
analog devices, connecting with, 85–86, 85–86f
bit-level logic instructions, 76–78
branch instructions, 79–81, 80f
Examine If Closed (XIC) instruction, 82–83, 82f
Examine If Open (XIO) instruction, 82–83, 82f
internal relay instructions, 81, 81f
ladder diagram, 83–85, 84f
languages, 73–75, 74f
modes of operation, 85, 85f
offline, 274
online, 274
processor memory organization, 68–71, 69f
data files, 68–71, 69–71f
program files, 68, 69f
program scan cycle, 71–73, 71f
Programming counters. *See* Counters
Programming device, 4, 6
Programming language. *See* Language

Programming timers. *See* Timers
Program mode, 85
Program/processor status toolbar, 83
Programs
  CLX system, 326–327, 327f
  data manipulation, 206–209, 207–209f
Program scan cycle, 71–73, 71f
  CLX system, 335–336, 335f
  data flow, 72, 72f
  horizontal scan, 73
  multiple rung program, 73f
  time, 72
  vertical scan, 73
Program tag, 328
Project, CLX system, 325–326, 325–326f
Project tree, 83
Project window, 83
PROM (programmable read-only memory), 32
Proportional band, for heating application, 297, 297f
Proportional control/controllers, 296–300, 296f
  derivative action and, 297
  droop, 297
  integral action and, 297
  offset, 297
  steady-state error, 297, 297f
Proportional controls, 213, 213f
Proportional-integral-derivative (PID) control, 213–214, 214f, 296–300, 298–300f
  fully automatic/intelligent, 299
  manual, 299
  output instruction and setup screen, 300f
  response of, 298–299
  semiautomatic/autotune, 299
Proportional plus derivative (PD) control, 298
Proportional plus integral (PI) control, 297–298
Protocol, 304
Proximity sensors, 96–98f, 96–99

application conditions, 96–97
bleeder resistor connected to, 98, 98f
capacitive, 98–99, 98f
inductive-type, 97, 97f
sensing range, 97, 98f
three-wire DC, 97, 97f
two-wire, 97, 97f
Pulse width modulation, 296
Pumping process, fluid, 132–133, 132f
Pushbutton interlocking, 107–108, 108f
Pushbutton switches, 94, 95f
PV. *See* Process variable (PV)

R

Rack-based I/O section, 15, 15f
Rack-based memory structures, 68
Rack-based systems, 68
Rack/slot-based addressing, 15, 15f
Radiated noise, 269
Random Access Memory (RAM), 32, 32f
Rate of change, PID controllers, 213
Reading, memory, 31
Read Only Memory (ROM), 32
Read-write (R/W) memory, 32
REAL (base tag), 330
Real-time system, PLC as, 2
Real-world devices, 6
Reciprocating timers, 136–137
CLX system, 357
Recommended standard, 307
Recording, data, 33
Redundant media, 309
Redundant processors, 30
References, FBD, 382, 382f
Registers, 194
Relay-based control panel, 2, 3f
Relay ladder logic (RLL) program, 6

diagram for modified process, 9f
Relay logic, 2, 6
Relay operation, 91, 91f
Relay schematics
conversion to PLC ladder programs, 111–114
for sequential control process, 112, 112f
Release coil, 108
Reliability, 2–3, 2f
Remote I/O rack, 15, 15f
Remote mode, 85
Repeatability, data communication and, 308
Repeater, 303
Reserved (file 1), 68
Reset (RES), 124
Resistance temperature detectors (RTDs), 102, 102f
Resolution, analog I/O modules, 29
Resolution, of analog input channel, 24
Response time, 3
Result window, 83
Retentive on-delay (RTO) timer, 124
alarm program, 134–135, 134f
application, 135
CLX system, 356–357, 356–357f
program, 133–134, 133f
timing chart, 134, 134f
Retentive timer, 133–135
Retentive timer reset (RES) instruction, 133
Retrieval, data, 33
Retroreflective scan technique, 100, 100f
Return (RET) output instruction, 179
Rotary switches, 238
Rotating cam limit switch, 238f
Routines, CLX system, 327, 327f
Routing, wire, guidelines for, 269–270
RSLinx Enterprise, 318
RSLinx software, 284–285, 324, 325f

RSLogix 500 controllers
control word, 149
counter file, 147, 149f
counter instructions, 151, 152f
counter table for, 149, 149f
count-up counter, 150f
CPT (compute) instruction, 220–221, 220f. *See also* Math instructions
input image table file, 69–70, 70f
on-delay timer instruction, 125–126, 125f
one-shot rising (OSR) instruction, 153–154, 153–154f
output image table file, 70–71, 70f
program and data file organization, 68, 69f
subroutine functions, 179, 180f
timer file, 125, 125f
timer selection toolbar, 124f
timer table, 127, 127f
RSLogix 5000 controllers, 16
BSL instruction, 256–257, 257f
BSR instruction, 256
counter instructions, 151
FIFO instruction pair, 260–261, 261f
immediate output instruction, 182
memory structure, 68
Modulo (MOD) instruction, 227, 227f
pass/fail inspection program, 256–257, 257f
tag-based addressing format, 16, 17f
timer instruction, 126, 126f
RSLogix 500 controller word and file addressing, 197–198, 198f
RSLogix 500 FIFO load (FFL) instruction, 258, 258f
RSLogix 500 FIFO unload (FFU) instruction, 258–259, 258f
RSLogix 5000 program, 55
Exclusive OR (XOR) logic function, 57
AND logic function, 55
NAND logic function, 57
NOT logic function, 56
three-wire motor control circuit, 16, 18f

RSLogix 500 Scale data (SCL), 228–229, 229f
RSLogix 5000 Sequencer Output (SQO) instruction, 243–244, 244f, 245f
  array, 244
  control, 244
  DEST, 244
  length, 244
  mask, 244
  position, 244
RSLogix software, 83, 284, 285, 324, 324f
  counter selection tool bar, 150f
  Data File screens, 84, 84f
  instruction toolbar with bit, 83f
  I/O Configuration screen, 84, 84f
  main window, 83–84, 83f
  project, 325
  Select Processor Type screen, 84, 84f
  sequencer menu tab for, 240f
RS-232 standard, 285
RSWho, 285, 286f, 324
RTDs (resistance temperature detectors), 102, 102f
RTO timer. *See* Retentive on-delay (RTO) timer
Rung, 61
Run mode, 85

S

Safety circuitry, 185–187f, 185–188
Safety PLCs, 186, 186f
  vs. standard PLC, 186
Safety requirements, 185–186, 185f
Same address, program with, 281–282, 281f
SCADA (supervisory control and data acquisition), 30–31, 302, 313–314, 313f
  alarm system, 313–314, 314f
  alert, 314
  FactoryTalk services platform, 314, 314f
Scale instruction, 228–229, 229–230f
Scale with Parameters (SCP) instruction, 228–229, 229–230f

Scan, 8
Scan cycle time, 72
Scan process
  data flow, 72, 72f
  horizontal scan, 73
  multiple rung program, 73f
  vertical scan, 73
Scan technique, 100, 100f
Scan time, 8
Scope, defined, 328
Seal-in circuits, 106, 106f
Security, SCADA, 314
Selection toolbar, timer, 124f
Selector switch, 95, 95f
Select Processor Type screen, RSLogix 500 software, 84, 84f
Self-detection, 277
Semiautomatic/autotune PID control, 299
Sensing range, proximity sensors, 97, 98f
Sensors, 25f, 96–103. *See also* Switches
  in control systems, 293
  flow measurement, 102–103, 103f
  light, 99–101, 99–101f
  magnetic reed switch, 99, 99f
  position, 103, 103f
  proximity, 96–98f, 96–99
  strain/weight, 102, 102f
  temperature, 102, 102f
  ultrasonic, 101, 101f
  velocity, 103, 103f
Sequencer chart, 244, 246f
Sequencer compare (SQC) instruction, 247–248, 249f
Sequencer instructions, 240–244, 240f. *See also* Instructions
  four-step sequencer, 241, 242f
  mask word, 241
    moving data through, 241–242, 243f
  moving data from file to output, 242–243, 243f

RSLogix 5000 Sequencer Output (SQO) instruction, 243–244, 244f, 245f
sequencer menu tab, 240f
SQC instruction, 247–248, 249f
SQL instruction, 248–249, 250f
SQO instruction, 240–244, 240f, 242–244f
Sequencer load (SQL) instruction, 248–249, 250f
Sequencer Output (SQO) instruction, 240–244, 240f, 242–244f
control, 241
destination, 241
file, 240
length, 241
mask, 240–241
position, 241
source, 241
Sequencer programs, 244–249
event-driven, 244, 245f, 247, 248f
sequencer chart, 244, 246f
SQC instruction, 247–248, 249f
SQL instruction, 248–249, 250f
time-driven, 244, 245f, 246, 247f
Sequencer switch, 238–239, 238f
cam-operated, 238, 238f, 239
dishwasher timed sequencer switch, 238–239, 239f
drum-operated, 238, 238f
operation, 238
Sequential control process, 111, 111f, 112f
flow diagram, 111, 112f
Sequential function chart (SFC) programming language, 74, 75, 75f
Sequential time-delayed motor-starting circuit
hardwired, 136f
PLC program of, 136f
SERCOS (Serial Real-time Communications System), 311–312, 312f
Serial data communication, 301, 307, 307f
point-to-point link, 301, 301f
Serial data transmission, 306, 306f

Server, 318
Server/client network, 305, 305f
Servo drive, motion control process, 301
Servo motors, 105
  closed-loop control, 105
  motion control process, 301
  open loop control, 105, 105f
Set Forward Direction bit, 316
Set-point, 214, 294
Set-point control, 212–213, 212f
Set Reverse Direction bit, 316
Seven-segment LED display board, 209–210, 210f
Shift registers
  bit, 249–257
  circulating shift register function, 250
  concept of, 250, 251f
  data in, 250
  in material handling processes, 253, 254f
  spray-painting operation, 253–254, 255f
  types of, 250, 251f
Short circuit protection, discrete I/O module, 28
Signal conditioning, in control system, 293
Signals, 85
Sign bit, 42, 223
Signed binary numbers, 42t
Sign, of binary floating-point number, 49
Single bit, 76, 209
Single-ended connections, 29
Single-ended PLC application, 11
Single precision, 50, 50f
Single-scan test mode, 85, 274
Single-step test mode, 85
Sinking I/O device, 22–23, 22f
SINT (base tag), 330
Six-axis robot arm, 301f
16-bit word, 41f

Size, PLCs, 11
memory, 11
SLC 500 controllers, bit level and word level addressing, 15, 17f
SLC family, of PLCs I/O address format, 69, 70f
Smart instruments, 117, 117f
SoftLogix 5800 controller, 323
Software, 4, 10, 284–285, 285–286f
Solar cell, 99, 99f
Solenoid, 104, 104f
Solenoid valve, 104
construction and operation, 104, 104f
Source register, 195
Sourcing I/O device, 22–23, 23f
Speed Feedback Word, 317
Speed Reference Word, 316
Split bar, 84
Spray-painting operation, by shift left register, 253–254, 255f
Square root (SQR) instruction, 227, 227f
Stack, 257
Stand-alone PLC application, 11
Standard PLCs, *vs.* safety PLCs, 186
Start Drive bit, 316
Star topology network, 303, 303f
Start/Stop code, 316, 316f
Station, 302
Status bar, 83
Status bits, to PLC, 315
Status file, 68
Stepper-motor module, 27
Stepper motors, 105, 105f
Stepper switches, 238
Stop buttons, wiring of, 187, 187f
Stop Drive bit, 316
Storage tank, water level control process in, 110, 110f
automatic mode, 109–110
manual mode, 109

OFF position, 109
program used for, 109–110
status indicating lights, 110
Strain gauge, 102, 102f
Strain gauge load cells, 102
Structured text (ST), 74, 75, 75f
Structure-type tag, CLX system, 330, 330f
Studio 5000 programming, 318
Subroutine functions, 178–181, 178–181f
Subroutine (SBR) input instruction, 179
Subroutine ladder program (files 3-255), 68
Subroutines, 327
Subtraction, 47–48
Subtraction instruction, 220, 223–224, 223f
Supervisory control and data acquisition (SCADA), 30–31, 302, 313–314, 313f, 314f
alarm system, 313–314, 314f
alert, 314
FactoryTalk services platform, 314, 314f
Suppression device, 272
Surges, 272–273, 273f
Suspend (SUS) instruction, 281, 281f
Switches. *See also* Sensors; Specific types
manually operated, 94–95, 95f
mechanically operated, 95–96, 95f
positions, 95
-( )- symbol, 8
Symbol factory, 318
System functions (file 0), 68

T

Tabbed instruction toolbar, 84
Table, 194
Tachometer generators, 103, 103f
Tag-based addressing, 16, 17f
CLX system, 337, 338f
rack/slot-based addressing *vs.,* 15

Tag-based memory structures, 68
Tag-based systems, 68
Tag database, 318
Tags, 16, 79. *See also* specific types
  CLX system, 327–330, 328–330f
    creating, 331, 331f
    edit tags, 332, 332f
    monitor tags, 332, 332f
Tank-filling process, analog control for, 86, 86f
Tasks, CLX system, 326, 326f
  continuous, 326
  event, 326
  periodic, 326
Temperature measurement, 26, 26f
Temperature sensors, 102, 102f
Temperature switch, 95–96, 96f
Temporary end (TND) instruction, 188, 188f, 280–281, 281f
Test mode, 85
Thermocouples, 102, 102f
Thermostat, 95–96, 96f
Three-phase magnetic motor starter, 93–94, 94f
Three-pole magnetic contactor, 93f
Three-wire DC proximity sensor, 97, 97f
Through-beam scan technique, 100, 100f
Thumbwheel switches (TWS), 44–45, 45f, 209, 210f
Time base, 124, 125
Timed contact symbols, 122, 122f
Time-driven sequencer program, 244, 245f, 246, 247f
Time proportioning, 296–297, 296f
Timer file, 68
Timer function, and counter function, 162–165, 364–365, 365f
Timer number, 125
Timers, 74, 121–139, 136–139f
  advantages, 125
  cascading timers, 136–138, 138f
  CLX system, 350–358

cascading, 357, 358f
off-delay timer (TOF), 354–355, 354–355f
on-delay timer (TON), 351–354, 351–354f
predefined structure, 350–351, 350f
reciprocating, 357
retentive on-delay timer (RTO), 356–357, 356–357f
instructions, 123–124, 124f
mechanical timing relays, 122–123, 122f
off-delay timer instruction, 130–132f, 130–133
on-delay timer instruction, 124–130, 125–127f
quantities, 124
reciprocating, 136–137
retentive timer, 133–134f, 133–135
selection toolbar, 124f
Timer-timing (TT) bit, 125
Timing relays, 122, 122f
Title bar, 83
TOF. *See* Off-delay timer (TOF)
Token passing network, 305, 305f
TON. *See* On-delay timer (TON)
Traffic lights, control of
in one direction, 137–138, 138f
timing chart, 138f
in two directions, 138, 138f, 139f
Transducer, 24–25
Transmitter, 25, 25f
Trend monitoring, 35, 35f
Troubleshooting, 3, 275–284
for discrete output module, 282, 283f
general methods of, 282, 282f
input guide, 283f
input malfunctions, 277–278, 278f
ladder logic program, 280–282, 280–284f
output guide, 284f
output malfunctions, 278–280, 279f
processor module, 277, 277f

Truth tables, 54
NAND gate symbol and, 56, 56f
NOR gate symbol and, 56, 56f
XOR gate symbol and, 57, 58f
Turbine flowmeter, 102–103, 103f
24-hour clock program, 160, 160f
Twisted pairs of wire, 302, 302f
Two-position control, 213
2's complement number, 42, 42t
Two-wire proximity sensor, 97, 97f

U

Ultrasonic sensors, 101, 101f
Underflow (UN) bit, 149
Unipolar modules, 24
Unlatch coil, 108
Unlatch instructions, CLX system, 343–345, 343–345f
Unload bit, 252
Up-counter, 146–155, 148f
alarm monitor program, 154f
one-shot instruction, 151–155, 152f
one-shot rising (OSR) instruction, 153–154, 153–154f
Up-counter program, 208–209, 209f
Update accumulator (UA) bit, 149
Up/down-counter, 155–158, 155–158f
User-defined structure, CLX system, 331, 331f

V

Variable Frequency Drive (VFD), 314–317, 315f
commands to, 315
Velocity sensors, 103, 103f
Verifying program errors, 280, 280f
Vertical contact
program with, 80, 81f
reprogrammed to eliminate, 81, 81f
Vertical scan, 73

Vessel filling operations, 209
VFD. *See* Variable Frequency Drive (VFD)
Volatile memory, 32
Voltage sensing, 23–24
Voltage variations, 272–273, 273f

W

Watchdog timer, 277
Weight sensors, 102
Windows environment, 83
Windows toolbar, 83
Wire connectors, FBD, 382–383, 383f
Wire identification sleeves, heat-shrinkable, 270, 270f
Wireless Wi-Fi Ethernet networks, 302
Wire routing
  guidelines for, 269–270
  heat-shrinkable wire identification sleeves, 270, 270f
Wires/wiring
  FBD, 382, 383f
  of stop buttons, 187, 187f
Wiring connections, process control scheme, 7f, 8–9, 9f
Word level addressing, 16, 17f
Word-level logic instructions, 63–64, 63–64f, 63t
Word-level math instructions, 227–229. *See also* Math instructions
Words, 31–32, 41, 194
Word shift operations, 257–261
Word-to-file moves, 197
  FAL instruction and, 200, 200f
Wraparound operation, BSL instruction with, 252–253, 254f
Writing, memory, 31
Writing over the existing data, 194

X

XOR (exclusive-OR) function, 57, 58f, 194

Z

Zero Bit, 222



OceanofPDF.com

Book Cover Graphic Extended Description (Front Matter)

Return to Book Cover Graphic

The author’s name Frank Petruzella labeled at bottom right. The cover features a photograph of a series of programmable logic controllers (P L Cs) mounted on a rack. The PLCs have various digital displays and control panels. The publisher’s logo, McGraw Hill, is visible at the bottom left corner.

Return to Book Cover Graphic

OceanofPDF.com

The Image Shows Graphic Extended Description (Front Matter)

Return to The Image Shows Graphic

The text below reads:

After completing this chapter, you will be able to:

Bullet point: Describe the operation of pneumatic on-delay and off-delay timers

Bullet point: Describe PLC timer instruction and differentiate between a nonretentive and retentive timer

Bullet point: Convert fundamental timer relay schematic diagrams to PLC programs

Bullet point: Analyze and interpret typical PLC timer programs

Bullet point: Program the control of outputs using the timer instruction control bits.

Return to The Image Shows Graphic

OceanofPDF.com

A Ladder Diagram Graphic Extended Description (Front Matter)

Return to A Ladder Diagram Graphic

There are four vertical lines. From the left, the first vertical line is labeled L 1 and the fourth vertical line is labeled L 2. As an input, motor stop and start switches with a light board consisting of stop and start lights, an enter L S switch, and a exits L S switch are connected from the right to line L 1. Three rungs are connected between the second and the third vertical lines with a label ladder logic program placed over the first rung. In the first rung, motor stop with address Local:1:I.Data.0, motor start with address Local:1:I.Data.1, and track motor with address Local:2:O.Data.0 are connected in series. A track motor with address Local:2:O.Data.0 is connected in parallel across motor start with address Local:1:I.Data.1. An enter L S with address Local:1:I.Data.2 and wash in use L in series is connected across the track motor with address Local:2:O.Data.0. In the second rung, wash in use and do not enter sign with address Local:2:O.Data.1 are connected in series. In the third rung, exit L S with address Local:1:O.Data.3 and wash in use U are connected in series. On the right, do not enter the sign and a track motor from the left are connected to line L 2. Below it, a car entering through a gate with a sign board above it labeled do not enter is depicted.

Return to A Ladder Diagram Graphic

OceanofPDF.com

An Illustration Shows Graphic Extended Description (Front Matter)

Return to An Illustration Shows Graphic

A rectangle has a dashed square on the left with its base the same as the rectangle. This dashed square is labeled H M I package. Within it, a blank table labeled Tag Database is depicted, and it is connected to an I slash O server on its right with a double-headed arrow. Below the Tag Database table, a graphic screen with start and stop lights is connected with a vertical double-headed arrow. On the right of the rectangle, a P L C system is depicted, which is connected to the I slash O server with communication ports.

Return to An Illustration Shows Graphic

OceanofPDF.com

A Schematic Diagram Shows Graphic Extended Description (Front Matter)

Return to A Schematic Diagram Shows Graphic

On the left, the diagram labels a conventional system, where an input slash output module is depicted at the top. Ten different components are connected by various cables to different slots of this input or output module. On the right, a DeviceNet scanner module depicted at the top is connected to a thick cable. Ten different devices are connected to this thick cable at different points.

Return to A Schematic Diagram Shows Graphic

OceanofPDF.com

The Screenshot Graphic Extended Description (Front Matter)

Return to The Screenshot Graphic

Top: The heading reads Creating Ladder Logic. The paragraph below it reads: Although other programming languages are available, ladder logic is the most common programming language for PLCs. The instructions in ladder logic programming can be divided into two broad categories: input and output instructions. The most common input instruction is equivalent to a relay contact and the most common output instruction is the equivalent of a relay coil (Figure 15-27). When creating ladder, I slash O bit instructions, the following rules apply: All input instructions must be to the left of an output instruction. A rung cannot begin with an output instruction if it also contains an input instruction. This is because the controller tests all inputs for true or false before deciding what value the output instruction should be. A rung does not need to contain any input instructions, but it must contain at least one output instruction.

Bottom: A table on the left has two column headers: If the Data Bit is: and X I C with its symbol instruction is:. The first row data is Logic 0 and False. The second row data is Logic 1 and True. Another table on its right has two column headers: If the Data Bit is: and X I C with its symbol instruction is:. The first row data is Logic 0 and True. The second row data is Logic 1 and False. Below these tables, a boxed XIC is connected from the bottom of the first column of the first table, a boxed Input is below the second column of the first table, a boxed X I O is connected from the bottom of the first column of the second table, and a boxed Output is below the second column of the second table.

A rung below these tables has X I C, X I O, and Output connected in series. Below this rung, on the bottom right, another table has two column headers: If the Data Bit is: and “O T E” with its symbol instruction is:. The first row data is Logic 0 and False. The second row data is Logic 1 and True. Above the second column of this table, a boxed “OTE” is connected by a vertical line.

Return to The Screenshot Graphic

OceanofPDF.com

An Illustration Graphic Extended Description (Front Matter)

Return to An Illustration Graphic

Top: There are four vertical lines. From the left, the first vertical line is labeled L 1 and the fourth vertical line is labeled L 2. As an input, the timer switch is connected to line L 1 from the right. A photo of the switch with ON and OFF sign is depicted below it. Four rungs are connected between the second and the third vertical lines with a label ladder logic placed over the first rung.

In the first rung, timer switch with address Local:1:I.Data.6, T O N box, and E N are connected in series. D N from the right side of the T O N box is connected. From the top, T O N box labels T O N, time on delay, timer with status timer on its right, present with 10000 on its right, and accumulated with o on its right are shown. In the second rung, status timer E N and E N pilot light P L with address Local:2:O.Data.1 are connected in series. In the third rung, status timer T T and T T pilot light P L with address Local:2:O.Data.2 are connected in series. In the fourth rung, status timer D N and D N pilot light P L with address Local:2:O.Data.3 are connected in series. On the right, E N, T T, and D N pilot lights from the left are connected to line L 2. An electric lamp with three kinds of pilot lights is shown below it.

Bottom: The F B D equivalent diagram consists of a box labeled T O N R_01 that has the following labels: T O N R with a box with three dots on the right, time on delay with reset, time enable with A C C labeled on the right, P R E with E N labeled on the right, T T labeled on the right, reset with D D labeled on the right. Inputs for time enable, P R E are 0 and 10000 depicted on the left. The inputs for A C C, E N, T T, and D N are 0, 0, 0, and 0, depicted on the right.

Return to An Illustration Graphic

OceanofPDF.com

The Book Page Graphic Extended Description (Front Matter)

Return to The Book Page Graphic

The first page consists of 8 questions on the left side and 10 questions on the right side as follows:

On the left side:

1. Explain the basic operating principle of an electromagnetic control relay.
2. What is the operating difference between a normally open and a normally closed relay contact?
3. In what ways are control relay coils and contacts rated?
4. How do contactors differ from relays?
5. What is the main difference between a contactor and a magnetic motor starter?
    a. Draw the schematic for an across-the-line A C magnetic motor starter.
    b. With reference to this schematic, explain the function of each of the following parts:
    i. Main contact M
    ii. Control contact M
    iii. Starter coil M
    iv. OL relay coils
    v. OL relay contact
6. The current requirement for the control circuit of a magnetic starter is normally much smaller than that required by the power circuit. Why?
7. Compare the method of operation of each of the following types of switches:
    a. Manually operated switch
    b. Mechanically operated switch
    c. Proximity switch

On the right side:

15. Compare the operation of the reflective-type and through-beam photoelectric sensors.
16. Give an explanation of how a scanner and a decoder act in conjunction with each other to read a bar code.
17. How does an ultrasonic sensor operate?
18. Explain the principle of operation of a strain gauge.
19. Explain the principle of operation of a thermocouple.
20. What is the most common approach taken with regard to the measurement of fluid flow?
21. Explain how a tachometer is used to measure rotational speed.

22. How does an optical encoder work?
23. Draw an electrical symbol used to represent each of the following PLC control devices:
    a. Pilot light,
    b. Relay,
    c. Motor starter coil,
    d. O L relay contact
    e. Alarm (on the left)
    f. Heater,
    g. Solenoid,
    h. Solenoid valve,
    i. Motor,
    j. Horn (on the right)
24. Explain the function of each of the following actuators:
    a. Solenoid
    b. Solenoid valve

Return to The Book Page Graphic

OceanofPDF.com

The Book Page Graphic Extended Description (Front Matter)

Return to The Book Page Graphic

The page consists of 3 questions on the left side and one question on the right side as follows:

On the left side:

1. Design and draw the schematic for a conventional hardwired relay circuit that will perform each of the following circuit functions when a normally closed pushbutton is pressed:
   The following options are listed below in bullets: Switch a pilot light on, De-energize a solenoid, Start a motor running, and Sound a horn.
2. Design and draw the schematic for a conventional hardwired circuit that will perform the following circuit functions using two break-make pushbuttons:
   The following options are listed below in bullets: Turn on light L 1 when pushbutton P B 1 is pressed, Turn on light L 2 when pushbutton P B 2 is pressed, and Electrically interlock the pushbuttons so that L 1 and L2 cannot both be turned on at the same time.
3. Study the ladder logic program in Figure 6-74, and answer the questions that follow:
   a. Under what condition will the latch rung 1 be true?
   b. Under what conditions will the unlatch rung 2 be true?
   c. Under what condition will rung 3 be true?
   d. When PL1 is on, the relay is in what state (latched or unlatched)?
   e. When PL2 is on, the relay is in what state (latched or unlatched)?
   f. If AC power is removed and then restored to the circuit, what pilot light will automatically come on when the power is restored?
   g. Assume the relay is in its latched state and all three inputs are false. What input change(s) must occur will correctly execute the hardwired control circuit in Figure 6-75.

   Assume: Stop pushbutton used is an NO type. Run pushbutton used is an NO type. Jog pushbutton used has one set of NO contacts. OL contact is hardwired.
   A figure at the top right of this page with caption ‘Figure 6-75 Hardwired control circuit for Problem 4.’ is shown.
   In the figure, a stop, run, a motor, and O L are connected in series across L 1 on the left and L 2 on the right. Below, a Jog key and a motor are in series. A separate open key is connected in parallel to this combination. All of this setup is connected in parallel across the run switch.
5. Design a PLC program and prepare a typical I slash O connection diagram and ladder logic program that will correctly execute the hardwired control circuit in Figure 6-76.
   Assume: P B 1 pushbutton used is an NO type.
   P B 2 pushbutton used is an NC type.
   P S 1 pressure switch used is an NO type.
   L S 1 limit switch used has only one set of NC contacts.

The diagram below this question shows a rung consisting of a start pushbutton P B 1, a stop pushbutton P B 2, and an encircled C R 1 connected in series. This combination is further connected to line L 1 on the left and line L 2 on the right.

Return to The Book Page Graphic

OceanofPDF.com

Mcgraw Hill Connect and Proctorio Logos Graphic Extended Description (Front Matter)

Return to Mcgraw Hill Connect and Proctorio Logos Graphic

The McGraw Hill Connect logo comprises McGraw Hill written within a red box on the left side and the superscript registered mark on Connect on the right. Adjacent to it is a plus sign, followed by four rectangles of varying shapes connected at a common point. To the right of these shapes is the text proctorio, denoting the Proctorio logo.

Return to Mcgraw Hill Connect and Proctorio Logos Graphic

OceanofPDF.com

The Laptop Screen Showcases Graphic Extended Description (Front Matter)

Return to The Laptop Screen Showcases Graphic

Experience Psychology is labeled on the top left, and an exit assignment button with a cross sign is positioned on the top right. Below Experience Psychology, the term concepts is situated between leftward and rightward scroll buttons. Beneath it, leftward and rightward arrows are shown, with a font size adjuster symbol positioned at the far right of these buttons. Following this, a seven-line paragraph is shown, with a Page symbol with two dollar symbols depicted on the right side of the paragraph. From the second sentence in the first line to three sentences covering half of the four lines in the paragraph, the text is highlighted in blue. The remainder of the paragraph from this point to the bottom is highlighted in yellow. Below the paragraph, an image is featured, depicting a smiling girl holding a dog. To the right of the girl's face, the terms vision, taste, smell, hearing, balance, and touch are labeled from top to bottom. On the left side of the girl and starting from the bottom of the image, an upward curved arrow labeled action potential leads to a sensory neuron, while another upward curved arrow from here leads to a brain image labeled sensation and perception.

Return to The Laptop Screen Showcases Graphic

OceanofPDF.com

The EVERGREEN Logo Graphic Extended Description (Front Matter)

Return to The EVERGREEN Logo Graphic

Two concentric circles are depicted, with the inner circle being white and the outer circle in blue. The outer circle is labeled evergreen in circular form at the top. In the lower portion of the outer circle, there is a clockwise curved arrow. Additionally, a tree symbol is situated within the inner circle.

Return to The EVERGREEN Logo Graphic

OceanofPDF.com

A Line with 5 Turns Is Shown Graphic Extended Description (Front Matter)

Return to A Line with 5 Turns is Shown Graphic

The line extends upward, then turns left and extends further. It then turns downward slightly, turns left for a short distance, turns upward in a straight line, and finally turns leftward for a short distance.

Return to A Line with 5 Turns is Shown Graphic

OceanofPDF.com

A Screenshot Graphic Extended Description (Chapter 1)

Return to A Screenshot Graphic

On the top left are backward and forward arrowhead buttons. To their right, there are 9 tabs: overview, blending, rapid mix, filling, labeling, packaging, CIP, alarm, and more. The packaging tab is selected. Below it, on the left, there are labels for clean in place, and on the right, there is date and time information.

Below left, there is a window titled CIP Valve depicting its features. On the right, there is a popup labeled CIP Cleaning Complete! depicted.

Below them, there is a schematic diagram of CIP cleaning. It consists of 4 tanks: chemicals, sanitizer, recovered water, and fresh water, with tank levels of 2000, 2000, 1250, and 6496 liters, respectively, connected through a pipe from the top. CIP return goes through the left end of the pipe, and a CIP clarity sensor is mounted above it. The right end of this pipe serves as a drain.

From the bottom, CIP supply through a CIP supply motor is distributed to all four tanks.

Return to A Screenshot Graphic

OceanofPDF.com

Figure 1-2 Extended Description (Chapter 1)

Return to Figure 1-2

A PLC memory with an arrow to a magnified user program circuit diagram. There are three capacitors in the circuit, with two parallel capacitors connected in series with the third capacitor. The combination is connected to a motor starter coil.

Return to Figure 1-2

OceanofPDF.com

Figure 1-3 Extended Description (Chapter 1)

Return to Figure 1-3

PLC memory with a user program circuit diagram, and labeled inputs at the bottom and outputs at the top. The inputs include three parts labeled pushbutton, limit switch, and sensor. The outputs include three parts labeled contactor, light, and solenoid.

Return to Figure 1-3

OceanofPDF.com

Figure 1-6 Extended Description (Chapter 1)

Return to Figure 1-6

The parts of a PLC are labeled and divided into the central processing unit (CPU) in the middle, the power supply module on the top, the input module on the left, the output module on the right. The programming device, labeled programmer, is at the bottom. The controller (CPU) includes a processor and memory. The input control devices to the left of the input module have a parallel combination of a limit switch, a pressure switch, closed pushbutton, and an open pushbutton that are connected to the input module. The input section and the CPU are connected by a double-sided arrow. The output load devices to the right of the output module has 4 loads; a solenoid valve, a motor, an LED light, and a horn. They are connected in parallel to the output module. The output module and the CPU are connected by a double-sided arrow. The programmer is connected to the CPU by a double-sided arrow. The power supply module is connected to the CPU by a down arrow.

Return to Figure 1-6

OceanofPDF.com

Figure 1-7 Extended Description (Chapter 1)

Return to Figure 1-7

At left, a PLC schematic diagram no separate, removable units. A horizontal row of squares at top is labeled input connections. Below, a horizontal rectangular box is labeled processor. A horizontal row of squares at the bottom is labeled output connections. At the top row of boxes, the first 3 from the left are shaded, with a down arrow pointing to each. The arrows connect to Line L 1 which has a pressure switch, a temperature switch, and a pushbutton that are connected to the first 3 terminals of the input section through the common power bus. At the bottom row of boxes, the first 3 from the left are shaded. Line L 2 has a relay, solenoid valve, and a pilot lamp that are connected to the first 3 terminals of the output connection through the common return bus. At right is a PLC.

Return to Figure 1-7

OceanofPDF.com

Figure 1-8 Extended Description (Chapter 1)

Return to Figure 1-8

A schematic diagram of a modular controller consisting of 5 sections. From left the compartments are labeled power supply, processor module, input module, output module, and combination I O module. On the bottom, 2 dashed lines show how the module slides into the rack.

Return to Figure 1-8

OceanofPDF.com

Figure 1-11 Extended Description (Chapter 1)

Return to Figure 1-11

At left, the diagram of an input module with seventeen pins labeled 0 through 15, and two DC com switch. a 24 VDC field device power supply connected to the input module, labeled from the top; pushbuttons, limit switches, sensors, and manual pushbutton. The pushbutton is connected to IN 0. The limit switch is connected to IN 4. The sensor is connected to IN 8. The manual pushbutton is connected to IN 12. At right, the diagram of an input module with seventeen pins labeled 0 through 15, and a VAC and DC com switch. a 24 VDC field device power supply connected to the output module, labeled from the top; small motors, motor starters, solenoid valves, and indicator lights. The small motor is connected to pin VAC. The motor starter is connected to OUT 4. The solenoid valve is connected to OUT 5. The indicator light is connected to AC com.

Return to Figure 1-11

OceanofPDF.com

Figure 1-12 Extended Description (Chapter 1)

Return to Figure 1-12

A clear cylinder is filled about 1 eight with liquid flowing from a faucet at left. At right a vertical mixer with the motor labeled is inside the container. Outside the container, a labeled pressure sensor switch is connected to the left of the container, and a labeled temperature sensor switch is connected to the right. A labeled manual pushbutton station is at lower right.

Return to Figure 1-12

OceanofPDF.com

Figure 1-13 Extended Description (Chapter 1)

Return to Figure 1-13

A ladder diagram with a vertical line at left labeled L1 and a vertical line at right labeled L2. The voltage across L1 and L2 is labeled 120 VAC with a double-sided arrow. From the left, L1 is connected to a labeled pressure switch and temperature switch. After the temperature switch, a vertical down line connects to another line that goes left and connects to L1. A labeled manual pushbutton is connected along that horizontal line. Between the temperature switch and L2 is a labeled motor starter coil and a closed overload relay labeled O L.

Return to Figure 1-13

OceanofPDF.com

Figure 1-14 Extended Description (Chapter 1)

Return to Figure 1-14

A schematic diagram of an input module with 9 pins labeled 0 through 7, and a common switch. Line L1 has 3 switches connected to the input module, labeled from the top; pressure, temperature, and manual pushbutton. The pressure switch is connected to pin

1. The temperature switch is connected to pin
2. The manual pushbutton is connected to pin
3. Line N is connected to the common pin. The voltage across the terminal L1 and N is 120 VAC.

Return to Figure 1-14

OceanofPDF.com

Figure 1-15 Extended Description (Chapter 1)

Return to Figure 1-15

A schematic diagram of an output module with 9 pins labeled L1 and 0 through 7. The line L1 is connected to pin L1. From the right, line N is connected to a closed overload relay, a motor starter coil, then pin 1. The voltage across the terminal L1 and N is labeled 120 VAC.

Return to Figure 1-15

OceanofPDF.com

Figure 1-16 Extended Description (Chapter 1)

Return to Figure 1-16

At top, line L1 has the parallel combination of input devices; a pressure switch with address I/1 connected to input pin 1, a temperature switch with address I/2 connected to input pin 2, and a manual pushbutton with address I/3 connected to input pin 3. Line L2 has a series connection of output devices; a motor starter coil with address O/1 connected to output pin 1, and an overload relay. At bottom, the three-step flow diagram includes; monitor inputs which checks the program, execute program which executes control program, and change outputs which updates the outputs.

Return to Figure 1-16

OceanofPDF.com

Figure 1-17 Extended Description (Chapter 1)

Return to Figure 1-17

A fixed PLC controller schematic diagram where L1 has 3 parallel switches. From left to right they are labeled pressure, temp, and P B. They are connected to the controller input terminals I1, I2, and I3, respectively. A fuse is connected between L1 and the input devices. L1 passes through the fuse and L2 and connects to L1 in the input module. L2 passes through L1 and connects to input terminal L2. At the bottom, the output pins are labeled Q1, Q2, Q3, and Q4, where each has 2 pairs of terminals. A motor starter coil, labeled M, connects directly to L2 and the Q1 relay of the output pins. A feedback line passes between the first output pin and L1. A magnified view of the program is the center of the PLC controller.

Return to Figure 1-17

OceanofPDF.com

Figure 1-18 Extended Description (Chapter 1)

Return to Figure 1-18

A ladder diagram with a vertical line at left labeled L1 and a vertical line at right labeled L2. The voltage across L1 and L2 is labeled 120 VAC with a double-sided arrow. From the left, L1 is connected to a labeled pressure switch and temperature switch. Between the switches, a vertical down line connects to another line that goes left and connects to L1. A labeled manual pushbutton is connected along that horizontal line. Between the temperature switch and L2 is a labeled motor starter coil and a closed overload relay labeled O L.

Return to Figure 1-18

OceanofPDF.com

Figure 1-19 Extended Description (Chapter 1)

Return to Figure 1-19

Two vertical lines are connected by a horizontal line. Along the line, from left to right, switches are labeled pressure switch, I1, temperature switch, I2, and motor starter coil, O1. Below, from the left line, a second horizontal line goes right, then up to connect to the first line between the pressure and temperature switches. Along the second line is a switch labeled manual pushbutton, I3.

Return to Figure 1-19

OceanofPDF.com

Figure 1-20 Extended Description (Chapter 1)

Return to Figure 1-20

A ladder diagram with a vertical line at left labeled L1 and a vertical line at right labeled L2. The voltage across L1 and L2 is labeled 120 VAC with a double-sided arrow. From the left, L1 is connected to a labeled on/off switch, a pressure switch, a temperature switch, a motor starter coil, and a closed overload relay switch. Between the on/off and pressure switches, a vertical down line connects to another line that goes right and has a labeled manual pushbutton, then goes up and connects to L1 between the pressure and temperature switches. After the temperature switch is a labeled motor starter coil and a closed overload relay labeled O L.

Return to Figure 1-20

OceanofPDF.com

Figure 1-21 Extended Description (Chapter 1)

Return to Figure 1-21

A schematic diagram of an input module with 9 pins labeled 0 through 7, and common. Line L1 has 4 switches from the top labeled pressure switch, temperature switch, manual pushbutton, and an ON/OFF switch connected to the input module. The pressure switch is connected to pin 1, the temperature switch is connected to pin 2, the manual pushbutton is connected to pin 3, and the on/off switch is connected to pin 4.The line N is connected to the pin labeled common. The voltage across the terminal L1 and N is 120 VAC. The magnified view of the program to the right of the input module has, between lines L1 and L2 and from left to right, the labeled parts as follows: ON/OFF switch, pressure switch, temperature switch, motor starter coil, and manual pushbutton.

Return to Figure 1-21

OceanofPDF.com

Figure 1-23 Extended Description (Chapter 1)

Return to Figure 1-23

A schematic diagram with a single PLC in the middle top. On the left it is connected to 2 different PLCs and a single computer. On the right it is connected to 2 different PLCs.

Return to Figure 1-23

OceanofPDF.com

Figure 1-24 Extended Description (Chapter 1)

Return to Figure 1-24

A ladder diagram with a vertical line at left and at right. The voltage between them is labeled 120 VAC with a double-sided arrow. From the left vertical line, a horizontal line is connected to switches labeled S1, LS1, and light symbol L1. After switch S1, a vertical down line connects to another line that goes right and connects to a switch labeled LS2. An up line connects to the horizontal line after a switch labeled LS1, which is followed by a light symbol labeled L1.

Return to Figure 1-24

OceanofPDF.com

Figure 1-25 Extended Description (Chapter 1)

Return to Figure 1-25

The pushbutton P B 1 is connected in series with the three parallel connection of three switches S1, S2, and S3. The parallel connection of the switches is connected to the pressure switch P S 1 , the temperature switch T S 1, and the light L1 in series. The input and output devices are connected between two terminal lines. The voltage across the terminal lines is 120 V A C.

Return to Figure 1-25

OceanofPDF.com

Figure 2-2 Extended Description (Chapter 2)

Return to Figure 2-2

At left, a device labeled local input-output has 4 ports, from left to right, labeled power, processor, input, output, and a switch. A labeled stop-start button is connected to the input port, and a labeled motor is connected to the output port. At right, a device labeled remote input-output has 3 ports, from left to right, labeled power, input, output, and a switch. A labeled sensor is connected from the input port to a catch container below a hopper device. A labeled on-off control is connected from the output port to a labeled hopper. A line between the switches of the local and remote devices is labeled communication.

Return to Figure 2-2

OceanofPDF.com

Figure 2-3 Extended Description (Chapter 2)

Return to Figure 2-3

(a) The schematic diagram shows the processor memory. The processor memory is divided into two sections: data files and program files. The input pins I/0, I/1, and output pins O/0, O/3 are inside the processor. A pushbutton addressed as I: 1/ 0 is connected to the input pin 1 in the data files and limit switch addressed as I: 1/1 is connected to input pin 0 in the data files through an interface. Output pin 0 from the data file is connected to a output device addressed as O: 3/0 through an interface. The inputs from the pin 1 and 0 are connected to the pins addressed as I: 1/0 and I: 1/1 in the program files section. The output from the motor starter coil addressed as O: 3/0 in the program file section is connected to the output pin in the data file section.

(b) The schematic diagram shows the word-level addressing. It has four sections labeled as processor, 1, 2, and 3 as analog module. The magnified view of the analog module has input output pins 0 and 1. The address format is labeled type, slot, word and bit. The input address is labeled 1, 2, 0, Not used and the output address is labeled O, 2, 0, Not used. The magnified view of the analog module represents a thermocouple analog input is connected to the terminals IN 0 plus and IN 0 negative using a twisted-pair cable. The input address is labeled I:1:2:0. The valve analog output is connected across the terminals OUT 0 plus and OUT 0 negative. The output address is labeled O0:2.0.

Return to Figure 2-3

OceanofPDF.com

Figure 2-4 Extended Description (Chapter 2)

Return to Figure 2-4

The symbolic representation includes, from top to bottom, start labeled description assigned to alias tag, I PBO labeled alias tag pointing to base address, Local:6:1:Data.0 labeled base address, and symbol with left and right brackets labeled input instruction.

Return to Figure 2-4

OceanofPDF.com

Figure 2-5 Extended Description (Chapter 2)

Return to Figure 2-5

Top: A schematic of RSLogix 5000 program. The mid portion is labeled program consisting of 3 connection lines joint in parallel. The line at the top consists of Stop_PB <Local:1:I.Data.0>, Start_PB <Local:1:I.Data.1>, and Motor <Local:2:O.Data.0> connected in series. A motor <Local:2:0.Data.0> is connected in parallel with Start_PB <Local:1:I.Data.1>. The connection line in the middle consists of Motor <Local:2:O.Data.0> and Green_PL <Local:2:O.Data.1> connected in series. The connection line at the bottom consists of Motor <Local:2:O.Data.0> and Red_PL <Local:2:O.Data.2> connected in series.

On the left is the input section consisting of a read and a green L E D light. The right terminal of the red and green L E Ds is connected to Stop_PB and Start_PB, respectively. The left terminals of each are connected in parallel to a line labeled L1.

On the right is the output, where one terminal of each, motor, Green_PL, and Red_PL are connected at 3 points of a line labeled L2.

Bottom: The middle portion labeled local rack consists of 2 slots labeled Slot 1 and Slot 2. Slot 1 has 120 volt A C input and 5 signal points, 0 in red, 1, 2, 3, and N. Slot 2 has 120 volt A C output and 5 signal points, 0, 1, 2, 3, and L1.

On the left are 2 tags, Stop_PB <Local:1:I.Data.0> and Start_PB <Local:1:I.Data.1> with their one ends connected to 0 and 1 signal points of slot 1, respectively, and their other ends connected to point L1 via 2 switches. Signal point N is connected to a point N O. The voltage across points L1 and N O is 120 volt A C.

On the right are 3 tags, Motor <Local:2O.Data.0>, Green_PL <Local:2O.Data.1>, and Red_PL <Local:2O.Data.2> with their one ends connected to 0, 1, and 2 signal points of slot 2, respectively, and their other ends connected to a point O N in parallel. A point L1 is connected to L1 signal point of slot 2. The voltage across O N and L1 IS 120 volt A C.

Return to Figure 2-5

OceanofPDF.com

Figure 2-7 Extended Description (Chapter 2)

Return to Figure 2-7

A module in the center has an up arrow above it labeled discrete outputs pointing to the following labeled devices: indicator lights, signaling column, relays, and motor starter. Below the module are the following labeled devices: pushbuttons, selector switch, limit switch, and proximity switches. An up arrow from the devices to the module is labeled discrete inputs.

Return to Figure 2-7

OceanofPDF.com

Figure 2-9 Extended Description (Chapter 2)

Return to Figure 2-9

A block diagram with the left section defined by a bracket labeled power. It is coupled by a box labeled opto-electrical isolation to the section at right defined by a bracket labeled logic. At left, L1 is wired to a pushbutton with 120 VAC voltage. L1 and L2, labeled input signal, enter a box labeled bridge rectifier, which is connected to a box labeled Zener diode level detection. Two lines leave that box and enter a box labeled opto-electrical isolation. Two lines leave that box and enter a box labeled logic circuits. An LED light is connected across the top of the logic circuits and is labeled input status indicator. Two lines leave the logic circuits box with right-facing arrows labeled to processor (low VDC).

Return to Figure 2-9

OceanofPDF.com

Figure 2-10 Extended Description (Chapter 2)

Return to Figure 2-10

L1 is wired to a fuse and a pushbutton in series. L1 and L2 are connected to the pins of the input module terminal strip. The input noise filter consists of a capacitor, C connected between the resistors R1 and R2 in series. The noise filter is connected across the two nodes of the bridge rectifier. The output nodes of the bridge rectifier is connected to a threshold detector. The threshold detector consists of a resistor R3 and a Zener diode, Zd. The threshold detector is connected to the optical isolator. The optical isolator consists of a phototransistor. The optical isolator is connected to the digital logic circuit. An LED is connected across the digital logic circuit. The LED act as the input status indicator.

Return to Figure 2-10

OceanofPDF.com

Figure 2-11 Extended Description (Chapter 2)

Return to Figure 2-11

A block diagram with the left section defined by a bracket labeled logic. It is coupled by a dashed-line box labeled opto-electrical isolation to the section at right defined by a bracket labeled power. Two right-facing arrows labeled digital signal from processor enter a box labeled logic circuits. An LED light is connected across the top of the logic circuit and is labeled output status indicator. The power section has a box labeled electronic switch module. L1 connected to a load, and L2 emerges out of the electronic switch. The voltage across lines L1 and L2 is 120 VAC.

Return to Figure 2-11

OceanofPDF.com

Figure 2-12 Extended Description (Chapter 2)

Return to Figure 2-12

An L E D is connected to the digital logic circuit. The L E D acts as an output status indicator. It is coupled to the optical isolator which contains a diode and a phototransistor. The optical isolator is connected to a Triac switch in series with a fuse, which is connected to L1 through output module terminal strip. The other end of the triac switch is connected to a load wired with L2 through the output module terminal strip.

Return to Figure 2-12

OceanofPDF.com

Figure 2-13 Extended Description (Chapter 2)

Return to Figure 2-13

Lines L1, L2, and L3 are connected to the terminals T1, T2, and T3 of a three-phase motor using a three-phase magnetic motor starter. The three-phase magnetic motor starter consists of open contact and an overload relay. L1 and L2 are connected to the interface terminals L1 and L2. In-between L1 and L2, a series connection of a control relay, motor starter coil, and overload relay is connected in a way such that one end of the control relay is connected to L1 and one end of a closed overload relay is connected to L2. An interposing relay coil is connected to line L2 and the other end is connected to a pin in the interface terminal.

Return to Figure 2-13

OceanofPDF.com

Figure 2-14 Extended Description (Chapter 2)

Return to Figure 2-14

Relay output circuit: A coil is connected to the A C or D C input with a load in series.

Triac output circuit: A triac is connected to the A C input with a load in series.

Transistor output circuit: A p n p transistor is connected to the D C input with a load in series.

Return to Figure 2-14

OceanofPDF.com

Figure 2-15 Extended Description (Chapter 2)

Return to Figure 2-15

At left, a schematic diagram of a labeled sinking input module. It has 9 terminals, labeled from the top 0 through 7 and a negative terminal labeled common. The positive terminal of the power supply is connected to 1 end of the sourcing sensor. The other end of the sourcing sensor is connected to pin 1. The negative terminal of the power supply is connected to the negative pin. A left-facing arrow at the bottom, labeled current, points away from the common label, showing the current flows inside the power supply terminal from the module. At right, a schematic diagram of a labeled sourcing input module. It has 9 terminals, labeled from the top 0 through 7 and a positive terminal labeled common. The negative terminal of the power supply is connected to 1 end of the sinking sensor. The other end of the sinking sensor is connected to pin 1. The positive terminal of the power supply is connected to the positive pin in the module. A right-facing arrow at the bottom, labeled current, points toward the common label, showing the current flows out of the power supply terminal into the module.

Return to Figure 2-15

OceanofPDF.com

Figure 2-16 Extended Description (Chapter 2)

Return to Figure 2-16

At left, a schematic diagram of a labeled sourcing output module. It has 9 terminals labeled from the top 0 through 7, and a positive terminal labeled common. The negative terminal of the power supply is connected to 1 end of the sinking field device. The other end of the sinking field device is connected to pin 1. The positive terminal of the power supply is connected to the positive pin in the module. A left-facing arrow at the bottom, labeled current, points to the common label, showing the current flows out of the power supply into the module. At right, a schematic diagram of a labeled sinking output module. It has 9 terminals labeled from the top 0 through 7 and a negative terminal labeled common. The positive terminal of the power supply is connected to 1 end of the sourcing field device. The other end of the sourcing field device is connected to pin 1. The negative terminal of the power supply is connected to the negative pin in the module. A right-facing arrow at the bottom, labeled current, points away from the common label, showing the current flows inside the power supply from the module.

Return to Figure 2-16

OceanofPDF.com

Figure 2-17 Extended Description (Chapter 2)

Return to Figure 2-17

A line with an arrow, labeled analog inputs, has 3 analog devices at left going into a module. From the top the devices are labeled proximity sensor, potentiometer, and temperature sensor. A line with an arrow, labeled analog outputs, has 3 analog outputs leaving the module at right. From the top the devices are labeled control valve, meter, and stepper motor.

Return to Figure 2-17

OceanofPDF.com

Figure 2-18 Extended Description (Chapter 2)

Return to Figure 2-18

From left, the level transmitter from a fluid tank is connected to a PLC module by a right arrow. The PLC module has 3 boxes labeled from left, analog input module, processor, and analog output module, each connecting to the next with a right arrow. A right arrow leaving the PLC connects to a meter labeled level indicator.

Return to Figure 2-18

OceanofPDF.com

Figure 2-19 Extended Description (Chapter 2)

Return to Figure 2-19

At left, micrologix 4-channel analog thermocouple input module. At right, a schematic diagram of a thermocouple input module with 10 terminals: CJC plus, CJC negative, IN 2 plus, IN 2 negative, IN 3 plus, IN 3 negative, IN 0 plus, IN 0 negative, IN 1 plus, and IN 1 plus. The labeled ungrounded thermocouple is connected across IN 2 plus and IN 2 negative. The labeled grounded thermocouple is connected across IN 0 plus and IN 0 negative. The cold junction compensating (CJC) thermistors CJC plus and CJC negative are connected to each other.

Return to Figure 2-19

OceanofPDF.com

Figure 2-20 Extended Description (Chapter 2)

Return to Figure 2-20

At left, a schematic diagram labeled sensor supplied power has the positive and negative terminals of the power supply module connected to a sensor module. The positive and negative terminals of the sensor are connected to the shielded twisted pair cable. One of the twisted pair cables connected to the negative terminal is grounded. The analog data range is given as 4 through 20 milliampere. The twisted-pair cable is connected across a resistor through an interface. At right, a schematic diagram labeled module supplied power has the positive and negative terminals of the sensor connected to the shielded twisted pair cable. One of the twisted pair cables connected to the positive terminal is grounded. The analog data range is given as 4 through 20 milliampere. One of the twisted pair cables is connected to the positive terminal of the power supply. The other cable is connected to 1 end of a resistor. The other end of the resistor is connected to the negative terminal of the power supply.

Return to Figure 2-20

OceanofPDF.com

Figure 2-21 Extended Description (Chapter 2)

Return to Figure 2-21

A schematic diagram of a transmitter unit, with a sensor transducer tip, connected to an analog input module at right. The analog input module has 8 terminals labeled from the top: 1, 1 C, 2, 2 C, 3, 3 C, 4, 4 C. The labeled sensor transducer is enlarged to show the detail of the tip of the transmitter unit. The positive terminal of the transmitter unit is connected to terminal 2 in the analog input module and the negative terminal is connected to terminal 2 C of the analog input module. The range of voltage that can be applied across the terminals of the transmitter unit is 0 to 10 VDC. A graph where the x-axis represents time and the y-axis represents voltage has a curve starting near the origin and gradually increasing over time.

Return to Figure 2-21

OceanofPDF.com

Figure 2-22 Extended Description (Chapter 2)

Return to Figure 2-22

Three block diagrams, each with a power supply, unique transmitter, and input module. From the top, they are labeled 2-, 3-, & 4-wire transmitter. Two-wire transmitter: The positive terminal of the power supply is connected to the positive terminal of the transmitter. The negative terminal of the transmitter is connected to the IN plus pin of the input module. The negative terminal of the power supply and IN negative pin of the input module are grounded together. Analog common (ANL COM) points are electrically connected inside the module but not to the ground. Three-wire transmitter: There are 3 terminals in the transmitter supply signal module. The positive terminal of the power supply is connected to the positive terminal of the transmitter. The negative terminal of the transmitter is connected to the IN plus pin of the input module. The negative terminal of the power supply, the ground terminal, and IN negative pin of the input module are grounded together. Analog common (ANL COM) points are electrically connected inside the module but not to the ground. Four-wire transmitter: There are 4 terminals in the transmitter supply signal module. The positive terminal of the power supply is connected to the positive terminal of the transmitter. The second positive terminal of the transmitter is connected to the IN plus pin of the input module. The negative terminal of the power supply, the 2 negative terminals of the transmitter, and IN negative pin of the input module are grounded together. Analog common (ANL COM) points are electrically connected inside the module but not to the ground.

Return to Figure 2-22

OceanofPDF.com

Figure 2-23 Extended Description (Chapter 2)

Return to Figure 2-23

The positive terminal of the power supply is connected to the positive terminals of 3 two-wire transistor and one four-wire transistor. The negative terminals of the first three two-wire transistor are connected to the IN 0 plus, IN 1 plus, and IN 2 plus of the input module. The positive terminal of the four-wire transmitter is connected to the IN 3 plus pin. The negative terminal of the power supply, ground pin of the four-wire transistor, and IN 3 negative are grounded together. The pins in the input module which are coupled together are IN 0 negative and ALL COM, IN 1 negative and ALL COM, IN 2 negative and ALL COM, and IN 3 negative and ALL COM.

Return to Figure 2-23

OceanofPDF.com

Figure 2-24 Extended Description (Chapter 2)

Return to Figure 2-24

The analog output from P L C output module is connected to a valve attached in the container and the level sensor in the container sends the analog input signal to the input module of the P L C.

Return to Figure 2-24

OceanofPDF.com

Figure 2-25 Extended Description (Chapter 2)

Return to Figure 2-25

Top: A thermometer with degrees centigrade (engineering units) ranges negative 100 through 200 in increments of 100 and a probe are dipped into a liquid-filled container. The probe is connected to a transducer module. The positive and negative terminals of the transducer are connected to the input module at the top right. A small box over the input box is labeled analog module. The information stored in binary units in the input module ranges from negative 32,768 through 32,767. The plus minus DC voltage from the transducer ranges from negative 10 through 10.

Bottom: The input voltage and the binary units are jointly labeled on the horizontal axis. The extreme left and extreme right sides of the horizontal axis have the following labels:

Extreme left:

Input, negative 10 volts,

Minimum, negative 32,768

Extreme right:

Input, plus 10

Maximum, plus 32,767

The vertical axis passes through the midpoint of the horizontal axis and labels temperatures ranging from negative 100 through plus 200 in increments of 100. A sloped graph line from the bottom left to the top right passes through quadrants 3, 2, and 1.

Return to Figure 2-25

OceanofPDF.com

Figure 2-27 Extended Description (Chapter 2)

Return to Figure 2-27

The schematic diagram of a rotary encoder with rotating shaft in counterclockwise direction. The output is a signal count pulses of 1, 0, 1, 0, 1, 0. 1.

Return to Figure 2-27

OceanofPDF.com

Figure 2-28 Extended Description (Chapter 2)

Return to Figure 2-28

At left, the PLC servo module is a rectangular module containing the four sets of pins on the right side labeled outputs and inputs. At middle, the servo motor drive is connected to the output and input to the servo module. At right, the servo motor is connected to the input of the servo drive and the output is connected to the encoder attached to the servo motor.

Return to Figure 2-28

OceanofPDF.com

Figure 2-29 Extended Description (Chapter 2)

Return to Figure 2-29

The input-output module has a discrete input-output module with a switch and a valve, and an analog input-out devices with a rheostat and a motor. The input-output modules are connected to the C P U of the processor module. The C P U is connected to the memory. The power supply is attached to the C P U and the communication interface is attached to the C P U.

Return to Figure 2-29

OceanofPDF.com

Figure 2-32 Extended Description (Chapter 2)

Return to Figure 2-32

Four examples of 32 bit data types, each with 2 rows of 32 blocks. Example 1, the top row is numbered from left to right 31 through 0. It is titled DINT (double integer) or double word (32 bits), number range from negative 2,147,483,648 to 2,147,483,647. The first row of blocks has the following labeled: 31, 24, 23, 16, 15, 8, 7, 0. The second row has all the blocks highlighted. Example 2, the top row is numbered from left to right 31 through 0. It is titled INT (integer) or word (16 bits), number range from negative 32,768 to 32,767. The first row of blocks has the following labeled: 31, 15, 8, 7, 0. The second row has sixteen highlighted blocks and the remaining blocks are not highlighted and labeled not used. Example 3, the top row is numbered from left to right 31 through 0. It is titled SINT (small integer) or byte (8 bits), number range from negative 128 to 127. The first row of blocks has the following labeled: 31, 7, and 0. The second row has seven highlighted blocks and the remaining blocks are not highlighted and labeled not used. Example 4, the top row is numbered from left to right 31 through 0. It is titled BOOL (Boolean) or 1 bit, number range from 0 to 1. The first row of blocks has the following labeled: 31 and 0. The second row has thirty highlighted blocks and one block is not highlighted and labeled not used.

Return to Figure 2-32

OceanofPDF.com

Figure 2-34 Extended Description (Chapter 2)

Return to Figure 2-34

At left, a memory module is labeled RAM (volatile), and from the top includes executed program, current data, memory bits, timers, counters. A double-headed arrow points to a memory module at right labeled EEPROM (nonvolatile), and from the top includes program backup, parameters

Return to Figure 2-34

OceanofPDF.com

Figure 2-39 Extended Description (Chapter 2)

Return to Figure 2-39

At left, a dashed-line box labeled HMI package contains a grid at top left labeled tag database. It is connected with a double-headed arrow to a labeled graphic screen below with labeled start and stop buttons. The tag database is connected with a double-headed arrow to a labeled input-output server at top right. The input-output server is connected to the input port of a labeled PLC module outside the box at top right.

Return to Figure 2-39

OceanofPDF.com

Figure 2-41 Extended Description (Chapter 2)

Return to Figure 2-41

A graph with the horizontal axis labeled 2 hours ranging from 0 to 8 in increments of 2. The vertical axis is labeled PSIG ranging from 0 to 300 in increments of 50. A horizontal line labeled shutdown is drawn parallel to the x-axis at point 300. Another horizontal line labeled alarm is drawn parallel to the x-axis at point 250. A third horizontal dashed line is drawn parallel to the x-axis at point 200. A curve begins at point 1, 200, and ends at point 8, 232, with a waveform trend that does not cross the alarm line. The line passes through the points (1, 200), (3, 240), and (8, 232) The values on the top are as follows: current pressure, 235.2 PSIG; alarm level, 250 PSIG; shutdown level, 300 PSIG.

Return to Figure 2-41

OceanofPDF.com

Figure 2-42 Extended Description (Chapter 2)

Return to Figure 2-42

Top left: The section labeled select mode has a button labeled automatic and manual on the top left, a button with a bent left arrow marked on it is at the top right, and two buttons are at the bottom with upward and downward arrowheads.

Top right: The section labeled motor status has two components: mode on the left and status on the right, each consisting of a button marked auto and running on them, respectively.

Bottom: In a row, there are four buttons labeled from left to right: start motor, stop motor, config screen, and motor speed.

Return to Figure 2-42

OceanofPDF.com

A Block Diagram Graphic Extended Description (Chapter 3)

Return to A Block Diagram Graphic

A block diagram with the first box at left containing 348 with base 10, labeled decimal. A right arrow connects that to a box containing 101011100 with base 2, labeled binary. A right arrow connects that to a box containing 534 with base 8, labeled octal. A right arrow connects that to a box containing 15C with base 16, labeled hexadecimal. A down arrow from the box labeled binary connects to a box containing 0011 0100 1000, labeled binary coded decimal.

Return to A Block Diagram Graphic

OceanofPDF.com

Figure 3-1 Extended Description (Chapter 3)

Return to Figure 3-1

From right to left, there is 0 above 2, 1 above 6, 2 above 9, and 3 above 1. A line from the number 2 connects to the weighted value of 2 written as 2 times 10 to the power of 0 equals 2 times 1 equals 2. A line from the number 6 connects to the weighted value of 6 written as 6 times 10 to the power of 1 equals 6 times 10 equals 60. A line from the number 9 connects to the weighted value of 9 written as 9 times 10 to the power of 2 equals 9 times 100 equals 900. A line from the number 1 connects to the weight value of 1 written as 1 times 10 to the power of 3 equals 1 times 1000 equals 1000. The sum of products 2, 60, 900, and 1000 is 1962 with base 10.

Return to Figure 3-1

OceanofPDF.com

Figure 3-2 Extended Description (Chapter 3)

Return to Figure 3-2

A digital sign waveform is plotted on a graph with the horizontal axis labeled time, and the vertical axis labeled volts, marked 0 and plus 5 in increments of 1. The plot has 2 steps with a peak value plus 5 volts, and 2 lows with a value 0. The peak value has a left-facing arrows level with the peak, and is labeled high (H) 1. The low value has a left-facing arrow level with the low, and is labeled low (L) 0.

Return to Figure 3-2

OceanofPDF.com

Figure 3-3 Extended Description (Chapter 3)

Return to Figure 3-3

The numbers 1 0 1 0 1 1 0 1 with base 2 are in a set of 8 blocks, labeled binary number. From right to left, there is 0 above 1, 1 above 0, 2 above 1, 3 above 1, 4 above 0, 5 above 1, 6 above 0, and 7 above 1. A line from the number 1 connects to the weighted value of 1 written as 1 times 2 to the power of 0 equals 1 times 1 equals 1. A line from the number 0 connects to the weighted value of 0 written as 0 is 0 times 2 to the power of 1 equals 0 times 2 equals 0. A line from the number 1 connects to the weighted value of 1 written as 1 times 2 to the power of 2 equals 1 times 4 equals 4. A line from the number 1 connects to the weighted value of 1 written as 1 times 2 to the power of 3 equals 1 times 8 equals 8. A line from the number 0 connects to the weighted value of 0 written as 0 times 2 to the power of 4 equals 0 times 16 equals 0. A line from the number 1 connects to the weighted value of 1 written as 1 times 2 to the power of 5 equals 1 times 32 equals 32. A line from the number 0 connects to the weighted value of 0 written as 0 times 2 to the power of 6 equals 0 times 64 equals 0. A line from the number 1 connects to the weighted value of 1 written as 1 times 2 to the power of 7 equals 1 times 128 equals 128. The sum of products 1, 0, 4, 8, 0, 32, 0, and 128 is the decimal number 173 with base 10.

Return to Figure 3-3

OceanofPDF.com

Figure 3-4 Extended Description (Chapter 3)

Return to Figure 3-4

A model with 16 blocks connected in a row. Right to left, the blocks are numbered 1, 1, 0, 1, 1, 1, 0, 0, 0, 1, 1, 0, 0, 1, 1, 0. The first block at left is labeled M S B. The 11th block from the left, containing the number 1, is labeled BIT. The last block is labeled L S B. Below the row is a double-sided arrow spanning the first 8 blocks labeled upper byte. A double-sided arrow spanning the next 8 blocks is labeled lower byte. At the bottom is double-sided arrow spanning all the blocks labeled 16-bit word.

Return to Figure 3-4

OceanofPDF.com

Figure 3-5 Extended Description (Chapter 3)

Return to Figure 3-5

A grid modeling the memory size of a 1-K word. Across the top of the grid, from left to right, the columns are labeled as 15, 14, 13, 12, 11, 10, 9, 8, 7, 6, 5, 4, 3, 2, 1, 0, and labeled bits. The vertical axis has 6 rows labeled from the top 0000 through 0005. There is a gap, then the label continues from 1018 through 1023 for another 6 rows. The axis is labeled word addresses. The row labeled 0004 contains the numbers 0 1 1 0 0 1 1 0 0 0 1 1 1 0 1 1.

Return to Figure 3-5

OceanofPDF.com

Figure 3-6 Extended Description (Chapter 3)

Return to Figure 3-6

A 6-step process for the conversion of the decimal number 47 to a binary number. In the first step, the decimal number 47 is divided by 2 equals 23 with a remainder of 1. In the second step, the quotient 23 from step 1 has a downward left arrow showing its position at the beginning of the next equation, 23 divided by 2 equals 11 with a remainder of 1. Each step follows the same pattern with the down arrow. In the third step, the quotient 11 from the previous step is divided by 2 equals 5 with a remainder of 1. In the fourth step, the quotient 5 from the previous step is divided by 2 equals 2 with a remainder of 1. In the fifth step, the quotient 2 from the previous step is divided by 2 equals 1 with a remainder of 0. In the sixth step, the quotient 1 from the previous step is divided by 2 equals blank with a remainder of 1. The remainders 1, 1, 1, 1, 0, and 1 from each step is arranged in a column of 6 rows. The binary number in row 1 is labeled as L S B and in row 6 is labeled as M S B. The binary number is represented as 101111.

Return to Figure 3-6

OceanofPDF.com

Table 3-2 Extended Description (Chapter 3)

Return to Table 3-2

The first column lists 15 four-digit numbers, where the first three digits combined labeled magnitude, and the fourth digit is labeled sign. Starting from the top, the first 8 values are combined labeled as same as binary numbers. The second column is labeled decimal value and lists 16 numbers from positive 7 through negative 7 in decrement of 1.

Return to Table 3-2

OceanofPDF.com

Table 3-3 Extended Description (Chapter 3)

Return to Table 3-3

The table compares signed decimal numbers with their 1's complement, 2's complement, and 16-bit 2's complement binary representations. Positive numbers (positive 1 through positive 7) have the same binary and complement values, while negative numbers (negative 1 through negative 8) show the transformations from 1's to 2's complement forms. Each column shows the respective binary values for different signed decimal numbers. The 16-bit column extends the 2's complement representation to 16 bits.

Return to Table 3-3

OceanofPDF.com

Figure 3-7 Extended Description (Chapter 3)

Return to Figure 3-7

The numbers 4 6 2 with base 8 are in a set of 3 blocks, labeled octal number. A line from the number 2 connects to the weighted value of 2 written as 2 times 8 to the power of 0 equals 2 times 1 equals 2. A line from the number 6 connects to the weighted value of 6 written as 6 times 8 to the power of 1 equals 6 times 8 equals 48. A line from the number 4 connects to the weighted value of 4 written as 4 times 8 to the power of 2 equals 4 times 64 equals 256. The sum of the products 2, 48, and 256 is the decimal number 306 with base 10.

Return to Figure 3-7

OceanofPDF.com

Figure 3-8 Extended Description (Chapter 3)

Return to Figure 3-8

The color-coded octal number 462 is shown in a set of 3 blocks. A color-coded down arrow from each octal number points to the color-coded binary number for each represented below in a set of 3 blocks. The green 2 is equivalent to 0 1 0. The blue 6 is equivalent to 1 1 0. The red 4 is equivalent to 1 0 0.

Return to Figure 3-8

OceanofPDF.com

Figure 3-9 Extended Description (Chapter 3)

Return to Figure 3-9

The numbers 1 B 7 are in a set of 3 blocks, labeled hex number. A line from the number 7 connects to the weighted value of 7 written as 7 times 16 to the power of 0 equals 7 times 1 equals 7. A line from B connects to the weighted value of B written as 11 times 16 to the power of 1 equals 11 times 16 equals 176. A line from the number 1 connects to the weighted value of 1 written as 1 times 16 to the power of 2 equals 1 times 256 equals 256. The sum of products 7, 176, and 256 is the decimal number 439 with base 10.

Return to Figure 3-9

OceanofPDF.com

Figure 3-10 Extended Description (Chapter 3)

Return to Figure 3-10

The color-coded hex number 1B7 is shown in a set of 3 blocks. A color-coded down arrow from each hex number points to the binary number for each represented below in a set of 12 corresponding color-coded numbers in blocks. The green 7 is equivalent to 0 1 1 1. The blue B is equivalent to 1 0 1 1. The red 1 is equivalent to 0 0 0 1.

Return to Figure 3-10

OceanofPDF.com

Figure 3-11 Extended Description (Chapter 3)

Return to Figure 3-11

The color-coded decimal number 7863 is shown in a set of 4 blocks. A color-coded down arrow from each decimal number points to the BCD number for each represented below in 4 sets of 4 blocks. The red 3 is equivalent to 0 1 1 1. The green 6 is equivalent to 0 1 1 0. The blue 8 is equivalent to 1 0 0 0. The red 7 is equivalent to 0 1 1 1. For the number 0 1 1 1, the first number 1 has an up arrow below it labeled 4 bits used for each decimal digit.

Return to Figure 3-11

OceanofPDF.com

Figure 3-12 Extended Description (Chapter 3)

Return to Figure 3-12

A thumbwheel switch with the number 8 displayed has 5 pins labeled 1, 2, 4, 8, and C connected to an input module. Each pin has a line labeled as follows: 1s input equals 0, 2s input equals 0, 4s input equals 0, and 8s input equals 1, which are connected to the first 4 terminals of the input module that has 18 pins. The C pin from the thumbwheel is connected to the positive terminal of a battery and the negative terminal of the battery is connected to the bottom left terminal of the input module.

Return to Figure 3-12

OceanofPDF.com

Figure 3-13 Extended Description (Chapter 3)

Return to Figure 3-13

The optical encoder disk uses a 4-bit Gray code to detect changes in angular position. The encoder disk is attached to a rotating shaft and outputs a digital Gray code signal that is used to determine the position of the shaft. The sensor senses the light rays from the light source. A Four-bit word is generated for each row of the disk. The disk has 4 rows and 16 columns. The four-bit words pointing each column in the disk are 0000, 0001, 0011, 0010, 0110, 0111, 0101, 0100, 1100, 1101, 1111, 1110, 1010, 1011, 1001, and 1000. Each bit of the four-bit word is represented in each row of the column. For binary number 0 is represented by the shaded portion and 1 is represented by the unshaded portion. The four-bit word 1011 is shown on the disk.

Return to Figure 3-13

OceanofPDF.com

Three Steps Graphic Extended Description (Chapter 3)

Return to Three Steps Graphic

Three steps of an equivalent binary addition problem adding 101 plus 10. In the first step, the addend in the first line is 101 and the addend in the second line is 10. The numbers in the 1s place column are 1 and 0, summed as 1; the numbers in the 2s place column are 0 and 1, summed as 1; and the number in the 4s place column is 1. The sum of the addends shown in the third line is 111. In the second step, the addend in the first line is 1010 and the addend in the second line is 11. The numbers in the 1s place column are 0 and 1, summed as 1; the numbers in the 2s place column are 1 and 1, summed as 0 and 1 is carried over to 4s place; the number in 4s place column is carried 1 and 0, summed as 1; and the number in the 8s place column is 1. The sum of the addend in the third line is 1101. In the third step; the addend in the first line is 11010 and in the second line is 1100. The numbers in the 1s place column are 0 and 0, summed as 0; the numbers in the 2s place column are 1 and 0, summed as 1; the number in the 4s place column are 0 and 1, summed as 1; the numbers in the 8s place column are 1 and 1, summed as 0 and 1 is carried over to 16s place; and the numbers in the 16s place column are carry 1 and 1, summed as 0 and 1 is carried over to 32s place. The sum of the addends in the third line is 100110.

Return to Three Steps Graphic

OceanofPDF.com

Calculation for Subtracting Numbers Graphic Extended Description (Chapter 3)

Return to Calculation for Subtracting Numbers Graphic

The subtraction of decimal numbers with minuend 10, the subtrahend 6, and the difference is 4. The equivalent binary representation is minuend in the first line 1010, the subtrahend in the second line 0110, and the difference is 100. To perform subtraction using 1s complement, first the subtrahend 0110 is changed to its 1s complement as 1001. Then the minuend in the first line is 1010, and the subtrahend in the second line 1001 are added together. The sum is 10011 in the third line. The bit in the 16s place is labeled end-around carry which is moved to the fourth line in the 1s place. The binary sum and the end-around carry are added and the resultant is 100.

Return to Calculation for Subtracting Numbers Graphic

OceanofPDF.com

Figure 3-14 Extended Description (Chapter 3)

Return to Figure 3-14

A floating point number system with 7 steps for the floating point representation of 4,234. From top to bottom, the numbers are as follows: 423,400.0 times 10 to the power of negative 2; 42,340.0 times 10 to the power of negative 1; 4,234.0 times 10; 423.4 times 10 to the power of 1; 42.34 times 10 to the power of 2; 4.234 times 10 to the power of 3; 0.4234 times 10 to the power of 4. The floating point representation of 4,234 is 4234.0 times 10 and is highlighted.

Return to Figure 3-14

OceanofPDF.com

Figure 3-15 Extended Description (Chapter 3)

Return to Figure 3-15

A floating point number model with a bar labeled 32 bits divided into three color-coded parts; small, medium, and large. The large part has an arrow to a color-coded bar labeled mantissa (23 bits). The medium part has an arrow to a color-coded bar labeled exponent (8 bits). The small part has an arrow to a color-coded bar labeled mantissa sign (1 bit).

Return to Figure 3-15

OceanofPDF.com

Figure 3-16 Extended Description (Chapter 3)

Return to Figure 3-16

From right to left, 32 blocks connected in a row are numbered 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 0, 1, 1, 0, 0, 0, 0, 0, 0, 1, 0. Below the first 23 blocks from right to left a double-sided arrow is labeled mantissa (23 bits). Below the next 8 blocks a double-sided arrow is labeled exponent (8 bits). The last block at left is labeled sign bit.

Return to Figure 3-16

OceanofPDF.com

Logic for Boolean Expression Graphic Extended Description (Chapter 4)

Return to Logic for Boolean Expression Graphic

A relay schematic is shown with two limit switches L S 1, L S 2 and solenoid switch S Q L connected in series. A ladder logic program is shown with ladder limit switches L S 1, L S 2 as A and B and solenoid switch S Q L as Y. The logic gate has inputs A and B entering an AND gate and leaving the gate as output Y.

Return to Logic for Boolean Expression Graphic

OceanofPDF.com

Figure 4-1 Extended Description (Chapter 4)

Return to Figure 4-1

An AND gate integrated circuit is shown with eight pins that is 4 pins on both right and left sides. The left side is inputs and the right is output. The first pin on the left side is light switch and the fourth pin is high beam switch. The second pin on the right side is high beam light.

Return to Figure 4-1

OceanofPDF.com

Figure 4-2 Extended Description (Chapter 4)

Return to Figure 4-2

An OR gate integrated circuit is shown with eight pins that is 4 pins on both right and left sides. The left side is inputs and the right is output. The first pin on the left side is passenger door switch and the fourth pin is driver door switch. The second pin on the right side is dome light.

Return to Figure 4-2

OceanofPDF.com

Figure 4-3 Extended Description (Chapter 4)

Return to Figure 4-3

A logic symbol for two input AND gate is shown. The two inputs are A and B with Y as the output. The truth table for AND gate is shown. The input logics for A and B are as (0 0), (0 1), (1 0) and (1 1). The output logics for Y against the inputs are as 0, 0, 0 and 1.

Return to Figure 4-3

OceanofPDF.com

Figure 4-4 Extended Description (Chapter 4)

Return to Figure 4-4

Four logic symbols for two input AND gate are shown. The two inputs are A and B with Y as the output. When A equals 0 and B equals 0, output Y equals 0. When A equals 0 and B equals 1, output Y equals 0. When A equals 1 and B equals 0, output Y equals 0. When A equals 1 and B equals 1, output Y equals 1.

The truth table for AND gate is shown. The input logics for A and B are as (0 0), (0 1), (1 0) and (1 1). The output logics for Y against the inputs are as 0, 0, 0 and 1.

Return to Figure 4-4

OceanofPDF.com

Figure 4-5 Extended Description (Chapter 4)

Return to Figure 4-5

The hardwired circuit consists of a battery, 2 switches, and a light bulb connected in series. The positive terminal of the battery is connected to one end of switch SW-A. The negative terminal of the battery is connected to one end of the bulb. The other end of the bulb is connected to one end of switch SW-B. Both switches are in the open position.

The truth table of AND gate is shown, where the switches SW-A and SW-B acts as inputs and light bulb acts as output. The input logics for SW-A and SW-B are (0 0), (0 1), (1 0), and (1 1). The output logics for bulb against the inputs are 0, 0, 0 and 1.

The logic representation for the circuit is shown in AND gate, where the switches SW-A and SW-B acts as inputs and light bulb acts as output.

RSLogix 5000 AND Rung is depicted at bottom. The central section consists of Input_A, <Local:1:I.Data.0>; Input_B, <Local:1:I.Data.1>; and Input_Y, <Local:2:O.Data.0> arranged in series. On the left, one end of each, Input A and Input B and with open switches in series are connected to line L1. On the right, one end of the Output_Y is connected to line L2.

Return to Figure 4-5

OceanofPDF.com

Figure 4-6 Extended Description (Chapter 4)

Return to Figure 4-6

A logic symbol for two input OR gate is shown. The two inputs are A and B with Y as the output. The truth table for OR gate is shown. The input logics for A and B are as (0 0), (0 1), (1 0) and (1 1). The output logics for Y against the inputs are as 0, 1, 1 and 1.

Return to Figure 4-6

OceanofPDF.com

Figure 4-7 Extended Description (Chapter 4)

Return to Figure 4-7

Four logic symbols for two input OR gate are shown. The two inputs are A and B with Y as the output. When A equals 0 and B equals 0, output Y equals 0. When A equals 0 and B equals 1, output Y equals 1. When A equals 1 and B equals 0, output Y equals 1. When A equals 1 and B equals 1, output Y equals 1.

The truth table for OR gate is shown. The input logics for A and B are as (0 0), (0 1), (1 0) and (1 1). The output logics for Y against the inputs are as 0, 1, 1 and 1.

Return to Figure 4-7

OceanofPDF.com

Figure 4-8 Extended Description (Chapter 4)

Return to Figure 4-8

The hardwired circuit consists of a battery, 2 parallel switches, and a light bulb connected in series. The positive terminal of the battery is connected to the joining point of switches SW-A and SW-B. The negative terminal of the battery is connected to one end of the bulb. The other end of the bulb is connected to the other joining point of both the switches. Both switches are in the open position.

The truth table of OR gate is shown, where the switches SW-A and SW-B acts as inputs and light bulb acts as output. The input logics for SW-A and SW-B are (0 0), (0 1), (1 0), and (1 1). The output logics for bulb against the inputs are 0, 1, 1 and 1.

The logic representation for the circuit is shown in OR gate, where the switches SW-A and SW-B acts as inputs and light bulb acts as output.

RSLogix 5000 OR Rung is depicted at bottom. The central section consists of Input_A, <Local:1:I.Data.0>; and Output_Y, <Local:2:O.Data.0> arranged in series. Input_B, <Local:1:I.Data.1> is connected across Input_A in the parallel combination. On the left, one end of each, Input A and Input B and with open switches in series are connected to line L1. On the right, one end of the Output_Y is connected to line L2.

Return to Figure 4-8

OceanofPDF.com

Figure 4-9 Extended Description (Chapter 4)

Return to Figure 4-9

The NOT gate has input A and the output comes out as A bar or NOT A.

The truth table for NOT gate is also shown where if the input logic is 1 output logic is 0 and if the input logic is 0 output logic is 1.

Return to Figure 4-9

OceanofPDF.com

Figure 4-10 Extended Description (Chapter 4)

Return to Figure 4-10

The hardwired circuit consists of a battery, a N.C. pushbutton, and a light bulb connected in series. The positive terminal of the battery is connected to one of the ends of the pushbutton. The negative terminal of the battery is connected to one end of the bulb. The other end of the bulb is connected to the other end of the pushbutton.

The truth table of NOT gate is shown, where the pushbutton acts as inputs and light bulb acts as output. The input logics for pushbutton are not pressed (0) and pressed (1). The output logics for bulb against the inputs are on (1) and off (0).

The logic representation for the circuit is shown in NOT gate, where the pushbutton acts as single input and light bulb acts as output.

RSLogix 5000 NOT Rung is depicted at bottom. The central section consists of pushbutton <Local:1:I.Data.0> and light <Local:2:O.Data.0> arranged in series. On the left is the pushbutton acts as input and its one end is connected to line L1. On the right, light bulb acts as output and its one end is connected to the line L2.

Return to Figure 4-10

OceanofPDF.com

Figure 4-11 Extended Description (Chapter 4)

Return to Figure 4-11

An AND gate is shown with input A equals 1. A NOT gate is shown whose input is B equals 0 where pressure switch is open. The output of this gate is 1 which is connected as input to the AND gate. The final output comes out as 1 which shows low pressure indicator is on and so logic is 1. In second circuit an AND gate is shown with input A equals 1. A NOT gate is shown whose input is B equals 1 where pressure switch is closed. The output of this gate is 0 which is connected as input to the AND gate. The final output comes out as 0 which shows low pressure indicator is off and so logic is 0. The power is on in both the circuits. Thirdly a truth table is shown where the pressure switch logics are 0 and 1. The power logics are 1 and 1. The pressure indicator logics are 1 and 0.

Return to Figure 4-11

OceanofPDF.com

Figure 4-12 Extended Description (Chapter 4)

Return to Figure 4-12

A two-input NAND gate, where A and B act as inputs and Y acts as single output.

The NAND gate truth table is shown. The input logics for A and B are (0 0), (0 1), (1 0), and (1 1). The output logics for Y are 1, 1, 1, and 0.

RSLogix 5000 NAND Rung is depicted at bottom. The central section consists of Input_A, <Local:1:I.Data.0>; Input_B, <Local:1:I.Data.1>; and Output_Y, <Local:2:O.Data.0> arranged in series. On the left, Input_A and Input_B, each with a switch, are connected to line L1. On the right, Output_Y is connected to the line L2.

Return to Figure 4-12

OceanofPDF.com

Figure 4-13 Extended Description (Chapter 4)

Return to Figure 4-13

The logic symbol for two input NOR gate is shown. The two inputs are A and B with Y as the output.

The truth table for OR gate is shown. The input logics for A and B are as (0 0), (0 1), (1 0) and (1 1). The output logics for Y against the inputs are as 1, 0, 0 and 0.

Return to Figure 4-13

OceanofPDF.com

Figure 4-14 Extended Description (Chapter 4)

Return to Figure 4-14

A two-input XOR gate, where A and B act as inputs and Y acts as single output.

The XOR gate truth table is shown. The input logics for A and B are (0 0), (0 1), (1 0), and (1 1). The output logics for Y are 0, 1, 1, and 1.

RSLogix 5000 XOR Rung is depicted at bottom. The central section consists of Input_A, <Local:1:I.Data.0>; Input_B, <Local:1:I.Data.1>; and Pilot_Light, <Local:2:O.Data.0> connected in series. Two other inputs, Input_A, <Local:1:I.Data.0>, and Input_B, <Local:1:I.Data.1> in the series combination connected across the main series, Input_A and Input_B. On the left, Input_A and Input_B, each with a switch, are connected to line L1. On the right, Pilot_Light is connected to the line L2.

Return to Figure 4-14

OceanofPDF.com

Figure 4-15 Extended Description (Chapter 4)

Return to Figure 4-15

Column is titled logic symbol, column 2 is titled logic statement, column 3 is titled Boolean equation, and column 4 is titled Boolean notation. Row 1: The logic symbol for AND gate is shown. The logic statement is “Y is 1 if A and B are 1.” The Boolean equation is Y equals A dot B or Y equals A B. Row 2: The logic symbol for OR gate is shown. The logic statement is “Y is 1 if A or B is 1.” The Boolean equation is Y equals A plus B. Row 3: The logic symbol for NOT gate is shown. The logic statement is “Y is 1 if A is 0. Y is 0 if A is 1.” The Boolean equation is Y equals A bar. The fourth column shows Boolean notations. The symbol and meaning is as follows: dot, and; plus, or; minus, not; asterisk, invert; equal, result in.

Return to Figure 4-15

OceanofPDF.com

Figure 4-16 Extended Description (Chapter 4)

Return to Figure 4-16

1. An AND gate is shown with inputs as A and B and output as A B.
2. An OR gate is shown with inputs as A and B and output as A plus B.
3. A NAND gate is shown with inputs as A and B and output as A B bar.
4. A NOR gate is shown with inputs as A and B and output as A plus B bar.
5. A NOT gate is shown with input as A and output as A bar.

Return to Figure 4-16

OceanofPDF.com

Figure 4-17 Extended Description (Chapter 4)

Return to Figure 4-17

Set 1: Two inputs A and B are connected to an AND gate where the output is A B. The output is connected as input to an OR gate whose second input is C. The final output comes out as Y equals A B plus C. Set 2: Two inputs A and B are connected to an OR gate where the output is A plus B. The output is connected as input to an AND gate whose second input is C. The final output comes out as Y equals A plus B times C. Set 3: Two inputs A and B are connected to a NAND gate where the output is A B bar. The output is connected as input to an OR gate whose second input is C. The final output comes out as Y equals A B bar plus C. Set 4: Two inputs A and B are connected to a NOR gate where the output is A plus B whole bar. The output is connected as input to an AND gate whose second input is C. The final output comes out as Y equals A plus B whole bar times C.

Return to Figure 4-17

OceanofPDF.com

Figure 4-18 Extended Description (Chapter 4)

Return to Figure 4-18

Two inputs A and B are connected to an AND gate where the output is A B. The output is connected as input to an OR gate whose second input is C. The final output comes out as Y equals A B plus C.

Return to Figure 4-18

OceanofPDF.com

Figure 4-19 Extended Description (Chapter 4)

Return to Figure 4-19

Two inputs B and C are connected to an AND gate where the output is B C. The output is connected as input to an OR gate whose second input is D. The output comes out as equals B C plus D. This output is then connected as input to another AND gate whose second input is A. The final output comes out as Y equals A times B C plus D.

Return to Figure 4-19

OceanofPDF.com

Figure 4-20 Extended Description (Chapter 4)

Return to Figure 4-20

An OR gate is shown with two inputs A and B which results in output as A plus B. A plus is connected to an AND gate as the input. The other input is C. A NOT gate is shown with input as D after which the output comes out as D bar. D bar behaves as the third input of the AND gate. The final output is Y.

Return to Figure 4-20

OceanofPDF.com

Figure 4-21 Extended Description (Chapter 4)

Return to Figure 4-21

A NOT gate is shown whose input is A and the output is A bar which again becomes the input of an AND gate 1. There is another NOT gate whose input is B and the output is B bar which again becomes the input of an AND gate 2. The other input of AND gate 1 is B and the second input of AND gate 2 s A. The output of AND gate 1 is A bar B and the output of AND gate 2 is A B bar. A bar B and A B bar is connected to an OR as inputs and the final output of OR gate is Y.

Return to Figure 4-21

OceanofPDF.com

Figure 4-22 Extended Description (Chapter 4)

Return to Figure 4-22

There are two vertical supply lines L1 and L2 at two ends. Three horizontal lines calswitch rung runs between these supply lines. These supply lines are calswitch rail. In the first horizontal line there is a pushbutton stop which is connected to another pushbutton start. An open relay contact equivalent is connected across the start button. Then the motor coil M is connected which in turn is connected to a closed overload switch. In the second horizontal line there is a closed relay contact equivalent M connected to a motor stop switch coil G. The third horizontal line contains an open relay contact equivalent M connected to a motor run switch coil R.

Return to Figure 4-22

OceanofPDF.com

Figure 4-23 Extended Description (Chapter 4)

Return to Figure 4-23

There are two vertical supply lines L1 and L2 at two ends. Three horizontal lines calswitch rung runs between these supply lines. These supply lines are calswitch rail. In the first horizontal line a pushbutton stop is connected to an input module which is then connected to rung 1. The rung 1 consists of a pushbutton stop which is connected to another pushbutton start. An open relay contact equivalent is connected across the start button. Then an open overload switch is connected to the start which in turn is connected to a relay coil equipment M. This line is then connected to an output module which in turn is connected to a motor switch M. M is connected to L2. In the second horizontal line a pushbutton start is connected to an input module which is then connected to rung 2. The rung 2 consists of a closed relay contact equivalent M which in turn is connected to a relay coil equipment G. G is connected to an output module which is again connected to a stop relay switch G. In the third horizontal line a closed overload is connected to an input module which is then connected to rung 3. The rung 3 consists of an open relay contact equivalent M which in turn is connected to a relay coil equipment R. R is connected to an output module which is again connected to a start relay switch R.

Return to Figure 4-23

OceanofPDF.com

Example 4-1 Extended Description (Chapter 4)

Return to Example 4-1

A relay schematic is shown with two limit switches L S 1, L S 2 and solenoid switch S Q L connected in series.

A ladder logic program is shown with ladder limit switches L S 1, L S 2 as A and B and solenoid switch S Q L as Y.

The gate logic is shown with inputs A and B entering an AND gate and leaving the gate as output Y.

Return to Example 4-1

OceanofPDF.com

Example 4-2 Extended Description (Chapter 4)

Return to Example 4-2

A relay schematic is shown with two limit switches L S 1, L S 2 connected in parallel and solenoid switch S Q L connected in series with the limit switches.

A ladder logic program is shown with ladder limit switches L S 1, L S 2 as A and B connected in parallel and solenoid switch S Q L as Y connected in series with the limit switches.

The gate logic is shown with inputs A and B entering an OR gate and leaving the gate as output Y.

Return to Example 4-2

OceanofPDF.com

Example 4-3 Extended Description (Chapter 4)

Return to Example 4-3

A relay schematic is shown with two limit switches L S 1, L S 2 connected in parallel and pressure switch P S connected in series with the limit switches. A pilot light P L is connected in series with PS.

A ladder logic program is shown with ladder limit switches L S 1, L S 2 as A and B connected in parallel and pressure switch P S as C connected in series with the limit switches. The output Y is the pilot light P L connected in series with P S.

The gate logic is shown with inputs A and B entering an OR gate and leaving the gate as output A plus B. This output is the input of an AND gate whose second input is C. The output of AND gate is Y.

Return to Example 4-3

OceanofPDF.com

Example 4-4 Extended Description (Chapter 4)

Return to Example 4-4

A relay schematic is shown with two limit switches L S 1, L S 2 connected in parallel and flow switches F S 1 and F S 2 are also connected in parallel and thus finally in series with the limit switches. A pilot light P L is connected in series with the flow switches.

A ladder logic program is shown with ladder limit switches L S 1, L S 2 as A and B connected in parallel and flow switches F S 1 and F S 2 also connected in parallel and in series with the limit switches as C and D. The output Y is the pilot light P L connected in series with the flow switches.

The e gate logic is shown with inputs A and B entering an OR gate and leaving the gate as output A plus B. This output is the input of an AND gate. There is another OR gate with inputs C and D and the output coming out as C plus D. C plus D behaves as the second input of the AND gate. The final output of the AND gate is Y.

Return to Example 4-4

OceanofPDF.com

Example 4-5 Extended Description (Chapter 4)

Return to Example 4-5

A relay schematic is shown where two limit switches L S 1 and L S 2 are connected in series and they are in parallel with a third limit switch L S 3. These switches are connected in series with a horn.

A ladder logic program is shown where two limit switches L S 1 and L S 2 as inputs A and B are connected in series and they are in parallel with a third limit switch L S 3 as input C. These switches are connected in series with a horn as output Y.

A logic gate is shown where two inputs A and B enter an AND gate and comes out as output AB which is the input of an OR gate. The second input of OR gate is C. Thus the final output is Y.

Return to Example 4-5

OceanofPDF.com

Example 4-6 Extended Description (Chapter 4)

Return to Example 4-6

A relay schematic is shown where two light switches L S 1 and L S 2 are connected in series and they both are connected in parallel to two other light switches L S 3 and L S 4. These light switches are connected in series with a pilot light P L.

A ladder logic program is shown where two light switches L S 1 and L S 2 as inputs A and B are connected in series and they both are connected in parallel to two other light switches L S 3 and L S 4 which are as inputs C and D. These light switches are connected in series with a pilot light P L acting as output Y.

A logic gate representation of the expression is shown. There are inputs A and B entering an AND gate and another two inputs C and D entering another AND gate. The output of the first AND and also the second AND gate enters as the inputs of an OR gate. The final output of this OR gate is Y.

Return to Example 4-6

OceanofPDF.com

Example 4-7 Extended Description (Chapter 4)

Return to Example 4-7

A relay schematic is shown where a light switch L S 1 is connected in series with a push button P B which is again series with a solenoid valve S Q L.

A ladder logic program is shown where a light switch L S 1 as input A is connected in series with a push button P B as input B which is again series with a solenoid valve S Q L acting as output Y.

A logic gate representation is shown where input B enters a NOT gate and comes out as B bar. B bar is the input of an AND gate whose second input is A. The final output of the AND gate us Y.

Return to Example 4-7

OceanofPDF.com

Example 4-8 Extended Description (Chapter 4)

Return to Example 4-8

A relay schematic is shown where two push buttons P B 1 and P B 2 are connected in series. If P B 1 is on then P B 2 is off and vice versa. These buttons are connected in series with a pilot light P L.

A ladder logic program is shown where two push buttons P B 1 and P B 2 acting as inputs A and B are connected in series. If P B 1 is on then P B 2 is off and vice versa. These buttons are connected in series with a pilot light P L acting as output Y.

A logic representation is shown where input A is connected to a NOT gate. The output of NOT gate is the input of an AND gate whose second input is B. There is another NOT gate whose input is B. The output of this NOT gate is connected as the input of another AND gate whose second input is A.

Return to Example 4-8

OceanofPDF.com

Example 4-9 Extended Description (Chapter 4)

Return to Example 4-9

A relay schematic is shown where the stop signal consists of two push buttons PB3 and PB4 acting as inputs C and D are connected in series. The stop button is then connected to a start button consisting of two push buttons P B 1 and P B 2 acting as inputs A and B and also a motor M. All these are connected in parallel. They are then connected in series with a motor output M.

A ladder logic program is shown where the stop signal consists of two push buttons PB3 and PB4 acting as inputs C and D are connected in series. The stop button is then connected to a start button consisting of two push buttons P B 1 and P B 2 acting as inputs A and B and also a motor M. All these are connected in parallel. They are then connected in series with a motor output M.

A gate logic is shown where starts A and B are connected as inputs to an OR gate. The output of the Or gate is connected as an input to an AND gate whose other two inputs are stop signals C ad D. The final output of the AND gate is M. This output is also connected as feedback input to the OR gate.

Return to Example 4-9

OceanofPDF.com

Figure 4-24 Extended Description (Chapter 4)

Return to Figure 4-24

The AND logic is encircled in the word level programming and the move or logical operation is selected. The input A enters the AND gate. This instruction tells the processor to perform an AND operation on B3 5 and B3 7 and to store the result in destination B3 10 when input device A is true. The bits for source A B3 5 are 1100110011001100 and the bits for source B3 7 are 1111111100000000. The destination B3 10 bits are shown as 1100110000000000.

Return to Figure 4-24

OceanofPDF.com

Figure 4-25 Extended Description (Chapter 4)

Return to Figure 4-25

The input enters the OR gate. This instruction tells the processor to perform an OR operation on B3 1 and B3 2 and to store the result in destination B3 20 when input device is true. The bits for source A B3 1 are 1100110011001100 and the bits for source B B3 2 are 1111111100000000. The destination B3 20 bits are shown as 1111111111001100.

Return to Figure 4-25

OceanofPDF.com

Figure 4-26 Extended Description (Chapter 4)

Return to Figure 4-26

The XOR logic is encircled in the word level programming and the move or logical operation is selected. The input enters the XOR gate. This instruction tells the processor to perform a XOR operation on I1.0 and I3.0 and to store the result in destination O4.0 when input device is true. The bits for source A I1.0 are 0000000010101010 and the bits for source B I3.0 are 0000000011101011. The destination O.40 bits are shown as 0000000001000001. An image of rotating cam limit switch is also shown.

Return to Figure 4-26

OceanofPDF.com

Figure 4-27 Extended Description (Chapter 4)

Return to Figure 4-27

The AND logic is encircled in the word level programming and the move or logical operation is selected. The input enters the AND gate with Source A, Source B, and Dest having the following information: 2#0000_0000_0000_0000_0000_0000_1010_1010; 2#0000_0000_0000_0000_0000_0000_1110_0011; and 2#0000_0000_0000_0000_0000_0000_1010_0010.

On the left, Inputs A and B having slots 1 and 3 respectively are depicts. On the right, Output with slot 2 is depicted.

At bottom, a truth table for AND gate is depicted. Input as sources A and B have following information: (1,1), (0,1), (1,1), (0,0), (1,0), (0,0), (1,1), and (0,1). The output as destination has the following information: 1, 0, 1, 0, 0, 0, 1, and 0.

Return to Figure 4-27

OceanofPDF.com

Figure 4-28(a-b) Extended Description (Chapter 4)

Return to Figure 4-28(a-b)

Figure (a) shows a relay schematic in which a light switch L S 1 is connected in series with two light switches L S 2 and L S 3 connected in parallel. They are further connected in series with a light R.

Figure (b) shows a relay schematic in which push button P B 1, light switch L S 1 and push switch P S 1 are connected in parallel. They are then connected in series with a push button P S 2 which is again connected in series with a horn H.

Return to Figure 4-28(a-b)

OceanofPDF.com

Figure 4-29 Extended Description (Chapter 4)

Return to Figure 4-29

Switches S1, S2 and S3 acts as inputs A, B and C. A and B are the inputs of an AND gate whose output is AB which is again input of an OR gate. B and C are inputs of an OR gate whose output is B plus C which is again an input of an AND gate. B and C are inputs of a third AND gate whose output is B C. The output B C acts as the input of the earlier AND gate and the output of this AND gate is B C times B plus C. The output B C times B plus C is the second input of the first OR gate whose output comes out as Y in the form of a pilot light P L.

Return to Figure 4-29

OceanofPDF.com

Figure 4-30 Extended Description (Chapter 4)

Return to Figure 4-30

Switches S1, S2, S3 and S4 acts as inputs A, B, C and D. A and B enters as inputs in a NOR gate. C and D enters as input in a NAND gate. The outputs of NAND and NOR are the inputs of a NOR gate whose output is Y and comes out in the form of a pilot light PL.

Return to Figure 4-30

OceanofPDF.com

Figure 4-31(a-f) Extended Description (Chapter 4)

Return to Figure 4-31(a-f)

(a) Two inputs A and B are fed to OR gate. The output from OR gate along two inputs C and D are fed to AND gate. The output from the AND gate is labeled Y.

(b) Two inputs A and B are fed to the first OR gate and the output from this OR gate is fed to the AND gate as first input. Input C is fed to the NOT gate and output from this NOT gate is fed to the second OR gate as its first input. The second input to this OR gate is labeled D. The output from the second OR gate is fed to the AND gate as its second input. The output from the AND gate is labeled Y.

(c) Two inputs A and B are fed to the AND gate. The output from AND gate along with an input C are fed to OR gate. The output from the OR gate is labeled Y.

(d) Two inputs A and B are fed to the first AND gate and the output from this AND gate is fed to the OR gate as first input. Input C is fed to the OR gate as its second input. The output from the OR gate is fed to the second AND gate as its first input. The second input for this AND gate is labeled D. An input E is fed to the NOT gate. The output from this NOT gate is fed to the second AND gate as its third input. The output from the second AND gate is labeled Y.

(e) The inputs for the first and the second OR gates are A, B and C, D, respectively. The outputs from these OR gates are fed to the AND gates as its inputs. The output from the AND gate is labeled Y.

(f) The inputs for the first, second, and third NOT gates are labeled B, D, and F, respectively. The first inputs for the first, second, and third AND gates are labeled A, C, and E, respectively. The outputs from the first, second, and third NOT gates are fed to the first, second, and third AND gates as their second inputs. The outputs from each AND gate is fed to the OR gate as its inputs. The output from the OR gate is labeled Y.

Return to Figure 4-31(a-f)

OceanofPDF.com

Figure 4-32 Extended Description (Chapter 4)

Return to Figure 4-32

A NOR gate is shown whose inputs are A and B and the output is connected as an input to an AND gate. Then an AND gate is shown whose inputs are C and D and the output is connected as an input to the final AND gate whose output is Y and is connected an alarm.

Return to Figure 4-32

OceanofPDF.com

A Table Graphic Extended Description (Chapter 5)

Return to A Table Graphic

From left to right, the first row is labeled as symbol, name, bit status, and instruction status. From top to bottom, the last two rows of the first column show a symbol using a pair of alphabet E in upper case, with their horizontal lines facing away and the middle horizontal line extended as a terminal, and a symbol same as the previous one but has a right slanting line over it. From top to bottom, the last two rows of the second column are labeled as X I C and X I O. The sub-divisions of the second row and the third column are labeled as zero and one. The sub-divisions of the third row and the third column are labeled as zero and one. The sub-divisions of the second row and the fourth column are labeled as false and true. The sub-divisions of the third row and the fourth column are labeled as true and false. The second row of the last three columns are highlighted in red. The third row of the last three columns are highlighted in blue.

Return to A Table Graphic

OceanofPDF.com

Figure 5-1 Extended Description (Chapter 5)

Return to Figure 5-1

From top to bottom, the blocks are labeled as data files and program files. The close-up view of data files is shown as 10 files arranged along a line. From back to forth, the 10 files are numbered as 0, 1, 2, 3, 4, 5, 6, 7, 8, and 9 to 255. The last nine files, which are from zero through eight are labeled as designated. The file in the front is labeled as user defined. From back to forth, the last nine files are labeled as the output file, input file, status file, bit file, timer file, counter file, control file, integer file, and floating point file. From top to bottom, the file in the front has labels B bit, T timer, C counter, R control, N integer, F floating point, S T string, and A ASCII. The close-up view of program files is shown as four files arranged along a line. From back to forth, the four files are numbered as 0, 1, 2, and 3 to 255. The last three files, which are from zero through two are labeled as main program. The file in the front is labeled as subroutines. From back to forth, the last three files are labeled as the system file zero, system file one, and system file two. The file in the front has a label program files 3 to 255.

Return to Figure 5-1

OceanofPDF.com

Figure 5-2 Extended Description (Chapter 5)

Return to Figure 5-2

A negative sign placed between a box is connected to the left side of the folder, through a dotted horizontal line. The dotted vertical line is connected to three dotted horizontal lines. From top to bottom, the horizontal lines are connected to a symbol of file with a red circle and a left slanting line over it, a symbol of file with a red circle and a left slanting line over it, and a symbol of ladder. From top to bottom, the symbols are labeled as S Y S zero underscore, S Y S one underscore, and L A D two underscore.

Return to Figure 5-2

OceanofPDF.com

Figure 5-3 Extended Description (Chapter 5)

Return to Figure 5-3

A negative sign placed between a box is connected to the left side of the folder, through a dotted horizontal line. The dotted vertical line is connected to 10 dotted horizontal lines. From top to bottom, the horizontal lines are connected to a symbol of file with a triple headed arrow joined together and an arrow head pointing in different directions, and nine symbols of a file. From top to bottom, the symbols are labeled as cross reference, O 0-Output, I 1-Input, S 2-Status, B 3-Binary, T 4-Timer, C 5-Counter, R 6-Control, N 7-Integer, and F 8-Float.

Return to Figure 5-3

OceanofPDF.com

Figure 5-4 Extended Description (Chapter 5)

Return to Figure 5-4

From top to bottom, both the modules are labeled from zero through 15. Each label of the input module is connected with a label of the output module, through a terminal. 16 switches are placed one over the other and connected to the left side of the input module, separately. The third switch is connected to the top side of the block labeled I : 1/2 (highlighted in blue). Label 11 on the output module is connected to the bottom side of the block labeled O : 2/11 (highlighted in red). The block labeled I : 1/2 is placed over the block labeled O : 2/11. Five labels are placed between them. From left to right, the labels are input or output, separator, slot number, bit designator, and bit number. All the labels point at both blocks.

Return to Figure 5-4

OceanofPDF.com

Figure 5-5 Extended Description (Chapter 5)

Return to Figure 5-5

The close-up view shows a row of 16 columns. It is labeled as word corresponding to input module. A label input image is placed over the data table files. From the left, the tenth column is labeled as one and the fourteenth column is labeled as zero. A vertical line L 1 is connected to the third terminal of the input module, through N O switch labeled as off (open). The vertical line L 1 is connected to the seventh terminal of the input module, through N C switch labeled as on (closed). The input module points at the label one and the label zero on the data table files, separately.

Return to Figure 5-5

OceanofPDF.com

Figure 5-6 Extended Description (Chapter 5)

Return to Figure 5-6

The close-up view shows a row of 16 columns. It is labeled as word corresponding to output module. A label output image is placed over the data table files. From the left, the ninth column is labeled as zero and the twelfth column is labeled as one. A vertical line L 2 is connected to the second terminal of the output module, through light labeled as off. The vertical line L 2 is connected to the fifth terminal of the input module, through light labeled as on. The label one and the label zero on the data table files points at the left side of the output module.

Return to Figure 5-6

OceanofPDF.com

Figure 5-7 Extended Description (Chapter 5)

Return to Figure 5-7

An input module is placed horizontally and has 21 terminals. From the left, the module is labeled as A C COM, I : 0, I : 1, I : 2, I : 3, A C COM, I : 4, I : 5, I : 6, I : 7, I : 8, I : 9, I : 10, I : 11, I : 12, I : 13, I : 14, I : 15, I : 16, I : 17, I : 18, and I : 19. Two vertical lines labeled as L 2 are connected to the terminals labeled as A C COM. A vertical line L 1 is connected to a horizontal line, which is connected to the terminals labeled as I : 0, I : 1, I : 2, and I : 3, through N O limit switches. Another vertical line L 1 is connected to a horizontal line, which is connected to the terminals labeled as I : 4, I : 5, I : 6, I : 7, I : 8, I : 9, I : 10, I : 11, I : 12, I : 13, I : 14, I : 15, I : 16, I : 17, I : 18, and I : 19, through N O limit switches. A label discrete input is preset above the ladder diagram. An output module of 20 terminals is placed horizontally below the input module. From the left, except for the first three unlabeled terminals, the remaining terminals are labeled as VAC VDC, O/0, VAC VDC, O/4, VAC VDC, O/2,O/3, VAC VDC O/4,O/5, O/6, O/7, VAC VDC, O/8, O/9, O/10, and O/11. A line L 1 is connected to the first VAC VDC terminal, through a node connected to the first unlabeled terminal. A line L 2 is connected to the terminal labeled as O/0, through a node connected to the second unlabeled terminal and a relay coil. The third unlabeled terminal is grounded. V A C 2 is connected to the second VAC VDC terminal. V A C 2 C O M is connected to the first terminal labeled O/4, through a solenoid. V D C 1 is connected to the third VAC VDC terminal. V D C 1 C O M is connected to the terminal labeled O/3, through a node and the second relay coil. The node is connected to a terminal labeled O/2, through a solenoid. V D C 2 is connected to the fourth VAC VDC terminal. V D C 2 C O M is connected to a terminal labeled as O/7, through a node. The node is connected to the second O/4 terminal, through two nodes. The first node is connected to the terminal labeled as O/5 and the second node is connected to the terminal labeled as O/6. V D C 3 is connected to the fifth VACVDC terminal. V D C 3 C O M is connected to a terminal labeled as O/11, through a node. The node is connected to the second O/8 terminal, through two nodes. The first node is connected to the terminal labeled as O/9 and the second node is connected to the terminal labeled as O/10. A label discrete outputs is placed below the ladder diagram.

Return to Figure 5-7

OceanofPDF.com

Figure 5-8 Extended Description (Chapter 5)

Return to Figure 5-8

In clockwise direction, the labels are start, input scan, program scan, and house-keeping. At the label input scan, an N C pushbutton is shown, which is labeled as the status of external inputs is written to the input image table (file or register). At the label program scan, a ladder diagram is shown, which is labeled as each ladder rung is scanned and solved using the date in the input file; the resulting logic is written to the output image table (file or register). From left to right, an X I O symbol (highlighted in green) and an output energize are connected in series along the rung in the ladder diagram. At the label output scan, a bulb is shown, which is labeled as the output image date is transferred to the external output circuits, turning the output device on or off. Labels internal checks on memory, speed, and operation and service any communication requests are placed at the left side of the label house-keeping.

Return to Figure 5-8

OceanofPDF.com

Figure 5-9 Extended Description (Chapter 5)

Return to Figure 5-9

From the left, the parts of the rectangle are labeled as input image table file and output image table file. A label input modules points at the left side of the rectangle, through a label input data. The right side of the rectangle points at the output modules, through a label output data. From left to right, X I C symbol, X I O symbol, the second X I C symbol, and an output energize are connected in series along a rung. The X I C and X I O symbols are highlighted and labeled as check or compare or examine specific conditions. The label input image table file in the rectangle points at the X I O symbol, through a label examine date. The output energize points at the label output image table file in the rectangle, through a label return result. An upwards pointing arrow labeled as take some action points at the output energize.

Return to Figure 5-9

OceanofPDF.com

Figure 5-10 Extended Description (Chapter 5)

Return to Figure 5-10

There are four vertical lines. From the left, the first vertical line is labeled as L 1 and the fourth vertical line is labeled as L 2. Two horizontal lines are connected at the right side of the vertical line L 1. A label input module is placed over the first horizontal line. From top to bottom, N O pushbutton labeled as P B 1 and N C pushbutton labeled as P B 2 are connected along the horizontal lines connected to the line L 1. There are two rungs between the second and the third vertical line. From left to right, X I C symbol labeled as P B 1 and an output energize labeled as P L 1 are connected in series along the first rung. From left to right, X I C symbol labeled as P B 2 and an output energize labeled as P L 2 are connected in series along the second rung. A label program is placed over the first rung. Two horizontal lines are connected at the left side of the vertical line L 2. A label output module is placed over the first horizontal line. From top to bottom, light labeled as P L 1 and light labeled as P L 2 are connected along the horizontal lines connected to the line L 2. The first two rows of the first column are merged and labeled as input device state. The first row of the second and the third columns are merged and labeled as input. The first row of the fourth and the fifth columns are merged and labeled as output. From the left, the second row except for the first column is labeled as P B 1, P B 2, P L 1, and P L 2. From left to right, the third row is labeled as no P Bs are pressed, zero, one, zero, and one. From left to right, the fourth row is labeled as P B 1 is pressed, one, one, one, and one. From left to right, the fifth row is labeled as P B 2 is pressed, zero, zero, zero, and zero. From left to right, the sixth row is labeled as both P Bs are pressed, one, zero, one, and zero.

Return to Figure 5-10

OceanofPDF.com

Figure 5-11 Extended Description (Chapter 5)

Return to Figure 5-11

A label start is placed over the ladder diagram and a label end below the ladder diagram. From left to right, two X I C symbols and an output energize are connected in series along the first rung. From left to right, the third X I C symbol and the second output energize are connected in series along the second rung. The fourth X I C symbol is connected across the third X I C symbol. From left to right, the fifth X I C symbol, an X I O symbol, the sixth X I C symbol, and the third output energize are connected in series along the third rung. From left to right, the seventh X I C symbol, the eighth X I C symbol, and the fourth output energize are connected in series along the fourth rung. From left to right, the ninth X I C symbol, and the fifth output energize are connected in series along the fifth rung. The second X I O symbol is connected across the ninth X I C symbol. The input image table consists of a row of 16 columns. From left to right, the input image table is labeled as 0, 0, 0, 1, 0,0, 0, 1, 0, 0, 1, 0, 0, 0, 1, and 0. An input module with two columns of eight terminals is labeled as step one read input module. The left side of the input module points at the right side of the input image table. The second and the sixth terminals of the first column are highlighted in red. The first and the fifth terminals of the second column are highlighted in red. The ladder diagram is labeled as step two solve the ladder program. The output image table consists of a row of 16 columns. From left to right, the input image table is labeled as 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, and 0. An output module with two columns of eight terminals is labeled as step three transfer to output module. The right side of the output image table points at the left side of the output module. The second and the fifth terminals of the first column are highlighted in red. The bottom side of the output module points at the top side of the input module.

Return to Figure 5-11

OceanofPDF.com

Figure 5-12 Extended Description (Chapter 5)

Return to Figure 5-12

From top to bottom, the rungs are labeled as zero, one, and two. From left to right, the first X I C symbol and the first output energize are connected in series along the rung zero. The second X I C symbol is connected across the first X I C symbol. From left to right, X I O symbol and the second output energize are connected in series along the rung one. From left to right, the third X I C symbol the fourth X I C symbol, and the third output energize are connected in series along the rung two. Arrow pointing from left to right are placed over the rungs, separately. The arrow above rung zero is labeled as left to right. An arrow labeled top rung to bottom rung points from the top-right side of the ladder diagram towards the label zero at the top-left side of the ladder diagram, through a label END at the bottom of the ladder diagram.

Return to Figure 5-12

OceanofPDF.com

Figure 5-13 Extended Description (Chapter 5)

Return to Figure 5-13

The textural language points at instruction list and structured text. The graphical language points at ladder diagram, functional block diagram, and sequential functional chart.

Return to Figure 5-13

OceanofPDF.com

Figure 5-14(a-c) Extended Description (Chapter 15)

Return to Figure 5-14(a-c)

In part a, the ladder diagram has one rung. From left to right, N O pushbutton labeled as P B 1, N O relay contact labeled C R 1, N C relay contact labeled C R 2, and a relay coil labeled as SOL are connected in series along the rung. An N O limit switch labeled L S 1 is connected across the series connection of the N O pushbutton P B 1 and the N O relay contact C R 1. In part b, ladder diagram labeled equivalent ladder diagram (L D) program has a rung. From left to right, X I C symbol labeled as (P B 1) and A, X I C symbol labeled as (C R 1) and B, XI O symbol labeled as (C R 2) and D, and an output energize labeled as SOL and Y are connected in series along the rung. An X I C symbol labeled as (L S 1) and C is connected across the series connection of the X I C symbol labeled A and X I C symbol labeled B. In part c, the table is labeled as equivalent instruction list (I L) program consists of two columns and five rows. From top to bottom, the first column is labeled as start, and, or, and not, out. From top to bottom, the second column is labeled as P B 1, C R 1, L S 1, C R 2, and SOL.

Return to Figure 5-14(a-c)

OceanofPDF.com

Figure 5-15 Extended Description (Chapter 5)

Return to Figure 5-15

From the left, the ladder logic shows two X I C symbols labeled A and B connected in series, for which a block labeled as AND_BOOL has two inputs A and B on the left side. The second ladder logic shows parallel connection of the X I C symbols labeled A and B, for which a block labeled OR_BOOL has two inputs A and B. The third ladder logic shows series connection of the X I C symbol labeled A and X I O symbol labeled B, for which a block labeled AND_BOOL has two inputs A and B. The input B is inverted. A label functional block diagram equivalent is placed over the first block. A label ladder logic is placed over the first ladder logic.

Return to Figure 5-15

OceanofPDF.com

Figure 5-16 Extended Description (Chapter 5)

Return to Figure 5-16

From left to right, X I C symbol labeled as sensor one, X I C symbol labeled as sensor two, and an output energize labeled as caution P L 1 are connected in series along a rung. The functional block diagram labeled as BAND_01 has two inputs labeled in one and in two on the left side. The first input is connected to the sensor one, through a label zero. The second input is connected to the sensor two, through a label zero. The right side of the block has an output terminal labeled as out, which is connected to a light labeled as P L 1 and caution, through a label zero. A label BAND Boolean and is placed within the block.

Return to Figure 5-16

OceanofPDF.com

Figure 5-17 Extended Description (Chapter 5)

Return to Figure 5-17

From the top, a block labeled initial step one is connected to a block labeled step two, through a wire with a marking in the middle labeled as transition. The step two is connected to a block labeled step three, through the wire with a marking labeled transition. The step three is connected to a circle labeled as stop, through the wire with a marking in the middle labeled as transition. The right side of each block is connected to another block labeled as action. A node at the bottom side of the step two points at the top side of the initial step, through a wire loop with a marking labeled as transition.

Return to Figure 5-17

OceanofPDF.com

Figure 5-18 Extended Description (Chapter 5)

Return to Figure 5-18

From left to right, X I C symbol labeled sensor one, X I C symbol labeled as sensor two, and an output energize labeled SOL 1 are connected in series along the rung. The series connection of the X I C symbol labeled sensor three, X I C symbol labeled sensor four, and X I O symbol (highlighted in green) labeled as sensor five is connected across the series connection of the series connection of the X I C symbol labeled sensor one and X I C symbol labeled as sensor two.

The program reads as:

1. IF Sensor_1 AND Sensor_2 THEN
2. (intended to level 1): SOL_1 := 1;
3. ELSEIF Sensor_3 AND Sensor_4 AND NOT Sensor_5 THEN
4. (intended to level 1): SOL_1 := 1;
5. END_IF;

Return to Figure 5-18

OceanofPDF.com

Figure 5-19 Extended Description (Chapter 5)

Return to Figure 5-19

At the top, the symbol of X I C instruction is shown, which is labeled as examine if closed (X I C) and examine-on. In the middle, a vertical line is connected to the fifth terminal of an input module, through an N C pushbutton labeled as I : 1/4. The input module has eight terminals below two columns of four dots. The top-right dot of the input terminal is shaded red. The right side of the input module points at a label one in a row of 16 columns. From left to right, the columns are labeled as from 15 through zero at the top. The column labeled four has label one within it. The numbers at the top are labeled as bit number and the columns are labeled as status. A symbol of X I C is shown whose rungs are highlighted in green. This X I C instruction is labeled as I: 1/4 at the top and instruction interpreted as true at the bottom. At the bottom, a vertical line is connected to the fifth terminal of an input module, through an N O pushbutton labeled as I : 1/4. The input module has eight terminals below two columns of four dots. The right side of the input module points at a label zero in a row of 16 columns. From left to right, the columns are labeled as from 15 through zero at the top. The column labeled four has the label zero within it. A symbol of X I C is shown. This X I C instruction is labeled as I: 1/4 at the top and instruction interpreted as false at the bottom.

Return to Figure 5-19

OceanofPDF.com

Figure 5-20 Extended Description (Chapter 5)

Return to Figure 5-20

At the top, the symbol of X I O instruction is shown, which is labeled as examine if open (X I O) and examine-off. In the middle, a vertical line is connected to the fifth terminal of an input module, through an N C pushbutton labeled as I : 1/4. The input module has eight terminals below two columns of four dots. The top-right dot of the input terminal is shaded red. The right side of the input module points at a label one in a row of 16 columns. From left to right, the columns are labeled as from 15 through zero at the top. The column labeled four has label one within it. The numbers at the top are labeled as bit number and the columns are labeled as status. A symbol of X I O is shown. This X I O instruction is labeled as I: 1/4 at the top and instruction interpreted as false at the bottom. At the bottom, a vertical line is connected to the fifth terminal of an input module, through an N O pushbutton labeled as I : 1/4. The input module has eight terminals below two columns of four dots. The right side of the input module points at a label zero in a row of 16 columns. From left to right, the columns are labeled as from 15 through zero at the top. The column labeled four has the label zero within it. A symbol of X I O is shown whose rungs are highlighted in green. This X I O instruction is labeled as I: 1/4 at the top and instruction interpreted as true at the bottom.

Return to Figure 5-20

OceanofPDF.com

Figure 5-22 Extended Description (Chapter 5)

Return to Figure 5-22

At the top, symbol of O T E instruction is shown and is labeled as output energize. In the middle, a row of 16 columns is place over a ladder diagram. From left to right, the columns are labeled from 15 through zero at the top. A label output data is placed at the left side of the row. The column labeled as one has a label one within it. This column points at the left side of an output module of eight terminals. The output module consists of two rows of four dots above the terminals and the second dot of the first column is shaded red. From the right side, the second terminal of the output module is connected to a vertical line, through a yellow light labeled on and O : 2/1. A ladder diagram labeled as program is placed at the left side of the output module. From left to right, X I C symbol (highlighted in green) labeled as I : 1/1, X I C symbol (highlighted in green) labeled as I: 1/4,and an output energize (highlighted in green) labeled as O : 2/1 are connected in series along a rung. At the bottom, a vertical line is connected to two rungs on the right. From left to right, the first N C limit switch is connected to the second terminal of an input module, along the first rung. From left to right, the second N C limit switch is connected to the fifth terminal of the input module, along the second rung. The output module consists of two rows of four dots above eight terminals. The second dot of the first column and he first dot of the second column are shaded red. A row with 16 columns with a label input data at the left side is placed above the ladder diagram. From left to right, the row is labeled from 15 through zero at the top. The columns labeled four and one have a label one within them. The second terminal of the input module points at label one in the column labeled four. The fifth terminal of the input module points at label one in the column labeled one.

Return to Figure 5-22

OceanofPDF.com

Figure 5-23 Extended Description (Chapter 5)

Return to Figure 5-23

The ladder diagram on the bottom-left side is labeled as button not actuated. The ladder diagram on the bottom-right side is labeled as button actuated. Both the ladder diagrams at the bottom shows two rungs connected between two vertical lines. Both the ladder diagrams have a label ladder logic program at the top. From left to right, X I C symbol labeled as A and an output energize off labeled as output are connected in series along the first rung of the first ladder diagram. A label false is placed over the X I C symbol labeled as A. From left to right, X I O symbol (highlighted in green) labeled as A and an output energize on are connected in series along the second rung of the first ladder diagram. A label true is placed below the X I O symbol (highlighted in green) labeled as A. From left to right, X I C symbol (highlighted in green) labeled as A and an output energize on labeled as output are connected in series along the first rung of the second ladder diagram. A label true is placed over the X I C symbol (highlighted in green) labeled as A. From left to right, X I C symbol labeled as A and an output energize off are connected in series along the second rung of the second ladder diagram. A label false is placed below the X I O symbol labeled as A. In both the ladder diagrams on the top, the input module has two columns of four dots above eight terminals. The ladder diagram at the top-left side has a rung connected at the right side of a vertical line. From left to right, N O pushbutton labeled as A and fifth terminal of the input module are connected along the horizontal line. The right side of the fifth terminal of the input module points at zero enclosed in a square labeled bit status. The ladder diagram at the top-right side has a rung connected at the right side of a vertical line. In this ladder diagram, the top-right dot of the input module is shaded red. From left to right, N C pushbutton labeled as A and fifth terminal of the input module are connected along the horizontal line. The right side of the fifth terminal of the input module points at one enclosed in a square labeled bit status.

Return to Figure 5-23

OceanofPDF.com

Figure 5-24 Extended Description (Chapter 5)

Return to Figure 5-24

From top to bottom, the rungs are labeled as rung zero and rung one. From left to right, X I C symbol labeled as I/1, X I O symbol labeled as I/2, X I O symbol labeled as I/3, and an output energize labeled as O/1 are connected in series along the rung zero. An X I C symbol labeled as O/1 is connected across the series connection of the X I C symbol labeled as I/1 and the X I O symbol labeled as I/2. From left to right, X I O symbol labeled as I/2, X I C symbol labeled as I/4, and an output energize labeled as O/2 are connected in series along the rung one. A label input is placed over the X I C and X I O symbols connected along the rung zero. A label output is placed over the output energize connected along the rung zero.

Return to Figure 5-24

OceanofPDF.com

Figure 5-25 Extended Description (Chapter 5)

Return to Figure 5-25

From top to bottom, the rungs are labeled as rung zero, rung one, and rung two. The relays are shown as rectangles. From left to right, relay labeled F, relay labeled T, the second relay labeled T, and output energize F are connected in series along the rung zero. From left to right, the third relay labeled T and the first output energize T are connected in series along the rung one. From left to right, the second relay labeled F, the fourth relay labeled T, the fifth relay labeled T, and the second output energize T are connected in series along the rung two. The sixth relay labeled as T is connected across the second relay labeled F. The third output energize T is connected across the second output energize T.

Return to Figure 5-25

OceanofPDF.com

Figure 5-26 Extended Description (Chapter 5)

Return to Figure 5-26

An output module is at the right side of the image output table. Both the image tables show a row of 16 columns, which are labeled zero, except for one column. Both the modules have four columns of four rectangles above the terminals labeled from zero through 16, from the top. From the left, the output image table has a label one in the tenth column, which points at the left side of the output module. The label one is highlighted and labeled as O : 4/6 bit address. The output image table also has a label file zero over the row. The sixth terminal of the output module is connected to a vertical line L 2, through a red bulb labeled as O : 4/6 and energized output. A label output file (O) slot (four) bit (six) points at the top side of the label O : 4/6. The third rectangle of the second column of the output module is shaded red. The twelfth terminal of the input module is connected to a vertical line L 1 on the left side, through a N C limit switch labeled as I : 3/12 and closed switch. A label input file (I) slot (three) bit (12) points at the top side of the label I : 3/12. The first rectangle of the fourth column of the input module is shaded red. The right side of the input module points at a label one in the fourth column of the input image table. The label one is highlighted and labeled as I : 3/12 bit address. The input image table has a label file one over the row. At the bottom, the ladder diagram has one rung. From left to right, an X I C symbol (highlighted in green) labeled as I : 3 and 12 is connected to an output energize labeled as O : 4 and 6 along the rung. A label user programmed rung is placed below the ladder diagram. The label one in the input image table points at the X I C symbol. The output energize of the ladder diagram points at the label one in the output image table.

Return to Figure 5-26

OceanofPDF.com

Figure 5-27 Extended Description (Chapter 5)

Return to Figure 5-27

The input addresses are enclosed in rectangles and the output addresses are enclosed in diamonds. Both the ladder diagrams show two rungs connected between two vertical lines L 1 and L2. From the left, the first ladder diagram has a label inputs and the second ladder diagram has a label outputs. From left to right, N O pushbutton labeled as P B 1 and an input address I : 4/5 are connected in series along the first rung of the first ladder diagram. From left to right, N O limit switch labeled as L S 1 and an input address I : 4/6 are connected in series along the second rung of the first ladder diagram. From left to right, solenoid and an output address O : 2/3 are connected in series along the first rung of the second ladder diagram. From left to right, red light labeled as P L 1 and an output address O : 3/6 are connected in series along the second rung of the second ladder diagram.

Return to Figure 5-27

OceanofPDF.com

Figure 5-28 Extended Description (Chapter 5)

Return to Figure 5-28

From left to right, X I C symbol labeled as A and output energize labeled as C are connected in series along a rung. The X I C symbol labeled as B is connected across the X I C symbol labeled A. A symbol of a ladder diagram, which has a rectangle connected at the bottom of a rung between two vertical lines is shown. The symbol points at the X I C symbol labeled B, through double-headed arrow.

Return to Figure 5-28

OceanofPDF.com

Figure 5-29 Extended Description (Chapter 5)

Return to Figure 5-29

From left to right, X I C symbol labeled as A, X I O symbol labeled as B, and output energize labeled as D are connected in series along a rung. The X IC symbol labeled as C is connected across the series connection of the X I C symbol labeled A and X I O symbol labeled as B.

Return to Figure 5-29

OceanofPDF.com

Figure 5-30 Extended Description (Chapter 5)

Return to Figure 5-30

From left to right, X I C symbol labeled as A and output energize labeled as C are connected in series along a rung. The X I C symbol labeled as B is connected across the X I C symbol labeled A. The parallel connection of the output energize labeled D, and E is connected across the output energize labeled as C.

Return to Figure 5-30

OceanofPDF.com

Figure 5-31 Extended Description (Chapter 5)

Return to Figure 5-31

From left to right, X I C symbol labeled as A and output energize labeled as C are connected in series along a rung. The X I C symbol labeled as B is connected across the X I C symbol labeled A. The series connection of the X I C symbol labeled D and output energize labeled E is connected across the output energize labeled as C.

Return to Figure 5-31

OceanofPDF.com

Figure 5-32 Extended Description (Chapter 5)

Return to Figure 5-32

From left to right, three X I C symbols, and the first output energize are connected in series along a rung. The fourth X IC symbol is connected across the second X I C symbol. The fifth X I C symbol is connected across the series connection of the first three X I C symbols. The series connection of the sixth and the seventh X I C symbols is connected across the parallel connection of the first and the fifth X I C symbols. The series connection of the eight and the ninth X I C symbols is connected across the series connection of the first three X I C symbols. The series connection of the tenth and the second output energize is connected across the first output energize. The series connection of the eleventh and the third output energize is connected across the series connection of the tenth and the second output energize.

Return to Figure 5-32

OceanofPDF.com

Figure 5-33 Extended Description (Chapter 5)

Return to Figure 5-33

From left to right, X I C symbols labeled as A, B, C, and an output energize labeled as Y are connected in series along a rung. The X I C symbol labeled D is connected across the series connection of the X I C symbols labeled as B, and C. The X I C symbol labeled E is connected across the series connection of the X I C symbols labeled as B, and C. The X I C symbol labeled D is highlighted and labeled as a branch within a branch.

Return to Figure 5-33

OceanofPDF.com

Figure 5-34 Extended Description (Chapter 5)

Return to Figure 5-34

From left to right, X I C symbols labeled as A, B, C, and an output energize labeled as Y are connected in series along a rung. The series connection of the X I C symbols labeled D and C is connected across the series connection of the X I C symbols labeled as B, and C. The X I C symbol labeled E is connected across the series connection of the X I C symbols labeled as D, and C. The second X I C symbol labeled C is highlighted and labeled as contact instruction C repeated.

Return to Figure 5-34

OceanofPDF.com

Figure 5-35 Extended Description (Chapter 5)

Return to Figure 5-35

From left to right, X I C open symbols labeled as A, D, and output energize Y, are connected in series along a rung. Series connection of X I C open symbols labeled as B and E, is parallelly connected across the series connection of the X I C open symbols labeled A and D. X I C open symbol labeled as C is parallelly connected between the X I C open symbols labeled as B and E. A label below the ladder diagram is Y = sum of product of A and D, product of B, C, and D, product of B and E, and product of A, C, and E.

Return to Figure 5-35

OceanofPDF.com

Figure 5-36 Extended Description (Chapter 5)

Return to Figure 5-36

From left to right, X I C symbols labeled A, D, and output energize labeled Y are connected in series along a rung. The series connection of the X I C symbols labeled B, C, and D is connected across the series connection of the X I C symbols labeled as A, and D. The series connection of the X I C symbols labeled B, C, and D is highlighted. The series connection of the X I C symbols labeled B, and E is connected across the series connection of the X I C symbols labeled as B, C, and D. The series connection of the X I C symbols labeled A, C, and E is connected across the series connection of the X I C symbols labeled as B, and E. The series connection of the X I C symbols labeled A, C, and E is highlighted.

Return to Figure 5-36

OceanofPDF.com

Figure 5-37 Extended Description (Chapter 5)

Return to Figure 5-37

From left to right, X I C symbols labeled as A, B, C, and an output energize labeled as Y are connected in series along a rung. The series connection of the X I C symbols labeled D, and E is connected across the series connection of the X I C symbols labeled as B, and C. The X I C symbol labeled F is connected across the parallel connection of the X I C symbols labeled as A, and D. A label below the ladder diagram is Boolean expression: Y = sum of product of A, B, and C, product of A, D, and E, product of F and E, and product of F, D, B, and C.

Return to Figure 5-37

OceanofPDF.com

Figure 5-38 Extended Description (Chapter 5)

Return to Figure 5-38

From left to right, X I C symbols labeled A, B, C, and output energize labeled Y are connected in series along a rung. The series connection of the X I C symbols labeled A, D, and E is connected across the series connection of the X I C symbols labeled as A, B, and C. The series connection of the X I C symbols labeled F, and E is connected across the series connection of the X I C symbols labeled as A, D, and E. The series connection of the X I C symbols labeled F, D, B, and C is connected across the series connection of the X I C symbols labeled as F, and E.

Return to Figure 5-38

OceanofPDF.com

Figure 5-39 Extended Description (Chapter 5)

Return to Figure 5-39

From left to right, X I C symbols labeled as one, two, three, four, five, six, seven, and an output energize labeled as internal output coil are connected in series along the rung one. From left to right, X I C symbols labeled as internal output contact, eight, nine, 10, 11, 12, and an output energize labeled as discrete output are connected in series along the rung two. A label discrete inputs is placed over the first rung and the second rung.

Return to Figure 5-39

OceanofPDF.com

Figure 5-40 Extended Description (Chapter 5)

Return to Figure 5-40

Both the ladder diagrams consist of two vertical lines. For hardwired circuit, there is only one rung. From left to right, N O pushbutton P B 1, N C pushbutton P B 2, and pilot light are connected in series along the rung. For the second ladder diagram, there is only one rung. From left to right, X I O symbol labeled as P B underscore 1, X I O symbol labeled as P B underscore 2, and output energize labeled as P L are connected in series along the rung.

Return to Figure 5-40

OceanofPDF.com

Figure 5-41 Extended Description (Chapter 5)

Return to Figure 5-41

Both the ladder diagrams consist of two vertical lines. For hardwired circuit, there are two rungs. From left to right, N O pushbutton P B 1, and relay coil are connected in series along the first rung. From left to right, N C relay contact labeled as C R, and pilot light are connected in series along the second rung. For the second ladder diagram, there is only one rung. From left to right, X I O symbol labeled as P B underscore 1, and output energize labeled as P L are connected in series along the rung.

Return to Figure 5-41

OceanofPDF.com

Figure 5-42 Extended Description (Chapter 5)

Return to Figure 5-42

The first table consists of four columns and four rows. The last three columns of the first row are merged and labeled as the status of the instruction is. The first two rows of the first column are merged and labeled as if the data table bit is. From left to right, the second row except for the first column is labeled as X I C examine if closed, X I O examine if open, and O T E output energize. The symbols of X I C, X I O, and O T E are shown below their names. From left to right, the third row is labeled as logic zero, false, true, and false. From left to right, the fourth row is labeled as logic one, true false, and false. For the ladder diagram, X I C instruction, X I O instruction, and output energize are connected in series along a rung. The X I C and X I O symbols are labeled as input instructions and the O T E is labeled as output instruction. The second table consists of six rows and four columns. The first row of the last column is removed. The first two rows of the first column are merged and labeled as time. The second and the third column of the first row are merged and labeled as instruction outcome. From the left, the second row except for the first column is labeled as X I C, X I O, and O T E. From left to right, the third row is labeled as t 1 (initial), false, true, and false. From left to right, the fourth row is labeled as t 2, true, false, and goes true. The labels in fourth row are in blue. From left to right, the fifth row is labeled as t 3, true, false, and goes false. From left to right, the sixth row is labeled as t 4, false, false, and remains false. The third table consists of six rows and three columns. The first row of the last column is removed. The first two columns of the first row are merged and labeled as input bit status. From left to right, the second row is labeled as X I C, X I O, and O T E. From left to right, the third row is labeled as zero, zero, and zero. From left to right, the fourth row is labeled as one, zero, and one. The labels in the fourth row are in blue. From left to right, the fifth row is labeled as one, one, and zero. From left to right, the sixth row is labeled as zero, one, and zero.

Return to Figure 5-42

OceanofPDF.com

Figure 5-43 Extended Description (Chapter 5)

Return to Figure 5-43

The toolbar is divided into three horizontal sections. From the left, the first section is labeled as file, edit, view, search, comms, tools, window, and help. The first letter of each label is underlined. From the left, the second section shows a symbol of file, a symbol of folder, symbol to save the file, symbol to print, symbol of cut, symbol to copy, symbol to paste (faded), symbol of undo (faded), symbol of redo (faded), a rectangle with a drop-down button on the left, three symbols faded, a symbol of file with a green check, another symbol with a green check, a symbol of zoom in, a symbol of zoom out, symbol of a table, positive sign, and an arrowhead pointing right. The third horizontal section is divided into two vertical sections. On the left section, labeled offline, no forces, no edits, and force disabled are enclosed in a rectangle with a symbol to download at the right side. These labels are placed on the left side of an illustration of a ladder diagram and a label driver: A B_D F 1 point 1 is placed below the label no edits. On the right section, a toolbar with an arrow-head pointing left at the top-left side and an arrow-head pointing right at the top-right side. Five symbols and a label enclosed in rectangles are placed relatively on the left side between the arrow-heads. From left to right, the symbols are of X I C, X I O, O T E, O T E with a label L enclosed in it, O T E with a label U enclosed in it, and a label O S R. Seven labels are shown at the bottom. From left to right, the labels are arrow-head facing, arrow-head facing right, user (highlighted), bit, timer or counter, input or output, and compare. Below the toolbar, there are two screens. The left screen is labeled as I C 500 D M 0.R S S and has the symbols to minimize, maximize, and close the file at the right side. From top to bottom, the left screen is labeled as project, (indented to level 1) help, controller, (indented to level 2) controller properties, (indented to level 2) processor status, (indented to level 2) I O configuration, (indented to level 2) channel configuration, (indented to level 1) multipoint monitor, (indented to level 1) program files, (indented to level 2), (indented to level 2) sys 0 -, (indented to level 2) sys 1 -, (indented to level 2) lad 2 - main_laddr, (indented to level 2) lad 3 - P I D_S U B_R T, (indented to level 2) lad 4 - S U B_R T_2, and (indented to level 2) lad 5 - S U B_R T_3. The second screen is labeled as lad 2 – main_laddr. At left, a red shaded rectangle is labeled as 0 0 0 0 is connected to a vertical line, through X I C labeled as B 3 : 0/0, X I C labeled as I : 1/0, and output energize labeled as O : 30. The first X I C is also labeled as control enabled CTRL_ENABLED. The second X I C is also labeled as toggle switch one on I or O sim mod. TOGGLE_S W_1. The output energize is also labeled as first start up check with I or O sim switch START_UP_CHECK. The second screen has labels start up diagnostic check and the rungs in this section perform a diagnostic check at start up are placed above the ladder diagram. At the bottom, the labels from the left are for help, press F 1, 2:0000, APP, READ (faded), and disabled (faded).

Return to Figure 5-43

OceanofPDF.com

Figure 5-44 Extended Description (Chapter 5)

Return to Figure 5-44

From left to right, the first icon is a normally open contact labeled X I C (Examine If Closed), the second is a normally closed contact labeled X I O (Examine If Open), followed by: O T E (Output Energize) represented by a coil, O T L (Output Latch), O T U (Output Unlatch), O S R (One Shot Rising).

Return to Figure 5-44

OceanofPDF.com

Figure 5-45 Extended Description (Chapter 5)

Return to Figure 5-45

A label processor name: (a rectangle) is placed at the top. A screen is shown below the label. The screen has various labels. From top to bottom, labels ok, cancel, and help are placed on the right side of the screen. A block labeled communication settings is placed below the screen. From the left, label driver (placed over a rectangle with a drop-down button), processor node: (placed over a rectangle with a drop-down button and a label octal = 1 decimal), who active (enclosed in a rectangle), and reply timeout: (placed over a rectangle with a drop-down button and a label sec) are placed in the block.

Return to Figure 5-45

OceanofPDF.com

Figure 5-46 Extended Description (Chapter 5)

Return to Figure 5-46

At the top-left side, a label racks is shown above three rectangles with a drop-down button on the right side. A label power supply is shown below the rectangles. At the right side of the racks, a label auto (online) config is shown. Below this, a label comms configuration and a label read I O configuration enclosed in rectangles are placed. Below the rectangles under the label racks and auto (online) config, a screen with labels #, part #, and description are shown. Under them various labels are shown on the screen. Below the screen, labels adv config, help, and hide all cards are shown at the right side of the label auto (online) config, another screen is placed. A label current cards available is placed at the top. A label filter: all (enclosed in a rectangle with drop-down button) is placed on the right side. The screen has two labels, part # and description over the screen. Various labels are given under part # and description on the screen.

Return to Figure 5-46

OceanofPDF.com

Figure 5-47 Extended Description (Chapter 5)

Return to Figure 5-47

From left to right, the first row is labeled as offset, 15, 14, 13, 12, 11, 10, nine, eight, seven, six, five, four, three, two, one, and zero. From top to bottom, the first column is labeled as offset, B 3 : 0/, B 3 : 1/, B 3 : 2/, B 3 : 3/, and B 3 : 4/. The first, second, ninth, and sixteenth columns of the second row are labeled as one. The first, second, fourth, seventh, ninth, and sixteenth columns of the third row are labeled as one. The first, second, fourth, fifth, seventh, ninth, eleventh, thirteenth, fourteenth, and sixteenth columns of the fourth row are labeled as one. The first, second, fifth, seventh, ninth, eleventh, thirteenth, fourteenth, and sixteenth columns of the fifth row are labeled as one. The first, second, ninth, eleventh, fourteenth, and sixteenth columns of the sixth row are labeled as one. The remaining cells are labeled as zero. The labels are given along four rows. From left to right, the labels just below the table are given as, 3/0 and radix: binary (enclosed in a rectangle with a drop-down button). From the left, the labels along the second row are given as symbol: (a rectangle with a drop-down button), and columns: 16 (enclosed in a rectangle with a drop-down button). The second label is faded. From the left, the label along the third row is desc: (a rectangle with a drop-down button). From the left, the labels at the bottom are B3 (enclosed in a rectangle with an arrow pointing upwards placed above the arrow pointing downwards on the left side), properties, usage, and help. The first letter of the last three labels at the bottom is underlined.

Return to Figure 5-47

OceanofPDF.com

Figure 5-48 Extended Description (Chapter 5)

Return to Figure 5-48

From left to right, the first X I O symbol, the first X I C symbol, and the first output energize are connected in series along the first rung. The second X I C symbol is connected across the first X I C symbol. From left to right, the second X I O symbol (highlighted in green), the third X I C symbol, and the second output energize (highlighted in green) are connected in series along the second rung. The fourth X I C symbol (highlighted in green) is connected across the third X I C symbol. The highlighted terminals of the fourth X I C symbol are labeled as highlighted rungs indicate the instruction is true.

Return to Figure 5-48

OceanofPDF.com

Figure 5-49 Extended Description (Chapter 5)

Return to Figure 5-49

The upper part also shows three labels at the bottom. From left to right, the labels are run, rem, and prog. Arrows from the left side, top side, and the right side of the switch points at the labels run, rem, and prog, respectively.

Return to Figure 5-49

OceanofPDF.com

Figure 5-50(a-c) Extended Description (Chapter 15)

Return to Figure 5-50(a-c)

(c) These terminals are labeled as binary output. A bar graph is placed at the right side of the terminals of the A D C. From left to right, labels along the horizontal axis are one, zero, zero, one, and one. The bars at label one are of the same height, whereas for the label zero, the bars are of the same height. The left side of the A D C is connected to a terminal, whose pair is grounded. The pair of terminals is labeled as analog signal input. A line graph is shown at its left side and the curve shows a downwards peak.

Return to Figure 5-50(a-c)

OceanofPDF.com

Figure 5-51 Extended Description (Chapter 5)

Return to Figure 5-51

An arrow labeled as analog input points from the level sensor towards the block. The top side of the block is connected to the top side of the valve. An arrow points along this connection towards the valve and is labeled as analog output. The block is divided into half. The left vertical section is labeled as processor and the right vertical section is labeled as the P L C analog I or O module. A curved arrow points at the right side of the P L C analog I or O module. An arrow labeled level preset point from the lower end of the curved arrow points at a point between the curved arrow.

Return to Figure 5-51

OceanofPDF.com

Figure 5-52 Extended Description (Chapter 5)

Return to Figure 5-52

From left to right, X I C open symbol labeled as A, X I C open symbol labeled as B, and output energize Y, are connected in series across the two vertical lines. Series connection of X I C open symbols labeled as C and D, is parallelly connected across the series connection of X I C open symbols labeled as A and B. X I C open symbol labeled as E is parallelly connected across the X I C open symbol labeled C.

Return to Figure 5-52

OceanofPDF.com

Figure 5-53 Extended Description (Chapter 5)

Return to Figure 5-53

From left to right, X I C open symbol labeled as A and output energize Y, are connected in series across the two vertical lines. Series connection of X I C open symbols labeled as C and D, is parallelly connected across X I C open symbol labeled A. X I C open symbol labeled as B is parallelly connected between the X I C open symbols labeled as C and D.

Return to Figure 5-53

OceanofPDF.com

Figure 5-54 Extended Description (Chapter 5)

Return to Figure 5-54

From left to right, X I C open symbol labeled as A, X I C open symbol labeled as B, and relay coil Y, are connected in series across the two vertical lines. Series connection of X I C open symbols labeled as C and D, is parallelly connected across X I C open symbol labeled B. X I C open symbol labeled as E is parallelly connected across the X I C open symbol A and the series connection of the X I C symbols labeled as C and D.

Return to Figure 5-54

OceanofPDF.com

Figure 5-56 Extended Description (Chapter 5)

Return to Figure 5-56

For the first rung from left to right, N O limit switch L S 1, and solenoid sol A, are connected in series. For the second rung from left to right, N C limit switch L S 1, and solenoid sol B, are connected in series. A vertical dashed line extends from bottom of the N O limit switch up to top of the N C limit switch.

Return to Figure 5-56

OceanofPDF.com

Figure 5-57 Extended Description (Chapter 5)

Return to Figure 5-57

For the first rung from left to right, N O pushbutton PB 1, N O limit switch L S 1, and relay coil C R 1, are connected in series. N O relay contact C R 1 is parallelly connected across the N O pushbutton P B 1. Pilot light P L 1 is parallelly connected across the relay coil C R 1. For the second rung from left to right, N O pressure switch P S 1, N C relay contact C R 1, N O pushbutton S S 1, and solenoid SOL A, are connected in series. Relay coil C R 2 is parallelly connected across the solenoid SOL A. For the third rung from left to right, N O limit switch L S 2, NC limit switch L S 3, N O relay contact C R 2, and solenoid SOL B, are connected in series. For the fourth rung from left to right, N O pushbutton P B 2, N C pushbutton P B 3, N C relay contact C R 2, and solenoid SOL C, are connected in series. Series connection of N C limit switch L S 4 and N O relay contact C R 3, is parallelly connected across N O pushbutton P B 2. Parallel connection of relay coil C R 3 and pilot light P L 2, is parallelly connected across solenoid SOL C.

Return to Figure 5-57

OceanofPDF.com

Figure 6-1 Extended Description (Chapter 6)

Return to Figure 6-1

An output circuit is shown which consists of a current source in series with a load and contact.

An input circuit is shown which consists of a coil in series with a switch and battery.

Return to Figure 6-1

OceanofPDF.com

Figure 6-3 Extended Description (Chapter 6)

Return to Figure 6-3

For de energized coil the normally closed contacts lies in the left hand side and the normally open in the right hand side. Here the coil goes to the armature.

For energized coil the normally closed contacts lies in the right hand side and the normally open in the left hand side. Here the armature goes towards the col.

Return to Figure 6-3

OceanofPDF.com

Figure 6-4 Extended Description (Chapter 6)

Return to Figure 6-4

Three rung are shown connected between two ladders L 1 and L 2. For switch open coil de energized in the first rung an open switch S is connected to a light CR. The second rung shows CR connected to a pilot switch R which is off. The third rung shows an off CR connected to a G which is on.

For switch closed coil energized in the first rung a closed switch S is connected to a light CR. The second rung shows a closed CR connected to a pilot switch R which is on. The third rung shows an on CR connected to a G which is off.

Return to Figure 6-4

OceanofPDF.com

Figure 6-5 Extended Description (Chapter 6)

Return to Figure 6-5

A three pole magnetic contactor is shown which has movable iron armature or plunger, coil, movable contacts, stationary contacts, line side and load side.

Return to Figure 6-5

OceanofPDF.com

Figure 6-6 Extended Description (Chapter 6)

Return to Figure 6-6

A contactor power contracts are shown which are connected to a pump and also to high current wiring. The contracts are then connected to coil terminals which in turn is connected to low current wiring and PLC output module.

Return to Figure 6-6

OceanofPDF.com

Figure 6-8 Extended Description (Chapter 6)

Return to Figure 6-8

Three rung lines L 1, L 2 and L 3 are shown. The three lines are connected to three motors M which is again connected to three overload relays O L which in turn is connected to T 1, T 2 and T 3. The three rungs converge at last in a three phase motor. L 1 and L 2 are also connected to a circuit which consists of stop and start. A motor M is connected across a start. Start is connected to a motor M which in turn is connected to an overload relay O L.

Return to Figure 6-8

OceanofPDF.com

Figure 6-9 Extended Description (Chapter 6)

Return to Figure 6-9

A PLC is shown connected to magnetic starter, motor and push button. A PLC logic program of motor is shown where a ladder L 1 is connected to start, stop and O L which are in parallel, then they are connected to a starter motor M which in turn is connected to L 2.

Return to Figure 6-9

OceanofPDF.com

Figure 6-11 Extended Description (Chapter 6)

Return to Figure 6-11

A three position selector switch is shown where if position 1 of contact A is closed contact B is open. Both contacts of position 2 are open and if contact of contact B of position 3 is closed contact A is open.

Return to Figure 6-11

OceanofPDF.com

Figure 6-13 Extended Description (Chapter 6)

Return to Figure 6-13

An enclosure containing contact mechanism is shown which has operating force, cam and operator. Then normally open and normally closed limit switches for NEMA are shown. Normally and normally closed switches for IEC are also shown.

Return to Figure 6-13

OceanofPDF.com

Figure 6-18 Extended Description (Chapter 6)

Return to Figure 6-18

Wave of a proximity sensor consisting of coil, oscillator, detector and output is shown hitting a metal output. A target is shown which when hit by a sensor induces an on signal on the output of a ladder logic program.

Return to Figure 6-18

OceanofPDF.com

Figure 6-19 Extended Description (Chapter 6)

Return to Figure 6-19

A normally open contact switch of a DC sensor is shown connected with a load in parallel. Then a connection of the sensor is shown. The senor is connected to a load in parallel and in series with a battery.

Return to Figure 6-19

OceanofPDF.com

Figure 6-20 Extended Description (Chapter 6)

Return to Figure 6-20

A normally open contact switch of a DC sensor is shown connected with a load in series. Then a connection of the sensor is shown. The senor is connected to a load in series and the ends are left open.

Return to Figure 6-20

OceanofPDF.com

Figure 6-21 Extended Description (Chapter 6)

Return to Figure 6-21

A range sensing by the proximity sensor is shown with the operating and release point. The hysteresis zone is located between the release point and target.

Return to Figure 6-21

OceanofPDF.com

Figure 6-23 Extended Description (Chapter 6)

Return to Figure 6-23

The electrodes of the sensor is shown entering an electrostatic field. When the sensor waves reaches the metallic or non metallic target the sensor senses the target when oscillator waveforms are generated.

Return to Figure 6-23

OceanofPDF.com

Figure 6-28 Extended Description (Chapter 6)

Return to Figure 6-28

Here modulated light beam from object to be sensed reaches the sensor. The receiver and transmitter ends are shown. The receiver end is connected to a load and also grounded.

Return to Figure 6-28

OceanofPDF.com

Figure 6-31 Extended Description (Chapter 6)

Return to Figure 6-31

First figure: The light from the transmitter end passes the sensor which is channeled to the receiver end sensor through beam.

Second figure: The transmitter end has two channels. The light is transmitted through one of the channel from the transmitter end. It is detected by the sensor and passes through the beam to retroreflective. The reflected signal from the retroreflective is again sensed by the sensor and passes through the channel to receiver.

Return to Figure 6-31

OceanofPDF.com

Figure 6-32 Extended Description (Chapter 6)

Return to Figure 6-32

The scanner/decoder block scans the data from the bar code symbol in an object and is given to the Programmable controller block and the processed data is given to the Diverter block. A dotted arrow line points out from the diverter block.

Return to Figure 6-32

OceanofPDF.com

Figure 6-33 Extended Description (Chapter 6)

Return to Figure 6-33

The detection of transparent bottles and the level of chocolate are shown in the figure. Level detection is illustrated for fluid. A container ranges from 5 to 30 in increments of 5 inches is shown with the fluid level till the mark of 15. The returning echo signal from the top level of the fluid is sensed by the sensor to produce the 4 to 20 milliampere output. The output is illustrated in a graph. The horizontal axis is labeled as output in milliampere ranging from 0 to 20 and the vertical axis is labeled as inches, ranging from 5 to 20. A linear sloping line is plotted to start from the origin.

Return to Figure 6-33

OceanofPDF.com

Figure 6-34 Extended Description (Chapter 6)

Return to Figure 6-34

Figure shows a hopper emits granules into a drum. The drum is connected to a load cell. The load cell is connected to the programmable controller through a PLC sensor input. The programmable controller is displayed as 560 lbs. The output from the programmable controller provides the ON/OFF control to the hopper.

Return to Figure 6-34

OceanofPDF.com

Figure 6-35 Extended Description (Chapter 6)

Return to Figure 6-35

Two leads from the cold junction are connected to two dissimilar metals, metal A and metal B. The other ends of the metals are joined together to form the hot junction. The thermocouple heated to a temperature of 300 degrees Celsius at the hot junction produces 12 point 2 millivolts at the cold junction. One of the metals is chromel (nickel-chromium) which is the positive lead and the other is Alumel (nickel-aluminum) type K thermocouple which is the negative lead.

Return to Figure 6-35

OceanofPDF.com

Figure 6-36 Extended Description (Chapter 6)

Return to Figure 6-36

The parts of the resistance temperature detector are sheath, ceramic support, and R T D element. The encapsulated type R T D is wounded with a platinum wire.

Temperature control system: The R T D is connected between a container and a controller. The container has two pipes labeled as cooling line and heating line. The controller is displayed as 35 degrees Celsius, which is connected to two blocks: Heat and Cool.

Return to Figure 6-36

OceanofPDF.com

Figure 6-37 Extended Description (Chapter 6)

Return to Figure 6-37

The turbine flowmeter consists of a bladed turbine rotor installed in a flow tube. A magnetic pickup sensor is positioned as close to the rotor. Fluid passing through the flow tube causes the rotor to rotate, which generates pulses in the pickup coil. The generated pulses are then transmitted to readout electronics and displayed as gallons per minute.

Return to Figure 6-37

OceanofPDF.com

Figure 6-38 Extended Description (Chapter 6)

Return to Figure 6-38

The tachometer is connected to a controller which displays 1450 revolutions per minute. The feedback output is given to the motor. The load is connected to the motor and tachometer.

Return to Figure 6-38

OceanofPDF.com

Figure 6-39 Extended Description (Chapter 6)

Return to Figure 6-39

The motor has an optical encoder attached at the rotary end that includes an optical sensor, a light source, and an optical disk. The cross-sectional view shows of the encoder has the lines on the brim.

Return to Figure 6-39

OceanofPDF.com

Figure 6-40 Extended Description (Chapter 6)

Return to Figure 6-40

The symbols shown for output control devices are pilot light (P L), control relay (CR1), control relay (CR-1; NO, CR-2; NC), Motor starter coil (M), motor overload relay contact (OL), Alarm, heater (HTR), Solenoid (SOL), solenoid valve (SV), motor (MTR), and horn.

Return to Figure 6-40

OceanofPDF.com

Figure 6-41 Extended Description (Chapter 6)

Return to Figure 6-41

The one end of the coil is connected in series to an A-C voltage and a switch. The other end of the coil is connected to the switch and a plunger is coupled to the coil. The valve body contains an orifice. The solenoid coil symbol is shown.

Return to Figure 6-41

OceanofPDF.com

Figure 6-42 Extended Description (Chapter 6)

Return to Figure 6-42

First figure: The inlet and outlet valve are shown. The coil is de-energized. The valve orifice is opened. The control circuit is shown. An open limit switch is connected in series with a de-energized solenoid coil. Both are connected across L1 and L2.

Second figure: The inlet and outlet valve are shown. The coil is energized. The valve orifice is closed. The control circuit is shown. A closed limit switch is connected in series with a solenoid coil which is energized. Both are connected across L1 and L2.

Return to Figure 6-42

OceanofPDF.com

Figure 6-44 Extended Description (Chapter 6)

Return to Figure 6-44

Open-loop system: The controller with speed setting control is connected to the motor. The output from the motor is obtained through a output shaft which is given to the load.

Closed-loop system: The controller with speed setting control is connected to the motor. The output from the motor is obtained through a output shaft which is given to the load. The output shaft is connected to a tachometer, which provides the feedback signal to the controller.

Return to Figure 6-44

OceanofPDF.com

Figure 6-45 Extended Description (Chapter 6)

Return to Figure 6-45

The controller module is connected to the servo amplifier module. The servo amplifier module is connected to the servo motor, which is connected to the load. The load is connected to the feedback device: Tachometer speed, Encoder position. Two lines flow out from the feedback device, one is speed feedback which is connected to the servo amplifier. another is position feedback connected to the controller.

Return to Figure 6-45

OceanofPDF.com

Figure 6-46 Extended Description (Chapter 6)

Return to Figure 6-46

The hardwired circuit consists of a normally closed stop button in series with a normally open start button. In the wiring diagram L1, L2, and L3 are connected to the terminals T1, T2, and T3 of a three-phase motor using a three-phase magnetic motor starter. The three-phase magnetic motor starter consists of open contact and an overload relay. A motor is connected in L2 between the starter and overload relay. One end of the start button is connected to one end of the motor, another end of the motor is connected to an overload relay in series. On the other end, the relay is connected to L2. The other end of the start button is connected to open contact M, which is connected back to form a loop. Two ends of the button are connected. The one end of the stop button is connected to L1. The seal-in auxiliary contact of the starter is connected in parallel with the start button. It is connected to the closed stop button in series with a motor starter coil. These are connected across L1 and L2. Programmed control circuit: The inputs stop pushbutton, start pushbutton. and an auxiliary contact is connected to L1 and the output motor starter coil is connected to L2. The start pushbutton, stop pushbutton and the motor starter coil is in series connection. The auxiliary contact is connected in parallel to the start pushbutton.

Return to Figure 6-46

OceanofPDF.com

Figure 6-47 Extended Description (Chapter 6)

Return to Figure 6-47

Three rungs are shown connected between two ladders L 1 and L 2. The first rung shows motor 1 stop pushbutton, motor 2 start pushbutton, motor starter coil, and overload relay are in series connection. The auxiliary contact M 1-1is connected in parallel across the motor 1 start pushbutton. The second rung shows motor 2 stop pushbutton, auxiliary contact M 1-2, motor 2 start pushbutton, motor starter coil, and overload relay are in series connection. The auxiliary contact M2-1 is connected in parallel across the motor 3 start pushbutton. The third rung shows motor 3 stop pushbutton, motor 3 start pushbutton, motor starter coil, and overload relay are in series connection. The auxiliary contact M3-1 is connected in parallel across the motor 3 start pushbutton.

Return to Figure 6-47

OceanofPDF.com

Figure 6-48 Extended Description (Chapter 6)

Return to Figure 6-48

The input devices motor 1 start pushbutton, motor 1 stop pushbutton, motor 2 start pushbutton, motor 2 stop pushbutton, motor 3 start pushbutton, motor 3 stop pushbutton are connected to L1. The output devices motor starter coil M 1 and overload relay in series connection, motor starter coil M 2 and overload relay in series connection, motor starter coil M 3 and overload relay in series connection are connected to L2. Three rungs are shown connected between two ladders L 1 and L 2. The first rung shows motor 1 stop pushbutton, motor 2 start pushbutton, motor starter coil M 1 are in series connection. The auxiliary contact M 1 is connected in parallel across the motor 1 start pushbutton. The second rung shows motor 2 stop pushbutton, auxiliary contact M 1, motor 2 start pushbutton, motor starter coil M 2 are in series connection. The auxiliary contact M2 is connected in parallel across the motor 2 start pushbutton. The third rung shows motor 3 stop pushbutton, motor 3 start pushbutton, motor starter coil M 3 are in series connection. The auxiliary contact M3 is connected in parallel across the motor 3 start pushbutton.

Return to Figure 6-48

OceanofPDF.com

Figure 6-49 Extended Description (Chapter 6)

Return to Figure 6-49

L1 is wired to a solenoid-a pushbutton. The solenoid-A pushbutton is cross-wired to solenoid pushbutton-B. The solenoid pushbutton-B is connected across the parallel connection of solenoid A and solenoid B, which is connected to L2.

Return to Figure 6-49

OceanofPDF.com

Figure 6-50 Extended Description (Chapter 6)

Return to Figure 6-50

The input devices such as solenoid A pushbutton, solenoid B pushbutton are connected to L1. The output devices solenoid A, solenoid B are connected to L2. Two rungs are shown connected between two ladders L 1 and L 2. The first rung shows solenoid A pushbutton, solenoid B pushbutton, and solenoid A are connected in series between L1 and L2. The second rung shows solenoid B pushbutton, solenoid A pushbutton, and solenoid B are connected in series between L1 and L2.

Return to Figure 6-50

OceanofPDF.com

Figure 6-52 Extended Description (Chapter 6)

Return to Figure 6-52

Three rungs are connected between L1 and L2. The first rung has an ON pushbutton and latches coil in series connection. The second rung has an OFF pushbutton and Unlatch coil in series connection. The third rung has relay contact and pilot light in series connection.

Return to Figure 6-52

OceanofPDF.com

Figure 6-53 Extended Description (Chapter 6)

Return to Figure 6-53

The table has four columns and two rows. The columns are labeled as Command, Name, Symbol, and description. The rows are labeled as OTL and OUT. Row 1, OTL: Input-output latch, symbol-latch coil L, description- OTL sets the bit to “1: when the rung becomes true and retains its state when the rung loses continuity or a power cycle occurs.” Row 2, OTU: Input-output unlatch, symbol-unlatch coil U, Description- OTU resets the bit to “0” when the rung becomes true and retains it. The latch coil and unlatch coil have the same address XXX.

Return to Figure 6-53

OceanofPDF.com

Figure 6-54 Extended Description (Chapter 6)

Return to Figure 6-54

The input devices such as the ON pushbutton, OFF pushbutton are connected to L1. The output device's output latch/unlatch coil and pilot light is connected to L2. Two rungs are shown connected between two ladders L 1 and L 2. The first rung shows relay contact and output latch 2/5 are connected in series between L1 and L2. The second rung shows relay contact and output unlatch 2/5 are connected in series between L1 and L2. The 5 in O: 2/5 represents the position of the status bit.

Return to Figure 6-54

OceanofPDF.com

Figure 6-55 Extended Description (Chapter 6)

Return to Figure 6-55

The storage tank is connected to a pump motor, high sensor switch, and low sensor switch. It has an ON/OFF button and MAN/AUTO knob. The status indicating light such as pump running, low level, and high level are shown.

Return to Figure 6-55

OceanofPDF.com

Figure 6-56 Extended Description (Chapter 6)

Return to Figure 6-56

The input devices such as the ON/OFF pushbutton, MAN/AUTO knob, low sensor switch, and high sensor switch are connected to L1. The output device's motor, pump running light, low-level light, and high-level light is connected to L2. There are six rungs connected between L1 and L2. The first rung has MAN/AUTO contact, low sensor switch, OFF/ON switch and a motor coil in series connection, and a MAN/AUTO switch, LATCH/Unlatch switch in series are connected across the MAN/AUTO contact, low sensor switch. The second rung has MAN/AUTO contact, a high sensor switch, and a latch coil in series connection. The third rung has MAN/AUTO contact, a low sensor switch in series, and MAN/AUTO switch is connected across to the series connection and a unlatch coil is connected in series. The fourth rung has a moto switch and a motor G in series connection. The fifth rung has a low sensor switch and a motor R in series connection. The sixth rung has a high sensor switch and a motor Y in series connection.

Return to Figure 6-56

OceanofPDF.com

Figure 6-57 Extended Description (Chapter 6)

Return to Figure 6-57

The input device, pushbutton, is connected to L1. The output device, light, is connected to L2. There are four rungs connected between L1 and L2. The first rung has light without contact with address Local:2:O.Data.0, program pushbutton in contact with address Local:1:I.Data.0, and internal relay L in series connection. The second rung has pushbutton without contact with address Local:1:I.Data.0, internal relay in contact, and light with address Local:2O.Data.0 in series connection. The third rung has pushbutton without contact with address Local:1:I.Data.0, internal relay without contact, and light in contact with address Local:2O.Data.0 in series connection. The fourth rung has pushbutton in contact with address Local:1:I.Data.0, light in contact with address Local:2O.Data.0, and internal relay in contact in series connection.

Return to Figure 6-57

OceanofPDF.com

Figure 6-60 Extended Description (Chapter 6)

Return to Figure 6-60

A furnace is connected from the bottom to a controller, with a digital screen on the controller reading 0 0 0 0. A feedback connection from the controller is linked to the upper portion of the furnace, and the controller is additionally connected to the P L C device.

Return to Figure 6-60

OceanofPDF.com

Figure 6-61 Extended Description (Chapter 6)

Return to Figure 6-61

A belt passes over a horizontal base and goes over a pulley mounted on the right, while below the horizontal base, there is a board with start/stop lights. Adjacent to it, a limit switch is mounted on the base. Above the horizontal base, there is a pilot lightboard with red and green lights. On the left, a block labeled package is placed on the belt.

Return to Figure 6-61

OceanofPDF.com

Figure 6-62 Extended Description (Chapter 6)

Return to Figure 6-62

Four rungs connected between L1 and L2. The first rung has start pushbutton 1, stop pushbutton 2, limit switch LS, and control relay CR in series connection. The contact CR-1 is connected across the start pushbutton 1. The second rung has contact CR-2 in closed position and red stop pilot light 1 in series connection. The third rung has contact CR-3 and green pilot light 2 in series connection. The fourth rung has contact CR-4, a motor starter coil M, and an overload relay OL in series connection.

Return to Figure 6-62

OceanofPDF.com

Figure 6-63 Extended Description (Chapter 6)

Return to Figure 6-63

The input devices such as start pushbutton 1, stop pushbutton 2, and limit switch is connected to L1. The output devices motor, stop red pilot light, and run green pilot light are connected to L2. There are three rungs connected between L1 and L2. The first rung has started pushbutton 1 with address Local:3:1.Data.0 with the internal relay in parallel, stop pushbutton 2 with address Local:3:1.Data.1, limit switch with address Local:3:1.Data.2, and an internal relay in series connection. The second rung has an internal relay and stops the pilot light with address Local:4:0.Data.3 in series connection. The third rung has an internal relay and motor with address Local:4:0.Data.1 in series connection.

Return to Figure 6-63

OceanofPDF.com

Figure 6-64 Extended Description (Chapter 6)

Return to Figure 6-64

The figure illustrates an example of an arrangement of series instructions of a rung programmed for optimum scan time. The series instructions are programmed from the most likely to be false (far left) to the least likely to be false (far right).

Return to Figure 6-64

OceanofPDF.com

Figure 6-65 Extended Description (Chapter 6)

Return to Figure 6-65

The parallel path that is most likely to be true is placed on the top of the rung and less likely is connected in the second rung and least likely in the third rung.

Return to Figure 6-65

OceanofPDF.com

Figure 6-66 Extended Description (Chapter 6)

Return to Figure 6-66

Two ends of the coil are cross-connected between A and B. The other side of the winding is connected to X1 and X2. X1 is connected to a fuse connected with a stop pushbutton. The other end of the pushbutton is connected to the parallel connection of the start pushbutton, Jog pushbutton, and contact CR 2, contact, M in series. The contact CR 1 is connected between the start and jog pushbutton. The junction node of the start and jog pushbutton is connected to CR and the junction node of contact CR1 and jog pushbutton is connected to a motor. The contact CR and motor in parallel are connected to an overload relay. The other end of the overload relay is connected to the X2 terminal.

Return to Figure 6-66

OceanofPDF.com

Figure 6-67 Extended Description (Chapter 6)

Return to Figure 6-67

The input devices such as the jog pushbutton start pushbutton and stop pushbutton is connected to L1. The output devices motor, overload relay are connected to L2. There are two rungs connected between L1 and L2. The first rung has a start switch with address I:3/0, stop switch with address I:3/1, and International relay with address B3:1/0, a switch with address B3:1/0 is connected across the start switch. The second rung has a jog switch with address I:3/2, stop switch with address I:3/1, and motor with address O:2/2, a switch with address B3:1/0 is connected across the jog switch.

Return to Figure 6-67

OceanofPDF.com

Figure 6-68 Extended Description (Chapter 6)

Return to Figure 6-68

The machine has a drill motor pointing to the part sensor. Two switches pushbutton 1 and pushbutton 2 are placed at the top.

Return to Figure 6-68

OceanofPDF.com

Figure 6-69 Extended Description (Chapter 6)

Return to Figure 6-69

The input devices such as pushbutton 1 addressed as I:3/4, pushbutton 2 addressed as I:3/5, and limit switch addressed as I:3/6 are connected to L1. The output device motor contactor addressed O:4/0 is connected to L2. The rung between L1 and L2 is the series connection of pushbutton 1 addressed as I:3/4, pushbutton 2 addressed as I:3/5, and limit switch addressed as I:3/6, and motor contactor addressed O:4/0.

Return to Figure 6-69

OceanofPDF.com

Figure 6-70 Extended Description (Chapter 6)

Return to Figure 6-70

The Input devices shown in unactuated condition are up limit switch addressed I:3/0, down limit switch addressed I:3/1, door up pushbutton addressedI:3/4, a door down pushbutton addressedI:3/5, and stop door pushbutton addressedI:3/7 are connected to L1. The output devices door ajar red light addressed O:4/0, door open green light addressed O:4/1, door closed yellow light addressed O:4/2, motor contactor, UP addressed O:4/3, and motor contactor, UP addressed O:4/4 are connected to L2. There are five rungs connected between L1 and L2. The first rung has the series connection of stop switch addressed as I:3/7, up switch addressed as I:3/4, down switch I:3/5, down interlock, addressed as O:4/4, Up limit switch addressed as I:3/0, and motor up switch addressed as O:4/3. The motor down addressed as O:4/4 is connected across up and down interlock switch. The second rung has the series connection of stop switch addressed as I:3/7, down switch I:3/5, up switch addressed as I:3/4, up interlock addressed as O:4/3, down limit switch addressed as I:3/1, and motor down switch addressed as O:4/4. The motor up addressed as O:4/3 is connected across up and down interlock switch. The third rung has the series connection of up limit switch addressed as I:3/0, down limit switch addressed as I:3/1, and door ajar addressed as O:4/0. The fourth rung has the series connection of up limit switch addressed as I:3/0, and door open addressed as O:4/1. The fifth rung has the series connection of down limit switch addressed as I:3/1, and the door closed addressed as O:4/2.

Return to Figure 6-70

OceanofPDF.com

Figure 6-71 Extended Description (Chapter 6)

Return to Figure 6-71

A box is moving on the conveyor attached to a motor. The start/stop switch and photoswitch are shown below the conveyor. A hopper controlled by a solenoid valve is shown right above the box. A level switch is shown above the box. The pilot light Run, Standby, and Full are shown in the top right end of the figure.

Return to Figure 6-71

OceanofPDF.com

Figure 6-72 Extended Description (Chapter 6)

Return to Figure 6-72

The input devices stop pushbutton, start pushbutton, photo limit switch, level limit switch are connected in parallel to L1. The output devices motor, solenoid, pilot light run, standby, and full are connected to L2. There are five rungs between L1 and L2. The first rung has the series connection of stop switch, start switch, and a motor at run state. The run switch is connected across the start switch as feedback. The second rung has the series connection of closed run switch and standby motor. The third rung has the series connection of level switch, photoswitch, and a full motor. The full switch is connected across the level switch. The fourth rung has a series connection of photoswitch, run switch, and a motor. A full switch is connected across the photoswitch. The fifth rung has the series connection of run switch, level switch, full switch, photoswitch, and a solenoid.

Return to Figure 6-72

OceanofPDF.com

Figure 6-73 Extended Description (Chapter 6)

Return to Figure 6-73

The smart instrumentation contains a microprocessor. The pressure input is given to the pressure sensor module. The pressure sensor gives the analog input to the analog-to-digital converter. It converts the analog to digital data and gives it to the microprocessor which processes the digital data and gives it to the digital to analog converter. Two terminals emerge from the digital to analog converter. The analog output of 4 to 20 milliampere is obtained across the terminals. An arrow moves inwards in the first terminal and outwards in the second terminal.

Return to Figure 6-73

OceanofPDF.com

Figure 6-74 Extended Description (Chapter 6)

Return to Figure 6-74

The input devices pushbutton addressed as I/1, pressure switch I/2, and limit switch I/3 is connected to L1. The output devices pilot light 1 addressed as O/9 and pilot light 2 addressed as O/10 are connected to L2. Three rungs are connected between L1 and L2. The first rung has the series connection of I/1, I/2, and O/9, L. The second rung has the series connection of I/3 and O/9, U. The third rung has the series connection of O/9 and)/10.

Return to Figure 6-74

OceanofPDF.com

Figure 6-75 Extended Description (Chapter 6)

Return to Figure 6-75

The series connection of stop pushbutton, run pushbutton, motor and overload relay is connected between L1 and L2. A series connection of jog pushbutton and contact relay, M is connected across the run pushbutton.

Return to Figure 6-75

OceanofPDF.com

Figure 6-76 Extended Description (Chapter 6)

Return to Figure 6-76

There are four rungs connected between L1 and L2. The first rung has start pushbutton 1, stop pushbutton 2, and a parallel connection of CR 1 and pilot light 1. The second rung has a series connection of pressure switch, contact relay CR 2, and a parallel connection of solenoid 1 and CR 2. The third rung has a series connection of open limit switch, contact relay CR 2-1, and a solenoid 2. The fourth rung has a series connection of closed limit switch, push button ss1, contact relay CR 2-2, and a parallel connection of solenoid 3 and pilot light 2. The limit switch in the third and fourth rung is connected by a dotted line.

Return to Figure 6-76

OceanofPDF.com

Figure 6-77 Extended Description (Chapter 6)

Return to Figure 6-77

The series connection of stop pushbutton 1, reverse pushbutton 2, forward pushbutton 3, relay R-3, Forward motor, and overload relay is connected between L1 and L2. A relay F-2 is connected across forwarding pushbutton 3. The series connection of stop pushbutton, open pushbutton, relay F-1, the reverse motor is wired between one end of stop pushbutton 1 and one end of overload relay. Pilot light 1 is connected across the forward motor and pilot light 2 is connected across the reverse motor.

Return to Figure 6-77

OceanofPDF.com

A Toolbar Window Graphic Extended Description (Chapter 7)

Return to A Toolbar Window Graphic

The timer commands are as T O N, T O F, R T O, C T U, C T D, R E S and H S C. The other quantities shown in the window are user, bit, timer or counter, input or output and compare. Timer or counter tab is selected.

Return to A Toolbar Window Graphic

OceanofPDF.com

Figure 7-2 Extended Description (Chapter 7)

Return to Figure 7-2

An on delay symbols chart is shown where the first column represents Normally open, timed closed contact (NOTC). Contact is open when relay coil is de-energized and when relay is energized, there is a time delay in closing. The second column represents Normally closed, timed open contact (NCTO). Contact is closed when relay coil is de-energized and when relay is energized, there is a time delay in opening.

An off delay symbols chart is shown where the first column represents symbols Normally open, timed open contact (NOTO). Contact is normally open when relay coil is de-energized. When relay coil is energized, contact closes instantly. When relay coil is de-energized, there is a time delay before the contact opens. The second column represents Normally closed, timed closed contact (NCTC). Contact is normally closed when relay coil is de-energized. When relay coil is energized, contact opens instantly. When relay coil is de-energized, there is a time delay before the contact closes.

Return to Figure 7-2

OceanofPDF.com

Figure 7-3 Extended Description (Chapter 7)

Return to Figure 7-3

A relay ladder schematic is shown. Two ladder rails L 1 and L 2 are shown. In the first rung a switch S 1 is connected to a time delay coil T D. In the second rung an open time delay contact T D 1 is connected to a light L 1.

A timing diagram is shown. A square wave is shown for input S 1 whose time delay is 10 second. Then a square wave form to output L 1 is shown where the light is off during the time delay of 10 seconds and then the light is on.

Return to Figure 7-3

OceanofPDF.com

Figure 7-4 Extended Description (Chapter 7)

Return to Figure 7-4

A relay ladder schematic is shown. Two ladder rails L 1 and L 2 are shown. In the first rung a switch S 1 is connected to a time delay coil T D of 10 seconds. In the second rung a closed time delay contact T D 1 is connected to a light L 1.

A timing diagram is shown. A square wave is shown for input S 1 whose time delay is 10 second. Then a square wave form to output L 1 is shown where the light is on during the time delay of 10 seconds and then the light is off.

Return to Figure 7-4

OceanofPDF.com

Figure 7-5 Extended Description (Chapter 7)

Return to Figure 7-5

A relay ladder schematic is shown. Two ladder rails L 1 and L 2 are shown. In the first rung a switch S 1 is connected to a time delay coil T D of 10 seconds. In the second rung time delay contact T D 1 is connected to a light L 1.

A timing diagram is shown. A square wave is shown for input S 1 whose time delay is 10 second. The wave starts and then enters the negative half cycle after delay of 10 seconds. Then a square wave form to output L 1 is shown where the light is on even during the time delay of 10 seconds.

Return to Figure 7-5

OceanofPDF.com

Figure 7-6 Extended Description (Chapter 7)

Return to Figure 7-6

A relay ladder schematic is shown. Two ladder rails L 1 and L 2 are shown. In the first rung a switch S 1 is connected to a time delay coil T D of 10 seconds. In the second rung time delay contact T D 1 is connected to a light L 1.

A timing diagram is shown. A square wave is shown for input S 1 whose time delay is 10 second. The wave starts and then enters the negative half cycle after delay of 10 seconds. Then a square wave form to output L 1 is shown where the light is off even when the square wave in the input has positive pulse and also during the time delay.

Return to Figure 7-6

OceanofPDF.com

Figure 7-7 Extended Description (Chapter 7)

Return to Figure 7-7

The timer commands are as T ON, T OF, R TO, C TU, C TD, R ES and H SC. The other quantities shown in the window are user, bit, timer or counter, input or output and compare.

Return to Figure 7-7

OceanofPDF.com

Figure 7-8 Extended Description (Chapter 7)

Return to Figure 7-8

Two ladders are shown. A control line and a reset line is connected to the input ladder which are in parallel. The lines are then connected to a block with instructions preset time, time base and accumulated time. The block is then connected to an output line.

Return to Figure 7-8

OceanofPDF.com

Figure 7-9 Extended Description (Chapter 7)

Return to Figure 7-9

An input and timer are connected to two ladders. The rung condition is shown as false when a square wave pulse is negative and true when a square wave pulse is positive. The timed period is shown as a square wave whose pulse rises during the true rung condition only and this is also the on delay timed duration. The times output bit is shown the condition is false and value is off with logic 0 during the time delay duration and becomes true and on with logic 1 after the time delay duration. Here preset value equals accumulated value.

Return to Figure 7-9

OceanofPDF.com

Figure 7-10 Extended Description (Chapter 7)

Return to Figure 7-10

T 4 2 is shown where T is file type 4 is file number and 2 is timer number. Word 0 instruction is shown as EN TT DN. Word 1 instruction is shown as preset value and word 2 instruction is shown as accumulated value.

Return to Figure 7-10

OceanofPDF.com

Figure 7-11 Extended Description (Chapter 7)

Return to Figure 7-11

A block of T ON or timer ON delay is shown which contains the instructions timer T 4 0, time base as 1.0, preset as 15 and accumulated as 0. The block is connected to E N and ON switches.

Return to Figure 7-11

OceanofPDF.com

Figure 7-12 Extended Description (Chapter 7)

Return to Figure 7-12

A windows screen is shown which contains buttons of T ON, T OF, R TO, C TU, C TD, R ES. It also contains options such as favorites, add on, alarms, bit and timer or counter.

A block is shown which consists of a rung connected with block of T ON or timer ON delay which contains the instructions timer T 4 0, time base as 1.0, preset as 15 and accumulated as 0. The block is connected to E N and ON switches.

A windows screen shows a table with column heads name, data type, style and description. The data under name are as pump motor, motor delay PRE, motor delay ACC, motor delay EN, motor delay TT and motor delay DN. The data under data type are as timer, DINT, DINT, BOOL, BOOL and BOOL. The first data under style is blank and then all the data are shown as decimal. The data under description is delay before starting motor. After that all the rows are blank.

Return to Figure 7-12

OceanofPDF.com

Figure 7-13 Extended Description (Chapter 7)

Return to Figure 7-13

A ladder L 1 is connected to input switch and then input A. Input A is connected to a block of T ON timer ON delay which contains the instructions timer T 4 0, time base as 1.0, preset as 15 and accumulated as 0. The block is connected to E N and ON switches. T 4 is connected as E N to output B, as T T to output C and as D N to output D. Output B is connected to G, output C to R and output D to Y. G, R and Y are connected to a ladder L 2.

Return to Figure 7-13

OceanofPDF.com

Figure 7-14 Extended Description (Chapter 7)

Return to Figure 7-14

For input condition A first the timer is off then becomes on indicated by a small square pulse then the pulse again decrease as the time is off, after that the timer is again on indicated by a longer square wave. For timer enable bit first the timer is off then becomes on indicated by a small square pulse then the pulse again decrease as the time is off, after that the timer is again on indicated by a longer square wave. For timer timing bit first the timer is off then becomes on indicated by a small square pulse of delay 4 seconds then the pulse again decrease as the time is off, after that the timer is again on indicated by a longer square wave of delay 10 seconds. For timer done bit the timer if off for a long time indicated by no rise in pulse until after the delay. For timer accumulated value The pulse rises slowly for 4 seconds delay and then decreases and the again rises slowly during the 10 seconds delay.

Return to Figure 7-14

OceanofPDF.com

Figure 7-15 Extended Description (Chapter 7)

Return to Figure 7-15

A table has six columns and seven rows. The column head represents address, E N, T T, D N, P R E and A C C. The data under address are T 4 0, T 4 1, T 4 2, T 4 3, T 4 4, T 4 5. The data under all other columns are 0.

Return to Figure 7-15

OceanofPDF.com

Figure 7-16 Extended Description (Chapter 7)

Return to Figure 7-16

A hardwired relay circuit is shown. A ladder L 1 is connected to a stop switch, then a start switch is connected. A time delay T D 1 is connected across the start. Start is then connected to a time delay T D in series. T D is then connected to a ladder L 2. A time delay T D 2 is connected to a motor M is series and they both are connected across T D.

A ladder logic program is shown. A ladder L 1 is connected to input stop and start which are in parallel. An internal relay is connected across the start which is again connected to an internal relay switch. The second rung shows an internal relay which is in parallel with a block of timer containing instructions P R is to 5 and T B is to 1 second. This line is then connected to a motor switch M which is connected to ladder L 2.

Return to Figure 7-16

OceanofPDF.com

Figure 7-17 Extended Description (Chapter 7)

Return to Figure 7-17

Figure shows a hardwired relay circuit where a start up push button P B 1 connected to reset push button P B 2. P B 2 is connected to a coil button C R. Another coil C R 1 is connected across P B 1. A second rung is also connected which shows coil C R 2 with a time delay T D. A third rung is shown where closed time delay T D 1 is connected to a coil C R 2 which in turn is connected to a horn.

Figure shows a ladder logic program where P B 1 is connected to P B 2 which in turn is connected to a T ON block. The T ON block consists of timer as T 4 0, time base 1 0, preset as 10 and accumulated as 0. This block is again connected to E N and D N. T 4 0 is also connected across P B 1 and the second rung consists of T 4 as D N connected to T 4 as E N which in turn is connected to the horn.

Return to Figure 7-17

OceanofPDF.com

Figure 7-18 Extended Description (Chapter 7)

Return to Figure 7-18

Figure shows a hardwired relay circuit where a stop push button P B 1 is connected to a start push button P B 2 which in turn is connected to a lube oil pump motor M which again is connected to an overload O L. M 1 1 is connected across P B 2. The second rung shows a lube oil pressure switch P S 1 connected to main drive motor M 2 which in turn is connected to an O L. A time delay T D is connected across M 2. The third rung shows a time delay T D 1 of 15 seconds connected to a feed motor M 3 which in turn is connected an overload O L.

Figure shows a ladder logic program where a stop push button P B 1 is connected to a start push button P B 2 which in turn is connected to a lube oil pump motor M which again is connected to an overload O L. M 1 1 is connected across P B 2. The second rung shows a lube oil pressure switch P S 1 connected to main drive motor M 2 which in turn is connected to an O L. A time delay T D is connected across M 2. The T ON block consists of timer as T 4 0, time base 1 0, preset as 10 and accumulated as 0. This block is again connected to E N and D N. The third rung shows a time delay T D 1 of 15 seconds connected to a feed motor M 3 which in turn is connected an overload O L.

Return to Figure 7-18

OceanofPDF.com

Figure 7-19 Extended Description (Chapter 7)

Return to Figure 7-19

An open switch S 1 is connected to a rung which consists of an input I 1 over 10 which is again connected to a timer OFF delay where timer is shown as T 4 3, time base as 1.0, preset as 15 and accumulated as 0. This delay is also connected to E N and D N which is again connected to a pilot light P L. The second rung consists of timer T 4 3 over D N which is connected to another timer 0 2 over 1.

Three square waves are shown where the first wave is for S 1 input enable bit E N, the second wave is for timed period timing bit (T T) and the third wave is for timed output done bit (D N).

Return to Figure 7-19

OceanofPDF.com

Figure 7-20 Extended Description (Chapter 7)

Return to Figure 7-20

An open input switch S W is shown which is connected to three off timer delays which are in parallel. The first off timer delay consists of timer as T 4 3, preset as 5 and accumulated as 0. The second off timer delay consists of timer as T 4 2, preset as 10 and accumulated as 0. The third off timer delay consists of timer as T 4 3, preset as 15 and accumulated as 0. These timers are connected to E N and D N which are again connected to M 1 , M 2 and M 3. M 1, M 2 and M 3 are again connected to three overloads O L.

Return to Figure 7-20

OceanofPDF.com

Figure 7-21 Extended Description (Chapter 7)

Return to Figure 7-21

First shows five rungs connected between L 1 and L 2. The first rung consists of a limit switch L S 1 connected to a time delay T D of 5 seconds. The second rung consists of a time delay T D 1 connected to a M 1 which in turn is connected to an overload O L. The third rung consists of a time delay T D 2 connected to a M 2 which in turn is connected to an overload O L. The fourth rung consists of a time delay T D 3 connected to a light G. The fifth rung consist of a time delay T D 4 connected to a light R.

Return to Figure 7-21

OceanofPDF.com

Figure 7-22 Extended Description (Chapter 7)

Return to Figure 7-22

A ladder logic program labeled RSLogix 5000 Program is shown. Limit switch L S 1 connected as inputs to L 1. Motors M 1 and M 2 with overload in series, green light and red light as outputs connected to L 2.

Five rungs are connected between L 1 and L 2. The first rung has a limit switch L S 1 with address Local:1:I.Data.2 connected to a timer OFF delay, where timer is as Timer_L S 1, preset as 5000 and accumulated is 0, in series. This delay is also connected to E N and D N. E N is further connected to L 2. The second rung has a closed timer L S 1. D N and a motor M 1 with address Local:2:O.Data.0 connected in series. The third rung has an open timer L S 1. E N and a motor M 2 with address Local:2:O.Data.1 connected in series. The fourth rung has an open timer L S 1. E N and a green light with address Local:2:O.Data.2 connected in series. The fifth rung has an open timer L S 1. E N and a red light with address Local:2:O.Data.2 connected in series.

Return to Figure 7-22

OceanofPDF.com

Figure 7-23 Extended Description (Chapter 7)

Return to Figure 7-23

Two tanks A and B are shown. Three push switches P S 1, PS 2 and P S 3 are connected between the two tanks and also to a pump.

A ladder logic program is shown. Stop, start, PS 1, PS 2 and P S 3 are connected as inputs to L 1. Also three rungs are shown. The first rung shows stop connected to start which is then connected to a time delay T 4 5. This time delay is then connected to PS 1 which is then connected to a pump switch. Pump switch is then connected to a motor M which in turn is connected to an overload. Another pump switch pump is connected across start. the second rung shows pump connected to a timer ON delay where timer is as T 4 6, time base is 1.0, preset us 5 and accumulated is 0. This delay is also connected to E N and D N. The third rung shows a time delay T 4 6 connected to a P S 2 which is connected to a timer OFF delay. The timer off delay consists of a timer as 4 5, time base as 1.0, preset as 14 and accumulated as 0. This delay is also connected to E N and D N. A P S 3 is connected across P S 2.

Return to Figure 7-23

OceanofPDF.com

Figure 7-25 Extended Description (Chapter 7)

Return to Figure 7-25

A reset button is connected to a push button P B 2 which in turn is connected to a timer delay T 4 2. The time button is connected to a push button P B 1 which is connected to a retentive timer on which consists of timer as T 4 2, time base as 1.0, preset as 9 and accumulated as 0. This timer is connected to E N and D N. E N is again connected to a pilot light P L.

Return to Figure 7-25

OceanofPDF.com

Figure 7-26 Extended Description (Chapter 7)

Return to Figure 7-26

The first square wave shows time input P B 1. The second square wave shows E N (enable bit). The third wave is a stepper function denoting accumulated value. The stepper after 3 is a bit longer which denotes accumulated value retained when rung condition goes false. The fourth wave denotes D N (done bit) and is off till stepper 9 in the previous wave and after that the switch is on and so the wave rises. The fifth wave denotes P L output is off till stepper 9 in the third wave and after that the switch is on and so the wave rises. The sixth wave denotes reset input P B 2. Here the wave is on at quite a later stage.

Return to Figure 7-26

OceanofPDF.com

Figure 7-27 Extended Description (Chapter 7)

Return to Figure 7-27

A reset key switch S 1 and a pressure switch as input is connected to L 1. An alarm horn as output is connected to L 2.

The RSLogix 5000 program has 3 rungs connected across L 1 and L 2. The first rung has a reset switch S 1 with address Local:3:I.Data.0 and alarm timer R E S connected in series. The second rung has pressure switch with address Local:3:I.Data.1 connected to a retentive timer ON, where timer is as alarm_timer, preset as 60000 and accumulated is as 0. This delay is also connected to E N and D N. E N is further connected to L 2. The third rung has alarm timer D N and alarm horn with address Local:4:O.Data.0 connected in series.

Return to Figure 7-27

OceanofPDF.com

Figure 7-28 Extended Description (Chapter 7)

Return to Figure 7-28

The first rung shows a switch S W connected to a delay T 4 2 which in turn is connected to a motor starting time delay block. This block consists of timer as T 4 0, time base as 1.0, preset as 10 and accumulated as 0. A block of pump off timer delay is also connected across the first block. pump off timer delay consists of timer as T 4 5, time base as 1.0, preset as 15 and accumulated as 0. This rung is also connected to a pump M 1 and then to a overload O L. The D N of the second block is connected to a motor M 2 which is again connected to an overload O L. The second rung shows a time delay T 4 1 connected to a pump. The third rung shows a time delay T 4 0 connected to a motor. The fourth rung shows a reset connected to a reset T 4 2. The fifth shows a pump connected to a block of pump running time block. This block consists of timer as T 4 2, time base as 1.0, preset as 10800 and accumulated as 0. The sixth rung shows a time delay T 4 2 connected to a pilot light P L.

Return to Figure 7-28

OceanofPDF.com

Figure 7-29 Extended Description (Chapter 7)

Return to Figure 7-29

First a stop switch P B 1 is shown connected to a start P B 2 which is connected in parallel to a M 1 which in turn is connected to an overload O L. A switch M 1 is connected across P B 2 and time delay T D 1 is connected across M 1 and O L. Second rung shows a time delay T D 1 1 of 20 seconds connected in parallel to M 2 which in turn is connected to an over load O L. A time delay T D 2 is connected across M 2. Third rung shows a time delay T D 2 1 of 20 seconds connected to a switch M 3 which in turn is connected to an overload O L.

Return to Figure 7-29

OceanofPDF.com

Figure 7-30 Extended Description (Chapter 7)

Return to Figure 7-30

Figure shows a stop as P B 1 connected to a start P B 2 which in turn is connected to M 1. M 1 is connected to an overload O L. Another M 1 is connected across P B 2. Second rung shoes M 1 connected to a block of timer ON delay. This block consists of timer as T 4 1, preset as 20 and accumulated as 0. Third rung shows time delay T 4 1 over D N connected to M 2which in turn is connected to an overload O L. Fourth rung shows time delay T 4 1 over D N connected to a block of time delay. This block consists of timer as T 4 2, preset as 20 and accumulated as 0. Fifth rung shows time delay T 4 2 over D N connected to M 3 which in turn is connected to an overload O L.

Return to Figure 7-30

OceanofPDF.com

Figure 7-31 Extended Description (Chapter 7)

Return to Figure 7-31

Temperature, pressure and limit switches as input are connected to L 1. Temperature, pressure, limit switches connected in series with green, red, and yellow lights, respectively, are further connected to L 2.

The RSLogix 5000 program for annunciator flasher has 5 rungs connected across L 1 and L 2. The first rung has a closed timer T 2. D N connected to timer ON delay, where timer is as timer_T 1, preset as 1000 and accumulated is as 0. This delay is also connected to E N and D N. E N is further connected to L 2. The second rung has an open timer T 1. D N connected to timer ON delay, where timer is as timer_T 2, preset as 1000 and accumulated is as 0. This delay is also connected to E N and D N. E N is further connected to L 2. The third rung has an open temperature switch with address Local:1:I.Data.0, an open timer T 1. D N and temperature switch P L with address Local:2:O.Data.0 connected in series. The fourth rung has an open pressure switch with address Local:1:I.Data.1, an open timer T 1. D N and pressure switch P L with address Local:2:O.Data.1 connected in series. The fifth rung has an open limit switch with address Local:1:I.Data.2, an open timer T 1. D N and pressure limit P L with address Local:2:O.Data.2 connected in series.

Return to Figure 7-31

OceanofPDF.com

Figure 7-32 Extended Description (Chapter 7)

Return to Figure 7-32

First rung shows a switch S W connected to a timer on delay block. This block consists of timer as T 4 1, time base as 1.0, preset as 30000 and accumulated as 0. Second rung shows a time delay T 4 1 as D N connected to a timer on delay block. This block consists of timer as T 4 2, time base as 1.0, preset as 12000 and accumulated as 0. Third rung shows T 4 2 as D N connected to a pilot light P L.

Return to Figure 7-32

OceanofPDF.com

Figure 7-33 Extended Description (Chapter 7)

Return to Figure 7-33

First rung shows a time delay T 4 2 as D N connected to a red time block. This block consists of timer as T 4 0, time base as 1.0, preset as 30 and accumulated as 0. Second rung shows a time delay T 4 0 as D N connected to a green time block. This block consists of timer as T 4 1, time base as 1.0, preset as 25 and accumulated as 0. Third rung shows a time delay T 4 1 as D N connected to an amber time block. This block consists of timer as T 4 2, time base as 1.0, preset as 5 and accumulated as 0. Fourth rung shows T 4 0 as E N connected to T 4 0 as D N which in turn is connected to a switch red. Fifth rung shows T 41 as E N connected to T 4 1 ad D N which in turn is connected to switch green. Sixth rung shows T 4 2 as E N connected to T 4 2 as D N which in turn is connected to switch amber.

Return to Figure 7-33

OceanofPDF.com

Figure 7-34 Extended Description (Chapter 7)

Return to Figure 7-34

When red equals north or south then green equals east or west and amber equals east or west. Here time delay is 25 seconds for green and 5 seconds for amber. When red equals east or west then green equals north or south and amber equalsnorth or south. Here time delay is 25 seconds for green and 5 seconds for amber.

Return to Figure 7-34

OceanofPDF.com

Figure 7-35 Extended Description (Chapter 7)

Return to Figure 7-35

First rung shows a time delay T 4 2 as D N connected to a time delay block. This block consists of timer as T 4 0, time base as 1.0, preset as 30 and accumulated as 0. Second rung shows a time delay T 4 0 as D N connected to a time delay block. This block consists of timer as T 4 1, time base as 1.0, preset as 25 and accumulated as 0. Third rung shows a time delay T 4 1 as D N connected to an time delay block. This block consists of timer as T 4 2, time base as 1.0, preset as 5 and accumulated as 0. Fourth rung shows a time delay T 4 0 as E N connected to a time delay block. This block consists of timer as T 4 3, time base as 1.0, preset as 25 and accumulated as 0. Fifth rung shows T 4 0 as E N connected to T 4 0 as D N which in turn is connected to a switch red in north or south. Sixth rung shows T 4 1 as E N connected to T 4 1 ad D N which in turn is connected to switch green as north or south. Seventh rung shows T 4 2 as E N connected to T 4 2 as D N which in turn is connected to switch amber in north or south. Eighth rung shoes T 4 3 as E N connected to T 4 3 as D N which in turn is connected to a green as east or west. Ninth rung shows T 4 as D N connected to T 4 0 as D N which in turn is connected to amber as east or west. Tenth rung shows T 4 0 as D N connected to red switch as east or west.

Return to Figure 7-35

OceanofPDF.com

Figure 7-36 Extended Description (Chapter 7)

Return to Figure 7-36

In first rung switch S 1 is connected to a time delay T D 1. Another time delay T D 2 is connected across T D 1. In second rung a time delay T D 1 of 5 seconds is connected to a pilot light P L 1. In third rung a time delay T D 2 of 5 seconds is connected to a pilot light P L 2. In fourth rung a time delay T D 2 1 of 5 seconds is connected to a pilot light P L 3. In fourth rung a time delay T D 2 2 of 5 seconds is connected to a pilot light P L 4.

Return to Figure 7-36

OceanofPDF.com

Figure 7-37 Extended Description (Chapter 7)

Return to Figure 7-37

A hand and auto switch is shown. A stop button P B 1 is connected to the hand. P B 1 is also connected in parallel to a start button P B 2. A switch M 1 is connected across P B 2 which is again connected in parallel to a switch M. M is also connected to an overload O L. Auto is also connected to a push button P S 1. A time delay T D 1 of 60 seconds is connected with a time delay T D 2, These two are connected in parallel to PS 1. A time delay T D is also connected across M and O L.

Return to Figure 7-37

OceanofPDF.com

Figure 7-38 Extended Description (Chapter 7)

Return to Figure 7-38

An input switch L S 1 is connected to timer on delay block in the first rung. This block consists of a timer as T 4 0, time base as 1, preset as 10 and accumulated as 0. Second rung shows a time delay T 4 0 as D N connected to a SOL A. Third rung shows a time delay T 4 O as D N connected to SOL B. Fourth rung shows a time delay T 4 0 as E N connected to a light R. Fifth rung shows a time delay T 4 0 as E N connected to a light Y.

Return to Figure 7-38

OceanofPDF.com

Figure 7-39 Extended Description (Chapter 7)

Return to Figure 7-39

An input switch L S 1 is connected to timer on delay block in the first rung. This block consists of a timer as T 4 0, time base as 1, preset as 10 and accumulated as 0. Second rung shows a time delay T 4 0 as D N connected to a SOL A. Third rung shows a time delay T 4 O as D N connected to SOL B. Fourth rung shows a time delay T 4 0 as E N connected to a light R. Fifth rung shows a time delay T 4 0 as E N connected to a light Y.

Return to Figure 7-39

OceanofPDF.com

Figure 7-40 Extended Description (Chapter 7)

Return to Figure 7-40

First rung shows a push button P B 1 connected to a reset delay T 4 5. Second rung shows push button P B 2 connected to a retentive timer on block. This block consists of a timer as T 4 5, time base as 1.0, preset as 50 and accumulated as 0. Third rung shows time delay T 4 5 as E N connected to a pilot light P L 1. Fourth rung shows time delay T 4 5 as E N connected to a pilot light P L 2. Fifth rung shows time delay T 4 5 as D N connected to a pilot light P L 3. Sixth rung shows time delay T 4 5 as D N connected to a pilot light P L 4.

Return to Figure 7-40

OceanofPDF.com

Figure 7-41 Extended Description (Chapter 7)

Return to Figure 7-41

First rung shows a push button P B 2 connected to a delay reset T 4 1. Another reset T 4 2 is connected across T 4 1. Second rung shows P B 1 connected to a retentive delay block. This block consists of timer as T 4 1, time base as 1.0, preset as 2900 and accumulated as 0. Third rung shows P B 1 connected to a delay T 4 1 as D N. This delay is connected to a retentive timer on block. This block consists of a timer as T 4 2, time base as 1.0, preset as 1780 and accumulated as 0. Fourth rung shows a time delay T 4 2 as D N connected to a pilot light P L.

Return to Figure 7-41

OceanofPDF.com

Figure 7-42 Extended Description (Chapter 7)

Return to Figure 7-42

First rung shows input connected to a timer off delay block. This block consists of timer as T 4 0, time base as 1.0, preset as 10 and accumulated as 0. Second rung shows T 4 0 as E N connected to a switch. Third rung shows T 4 0 as T T connected to a switch. Fourth rung shows T 4 0 as D N connected to a switch.

Return to Figure 7-42

OceanofPDF.com

Figure 7-43 Extended Description (Chapter 7)

Return to Figure 7-43

First rung shows input connected to a timer off delay block. This block consists of timer as T 4 0, time base as 1.0, preset as 10 and accumulated as 0. Second rung shows T 4 0 as E N connected to a switch. Third rung shows T 4 0 as T T connected to a switch. Fourth rung shows T 4 0 as D N connected to a switch.

Return to Figure 7-43

OceanofPDF.com

Figure 7-44 Extended Description (Chapter 7)

Return to Figure 7-44

The pump consists of a solenoid SOL A through which water flows. A motor rotates the pump. A full sensor switch is shown along with an empty sensor switch. The water goes out through solenoid SOL B. The start or stop control station is shown.

Return to Figure 7-44

OceanofPDF.com

A Toolbar Window Graphic Extended Description (Chapter 8)

Return to A Toolbar Window Graphic

The timer commands are as T O N, T O F, R T O, C T U, C T D, R E S and H S C. The other quantities shown in the window are user, bit, timer or counter, input or output and compare. Timer or counter tab is selected.

Return to A Toolbar Window Graphic

OceanofPDF.com

Figure 8-1 Extended Description (Chapter 8)

Return to Figure 8-1

The back side of the device is blackened and raised up to a height slightly greater than the device. The digital screen reads zero zero zero zero three one two. The red button is labeled as reset button and the lever is labeled as actuating lever.

Return to Figure 8-1

OceanofPDF.com

Figure 8-2 Extended Description (Chapter 8)

Return to Figure 8-2

Four blocks placed on the conveyor belt in a row move leftward, with a proximity switch positioned over the second block from the left. An upward arrow above this proximity switch points to the P L C chart, which consists of 6 columns and 4 rows. Columns 1 through 5. Column 6 has a blank header. Only row 2 has header labeled P L C. A bent rightward arrow emerging from the bottom of column 5 displays a number: 1, 2, 3, 4.

Return to Figure 8-2

OceanofPDF.com

Figure 8-3 Extended Description (Chapter 8)

Return to Figure 8-3

From left to right, sensor is connected to the second terminal of the input module and N O pushbutton is connected to the sixth terminal of the input module. A ladder diagram is shown at the right side of the input module. A vertical line is placed at the right side of the input module. From right side, the vertical strip is connected to the left side of the rectangular block with three labels, through two horizontal lines. The first N O open symbol is connected along the top horizontal line labeled as count line. The second N O open symbol is connected along the bottom horizontal line labeled as reset line. From top to bottom, labels in the rectangular block are: type of counter, preset value, and accumulated value. The right side of the rectangular block is connected to the second vertical line, through the output energize. The line between the right side of the rectangular block and the left side of the output energize is labeled as output line. The tip of a finger is shown over the pushbutton.

Return to Figure 8-3

OceanofPDF.com

Figure 8-4 Extended Description (Chapter 8)

Return to Figure 8-4

In the left ladder diagram, there is a horizontal line between two vertical lines. From left to right, an XIO symbol labeled as limit switch and a block with a label counter value are connected in series along the horizontal line. An illustration of a limit switch is shown over the XIO open symbol. A label counter-up is placed over the block labeled as counter value. A rectangular wave with four peaks is placed below the ladder diagram. An arrow pointing right side points from the horizontal wave towards a label positive four. An upwards pointing arrow points from positive four towards the block labeled counter value. A label is shown below the rectangular wave. The label is accumulated equals to preset equals to output. A label between the second equals to sign and the label output is a symbol of wave. The wave saturates at a label off, then increases up to a label on at a point and finally saturates.

In the right ladder diagram, there is a horizontal line between two vertical lines. From left to right, an XIO symbol labeled as parts sensor and a block with a label counter value are connected in series along the horizontal line. An illustration of a sensor is shown over the XIO open symbol. A label counter-down is placed over the block labeled as counter value. A rectangular wave with five peaks is placed below the ladder diagram. An arrow pointing right side points from the horizontal wave towards a label negative five. An upwards pointing arrow points from negative five towards the block labeled counter value. A label is shown below the rectangular wave. The label is accumulated equals to preset equals to output. A label between the second equals to sign and the label output is a symbol of wave. The wave saturates at a label off, then increases up to a label on at a point and finally saturates.

Return to Figure 8-4

OceanofPDF.com

Figure 8-5(a-b) Extended Description (Chapter 8)

Return to Figure 8-5(a-b)

In part a, four vertical lines are placed beside each other. From left to right, the first vertical line is labeled as L1 and the fourth vertical line is labeled as L2. From the right side, the line L1 is connected to two horizontal lines, which are labeled as I ratio one slash zero, and I ratio one slash one. An N O pushbutton labeled PB1 (count) is connected along the top line I ratio one slash zero. An N O pushbutton labeled PB2 (reset) is connected along the top line I ratio one slash zero. A label inputs is placed over the horizontal lines connected to L1. From top to bottom, four horizontal lines between the second and the third vertical line are labeled as: rung 1, rung 2, rung 3, and rung 4. From left to right, an N O symbol labeled as I ratio one slash zero at the top and PB1 (count) at the bottom, a count-up counter, and an output energize C U, are connected in series along the rung 1. The right side of the count-up counter is also connected to output energize D N. From top to bottom, the count-up counter have labels as C T U, count-up counter, counter C five ratio one, preset, and accumulated. From left to right, an N O symbol labeled as C five ratio one slash D N at the top and counter done bit at the bottom, and the output energize labeled as O ratio two slash zero at the top, and red P L at the bottom are connected along the rung 2. From left to right, an NC symbol labeled as C five ratio one slash D N at the top and counter done bit at the bottom, and the output energize labeled as O ratio two slash one at the top, and green P L at the bottom are connected along the rung 3. The left and the right ends of the NC symbol labeled as C five ratio one slash D N, and the output energize labeled as O ratio two slash one are highlighted in green. From left to right, an N O symbol labeled as I ratio one slash one at the top and PB2 (reset) at the bottom, and the output energize R E S labeled as C five ratio one are connected along the rung 4. A label ladder logic program is placed over the horizontal lines between the second and the third vertical lines. From left, the line L2 is connected with two horizontal lines labeled as: O ratio two slash zero and O ratio two slash one. A pilot light R labeled as red P L is connected across the top horizontal line labeled as O ratio two slash zero. A pilot light G labeled as green P L is connected across the bottom horizontal line labeled as O ratio two slash one. A label outputs is placed over the horizontal lines connected to L2. In part b, a timing diagram is shown with six label at its left side. From top to bottom, the labels are rung 1 (count), rung 2, rung 3, rung 4 (reset), preset value (7), and accumulated value. For accumulated value and preset value, a stairstep graph is shown with seven steps. Each step is labeled from one through seven, from bottom to top. An arrow from accumulated value is pointed towards the step labeled two. An arrow from preset value (7) is pointed towards the step labeled seven. For rung 4, graph remains at the least value, then near the end shows a rectangular peak, and finally drops down to saturate. For rung 3, graph remains at a point, then at the time at which seventh step starts for preset value, the graph for rung 3 shows a downward rectangular peak. Finally at the time at which graph for rung 4 increases to peak, then the graph for rung 3 increases at the point and saturates. For rung 2, graph remains at the least value, then at the time at which seventh step starts for preset value, the graph for rung 2 shows a rectangular peak. Finally at the time at which graph for rung 4 increases to peak, then the graph for rung 2 drops down to the least point to saturate. For rung 1, a rectangular wave is shown with seven rectangular peaks.

For rung 1, there are two labels across which the graph varies. The label at the bottom is false and the label at the top is true. The seventh rectangular peak starts at the time at which seventh step starts for preset value. Area under the graphs is shaded.

Return to Figure 8-5(a-b)

OceanofPDF.com

Figure 8-6 Extended Description (Chapter 8)

Return to Figure 8-6

In the label C five colon three, C is labeled as file type, five is labeled as file number, and three is labeled as counter number. The symbol of folder is placed at the left side. The table labeled as counter address have four rows and is placed at the bottom-right side. The first row has 18 columns. From left to right, the first row is labeled as: C five colon three, bit, 15, 14, 13, 12, 11, ten, zero nine, zero eight, zero seven, zero six, zero five, zero four, zero three, zero two, zero one, and zero zero. For the second row, last ten columns are merged into one. From left to right, the second row is labeled as: C five colon three-point-zero, word zero, C U, C D, D N, O V, U N, U A, and internal use (not addressable). For the third row, last 16 columns are merged into one. From left to right, the third row is labeled as: C five colon three-point-one, word one, and preset value. For the fourth row, last 16 columns are merged into one. From left to right, the fourth row is labeled as: C five colon three-point-two, word two, and accumulated value.

Return to Figure 8-6

OceanofPDF.com

Figure 8-7 Extended Description (Chapter 8)

Return to Figure 8-7

The first cell of the first row is unlabeled. From left to right, the labels in the first row, except for the first cell are slash C U, slash C D, slash D N, slash O V, slash U N, dot P R E, and dot A C C. From left to right, the labels in the first column, except for the first cell are C five ratio zero, C five ratio one, C five ratio two, C five ratio three (highlighted in blue), C five ratio four, and C five ratio five. A cell in eight column and the fifth row is labeled as 50. All the remaining cells are labeled as zero.

Return to Figure 8-7

OceanofPDF.com

Figure 8-8 Extended Description (Chapter 8)

Return to Figure 8-8

The count-up counter is a rectangular block with five labels, from top to bottom. It has a terminal at left side and two terminals at right side. The count-up counter is highlighted in blue. At the right side, the top terminal is connected to output energize C U and the bottom terminal is connected to output energize DN. From top to bottom, the labels in the rectangular block are: C T U, count-up counter, counter C five ratio zero, preset ten, and accumulated zero. A label C five ratio zero slash C U is shown over the first N O symbol. A label C five ratio zero slash D N is shown over the second N O symbol. A label C five ratio zero slash O V is shown over the third N O symbol. The output energize R E S has a label C five ratio zero at the top and a label the reset instruction resets the counter's accumulated value back to zero at the right side.

Return to Figure 8-8

OceanofPDF.com

Figure 8-9 Extended Description (Chapter 8)

Return to Figure 8-9

From left to right, the buttons are labeled: T O N (Timer On Delay), T O F (Timer Off Delay), R T O (Retentive Timer On), C T U (Count Up), C T D (Count Down), R E S (Reset), H S C (High-Speed Counter).

Return to Figure 8-9

OceanofPDF.com

Figure 8-10 Extended Description (Chapter 8)

Return to Figure 8-10

Four vertical lines are placed beside each other. From left to right, the first vertical line is labeled as L1 and the fourth vertical line is labeled as L2. From the right side, the line L1 is connected to three horizontal lines. An N C pushbutton labeled stop is connected along the first horizontal line. An N O pushbutton labeled start is connected along the second horizontal line. An N O pushbutton labeled reset is connected along the third horizontal line. A label inputs is placed over the horizontal lines connected to L1. From top to bottom, three horizontal lines are placed between the second and the third vertical line. From left to right, an N O symbol labeled as stop (highlighted in green), N O symbol labeled start, N C symbol labeled C five ratio zero slash D N (highlighted in green), and output energize motor are connected in series along the first horizontal line. An N O symbol labeled as motor is parallelly connected across the series connection of the N O symbol labeled start, and N C symbol labeled C five ratio zero slash D N.

From left to right, the second N O symbol labeled as motor, the count-up counter, and an output energize C U, are connected in series along the second horizontal line. The right side of the count-up counter is also connected to output energize D N. From top to bottom, the count-up counter have labels as C T U, count-up counter, counter C five ratio one, preset ten, and accumulated zero. From left to right, an N O symbol labeled as reset, and the output energize R E S labeled as C five ratio zero are connected in series along the third horizontal line. A label ladder logic program is placed over the horizontal lines between the second and the third vertical line. From left, the line L2 is connected with a horizontal line. From left to right, the motor starter coil labeled as motor and N C relay contact labeled O L are connected in series along the horizontal line at the left side of the L2. A label outputs is placed over the horizontal line connected to L2. A vertical rectangle with three circles is placed at the bottom-left of the ladder diagram. From top to bottom, red colored circle labeled as stop, green colored circle labeled as start, and blue colored circle labeled as reset are shown in the vertical rectangle.

Return to Figure 8-10

OceanofPDF.com

Figure 8-11 Extended Description (Chapter 8)

Return to Figure 8-11

Three vertical lines are placed beside each other. From left to right, the first vertical line is labeled as L1. From the right side, the line L1 is connected to two horizontal lines. A proximity switch labeled as PROX-SW is connected along the first horizontal line. An N O pushbutton labeled reset P B is connected along the second horizontal line. A label inputs is placed over the horizontal lines connected to L1. From top to bottom, four horizontal lines are placed between the second and the third vertical line. From left to right, an N O symbol labeled as PROX-SW, the first count-up counter (highlighted in blue), and an output energize C U, are connected in series along the first horizontal line. The right side of the first count-up counter is also connected to output energize D N. From top to bottom, the first count-up counter have labels as C T U, count-up counter, counter C five ratio one, preset ten, and accumulated zero. The series connection of the second count-up counter (highlighted in blue), and the second output energize C U, are parallelly connected across the first count-up counter and the first output energize C U. The right side of the second count-up counter is also connected to the second output energize D N. From top to bottom, the second count-up counter have labels as C T U, count-up counter, counter C five ratio two, preset 32767, and accumulated zero. A label parts is shown over the first count-up counter. From left to right, the second N O symbol labeled as C five ratio one slash D N, and an output energize labeled as close box B three ratio zero slash one, are connected in series along the second horizontal line. An output energize R E S labeled as C five ratio one is parallelly connected across the output energize labeled as close box B three ratio zero slash one. From left to right, an N O symbol labeled as B three ratio zero slash one, the third count-up counter (highlighted in green), and the output energize C U are connected in series along the third horizontal line. The right side of the third count-up counter is also connected to the third output energize D N. From top to bottom, the third count-up counter have labels as C T U, count-up counter, counter C five ratio three, preset 300, and accumulated zero. A label packages is shown over the third count-up counter. From left to right, an N O symbol labeled as reset P B, and the output energize R E S labeled as C five ratio two are connected in series along the fourth horizontal line. An output energize R E S labeled as C five ratio three is parallelly connected across the output energize R E S labeled as C five ratio two. A label ladder logic program is placed over the horizontal lines between the second and the third vertical line. An illustration is placed at the bottom-left side of the ladder diagram. The illustration shows three cans over a conveyor. The can in the extreme right is passing through a box.

Return to Figure 8-11

OceanofPDF.com

Figure 8-12 Extended Description (Chapter 8)

Return to Figure 8-12

The toolbar shows an arrowhead pointing left at the top-left side and an arrowhead pointing right at the top-right side. Three symbols and six labels enclosed in rectangles are placed relatively on the left side between the arrowheads. From left to right, the symbols are, an upper case letter I rotated at ninety degrees, same symbol as the first but includes a rectangle at the bottom of the middle line of the letter I, and same symbol as the second but includes a horizontal line dividing the rectangle into half. From left to right, the labels are T O N, T O F, R T O, C T U, C T D, and R E S. Seven labels are shown at the bottom. From left to right, the labels are arrow-head facing, arrow-head facing right, favorites, add-on, alarms, bit, and timer or counter. The ladder diagram shows a vertical line on the right side. Two horizontal lines are connected to the vertical line from its left side. A transition of a rectangular wave from a label false to a label true is placed at the left side of three dots. The three dots are placed at the left side of the first horizontal line. from left to right, a count-up counter and an output energize C U are connected in series along the first horizontal line. The right side of the count-up counter is also connected to output energize D N. From top to bottom, the count-up counter is labeled as C T U, count-up, package underscore count, preset 12 with an arrow point at 12 from right side, and accum zero with an arrow point at zero from right side. An arrow from a label tag name over the count-up counter points towards the label package underscore count. A label input side of rung is placed at the bottom-left side of the count-up counter. Three dots are placed at the left side of the second horizontal line. The output energize R E S labeled as package underscore count is connected across the second horizontal line. A double headed arrow points at the right side of the count-up counter and at the label package underscore count over R E S. The table at the bottom has six columns and nine columns. From left to right, the first row is labeled as tag name, alias for, base tag, type, style, and description. From top to bottom, the first column from the left is labeled as tag name, package underscore count, package underscore count dot P R E, package underscore count dot A C C, package underscore count dot C U, package underscore count dot C D, package underscore count dot D N, package underscore count dot O V, and package underscore count dot U N. A negative sign enclosed in a square is shown at the left side of the label package underscore count. A positive sign enclosed in a square is shown at the left side of the labels, package underscore count dot P R E, and package underscore count dot A C C. From top to bottom, the fourth column is labeled as type, counter, dint, dint, bool, bool, bool, bool, and bool. The fifth row is labeled as style in the first row and decimal in all the remaining rows. The second row of the sixth column is labeled as 12 can counter.

Return to Figure 8-12

OceanofPDF.com

Figure 8-13 Extended Description (Chapter 8)

Return to Figure 8-13

From the left, a vertical line L1 is connected to a horizontal line. The N O pushbutton labeled input is connected across the horizontal line. On the right side of the horizontal line, a ladder diagram has two horizontal lines between two vertical lines. From left to right, the first N O symbol, internal relay contact (highlighted in green), and the output energize labeled as one-shot output are connected in series along the first horizontal line. From left to right, the second N O symbol, and the output energize labeled as internal relay coil are connected in series along the second horizontal line. Both the N O symbols and the N O pushbutton are also labeled as A. The symbols of the internal relay contact and the internal relay coil are of red color. A double-headed arrow is pointing at the internal relay contact and the internal relay coil.

Return to Figure 8-13

OceanofPDF.com

Figure 8-14 Extended Description (Chapter 8)

Return to Figure 8-14

In part a, the symbol of off-to-on transitional contact is placed on the left side. Two vertical and parallel lines are connected to a terminal from the outer sides. An upward pointing arrow of the same height as that of the vertical lines is placed between the vertical lines. The one-shot pulse is shown at the right side. A pulse remains at off, then switches from off to on, and finally after one scan drops down to saturate at the label off. Another pulse above it, remains at off for the same time as the pulse below. Then, the second pulse increases up to the label on and saturates at this label.

In part b, the symbol of on-to-off transitional contact is placed on the left side. Two vertical and parallel lines are connected to a terminal from the outer sides. A downwards pointing arrow of the same height as that of the vertical lines is placed between the vertical lines. The one-shot pulse is shown at the right side. A pulse remains at off, then switches from off to on, and finally after one scan drops down to saturate at the label off. Another pulse above it, remains at the label on for the same time as the pulse below remains at the label off. Then, the second pulse decreases up to the label off and saturates at this label.

Return to Figure 8-14

OceanofPDF.com

Figure 8-15 Extended Description (Chapter 8)

Return to Figure 8-15

Four vertical lines are placed beside each other. From left to right, the first vertical line is labeled as L1 and the fourth vertical line is labeled as L2. From the right side, the line L1 is connected to four horizontal lines. An N C pushbutton labeled stop is connected along the first horizontal line labeled as I ratio one slash one. An N O pushbutton labeled start is connected along the second horizontal line labeled as I ratio one slash two. An N O proximity switch is connected along the third horizontal line labeled as I ratio one slash three. An N O pushbutton labeled reset is connected along the fourth horizontal line labeled as I ratio one slash four. A label inputs is placed over the horizontal lines connected to L1. From top to bottom, three horizontal lines are placed between the second and the third vertical line. From left to right, an N O symbol labeled as I ratio one slash three, O S R labeled as B three ratio zero slash zero, count-up counter, and output energize C U are connected in series along the first horizontal line. The right side of the count-up counter is also connected to output energize D N. From top to bottom, the count-up counter have labels as C T U, count-up counter, counter C five ratio zero, preset 50, and accumulated zero. From left to right, an N O symbol (highlighted in green) labeled as I ratio one slash one, N O symbol labeled as I ratio one slash two, and an output energize labeled O ratio two slash zero, are connected in series along the second horizontal line. The series connection of an N C symbol labeled as C five ratio zero slash D N and an N O symbol labeled O ratio two slash zero is parallelly connected across the N O symbol labeled as I ratio one slash two. From left to right, an N O symbol labeled as C five ratio zero slash D N, and the output energize R E S labeled as C five ratio zero are connected in series along the third horizontal line. An N O symbol labeled as I ratio one slash four is parallelly connected across the N O symbol labeled as C five ratio zero slash D N. A label ladder logic program is placed over the horizontal lines between the second and the third vertical line. From left, the line L2 is connected with a horizontal line. From left to right, the motor starter coil M1 and N C relay contact labeled O L are connected in series along the horizontal line labeled as O ratio two slash zero, at the left side of the L2. A label outputs is placed over the horizontal line connected to L2. An illustration placed below the ladder diagram shows two cartons filled with four objects are placed on a conveyor.

Return to Figure 8-15

OceanofPDF.com

Figure 8-16 Extended Description (Chapter 8)

Return to Figure 8-16

From the left, a vertical line L1 is connected to a horizontal line. The N O limit switch LS1 is connected across the horizontal line. On the right side of the horizontal line, a ladder diagram has a horizontal line between two vertical lines. From left to right, the first N O symbol LS1, O S R labeled as B three ratio zero slash zero, and the output energize R E S labeled as C five ratio one are connected in series along the horizontal line. A label O S R rung is shown over the horizontal line. An actual limit switch is illustrated below the limit switch LS1. The pulse diagram for O S R instruction is placed below the ladder diagram. The symbol of the output energize R E S labeled as C five ratio one is placed at the left side of a pulse with a rectangular peak for a short time. The width of the peak for the pulse is labeled as one P L C scan. A label LS1 is placed at the left side of the second pulse, which is shown above the first pulse. The second pule also shown a rectangular peak, but for a longer time period. Also the rising edges of both pulses are shown at the same time.

Return to Figure 8-16

OceanofPDF.com

Figure 8-17 Extended Description (Chapter 8)

Return to Figure 8-17

Three vertical lines are placed beside each other. From left to right, the first vertical line is labeled as L1. From the right side, the line L1 is connected to two horizontal lines. An N O pushbutton P B is connected along the first horizontal line labeled as I ratio one slash zero. An N O switch S W is connected along the second horizontal line labeled as I ratio one slash one. A label inputs is placed over the horizontal lines connected to L1. From top to bottom, two horizontal lines are placed between the second and the third vertical line. From left to right, an N O symbol labeled as I ratio one slash zero, an O S R labeled as B three ratio zero slash zero, and the T O D instruction are connected in series along the first horizontal line. From top to bottom, T O D instruction is labeled as T O D, to B C D, source T four ratio one dot A C C, and destination O ratio six. From left to right, an N O symbol labeled as I ratio one slash one, T O N instruction, and output energize E N are connected in series along the second horizontal line. The right side of the T O N is connected to an output energize D N. From top to bottom, T O N instruction is labeled as T O N, timer on delay, timer T four ratio one, time base one-point-zero, preset 1000, and accumulated zero. At the right side of the ladder diagram, a vertical rectangular strip is shown. This strip is labeled as output and has 16 terminals. The terminals are labeled from zero through 15. The left side of the strip is connected to a label O ratio six and the right side of the strip is connected to an L E D display which reads zero zero zero zero.

Return to Figure 8-17

OceanofPDF.com

Figure 8-18 Extended Description (Chapter 8)

Return to Figure 8-18

Four vertical lines are placed beside each other. From left to right, the first vertical line is labeled as L1 and the fourth vertical line is labeled as L2. From the right side, the line L1 is connected to two horizontal lines. An N O float switch F S is connected along the first horizontal line. An N O selector switch SS is connected along the second horizontal line. A label inputs is placed over the horizontal lines connected to L1. Five horizontal lines are placed between the second and the third vertical line. From left, N C symbol labeled as T four ratio six at the top and D N at the bottom, T O N instruction, and an output energize E N are connected in series along the first horizontal line. The right side of the T O N is also connected to output energize D N. From top to bottom, the T O N instruction is labeled as T O N, timer on delay, timer T four ratio five, time base one-point-zero, preset one, and accumulated zero. From left to right, an N O symbol labeled T four ratio five at the top and D N at the bottom, the second T O N instruction, and the second output energize E N are connected in series along the second horizontal line. The right side of the second T O N is also connected to the second output energize D N. From top to bottom, the second T O N instruction is labeled as T O N, timer on delay, timer T four ratio six, time base one-point-zero, preset one, and accumulated zero. From left to right, an N O symbol labeled F S, C T U instruction, and an output energize C U are connected in series along the third horizontal line. The right side of the count-up counter is also connected to the third output energize D N. From top to bottom, the count-up counter instruction is labeled as C T U, count-up counter, counter C five ratio one, preset one, and accumulated zero. From left to right, an N O symbol labeled as C five ratio one at the top and D N at the bottom, the second N O symbol labeled as T four ratio five at the top and D N at the bottom, and an output energize labeled as light are connected in series along the fourth horizontal line. The series connection of an N C symbol labeled as C five ratio one at the top and D N at the bottom and the second N O symbol labeled as F S is parallelly connected across the series connection of the N O symbol labeled as C five ratio one and the second N O symbol labeled as T four ratio five. From left, an N O symbol labeled as S S, and an output energize R E S labeled as C five ratio one are connected in series along the fifth horizontal line. A label ladder logic program is placed over the horizontal lines between the second and the third vertical line. From left, the line L2 is connected with a horizontal line. A symbol of light is connected along the horizontal line. The vertical line L2, the horizontal line and the symbol of light are illustrated using red color. A label output is placed over the horizontal line connected to the line L2. Illustration for float switch, selector switch, and light are shown below their symbols.

Return to Figure 8-18

OceanofPDF.com

Figure 8-19(a-b) Extended Description (Chapter 8)

Return to Figure 8-19(a-b)

In part a, four vertical lines are placed beside each other. From left to right, the first vertical line is labeled as L1 and the fourth vertical line is labeled as L2. From the right side, the line L1 is connected to three horizontal lines. An N O pushbutton PB1 is connected along the first horizontal line. An N O pushbutton PB2 is connected along the second horizontal line. An N O pushbutton PB3 is connected along the third horizontal line. A label inputs is placed over the horizontal lines connected to L1. A horizontal line is placed between the second and the third vertical line. From left, the right side of the parallel connection of three N O relay contacts labeled as PB1, PB2, and PB3, is connected to the left side of a vertical rectangular block. From top to bottom, the block is labeled as U D C, P R: zero zero three, and A C: zero zero zero. The right side of the block is connected to the third vertical line, through an output energize labeled as light. A label ladder logic program is placed over the horizontal lines between the second and the third vertical line. From left, the line L2 is connected with a horizontal line. A symbol of light is connected along the horizontal line. The vertical line L2, the horizontal line and the symbol of light are illustrated using red color. A label output is placed over the horizontal line connected to the line L2. In part b, there are five counting diagrams. Except for the middle diagram, all the other have labels off at the lower side and on at the upper side. For the middle diagram, only a label zero is shown. From bottom, diagram for first label reset shows a rectangular peak close to the end. For the second label output, diagram shows a wider rectangular peak. Diagram for the third label counter accumulated value, shows a step pyramid which begins from zero and have six rising edges and five falling edges. The third rising edge and the second falling edge coincides with the rising edge and the falling edge of the peak of second diagram, respectively. Diagram for the fourth label count down, shows four rectangular peaks, labeled from left to right as one, two, three, and four. The rising edge of the fourth peak coincides with the second falling edge of the third diagram. Diagram for the fifth label count up, shows six rectangular peaks, labeled from left to right as one, two, three, and four. The rising edge of the third peak coincides with the third rising edge of the third diagram.

Return to Figure 8-19(a-b)

OceanofPDF.com

Figure 8-20 Extended Description (Chapter 8)

Return to Figure 8-20

Four vertical lines are placed beside each other. From left to right, the first vertical line is labeled as L1 and the fourth vertical line is labeled as L2. From the right side, the line L1 is connected to three horizontal lines. An N O limit switch labeled as enter switch is connected along the first horizontal line. An N O limit switch labeled as exit switch is connected along the second horizontal line. An N O pushbutton labeled as reset is connected along the third horizontal line. A label inputs is placed over the horizontal line connected to L1. From top to bottom, four horizontal lines are placed between the second and the third vertical line. From left to right, an N O symbol labeled as enter switch, the C T U instruction, and output energize C U are connected in series along the first horizontal line. The right side of the C T U is also connected to output energize D N. From top to bottom, the count-up counter instruction is labeled as C T U, count-up counter, counter C five ratio one, preset 150, and accumulated zero. From left to right, an N O symbol labeled exit switch, the C T D instruction, and an output energize C D, are connected in series along the second horizontal line. The right side of the count-down counter is also connected to the second output energize D N. From top to bottom, the count-down counter is labeled as C T D, count-down counter, counter C five ratio one, preset 150, and accumulated zero. From left to right, an N O symbol labeled as C five ratio one at the top and D N at the bottom, and an output energize labeled lot full light are connected in series along the third horizontal line. From left to right, an N O symbol labeled as reset, and the output energize R E S labeled as C five ratio one are connected in series along the fourth horizontal line. A label ladder logic program is placed over the horizontal lines between the second and the third vertical line. From left, the line L2 is connected with a horizontal line. A symbol of light is connected along the horizontal line. The vertical line L2, the horizontal line and the symbol of light are illustrated using red color. A label output is placed over the horizontal line connected to the line L2. Illustration for the both the limit switches, pushbutton, and the light are shown below their symbols.

Return to Figure 8-20

OceanofPDF.com

Figure 8-21 Extended Description (Chapter 8)

Return to Figure 8-21

Four vertical lines are placed beside each other. From left to right, the first vertical line is labeled as L1 and the fourth vertical line is labeled as L2. From the right side, the line L1 is connected to a horizontal line. An N O pushbutton labeled as I ratio one slash zero is connected along the horizontal line. A label input is placed over the horizontal line connected to L1. From top to bottom, two horizontal lines are placed between the second and the third vertical line. From left to right, an N O symbol labeled I ratio one slash zero, C T D instruction, and an output energize C U are connected in series along the first horizontal line. The right side of the count-down counter is also connected to an output energize D N. From top to bottom, the count-down counter is labeled as C T D, count-down counter, counter C five ratio zero, preset two, and accumulated four. From left to right, an N O symbol labeled as C five ratio zero at the top and D N at the bottom, and the output energize labeled as O ratio two slash zero are connected in series along the second horizontal line. A label ladder logic program is placed over the horizontal lines between the second and the third vertical line. From left, the line L2 is connected with a horizontal line. A pilot light labeled as O ratio two slash zero is connected along the horizontal line. The vertical line L2, the horizontal line and the pilot light are illustrated using red color. A label output is placed over the horizontal line connected to the line L2.

Return to Figure 8-21

OceanofPDF.com

Figure 8-22 Extended Description (Chapter 8)

Return to Figure 8-22

Four vertical lines are placed beside each other. From left to right, the first vertical line is labeled as L1 and the fourth vertical line is labeled as L2. From the right side, the line L1 is connected to three horizontal lines. An N O pushbutton PB1 labeled as input A is connected along the first horizontal line. An N O pushbutton PB2 labeled as input B is connected along the second horizontal line. An N O pushbutton labeled reset at the top and input C at the right side is connected along the third horizontal line. A label inputs is placed over the horizontal lines connected to L1. From top to bottom, six horizontal lines are placed between the second and the third vertical line. From left to right, an N O symbol labeled as input A, count-up counter, and output energize C U are connected in series along the first horizontal line. The right side of the count-up counter is also connected to output energize D N. From top to bottom, the count-up counter is labeled as C T U, count-up counter, counter C five ratio two, preset ten, and accumulated zero. From left to right, an N O symbol labeled as input B, count-down counter, and the second output energize C U are connected in series along the second horizontal line. The right side of the count-down counter is also connected to the second output energize D N. From top to bottom, the count-down counter is labeled as C T D, count-down counter, counter C five ratio two, preset ten, and accumulated zero. From left to right, an N O symbol labeled as C five ratio two at the top and C U at the bottom, and the output energize labeled as output A are connected in series along the third horizontal line. From left to right, an N O symbol labeled as C five ratio two at the top and C D at the bottom, and the output energize labeled as output B are connected in series along the fourth horizontal line. From left to right, an N O symbol labeled as C five ratio two at the top and D N at the bottom, and the output energize labeled as output C are connected in series along the fifth horizontal line. From left to right, an N O symbol labeled as input C, and the output energize R E S labeled as C five ratio two are connected in series along the sixth horizontal line. A label ladder logic program is placed over the horizontal lines between the second and the third vertical line. From the left, the line L2 is connected with three horizontal lines. Relay coil A labeled as output A is connected along the first horizontal line. Relay coil B labeled as output B is connected along the second horizontal line. Relay coil C labeled as output C is connected along the third horizontal line. A label outputs is placed over the horizontal lines connected to L2.

Return to Figure 8-22

OceanofPDF.com

Figure 8-23(a-b) Extended Description (Chapter 8)

Return to Figure 8-23(a-b)

In part a, a conveyor passes through a box labeled material processing system. A photoelectric sensor is placed at both the sides of the material processing system. Three vertical cylinders of light shade are placed just before the box. Four vertical cylinders of dark shade are placed after the box. An arrow labeled in-feed (raw parts) below the three cylinders and points towards the box. An arrow labeled out-feed (finished parts) below the four cylinders and points away from the box. In part b, three vertical lines are placed beside each other. From left to right, the first vertical line is labeled as L1. From the right side, the line L1 is connected to three horizontal lines. An N O proximity switch labeled as in-feed count is connected along the first horizontal line. An N O proximity switch labeled as out-feed count is connected along the second horizontal line. An N O pushbutton labeled as reset to zero is connected along the third horizontal line. A label inputs is placed over the horizontal line connected to L1. From top to bottom, three horizontal lines are placed between the second and the third vertical line. From left to right, an N O symbol labeled as in-feed count, count-up counter instruction, and output energize C U are connected in series along the first horizontal line. The right side of the count-up counter is also connected to an output energize D N. From top to bottom, the count-up counter instruction is labeled as C T U, count-up counter, counter C five ratio one, preset 50, and accumulated zero. From left to right, an N O symbol labeled as out-feed count, count-down counter, and an output energize C D, are connected in series along the second horizontal line. The right side of the count-down counter is also connected to the second output energize D N. From top to bottom, the count-down counter is labeled as C T D, count-down counter, counter C five ratio one, preset 50, and accumulated zero. From left to right, an N O symbol labeled as reset, and the output energize R E S labeled as C five ratio one are connected in series along the third horizontal line. A label ladder logic program is placed over the horizontal lines between the second and the third vertical line. A label accumulated equals to number in process parts is placed over the C T D instruction.

Return to Figure 8-23(a-b)

OceanofPDF.com

Figure 8-24 Extended Description (Chapter 8)

Return to Figure 8-24

Four vertical lines are placed beside each other. From left to right, the first vertical line is labeled as L1 and the fourth vertical line is labeled as L2. From the right side, the line L1 is connected to two horizontal lines. An N O pushbutton PB1 labeled as count button is connected along the first horizontal line. An N O pushbutton PB2 labeled as reset button is connected along the second horizontal line. A label inputs is placed over the horizontal line connected to L1. From top to bottom, three horizontal lines are placed between the second and the third vertical line. From left to right, an N O symbol labeled as PB1, N O symbol labeled as C five ratio slash DN, O S R labeled as B three ratio zero slash zero, the first C T U instruction, and output energize C U are connected in series along the first horizontal line. The right side of the first C T U is also connected to output energize D N. From top to bottom, the count-up counter instruction is labeled as C T U, count-up counter, counter C five ratio one, preset 32000, and accumulated zero. The series connection of the second O S R labeled as B three ratio zero slash zero, the second count-up counter, and the second output energize C U, is parallelly connected across the series connection of N O symbol C five ration zero slash DN, the first O S R, first C T U, and the output energize C U. From left to right, the second N O symbol labeled as C five ratio zero at the top and D N at the bottom, an N O symbol labeled C five ratio one slash D N, and an output energize labeled light are connected in series along the second horizontal line. From left to right, the second N O symbol labeled as PB2, and the output energize R E S labeled as C five ratio zero are connected in series along the third horizontal line. An output energize R E S labeled as C five ratio one is parallelly connected across the output energize R E S labeled as C five ratio zero. A label ladder logic program is placed over the horizontal lines between the second and the third vertical line. From left, the line L2 is connected with a horizontal line. A symbol of light is connected along the horizontal line. The vertical line L2, the horizontal line and the symbol of light are illustrated using red color. A label output is placed over the horizontal line connected to the line L2.

Return to Figure 8-24

OceanofPDF.com

Figure 8-25 Extended Description (Chapter 8)

Return to Figure 8-25

Four vertical lines are placed beside each other. From left to right, the first vertical line is labeled as L1 and the fourth vertical line is labeled as L2. From the right side, the line L1 is connected to two horizontal lines. An N O pushbutton PB1 labeled as count is connected along the first horizontal line. An N O pushbutton PB2 labeled as reset is connected along the second horizontal line. A label inputs is placed over the horizontal line connected to L1. From top to bottom, five horizontal lines are placed between the second and the third vertical line. From left to right, an N O symbol labeled as count, O S R labeled as B three ratio zero slash zero, the first C T U instruction, and output energize C U are connected in series along the first horizontal line. The right side of the first C T U is also connected to output energize D N. From top to bottom, the count-up counter instruction is labeled as C T U, count-up counter, counter C five ratio one, preset 500, and accumulated zero. From left to right, an N O symbol labeled as C five ratio one at the top and D N at the bottom, the second count-up counter, and the second output energize C U are connected in series along the second horizontal line. The right side of the second count-up counter is also connected to the second output energize D N. From top to bottom, the count-up counter instruction is labeled as C T U, count-up counter, counter C five ratio two, preset 500, and accumulated zero. From left to right, the second N O symbol labeled as C five ratio one at the top and DN at the bottom, and the output energize R E S labeled as C five ratio one are connected in series along the third horizontal line. An N O symbol labeled reset is parallelly connected across the second N O symbol labeled as C five ration one at the top and DN at the bottom. From left to right, the second N O symbol labeled as reset, and the output energize R E S labeled as C five ratio two are connected in series along the fourth horizontal line. From left to right, an N O symbol labeled as C five ratio two at the top and D N at the bottom, and the output energize labeled as light are connected in series along the fifth horizontal line. A label ladder logic program is placed over the horizontal lines between the second and the third vertical line. From left, the line L2 is connected with a horizontal line. A symbol of light is connected along the horizontal line. The vertical line L2, the horizontal line and the symbol of light are illustrated using red color. A label output is placed over the horizontal line connected to the line L2.

Return to Figure 8-25

OceanofPDF.com

Figure 8-26 Extended Description (Chapter 8)

Return to Figure 8-26

From top to bottom, nine horizontal lines are placed between two vertical lines. From left to right, R T O instruction, and output energize E N are connected in series along the first horizontal line labeled zero zero zero. The right side of the R T O is also connected to output energize D N. From top to bottom, the R T O instruction is labeled as R T O, retentive timer on, timer T four ratio zero, time base one-point-zero, preset 60, and accumulated zero. From left to right, an N O symbol labeled T 4 ratio zero slash D N, the first C T U instruction, and an output energize C U are connected in series along the second horizontal line labeled zero zero one. The right side of the first C T U is also connected to the second output energize D N. From top to bottom, the count-up counter instruction is labeled as C T U, count-up counter, counter C five ratio zero, preset 60, and accumulated zero. From left to right, an N O symbol labeled as C five ratio zero slash D N, the second count-up counter, and the second output energize C U are connected in series along the third horizontal line labeled zero zero two. The right side of the second count-up counter is also connected to the third output energize D N. From top to bottom, the second count-up counter instruction is labeled as C T U, count-up counter, counter C five ratio one, preset 24, and accumulated zero. From left to right, the second N O symbol labeled as T four ratio zero slash D N, and the output energize R E S labeled as T four ratio zero are connected in series along the fourth horizontal line labeled zero zero three. From left to right, an N O symbol labeled as C five ratio zero slash D N, and the output energize R E S labeled as C five ratio zero are connected in series along the fifth horizontal line labeled zero zero four. From left to right, an N O symbol labeled as C fiver ratio one slash D N, and the output energize R E S labeled as C five ratio one are connected in series along the sixth horizontal line labeled zero zero five. A label ladder logic program is placed over the horizontal line. A label seconds is placed over the R T O instruction. A label minutes is placed over the first C T U instruction. A label hours is placed over the second C T U instruction.

Return to Figure 8-26

OceanofPDF.com

Figure 8-27 Extended Description (Chapter 8)

Return to Figure 8-27

Four vertical lines are placed beside each other. From left to right, the first vertical line is labeled as L1 and the fourth vertical line is labeled as L2. From the right side, the line L1 is connected to two horizontal lines. The N C pressure switch is connected along the first horizontal line. An N O pushbutton labeled as reset is connected along the second horizontal line. A label inputs is placed over the horizontal lines connected to L1. From top to bottom, nine horizontal lines are placed between the second and the third vertical line. From left to right, N O symbol (highlighted in green) labeled as pressure switch, N O symbol labeled as reset, and an output energize labeled as internal B three ratio zero slash zero are connected along the first horizontal line. An N O symbol labeled Internal B three ratio zero slash zero is parallelly connected to the N O symbol start. From left to right, the second N O symbol labeled Internal B three ratio zero slash zero and an output energize labeled as set are connected in series along the second horizontal line. From left to right, the third N O symbol (highlighted in green) labeled Internal B three ratio zero slash zero, and an output energize (highlighted in green) labeled as trip are connected in series along the third horizontal line. From left to right, the fourth N O symbol labeled Internal B three ratio zero slash zero, R T O instruction, and output energize E N are connected in series along the fourth horizontal line. The right side of the R T O is also connected to output energize D N. From top to bottom, the R T O instruction is labeled as R T O, retentive timer on, timer T four ratio zero, time base one-point-zero, preset 60, and accumulated zero. From left to right, an N O symbol labeled T 4 ratio zero slash D N, the first C T U instruction, and an output energize C U are connected in series along the fifth horizontal line. The right side of the first C T U is also connected to the second output energize D N. From top to bottom, the count-up counter instruction is labeled as C T U, count-up counter, counter C five ratio zero, preset 60, and accumulated zero. From left to right, an N O symbol labeled as C five ratio zero slash D N, the second count-up counter, and the second output energize C U are connected in series along the sixth horizontal line. The right side of the second count-up counter is also connected to the third output energize D N. From top to bottom, the second count-up counter instruction is labeled as C T U, count-up counter, counter C five ratio one, preset 24, and accumulated zero. From left to right, the second N O symbol labeled as T four ratio zero slash D N, and the output energize R E S labeled as T four ratio zero are connected in series along the seventh horizontal line. From left to right, an N O symbol labeled as C five ratio zero slash D N, and the output energize R E S labeled as C five ratio zero are connected in series along the eighth horizontal line. From left to right, an N O symbol labeled as C fiver ratio one slash D N, and the output energize R E S labeled as C five ratio one are connected in series along the ninth horizontal line. A label ladder logic program is placed over the horizontal lines between the second and the third vertical line. From left, the line L2 is connected with two horizontal lines. The symbol of light labeled as set is connected along the first horizontal line. The symbol of light labeled as trip is connected along the second horizontal line. The vertical line L2, the horizontal lines, and the symbol of the light are illustrated using red color. The symbol of light labeled trip is filled with red color. A label outputs is placed over the horizontal line connected to the line L2. A label seconds

is placed over the R T O instruction. A label minutes is placed over the first C T U instruction. A label hours is placed over the second C T U instruction.

Return to Figure 8-27

OceanofPDF.com

Figure 8-28 Extended Description (Chapter 8)

Return to Figure 8-28

The rectangular wave is shown at the left side. At the top-right side, structure of the encoder shaft is shown. For encoder shaft, the left side of a horizontal cylinder is connected to another horizontal cylinder with a smaller radius. The cross-section of the second cylinder is shown. Another cylinder with the smallest radius is shown at the left side of the second cylinder. The right side of the third cylinder is also connected to the first cylinder, through the second cylinder. An optical disk is placed in between the second cylinder, such that their centers coincides. A rectangle surrounding the top of the optical disk is labeled as optical sensor. A light source is placed at the bottom of the optical sensor. At the bottom-right side the optical disk is shown. The disk has a thick border with a pattern. A small section of the disk is highlighted and close-up view of the pattern on the disk is shown. The close-up view shows pattern of lines perpendicular to the boundary of the disk.

Return to Figure 8-28

OceanofPDF.com

Figure 8-29 Extended Description (Chapter 8)

Return to Figure 8-29

Rectangular wooden planks are shown over the conveyor. The conveyor moves over three horizontal cylinders. The cylinder closest to the box is connected to a rotatory encoder. A device labeled programmable controller is placed beside the movable cutter. An L-shaped arrow points away from the rotatory encoder towards the device labeled programmable controller. A rectangular wave labeled as pulses is shown beside the L-shaped line. An arrow points from the programmable controller towards the movable cutter.

Return to Figure 8-29

OceanofPDF.com

Figure 8-30(a-b) Extended Description (Chapter 8)

Return to Figure 8-30(a-b)

In part a, three horizontal rectangular blocks are placed over a conveyor. A dashed line is shown between a sprocket at the top-left side and the center of the left end of the conveyor. A magnetic sensor is at the right side of the sprocket. A reflector is placed at the conveyor between the second and the third block. A photoelectric sensor illuminates the reflector. An arrow shows the movement of reflector from left to right. In part b, three vertical lines are placed beside each other. From left to right, the first vertical line is labeled as L 1. From the right side, the line L 1 is connected to three horizontal lines. An N O proximity switch labeled as magnetic sensor with address Local:3:l.Data.0 is connected along the first horizontal line. An N O proximity switch labeled as photo sensor is connected along the second horizontal line. An N O pushbutton labeled as reset is is connected along the third horizontal line. A label inputs is placed over the horizontal line connected to L 1. From top to bottom, two horizontal lines are placed between the second and the third vertical line. From left to right, an N O symbol labeled as magnetic sensor with address Local:3:l.Data.0, N C symbol labeled as photo sensor with address Local:3:l.Data.1, N O symbol labeled as sensor, count-up counter instruction, and output energize C U are connected in series along the first horizontal line. The right side of the count-up counter is also connected to an output energize D N. From top to bottom, the count-up counter instruction is labeled as C T U, count-up counter, counter C five ratio one, preset ten, and accumulated zero. From left to right, an N O symbol labeled as reset with address Local:3:l.Data.2 , and the output energize R E S labeled as C five ratio one are connected in series along the second horizontal line. A label ladder logic program is placed over the horizontal lines between the second and the third vertical line. A label ten counts per foot is placed over the C T U instruction.

Return to Figure 8-30(a-b)

OceanofPDF.com

Figure 8-31(a-b) Extended Description (Chapter 8)

Return to Figure 8-31(a-b)

In part a, conveyor labeled M1 conveyor is placed perpendicular to a conveyor labeled M2 conveyor. Two metal plates are placed beside each other on M1 conveyor. A light source is kept at the left side and sensor is placed at the right side of the M2 conveyor. From bottom to top, stack of two metal plates and stack of four metal plates labeled as complete stack are placed over the M2 conveyor. In part b, four vertical lines are placed beside each other. From left to right, the first vertical line is labeled as L1 and the fourth vertical line is labeled as L2. From the right side, the line L1 is connected to three horizontal lines. An N C pushbutton labeled as stop is connected along the first horizontal line. An N O pushbutton labeled as start is connected along the second horizontal line. An N O proximity switch labeled as photo sensor is connected along the third horizontal line. A label inputs is placed over the horizontal lines connected to L1. From top to bottom, six horizontal lines are placed between the second and the third vertical line. From left to right, an N O symbol labeled as M2, T O N instruction, and output energize E N are connected in series along the first horizontal line. The right side of the T O N is also connected to output energize D N. From top to bottom, the T O N instruction is labeled as T O N, timer on delay, timer T four ratio one, time base one-point-zero, preset five, and accumulated zero. From left to right, an N O symbol (highlighted in green) labeled stop, an N O symbol labeled as start, an N C symbol (highlighted in green) labeled as M2 and an output energize labeled M1 are connected in series along the second horizontal line. The series connection of N O symbol labeled M1 and N C symbol labeled M2 is parallelly connected to the series connection of the N O symbol labeled as start and the N C symbol labeled as M2. An N O symbol labeled as T four ratio one at the top and D N at the bottom is parallelly connected across the series connection of the N O symbol labeled M1 and the N C symbol labeled M2. From left to right, N O symbol labeled as photo sensor, the C T U instruction, and the output energize C U are connected in series along the third horizontal line. The right side of the count-up counter is also connected to the third output energize D N. From top to bottom, the count-up counter instruction is labeled as C T U, count-up counter, counter C five ratio one, preset 15, and accumulated zero. From left to right, an N O symbol labeled as C five ratio one at the top and D N at the bottom, and the output energize labeled as M2 are connected in series along the fourth horizontal line. From left to right, an N O symbol labeled as T four ratio one at the top and D N at the bottom, and the output energize R E S labeled as C five ratio one are connected in series along the fifth horizontal line. An N C symbol labeled as stop is parallelly connected to the N O symbol T four ratio one. From left to right, the second N O symbol labeled as T four ratio one at the top and D N at the bottom, and the output energize R E S labeled as T four ratio one are connected in series along the sixth horizontal line. The second N C symbol labeled as stop is parallelly connected to the second N O symbol T four ratio one. A label ladder logic program is placed over the horizontal lines between the second and the third vertical line. From left, the line L2 is connected with two horizontal lines. The relay coil labeled as M1 is connected along the first horizontal line. The relay coil labeled as M2 is connected along the second horizontal line. The vertical line L2, the horizontal lines, and the relay coils are illustrated using red color. A label outputs

is placed over the horizontal line connected to the line L2. A label M2 run time is placed over the T O N instruction. A label number of plates is placed over the C T U instruction.

Return to Figure 8-31(a-b)

OceanofPDF.com

Figure 8-32 Extended Description (Chapter 8)

Return to Figure 8-32

Four vertical lines are placed beside each other. From left to right, the first vertical line is labeled as L1 and the fourth vertical line is labeled as L2. From the right side, the line L1 is connected to four horizontal lines. An N C pushbutton labeled as stop is connected along the first horizontal line. An N O pushbutton labeled as start is connected along the second horizontal line. An N O relay contact labeled as O L is connected along the third horizontal line. An N O pushbutton labeled as reset P B is connected along the fourth horizontal line. A label inputs is placed over the horizontal lines connected to L1. From top to bottom, seven horizontal lines are placed between the second and the third vertical line. From left to right, N O symbol labeled as start, N O symbol (highlighted in green) labeled as stop, three N C symbols (highlighted in green) labeled as O L, T four ratio zero at the top and E N at the bottom, lock out light, and an output energize labeled as motor are connected along the first horizontal line. An N O symbol labeled motor is parallelly connected to the N O symbol start. From left to right, an N O symbol labeled as O L, N C symbol (highlighted in green) labeled as T four ratio zero at the top and D N at the bottom, T O N instruction, and output energize E N are connected in series along the second horizontal line. The right side of the T O N is also connected to output energize D N. From top to bottom, the T O N instruction is labeled as T O N, timer on delay, timer T four ratio zero, time base one, preset 300, and accumulated zero. An N O symbol labeled as T 4 ratio zero at the top and E N at the bottom is parallelly connected to the N O symbol labeled O L. From left to right, N C symbol (highlighted in green) labeled as T four ratio one at the top and D N at the bottom, the second T O N instruction, and the second output energize E N are connected in series along the third horizontal line. The right side of the second T O N is also connected to the second output energize D N. From top to bottom, the second T O N instruction is labeled as T O N, timer on delay, timer T four ratio one, time base one, preset 3600, and accumulated zero. From left to right, an N O symbol labeled as O L, count-up counter, and output energize C U are connected in series along the fourth horizontal line. The right side of the count-up counter is also connected to the third output energize D N. From top to bottom, the count-up counter instruction is labeled as C T U, count-up counter, counter C five ratio zero, preset six, and accumulated zero. From left to right, an N O symbol labeled as T four ratio one at the top and D N at the bottom, and the output energize R E S labeled as C five ratio zero are connected in series along the fifth horizontal line. An N O symbol labeled reset-PB is parallelly connected to the N O symbol T four ratio one. From left to right, an N O symbol labeled as C five ratio zero at the top and D N at the bottom, and the output energize L labeled as lock-out light are connected in series along the sixth horizontal line. From left to right, the second N O symbol labeled reset-PB, and output energize U labeled as lock-out light are connected in series along the seventh horizontal line. A label ladder logic program is placed over the horizontal lines between the second and the third vertical line. From left, the line L2 is connected with two horizontal lines. The relay coil labeled as motor is connected along the first horizontal line. The symbol of light labeled as lock-out light is connected along the second horizontal line. The vertical line L2, the horizontal lines, the symbol of the relay coil and the light are illustrated using red color. A label outputs is placed

over the horizontal line connected to the line L2. A label five minute is placed over the first T O N instruction. A label one hours is placed over the second T O N instruction.

Return to Figure 8-32

OceanofPDF.com

Figure 8-33 Extended Description (Chapter 8)

Return to Figure 8-33

Three vertical lines are placed beside each other. From left to right, the first vertical line is labeled as L1. From the right side, the line L1 is connected to two horizontal lines. An N O start switch labeled as start SW is connected along the first horizontal line. An N O proximity switch labeled as sensor is connected along the second horizontal line. A label inputs is placed over the horizontal line connected to L1. From top to bottom, three horizontal lines are placed between the second and the third vertical line. From left to right, an N O symbol labeled as start S W, T O N instruction, and output energize E N are connected in series along the first horizontal line. The right side of the T O N is also connected to output energize D N. From top to bottom, the T O N instruction is labeled as T O N, timer on delay, timer T four ratio zero, time base one-point-zero, preset 60, and accumulated zero. From left to right, an N O symbol labeled as start SW, N C symbol (highlighted in green) labeled as T four ratio one at the top and D N at the bottom, N O symbol labeled as sensor, count-up counter instruction, and output energize C U are connected in series along the second horizontal line. The right side of the count-up counter is also connected to the second output energize D N. From top to bottom, the count-up counter instruction is labeled as C T U, count-up counter, counter C five ratio one, preset zero, and accumulated zero. From left to right, an N C symbol (highlighted in green) labeled as start SW, and the output energize R E S labeled as C five ratio one are connected in series along the third horizontal line. A label ladder logic program is placed over the horizontal lines between the second and the third vertical line. A label one minute timer is placed over the T O N instruction. A label total parts is placed over the C T U instruction.

Return to Figure 8-33

OceanofPDF.com

Figure 8-34 Extended Description (Chapter 8)

Return to Figure 8-34

Four vertical lines are placed beside each other. From left to right, the first vertical line is labeled as L1 and the fourth vertical line is labeled as L2. From the right side, the line L1 is connected to a horizontal line. An N O switch S1 labeled as timer input is connected along the horizontal line. A label input is placed over the horizontal line connected to L1. From top to bottom, four horizontal lines are placed between the second and the third vertical line. From left to right, an N O symbol labeled as timer input, N C symbol (highlighted in green) labeled as T four ratio zero at the top and D N at the bottom, T O N instruction, and output energize E N are connected in series along the first horizontal line. The right side of the T O N is also connected to output energize D N. From top to bottom, the T O N instruction is labeled as T O N, timer on delay, timer T four ratio zero, time base one-point-zero, preset 10000, and accumulated zero. From left to right, an N O symbol labeled as T four ratio zero at the top and D N at the bottom, count-up counter, and output energize C U are connected in series along the second horizontal line. The right side of the count-up counter is also connected to the second output energize D N. From top to bottom, the count-up counter instruction is labeled as C T U, count-up counter, counter C five ratio zero, preset 100, and accumulated zero. From left to right, an N C symbol (highlighted in green) labeled as timer input, and the output energize R E S labeled as C five ratio zero are connected in series along the third horizontal line. From left to right, an N O symbol labeled as C five ratio zero at the top and D N at the bottom, and the output energize labeled as light are connected in series along the fourth horizontal line. A label ladder logic program is placed over the horizontal lines between the second and the third vertical line. From left, the line L2 is connected with a horizontal line. A symbol of light is connected along the horizontal line. The vertical line L2, the horizontal line and the symbol of light are illustrated using red color. A label output is placed over the horizontal line connected to the line L2.

Return to Figure 8-34

OceanofPDF.com

Figure 8-35 Extended Description (Chapter 8)

Return to Figure 8-35

From top to bottom, four horizontal lines between two vertical lines are labeled as: rung 1, rung 2, rung 3, and rung 4. From left to right, an N O symbol labeled as I ratio one slash one at the top, a count-up counter, and an output energize C U, are connected in series along the rung 1. The right side of the count-up counter is also connected to output energize DN. From top to bottom, the count-up counter have labels as C T U, count-up counter, counter C five ratio one, preset 50, and accumulated zero. From left to right, an N O symbol labeled as C five ratio one slash D N, and the output energize labeled as O ratio two slash zero at the top are connected in series along the rung 2. From left to right, an NC symbol labeled as C five ratio one slash D N at the top, and the output energize labeled as O ratio two slash one at the top are connected in series along the rung 3. From left to right, an N O symbol labeled as I ratio one slash two at the top, and the output energize R E S labeled as C five ratio one are connected in series along the rung 4. A label ladder logic program is placed over the first horizontal line. The left and the right ends of the NC symbol labeled as C five ratio one slash D N, and the output energize labeled as O ratio two slash one are highlighted in green.

Return to Figure 8-35

OceanofPDF.com

Figure 8-36 Extended Description (Chapter 8)

Return to Figure 8-36

From the left, a vertical line L1 is connected to a horizontal line. The N O pushbutton labeled input is connected across the horizontal line. On the right side of the horizontal line, a ladder diagram has two horizontal lines between two vertical lines. From left to right, the first N O symbol, N C symbol labeled as B three ratio zero slash one (highlighted in green), and the output energize labeled as B three ratio zero slash nine are connected in series along the first horizontal line. From left to right, the second N O symbol, and the output energize labeled as B three ratio zero slash one are connected in series along the second horizontal line.

Return to Figure 8-36

OceanofPDF.com

Figure 8-37 Extended Description (Chapter 8)

Return to Figure 8-37

From top to bottom, four horizontal lines are placed between two vertical lines. From left to right, an N O symbol labeled as I ratio two slash six, count-up counter, and output energize C U are connected in series along the first horizontal line. The right side of the count-up counter is also connected to output energize D N. From top to bottom, the count-up counter is labeled as C T U, count-up counter, counter C five ratio two, preset 25, and accumulated zero. From left to right, an N O symbol labeled as I ratio three slash eight, count-down counter, and the second output energize C U are connected in series along the second horizontal line. The right side of the count-down counter is also connected to the second output energize D N. From top to bottom, the count-down counter is labeled as C T D, count-down counter, counter C five ratio two, preset 25, and accumulated zero. From left to right, an N O symbol labeled as C five ratio two at the top and D N at the bottom, and the output energize labeled as O ratio six slash two are connected in series along the third horizontal line. From left to right, an N O symbol labeled as I ratio four slash one, and the output energize R E S labeled as C five ratio two are connected in series along the fourth horizontal line. A label ladder logic program is shown over the ladder diagram.

Return to Figure 8-37

OceanofPDF.com

Figure 8-38 Extended Description (Chapter 8)

Return to Figure 8-38

A square is placed on the conveyor at the limit switch LS1. The center of a circle at the left side of the conveyor is connected to the center of a circle labeled encoder, through a dashed line. A downwards point arrow labeled as position A is shown over the conveyor in the middle. An arrow labeled as forward, points from the square at LS1 through the right side.

Return to Figure 8-38

OceanofPDF.com

Figure 8-39 Extended Description (Chapter 8)

Return to Figure 8-39

Four vertical lines are placed beside each other. From left to right, the first vertical line is labeled as L1 and the fourth vertical line is labeled as L2. From the right side, the line L1 is connected to three horizontal lines. An N O pushbutton PB1 labeled as input A is connected along the first horizontal line. An N O pushbutton PB2 labeled as input B is connected along the second horizontal line. An N O pushbutton labeled reset at the top and input C at the right side, is connected along the third horizontal line. A label inputs is placed over the horizontal lines connected to L1. From top to bottom, six horizontal lines are placed between the second and the third vertical line. From left to right, an N O symbol labeled as input A, count-up counter, and output energize C U are connected in series along the first horizontal line. The right side of the count-up counter is also connected to output energize D N. From top to bottom, the count-up counter is labeled as C T U, count-up counter, counter C five ratio two, preset ten, and accumulated zero. From left to right, an N O symbol labeled as input B, count-down counter, and the second output energize C U are connected in series along the second horizontal line. The right side of the count-down counter is also connected to the second output energize D N. From top to bottom, the count-down counter is labeled as C T D, count-down counter, counter C five ratio two, preset ten, and accumulated zero. From left to right, an N O symbol labeled as C five ratio two at the top and C U at the bottom, and the output energize labeled as output A are connected in series along the third horizontal line. From left to right, an N O symbol labeled as C five ratio two at the top and C D at the bottom, and the output energize labeled as output B are connected in series along the fourth horizontal line. From left to right, an N O symbol labeled as C five ratio two at the top and D N at the bottom, and the output energize labeled as output C are connected in series along the fifth horizontal line. From left to right, an N O symbol labeled as input C, and the output energize R E S labeled as C five ratio two are connected in series along the sixth horizontal line. A label ladder logic program is placed over the horizontal lines between the second and the third vertical line. From left, the line L2 is connected with three horizontal lines. Relay coil A labeled as output A is connected along the first horizontal line. Relay coil B labeled as output B is connected along the second horizontal line. Relay coil C labeled as output C is connected along the third horizontal line. A label outputs is placed over the horizontal lines connected to L2.

Return to Figure 8-39

OceanofPDF.com

Figure 8-40 Extended Description (Chapter 8)

Return to Figure 8-40

The second conveyor is in the form of right slanting line. A proximity switch is placed close to the left side of the horizontal conveyor. A movable gate is shown at the junction of the horizontal conveyor and the second conveyor. The gate moves between the bottom side of the horizontal conveyor and the bottom side of the second conveyor. The movement of the gate is shown by a curved arrow. A gate solenoid drive is placed at the bottom of the horizontal conveyor. Two cylinders are shown at the left side of the gate and two more cylinders are shown at the right side of the gate. A cylinder is placed on the second conveyor.

Return to Figure 8-40

OceanofPDF.com

Figure 8-41 Extended Description (Chapter 8)

Return to Figure 8-41

An arrow shows the movement of the conveyor from left to right. The conveyor have 16 electronic assembly kits. For electronic assembly kits, a vertical rectangle with a circle in the middle is shown. A through beam sensor is placed between two assembly kits and close to the bottom of the kits.

Return to Figure 8-41

OceanofPDF.com

Figure 8-42 Extended Description (Chapter 8)

Return to Figure 8-42

From left, a horizontal oval labeled reflector is placed between the sixth square and the stacker. Two lines from a point at the bottom of the oval is connected through two small semi-circles, which are attached at relatively right side of a block labeled sensor. The sensor is placed below the horizontal line between the sixth square and the stacker. A label cartons of ceiling tile is shown over the third square. An arrow from this label points right side. Another arrow from left side, points at the first square and the second square.

Return to Figure 8-42

OceanofPDF.com

Ladder Diagram Graphic Extended Description (Chapter 9)

Return to Ladder Diagram Graphic

There are four vertical lines. The first vertical line is labeled L1 and the fourth vertical line is labeled L2. L1 is connected with two horizontal lines from the right side of L1. The N O pushbutton labeled P B is connected along the first horizontal line. The N O switch labeled as switch is connected along the second horizontal line. A label inputs is shown over the first horizontal line connected to the vertical line L1. There are four horizontal lines between the second and the third vertical lines. From top to bottom, the horizontal lines are labeled as one, two, three, and four. A label ladder logic program is shown over the horizontal line labeled one, which is connected between the second and the third vertical line. From left to right, XIC open symbol PB, and output energize labeled as PL1, are connected in series along the horizontal line one. From left to right, XIC open symbol labeled switch and the jump instruction labeled as Q two ratio zero, are connected in series along the horizontal line two. From left to right, the second XIC open symbol PB, and an output energize labeled as PL2, are connected in series along the horizontal line three. From left to right, label instruction labeled as Q two ratio zero, the third XIC open symbol PB, and an output energize labeled as PL3, are connected in series along the horizontal line four. Three horizontal lines are connected to the left side of the line L2. A label outputs is shown over the region between the third and the fourth vertical lines. From top to bottom, pilot lights PL1, PL2, and PL3 are connected along the first, the second, and the third horizontal line, respectively. An illustration of a traffic light is shown below the pilot light PL3.

Return to Ladder Diagram Graphic

OceanofPDF.com

Figure 9-1 Extended Description (Chapter 9)

Return to Figure 9-1

A graphical menu bar from a PLC programming interface. At the top section, there are eight adjacent rectangular buttons with uppercase labels. From left to right, the blocks read: J M P (Jump), L B L (Label), J S R (Jump to Subroutine), R E T (Return), S E R (Set Error), T N D (Temporary End), M C R (Master Control Reset), S U S (Suspend).

Return to Figure 9-1

OceanofPDF.com

Figure 9-2 Extended Description (Chapter 9)

Return to Figure 9-2

Four horizontal lines are connected between two vertical lines. From the left, the first vertical line is labeled as L1 and the second vertical line is labeled as L2. From left to right, NC pushbutton labeled as master stop, NO pushbutton labeled as master start, and relay coil M C R are connected in series along the first horizontal line. An N O relay contact labeled as M C R is parallelly connected across the master start pushbutton. From left to right, N C pushbutton, N O pushbutton, and a relay coil C R are connected in series along the second horizontal line. An N O relay contact labeled as C R4 is parallelly connected across the N O pushbutton. The second and the third N O relay contact labeled M C R are connected along with L1 and L2, respectively between the first and the second horizontal line. From left to right, an N O relay contact labeled as C R1, a relay coil M1, and an NC relay contact labeled as O L are connected in series along the third horizontal line. Another N O pushbutton is parallelly connected across the N O relay contact C R1. From left to right, an NC relay contact labeled as C R2, an N O relay contact labeled as M1, a relay coil M2, and the second NC relay contact labeled as O L are connected in series along the fourth horizontal line.

Return to Figure 9-2

OceanofPDF.com

Figure 9-3 Extended Description (Chapter 9)

Return to Figure 9-3

There are four vertical lines. The first vertical line is labeled L1 and the fourth vertical line is labeled L2. L1 is connected with six horizontal lines from the right side of L1. From top to bottom, N O selector pushbutton labeled on or off, NC pushbutton labeled stop, N O pushbutton labeled start, NC level switch labeled level switch, N O limit switch LS1, and NC limit switch LS2 are connected along the six horizontal lines. A label inputs is shown over the first horizontal line connected to the vertical line L1. There are six horizontal lines between the second and the third vertical lines. A label ladder logic program is shown over the horizontal line labeled one, which is connected between the second and the third vertical line. From left to right, XIC open symbol labeled on or off, and output energize MCR, are connected in series along the first horizontal line. From left to right, XIC open symbol (highlighted in green) labeled as stop, XIC open symbol labeled start, and an output energize labeled as M are connected in series along the second horizontal line. An XIC open symbol labeled as M is parallelly connected across the XIC symbol labeled start. From left to right, XIC open symbol (highlighted in green) labeled as level switch, and an output energize labeled as PL1, are connected in series along the third horizontal line. From left to right, XIC open symbol LS1, and an output energize L labeled as SOL, are connected in series along the fourth horizontal line. From left to right, XIC open symbol (highlighted in green) labeled as LS2, and an output energize U labeled as SOL, are connected in series along the fifth horizontal line. An output energize M C R is connected along the sixth horizontal line. Three horizontal lines are connected to the left side of the line L2. A label outputs is shown over the first horizontal line connected to the vertical line L2. From top to bottom, relay coil labeled as M, pilot light PL1, and the solenoid labeled as SOL are connected along the first, the second, and the third horizontal line, respectively. A double-headed arrow labeled as when M C R is de-energized, all nonretentive outputs de-energize, points at the first and the second horizontal line connected to the vertical line L2. An arrow labeled when M C R is de-energized, all retentive outputs remain in last state, points at the third horizontal line connected to the vertical line L2.

Return to Figure 9-3

OceanofPDF.com

Figure 9-4 Extended Description (Chapter 9)

Return to Figure 9-4

There are four vertical lines. The first vertical line is labeled L1 and the fourth vertical line is labeled L2. L1 is connected with five horizontal lines at the right side of L1. From top to bottom, NC switch labeled input A, NC pushbutton labeled input B, N O pushbutton labeled input C, N O proximity switch labeled input D, and N O proximity switch labeled as input E are connected along the five horizontal lines. A label inputs is shown over the first horizontal line connected to the vertical line L1. There are six horizontal lines between the second and the third vertical line. A label ladder logic program is shown over the horizontal line labeled one, which is connected between the second and the third vertical line. From left to right, XIC open symbol (highlighted in green) labeled input A, and output energize MCR labeled as active, are connected in series along the first horizontal line labeled as start fence. From left to right, XIC open symbol (highlighted in green) labeled as input B, and an output energize (highlighted in green) labeled as output A are connected in series along the second horizontal line. From left to right, XIC open symbol labeled as input C, T O N instruction, and an output energize E N, are connected in series along the third horizontal line. The right side of the T O N is connected to an output energize D N. From top to bottom, T O N instruction is labeled as T O N, timer on delay, timer T four ratio one, time base one-point-zero, preset ten, and accumulated zero. From left to right, XIC open symbol labeled as input D, and an output energize L (highlighted in green) labeled as latch output B, are connected in series along the fourth horizontal line. From left to right, XIC open symbol labeled as input E, and an output energize U (highlighted in green) labeled as unlatch output B, are connected in series along the fifth horizontal line. An output energize M C R is connected along the sixth horizontal line labeled as end fence. Two horizontal lines are connected to the left side of the line L2. A label outputs is shown over the first horizontal line connected to the vertical line L2. From top to bottom, relay coil ON labeled as output A, and another relay coil ON labeled as output B are connected along the first, and the second horizontal line, respectively. The symbols of relay coil ON are shaded.

Return to Figure 9-4

OceanofPDF.com

Figure 9-5 Extended Description (Chapter 9)

Return to Figure 9-5

There are four vertical lines. The first vertical line is labeled L1 and the fourth vertical line is labeled L2. L1 is connected with five horizontal lines at the right side of L1. From top to bottom, N O switch labeled input A, NC pushbutton labeled input B, N O pushbutton labeled input C, N O proximity switch labeled input D, and N O proximity switch labeled as input E are connected along the five horizontal lines. A label inputs is shown over the first horizontal line connected to the vertical line L1. There are six horizontal lines between the second and the third vertical line. A label ladder logic program is shown over the horizontal line labeled one, which is connected between the second and the third vertical line. From left to right, XIC open symbol labeled input A, and output energize MCR, are connected in series along the first horizontal line labeled as start fence. From left to right, XIC open symbol (highlighted in green) labeled as input B, and an output energize labeled as output A are connected in series along the second horizontal line. From left to right, XIC open symbol labeled as input C, T O N instruction, and an output energize E N, are connected in series along the third horizontal line. The right side of the T O N is connected to an output energize D N. From top to bottom, T O N instruction is labeled as T O N, timer on delay, timer T four ratio one, time base one-point-zero, preset ten, and accumulated zero. From left to right, XIC open symbol labeled as input D, and an output energize L (highlighted in green) labeled as latch output B, are connected in series along the fourth horizontal line. From left to right, XIC open symbol labeled as input E, and an output energize U (highlighted in green) labeled as unlatch output B, are connected in series along the fifth horizontal line. An output energize M C R is connected along the sixth horizontal line labeled as end fence. Two horizontal lines are connected to the left side of the line L2. A label outputs is shown over the first horizontal line connected to the vertical line L2. From top to bottom, relay coil OFF labeled as output A, and another relay coil ON labeled as output B are connected along the first, and the second horizontal line, respectively. The symbol of relay coil ON is shaded.

Return to Figure 9-5

OceanofPDF.com

Figure 9-6 Extended Description (Chapter 9)

Return to Figure 9-6

There are four vertical lines. The first vertical line is labeled L1 and the fourth vertical line is labeled L2. L1 is connected with two horizontal lines from the right side of L1. The N O pushbutton labeled P B is connected along the first horizontal line. The N O switch labeled as switch is connected along the second horizontal line. A label inputs is shown over the first horizontal line connected to the vertical line L1. There are four horizontal lines between the second and the third vertical lines. From top to bottom, the horizontal lines are labeled as one, two, three, and four. A label ladder logic program is shown over the horizontal line labeled one, which is connected between the second and the third vertical line. From left to right, XIC open symbol PB, and output energize labeled as PL1, are connected in series along the horizontal line one. From left to right, XIC open symbol labeled switch and the jump instruction labeled as Q two ratio zero, are connected in series along the horizontal line two. From left to right, the second XIC open symbol PB, and an output energize labeled as PL2, are connected in series along the horizontal line three. From left to right, label instruction labeled as Q two ratio zero, the third XIC open symbol PB, and an output energize labeled as PL3, are connected in series along the horizontal line four. Three horizontal lines are connected to the left side of the line L2. A label outputs is shown over the region between the third and the fourth vertical lines. From top to bottom, pilot lights PL1, PL2, and PL3 are connected along the first, the second, and the third horizontal line, respectively. An illustration of a traffic light is shown below the pilot light PL3.

Return to Figure 9-6

OceanofPDF.com

Figure 9-7 Extended Description (Chapter 9)

Return to Figure 9-7

There are four vertical lines. The first vertical line is labeled L1 and the fourth vertical line is labeled L2. L1 is connected with ten horizontal lines at the right side of L1. From top to bottom, N O pushbutton labeled PB1, NC pushbutton labeled PB2, N O pressure switch labeled PS1, NC level switch labeled LLS1, N O limit switch labeled LS1, N O limit switch labeled LS2, NC limit switch labeled LS3, N O pushbutton labeled PB3, NC limit switch labeled LS4, and N O temperature switch labeled TS1 are connected along the ten horizontal lines. A label inputs is shown over the first horizontal line connected to the vertical line L1. From top to bottom, ten horizontal lines between the second and the third vertical line are labeled from one through ten. A label ladder logic program is shown over the horizontal line labeled one, which is connected between the second and the third vertical line. From left to right, XIC open symbol labeled PB1, XIC open symbol (highlighted in green) labeled PB2, and output energize M, are connected in series along the horizontal line one. An XIC open symbol labeled M is parallelly connected across the XIC open symbol labeled PB1. From left to right, the second XIC open symbol labeled M, an XIC open symbol labeled PS1, T O N instruction, and an output energize EN are connected in series along the horizontal line two. The right side of the T O N is connected to an output energize D N. From top to bottom, T O N instruction is labeled as T O N, timer on delay, timer T four ratio six, time base one-point-zero, preset five, and accumulated zero. From left to right, XIC open symbol labeled T four ratio six at the top and D N at the bottom, and an output energize labeled as PL1, are connected in series along the horizontal line three. From left to right, XIC open symbol (highlighted in green) labeled as LLS1, and an output energize J M P (highlighted in blue) labeled as Q two ratio one, are connected in series along the horizontal line four. From left to right, XIC open symbol labeled as LS1, and an output energize labeled as SOL 1, are connected in series along the horizontal line five. From left to right, an XIO open symbol (highlighted in green) labeled as LS1, an XIC open symbol labeled as LS2, and an output energize labeled as SOL 2 are connected along the horizontal line six. From left to right, an XIC open symbol (highlighted in green) labeled as LS3, and an output energize (highlighted in green) labeled as PL2 are connected along the horizontal line seven. From left to right, L B L instruction (highlighted in blue) labeled as Q two ratio one, an XIC open symbol labeled as PB3, the second XIC open symbol labeled as T four ratio six at the top and D N at the bottom, and an output energize labeled as SOL 3 are connected along the horizontal line eight. An XIC open symbol labeled as SOL 3 is parallelly connected across the XIC open symbol labeled as PB3. From left to right, an XIC open symbol labeled as SOL 3, an XIC open symbol (highlighted in green) labeled as LS4, and an output energize labeled as SOL 4 are connected along the horizontal line nine. From left to right, an XIO open symbol (highlighted in green) labeled as TS1, and an output energize (highlighted in green) labeled as heater are connected along the horizontal line ten. Eight horizontal lines are connected to the left side of the line L2. A label outputs is shown over the first horizontal line connected to the vertical line L2. From top to bottom, motor starter coil labeled as M, pilot light PL1 labeled as PL1, solenoid labeled as SOL 1, solenoid labeled as SOL 2, pilot light PL2 labeled as PL2, solenoid labeled as SOL 3, solenoid labeled as SOL 4, and heater labeled as heater are connected along the horizontal lines connected to

the vertical line L2. The symbols of pilot light PL2, and heater are shaded. A double-headed arrow is labeled as timers should be programmed outside the jumped section and points at the horizontal lines with relay coil M, and the pilot light PL1. Another double-headed arrow is labeled as jumped program rungs are not scanned by the processor and input conditions are not examined, and outputs remain in their last state. The second double-headed arrow points below the third horizontal line and below the fifth horizontal line.

Return to Figure 9-7

OceanofPDF.com

Figure 9-8 Extended Description (Chapter 9)

Return to Figure 9-8

There are five horizontal lines connected between two vertical lines. A label ladder logic program is placed over the first horizontal line. From left to right, XIC open symbol labeled input A, and an output energize J M P labeled as Q two ratio five are connected in series along the first horizontal line. From left to right, XIC open symbol labeled input B, and an output energize labeled as output A are connected in series along the second horizontal line. From left to right, XIC open symbol labeled input D, and the second output energize J M P labeled as Q two ratio five are connected in series along the third horizontal line. From left to right, XIC open symbol labeled input E, and an output energize labeled as output C are connected in series along the fourth horizontal line. From left to right, L B L instruction labeled as Q two ratio five, XIC open symbol labeled input F, and an output energize labeled as output D are connected in series along the fifth horizontal line. Two arrows from the J M P instructions point at the L B L instruction.

Return to Figure 9-8

OceanofPDF.com

Figure 9-9 Extended Description (Chapter 9)

Return to Figure 9-9

The ladder diagram at the left side has 17 rungs and a label main program rungs points at the top of the ladder diagram. From left to right, an XIC open symbol (highlighted in green), and the J S R instruction are connected in series along the twelfth rung. The ladder diagram at the right side has ten rungs. The S B R instruction labeled as subroutine area is connected along the first rung and the R E T instruction labeled unconditional return is connected along the tenth rung. An arrow labeled jumps points from the rung with J S R towards the label subroutine area. Another arrow labeled as returns to next instruction after J S R, points from the rung with R E T towards the thirteenth rung of the ladder diagram on the left side.

Return to Figure 9-9

OceanofPDF.com

Figure 9-10 Extended Description (Chapter 9)

Return to Figure 9-10

From top to bottom, J S R is connected along a horizontal line, which is connected to a vertical line on the right side. From top to bottom, J S R instruction is labeled as J S R, jump-to-subroutine, and S B R file number U ratio three. At the middle, S B R is connected along a horizontal line, which is connected to a vertical line on the left side. S B R instruction is labeled as S B R, and subroutine. At the bottom, R E T instruction is connected along a horizontal line, which is connected to a vertical line on the right side. R E T instruction is labeled as R E T, and return.

Return to Figure 9-10

OceanofPDF.com

Figure 9-11(a-b) Extended Description (Chapter 9)

Return to Figure 9-11(a-b)

In part a, a rotator at the left is connected to the rotator of a conveyor at the right side. The conveyor has a weight sensor within it. An inverted container with a solenoid fixed at the left side of its lid. A liquid is poured through the open lid of the container. A pilot light is present on the right side of the container. In part b, there are four vertical lines. The first vertical line is labeled L1 and the fourth vertical line is labeled L2. L1 is connected with four horizontal lines at the right side of L1. From top to bottom, N O selector switch labeled off or on, NC pushbutton, N O pushbutton labeled start, and proximity switch labeled sensor are connected along the four horizontal lines. An illustration of sensor is shown below the symbol labeled sensor. A label inputs is shown over the first horizontal line connected to the vertical line L1. From top to bottom, seven horizontal lines are connected between the second and the third vertical line. A label main program file two is shown over the horizontal line labeled one, which is connected between the second and the third vertical line. From left to right, an XIC open symbol labeled sensor, and J S R instruction are connected in series along the first horizontal line. From top to bottom, J S R instruction is labeled as J S R, jump-to-subroutine, and S B R file number U ratio three. From left to right, an XIO open symbol (highlighted in green) labeled sensor, and an output energize (highlighted in green) labeled as PL1 are connected in series along the second horizontal line. An XIC open symbol labeled as T four ratio one slash E N is parallelly connected across the XIO open symbol labeled sensor. From left to right, an XIC open symbol (highlighted in green) labeled as stop, an XIC open symbol labeled as start, and an output energize labeled as motor are connected in series along the third horizontal line. An XIC open symbol labeled as motor is parallelly connected across the XIC open symbol labeled as start. From left to right, an XIC open symbol labeled off or on, an XIO open symbol (highlighted in green) labeled as sensor, and an output energize labeled SOL are connected in series along the fourth horizontal line. From left to right, S B R instruction, an XIO open symbol (highlighted in green) labeled T four ratio one slash D N, an T O N instruction, and an output energize E N are connected in series along the fifth horizontal line. S B R instruction is labeled as S B R, and subroutine. The right side of the T O N is connected to an output energize D N. From top to bottom, T O N instruction is labeled as T O N, timer on delay, timer T four ratio zero, time base one-point-zero, preset one, and accumulated zero. From left to right, an XIC open symbol labeled T four ratio zero slash D N, an T O N instruction, and an output energize E N are connected in series along the sixth horizontal line. The right side of the T O N is connected to an output energize D N. From top to bottom, T O N instruction is labeled as T O N, timer on delay, timer T four ratio one, time base one-point-zero, preset one, and accumulated zero. R E T instruction is connected along the seventh horizontal line. R E T instruction is labeled as R E T, and return. Three horizontal lines are connected to the left side of the line L2. A label outputs is shown over the first horizontal line connected to the vertical line L2. From top to bottom, motor starter coil M1 labeled as motor, pilot light PL1 labeled as PL1, and solenoid labeled as SOL are connected along the horizontal lines connected to the vertical line L2. The second and the third vertical lines shows discontinuity between the fourth and the fifth horizontal. This gap is labeled as subroutine file three.

Return to Figure 9-11(a-b)

OceanofPDF.com

Figure 9-12 Extended Description (Chapter 9)

Return to Figure 9-12

The ladder diagram labeled main program file two has six rungs. From left to right, an XIC symbol, and an output energize are connected in series along each rung, except for the third rung. The third rung consists of J S R instruction in place of output energize. From top to bottom, J S R instruction is labeled as J S R, jump-to-subroutine, and S B R file number three. The ladder diagram labeled subroutine program file three has seven rungs. From left to right, an XIC symbol, and an output energize are connected in series along each rung, except for the first and the seventh rung. The first rung consists of S B R instruction in place of XIC open symbol. From top to bottom, S B R instruction is labeled as S B R, and subroutine. The seventh rung consists of only R E T instruction. From top to bottom, R E T instruction is labeled as R E T, and return. An arrow from the third rung of the ladder diagram in left points at the S B R connected in the ladder diagram in the right. An arrow from the seventh rung of the ladder diagram in the right points at the XIC open symbol along the fourth rung of the ladder diagram in the left.

Return to Figure 9-12

OceanofPDF.com

Figure 9-13 Extended Description (Chapter 9)

Return to Figure 9-13

The ladder diagram labeled main program file two has six rungs. From left to right, an XIC symbol, and an output energize are connected in series along each rung, except for the third rung. The third rung consists of J S R instruction in place of output energize. From top to bottom, J S R instruction is labeled as J S R, jump-to-subroutine, and S B R file number three.

The ladder diagram labeled level one file three has six rungs. From left to right, an XIC symbol, and an output energize are connected in series along each rung, except for the first, the third and the sixth rung. The first rung consists of S B R instruction in place of XIC open symbol. From top to bottom, S B R instruction is labeled as S B R, and subroutine. The third rung consists of J S R instruction in place of output energize. From top to bottom, J S R instruction is labeled as J S R, jump-to-subroutine, and S B R file number four. The sixth rung consists of only R E T instruction. From top to bottom, R E T instruction is labeled as R E T, and return.

The ladder diagram labeled level two file four has six rungs. It is same as the ladder diagram labeled as level one file three, except for the labels in J S R instruction. In this ladder diagram, from top to bottom, J S R instruction is labeled as J S R, jump-to-subroutine, and S B R file number five.

The ladder diagram labeled level three file five has seven rungs. From left to right, an XIC symbol, and an output energize are connected in series along each rung, except for the first and the seventh rung. The first rung consists of S B R instruction in place of XIC open symbol. From top to bottom, S B R instruction is labeled as S B R, and subroutine. The seventh rung consists of only R E T instruction. From top to bottom, R E T instruction is labeled as R E T, and return. An arrow from the last rung of each ladder diagram points at the XIC open symbol along the third last rung in the ladder diagram placed left to each ladder diagram. Another arrow from the third rung of each ladder diagram points at the S B R along the first rung in the ladder diagram placed right to each ladder diagram.

Return to Figure 9-13

OceanofPDF.com

Figure 9-14 Extended Description (Chapter 9)

Return to Figure 9-14

The rectangle labeled I or O data files has labels in three rows. From left to right, labels along the first row are: one, one, one, one, zero, and zero. From left to right, labels along the second row are: zero, zero, zero, zero, one, and zero. From left to right, labels along the third row are: zero, zero, zero, zero, zero, and one. The ladder diagram labeled program logic scan shows a horizontal line connected between two vertical lines. From left to right, a relay contact, the second relay contact, and an output energize are connected along the first horizontal line. The third relay contact is connected across the first relay contact. An arrow labeled read inputs points away from the right side of the rectangle labeled as I or O data files. An arrow labeled write inputs points towards the right side of the rectangle labeled as I or O data files. A curved arrow from the top side of the arrow label read inputs, points towards the top side of the ladder diagram. A curved arrow from the bottom of the ladder diagram points at the bottom of the arrow labeled write inputs. An arrow labeled immediate output points away from the right side of the ladder diagram. An arrow labeled immediate input points towards the right side of the ladder diagram.

Return to Figure 9-14

OceanofPDF.com

Figure 9-15 Extended Description (Chapter 9)

Return to Figure 9-15

At the top, an illustration shows a toolbar with an arrow-head pointing left at the top-left side and an arrow-head pointing right at the top-right side. Eight labels enclosed in rectangles are placed relatively on the left side between the arrow-heads. From left to right, the labels are I I M (highlighted in blue), I O M, S Y C, M S G, I I E, I I D, R P I, and R E F. From bottom-left to bottom-right, an arrow-head facing left, an arrow-head facing right, labels: input or output, compare, compute or math, and move or logic, are shown. At the bottom, the I I M instruction is connected along a horizontal line, which is connected to a vertical line on the right side. From top to bottom, I I M instruction is labeled as I I M, immediate input w slash mask, slot I ratio one-point-zero, mask zero zero zero F h, and length one.

Return to Figure 9-15

OceanofPDF.com

Figure 9-16 Extended Description (Chapter 9)

Return to Figure 9-16

At the top, an illustration shows a toolbar with an arrow-head pointing left at the top-left side and an arrow-head pointing right at the top-right side. Eight labels enclosed in rectangles are placed relatively on the left side between the arrow-heads. From left to right, the labels are I I M, I O M (highlighted in blue), S Y C, M S G, I I E, I I D, R P I, and R E F. From bottom-left to bottom-right, an arrow-head facing left, an arrow-head facing right, labels: input or output, compare, compute or math, and move or logic, are shown. At the bottom, the I O M instruction is connected along a horizontal line, which is connected to a vertical line on the right side. From top to bottom, I O M instruction is labeled as I O M, immediate output w slash mask, slot O ratio four-point-zero, mask zero F F F h, and length two.

Return to Figure 9-16

OceanofPDF.com

Figure 9-17 Extended Description (Chapter 9)

Return to Figure 9-17

There are three vertical lines. The first vertical line is connected to a horizontal line from the right side of the first vertical line. The proximity switch labeled as label underscore sensor is connected along the first horizontal line. A horizontal line is connected between the second and the third vertical line. From left to right, the XIC open symbol labeled as label underscore sensor, and immediate output instruction, are connected in series along the horizontal line. From top to bottom, the immediate output instruction is labeled as I O T, immediate output, and update tag local ratio three ratio zero. The illustration of the sensor is placed at the bottom-right side of the proximity switch.

Return to Figure 9-17

OceanofPDF.com

Figure 9-18 Extended Description (Chapter 9)

Return to Figure 9-18

At the top, an illustration of a limit switch connected to an input module with eight terminals is shown. An N O limit switch labeled as OFF is placed over a field input device. The right terminal of the switch is connected to the fourth terminal of the input module, through a label I ratio one slash three. An actual limit switch is shown at the bottom-left side of the symbol of the limit switch. A dashed line labeled actual state of input device (zero) ignored, extends from the fourth terminal up to a label one at the fourth row of a table with ten rows. This table is labeled as input image table. The fourth row is divided into eight columns. The fifth column from the left has the label one. An arrow from a label programming terminal forces the state of input I ratio one slash three ON (one) points at the label one on the input image table. Below the input module, a label remote run is enclosed in a rectangle with an arrow-head pointing downwards at the right side of the rectangle. At the right side of this rectangle, two rectangles with a downwards pointing arrow-head at the right side are placed one over the other. The first rectangle is labeled as forces installed and the second rectangle is labeled as forces enabled. Below the input image table, a laptop with a label ladder logic program on its screen is shown. At the bottom, a ladder diagram is shown. There are four vertical lines. From left, the first vertical line is labeled as L1 and the fourth vertical line is labeled as L2. A horizontal line is connected at the right side of the line L1. An N O limit switch labeled I ratio one slash three is connected along the horizontal line. A label input is placed over the horizontal line. Two horizontal lines are connected between the second and the third vertical line. A label ladder logic program is placed over the first horizontal line connected between the second and the third vertical line. From left to right, an XIC open symbol (highlighted in green) labeled as I ratio one slash three at the top and force greater than ON at the bottom, and an output energize (highlighted in green) labeled as O ratio two slash five are connected in series along the first horizontal line. From left to right, an XIC open symbol (highlighted in green) labeled as O ratio two slash five, and an output energize (highlighted in green) labeled as O ratio two slash six are connected in series along the second horizontal line. Two horizontal lines are connected at the left side of the vertical line L2. From top to bottom, solenoid labeled as O ratio two slash five, and motor starter coil labeled O ratio two slash six are connected along the horizontal lines connected to the line L2. A label ON is shown below the solenoid and the motor starter coil. A label outputs is shown over the first horizontal line connected to the line L2.

Return to Figure 9-18

OceanofPDF.com

Figure 9-19 Extended Description (Chapter 9)

Return to Figure 9-19

At the top, an illustration of an output module with eight terminals connected to a solenoid and a motor starter coil is shown. The solenoid and the motor starter coil are placed at the right side of the output module. A label field output devices is shown below them. The sixth terminal of the output module is connected to the solenoid labeled as ON, through a label O ratio two slash five. The seventh terminal of the output module is connected to the motor starter coil labeled as OFF, through a label O ratio two slash six. A table placed at the left side of the output module is labeled as input image table. The table has ten rows, where the fourth row is divided into eight columns. From left, the second and the third column encloses a label zero. A dashed line labeled status of bit O ratio two slash five remains at zero, extends from the label zero in the third column of the table points towards the sixth terminal. Another dashed line extends from the label zero in the second column of the table points towards the seventh terminal. Below the output image table, a label remote run is enclosed in a rectangle with an arrow-head pointing downwards at the right side of the rectangle. At the right side of this rectangle, two rectangles with a downwards pointing arrow-head at the right side are placed one over the other. The first rectangle is labeled as forces installed and the second rectangle is labeled as forces enabled. Below the output module, a laptop with a label ladder logic program on its screen is shown. At the bottom, a ladder diagram is shown. There are four vertical lines. From left, the first vertical line is labeled as L1 and the fourth vertical line is labeled as L2. A horizontal line is connected at the right side of the line L1. An N O limit switch labeled I ratio one slash three is connected along the horizontal line. A label input is placed over the horizontal line. Two horizontal lines are connected between the second and the third vertical line. A label ladder logic program is placed over the first horizontal line connected between the second and the third vertical line. From left to right, an XIC open symbol labeled as I ratio one slash three, and an output energize (highlighted in green) labeled as O ratio two slash five at the top and force greater than ON at the bottom are connected in series along the first horizontal line. From left to right, an XIC open symbol labeled as O ratio two slash five at the top and force greater than ON at the bottom, and an output energize labeled as O ratio two slash six are connected in series along the second horizontal line. Two horizontal lines are connected at the left side of the vertical line L2. From top to bottom, solenoid labeled as O ratio two slash five, and motor starter coil labeled O ratio two slash six are connected along the horizontal lines connected to the line L2. A label ON is shown below the solenoid and the motor starter coil. A label outputs is shown over the first horizontal line connected to the line L2.

Return to Figure 9-19

OceanofPDF.com

Figure 9-20 Extended Description (Chapter 9)

Return to Figure 9-20

From top, three horizontal lines L1, L2, and L3 are labeled as power mains. A node at L1 is connected to the left end of the primary of the transformer, through an open switch. A node at L2 is connected to the right end of the primary of the transformer, through another open switch. The open switches are highlighted and labeled as main disconnect switch. The transformer is labeled as step-down isolation transformer. The secondary of the transformer is connected across two terminals labeled as 120 V A C. The left terminal is connected to the fuse and the right terminal is connected to a grounded node. A ladder diagram with four horizontal lines connected between two vertical lines L1 and L2 is shown. The left end of the fuse is connected to the top end of the L1. The grounded node is connected to the top end of the L2. From left to right, NC pushbutton labeled as emergency stop, NC limit switch labeled as overtravel limit switch, NC pushbutton labeled as stop, N O pushbutton labeled as start, and a relay coil M C R labeled as master control relay are connected in series along the first horizontal line. An N O relay contact labeled as M C R is connected across the N O pushbutton. The NC and N O pushbutton are labeled as machine start or stop buttons. The NC pushbutton labeled as emergency stop, and the NC limit switch labeled as overtravel limit switch are highlighted and labeled as emergency stop switches. P L C power supply is connected along the second horizontal line. From left to right, P L C power supply is labeled as L1, L2, and G N D. The G N D label is grounded. An N O relay contact labeled as M C R is connected along the L1 between the second and the third horizontal lines. P L C input module is connected across the third horizontal line. From left to right, P L C input module is labeled as L1, one, two, three, four, five, six, seven, eight, and L2. An N O pushbutton is connected across the labels L1, and one of the P L C input module. An NC limit switch is connected across the labels L1, one, and two of the P L C input module. P L C output module is connected along the fourth horizontal line. From left to right, P L C output module is labeled as L1, one, two, three, four, five, six, seven, eight, and L2. A solenoid is connected across the labels eight, and L2 of the P L C output module. A pilot light is connected across the labels seven, eight, and L2 of the P L C output module.

Return to Figure 9-20

OceanofPDF.com

Figure 9-21 Extended Description (Chapter 9)

Return to Figure 9-21

At top side of the device, markings from left to right are one, two, and seven. Number two points at a display. In the middle region of the device, markings from left to right are ten, five, six, four, three, 12, eight, and nine. In the bottom region of the device, markings from left to right are 11, and seven. The table has two columns and 13 rows. The labels along the first column are: Number, one, two, three, four, five, six, seven, eight, nine, ten, 11, and 12. The labels along the second column are: module status indicators, alphanumeric display, node address switches, baud rate switches, U S B port, device net communication connector, terminal connectors, input status indicators, output status indicators, I P address display switch, ethernet connector, and service switch.

Return to Figure 9-21

OceanofPDF.com

Figure 9-22 Extended Description (Chapter 9)

Return to Figure 9-22

Four vertical lines are placed beside each other. From left to right, the first vertical line is labeled as L1 and the fourth vertical line is labeled as L2. From the right side, the line L1 is connected to three horizontal lines. An N C pushbutton labeled stop is connected along the first horizontal line. An N O pushbutton labeled start is connected along the second horizontal line. An N O relay contact labeled starter auxiliary contact. A label inputs is placed over the horizontal lines connected to L1. A horizontal line is placed between the second and the third vertical line. From left to right, an XIC open symbol labeled as stop (highlighted in green), XIC open symbol labeled start, and an output energize labeled M are connected in series along the horizontal line. An XIC open symbol labeled as starter auxiliary contact is parallelly connected across the XIC open symbol labeled start. A label ladder logic program is placed over the horizontal lines between the second and the third vertical line. From left, the line L2 is connected with a horizontal line. From left to right, the motor starter coil labeled as M and NC relay contact labeled O L are connected in series along the horizontal line at the left side of the L2. A label output is placed over the horizontal line connected to L2. An illustration of a device with a label auxiliary contact at the bottom-left side and a label main power contact at the top-right side.

Return to Figure 9-22

OceanofPDF.com

Figure 9-23 Extended Description (Chapter 9)

Return to Figure 9-23

Four vertical lines are placed beside each other. From left to right, the first vertical line is labeled as L1 and the fourth vertical line is labeled as L2. From the right side, the line L1 is connected with two horizontal lines. An N C pushbutton labeled stop is connected along the first horizontal line. An N O pushbutton labeled start is connected along the second horizontal line. A label inputs is placed over the horizontal lines connected to L1. An illustration of actual pushbutton is shown between the first and the second horizontal lines connected to the vertical line L1. A horizontal line is placed between the second and the third vertical line. From left to right, an XIC open symbol labeled as stop (highlighted in green), XIC open symbol labeled start, and an output energize labeled M are connected in series along the horizontal line. An XIC open symbol labeled as M is parallelly connected across the XIC open symbol labeled start. A label ladder logic program is placed over the horizontal lines between the second and the third vertical line. From left, the line L2 is connected with a horizontal line. From left to right, the motor starter coil labeled as M and NC relay contact labeled O L are connected in series along the horizontal line at the left side of the L2. A label output is placed over the horizontal line connected to L2.

Return to Figure 9-23

OceanofPDF.com

Figure 9-24 Extended Description (Chapter 9)

Return to Figure 9-24

There are three vertical lines. The first vertical line is labeled L1. The line L1 is connected with a horizontal line at the right side of L1. An N O switch labeled SW 13 is connected along the horizontal line. At the bottom of the vertical line L1, there are three dots, and a downwards pointing arrow attached to the third dot. A label inputs is shown over the horizontal line connected to the vertical line L1. From top to bottom, seven horizontal lines are connected between the second and the third vertical line. A label main program is shown over the first horizontal line connected between the second and the third vertical line. Another label remainder of main program is placed below the seventh horizontal line connected between the second and the third vertical line. Below both the labels, there are three dots, and a downwards pointing arrow attached to the third dot. From left to right, an XIC open symbol labeled SW 12, T O N instruction, and an output energize E N are connected in series along the first horizontal line. The right side of the T O N is connected to an output energize D N. From top to bottom, T O N instruction is labeled as T O N, timer on delay, timer T four ratio two, time base one-point-zero, preset five, and accumulated zero. From left to right, an XIC open symbol labeled T four ratio two at the top and D N at the bottom, and an output energize labeled LT4 are connected in series along the second horizontal line. From left to right, an XIC open symbol labeled SW 13, and an output energize T N D are connected in series along the third horizontal line. A label temporary end is placed between the second and the third horizontal line. From left to right, an XIC open symbol labeled SW 1, and an output energize L labeled LT1 are connected in series along the fourth horizontal line. From left to right, L B L instruction, an XIC open symbol labeled SW 2, and an output energize U labeled LT1 are connected in series along the fifth horizontal line. From left to right, an XIC open symbol labeled SW 3, R T O instruction, and an output energize E N are connected in series along the sixth horizontal line. The right side of the R T O is connected to an output energize D N. From top to bottom, R T O instruction is labeled as R T O, retentive timer on, timer T four ratio three, time base (sec) one-point-zero, preset 50, and accumulated zero. From left to right, an XIC open symbol labeled PB1, and an output energize R E S labeled T four ratio three are connected in series along the seventh horizontal line.

Return to Figure 9-24

OceanofPDF.com

Figure 9-25 Extended Description (Chapter 9)

Return to Figure 9-25

There are four vertical lines. The first vertical line is labeled L1 and the fourth vertical line is labeled L2. L1 is connected with six horizontal lines at the right side of L1. From top to bottom, N O switches labeled S1, S2, S3, S4, S5, and S6 are connected along the six horizontal lines. A label inputs is shown over the first horizontal line connected to the vertical line L1. From top to bottom, eight horizontal lines between the second and the third vertical line are labeled from one through eight. A label ladder logic program is shown over the horizontal line labeled one, which is connected between the second and the third vertical line. From left to right, an XIC open symbol labeled S1, and an output energize M C R are connected in series along the horizontal line one. From left to right, an XIC open symbol labeled S2, and an output energize labeled PL1 are connected in series along the horizontal line two. From left to right, an XIC open symbol labeled S3, and an output energize L labeled as PL2 are connected in series along the horizontal line three. From left to right, an XIC open symbol labeled S4, and an output energize U labeled PL2 are connected in series along the horizontal line four. An output energize M C R is connected along the horizontal line five. From left to right, an XIC open symbol labeled S5, and an output energize M C R are connected in series along the horizontal line six. From left to right, an XIC open symbol labeled S6, T O N instruction, and an output energize E N are connected in series along the horizontal line seven. The right side of the T O N is connected to an output energize D N. From top to bottom, T O N instruction is labeled as T O N, timer on delay, timer T four ratio one, time base one-point-zero, preset ten, and accumulated zero. An output energize M C R is connected along the horizontal line eight. Two horizontal lines are connected to the left side of the line L2. A label outputs is shown over the first horizontal line connected to the vertical line L2. From top to bottom, pilot lights labeled as PL1, and PL2 are connected along the horizontal lines connected to the vertical line L2.

Return to Figure 9-25

OceanofPDF.com

Figure 9-26 Extended Description (Chapter 9)

Return to Figure 9-26

There are four vertical lines. The first vertical line is labeled L1 and the fourth vertical line is labeled L2. L1 is connected with five horizontal lines at the right side of L1. From top to bottom, N O switches labeled S1, S2, S3, S4, and S5 are connected along the five horizontal lines. A label inputs is shown over the first horizontal line connected to the vertical line L1. From top to bottom, five horizontal lines between the second and the third vertical line are labeled from one through five. A label ladder logic program is shown over the horizontal line labeled one, which is connected between the second and the third vertical line. From left to right, an XIC open symbol labeled S1, and an output energize labeled PL3 are connected in series along the horizontal line one. From left to right, an XIC open symbol labeled S2, and J M P instruction labeled ten are connected in series along the horizontal line two. From left to right, an XIC open symbol labeled S3, and an output energize labeled as PL1 are connected in series along the horizontal line three. From left to right, an XIC open symbol labeled S5, and an output energize labeled PL4 are connected in series along the horizontal line four. From left to right, L B L instruction, an XIC open symbol labeled S4, and an output energize labeled PL2 are connected in series along the horizontal line five. Four horizontal lines are connected to the left side of the line L2. A label outputs is shown over the first horizontal line connected to the vertical line L2. From top to bottom, pilot lights labeled as PL1, PL2, PL3, and PL4 are connected along the horizontal lines connected to the vertical line L2.

Return to Figure 9-26

OceanofPDF.com

Figure 9-27 Extended Description (Chapter 9)

Return to Figure 9-27

There are four vertical lines. The first vertical line is labeled L1 and the fourth vertical line is labeled L2. L1 is connected with five horizontal lines at the right side of L1. From top to bottom, N O switches labeled S1, S2, S3, S4, and S5 are connected along the five horizontal lines. A label inputs is shown over the first horizontal line connected to the vertical line L1. From top to bottom, six horizontal lines between the second and the third vertical line are labeled from one through six. A label ladder logic program main program file two is shown over the horizontal line labeled one, which is connected between the second and the third vertical line. From left to right, an XIC open symbol labeled S1, and an output energize labeled PL1 are connected in series along the horizontal line one. From left to right, an XIC open symbol labeled S2, and J S R instruction are connected in series along the horizontal line two. From top to bottom, the J S R instruction is labeled as J S R, jump-to-subroutine, and S B R file number U ratio three. From left to right, an XIC open symbol labeled S3, and an output energize labeled as PL2 are connected in series along the horizontal line three. From left to right, an XIC open symbol labeled S5, and an output energize labeled PL4 are connected in series along the horizontal line four. From left to right, S B R instruction, an XIC open symbol labeled S4, and an output energize labeled PL3 are connected in series along the horizontal line five. S B R instruction is labeled as S B R, and subroutine. R E T instruction is connected along the horizontal line six. R E T instruction is labeled as R E T, and return. Four horizontal lines are connected to the left side of the line L2. A label outputs is shown over the first horizontal line connected to the vertical line L2. From top to bottom, pilot lights labeled as PL1, PL2, PL3, and PL4 are connected along the horizontal lines connected to the vertical line L2. The second and the third vertical lines shows discontinuity between the horizontal line four and five. This gap is labeled as subroutine file three.

Return to Figure 9-27

OceanofPDF.com

Figure 9-28 Extended Description (Chapter 9)

Return to Figure 9-28

There are four vertical lines. The first vertical line is labeled L1 and the fourth vertical line is labeled L2. L1 is connected with 13 horizontal lines at the right side of L1. From top to bottom, N O switches labeled S1, S2, S3, S4, S5, S6, S7, S8, S9, S10, S11, S12, and S13 are connected along the 13 horizontal lines. A label inputs is shown over the first horizontal line connected to the vertical line L1. From top to bottom, 15 horizontal lines between the second and the third vertical line are labeled from one through 15. A label ladder logic program main program file two is shown over the horizontal line labeled one, which is connected between the second and the third vertical line. From left to right, an XIC open symbol labeled S1, and an output energize labeled PL1 are connected in series along the horizontal line one. From left to right, an XIC open symbol labeled S2, and J S R instruction are connected in series along the horizontal line two. From top to bottom, the J S R instruction is labeled as J S R, jump-to-subroutine, and S B R file number U ratio three. From left to right, an XIC open symbol labeled S3, and an output energize J M P labeled as 20 are connected in series along the horizontal line three. From left to right, an XIC open symbol labeled S4, and an output energize labeled PL6 are connected in series along the horizontal line four. From left to right, an XIC open symbol labeled S9, and an output energize J M P labeled 20 are connected in series along the horizontal line five. From left to right, an XIC open symbol labeled S8, and an output energize labeled PL2 are connected in series along the horizontal line six. From left to right, an XIC open symbol labeled S10, and an output energize J M P labeled 20 are connected in series along the horizontal line seven. From left to right, an XIC open symbol labeled S11, and an output energize labeled PL4 are connected along the horizontal line eight. From left to right, L B L instruction labeled 20, an XIC open symbol labeled as S7, R T O instruction, and an output energize E N are connected in series along the horizontal line nine. The right side of the R T O is connected to an output energize D N. From top to bottom, R T O instruction is labeled as R T O, retentive timer on, timer T four ratio three, time base (sec) one-point-zero, preset 50, and accumulated zero. From left to right, an XIC open symbol labeled as T four ratio three slash D N, and an output energize R E S labeled as T four ratio three are connected in series along the horizontal line ten. From left to right, S B R instruction, an XIC open symbol labeled subroutine file three S13, and an output energize labeled PL3 are connected in series along the horizontal line 11. S B R instruction is labeled as S B R, and subroutine. From left to right, an XIC open symbol labeled S12, and an output energize J M P labeled 12 are connected in series along the horizontal line 12. From left to right, an XIC open symbol labeled S5, and an output energize labeled PL5 are connected along the horizontal line 13. From left to right, L B L instruction labeled 12, an XIC open symbol labeled S6, T O N instruction, and an output energize E N are connected in series along the horizontal line 14. The right side of the T O N is connected to an output energize D N. From top to bottom, T O N instruction is labeled as T O N, timer on delay, timer T four ratio six, time base one-point-zero, preset five, and accumulated zero. R E T instruction is connected along the horizontal line 15. R E T instruction is labeled as R E T, and return. Six horizontal lines are connected to the left side of the line L2. A label outputs is shown over the first horizontal line

connected to the vertical line L2. From top to bottom, pilot lights labeled as PL1, PL2, PL3, PL4, PL5, and PL6 are connected along the horizontal lines connected to the vertical line L2.

Return to Figure 9-28

OceanofPDF.com

A Ladder Diagram Graphic Extended Description (Chapter 10)

Return to A Ladder Diagram Graphic

There are three vertical lines. From left, the first line is labeled as L 1. A label inputs is placed over the horizontal line connected at the right side of the line L 1. A selector switch with three terminals labeled from top to bottom as lime one, cola two, and cheery three is connected across the horizontal line. An illustration of the selector switch is placed below the symbol. There are three horizontal lines between the second and the third horizontal line. A label ladder logic program is placed over the first horizontal line connected between the second and the third line. From left to right, an X I C open symbol labeled as lime one, O S R instruction labeled as B 3 : 0/0, and C O P instruction are connected in series along the first horizontal line. From top to bottom, the C O P instruction is labeled as C O P, copy file, source # N 1 2 : 0, destination # N 7 : 5 0, and length five. From left to right, an X I C open symbol labeled as cola two, O S R instruction labeled as B 3 : 0/1, and C O P instruction are connected in series along the second horizontal line. From top to bottom, the C O P instruction is labeled as C O P, copy file, source # N 1 2 : 5, destination # N 7 : 5 0, and length five. From left to right, an X I C open symbol labeled as cherry three, O S R instruction labeled as B 3 : 0/2, and C O P instruction are connected in series along the third horizontal line. From top to bottom, the C O P instruction is labeled as C O P, copy file, source # N 1 2 : 1 0, destination # N 7 : 5 0, and length five.

Return to A Ladder Diagram Graphic

OceanofPDF.com

Figure 10-1 Extended Description (Chapter 10)

Return to Figure 10-1

Seven labels enclosed in rectangles are placed relatively on the left side between the arrow-heads. From left to right, the labels highlighted in blue are M O V, M V M, A N D, O R, X O R, N O T, and C L R. From bottom-left to bottom-right, an arrow-head facing left, an arrow-head facing right, labels: compare, compute or math, move or logical (highlighted in blue), and file or misc are shown.

Return to Figure 10-1

OceanofPDF.com

Figure 10-2(a-b) Extended Description (Chapter 10)

Return to Figure 10-2(a-b)

In part a, there is a label original data stored in registers N 7 : 3 0 and N 7 : 2 0, which is placed between the single set and the four sets. The single set of 16 blocks has a label N 7: 2 0 (highlighted in blue) at the left. From left to right the blocks of the single set are labeled as 0, 0, 0, 0, 1, 1, 1, 1, 0, 0, 0, 0, 1, 1, 1, and 1. All the labels in the blocks of the single set are highlighted in blue. From top to bottom, labels at the left side of the four sets are as follows: N 7 : 2 8, N 7 : 2 9, N 7 : 3 0 (highlighted in red), and N 7 : 3 1. From the top, only the third set has labels, which are highlighted in red. From left to right, the blocks of the third set among the four sets are labeled as 1, 1, 0, 0, 1, 1, 1, 0, 0, 1, 1, 0, 1, 1, 1, and 0. The vertical line after every four blocks is bolded in the four sets of 16 blocks. In part b, there is a label data transferred from register N 7 : 3 0 through N 7 : 2 0, which is placed between the single set and the four sets. The single set of 16 blocks has a label N 7 : 2 0 (highlighted in blue) at the left. From left to right the blocks of the single set are labeled as 1, 1, 0, 0, 1, 1, 1, 0, 0, 1, 1, 0, 1, 1, 1, and 0. All the labels in the blocks of the single set are highlighted in blue. From top to bottom, labels at the left side of the four sets are as follows: N 7 : 2 8, N 7 : 2 9, N 7 : 3 0 (highlighted in red), and N 7 : 3 1. From the top, only the third set has labels, which are highlighted in red. From left to right, the blocks of the third set among the four sets are labeled as 1, 1, 0, 0, 1, 1, 1, 0, 0, 1, 1, 0, 1, 1, 1, and 0. The vertical line after every four blocks is bolded in the four sets and the single set of 16 blocks. An arrow from the right side of the third set points at the right side of the single set.

Return to Figure 10-2(a-b)

OceanofPDF.com

Figure 10-3 Extended Description (Chapter 10)

Return to Figure 10-3

There are three vertical lines. From left the first vertical line is labeled as L 1. The N O switch labeled A is connected along a horizontal line connected at the right side of the vertical line L 1. A label input is shown over the horizontal line connected to the L 2. From left to right, an X I C open symbol labeled as A, and M O V instruction are connected in series along a horizontal line connected between the second and the third vertical line. From top to bottom, the M O V instruction is labeled as M O V, move, source N 7 : 3 0 (highlighted in blue), and destination N 7 : 2 0 (highlighted in red). A label ladder logic program is placed over the horizontal line connected between the second and the third vertical line. The table is placed at the right side. The eighth row of the table is longer than the other rows. An arrow from the twelfth row with a label N 7 : 3 0 points towards the fourth row with a label N 7 : 2 0.

Return to Figure 10-3

OceanofPDF.com

Figure 10-4 Extended Description (Chapter 10)

Return to Figure 10-4

There are three vertical lines. From left the first vertical line is labeled as L 1. The N O switch labeled A is connected along a horizontal line connected at the right side of the vertical line L 1. A label input is shown over the horizontal line connected to the L 2. From left to right, an X I C open symbol labeled as A, and M V M instruction are connected in series along a horizontal line connected between the second and the third vertical line. From top to bottom, the M V M instruction is labeled as M V M, masked move, source B 3 : 0, 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 (highlighted in blue), mask B 3 : 1, F F 0 F (placed at the right side and highlighted in green), destination B 3 : 4, and 1 0 1 0 1 0 1 0 1 1 0 0 1 0 1 0 (highlighted in red). A label ladder logic program is placed over the horizontal line connected between the second and the third vertical line. The illustration at the bottom shows four sets of 16 bits. From left to right, bits in the first set are highlighted in blue and given as 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0. From left to right, bits in the second set are highlighted in green and given as 1 1 1 1 1 1 1 0 0 0 0 1 1 1 1. From left to right, bits in the third set are given as 1 1 0 0 1 1 0 0 1 1 0 0 1 1 0 0. From left to right, bits in the fourth set are highlighted in red and given as 1 0 1 0 1 0 1 0 1 1 0 0 1 0 1 0. The ninth, tenth, eleventh, and twelfth bit in the fourth set are labeled as unchanged. The ninth, tenth, eleventh, and twelfth bit from the left is in red for every set. From top to bottom the sets are labeled as source B three : zero, mask F F zero F, destination B three : four before instruction went true, and destination B 3 : 4 after instruction went true.

Return to Figure 10-4

OceanofPDF.com

Figure 10-5 Extended Description (Chapter 10)

Return to Figure 10-5

There are four vertical lines. From left, the first line is labeled as L 1 and the fourth line is labeled as L 2. Two horizontal lines are connected to the right side of the vertical line L 1. A label inputs is placed over the horizontal line connected to the line L 1. From top to bottom, N O pushbutton labeled as P B 1, and N O selector switch labeled as S S 1 are connected along the horizontal lines connected to L 1. Four horizontal lines are connected between the second and the third horizontal line. A label ladder logic program is placed over the first horizontal line connected between the second and the third vertical line. From left to right, X I C open symbol labeled as P B 1, T O N instruction, and an output energize E N are connected in series along the first horizontal line labeled as one. The right side of the T O N is connected to an output energize D N. From top to bottom, T O N instruction is labeled as T O N, timer on delay, timer T 4 : 1, time base 1.0, preset 0, and accumulated 0. From left to right, X I O open symbol (highlighted in green) labeled as S S 1, and M O V instruction are connected in series along the second horizontal line labeled as two. From top to bottom, the M O V instruction is labeled as M O V, move, source N 7 : 1, 10 (at the right side), destination T 4 : 1 dot P R E, and 0 (at the right side). From left to right, X I C open symbol labeled as S S 1, and M O V instruction are connected in series along the third horizontal line labeled as three. From top to bottom, the M O V instruction is labeled as M O V, move, source N 7 : 2, 5 (at the right side), destination T 4 : 1 dot P R E, and 0 (at the right side). From left to right, X I C open symbol labeled as T 4 : /D N, and an output energize labeled as P L 1 are connected in series along the fourth horizontal line labeled as four. A horizontal line is connected at the left side of the line L 2. The pilot light labeled P L 1 is connected along the horizontal line connected to the line L 2. A label output is placed over the first horizontal line connected to L 2.

Return to Figure 10-5

OceanofPDF.com

Figure 10-6 Extended Description (Chapter 10)

Return to Figure 10-6

There are four vertical lines. From left, the first line is labeled as L 1 and the fourth line is labeled as L 2. Five horizontal lines are connected to the right side of the vertical line L 1. A label inputs is placed over the horizontal line connected to the line L 1. From top to bottom, N O limit switch labeled as L S 1, N O switch labeled as A, N O switch labeled as B, N O switch labeled as C, and N O pushbutton labeled as reset are connected along the horizontal lines connected to L 1. Six horizontal lines are connected between the second and the third horizontal line. A label ladder logic program is placed over the first horizontal line connected between the second and the third vertical line. From left to right, X I C open symbol labeled as L S 1, C T U instruction, and an output energize C U are connected in series along the first horizontal line labeled as one. The right side of the C T U is connected to an output energize D N. From top to bottom, C T U instruction is labeled as C T U, count up, counter C 5 : 1, preset 0, and accumulated 0. From left to right, X I C open symbol labeled as A, and M O V instruction are connected in series along the second horizontal line labeled as two. From top to bottom, the M O V instruction is labeled as M O V, move, source N 7 : 1, 300 (at the right side), destination C 5 : 1 dot P R E, and 0 (at the right side). From left to right, X I C open symbol labeled as B, and M O V instruction are connected in series along the third horizontal line labeled as three. From top to bottom, the M O V instruction is labeled as M O V, move, source N 7 : 2, 175 (at the right side), destination C 5 : 1 dot P R E, and 0 (at the right side). From left to right, X I C open symbol labeled as C, and M O V instruction are connected in series along the fourth horizontal line labeled as four. From top to bottom, the M O V instruction is labeled as M O V, move, source N 7 : 3, 50 (at the right side), destination C 5 : 1 dot P R E, and 0 (at the right side). From left to right, X I C open symbol labeled as C 5 : 1/D N, and an output energize labeled as full are connected in series along the fifth horizontal line labeled as five. From left to right, X I C open symbol labeled as reset, and an output energize R E S labeled as C 5 : 1 are connected in series along the sixth horizontal line labeled as six. A horizontal line is connected at the left side of the line L 2. The pilot light labeled full is connected along the horizontal line connected to the line L 2. A label output is placed over the first horizontal line connected to L 2.

Return to Figure 10-6

OceanofPDF.com

Figure 10-7 Extended Description (Chapter 10)

Return to Figure 10-7

For word-to-file move, six arrows from a block labeled word points at six rows of a block labeled file. For file-to-file move, an arrow from each row of a block with six rows points towards six rows of another block. Both the blocks are labeled as file. For file-to-word move, an arrow from each row of a block with six rows points towards a block labeled word. The block with six rows is labeled as file.

Return to Figure 10-7

OceanofPDF.com

Figure 10-8 Extended Description (Chapter 10)

Return to Figure 10-8

From left to right, the first cell of the first row is unlabeled, whereas remaining cells of the first row are labeled as 15, 14, 13, 12, 11, 10, nine, eight, seven, six, five, four, three, two, one, and zero. From top to bottom, except for the first cell, the other cells in the first column are labeled as N 7 : 3 0/, N 7 : 3 1/, N 7 : 3 2/, N 7 : 3 3/, N 7 : 3 4/, N 7 : 3 5/, N 7 : 3 6/, and N 7 : 3 7/. The remaining cells are labeled as zero. The second row is highlighted in blue and the last row is highlighted in red. An arrow from a label word N 7 : 30 points towards the second cell of the first column. A double headed arrow labeled as # N 7 : 3 0/ length = eight points at the last cell of the second row with a label file and points at the last cell of the last row with a label N 7 : 37. From left to right, the labels at the bottom are given as, radix: binary (enclosed in a rectangle with downwards pointing arrow-head at the right side), and table: N 7 : integer (enclosed in a rectangle with downwards pointing arrow-head at the right side).

Return to Figure 10-8

OceanofPDF.com

Figure 10-9 Extended Description (Chapter 10)

Return to Figure 10-9

From top to bottom, the second terminal is connected to an output energize D N, and the third terminal is connected to an output energize E R. From top to bottom, F A L instruction is labeled as F A L, file arith or logical, control, length, position, mode, destination, and expression. A shaded horizontal rectangle is placed at the right side of each label: control, length, position, mode, destination, and expression.

Return to Figure 10-9

OceanofPDF.com

Figure 10-10 Extended Description (Chapter 10)

Return to Figure 10-10

There are three vertical lines. From the left, the first vertical line is labeled as L 1. The N O switch labeled A is connected along a horizontal line connected at the right side of the vertical line L 1. A label input is shown over the horizontal line connected to the L 2. From left to right, X I C open symbol labeled as A, F A L instruction, and an output energize E N are connected in series along a horizontal line connected between the second and the third vertical line. From the right side, the second terminal of the F A L is connected to an output energize D N, and the third terminal of the F A L is connected to an output energize E R. From top to bottom, F A L instruction is labeled as F A L, file arith or logical, control, length, position, mode, destination, and expression. A shaded horizontal rectangle is placed at the right side of each label. From top to bottom, the shaded rectangles are labeled as R six : one, six, zero, all, # N 7 : 5 0, 528, and # N 7 : 2 0. A label ladder logic program is placed over the horizontal line connected between the second and the third vertical line. For the illustration, an arrow from each row of a block with six rows points towards six rows of another block. The block from which the arrow diverges is at the left side and labeled as expression # N 7 : 2 0. The block towards which the arrow converges is at the right side and labeled as destination # N 7 : 5 0. From top to bottom, the rows of both the blocks are labeled as 528, 621, 778, 986, 342, and 135. A label file is shown at the left side of the block from which arrows diverge. A label N 7 : 2 0 is placed at the left of the row labeled 528 in the left block. A label N 7 : 2 5 is placed at the left of the row labeled 135 in the left block. A label N 7 : 5 0 is placed at the right of the row labeled 528 in the right block. A label N 7 : 5 5 is placed at the right of the row labeled 135 in the right block.

Return to Figure 10-10

OceanofPDF.com

Figure 10-11 Extended Description (Chapter 10)

Return to Figure 10-11

There are three vertical lines. From the left, the first vertical line is labeled as L 1. The N O switch labeled A is connected along a horizontal line connected at the right side of the vertical line L 1. A label input is shown over the horizontal line connected to the L 2. From left to right, X I C open symbol labeled as A, F A L instruction, and an output energize E N are connected in series along a horizontal line connected between the second and the third vertical line. From the right side, the second terminal of the F A L is connected to an output energize D N, and the third terminal of the F A L is connected to an output energize E R. From top to bottom, F A L instruction is labeled as F A L, file arith or logical, control R 6 : 6, length five, position zero, mode I N C, destination N 2 9 : 5, and expression # N 2 9 : 0. A label ladder logic program is placed over the horizontal line connected between the second and the third vertical line. For the illustration, an arrow from each row of a block with five rows points towards a block labeled word. The block from which the arrow diverges is at the left side and labeled as file # N 2 9 : 0. From top to bottom, the rows of the left block have labels at the left side as word zero, one, two, three, and four. The vertical lines of the block are extended downwards and has a forward slash at the bottom. A label word N 29 : five is placed over the block labeled word. From top to bottom, the arrows are labeled as first move, second move, third move, fourth move, and fifth move.

Return to Figure 10-11

OceanofPDF.com

Figure 10-12 Extended Description (Chapter 10)

Return to Figure 10-12

There are three vertical lines. From the left, the first vertical line is labeled as L 1. The N O switch labeled A is connected along a horizontal line connected at the right side of the vertical line L 1. A label input is shown over the horizontal line connected to the L 2. From left to right, X I C open symbol labeled as A, F A L instruction, and an output energize E N are connected in series along a horizontal line connected between the second and the third vertical line. From the right side, the second terminal of the F A L is connected to an output energize D N, and the third terminal of the F A L is connected to an output energize E R. From top to bottom, F A L instruction is labeled as F A L, file arith or logical, control R 6 : 2, length six, position zero, mode incremental, destination # N 7 : 1 0 1, 0 (at the right side), and expression N 7 : 1 0 0. A label ladder logic program is placed over the horizontal line connected between the second and the third vertical line. For the illustration, an arrow from a block labeled word points towards each row of a block with five rows. The block from which the arrow diverges is at the left side and labeled as N 7 : 1 0 0. A label word N 7 : 1 0 1 is placed at the right side of the first row of the right block. A label word N 7 : 1 0 6 is placed at the right side of the last row of the right block. From top to bottom, the first arrow is labeled as first move.

Return to Figure 10-12

OceanofPDF.com

Figure 10-13 Extended Description (Chapter 10)

Return to Figure 10-13

The rows in the top block are labeled as 3452, 6789, 8321, and 983. A label # N 7 : 1 0 length = four is placed at the right side of the first block. An arrow labeled file-to-file copy from first block points towards the second block. The rows in the second block are labeled as T 4 : 0 preset value, T 4 : 1 preset value, T 4 : 2 preset value, and T 4 : 3 preset value. A label # T 4 : 0 dot P R E length = four is placed at the right side of the second block. Another arrow labeled file-to-file copy points from the third block towards the second block. The rows in the third block are labeled as 778, 986, 342, and 135. A label # N 7 : 2 0 length = four is placed at the right side of the third block.

Return to Figure 10-13

OceanofPDF.com

Figure 10-14 Extended Description (Chapter 10)

Return to Figure 10-14

From top, a toolbar labeled as location in R S logix software has an arrow-head pointing left at the top-left side and an arrow-head pointing right at the top-right side is shown. Nine labels enclosed in rectangles are placed relatively on the left side between the arrow-heads. From left to right, the labels highlighted in blue are C O P (highlighted in blue), F L L, D D V, S C L, I N T, S T E, S T S, S T D, and P I D. From bottom-left to bottom-right, an arrow-head facing left, an arrow-head facing right, labels: move or logical, file or misc (highlighted in blue), file shift or sequencer, and prog are shown. Below the toolbar, the ladder diagram is shown. There are three horizontal lines. From left, the first vertical line is labeled as L 1. An N O switch labeled as A is connected along the horizontal line connected to the right side of the vertical line L 1. A label input is placed above the first horizontal line connected to L 1. From left to right, an X I C open symbol labeled as A, and C O P instruction are connected in series along the horizontal line connected between the second and the third horizontal line. From top to bottom, C O P instruction is labeled as C O P, copy file, source # N 4 0 : 1, destination # N 2 0 : 1, and length 6. A label ladder logic program is placed over the horizontal line connected between the second and the third line. At the bottom, an illustration of two tables with six rows are placed next to each other. An arrow from each row of the table on the left side points towards the rows of the table on the right side. From top to bottom, the table on the left side is labeled as N 4 0 : 1, N 4 0 : 2, N 4 0 : 3, N 4 0 : 4, N 4 0 : 5, and N 4 0 : 6. From top to bottom, the table on the right side is labeled as N 2 0 : 1, N 2 0 : 2, N 2 0 : 3, N 2 0 : 4, N 2 0 : 5, and N 2 0 : 6.

Return to Figure 10-14

OceanofPDF.com

Figure 10-15 Extended Description (Chapter 10)

Return to Figure 10-15

From top, a toolbar labeled as location in R S logix software has an arrow-head pointing left at the top-left side and an arrow-head pointing right at the top-right side is shown. Nine labels enclosed in rectangles are placed relatively on the left side between the arrow-heads. From left to right, the labels highlighted in blue are C O P, F L L (highlighted in blue), D D V, S C L, I N T, S T E, S T S, S T D, and P I D. From bottom-left to bottom-right, an arrow-head facing left, an arrow-head facing right, labels: move or logical, file or misc (highlighted in blue), file shift or sequencer, and prog are shown. Below the toolbar, the ladder diagram is shown. There are three horizontal lines. From left, the first vertical line is labeled as L 1. An N O switch labeled as A is connected along the horizontal line connected to the right side of the vertical line L 1. A label input is placed above the first horizontal line connected to L 1. From left to right, an X I C open symbol labeled as A, and F L L instruction are connected in series along the horizontal line connected between the second and the third horizontal line. From top to bottom, F L L instruction is labeled as F L L, fill file, source N 1 5 : 5, destination # N 2 0 : 1, and length 6. A label ladder logic program is placed over the horizontal line connected between the second and the third line. At the bottom, an illustration of a table with a row (left) is placed next to a table with six rows (right). Six arrows from the row of the table on the left side points towards the six rows of the table on the right side. The table on the left side is labeled as N 1 5 : 5. From top to bottom, the table on the right side is labeled as N 2 0 : 1, N 2 0 : 2, N 2 0 : 3, N 2 0 : 4, N 2 0 : 5, and N 2 0 : 6.

Return to Figure 10-15

OceanofPDF.com

Figure 10-16 Extended Description (Chapter 10)

Return to Figure 10-16

There are three vertical lines. From left the first line is labeled as L 1. Two horizontal lines are connected at the right side of the line L 1. From top to bottom, N O pushbutton labeled as P B 1, and N O pushbutton labeled as P B 2 are connected along the horizontal lines connected to L 1. A label inputs is placed over the first horizontal line connected to L 1. There are two horizontal lines connected between the second and the third vertical lines. From left to right, X IC open symbol labeled as P B 1, and C O P instruction are connected in series along the first horizontal line. From top to bottom, C O P instruction is labeled as C O P, copy file, source #N 1 0 : 0, destination # N 1 2 : 0, and length 4. From left to right, X IC open symbol labeled as P B 2, and F L L instruction are connected in series along the second horizontal line. From top to bottom, F L L instruction is labeled as F L L, fill file, source 0, destination # N 1 2 : 0, and length 4. A label ladder logic program is placed over the first horizontal line connected between the second and the third vertical line.

Return to Figure 10-16

OceanofPDF.com

Figure 10-17 Extended Description (Chapter 10)

Return to Figure 10-17

There are three vertical lines. From left, the first line is labeled as L 1. A label inputs is placed over the horizontal line connected at the right side of the line L 1. A selector switch with three terminals labeled from top to bottom as lime one, cola two, and cheery three is connected across the horizontal line. An illustration of the selector switch is placed below the symbol. There are three horizontal lines between the second and the third horizontal line. A label ladder logic program is placed over the first horizontal line connected between the second and the third line. From left to right, an X I C open symbol labeled as lime one, O S R instruction labeled as B 3 : 0/0 and C O P instruction are connected in series along the first horizontal line. From top to bottom, the C O P instruction is labeled as C O P, copy file, source # N 1 2 : 0, destination # N 7 : 5 0, and length five. From left to right, an X I C open symbol labeled as cola two, O S R instruction labeled as B 3 : 0/1, and C O P instruction are connected in series along the second horizontal line. From top to bottom, the C O P instruction is labeled as C O P, copy file, source # N 1 2 : 5, destination # N 7 : 5 0, and length five. From left to right, an X I C open symbol labeled as cherry three, O S R instruction labeled as B 3 : 0/2 and C O P instruction are connected in series along the third horizontal line. From top to bottom, the C O P instruction is labeled as C O P, copy file, source # N 1 2 : 1 0, destination # N 7 : 5 0, and length five.

Return to Figure 10-17

OceanofPDF.com

Figure 10-19 Extended Description (Chapter 10)

Return to Figure 10-19

A P L C programming interface menu. There are eight rectangular blocks in a horizontal row, each with uppercase labels. From left to right, the labels read: L I M (Limit), M E Q (Masked Equal), E Q U (Equal), N E Q (Not Equal), L E S (Less Than), G R T (Greater Than), L E Q (Less Than or Equal), G E Q (Greater Than or Equal).

Return to Figure 10-19

OceanofPDF.com

Figure 10-20 Extended Description (Chapter 10)

Return to Figure 10-20

There are three vertical lines. From left, the third line is labeled as L 2. A label ladder logic program is placed over the horizontal line connected between the first and the second vertical line. From left to right, E Q U instruction, and an output energize labeled as P L 1 are connected across a horizontal line connected between the first and the second vertical line. From top to bottom, E Q U instruction is labeled as E Q U, equal, source A, T 4 : 0 dot A C C (highlighted in blue), source B, and N 7 : 4 0 (highlighted in red). A label output is placed over a horizontal line connected at the left side of the line L 2. The pilot light labeled as P L 1 is connected across the horizontal line connected to the L 2.

Return to Figure 10-20

OceanofPDF.com

Figure 10-21 Extended Description (Chapter 10)

Return to Figure 10-21

There are three vertical lines. From left, the third line is labeled as L 2. A label ladder logic program is placed over the horizontal line connected between the first and the second vertical line. From left to right, N E Q instruction, and an output energize (highlighted in green) labeled as P L 1 are connected across a horizontal line connected between the first and the second vertical line. From top to bottom, N E Q instruction is labeled as N E Q, not equal, source A N 7 : 5, 30 (highlighted in blue and placed at the right side), and source B 25 (highlighted in red). A label output is placed over a horizontal line connected at the left side of the line L 2. The pilot light labeled as P L 1 is connected across the horizontal line connected to the L 2. The symbol of pilot light is shaded.

Return to Figure 10-21

OceanofPDF.com

Figure 10-22 Extended Description (Chapter 10)

Return to Figure 10-22

There are three vertical lines. From left, the third line is labeled as L 2. A label ladder logic program is placed over the horizontal line connected between the first and the second vertical line. From left to right, G R T instruction, and an output energize labeled as P L 1 are connected across a horizontal line connected between the first and the second vertical line. From top to bottom, G R T instruction is labeled as G R T, greater than (A greater than B), source A, T 4 : 1 0 dot A C C (highlighted in blue), source B, and 200 (highlighted in red). A label output is placed over a horizontal line connected at the left side of the line L 2. The pilot light labeled as P L 1 is connected across the horizontal line connected to the L 2.

Return to Figure 10-22

OceanofPDF.com

Figure 10-23 Extended Description (Chapter 10)

Return to Figure 10-23

There are three vertical lines. From left, the third line is labeled as L 2. A label ladder logic program is placed over the horizontal line connected between the first and the second vertical line. From left to right, L E S instruction, and an output energize labeled as P L 1 are connected across a horizontal line connected between the first and the second vertical line. From top to bottom, L E S instruction is labeled as L E S, less than (A less than B), source A, C 5 : 1 0 dot A C C (highlighted in blue), source B, and 350 (highlighted in red). A label output is placed over a horizontal line connected at the left side of the line L 2. The pilot light labeled as P L 1 is connected across the horizontal line connected to the L 2.

Return to Figure 10-23

OceanofPDF.com

Figure 10-24 Extended Description (Chapter 10)

Return to Figure 10-24

There are three vertical lines. From left, the third line is labeled as L 2. A label ladder logic program is placed over the horizontal line connected between the first and the second vertical line. From left to right, G E Q instruction, and an output energize (highlighted in green) labeled as P L 1 are connected across a horizontal line connected between the first and the second vertical line. From top to bottom, G E Q instruction is labeled as G E Q, greater than or equal, A greater than or equal to B (enclosed in bracket and placed in the middle), source A N 7 : 5 5, 100 (highlighted in blue and placed at the right side), source B N 7 : 1 2, and 23 (highlighted in red and placed at the right side). A label output is placed over a horizontal line connected at the left side of the line L 2. The pilot light labeled as P L 1 is connected across the horizontal line connected to the L 2. The symbol of pilot light is shaded.

Return to Figure 10-24

OceanofPDF.com

Figure 10-25 Extended Description (Chapter 10)

Return to Figure 10-25

There are three vertical lines. From left, the third line is labeled as L 2. A label ladder logic program is placed over the horizontal line connected between the first and the second vertical line. From left to right, L E Q instruction, and an output energize labeled as P L 1 are connected across a horizontal line connected between the first and the second vertical line. From top to bottom, L E Q instruction is labeled as L E Q, less than or equal, A less than or equal to B (enclosed in brackets), source A, C 5 : 1 dot A C C (highlighted in blue), source B, and 457 (highlighted in red). A label output is placed over a horizontal line connected at the left side of the line L 2. The pilot light labeled as P L 1 is connected across the horizontal line connected to the L 2.

Return to Figure 10-25

OceanofPDF.com

Figure 10-26 Extended Description (Chapter 10)

Return to Figure 10-26

There are four vertical lines. From left, the first line is labeled as L 1 and the fourth line is labeled as L 2. Two horizontal lines are connected to the right side of the vertical line L 1. A label input is placed over the horizontal line connected to the line L 1. From top to bottom, N O pushbutton labeled as counter input, and N O pushbutton labeled as counter reset are connected along the horizontal lines connected to L 1. Five horizontal lines are connected between the second and the third horizontal line. A label ladder logic program is placed over the first horizontal line connected between the second and the third vertical line. From left to right, X I C open symbol labeled as counter input, C T U instruction labeled as compare counter, and an output energize C U are connected in series along the first horizontal line. The right side of the C T U is connected to an output energize D N. From top to bottom, C T U instruction is labeled as C T U, count up, counter C 5 : 1, preset 50, and accumulated 0. From left to right, L E S instruction, and an output energize labeled as less output are connected in series along the second horizontal line. From top to bottom, L E S instruction is labeled as L E S, less than, source A C 5 : 1 dot A C C, 0 (at the right side), and source B 10. From left to right, E Q U instruction, and an output energize labeled as E Q U output are connected in series along the third horizontal line. From top to bottom, the E Q U instruction is labeled as E Q U, equal, source A C 5 : 1 dot A C C, 0 (at the right side), and source B 20. From left to right, the G R T instruction, and an output energize labeled as G R T output are connected in series along the fourth horizontal line. From top to bottom, the G R T instruction is labeled as G R T, greater than, source A C 5 : 1 dot A C C, 0 (at the right side), and source B 30. From left to right, X I C open symbol labeled as counter reset, and an output energize R E S labeled as C5 : 1 are connected in series along the fifth horizontal line. Three horizontal lines are connected at the left side of the line L 2. From top to bottom, lights labeled as less output, E Q U output, and G R T output are connected along the horizontal lines connected to the line L 2. A label output is placed over the first horizontal line connected to L 2.

Return to Figure 10-26

OceanofPDF.com

Figure 10-27 Extended Description (Chapter 10)

Return to Figure 10-27

There are three vertical lines. The third vertical line is labeled as L 2. From left to right, L I M instruction, the output energize are connected in series along the horizontal line connected between the second and the third vertical line. From top to bottom, L I M is labeled as L I M, limit test, low limit (highlighted in blue) N 7 : 2 2, 25 (highlighted in blue and placed at the right side), test (highlighted in green) N 7 : 2 3, 48 (highlighted in green and placed at the right side), high limit (highlighted in red) N 7 : 2 4, and 50 (highlighted in red and placed at the right side). A label ladder logic program is placed over the horizontal line connected between the second and the third vertical line. The pilot light (shaded) is connected along a horizontal line connected at the left side of the line L 2. A label output is placed over the horizontal line connected to L 2. A double headed arrow is divided into three sections. From left, the first section is labeled as false, (less than 25), and 25 (highlighted in blue). The second section is labeled true. The third section is labeled as 50 (highlighted in red), false, and (greater than 50). The middle section of the arrow is highlighted in green.

Return to Figure 10-27

OceanofPDF.com

Figure 10-28 Extended Description (Chapter 10)

Return to Figure 10-28

There are three vertical lines. The third vertical line is labeled as L 2. From left to right, L I M instruction, the output energize are connected in series along the horizontal line connected between the second and the third vertical line. From top to bottom, L I M is labeled as L I M, limit test, low limit (highlighted in blue) N 7 : 2 8, 100 (highlighted in blue and placed at the right side), test (highlighted in green) N 7 : 2 9, 125 (highlighted in green and placed at the right side), high limit (highlighted in red) N 7 : 2 7, and 50 (highlighted in red and placed at the right side). A label ladder logic program is placed over the horizontal line connected between the second and the third vertical line. The pilot light (shaded) is connected along a horizontal line connected at the left side of the line L 2. A label output is placed over the horizontal line connected to L 2. A double headed arrow is divided into three sections. From left, the first section is labeled as true (highlighted in green), (less than or equal to 50), and 50 (highlighted in red). The second section is labeled false. The third section is labeled as 100 (highlighted in blue), (greater than or equal to 100), and true (highlighted in green). Except for the second section, the arrow is highlighted in green.

Return to Figure 10-28

OceanofPDF.com

Figure 10-29 Extended Description (Chapter 10)

Return to Figure 10-29

The toolbar shows an arrowhead pointing left at the top-left side and an arrowhead pointing right at the top-right side. Three symbols and seven labels enclosed in rectangles are placed relatively on the left side between the arrowheads. From left to right, the symbols are, an upper-case letter I rotated at ninety degrees, same symbol as the first but includes a rectangle at the bottom of the middle line of the letter I, and same symbol as the second but includes a horizontal line dividing the rectangle into half. From left to right, the labels are C M P (highlighted in blue), L I M (highlighted in blue),M E Q (highlighted in blue), E Q U, N E Q, L E (highlighted in blue), and G R T (highlighted in blue). Seven labels are shown at the bottom. From left to right, the labels are arrow-head facing, arrow-head facing right, bit, timer or counter, input or output, and compare (highlighted in blue). The ladder diagram is placed below the toolbar. There are four vertical lines. From left, the first line is labeled as L 1 and the fourth line is labeled as L 2. Two horizontal lines are connected to the right side of the vertical line L 1. A label inputs is placed over the first horizontal line connected to the line L 1. From top to bottom, N O pushbutton labeled as counter_input, and N O pushbutton labeled as counter_reset are connected along the horizontal lines connected to L 1. An illustration of an actual pushbuttons are placed below their symbols. Three horizontal lines are connected between the second and the third horizontal line. A label ladder logic program is placed over the first horizontal line connected between the second and the third vertical line. From left to right, X I C open symbol labeled as counter_input <local:1:I:data.1>, C T U instruction, and an output energize C U are connected in series along the first horizontal line. The right side of the C T U is connected to an output energize D N. From top to bottom, C T U instruction is labeled as C T U, count up counter_range, counter, preset 20 (an arrow pointing at left), and accumulated 0 (an arrow pointing at left). From left to right, L I M instruction, and an output energize labeled as count_within_range_output <local:2:O.data.1> are connected in series along the second horizontal line. From top to bottom, the L I M instruction is labeled as L I M, limit test (C I R C), low limit 6, test counter_range dot A C C, 0 (at the right side), and high limit 12.. From left to right, an X I C open symbol labeled as counter_reset <local:1:I:data.2>, and an output energize R E labeled as counter range are connected in series along the third horizontal line. The light labeled as count_within_range_output is connected along a horizontal line connected at the left side of the vertical line L 2. A label output is placed over the horizontal line connected to L 2.

Return to Figure 10-29

OceanofPDF.com

Figure 10-31 Extended Description (Chapter 10)

Return to Figure 10-31

There are three vertical lines. The third vertical line is labeled as L 2. From left to right, M E Q instruction, the output energize are connected in series along the horizontal line connected between the second and the third vertical line. From top to bottom, M E Q is labeled as M E Q, source (highlighted in blue) N 7 : 5, mask (highlighted in green) N 7 : 6, and compare (highlighted in red) N 7 : 1 0. A label ladder logic program is placed over the horizontal line connected between the second and the third vertical line. The pilot light (shaded) is connected along a horizontal line connected at the left side of the line L 2. A label output is placed over the horizontal line connected to L 2. Some labels are given below the ladder diagram. From left to right, source, N 7 : 5, 0 1 0 1 0 1 0 1 0 1 0 1 1 1 1 1 (highlighted in blue) are shown along the first line. From left to right, mask, N 7 : 6, 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 (highlighted in green) are shown along the second line. From left to right, compare, N 7 : 1 0, 0 1 0 1 0 1 0 1 0 1 0 1 x x x x (highlighted in red) are shown along the third line. Last four bits in all the three lines are shown in red.

Return to Figure 10-31

OceanofPDF.com

Figure 10-32 Extended Description (Chapter 10)

Return to Figure 10-32

There are two vertical lines. From left to right, the vertical lines are labeled as L 1, and L 2. There are four horizontal lines connected between the lines L 1 and L 2. From left to right, N C pushbutton labeled as stop, N O pushbutton labeled as start, solenoid labeled as SOL A are connected in series along the first horizontal line. An N O relay contact labeled as C R is connected across the N O pushbutton labeled as start. A relay coil C R is connected across the solenoid labeled as SOL A, from the top. A relay coil 1 T D is connected across the solenoid labeled as SOL A, from the bottom. From left to right, N O time-delay relay contact labeled as 1 T D (5 s), and solenoid labeled as SOL B are connected in series along the second horizontal line. A relay coil labeled as 2 T D is connected across the solenoid labeled as SOL B. From left to right, N O time-delay relay contact labeled as 2 T D (5 s), and solenoid labeled as SOL C are connected in series along the third horizontal line. A relay coil labeled as 3 T D is connected across the solenoid labeled as SOL C. From left to right, N O time-delay relay contact labeled as 3 T D (5 s), and solenoid labeled as SOL D are connected in series along the fourth horizontal line.

Return to Figure 10-32

OceanofPDF.com

Figure 10-33 Extended Description (Chapter 10)

Return to Figure 10-33

There are four vertical lines. From left, the first line is labeled as L 1 and the fourth line is labeled as L 2. Two horizontal lines are connected to the right side of the vertical line L 1. A label inputs is placed over the horizontal line connected to the line L 1. From top to bottom, N C pushbutton labeled as stop, and N O pushbutton labeled as start are connected along the horizontal lines connected to L 1. An illustration of an actual N O and N C pushbuttons is placed below the second horizontal line. Five horizontal lines are connected between the second and the third horizontal line. A label ladder logic program is placed over the first horizontal line connected between the second and the third vertical line. From left to right, X I C open symbol (highlighted in green) labeled as stop, X I C open symbol labeled as start, and an output energize labeled as SOL A are connected in series along the first horizontal line. From left to right, X I C open symbol labeled as SOL A, T O N instruction, and an output energize E N are connected in series along the second horizontal line. The right side of the T O N is connected to an output energize D N. From top to bottom, T O N instruction is labeled as T O N, timer on delay, timer T 4 : 1, time base 1.0, preset 15, and accumulated 0. From left to right, G E Q instruction, and an output energize labeled as SOL B are connected in series along the third horizontal line. From top to bottom, the G E Q instruction is labeled as G E Q, greater than or equal, source A T 4 : 1 dot A C C, 0 (at the right side), and source B 5. From left to right, the second G E Q instruction, and an output energize labeled as SOL C are connected in series along the fourth horizontal line. From top to bottom, the G E Q instruction is labeled as G E Q, greater than or equal, source A T 4 : 1 dot A C C, 0 (at the right side), and source B 10. From left to right, X I C open symbol labeled as T 4 : 1 at the top and D N at the bottom, and an output energize labeled as SOL D are connected in series along the fifth horizontal line. Four horizontal lines are connected at the left side of the line L 2. From top to bottom, solenoid valves labeled as A, B, C, and D are connected along the horizontal lines connected to the line L 2. A label outputs is placed over the first horizontal line connected to L 2. An illustration of 4 actual solenoid valves attached to each other is placed below the fourth horizontal line. The left side of the illustration has labels A, B, C, and D, from top to bottom.

Return to Figure 10-33

OceanofPDF.com

Figure 10-34 Extended Description (Chapter 10)

Return to Figure 10-34

There are four vertical lines. From left, the first line is labeled as L 1 and the fourth line is labeled as L 2. A horizontal line is connected to the right side of the vertical line L 1. A label input is placed over the horizontal line connected to the line L 1. The N O switch labeled as S 1 is connected along the horizontal line connected to L 1. An illustration of an actual switch is placed below its symbol. Three horizontal lines connected between the second and the third horizontal line are labeled as one, two, and three. A label ladder logic program is placed over the first horizontal line connected between the second and the third vertical line. From left to right, X I C open symbol labeled as S 1, T O N instruction, and an output energize E N are connected in series along the first horizontal line labeled as one. The right side of the T O N is connected to an output energize D N. From top to bottom, T O N instruction is labeled as T O N, timer on delay, timer T 4 : 1, time base 1.0, preset 20, and accumulated 0. From left to right, E Q U instruction, and an output energize L labeled as P L 1 are connected in series along the second horizontal line labeled as two. From top to bottom, the E Q U instruction is labeled as E Q U, equal, source T 4 : 1 dot A C C, and source B 5. From left to right, the second E Q U instruction, and an output energize U labeled as P L 1 are connected in series along the third horizontal line labeled as three. From top to bottom, the E Q U instruction is labeled as E Q U, equal, source T 4 : 1 dot A C C, and source B 15. The pilot light labeled as P L 1 is connected along a horizontal line connected at the left side of the vertical line L 2. A label output is placed over the horizontal line connected to L 2. An illustration of an actual pilot light is placed below its symbol.

Return to Figure 10-34

OceanofPDF.com

Figure 10-35 Extended Description (Chapter 10)

Return to Figure 10-35

There are four vertical lines. From left, the first line is labeled as L 1 and the fourth line is labeled as L 2. A horizontal line is connected to the right side of the vertical line L 1. A label input is placed over the horizontal line connected to the line L 1. The N O proximity switch labeled as sensor is connected along the horizontal line connected to L 1. There are three horizontal lines connected between the second and the third horizontal line. A label ladder logic program is placed over the first horizontal line connected between the second and the third vertical line. From left to right, X I C open symbol labeled as sensor, C T U instruction, and an output energize C U are connected in series along the first horizontal line. The right side of the C T U is connected to an output energize D N. From top to bottom, C T U instruction is labeled as C T U, count up, counter C 5 : 1, preset 50, and accumulated 0. From left to right, L E S instruction, and an output energize labeled as S O L are connected in series along the second horizontal line. From top to bottom, the L E S instruction is labeled as L E S, less than, source A C 5 : 1 dot A C C, 0 (at the right side), and source B 20. From left to right, X I C open symbol labeled as C 5 : 1 at the top and D N at the bottom, and an output energize R E S labeled as C 5 : 1 are connected in series along the third horizontal line. A solenoid valve labeled as S O L is connected along a horizontal line connected at the left side of the vertical line L 2. A label output is placed over the horizontal line connected to L 2. An illustration of solenoid valve is placed below its symbol.

Return to Figure 10-35

OceanofPDF.com

Figure 10-36 Extended Description (Chapter 10)

Return to Figure 10-36

From top to bottom the blocks enclose the number five, six, seven, zero, separately. A label decimal is placed at the top left side of the block enclosing number five. From top to bottom, the blocks are labeled from the left as one s units, ten s units, 100s units, and 1000 s units. All the blocks are connected in series and the bottom of the block enclosing the number zero is connected to a line L 1. The B C D input module has four blocks below the label bit address, where each block has four terminals and a separate terminal is shown at the bottom. From top to bottom, the terminals are labeled from zero through 15, except for the last terminal. The connection between the blocks enclosing number five and the B C D input module has a label B C D over it. Four terminals from the block enclosing number five are connected to the input module terminals labeled from zero through three. From top to bottom, the connection is labeled as one, zero, one, and one. Four terminals from the block enclosing number six are connected to the input module terminals labeled from four through seven. From top to bottom, the connection is labeled as zero, one, one, and zero. Four terminals from the block enclosing number seven are connected to the input module terminals labeled from eight through 11. From top to bottom, the connection is labeled as one, one, one, and zero. Four terminals from the block enclosing number zero are connected to the input module terminals labeled from 12 through 15. From top to bottom, the connection is labeled as zero, zero, zero, and zero. The last terminal is connected to a line L 2.

Return to Figure 10-36

OceanofPDF.com

Figure 10-37 Extended Description (Chapter 10)

Return to Figure 10-37

From top to bottom, the blocks enclose the number eight, four, three, and two, separately. A label decimal is placed at the top left side of the block enclosing number eight. From top to bottom, the blocks are labeled from the left as one s units, ten s units, 100s units, and 1000 s units. All the blocks are connected in series and the bottom of the block enclosing the number two is connected to a line L 1. The B C D output module has four blocks below the label bit address, where each block has four terminals and a separate terminal is shown at the bottom. From top to bottom, the terminals are labeled from zero through 15, except for the last terminal. The connection between the blocks enclosing number eight and the B C D output module has a label B C D over it. Four terminals from the block enclosing number eight are connected to the output module terminals labeled from zero through three. From top to bottom, the connection is labeled as zero, zero, zero, and one. Four terminals from the block enclosing number four are connected to the output module terminals labeled from four through seven. From top to bottom, the connection is labeled as zero, zero, one, and zero. Four terminals from the block enclosing number three are connected to the output module terminals labeled from eight through 11. From top to bottom, the connection is labeled as one, one, zero, and zero. Four terminals from the block enclosing number two are connected to the output module terminals labeled from 12 through 15. From top to bottom, the connection is labeled as zero, one, zero, and zero. The last terminal is connected to a line L 2.

Return to Figure 10-37

OceanofPDF.com

The Binary Number Graphic Extended Description (Chapter 10)

Return to The Binary Number Graphic

From left to right, there is eight above zero, four above one, two above one, one above one, eight above one, four above zero, two above zero, and one above one. At the right side of the labels, symbol of equal to 79 is shown.

Return to The Binary Number Graphic

OceanofPDF.com

The Binary Number Graphic Extended Description (Chapter 10)

Return to The Binary Number Graphic

From left to right, there is 64 over the second block, 32 over the third block, 16 over fourth block, eight over fifth block, and one over eighth block. At the right side of the labels, symbol of equal to 121 is shown.

Return to The Binary Number Graphic

OceanofPDF.com

Figure 10-38 Extended Description (Chapter 10)

Return to Figure 10-38

There are four vertical lines. From left, the first line is labeled as L 1 and the fourth line is labeled as L 2. Two horizontal lines are connected to the right side of the vertical line L 1. A label inputs is placed over the first horizontal line connected to the line L 1. From top to bottom, the N O switch labeled S 1, and thumbwheel switch labeled as T W S are connected along the first and the second horizontal line, respectively. At the bottom of the symbol of thumbwheel switch, an illustration of actual thumbwheel switch is shown. From left to right, the blocks in the thumbwheel switch are labeled as zero, zero, seven, and nine. There are two horizontal lines connected between the second and the third vertical line. The M O V instruction is connected along the first horizontal line. From top to bottom, the M O V instruction is labeled as M O V, move, source T W S, 121 (at the right side), dest L E D, and 121 (at the right side). The right side of the F R D instruction, which is placed below the M O V instruction, is connected to the third vertical line. From top to bottom, the F R D instruction is labeled as F R D, from B C D, source T W S, 121 (at the right side), dest integer, and 79 (at the right side). From left to right, X I C open symbol (highlighted in green) labeled as S 1, E Q U instruction, and an output energize (highlighted in green) labeled as P L are connected in series along the second horizontal line. From top to bottom, the E Q U instruction is labeled as E Q U, equal, source A integer, 79 (at the right side), and source B 79. A label ladder logic program is placed over the horizontal line connected between the second and the third vertical line. Two horizontal lines are connected at the left side of the vertical line L 2. From top to bottom, pilot light labeled as P L 1, and L E D display labeled as L E D is connected along the first and the second horizontal line, respectively. From top to bottom, the L E D display is labeled as zero, zero, zero, and zero. An illustration of actual L E D display is placed below the symbol of L E D display. From left to right, the blocks in the actual L E D display are labeled as zero, zero, seven, and nine. A label output is placed over the horizontal line connected to the vertical line L 2.

Return to Figure 10-38

OceanofPDF.com

Figure 10-39 Extended Description (Chapter 10)

Return to Figure 10-39

The right side of the block is connected to 15 terminals labeled as binary output. A line graph is shown at the right side of the illustration. A line for analog input shows a curve bulging upwards. A line for digital output shows a staircase wave, such that the left end of each step is connected to the curve labeled analog input.

Return to Figure 10-39

OceanofPDF.com

Figure 10-40 Extended Description (Chapter 10)

Return to Figure 10-40

From left to right, blocks labeled digital to analog converter, isolation, amplifier, and voltage and current driver are connected. An arrow from the voltage and current driver points at the left side of the servo motor labeled as analog output. The servo motor is connected to a block labeled load, which is placed at the right side.

Return to Figure 10-40

OceanofPDF.com

Figure 10-41 Extended Description (Chapter 10)

Return to Figure 10-41

There are four vertical lines. From left, the first line is labeled as L 1 and the fourth line is labeled as L 2. Two horizontal lines are connected to the right side of the vertical line L 1. A label inputs is placed over the first horizontal line connected to the line L 1. From top to bottom, the NC switch labeled S 1, and thermocouple are connected along the first and the second horizontal line, respectively. At the bottom of the symbol of thermocouple, an illustration of actual thermocouple is shown. There are four horizontal lines connected between the second and the third vertical line. From left to right, X I C open symbol (highlighted in green) labeled as S 1, and M O V instruction is connected along the first horizontal line. From top to bottom, the M O V instruction is labeled as M O V, move, source thermocouple, and, destination L E D. From left to right, L E Q instruction, and output energize (highlighted in green) labeled as B 3 : 0/1 at the top and (internal) at the bottom are connected in series along the second horizontal line. From top to bottom, the L E Q instruction is labeled as L E Q, less than or equal, source A thermocouple, and source B 597. From left to right, G E Q instruction, and an output energize labeled as B 3 : 0/2 at the top and (internal) at the bottom are connected in series along the third horizontal line. From top to bottom, the G E Q instruction is labeled as G E Q, greater than or equal, source A thermocouple, 79, and source B 603. From left to right, the second X I C open symbol (highlighted in green) labeled as S1, X I O open symbol (highlighted in green) labeled as high temperature B 3 : 0/2, X I C open symbol (highlighted in green) labeled as low temperature B 3 : 0/1, and an output energize (highlighted in green) labeled as heater are connected along the fourth horizontal line. An X I C open symbol (highlighted in green) labeled as heater is parallelly connected across the X I C open symbol (highlighted in green) labeled as low temperature B 3 : 0/1. A label ladder logic program is placed over the horizontal line connected between the second and the third vertical line. Two horizontal lines are connected at the left side of the vertical line L 2. From top to bottom, a rectangle with a terminal at each end along a wire wounded over the rectangle labeled as heater, and L E D display labeled as L E D is connected along the first and the second horizontal line, respectively. From top to bottom, the L E D display is labeled as eight, eight, eight, and eight. An illustration of actual L E D display is placed below the symbol of L E D display. From left to right, the blocks in the actual L E D display are labeled as five, nine, and five. A label output is placed over the horizontal line connected to the vertical line L 2.

Return to Figure 10-41

OceanofPDF.com

Figure 10-42 Extended Description (Chapter 10)

Return to Figure 10-42

An arrow from the right side of the P L C controller points at a block labeled as control element (heater-valve). An arrow from a block labeled as measurement of variable to be controlled (sensor) points at the left side of the P L C controller. Illustrations of actual potentiometer, sensor, and heater-valve are shown around the labels.

Return to Figure 10-42

OceanofPDF.com

Figure 10-43 Extended Description (Chapter 10)

Return to Figure 10-43

The right side of the adjustable valve is connected to the holding tank. An arrow pointing left is labeled analog output and is placed over the connection of the top sides of the adjustable valve, and the P L C module. An arrow-head pointing right is labeled four to 20 milli-amperes analog input and is labeled below the connection of the ultrasonic level sensor and the bottom side of the P L C module. Waves from the ultrasonic level sensor is extending towards the fluid inside the holding tank.

Return to Figure 10-43

OceanofPDF.com

Figure 10-44 Extended Description (Chapter 10)

Return to Figure 10-44

From left, an arrow points from a label set-point (S P) towards the left side of the circle enclosing symbol of summation. An arrow labeled error points from the right side of the circle enclosing summation towards the left side of a block labeled PI D equation. The right side of the P I D equation points at the top side of the control valve, through an arrow labeled control variable (C V). An arrow points away from the right side of the control valve. The bottom of the holding tank is connected to the left side of the control valve. An arrow points from a label flow rate towards the top side of the fluid. An arrow from the level detector placed over the holding tank points at the bottom side of the circle enclosing the summation, through an arrow labeled process variable (P V).

Return to Figure 10-44

OceanofPDF.com

Figure 10-45 Extended Description (Chapter 10)

Return to Figure 10-45

From top to bottom, P I D instruction is labeled as P I D, P I D, control block N 1 0 : 0, process variable N 1 0 : 2 8, control variable N 1 0 : 2 9, control block length 23, and setup screen.

Return to Figure 10-45

OceanofPDF.com

Figure 10-46(a-c) Extended Description (Chapter 10)

Return to Figure 10-46(a-c)

In all the parts, there are four vertical lines. From left, the first line is labeled as L 1 and the fourth line is labeled as L 2. A horizontal line is connected to the right side of the vertical line L 1. A label input is placed over the first horizontal line connected to the line L 1. In part a, the N O switch labeled A is connected along the horizontal line connected to the line L 1. From left to right, X I C open symbol labeled as A, E Q U instruction, and an output energize labeled as B are connected in series along the horizontal line connected between the second and the third vertical line. From top to bottom, the E Q U instruction is labeled as E Q U, equal, source A N 7 : 4 0 0, and source B N 7 : 4 0 1. A label ladder logic program is placed over the horizontal line connected between the second and the third vertical line. The pilot light labeled as B is connected along the horizontal line connected at the left side of the vertical line L 2. A label output is placed over the horizontal line connected to the vertical line L 2. In part b, the N O switch labeled C is connected along the horizontal line connected to the line L 1. From left to right, X I C open symbol labeled as C, L E S instruction, and an output energize labeled as D are connected in series along the horizontal line connected between the second and the third vertical line. From top to bottom, the L E S instruction is labeled as L E S, less than, source A N 7 : 5 0 0, and source B 250. A label ladder logic program is placed over the horizontal line connected between the second and the third vertical line. The pilot light labeled as D is connected along the horizontal line connected at the left side of the vertical line L 2. A label output is placed over the horizontal line connected to the vertical line L 2. In part c, the N O switch labeled E is connected along the horizontal line connected to the line L 1. From left to right, X I C open symbol labeled as E, G E Q instruction, and an output energize labeled as F are connected in series along the horizontal line connected between the second and the third vertical line. From top to bottom, the G E Q instruction is labeled as G E Q, greater than or equal, source A N 7 : 6 0 1, source B N 7 : 6 0 0, and 30 (on the right side). A label ladder logic program is placed over the horizontal line connected between the second and the third vertical line. The pilot light labeled as F is connected along the horizontal line connected at the left side of the vertical line L 2. A label output is placed over the horizontal line connected to the vertical line L 2.

Return to Figure 10-46(a-c)

OceanofPDF.com

Figure 10-47 Extended Description (Chapter 10)

Return to Figure 10-47

There are four vertical lines. From left, the first line is labeled as L 1 and the fourth line is labeled as L 2. Two horizontal lines are connected to the right side of the vertical line L 1. A label inputs is placed over the first horizontal line connected to the line L 1. From top to bottom, the N O switch labeled S 1, and thumbwheel switch labeled as N 7 : 1 1 2 are connected along the first and the second horizontal line, respectively. From top to bottom, the blocks in the thumbwheel switch are labeled as four, eight, and five. From left to right, X I C open symbol labeled as S 1, and M O V instruction are connected in series along the horizontal line connected between the second and the third vertical line. A label ladder logic program is placed over the horizontal line connected between the second and the third vertical line. From top to bottom, the M O V instruction is labeled as M O V, move, source N 7 : 1 1 2, and destination N 7 : 1 3. The L E D display labeled as N 7 : 1 3 is connected along the horizontal line connected at the left side of the vertical line L 2. From top to bottom, the blocks in the L E D display are labeled as zero, zero, and zero. A label output is placed over the horizontal line connected to the vertical line L 2.

Return to Figure 10-47

OceanofPDF.com

Figure 10-48 Extended Description (Chapter 10)

Return to Figure 10-48

There are four vertical lines. From left, the first line is labeled as L 1 and the fourth line is labeled as L 2. Three horizontal lines are connected to the right side of the vertical line L 1. A label inputs is placed over the first horizontal line connected to the line L 1. From top to bottom, the N O pushbutton labeled count P B, N O pushbutton reset P B, and thumbwheel switch labeled as N 7 : 1 0 are connected along the horizontal lines. There are four horizontal lines connected between the second and the third vertical line. From left to right, X I C open symbol labeled as count P B, C T U instruction, and an output energize C U are connected in series along the first horizontal line. The right side of the C T U is connected to an output energize D N. From top to bottom, the C T U instruction is labeled as C T U, count up, counter C 5 : 1, preset 0 0 0, and accumulated 0 0 0. M O V instruction is connected along the second horizontal line. From top to bottom, the M O V instruction is labeled as M O V, move, source N 7 : 1 0, and destination C 5 : 1 dot P R E. From left to right, an X I C open symbol labeled as C 5 : 1 / D N, and an output energize labeled as P L 1 are connected in series along the third horizontal line. From left to right, an X I C open symbol labeled as reset P B, and an output energize R E S labeled as C 5 : 1 are connected in series along the fourth horizontal line. A label ladder logic program is placed over the first horizontal line connected between the second and the third vertical line. The pilot light labeled as P L 1 is connected along the horizontal line connected at the left side of the vertical line L 2. A label output is placed over the horizontal line connected to the vertical line L 2.

Return to Figure 10-48

OceanofPDF.com

Figure 10-49 Extended Description (Chapter 10)

Return to Figure 10-49

There are four vertical lines. From left, the first line is labeled as L 1 and the fourth line is labeled as L 2. Two horizontal lines are connected to the right side of the vertical line L 1. A label inputs is placed over the first horizontal line connected to the line L 1. From top to bottom, the N O switch labeled S 1, and thumbwheel switch labeled as N 7 : 1 1 2 are connected along the first and the second horizontal line, respectively. From left to right, X I C open symbol labeled as S 1, E Q U instruction, and an output energize labeled as P L 1 are connected in series along the horizontal line connected between the second and the third vertical line. From top to bottom, the E Q U instruction is labeled as E Q U, equal, source A N 7 : 1 1 2, and source B 0 0 4. A label ladder logic program is placed over the horizontal line connected between the second and the third vertical line. The pilot light labeled as P L 1 is connected along the horizontal line connected at the left side of the vertical line L 2. A label output is placed over the horizontal line connected to the vertical line L 2.

Return to Figure 10-49

OceanofPDF.com

Figure 10-50 Extended Description (Chapter 10)

Return to Figure 10-50

There are four vertical lines. From left, the first line is labeled as L 1 and the fourth line is labeled as L 2. Two horizontal lines are connected to the right side of the vertical line L 1. A label inputs is placed over the first horizontal line connected to the line L 1. From top to bottom, the N O switch labeled S 1, and thumbwheel switch labeled as N 7 : 1 1 2 are connected along the first and the second horizontal line, respectively. From left to right, X I C open symbol labeled as S 1, G R T instruction, and an output energize labeled as P L 1 are connected in series along the horizontal line connected between the second and the third vertical line. A label ladder logic program is placed over the horizontal line connected between the second and the third vertical line. From top to bottom, the G R T instruction is labeled as G R T, greater than, source A N 7 : 1 1 2, and source B 12. The pilot light labeled as P L 1 is connected along the horizontal line connected at the left side of the vertical line L 2. A label output is placed over the horizontal line connected to the vertical line L 2.

Return to Figure 10-50

OceanofPDF.com

A Ladder Diagram Graphic Extended Description (Chapter 11)

Return to A Ladder Diagram Graphic

The ladder diagram consists of four vertical lines. From the left, the first vertical line is labeled as L 1 and the second vertical line is labeled as L 2. The N C switch labeled as S W is connected along a horizontal line that is connected at the right side of the vertical line L 1. A label input is placed over the horizontal line connected to the vertical line L 1. An illustration of actual switch is placed below its symbol. Two horizontal lines are connected between the second and the third vertical line. A label ladder logic program is placed over the first horizontal line connected between the second and the third vertical line. From left to right, X I C open symbol (highlighted in green) labeled as S W, and M U L instruction are connected in series along the first horizontal line. From top to bottom, M U L instruction is labeled as M U L, multiply, source A N 7 : 1, 123 (highlighted in blue and placed at the right side), source B N 7 : 2, 61 (highlighted in red and placed at the right side), destination N 7 : 3, and 7503 (highlighted in red and placed at the right side). From left to right, E Q U instruction, and output energize (highlighted in green) labeled as P L 1 are connected in series along the second horizontal line. From top to bottom, E Q U instruction is labeled as E Q U, equal, source A N 7 : 3, 7503 (highlighted in red and placed at the right side), and source B 7503. The pilot light labeled as P L 1is connected along a horizontal line that is connected to the left side of the vertical line L 2. An illustration of pilot light is placed below its symbol. A label output is placed over the horizontal line connected to the line L 2.

Return to A Ladder Diagram Graphic

OceanofPDF.com

Figure 11-1 Extended Description (Chapter 11)

Return to Figure 11-1

A P L C programming menu interface. It features a horizontal row of nine rectangular blocks, each with uppercase, spaced letters. From left to right, the labels are: C P T (Compute), A D D (Add), S U B (Subtract), M U L (Multiply), D I V (Divide), S Q R (Square Root), N E G (Negate), T O D (Time to Day), F R D (Floating Point to Real Data).

Return to Figure 11-1

OceanofPDF.com

Figure 11-3 Extended Description (Chapter 11)

Return to Figure 11-3

The ladder diagram consists of three vertical lines. From the left, the first line is labeled as L 1. The N C switch labeled as temp_convert is connected along a horizontal line connected at the right side of the line L 1. A label input is placed over the horizontal line connected to the L 1. From left to right, X I C open symbol (highlighted in green) labeled as temp_convert, and C P T instruction are connected in series along a horizontal line connected between the second and the third vertical line. A label ladder logic program is placed over the horizontal line connected between the second and the third vertical line. From top to bottom, C P T instruction is labeled as C P T, compute, dest result, 60 (highlighted in red and placed at the right side)., and expression (N 7 : 5-32)*5/9. An illustration of switch is placed below its symbol.

Return to Figure 11-3

OceanofPDF.com

Figure 11-4 Extended Description (Chapter 11)

Return to Figure 11-4

The ladder diagram consists of three vertical lines. From the left, the first line is labeled as L 1. The N C switch labeled as S W is connected along a horizontal line connected at the right side of the line L 1. A label input is placed over the horizontal line connected to the L 1. From left to right, X I C open symbol (highlighted in green) labeled as S W, and A D D instruction are connected in series along a horizontal line connected between the second and the third vertical line. A label ladder logic program is placed over the horizontal line connected between the second and the third vertical line. From top to bottom, A D D instruction is labeled as A D D, add, source A N 7 : 0 (both the labels highlighted in blue), 25 (at the right side), source B N 7 : 1 (both the labels highlighted in red), 50 (at the right side), destination N 7 : 2 (both the labels highlighted in red), and 75 (at the right side). An illustration of switch is placed below its symbol.

Return to Figure 11-4

OceanofPDF.com

Figure 11-5 Extended Description (Chapter 11)

Return to Figure 11-5

The ladder diagram consists of four vertical lines. From the left, the first line is labeled as L 1, and the fourth line is labeled as L 2. Three horizontal lines are connected at the right side of the line L 1. From top to bottom, N O proximity switch labeled as S 1, N O proximity switch labeled as S 2, and N O pushbutton labeled as reset are connected along the horizontal lines. A label inputs is placed over the first horizontal line connected to the line L 1. A label ladder logic program is placed over the five horizontal lines, which are connected between the second and the third vertical line. From left to right, X I C open symbol labeled as S 1, C T U instruction, and output energize C U are connected in series along the first horizontal line. The right side of the C T U instruction is connected to an output energize D N. From top to bottom, C T U is labeled as C T U, count up counter, counter C 5 : 0, preset 350, and accumulated zero. From left to right, X I C open symbol labeled as S 2, C T U instruction, and output energize C U are connected in series along the second horizontal line. The right side of the C T U instruction is connected to an output energize D N. From top to bottom, C T U is labeled as C T U, count up counter, counter C 5 : 1, preset 350, and accumulated zero. The A D D instruction is connected along the third horizontal line. From top to bottom, A D D instruction is labeled as A D D, add, source A C 5 : 0 dot A C C, source B C 5 : 1 dot A C C, and destination N 7 : 1. From left to right, G E Q instruction, and output energize labeled as P L 1 are connected in series along the fourth horizontal line. From top to bottom, G E Q instruction is labeled as G E Q, greater than or equal, source N 7 : 1, and source B 350. From left to right, X I C open symbol labeled as reset, and output energize R E S labeled as C 5 : 0 are connected in series along the fifth horizontal line. An output energize R E S labeled as C 5 : 1 is connected across the output energize R E S labeled as C 5 : 0. A label output is placed over the horizontal line connected at the left side of the line L 2. A pilot light labeled as P L 1 is connected along the horizontal line connected to the line L 2. Illustration of the proximity switches, pushbutton, and the pilot light are placed below their symbols. L E D display labeled as 125 is placed at the right side of the first C T U instruction. L E D display labeled as 100 is placed at the right side of the second C T U instruction. L E D display labeled as 250 is placed at the right side of the A D D instruction.

Return to Figure 11-5

OceanofPDF.com

Figure 11-6 Extended Description (Chapter 11)

Return to Figure 11-6

From left to right, the first cell of the first row is unlabeled, whereas remaining cells of the first row are labeled as 15, 14, 13, 12, 11, 10, nine, eight, seven, six, five, four, three, two, one, and zero. From top to bottom, except for the first cell, the other cells in the first column are labeled as S 2 : 0/ (highlighted), S 2 : 1/, S 2 : 2/, S 2 : 3/, S 2 : 4/, and S 2 : 5. The third, ninth, tenth, and seventeenth columns of sixth row are labeled as one. The remaining cells are labeled as zero. The last four columns of the first two rows are highlighted. From left to right, the labels at the bottom are given as, address: S 2 : 0 (enclosed in a rectangle), and table: S 2 : status (enclosed in a rectangle with downwards pointing arrow-head at the right side).

Return to Figure 11-6

OceanofPDF.com

Figure 11-7 Extended Description (Chapter 11)

Return to Figure 11-7

The ladder diagram consists of three vertical lines. From the left, the first line is labeled as L 1. The N C switch labeled as S W is connected along a horizontal line connected at the right side of the line L 1. A label input is placed over the horizontal line connected to the L 1. From left to right, X I C open symbol (highlighted in green) labeled as S W, and S U B instruction are connected in series along a horizontal line connected between the second and the third vertical line. A label ladder logic program is placed over the horizontal line connected between the second and the third vertical line. From top to bottom, S U B instruction is labeled as S U B, subtract, source A N 7 : 1 0 (both the labels highlighted in blue), 520 (at the right side), source B N 7 : 0 5 (both the labels highlighted in red), 322 (at the right side), destination N 7 : 2 0 (both the labels highlighted in red), and 198 (at the right side). An illustration of switch is placed below its symbol.

Return to Figure 11-7

OceanofPDF.com

Figure 11-8 Extended Description (Chapter 11)

Return to Figure 11-8

The ladder diagram consists of four vertical lines. From the left, the first line is labeled as L 1, and the fourth line is labeled as L 2. Three horizontal lines are connected at the right side of the line L 1. From top to bottom, N O pushbutton labeled as start, N C pushbutton labeled as stop, and transducer labeled as weight transducer (all 16 bits) at the bottom and I : 0 1 2 at the right side are connected along the horizontal lines. A label inputs is placed over the first horizontal line connected to the line L 1. A label ladder logic program is placed over the five horizontal lines, which are connected between the second and the third vertical line. From left to right, X I C open symbol (highlighted in green) labeled as stop, X I C open symbol labeled as start, X I O open symbol (highlighted in green) labeled as full, and output energize labeled as fill solenoid are connected in series along the first horizontal line labeled as one. An X I C open symbol labeled as fill solenoid is connected across the X I C open symbol labeled as start. From left to right, X I C open symbol labeled as fill solenoid, X I O open symbol (highlighted in green) labeled as full, and output energize labeled as filling are connected in series along the second horizontal line labeled as two. From left to right, G E Q instruction, and output energize labeled as full are connected in series along the third horizontal line labeled as three. From top to bottom, G E Q instruction is labeled as G E Q, greater than or equal, source A I : 0 1 2 (highlighted in yellow), and source B 500. From left to right, X I C open symbol labeled as full, and S U B instruction are connected in series along the fourth horizontal line labeled as four. From top to bottom, S U B instruction is labeled as S U B, subtract, source A I : 0 1 2 (highlighted in yellow), source B 500, and destination N 7 : 1. From left to right, G E Q instruction, and output energize labeled as alarm are connected in series along the fifth horizontal line labeled as five. From top to bottom, G E Q instruction is labeled as G E Q, greater than or equal, source N 7 : 1, and source B 5. A label outputs is placed over the four horizontal lines connected at the left side of the line L 2. From top to bottom, solenoid labeled as fill solenoid, pilot light labeled filling, pilot light labeled full, and alarm are connected along the horizontal lines connected to the line L 2. Illustration of start and stop pushbuttons, weight transducer, solenoid, pilot light, and alarm are shown below their symbols.

Return to Figure 11-8

OceanofPDF.com

Figure 11-9 Extended Description (Chapter 11)

Return to Figure 11-9

The ladder diagram consists of three vertical lines. From the left, the first line is labeled as L 1. The N C switch labeled as S W is connected along a horizontal line connected at the right side of the line L 1. A label input is placed over the horizontal line connected to the L 1. From left to right, X I C open symbol (highlighted in green) labeled as S W, and M U L instruction are connected in series along a horizontal line connected between the second and the third vertical line. A label ladder logic program is placed over the horizontal line connected between the second and the third vertical line. From top to bottom, M U L instruction is labeled as M U L, multiply, source A 20 (both the labels highlighted in blue), source B C 5 : 1 0 dot A C C (both the labels highlighted in red), and destination N 7 : 2 (both the labels highlighted in red). An illustration of switch is placed below its symbol.

Return to Figure 11-9

OceanofPDF.com

Figure 11-10 Extended Description (Chapter 11)

Return to Figure 11-10

The ladder diagram consists of four vertical lines. From the left, the first vertical line is labeled as L 1 and the second vertical line is labeled as L 2. The N C switch labeled as S W is connected along a horizontal line that is connected at the right side of the vertical line L 1. A label input is placed over the horizontal line connected to the vertical line L 1. An illustration of actual switch is placed below its symbol. Two horizontal lines are connected between the second and the third vertical line. A label ladder logic program is placed over the first horizontal line connected between the second and the third vertical line. From left to right, X I C open symbol (highlighted in green) labeled as S W, and M U L instruction are connected in series along the first horizontal line. From top to bottom, M U L instruction is labeled as M U L, multiply, source A N 7 : 1, 123 (highlighted in blue and placed at the right side), source B N 7 : 2, 61 (highlighted in red and placed at the right side), destination N 7 : 3, and 7503 (highlighted in red and placed at the right side). From left to right, E Q U instruction, and output energize (highlighted in green) labeled as P L 1 are connected in series along the second horizontal line. From top to bottom, E Q U instruction is labeled as E Q U, equal, source A N 7 : 3, 7503 (highlighted in red and placed at the right side), and source B 7503. The pilot light labeled as P L 1 is connected along a horizontal line that is connected to the left side of the vertical line L 2. An illustration of pilot light is placed below its symbol. A label output is placed over the horizontal line connected to the line L 2.

Return to Figure 11-10

OceanofPDF.com

Figure 11-11 Extended Description (Chapter 11)

Return to Figure 11-11

The ladder diagram consists of four vertical lines. From the left, the first line is labeled as L 1, and the fourth line is labeled as L 2. Three horizontal lines are connected at the right side of the line L 1. From top to bottom, N O switch labeled as on or off, thumbwheel switch labeled as T W S at the top, and I : 0 1 2 at the right side, and thermocouple labeled as thermocouple input at the bottom and I : 0 1 3 within the symbol are connected along the horizontal lines. A label inputs is placed over the first horizontal line connected to the line L 1. A label ladder logic program is placed over the six horizontal lines, which are connected between the second and the third vertical line. From top to bottom, M U L, A D D, and S U B instruction are connected along the first, the second, and the third horizontal lines, respectively. From top to bottom, M U L instruction is labeled as M U L, multiply, source A I : 0 1 2 (highlighted in yellow), 400 (at the right side), source B 0 point 0 1 0 0 0 0 0, destination N 7 : 0, and 4 (at the right side). From top to bottom, A D D instruction is labeled as A D D, add, source A I : 0 1 2 (highlighted in yellow), 400 (at the right side), source B N 7 : 0, 4 (at the right side), destination N 7 : 1, and 404 (at the right side). From top to bottom, S U B instruction is labeled as S U B, subtract, source A I : 0 1 2 (highlighted in yellow), 400 (at the right side), source B N 7 : 0, 4 (at the right side), destination N 7 : 2, and 396 (at the right side). From left to right, L E S instruction, and output energize (highlighted in green) labeled as P L 1 are connected along the fourth horizontal line. From top to bottom, the L E S instruction is labeled as L E S, less than, source A I : 0 1 3 (highlighted in yellow), 0 (at the right side), source B N 7 : 2, and 396 (at the right side). From left to right, G R T instruction, and output energize labeled as P L 2 are connected along the fifth horizontal line. From top to bottom, the G R T instruction is labeled as G R T, greater than, source A I : 0 1 3 (highlighted in yellow), 0 (at the right side), source B N 7 : 1, and 404 (at the right side). From left to right, XI C open symbol labeled as on or off, X I C open symbol (highlighted in green) labeled as P L 1, X I O open symbol (highlighted in green) labeled as P L 2, and an output energize labeled as heater are connected in series along the sixth horizontal line. An X I C open symbol labeled as heater is connected across the X I C open symbol (highlighted in green) labeled as P L 1. A label outputs is placed over the three horizontal lines connected at the left side of the line L 2. From top to bottom, heater, pilot light labeled P L 1 (shaded blue), and pilot light labeled P L 2 are connected along the horizontal lines connected to the line L 2. An illustration of thumbwheel switch, thermocouple, and pilot light P L 1 are placed below their symbols. From left to right, the blocks in the illustration of thumbwheel switch are labeled as four, zero, and zero. An illustration of thermometer labeled as high limit is placed at the left side of the A D D instruction. An illustration of thermometer labeled as low limit is placed at the left side of the S U B instruction.

Return to Figure 11-11

OceanofPDF.com

Figure 11-12 Extended Description (Chapter 11)

Return to Figure 11-12

The ladder diagram consists of three vertical lines. From the left, the first line is labeled as L 1. The N C switch labeled as S W is connected along a horizontal line connected at the right side of the line L 1. A label input is placed over the horizontal line connected to the L 1. From left to right, X I C open symbol (highlighted in green) labeled as S W, and D I V instruction are connected in series along a horizontal line connected between the second and the third vertical line. A label ladder logic program is placed over the horizontal line connected between the second and the third vertical line. From top to bottom, D I V instruction is labeled as D I V, divide, source A C 5 : 1 0 dot A C C (both the labels highlighted in blue), source B 2 (both the labels highlighted in red), and destination N 7 : 3 (both the labels highlighted in red). An illustration of switch is placed below its symbol.

Return to Figure 11-12

OceanofPDF.com

Figure 11-13 Extended Description (Chapter 11)

Return to Figure 11-13

The ladder diagram consists of four vertical lines. From the left, the first vertical line is labeled as L 1 and the second vertical line is labeled as L 2. The N C switch labeled as S W is connected along a horizontal line that is connected at the right side of the vertical line L 1. A label input is placed over the horizontal line connected to the vertical line L 1. An illustration of actual switch is placed below its symbol. Two horizontal lines are connected between the second and the third vertical line. A label ladder logic program is placed over the first horizontal line connected between the second and the third vertical line. From left to right, X I C open symbol (highlighted in green) labeled as S W, and D I V instruction are connected in series along the first horizontal line. From top to bottom, D I V instruction is labeled as D I V, divide, source A N 7 : 0, 120 (highlighted in blue and placed at the right side), source B N 7 : 1, 4 (highlighted in red and placed at the right side), destination N 7 : 5, and 30 (highlighted in red and placed at the right side). From left to right, E Q U instruction, and output energize (highlighted in green) labeled as P L 1 are connected in series along the second horizontal line. From top to bottom, E Q U instruction is labeled as E Q U, equal, source A N 7 : 5, 30 (highlighted in red and placed at the right side), and source B 30. The pilot light labeled as P L 1 is connected along a horizontal line that is connected to the left side of the vertical line L 2. An illustration of pilot light is placed below its symbol. A label output is placed over the horizontal line connected to the line L 2.

Return to Figure 11-13

OceanofPDF.com

Figure 11-14 Extended Description (Chapter 11)

Return to Figure 11-14

The ladder diagram consists of four vertical lines. From the left, the first line is labeled as L 1, and the fourth line is labeled as L 2. The thumbwheel switch is labeled as thumbwheel switch at the bottom, and I : 0 1 2 at the right side. The thumbwheel switch is connected along a horizontal line that is connected at the right side of line L 1. From top to bottom, the blocks in the thumbwheel switch are labeled as one, two, and three. An illustration of the thumbwheel switch labeled as Celsius at the bottom and is placed below its symbol. From left to right, the blocks in the illustration are labeled as zero, six, and zero. A label input is placed over the symbol of the thumbwheel switch. There are five horizontal lines connected between the second and the third vertical line. A label ladder logic program is placed over the first horizontal line connected between the second and the third vertical line. From top to bottom, F R D, M U L, D I V, A D D, and T O D instructions are connected along the horizontal lines, which are connected between the second and the third vertical line. From top to bottom, F R D instruction is labeled as F R D, from B C D, source I : 0 1 2 (highlighted in yellow), destination B 3 : 0, and 60. From top to bottom, M U L instruction is labeled as M U L, multiply, source A B 3 : 0, 60 (placed at the right side), source B 9, destination N 7 : 0, and 540 (placed at the right side). From top to bottom, D I V instruction is labeled as D I V, divide, source A N 7 : 0, 540 (placed at the right side), source B 5, destination N 7 : 1, and 108 (placed at the right side). From top to bottom, A D D instruction is labeled as A D D, add, source A N 7 : 1, 108 (placed at the right side), source B 32, destination B 3 : 1, and 140 (placed at the right side). From top to bottom, T O D instruction is labeled as T O D, to B C D, source B 3 : 1, 140 (at the right side), and destination O : 0 1 3 (highlighted in yellow). The L E D display is labeled as L E D display at the bottom, and O : 0 1 3 at the left side. The L E D display is connected along a horizontal line that is connected at the left side of line L 2. From top to bottom, the blocks in the L E D switch are labeled as zero, zero, and zero. An illustration of the L E D display is labeled Fahrenheit at the bottom, and is placed below its symbol. From left to right, the blocks in the illustration are labeled as one, four, and zero. A label output is placed over the symbol of the L E D display.

Return to Figure 11-14

OceanofPDF.com

Figure 11-15 Extended Description (Chapter 11)

Return to Figure 11-15

The toolbar shows an arrowhead pointing left at the top-left side and an arrowhead pointing right at the top-right side. Three symbols and seven labels enclosed in rectangles are placed relatively on the left side between the arrowheads. From left to right, the symbols are, an upper-case letter I rotated at ninety degrees, same symbol as the first but includes a rectangle at the bottom of the middle line of the letter I, and same symbol as the second but includes a horizontal line dividing the rectangle into half. From left to right, the labels highlighted in blue are C P T, A D D, S U B, M U L, D I V, M O D, and S Q R. The label M O D is circled in red. Six labels are shown at the bottom. From left to right, the labels are arrow-head facing, arrow-head facing right, bit, compare, compute or math (highlighted in blue), move or logical, and file or misc. The ladder diagram placed below the toolbar shows M O D instruction connected along a horizontal line, which is connected between two vertical lines. From top to bottom, the M O D instruction is labeled as M O D, module, source A thumb_value, 7 (highlighted in blue and placed at the right side), source B 2 (highlighted in blue), dest actual_value, and 1 (highlighted in red and placed at the right side).

Return to Figure 11-15

OceanofPDF.com

Figure 11-16 Extended Description (Chapter 11)

Return to Figure 11-16

The ladder diagram consists of three vertical lines. From the left, the first line is labeled as L 1. The N C switch labeled as S W is connected along a horizontal line connected at the right side of the line L 1. A label input is placed over the horizontal line connected to the L 1. From left to right, X I C open symbol (highlighted in green) labeled as S W, and S Q R instruction are connected in series along a horizontal line connected between the second and the third vertical line. A label ladder logic program is placed over the horizontal line connected between the second and the third vertical line. From top to bottom, S Q R instruction is labeled as S Q R, square root, source N 7 : 1 0 1 , 144 (highlighted in blue and placed at the right side), destination N 7 : 1 0 5, and 12 (highlighted in red and placed at the right side). A label (square root of 144 = 12) is placed below the ladder diagram. An illustration of switch is placed below its symbol.

Return to Figure 11-16

OceanofPDF.com

Figure 11-17 Extended Description (Chapter 11)

Return to Figure 11-17

The ladder diagram consists of three vertical lines. From the left, the first line is labeled as L 1. The N C switch labeled as S W is connected along a horizontal line connected at the right side of the line L 1. A label input is placed over the horizontal line connected to the L 1. From left to right, X I C open symbol (highlighted in green) labeled as S W, and N E G instruction are connected in series along a horizontal line connected between the second and the third vertical line. A label ladder logic program is placed over the horizontal line connected between the second and the third vertical line. From top to bottom, N E G instruction is labeled as N E G, negate, source N 7 : 5 2 , 101 (highlighted in blue and placed at the right side), destination N 7 : 5 3, and negative 101 (highlighted in red and placed at the right side). An illustration of switch is placed below its symbol.

Return to Figure 11-17

OceanofPDF.com

Figure 11-18 Extended Description (Chapter 11)

Return to Figure 11-18

The ladder diagram consists of three vertical lines. From the left, the first line is labeled as L 1. The N C switch labeled as S W is connected along a horizontal line connected at the right side of the line L 1. A label input is placed over the horizontal line connected to the L 1. From left to right, X I C open symbol (highlighted in green) labeled as S W, and C L R instruction are connected in series along a horizontal line connected between the second and the third vertical line. A label ladder logic program is placed over the horizontal line connected between the second and the third vertical line. From top to bottom, C L R instruction is labeled as C L R, clear, destination N 7 : 2 2, and 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 (highlighted in blue). An illustration of switch is placed below its symbol.

Return to Figure 11-18

OceanofPDF.com

Figure 11-19 Extended Description (Chapter 11)

Return to Figure 11-19

The ladder diagram consists of three vertical lines. From the left, the first line is labeled as L 1. The N C switch labeled as S W is connected along a horizontal line connected at the right side of the line L 1. A label input is placed over the horizontal line connected to the L 1. From left to right, X I C open symbol (highlighted in green) labeled as S W, and T O D instruction are connected in series along a horizontal line connected between the second and the third vertical line. A label ladder logic program is placed over the horizontal line connected between the second and the third vertical line. From top to bottom, T O D instruction is labeled as T O D, to B C D, source N 7 : 2 3, 10 (highlighted in blue and placed at the right side), destination O : 2 0, and 16 (highlighted in red and placed at the right side). An illustration of switch is placed below its symbol.

Return to Figure 11-19

OceanofPDF.com

Figure 11-20 Extended Description (Chapter 11)

Return to Figure 11-20

The ladder diagram consists of three vertical lines. From the left, the first line is labeled as L 1. The N C switch labeled as S W is connected along a horizontal line connected at the right side of the line L 1. A label input is placed over the horizontal line connected to the L 1. From left to right, X I C open symbol (highlighted in green) labeled as S W, and F R D instruction are connected in series along a horizontal line connected between the second and the third vertical line. A label ladder logic program is placed over the horizontal line connected between the second and the third vertical line. From top to bottom, F R D instruction is labeled as F R D, from B C D, source I : 3 0, 16 (highlighted in blue and placed at the right side), destination N 7 : 2 4, and 10 (highlighted in red and placed at the right side). An illustration of switch is placed below its symbol.

Return to Figure 11-20

OceanofPDF.com

Figure 11-21 Extended Description (Chapter 11)

Return to Figure 11-21

The ladder diagram consists of three vertical lines. From the left, the first line is labeled as L 1. The N C switch labeled as S W is connected along a horizontal line connected at the right side of the line L 1. A label input is placed over the horizontal line connected to the L 1. From left to right, X I C open symbol (highlighted in green) labeled as S W, and S C L instruction are connected in series along a horizontal line connected between the second and the third vertical line. A label ladder logic program is placed over the horizontal line connected between the second and the third vertical line. From top to bottom, S C L instruction is labeled as S C L, scale, source N 7 : 0, 100 (highlighted in blue and placed at the right side), rate (/1 0 0 0 0) 25000 (both the labels highlighted in red), offset 127 (both the labels highlighted in red), destination N 7 : 1, and 377 (highlighted in red and placed at the right side). An illustration of switch is placed below its symbol.

Return to Figure 11-21

OceanofPDF.com

Figure 11-22 Extended Description (Chapter 11)

Return to Figure 11-22

The table labeled scale analog input is placed at the left side. From top to bottom, the table is labeled as scale w/parameters (center), input I : 1 dot 0, 0 < (right), input minimum 0, 0 < (right), input maximum 32767, 32767 < (right), scaled minimum 0, 0 < (right), scaled maximum 1000, 1000 < (right), scaled output N 7 : 2 0, and 0 < (right). From the left, the illustration begins with an arrow labeled applied pressure pointing at the left side of a strain-gauge pressure transducer. The transducer leads to transmitter. The transmitter is connected to fifth, and sixth terminal of analog input module, through a label input I : 1 dot 0. The analog input module points at A D C, which further points at the right side of a table with 16 columns. The table is labeled as scaled output N 7 : 2 0. All the cells are labeled as zero. A label 15 and zero are placed at the bottom-left side and the bottom-right side of the table.

Return to Figure 11-22

OceanofPDF.com

Figure 11-23 Extended Description (Chapter 11)

Return to Figure 11-23

The table labeled scale analog output is placed at the left side. From top to bottom, the table is labeled as scale w/parameters (center), input N 7 : 2 1, 0 < (right), input minimum 0, 0 < (right), input maximum 100, 100 < (right), scaled minimum 6242, 6242 < (right), scaled maximum 31208, 31208 < (right), scaled output O : 1 dot 0, and 0 < (right). From the right, the illustration shows that a water source with controlled water flow is connected to a transducer, through a label proportional control valve. The transducer is connected to fifth terminal of analog output module, through a label four to 20 milli-ampere analog signal. The transducer is also connected to sixth terminal of analog output module, through a label scaled output O : 1 dot 0. The D A C points at the analog output module from the left side. An arrow points towards the base of the triangle labeled D A C from the left side of a table labeled as percent open register N 7 : 2 1, through a label from processor. The table has 16 columns. All the cells of the table are labeled as zero. A label 15 and zero are placed at the bottom-left side and the bottom-right side of the table.

Return to Figure 11-23

OceanofPDF.com

Figure 11-24 Extended Description (Chapter 11)

Return to Figure 11-24

The ladder diagram consists of three vertical lines. From the left, the first line is labeled as L 1. The N O switch labeled as S W is connected along a horizontal line connected at the right side of the line L 1. A label input is placed over the horizontal line connected to the L 1. From left to right, X I C open symbol (highlighted in green) labeled as S W, F A L instruction, and output energize E N are connected in series along a horizontal line connected between the second and the third vertical line. From top to bottom, the right side of the F A L instruction is connected to output energize D N, and output energize E R. A label ladder logic program is placed over the horizontal line connected between the second and the third vertical line. From top to bottom, F A L instruction is labeled as F A L, file arith or logical, control R 6 : 1, length 4, position 0, mode all, destination # N 7 : 1 0 0 (highlighted in red), expression, and # N 7 : 2 5 (highlighted in blue) + # N 7 : 5 0 (highlighted in red). An illustration of switch is placed below its symbol. From left to right, the tables are labeled as # N 7 : 2 5, # N 7 : 5 0, and # N 7 : 1 0 0. From top to bottom, the rows of the first table are labeled as 25, 234, 1256, and 77. From top to bottom, the rows of the second table are labeled as 50, 22, 456, and 100. From top to bottom, the rows of the third table are labeled as 75, 256, 1712, and 177. A label N 7 : 2 5, and N 7 : 2 8 are placed at the top-left and the bottom-left side of first table, respectively. Four positive signs are present at the right side of the first table, such that they are aligned from top to bottom. A label N 7 : 5 0 is placed at the right side of the first positive sign and a label N 7 : 5 3 is placed at the right side of the fourth positive sign. A label + N 7 : 1 0 0 and = N 7 : 1 0 3 are placed at the right side of the first and the fourth row of the second table, respectively.

Return to Figure 11-24

OceanofPDF.com

Figure 11-25 Extended Description (Chapter 11)

Return to Figure 11-25

The ladder diagram consists of three vertical lines. From the left, the first line is labeled as L 1. The N O switch labeled as S W is connected along a horizontal line connected at the right side of the line L 1. A label input is placed over the horizontal line connected to the L 1. From left to right, X I C open symbol (highlighted in green) labeled as S W, F A L instruction, and output energize E N are connected in series along a horizontal line connected between the second and the third vertical line. From top to bottom, the right side of the F A L instruction is connected to output energize D N, and output energize E R. A label ladder logic program is placed over the horizontal line connected between the second and the third vertical line. From top to bottom, F A L instruction is labeled as F A L, file arith or logical, control R 6 : 5, length 4, position 0, mode 2, destination # N 7 : 2 5 5 (highlighted in red), expression, and # N 1 0 : 0 (highlighted in blue) minus 255 (highlighted in red). An illustration of switch is placed below its symbol. From left to right, the tables are labeled as # N 1 0 : 0, and # N 7 : 2 5 5. From top to bottom, the rows of the first table are labeled as 680, 950, 20, and 100. From top to bottom, the rows of the second table are labeled as 425, 695, negative 235, and negative 155. A label N 1 0 : 0, and N 1 0 : 3 are placed at the top-left and the bottom-left side of first table, respectively. A label negative 255 (enclosed in a cell and is highlighted in red) = N 7 : 2 5 5 is placed at the right side of the first row and a label = N 7 : 2 5 8 is placed at the right side of the fourth row of the first table.

Return to Figure 11-25

OceanofPDF.com

Figure 11-26 Extended Description (Chapter 11)

Return to Figure 11-26

The ladder diagram consists of three vertical lines. From the left, the first line is labeled as L 1. The N O switch labeled as S W is connected along a horizontal line connected at the right side of the line L 1. A label input is placed over the horizontal line connected to the L 1. From left to right, X I C open symbol (highlighted in green) labeled as S W, F A L instruction, and output energize E N are connected in series along a horizontal line connected between the second and the third vertical line. From top to bottom, the right side of the F A L instruction is connected to output energize D N, and output energize E R. A label ladder logic program is placed over the horizontal line connected between the second and the third vertical line. From top to bottom, F A L instruction is labeled as F A L, file arith or logical, control R 6 : 8, length 4, position 0, mode all, destination # N 7 : 5 0 0 (highlighted in red), expression, and # N 7 : 3 3 0 (highlighted in blue) * N 7 : 2 3 (highlighted in red). An illustration of switch is placed below its symbol. From left to right, the tables are labeled as # N 7 : 3 3 0, and # N 7 : 5 0 0. From top to bottom, the rows of the first table are labeled as 20, 240, negative 78, and 321. From top to bottom, the rows of the second table are labeled as 2000, 24000, negative 7800, and 32100. A label N 7 : 3 3 0, and N 7 : 3 3 3 are placed at the top-left and the bottom-left side of first table, respectively. A label * N 7 : 2 3 is placed at the right side of the first row and a label = N 7 : 5 0 3 is placed at the left side of the fourth row of the second table. A cell labeled N 7 : 2 3 at the top has a label 100 and is placed between the first and the second table. A label = N 7 : 5 0 0 is placed at the right side of the cell.

Return to Figure 11-26

OceanofPDF.com

Figure 11-27 Extended Description (Chapter 11)

Return to Figure 11-27

The ladder diagram consists of three vertical lines. From the left, the first line is labeled as L 1. The N O switch labeled as S W is connected along a horizontal line connected at the right side of the line L 1. A label input is placed over the horizontal line connected to the L 1. From left to right, X I C open symbol (highlighted in green) labeled as S W, F A L instruction, and output energize E N are connected in series along a horizontal line connected between the second and the third vertical line. From top to bottom, the right side of the F A L instruction is connected to output energize D N, and output energize E R. A label ladder logic program is placed over the horizontal line connected between the second and the third vertical line. From top to bottom, F A L instruction is labeled as F A L, file arith or logical, control R 6 : 7, length 4, position 1, mode incremental, destination F 8 : 2 0 0 (highlighted in red), expression, and # F 8 : 2 0 (highlighted in blue) / F 8 : 1 0 0 (highlighted in red). An illustration of switch is placed below its symbol. From left to right, the tables are labeled as # F 8 : 2 0, and # F 8 : 1 0 0. From top to bottom, the rows of the first table are labeled as 100, 25, 1.33, and 586. From top to bottom, the rows of the second table are labeled as 1000, 2, 1.5, and 3. A label F 8 : 2 0, and F 8 : 2 3 are placed at the top-left and the bottom-left side of first table, respectively. A symbol of division along with a label F 8 : 1 0 0 is placed at the right side of the first row and a label F 8 : 1 0 3 is placed at the right side of the row of the first table. A label = F 8 : 2 0 0 is placed at the right side of the first row of the second table. A cell labeled F 8 : 2 0 0 at the top has a label 0.1 and is placed at the right side of the second table.

Return to Figure 11-27

OceanofPDF.com

Figure 11-35 Extended Description (Chapter 11)

Return to Figure 11-35

The ladder diagram consists of a horizontal line connected between two vertical lines. From left to right, X I C open symbol, F A L instruction, and output energize E N are connected in series along the horizontal line. From top to bottom, the right side of the F A L instruction is connected to output energize D N, and output energize E R. From top to bottom, the F A L instruction is labeled as F A L, file arith or logical, control R 6 : 0, length 5, position 0, mode all, destination # N 7 : 1 0, expression, and # N 1 1 : 0 plus 1 0. The table labeled as file # N 1 1 : 0 is placed below the ladder diagram. From top to bottom, the rows of the table are labeled as 328, 150, 10, 32, and zero.

Return to Figure 11-35

OceanofPDF.com

Figure 11-36 Extended Description (Chapter 11)

Return to Figure 11-36

The ladder diagram consists of four vertical lines. From the left, the first line is labeled as L 1, and the fourth line is labeled as L 2. Three horizontal lines are connected at the right side of the line L 1. From top to bottom, N O proximity switch labeled as S 1, N O proximity switch labeled as S 2, and N O pushbutton labeled as reset are connected along the horizontal lines. A label inputs is placed over the first horizontal line connected to the line L 1. A label ladder logic program is placed over the five horizontal lines, which are connected between the second and the third vertical line. From left to right, X I C open symbol labeled as S 1, C T U instruction, and output energize C U are connected in series along the first horizontal line. The right side of the C T U instruction is connected to an output energize D N. From top to bottom, C T U is labeled as C T U, count up counter, counter C 5 : 0, preset 350, and accumulated zero. From left to right, X I C open symbol labeled as S 2, C T U instruction, and output energize C U are connected in series along the second horizontal line. The right side of the C T U instruction is connected to an output energize D N. From top to bottom, C T U is labeled as C T U, count up counter, counter C 5 : 1, preset 350, and accumulated zero. The A D D instruction is connected along the third horizontal line. From top to bottom, A D D instruction is labeled as A D D, add, source A C 5 : 0 dot A C C, source B C 5 : 1 dot A C C, and destination N 7 : 1. From left to right, G E Q instruction, and output energize labeled as P L 1 are connected in series along the fourth horizontal line. From top to bottom, G E Q instruction is labeled as G E Q, greater than or equal, source N 7 : 1, and source B 350. From left to right, X I C open symbol labeled as reset, and output energize R E S labeled as C 5 : 0 are connected in series along the fifth horizontal line. An output energize R E S labeled as C 5 : 1 is connected across the output energize R E S labeled as C 5 : 0. A label output is placed over the horizontal line connected at the left side of the line L 2. A pilot light labeled as P L 1 is connected along the horizontal line connected to the line L 2.

Return to Figure 11-36

OceanofPDF.com

Figure 11-37 Extended Description (Chapter 11)

Return to Figure 11-37

The ladder diagram consists of four vertical lines. From the left, the first line is labeled as L 1, and the fourth line is labeled as L 2. Three horizontal lines are connected at the right side of the line L 1. From top to bottom, N O pushbutton labeled as start, N C pushbutton labeled as stop, and transducer labeled as weight transducer (all 16 bits) at the bottom and I : 0 1 2 at the right side are connected along the horizontal lines. A label inputs is placed over the first horizontal line connected to the line L 1. A label ladder logic program is placed over the five horizontal lines, which are connected between the second and the third vertical line. From left to right, X I C open symbol (highlighted in green) labeled as stop, X I C open symbol labeled as start, X I O open symbol (highlighted in green) labeled as full, and output energize labeled as fill solenoid are connected in series along the first horizontal line labeled as one. An X I C open symbol labeled as fill solenoid is connected across the X I C open symbol labeled as start. From left to right, X I C open symbol labeled as fill solenoid, X I O open symbol (highlighted in green) labeled as full, and output energize labeled as filling are connected in series along the second horizontal line labeled as two. From left to right, G E Q instruction, and output energize labeled as full are connected in series along the third horizontal line labeled as three. From top to bottom, G E Q instruction is labeled as G E Q, greater than or equal, source A I : 0 1 2 (highlighted in yellow), and source B 500. From left to right, X I C open symbol labeled as full, and S U B instruction are connected in series along the fourth horizontal line labeled as four. From top to bottom, S U B instruction is labeled as S U B, subtract, source A I : 0 1 2 (highlighted in yellow), source B 500, and destination N 7 : 1. From left to right, G E Q instruction, and output energize labeled as alarm are connected in series along the fifth horizontal line labeled as five. From top to bottom, G E Q instruction is labeled as G E Q, greater than or equal, source N 7 : 1, and source B 5. A label outputs is placed over the four horizontal lines connected at the left side of the line L 2. From top to bottom, solenoid labeled as fill solenoid, pilot light labeled filling, pilot light labeled full, and alarm are connected along the horizontal lines connected to the line L 2.

Return to Figure 11-37

OceanofPDF.com

Figure 11-38 Extended Description (Chapter 11)

Return to Figure 11-38

The ladder diagram consists of four vertical lines. From the left, the first line is labeled as L 1, and the fourth line is labeled as L 2. Three horizontal lines are connected at the right side of the line L 1. From top to bottom, N O switch labeled as on or off, thumbwheel switch labeled as T W S at the top, and I : 0 1 2 at the right side, and thermocouple labeled as thermocouple input at the bottom and I : 0 1 3 within the symbol are connected along the horizontal lines. A label inputs is placed over the first horizontal line connected to the line L 1. A label ladder logic program is placed over the six horizontal lines, which are connected between the second and the third vertical line. From top to bottom, M U L, A D D, and S U B instruction are connected along the first, the second, and the third horizontal lines, respectively. From top to bottom, M U L instruction is labeled as M U L, multiply, source A I : 0 1 2 (highlighted in yellow), 400 (at the right side), source B 0 point 0 1 0 0 0 0 0, destination N 7 : 0, and 4 (at the right side). From top to bottom, A D D instruction is labeled as A D D, add, source A I : 0 1 2 (highlighted in yellow), 400 (at the right side), source B N 7 : 0, 4 (at the right side), destination N 7 : 1, and 404 (at the right side). From top to bottom, S U B instruction is labeled as S U B, subtract, source A I : 0 1 2 (highlighted in yellow), 400 (at the right side), source B N 7 : 0, 4 (at the right side), destination N 7 : 2, and 396 (at the right side). From left to right, L E S instruction, and output energize (highlighted in green) labeled as P L 1 are connected along the fourth horizontal line. From top to bottom, the L E S instruction is labeled as L E S, less than, source A I : 0 1 3 (highlighted in yellow), 0 (at the right side), source B N 7 : 2, and 396 (at the right side). From left to right, G R T instruction, and output energize labeled as P L 2 are connected along the fifth horizontal line. From top to bottom, the G R T instruction is labeled as G R T, greater than, source A I : 0 1 3 (highlighted in yellow), 0 (at the right side), source B N 7 : 1, and 404 (at the right side). From left to right, XI C open symbol labeled as on or off, X I C open symbol (highlighted in green) labeled as P L 1, X I O open symbol (highlighted in green) labeled as P L 2, and an output energize labeled as heater are connected in series along the sixth horizontal line. An X I C open symbol labeled as heater is connected across the X I C open symbol (highlighted in green) labeled as P L 1. A label outputs is placed over the three horizontal lines connected at the left side of the line L 2. From top to bottom, heater, pilot light labeled P L 1, and pilot light labeled P L 2 are connected along the horizontal lines connected to the line L 2.

Return to Figure 11-38

OceanofPDF.com

Figure 11-39 Extended Description (Chapter 11)

Return to Figure 11-39

The ladder diagram consists of four vertical lines. From the left, the first line is labeled as L 1, and the fourth line is labeled as L 2. The thumbwheel switch is labeled as Celsius at the top, the thumbwheel switch at the bottom, and I : 0 1 2 at the right side. The thumbwheel switch is connected along a horizontal line that is connected at the right side of line L 1. From top to bottom, the blocks in the thumbwheel switch are labeled as one, two, and three. A label input is placed over the symbol of the thumbwheel switch. There are three horizontal lines connected between the second and the third vertical line. A label ladder logic program is placed over the first horizontal line connected between the second and the third vertical line. From top to bottom, M U L, D I V, and A D D, instructions are connected along the horizontal lines, which are connected between the second and the third vertical line. From top to bottom, M U L instruction is labeled as M U L, multiply, source A I : 0 1 2, source B 9, and destination N 7 : 0. From top to bottom, D I V instruction is labeled as D I V, divide, source A N 7 : 0, source B 5, and destination N 7 : 1. From top to bottom, A D D instruction is labeled as A D D, add, source A N 7 : 1, source B 32, and destination O : 0 1 3. The L E D display is labeled as Fahrenheit at the top, L E D display at the bottom, and O : 0 1 3 at the left side. The L E D display is connected along a horizontal line that is connected at the left side of line L 2. From top to bottom, the blocks in the L E D switch are labeled as zero, zero, and zero. A label output is placed over the symbol of the L E D display.

Return to Figure 11-39

OceanofPDF.com

Figure 11-40 Extended Description (Chapter 11)

Return to Figure 11-40

The ladder diagram consists of a horizontal line connected between two vertical lines. From left to right, X I C open symbol labeled as input, and A D D instruction are connected in series along the horizontal line. From top to bottom, A D D instruction is labeled as A D D, add, source A N 7 : 1, 208 (at the right side), source B N 7 : 2, 114 (at the right side), and destination N 7 : 3. From top to bottom, M U L instruction, and D I V instruction are parallelly connected across the A D D instruction. From top to bottom, M U L instruction is labeled as M U L, multiply, source A N 7 : 3, source B N 7 : 4, 4 (at the right side), and destination N 7 : 5. From top to bottom, D I V instruction is labeled as D I V, divide, source A N 7 : 5, source B 5 dot 0 0 0 0 0 0, and destination F 8 : 1.

Return to Figure 11-40

OceanofPDF.com

Figure 11-41 Extended Description (Chapter 11)

Return to Figure 11-41

The ladder diagram consists of a horizontal line connected between two vertical lines. From left to right, X I C open symbol labeled as input, and S U B instruction are connected in series along the horizontal line. From top to bottom, S U B instruction is labeled as S U B, subtract, source A N 7 : 1, 80 (at the right side), source B N 7 : 2, 20 (at the right side), and destination N 7 : 3. From top to bottom, M U L instruction, A D D instruction, and S Q R instruction are parallelly connected across the S U B instruction. From top to bottom, M U L instruction is labeled as M U L, multiply, source A N 7 : 3, source B 2, and destination N 7 : 4. From top to bottom, A D D instruction is labeled as A D D, add, source A N 7 : 4, source B 24, and destination N 7 : 5. From top to bottom, S Q R instruction is labeled as S Q R, square root, source N 7 : 5, and destination N 7 : 6.

Return to Figure 11-41

OceanofPDF.com

Figure 12-1 Extended Description (Chapter 12)

Return to Figure 12-1

From left to right, the symbol shows 2 N C limit switches with a double headed curve arrow pointing at the left side.

Return to Figure 12-1

OceanofPDF.com

Figure 12-2 Extended Description (Chapter 12)

Return to Figure 12-2

Operation of the sequencer is shown in four steps. The rotatory sequencer is shown as a distorted circle with a terminal and placed at the left side of three contacts, which are mounted from top. A curved arrow is shown in clock-wise direction over the cam. In the first step, the cam is not touching the contacts as its distorted side is close to the contact. In the second step, the curved surface of the cam touches the first contact, which further touches the second contact. In the third step, the cam rotates, such that the another distorted part touches the first contact, and all the contacts gets connected. In the last step, the curved surface touches the contacts, such that the third contact gets de-attached again. At the bottom, an illustration shows a cylinder labeled as motor, which is connected to another cylinder of smaller radius. The second cylinder passes through four discs labeled as cam.

Return to Figure 12-2

OceanofPDF.com

Figure 12-3 Extended Description (Chapter 12)

Return to Figure 12-3

A check pattern is placed over the cylinder with 16 columns. The cylinder rotates in anti-clockwise direction. Red circles labeled as peg are placed within boxes over the cylinder. A label N O switch, and steps are placed at the left side and the right side of the N O switches, respectively. From top to bottom, numbers one, two, three, and four are shown at the right side of the cylinder. The pegs are placed over the cylinder at the first, second, third, fourth, seventh, eighth, eleventh, thirteenth, fifteenth, and sixteenth positions of the row labeled as one. The pegs are placed over the cylinder at the second, third, fourth, seventh, ninth, eleventh, thirteenth, and fifteenth positions of the row labeled as two. The pegs are placed over the cylinder at the first, tenth, and fourteenth positions of the row labeled as three. The pegs are placed over the cylinder at second, fourth, sixth, seventh, eleventh, thirteenth, and fifteenth positions of the row labeled as four. The table placed over the switches are labeled as equivalent sequencer data table. It consists of sixteen columns and four rows. From left to right, label one is placed at the second, fourth, sixth, seventh, eleventh, thirteenth, and fifteenth cells of the first row labeled as four. From left to right, label one is placed at the first, tenth, and fourteenth cells of the second row labeled as three. From left to right, label one is placed at the second, third, fourth, seventh, ninth, eleventh, thirteenth, and fifteenth cells of the third row labeled as two. From left to right, label one is placed at the first, second, third, fourth, seventh, eighth, eleventh, thirteenth, fifteenth, and sixteenth cells of the fourth row labeled as one. The remaining cells are labeled as zero.

Return to Figure 12-3

OceanofPDF.com

Figure 12-5 Extended Description (Chapter 12)

Return to Figure 12-5

The ladder diagram consists of two vertical lines, whose top end are connected, through a grounded dishwasher. From left, the first vertical line is labeled as L 1, and the second vertical line is labeled as L 2. From the right side, the L 1 is connected to an N O switch, through a terminal. The right side of the first N O switch is connected to a switch (highlighted in green) labeled as encircled one line switch, through two terminals. The line switch is connected to the left side of the second N O switch, through two terminals. The right side of the second N O switch is connected to the line L 2, through a terminal, another terminal, a motor labeled as encircled two drain pump, and a terminal. The left end of the second N O switch is connected to left end of fifth N O switch at the bottom, through three nodes. The first node is connected to the line L 2, through the third N O switch, two terminals, an N C switch (highlighted in green) labeled as safety water level switch, another two terminals, a solenoid labeled as encircled three fill valve, and a terminal. The second node is connected to the line L 2, through two terminals, the second motor labeled as encircled four timer motor, and a terminal. The third node is connected to line L 2, through the fourth N O switch, two terminals, the third motor labeled as encircled five circulating motor, and a terminal. The right side of the fifth N O switch is connected to the line L 2, through two terminals, a resistor labeled as encircled six heater, and a terminal. All the N O switches are enclosed in a rectangle labeled as timer, and are highlighted blue. The table at the right side consists of three columns and nine rows. The first column, from third row is subdivided into two columns. The first and the second row of the first column are labeled as machine function, and off, relatively. From top to bottom, the left sub-column is labeled as first prerinse, prewash, second prerinse, wash, first rinse, second rinse, and dry. From top to bottom, the right sub-column is labeled as drain, fill, rinse, drain; fill, wash, drain; fill, rinse, drain; fill, wash, drain; fill, rinse, drain; fill, rinse, drain; and dry, drain, dry. From top to bottom, the second column is labeled as timer increment; zero to one; two, three, four to five, six; seven, eight to ten, 11; 12, 13 to 15, 16; 17, 18 to 30, 31; 32, 33 to 34, 35; 36, 37 to 41, 42; and 43 to 58, 59, 60. From top to bottom, the first row of the third column is labeled as active devices, and the second row of the third column is unlabeled. From top to bottom, the remaining rows of the third column are labeled as 1 2 4, 1 3 4 5, 1 4 5 6, 1 2 4 5; 1 3 4 5, 1 4 5 6, 1 2 4 5; 1 3 4 5, 1 4 5 6, 1 2 4; 1 3 4, 1 4 5 6, 1 2 4 5; 1 3 4 5, 1 4 5 6, 1 2 4 5; 1 3 4 5, 1 4 5 6, 1 2 4 5; and 1 4 6, 1 2 4 6, 1 4 6.

Return to Figure 12-5

OceanofPDF.com

Figure 12-6 Extended Description (Chapter 12)

Return to Figure 12-6

A P L C programming interface menu. It contains a row of nine rectangular buttons, each labeled in uppercase letters. From left to right, the button labels are: B S L (Bit Shift Left), B S R (Bit Shift Right), S Q C (Sequencer Compare), S Q L (Sequencer Load), S Q O (Sequencer Output), F F L (First In First Out Load), F F U (First In First Out Unload), L F L (Last In First Out Load), L F U (Last In First Out Unload).

Return to Figure 12-6

OceanofPDF.com

Figure 12-7 Extended Description (Chapter 12)

Return to Figure 12-7

From the top, the second terminal is connected to an output energize D N. From top to bottom, S Q O instruction is labeled as S Q O, sequencer output, file, mask, destination, control, length, and position. A shaded horizontal rectangle is placed at the right side of each the last six labels.

Return to Figure 12-7

OceanofPDF.com

Figure 12-8 Extended Description (Chapter 12)

Return to Figure 12-8

In all the steps, the traffic lights are labeled as N/S and E/W, from left to right. In the first step, the traffic light N/S is labeled as O : 2/0, O : 2/1, and O : 2/2, from top to bottom. The traffic light labeled as E/W is labeled as O : 2/3, O : 2/4, and O : 2/5, from top to bottom. In the first step, the first light of the traffic light N/S glows red and the third light of the traffic light E/W glows green. In the second step, the first light of the traffic light N/S glows red and the second light of the traffic light E/W glows yellow. In the third step, the third light of the traffic light N/S glows green and the first light of the traffic light E/W glows red. In the fourth step, the second light of the traffic light N/S glows yellow and the first light of the traffic light E/W glows red. The first table consists of a row and 16 columns. From left to right, the 16 columns have labels 15, 14, 13, 12, 11, 10, nine, eight, seven, six, five, four, three, two, one, and zero. These labels are placed the top of the columns. All the 16 columns are labeled as zero, where last six columns are highlighted in red. A label output word (highlighted in blue) points at a label O : 2, and are placed at the left side of the first table. The second table consists of five rows and 16 columns. From top to bottom, labels B 3 : 0, B 3 : 1, B 3 : 2, B 3 : 3, and B 3 : 4 are placed at the left side of the second table. A double headed arrow labeled as sequencer file points above the label B 3 : 0, and below the label B 3 : 4. A label position is placed at the top-right corner of the second table. From top to bottom, labels start, step one, step two, step three, and step four are placed at the right side of the second table. From left to right, the eleventh and the sixteenth columns of the second row are labeled as one. From left to right, the twelfth and the sixteenth columns of the third row are labeled as one. From left to right, the thirteenth and the fourteenth columns of the fourth row are labeled as one. From left to right, the thirteenth and the fourteenth columns of the fifth row are labeled one. All the cells labeled as one are highlighted in yellow. The remaining cells are labeled as zero. All the five rows of last six columns are highlighted in blue, except for the cells labeled as one.

Return to Figure 12-8

OceanofPDF.com

Figure 12-9 Extended Description (Chapter 12)

Return to Figure 12-9

From top to bottom, the S Q O instruction is labeled as S Q O, sequencer output, array # B 3 : 0, mask 0 0 3 F h, dest O : 2, control R 6 : 0, length 4, and position 1. From top to bottom, the right side of the S Q O instruction is connected to an output energize E N, and an output energize D N. The output module is placed at the right side of the S Q O instruction. Sixteen rectangles in four by four arrangement, are placed below the output module. Below this arrangement, the two columns of eight terminals are placed. Eight terminals of the second column are connected to a vertical line labeled as L 2, through separate pilot lights. From top to bottom, the pilot lights are labeled as zero, one, two, three, four, and five. A downwards pointing arrow from the pilot light labeled as five points towards the first table with a row of 16 columns. From left to right, the 16 columns have labels 15, 14, 13, 12, 11, 10, nine, eight, seven, six, five, four, three, two, one, and zero. These labels are placed at the top of the columns. The first 10 columns are labeled as dash. The last six columns are labeled as zero and are highlighted in red. A label O : 2 destination is placed at the left side of the first table. The second table with a row of 16 columns is placed below the first table. The first 10 columns are labeled as zero. The last six columns are labeled as one and are highlighted in green. Labels mask 0 0 3 F h (hexadecimal) 1 1 1 1 1 1 B, 1 1 1 1 1 1 B (binary) are placed at the left side of the second table. An arrow from this label points at the label mask in the S Q O instruction. Arrows from last six columns of the second table points at the bottom of the last six columns of the first table. The third table placed below the second table consists of five rows and 16 columns. From top to bottom, labels B 3 : 0, B 3 : 1, B 3 : 2, B 3 : 3, and B 3 : 4 are placed at the left side of the third table. A double headed arrow labeled as sequencer file points above the label B 3 : 0, and below the label B 3 : 4. From top to bottom, labels start, step one, step two, step three, and step four are placed at the right side of the third table. From left to right, the eleventh and the sixteenth columns of the second row are labeled as one. From left to right, the twelfth and the sixteenth columns of the third row are labeled as one. From left to right, the thirteenth and the fourteenth columns of the fourth row are labeled as one. From left to right, the thirteenth and the fourteenth columns of the fifth row are labeled one. All the cells labeled as one are highlighted in yellow. The remaining cells are labeled as zero. All the five rows of last six columns are highlighted in blue, except for the cells labeled as one. Arrows from last six columns of the third table points at the bottom of the last six columns of the second table.

Return to Figure 12-9

OceanofPDF.com

Figure 12-10 Extended Description (Chapter 12)

Return to Figure 12-10

There are four vertical lines. From the left, the first vertical line is labeled as L 1 and the fourth vertical line is labeled as L 2. The N O pushbutton labeled P B 1 is connected along a horizontal line connected at the right side of the vertical line L 1. A label input is shown over the horizontal line connected to the L 1. From left to right, X I C open symbol labeled as P B 1, S Q O instruction, and an output energize E N are connected in series along a horizontal line connected between the second and the third vertical line. From the right side, the second terminal of the S Q O is connected to an output energize D N. From top to bottom, S Q O instruction is labeled as S Q O, sequencer output, file # B 3 : 0, mask 3 F h, dest O : 2, control R 6 : 0, length 4, and position 1. A label ladder logic program is placed over the horizontal line connected between the second and the third vertical line. Six horizontal lines are connected at the left side of the line L 2. From top to bottom, red light labeled O : 2/0, yellow light labeled O : 2/1, green light labeled O : 2/2, red light labeled O : 2/3, yellow light labeled O : 2/4, and green light labeled as O : 2/5. A label output is placed above the first horizontal line connected to the vertical line L 2. For the illustration, an arrow from each row of a block with four rows points towards a block labeled destination. The block from which the arrow diverges is at the left side and labeled as file # B 3 : 0. A label positions is placed at the top-left corner of the table on the left side. From top to bottom, the rows of the left block are labeled as 1 0 0 0 0 1, 0 1 0 0 0 1, 0 0 1 1 0 0, and 0 0 1 0 1 0. From top to bottom, the labels one, two, three, and four are placed at the left side of the left block. A label O : 2 is placed over the block labeled destination. An illustration of a traffic light is placed at the right side of the illustration.

Return to Figure 12-10

OceanofPDF.com

Figure 12-11 Extended Description (Chapter 12)

Return to Figure 12-11

Question mark is placed at the right side of the last six labels. From top to bottom, the right side of the S Q O instruction is connected to output energize E N, and output energize D N.

Return to Figure 12-11

OceanofPDF.com

Figure 12-12 Extended Description (Chapter 12)

Return to Figure 12-12

There are four vertical lines. From the left, the first vertical line is labeled as L 1 and the fourth vertical line is labeled as L 2. The N O switch labeled stop_start_timer is connected along a horizontal line connected at the right side of the vertical line L 1. A label input is shown over the horizontal line connected to the L 1. An illustration of an N O switch is placed below its symbol. There are two horizontal lines connected between the second and the third vertical lines. From left to right, X I C open symbol labeled as stop_start_timer <Local:1:I dot Data dot 0, X I O symbol labeled as step_timer dot D N, T O N instruction, and an output energize E N are connected in series along the first horizontal line. The right side of the T O N instruction is connected to an output energize D N. From top to bottom, the T O N instruction is labeled as T O N, timer on delay, timer step_timer, preset 5000, and accum 0. From left to right, X I O open symbol labeled as step_timer dot D N, S Q O instruction, and an output energize E N are connected in series along the second horizontal line. The right side of the S Q O instruction is connected to an output energize D N. From top to bottom, the S Q O instruction is labeled as S Q O, sequencer output, array S Q O_Data_Array[0], mask 1 6 # 0 0 0 0 _0 0 F E, dest output_card, <local:2:0 dot data> (at the right side), control control_tag, length 5, and position 0. Seven horizontal lines are connected at the left side of the vertical line L 2. From top to bottom, the lights labeled as 2:O dot data dot 1, 2:O dot data dot 2, 2:O dot data dot 3, 2:O dot data dot 4, 2:O dot data dot 5, 2:O dot data dot 6, and 2:O dot data dot 7 are connected along the horizontal lines connected to the line L 2. A label output is placed over the first horizontal line connected to the line L 2. A table labeled as output states entered into sequencer data array is placed at the bottom of the ladder diagram. The table consists of two columns and seven rows. From top to bottom, the rows of the first column are labeled as [dash] S Q O_data_array, [+] S Q O_data_array[0], [+] S Q O_data_array[1], [+] S Q O_data_array[2], [+] S Q O_data_array[3], [+] S Q O_data_array[4], and [+] S Q O_data_array[5]. From top to bottom, the rows of the second column are labeled as {dot dot dot}, 2#0000_0000_0000_0000_0000_0000_0000_0000, 2#0000_0000_0000_0000_0000_0000_0000_0010, 2#0000_0000_0000_0000_0000_0000_0000_0110, 2#0000_0000_0000_0000_0000_0000_0001_0000, 2#0000_0000_0000_0000_0000_0000_1100_0000, and 2#0000_0000_0000_0000_0000_0000_1111_0000. The second table is labeled as outputs which are on for each step and is placed below the first table. The second table consists of eight columns under a heading outputs, and a column under a heading step. The second table consists of seven rows. From left to right, the first eight columns of the first table is labeled as seven, six, five, four, three, two, one, and zero. The last column of the first row is unlabeled. From top to bottom, last six rows of the last column is labeled as zero, one, two, three, four, and five. The seventh column of the third row is labeled as on. The sixth and the seventh column of the fourth row are labeled on. The fourth column of the fifth row is labeled on. The first, the second, and the fifth columns of the sixth row are labeled

on. The first, the second, the third, and the fourth columns of the seventh row are labeled on. The remaining cells are unlabeled.

Return to Figure 12-12

OceanofPDF.com

Figure 12-13 Extended Description (Chapter 12)

Return to Figure 12-13

The first table is labeled as matrix-style chart. A label outputs is placed at the top side and a label position is placed at the left side of the first table. From left to right, labels seven, six, five, four, three, two, one, and zero are placed at the top side of the first table. From top to bottom, the labels one, two, three, four, five, six, seven, and eight are placed at the left side of the first table. A label asterisk indicates that output is energized is placed at the bottom of the matrix-style chart. The asterisk is placed at the first, the fifth, and the eighth rows of the first column. The asterisk is placed at the second, the fifth, and the seventh rows of the second column. The asterisk is placed at the third, the fifth, and the sixth rows of the third column. The asterisk is placed at the fourth row of the fourth column. The asterisk is placed at the fourth row of the fifth column. The asterisk is placed at the third, the fifth, and the sixth rows of the sixth column. The asterisk is placed at the second, the fifth, and the seventh rows of the seventh column. The asterisk is placed at the first, the fifth, and the eighth rows of the eighth column. The second table is labeled as sequencer output file words and is placed at the right side of the first table. Eight rows of the first eight columns are highlighted in blue. The first, the fifth, and the eighth rows of the ninth column are labeled as one. The second, the fifth, and the seventh rows of the tenth column are labeled as one. The third, the fifth, and the sixth rows of the eleventh column are labeled as one. The fourth row of the twelfth column are labeled as one. The fourth row of the thirteenth column are labeled as one. The third, the fifth, and the sixth rows of the fourteenth column are labeled as one. The second, the fifth, and the seventh rows of the fifteenth column are labeled as one. The first, the fifth, and the eighth rows of the sixteenth column are labeled as one. From top to bottom, the 16 terminals of the output module are labeled from zero through 15. The terminals labeled as zero and seven are connected to a vertical line L 2, through shaded lights. The terminals labeled from one through six are connected to the vertical line L 2, through shaded circles. A label output module at position one is placed at the right side of the output module.

Return to Figure 12-13

OceanofPDF.com

Figure 12-14 Extended Description (Chapter 12)

Return to Figure 12-14

The ladder diagram consists of three vertical lines. From left, the third vertical line is labeled as L 2. Two horizontal lines are connected between the first and the second vertical line. A label ladder logic program is placed over the first horizontal line connected between the first and the second vertical line. From left to right, X I C open symbol labeled as T 4 : 1/D N, the first S Q O instruction, and an output energize E N is connected in series along the first horizontal line. The right side of the first S Q O instruction is connected to an output energize D N. From top to bottom, the first S Q O instruction is labeled as S Q O, sequencer output, file # N 7 : 0, mask 0 0 F F h, dest O : 2, control R 6 : 0, length 4, and position 0. The series connection of the second S Q O instruction, and the second output energize E N is parallelly connected across the series connection of the first S Q O instruction and the first output energize E N. The right side of the second S Q O instruction is connected to the second output energize D N. From top to bottom, the second S Q O instruction is labeled as S Q O, sequencer output, file # N 7 : 1 0, mask 0 0 F F h, dest T 4 : 1 dot P R E, control R 6 : 0, length 4, and position 0. From left to right, X I O open symbol (highlighted in green), T O N instruction, and the third output energize E N are connected in series along the second horizontal line. The right side of the T O N instruction is connected to the third output energize D N. From top to bottom, the T O N instruction is labeled as T O N, timer on delay, timer T 4 : 1, time base 1.0, preset 25, and accumulated 0. Six horizontal lines are connected to the left side of the vertical line L 2. From top to bottom, red light labeled O : 2/0, yellow light labeled O : 2/1, green light labeled O : 2/2, red light labeled O : 2/4, yellow light labeled O : 2/5, and green light labeled as O : 2/6. A label north or south is placed above the first horizontal line connected to the vertical line L 2. A label east or west is placed above the fourth horizontal line connected to the vertical line L 2. A label outputs is placed over the label north or south. An illustration of a traffic light is placed below the sixth horizontal line connected to the line L 2. The timing chart is placed below the ladder diagram. From bottom-left, timing chart for E/W remains green for 25 seconds, remains yellow for five second, and remains red for 30 seconds. From top-left, timing chart for N/S remains red for 30 seconds, remains green for 25 second, and remains yellow for five seconds.

Return to Figure 12-14

OceanofPDF.com

Figure 12-15 Extended Description (Chapter 12)

Return to Figure 12-15

From left to right, the first cell of the first row is unlabeled, whereas remaining cells of the first row are labeled as 15, 14, 13, 12, 11, 10, nine, eight, seven, six, five, four, three, two, one, and zero. From top to bottom, except for the first cell, the other cells in the first column are labeled as N 7 : 0/, N 7 : 1/, N 7 : 2/, N 7 : 3/, and N 7 : 4/. The tenth, and the sixteenth cells of the third row are labeled as one. The eleventh, and the sixteenth cells of the fourth row are labeled as one. The twelfth, and fourteenth cells of the fifth row are labeled as one. The twelfth, and fifteenth cells of the sixth row are labeled as one. The remaining cells are labeled as zero. From left to right, the labels at the bottom are given as, radix: binary (enclosed in a rectangle with downwards pointing arrow-head at the right side), and table: N 7 : integer (enclosed in a rectangle with downwards pointing arrow-head at the right side).

Return to Figure 12-15

OceanofPDF.com

Figure 12-16 Extended Description (Chapter 12)

Return to Figure 12-16

From left to right, the first cell of the first row is unlabeled, whereas the second cell of the first row is labeled as value. From top to bottom, except for the first cell, the other cells in the first column are labeled as N 7 : 1 0/, N 7 : 1 1/, N 7 : 1 2/, N 7 : 1 3/, and N 7 : 1 4/. From top to bottom, except for the first cell, the other cells in the second column are labeled as zero, 25, five, 25, and five. The label at the bottom is given as, radix: decimal (enclosed in a rectangle with downwards pointing arrow-head at the right side).

Return to Figure 12-16

OceanofPDF.com

Figure 12-17 Extended Description (Chapter 12)

Return to Figure 12-17

There are four vertical lines. From the left, the first vertical line is labeled as L 1 and the fourth vertical line is labeled as L 2. The N O switch labeled S W is connected along a horizontal line connected at the right side of the vertical line L 1. A label input is shown over the horizontal line connected to the L 1. An illustration of an N O switch is placed below its symbol. There are two horizontal lines that are connected between the second and the third vertical lines. From left to right, X I C open symbol labeled as S W, X I O open symbol (highlighted in green) labeled as T 4 : 0/D N, T O N instruction, and an output energize E N are connected in series along the first horizontal line. The right side of the T O N instruction is connected to an output energize D N. From top to bottom, the T O N instruction is labeled as T O N, timer on delay, timer T 4 : 1, time base 1.0, preset 3, and accumulated 0. From left to right, X I C open symbol labeled as T 4 : 0/D N, S Q O instruction, and an output energize E N are connected in series along the second horizontal line. The right side of the S Q O instruction is connected to an output energize D N. From top to bottom, the S Q O instruction is labeled as S Q O, sequencer output, file # B 3 : 0, mask F F F F h, dest O : 2.0, control R 6 : 0, length 8, and position 0. The output module consisting of 16 terminals labeled from zero through 15 is placed between the third and the fourth vertical lines. Each terminal is connected to the vertical line L 2, through shaded circles. A label O : 2 is placed over the output module connected to the line L 2. The table labeled as binary table file # B 3 : 0 is placed below the ladder diagram. The table consists of 10 rows and 17 columns. The first cell of the first row is unlabeled. From left to right, the first row except for the first cell is labeled as 15, 14, 13, 12, 11, 10, nine, eight, seven, six, five, four, three, two, one, and zero. From top to bottom, the first column except for the first cell is labeled as B 3 : 0, B 3 : 1, B 3 : 2, B 3 : 3, B 3 : 4, B 3 : 5, B 3 : 6, B 3 : 7, and B 3 : 8. The third, the fourth, eleventh, twelfth, fifteenth, and sixteenth columns of the third row are labeled as one. The thirteenth, fourteenth, fifteenth, and sixteenth columns of the fourth row are labeled as one. The first, the second, ninth, tenth, thirteenth, and fourteenth columns of the fifth row are labeled as one. From ninth through sixteenth column of the sixth row are labeled as one. The ninth, and the tenth columns of the seventh row are labeled as one. The third, the fourth, ninth, tenth, eleventh, twelfth, thirteenth, fourteenth, fifteenth, and sixteenth columns of the eight row are labeled as one. The fourth, twelfth, thirteenth, fourteenth, fifteenth, and sixteenth columns of the ninth row are labeled as one. The second, the fourth, tenth, twelfth, fourteenth, and sixteenth columns of the tenth row are labeled as one. All the cells labeled one are highlighted in yellow. All the remaining cells are labeled zero.

Return to Figure 12-17

OceanofPDF.com

Figure 12-18 Extended Description (Chapter 12)

Return to Figure 12-18

There are four vertical lines. From the left, the first vertical line is labeled as L 1 and the fourth vertical line is labeled as L 2. From top to bottom, the N O proximity switch labeled S 1, and the second NO proximity switch labeled as S 2 are connected along the horizontal lines connected at the right side of the vertical line L 1. A label inputs is shown over the first horizontal line connected to the L 1. An illustration of two N O proximity switches fixed at a point are labeled as field triggered input events and are placed below the second horizontal line. There is a horizontal line connected between the second and the third vertical lines. From left to right, X I C open symbol labeled as S 1, S Q O instruction, and an output energize E N are connected in series along the horizontal line. The right side of the S Q O instruction is connected to an output energize D N. Another X I C open symbol labeled as S 2 is parallelly connected across the X I C open symbol labeled as S 1. From top to bottom, the S Q O instruction is labeled as S Q O, sequencer output, file # B 3 : 0, mask F 0 F F, dest O : 2, control R 6 : 0, length 8, and position 0. An output module consisting of 16 terminals labeled from zero through 15 is placed between the third and the fourth vertical lines. A label O : 2 is placed above the output module. Terminals labeled from zero through two are connected to the vertical line L 2, through shaded lights. Terminals labeled from three through seven and from 12 through 15 are connected to the vertical line L 2, through shaded circles. A label output is placed above the connection of the output module and the line L 2. An arrow pointing right is from a label current step points towards a label pos two and is placed below the output module. The first table with a row of 16 columns is placed below the ladder diagram. From left to right, the first, fifth, sixth, seventh, eighth, and sixteenth columns have labels 15, 11, 10, nine, eight, and zero. These labels are placed at the top of the columns. The fifth, sixth, seventh, and eighth columns are unlabeled. The last four columns are labeled as one and are highlighted in red. The remains columns are labeled as zero and highlighted in red. A label destination O : 2 is placed at the right side of the first table. The second table with a row of 16 columns is placed below the first table. The fifth, sixth, seventh, and eighth columns are labeled as zero. The remaining columns are labeled as one. All the columns are highlighted in green. A label mask is placed at the right side of the second table. Arrows from the first four columns and the last eight columns of the second table points at the bottom of the first four columns and the last eight columns of the first table. The third table placed below the second table consists of nine rows and 16 columns. From top to bottom, labels zero, one, pos two, three, four, five, six, seven, and eight are placed at the left side of the third table. A label pos is placed at the top-left corner of the third table. A label file # B 3 : 0 is placed at the top-right corner of the third table. From left to right, the third, fourth, eleventh, twelfth, fifteenth, and sixteenth columns of the second row are labeled as one. From left to right, the thirteenth, fourteenth, fifteenth, and sixteenth columns of the third row are labeled as one. From left to right, the first, second, ninth, tenth, thirteenth and the fourteenth columns of the fourth row are labeled as one. From left to right, the last eight columns of the fifth row are labeled one. From left to right, the ninth, and the tenth columns of the sixth row are labeled as one. From left to right, the third, fourth, and the last eight columns of the seventh row are labeled one. From left to

right, the fourth, and the last five columns of the eighth row are labeled one. From left to right, second, fourth, tenth, twelfth, fourteenth, and the sixteenth columns of the ninth row are labeled as one. The remaining cells are labeled as zero. All the cells are highlighted in blue, except for the last four columns of the third row. Arrows from all the columns of the first row of the third table points at the bottom of all the columns of the second table.

Return to Figure 12-18

OceanofPDF.com

Figure 12-19 Extended Description (Chapter 12)

Return to Figure 12-19

There are four vertical lines. From the left, the first vertical line is labeled as L 1 and the fourth vertical line is labeled as L 2. An input module consisting of 16 terminals labeled from zero through 15 is placed between the first and the second vertical lines. A label input module I : 1 is placed above the input module. Terminal labeled zero is connected to the vertical line L 1, through the N O pushbutton labeled as I : 0/1. Terminals labeled from 12 through 15 are connected to the vertical line L 1, through N O switches. Illustration of 4 N O switches, and an N O pushbutton is placed either above or below their symbols. A label inputs is shown over the first horizontal line connected to the L 1. There are two horizontal lines connected between the second and the third vertical lines. From left to right, X I C open symbol labeled as I : 1/0, S Q C instruction, and an output energize E N are connected in series along the first horizontal line. From top to bottom, the right side of the S Q C instruction is connected to an output energize D N, and an output energize F D (in green). From left to right, an X I C open symbol labeled as R 6 : 7 at the top and F D at the bottom, and an output energize labeled as P L 1 is connected in series along the second horizontal line. From top to bottom, the S Q C instruction is labeled as S Q C, sequencer compare, file # B 3 : 2 2, mask F 0 0 0 h, source I : 1, control R 6 : 7, length 5, and position 0. The light labeled as P L 1 is connected across a horizontal line connected to the left side of the line L 2. A label output is placed above the horizontal line connected to the line L 2. Below the ladder diagram, a table labeled as binary table of seven rows and 17 columns is placed. The first cell of the first row is unlabeled. From left to right, the first row except for the first cell is labeled as 15, 14, 13, 12, 11, 10, nine, eight, seven, six, five, four, three, two, one, and zero. From top to bottom, the first column except for the first cell is labeled as B 3 : 2 2, B 3 : 2 3, B 3 : 2 4, B 3 : 2 5, B 3 : 2 6, and B 3 : 2 7. The fifth column of the third row is labeled as one. The fourth column of the fourth row is labeled as one. The fourth, and the fifth columns of the fifth row are labeled as one. The third column of the sixth row is labeled as one. The second column of the seventh row is labeled as one. All the remaining cells are labeled as zero. The third, fourth, and the fifth columns of the fifth row are highlighted in yellow. Two labels are at the bottom of the table. From left to right, the labels at the bottom are given as, radix: binary (enclosed in a rectangle with downwards pointing arrow-head at the right side), and table: B 3 : binary (enclosed in a rectangle with downwards pointing arrow-head at the right side).

Return to Figure 12-19

OceanofPDF.com

Figure 12-20 Extended Description (Chapter 12)

Return to Figure 12-20

There are three vertical lines. From the left, the first vertical line is labeled as L 1. An input module consisting of 16 terminals labeled from zero through 15 is placed between the first and the second vertical lines. A label source word I : 2 is placed above the input module. Terminals labeled from zero through 15 are connected to the vertical line L 1, through N O limit switches. Illustration of an N O pushbutton is placed above its symbol. A label inputs is shown over the first horizontal line connected to the L 1. There is a horizontal line connected between the second and the third vertical lines. From left to right, X I C open symbol (highlighted in green) labeled as P B 1, S Q L instruction, and an output energize E N are connected in series along the horizontal line. The right side of the S Q L instruction is connected to an output energize D N. From top to bottom, the S Q L instruction is labeled as S Q L, sequencer load, file # N 7 : 2 0 (highlighted in red), source I : 2 (highlighted in yellow), control R 6 : 2 2, length 5, and position 3 (highlighted in red). An illustration of a limit switch with a cam attached on its top-right side is placed below the input module. An operating force is applied on the cam (on machine) which is attached to the limit switch. A table labeled as source word I : 2 with two columns and a row is placed below the S Q L instruction. The left-most and the right-most vertical lines of the table are extended upwards and downwards. A label source I : 2 is placed at the right side of the table. Labels 15 and zero are placed over the left and the right column, respectively. Both the columns are subdivided into two columns. From left to right, the left column is labeled as 0 0 0 0 , and 1 0 1 0. From left to right, the right column is labeled as 1 1 0 0 , and 1 1 0 1. Another table labeled as destination file # N 7 : 2 0 is placed below the first table. It consists of two columns and six rows. The left-most and the right-most vertical lines of the table are extended upwards and downwards. At each end of the vertical lines, there is a marking. A label word is placed at the top-left side of the table. A label current step (highlighted in red) points at the right side of the fourth row. Labels 15 and zero are placed over the left and the right column, respectively. From top to bottom, labels N 7 0 : 2 0, 21, 22, 23, 24, and 25 are placed at the left side of rows of the table. From top to bottom, labels zero, one, two, three, four, and five are placed at the right side of rows of the table. A horizontal rectangle is placed over the fourth row labeled as three. From left to right, the columns of the fourth row are labeled as 0 0 0 0 1 0 1 0, and 1 1 0 0 1 1 0 1. An arrow from the left side of the first table points at the left side of the horizontal rectangle placed over the fourth row of the second table.

Return to Figure 12-20

OceanofPDF.com

Figure 12-21 Extended Description (Chapter 12)

Return to Figure 12-21

The first table is labeled as original data in initial position. From left to right, the columns of the second table is labeled as 1, 1, 0, 1, 1, 0, 0, 1, 0, 1, 1, 1, 0, 1, 0, and 0. The second table is labeled as new data in shift right position. From left to right, the columns of the third table is labeled as 0, 1, 1, 0, 1, 1, 0, 0, 1, 0, 1, 1, 1, 0, 1, and 0. The third table is labeled as new data in shift right position. A rectangular wave labeled as data in = one points at the top-left side of the second table. A rectangular wave with shaded area is labeled as shift data (clock) and points at the bottom-left side of the second table. A rectangular wave labeled as data in = zero points at the top-left side of the third table. A rectangular wave with shaded area is labeled as shift data (clock) and points at the bottom-left side of the third table. The first column of the second table is highlighted in red. The first, and the second columns of the third table are highlighted in red. All the remaining columns of all three rows are highlighted in yellow.

Return to Figure 12-21

OceanofPDF.com

Figure 12-22 Extended Description (Chapter 12)

Return to Figure 12-22

A label most significant bit address (highlighted in red) points at the top side of the first column of the first table. A label one or zero points across the left side of the first column. Similarly, eight arrows pointing at the right side are placed across the right side of each column of the first table. The second table is labeled as bit shift less (B L) register. A label least significant bit address (highlighted in blue) points at the top side of the last column of the second table. A label one or zero points across the right side of the first column. Similarly, eight arrows pointing at the left side are placed across the left side of each column of the second table. The third table is labeled as wraparound or circulating shift register. An arrow from the left side of the third table points at the right side of the third table, such that it forms loop. Eight arrows pointing at the left side are placed across the left side of each column of the third table.

Return to Figure 12-22

OceanofPDF.com

Figure 12-23 Extended Description (Chapter 12)

Return to Figure 12-23

The toolbar consists of an arrow-head pointing left at the top-left side and an arrow-head pointing right at the top-right side. Nine labels enclosed in rectangles are placed relatively on the left side between the arrow-heads. From left to right, the labels are B S L, B S R, S Q C, S Q L, S Q O, F F L (highlighted in blue), F F U (highlighted in blue), L F L, and L F U. Five labels are shown at the bottom. From left to right, the labels are arrow-head facing left, arrow-head facing right, file or misc, file shift or sequencer (highlighted in blue), and program control. The right side of the B S L instruction has two terminals. An output energize E N is connected across the first terminal, from top. From the top, the second terminal is connected to an output energize D N. From top to bottom, B S L instruction is labeled as B S L, bit shift left (highlighted in blue), file, control, bit address, and length. A shaded horizontal rectangle is placed at the right side of the each last four labels. The right side of the B S R instruction has two terminals. An output energize E N is connected across the first terminal, from top. From the top, the second terminal is connected to an output energize D N. From top to bottom, B S R instruction is labeled as B S R, bit shift right (highlighted in blue), file, control, bit address, and length. A shaded horizontal rectangle is placed at the right side of the each last four labels.

Return to Figure 12-23

OceanofPDF.com

Figure 12-24 Extended Description (Chapter 12)

Return to Figure 12-24

There are three vertical lines. From the left, the first vertical line is labeled as L 1. Two horizontal lines are connected at the right side of the line L 1. From top to bottom, N O limit switch labeled as L S, and NC proximity switch labeled as sensor and I : 1/1 (highlighted in yellow) are connected along the horizontal lines connected to the line L 1. A label inputs is placed over the first horizontal line. A horizontal line is connected between the second and the third vertical lines. From left to right, an X I C open symbol labeled as L S, B S L instruction, and an output energize E N are connected in series along the horizontal line. The right side of the B S L instruction is connected to an output energize D N. From top to bottom, B S L instruction is labeled as B S L, bit shift left, file # B 3 : 1 0, control R 6 : 0, bit address I : 1/1 (highlighted in yellow), and length 20. A label ladder logic program is placed over the horizontal line. Table below the ladder diagram is labeled as B 3 : table - before limit switch clock pulse. It consists of three rows and 17 columns. The first cell of the first column is unlabeled. From left to right, except for the first cell, the first row is labeled from 15 through zero. From top to bottom, except for the first cell, the first column is labeled as B 3 : 1 0, and B 3 : 1 1. The second, third, sixth, seventh, tenth, eleventh, thirteenth, and fourteenth columns of the second row are labeled one. The fourteenth, fifteenth, and sixteenth columns of the third row are labeled one. The remaining cells are labeled as zero. Except for the first cell, the second row is highlighted in blue. The last four cells of the third row are highlighted in blue. The second table has two rows of 16 columns. The first 12 columns of the second row are merged and labeled invalid. A label 15 and a label zero is placed at the top of the first column and the last column, respectively. An arrow pointing left is labeled as shift direction and placed over the table. From top to bottom, labels B 3 : 1 0, and B 3 : 1 1 are placed at the right side of the second table. The first, second, fifth, sixth, ninth, tenth, twelfth, and thirteenth columns of the first row are labeled one. The thirteenth, fourteenth, and fifteenth columns of the second row are labeled as one. The remaining cells are labeled zero. A square labeled one and highlighted in red points at the top-right corner of the second table. A label bit address I : 1/1 (highlighted in yellow) is placed at the top-left side of the square. The thirteenth column of the second row points at a square labeled as unload bit R 6 : 0/U l, through a label 20. The third table is labeled as B 3 : table - after limit switch clock pulse. It consists of three rows and 17 columns. The first cell of the first column is unlabeled. From left to right, except for the first cell, the first row is labeled from 15 through zero. From top to bottom, except for the first cell, the first column is labeled as B 3 : 1 0, and B 3 : 1 1. The second, fifth, sixth, ninth, tenth, twelfth, thirteenth, and seventeenth columns of the second row are labeled one. The label one in the last column of the second row is circled. The fourteenth, fifteenth, and seventeenth columns of the third row are labeled one. The remaining cells are labeled as zero. Except for the first cell, the second row is highlighted in blue. The last four cells of the third row are highlighted in blue. The fourth table has two rows of 16 columns. The first 12 columns of the second row are merged and labeled invalid. A label 15 is placed at the top of the first column. An arrow pointing left is labeled as shift direction and placed over the table. From top to bottom, labels B 3 : 1 0, and B 3 : 1 1 are placed at the right side of the second table. The first, fourth, fifth, eighth, ninth, eleventh, twelfth, and

sixteenth columns of the first row are labeled one. The thirteenth, fourteenth, and sixteenth columns of the second row are labeled as one. The remaining cells are labeled zero. A square labeled one and highlighted in red points at the top-right corner of the second table. A label bit address I : 1/1 (highlighted in yellow) is placed at the top-left side of the square. The thirteenth column of the second row points at a square labeled as unload bit R 6 : 0/U l, through a label 20. This square has a label one within it and is highlighted in red.

Return to Figure 12-24

OceanofPDF.com

Figure 12-25 Extended Description (Chapter 12)

Return to Figure 12-25

There are three vertical lines. From the left, the first vertical line is labeled as L 1. Two horizontal lines are connected at the right side of the line L 1. From top to bottom, N O limit switch labeled as L S, and NC proximity switch labeled as sensor and I : 3/5 (highlighted in yellow) are connected along the horizontal lines connected to the line L 1. A label inputs is placed over the first horizontal line. A horizontal line is connected between the second and the third vertical lines. From left to right, an X I C open symbol labeled as L S, B S R instruction, and an output energize E N are connected in series along the horizontal line. The right side of the B S R instruction is connected to an output energize D N. From top to bottom, B S R instruction is labeled as B S R, bit shift right, file # B 3 : 5 0, control R 6 : 1, bit address I : 3/5 (highlighted in yellow), and length 24. A label ladder logic program is placed over the horizontal line. Table below the ladder diagram is labeled as B 3 : table - before limit switch clock pulse. It consists of three rows and 17 columns. The first cell of the first column is unlabeled. From left to right, except for the first cell, the first row is labeled from 15 through zero. From top to bottom, except for the first cell, the first column is labeled as B 3 : 5 0, and B 3 : 5 1. The second, fourth, fifth, ninth, tenth, thirteenth, fifteenth, and sixteenth columns of the second row are labeled one. The tenth, twelfth, thirteenth, sixteenth, and seventeenth columns of the third row are labeled one. The remaining cells are labeled as zero. Except for the first cell, the second row is highlighted in blue. The last eight cells of the third row are highlighted in blue. The second table has two rows of 16 columns. The first eight columns of the second row are merged and labeled invalid. A label 15 and a label zero is placed at the top of the first column and the last column, respectively. An arrow pointing right is labeled as shift direction and placed over the table. From top to bottom, labels B 3 : 5 0, and B 3 : 5 1 are placed at the right side of the second table. The first, third, fourth, eighth, ninth, twelfth, fourteenth, and fifteenth columns of the first row are labeled one. The ninth, eleventh, twelfth, fifteenth, and sixteenth columns of the second row are labeled as one. The remaining cells are labeled zero. Top-right corner of the second table points at a square labeled one and highlighted in red. A label unload bit R 6 : 1/U L is placed at the top-right side of the square. A square labeled as bit address I : 3/5 (highlighted in yellow) points at the ninth column of the second row, through a label 24. The third table is labeled as B 3 : table - after limit switch clock pulse. It consists of three rows and 17 columns. The first cell of the first column is unlabeled. From left to right, except for the first cell, the first row is labeled from 15 through zero. From top to bottom, except for the first cell, the first column is labeled as B 3 : 5 0, and B 3 : 5 1. The second, third, fifth, sixth, tenth, eleventh, fourteenth, sixteenth, and seventeenth columns of the second row are labeled one. The tenth, twelfth, thirteenth, and sixteenth columns of the third row are labeled one. The remaining cells are labeled as zero. Except for the first cell, the second row is highlighted in blue. The last eight cells of the third row are highlighted in blue. The zero in ninth cell of the second row is circled. The fourth table has two rows of 16 columns. The first eight columns of the second row are merged and labeled invalid. An arrow pointing right is labeled as shift direction and placed over the table. From top to bottom, labels B 3 : 5 0, and B 3 : 5 1 are placed at the right side of the second table. The first, second, fourth, fifth, ninth, tenth, thirteenth,

fifteenth, and sixteenth columns of the first row are labeled one. The tenth, twelfth, thirteenth, and sixteenth columns of the second row are labeled as one. The remaining cells are labeled zero. The top-right corner of the second table points at a square labeled zero and highlighted in red. A label unload bit R 6 : 1/U L is placed at the top-right side of the square. A square labeled as bit address I : 3/5 (highlighted in yellow) points at the ninth column of the second row, through a label 24. This square has a label zero within it and is highlighted in red. The ninth column of the second row is also highlighted in red.

Return to Figure 12-25

OceanofPDF.com

Figure 12-26 Extended Description (Chapter 12)

Return to Figure 12-26

There are four vertical lines. From the left, the first vertical line is labeled as L 1 and the fourth vertical line is labeled as L 2. The N O switch labeled as S W is connected along a horizontal line connected at the right side of the vertical line L 1. A label input is shown over the horizontal line connected to the L 1. There are six horizontal lines connected between the second and the third vertical lines. From left to right, X I C open symbol labeled as S W, X I O symbol (highlighted in green) labeled as T 4 : 0/D N, T O N instruction, and an output energize E N are connected in series along the first horizontal line. The right side of the T O N instruction is connected to an output energize D N. From top to bottom, the T O N instruction is labeled as T O N, timer on delay, timer T 4 : 0, time base 1.0, preset 3, and accumulated 0. From left to right, X I C open symbol labeled as T 4 : 0/D N, B S L instruction, and an output energize E N are connected in series along the second horizontal line. The right side of the B S L instruction is connected to an output energize D N. From top to bottom, B S L instruction is labeled as B S L, bit shift left, file # B 3 : 0, control R 6 : 0, bit address B 3 : 0/0, and length 3. From left to right, X I C open symbol labeled as R 6 : 0/U L, and output energize labeled as B 3 : 0/0 are connected in series along the third horizontal line. From left to right, X I C open symbol labeled as B 3 : 0/0, and output energize labeled as P L 1 (highlighted in yellow) are connected in series along the fourth horizontal line. From left to right, X I C open symbol labeled as B 3 : 0/1, and output energize labeled as P L 2 (highlighted in yellow) are connected in series along the fifth horizontal line. From left to right, X I C open symbol labeled as B 3 : 0/3, and output energize labeled as P L 3 (highlighted in yellow) are connected in series along the sixth horizontal line. A label ladder logic program is placed over the first horizontal line connected between the second and the third vertical lines. There are three horizontal lines connected at the left side of the line L 2. From top to bottom, pilot lights labeled as P L1, P L 2, and P L 3 are connected along the horizontal lines connected to the line L 2. A label outputs is placed over the first horizontal line connected to the line L 2.

Return to Figure 12-26

OceanofPDF.com

Figure 12-28 Extended Description (Chapter 12)

Return to Figure 12-28

The table labeled as station consists of a row of eight columns. From left to right, the row is labeled as B 3 : 0/7, B 3 : 0/6, B 3 : 0/5, B 3 : 0/4, B 3 : 0/3, B 3 : 0/2, B 3 : 0/1, and B 3 : 0/0. From left to right, the labels above the columns are from eight through one. A label file # B 3 : 0 is placed at the right side of the table labeled as station. From left to right, the labels below the row are part, no, 0, 0, 0, 0, 0, 0, 0, and 0. From left to right, the second row of the labels are present, yes, 1, 1, 1, 1, 1, 1, 1, and 1. The illustration shows conveyor hangers. From the left, a brown colored part is hanging, two limit switches labeled L S 1 (hanger) and L S 2 (part enter) are placed at opposite side of the hanger, a red colored part with a label one hanging above the undercoat spray gun one, two red colored parts in the over one with labels two and three, a label four between the empty hangers, a brown colored part with a label five is placed above topcoat spray gun two, a label six between the empty hangers, brown colored part in the oven 2 with a label seven, a label eight between the empty hangers enclosed in oven two, limit switch L S 3 labeled as (part exit). A brown colored part is shown in the storage, which is placed below the L S 3. A reset button and a light labeled as P L 1 are placed below the illustration.

Return to Figure 12-28

OceanofPDF.com

Figure 12-29 Extended Description (Chapter 12)

Return to Figure 12-29

There are four vertical lines. From the left, the first vertical line is labeled as L 1 and the fourth vertical line is labeled as L 2. Four horizontal lines are connected at the right side of the vertical line L 1. From top to bottom, N O limit switch labeled as clock pulse, L S 1, and I : 1/1; N O limit switch labeled as data pulse, L S 2, and I : 1/2; N O limit switch labeled as part exit, L S 3, and I : 1/3; and N O pushbutton labeled as reset and I : 1/4 are connected along the horizontal lines connected at the right side of the vertical line L 1. A label inputs is shown over the horizontal line connected to the L 1. There are seven horizontal lines connected between the second and the third vertical lines. From left to right, X I C open symbol labeled as L S 1, B S L instruction, and an output energize E N are connected in series along the first horizontal line. The right side of the B S L instruction is connected to an output energize D N. From top to bottom, B S L instruction is labeled as B S L, bit shift left, file # B 3 : 0, control R 6 : 1, bit address I : 1/2 (highlighted in yellow), and length 8. From left to right, X I C open symbol labeled as B 3 : 0/0, and an output energize labeled as SOL 1 (highlighted in yellow) are connected in series along the second horizontal line. From left to right, X I C open symbol labeled as B 3 : 0 4, and output energize labeled as SOL 2 are connected in series along the third horizontal line. From left to right, X I C open symbol labeled as L S 2 (highlighted in yellow), C T U instruction, and output energize C U are connected in series along the fourth horizontal line. The right side of the C T U instruction is connected to an output energize D N. From top to bottom, C T U instruction is labeled as count-up counter, counter C 5 : 1, preset, and accumulated. From left to right, X I C open symbol labeled as L S 3, the second C T U instruction, and output energize C U are connected in series along the fifth horizontal line. The right side of the second C T U instruction is connected to an output energize D N. From top to bottom, the second C T U instruction is labeled as count-up counter, counter C 5 : 2, preset, and accumulated. From left to right, X I C open symbol labeled as reset, and output energize R E S labeled as C 5 : 1 are connected in series along the sixth horizontal line. An output energize R E S labeled as C 5 : 2 is connected across the output energize R E S labeled as C 5 : 1. From left to right, E Q U instruction, and output energize labeled as P L 1 (highlighted in yellow) are connected in series along the seventh horizontal line. From top to bottom, E Q U instruction is labeled as E Q U, equal, source A C 5 : 1 dot A C C, and source B C 5 : 2 dot A C C. A label ladder logic program is placed over the first horizontal line connected between the second and the third vertical lines. There are three horizontal lines connected at the left side of the line L 2. From top to bottom, solenoid labeled as SOL 1 spray gun 1 and O : 2/0 (highlighted in yellow), solenoid labeled as SOL 2 spray gun 2 and O : 2/1 (highlighted in yellow), and pilot light labeled as P L 1 and O : 2/2 (highlighted in yellow) are connected along the horizontal lines connected to the line L 2. A label outputs is placed over the first horizontal line connected to the line L 2.

Return to Figure 12-29

OceanofPDF.com

Figure 12-30 Extended Description (Chapter 12)

Return to Figure 12-30

There are four vertical lines. From the left, the first vertical line is labeled as L 1 and the fourth vertical line is labeled as L 2. Four horizontal lines are connected at the right side of the line L 1. From top to bottom, N O proximity switch labeled prox # 1 (carrier detection) and I : 1/1 (highlighted in yellow); N O proximity switch labeled prox # 2 (part detection) and I : 1/2 (highlighted in yellow); N O pushbutton labeled P B 1 (failed part) and I : 1/3 (highlighted in yellow); and N O pushbutton labeled P B 2 (reworked part) and I : 1/4 (highlighted in yellow) are connected along the horizontal lines connected at the right side of the vertical line L 1. A label inputs is shown over the first horizontal line connected to the L 1. There are three horizontal lines connected between the second and the third vertical lines. From left to right, X I C open symbol labeled as I : 1/1 (highlighted in yellow) at the top and prox # 1 at the bottom, B S L instruction, and an output energize E N are connected in series along the first horizontal line. The right side of the B S L instruction is connected to an output energize D N. From top to bottom, the B S L instruction is labeled as B S L, bit shift left, file # O : 4 (highlighted in red), control R 6 : 0, bit address I : 1/2 (highlighted in red), and length 8. From left to right, X I C open symbol labeled as I : 1/3 (highlighted in yellow) at the top and P B 1 at the bottom, and an output energize U labeled as O : 4/4 (highlighted in yellow) at the top and P L 4 at the bottom are connected in series along the second horizontal line. From left to right, X I C open symbol labeled as I : 1/4 (highlighted in yellow) at the top and P B 2 at the bottom, X I O open symbol (highlighted in green) labeled as O : 4/6, and an output energize U labeled as O : 4/6 (highlighted in yellow) at the top and P L 6 at the bottom are connected in series along the third horizontal line. An output module consisting of 16 terminals labeled from zero through 15 is placed between the third and the fourth vertical lines. A label O : 4 is placed above the output module. Terminals labeled one, three, eight, and 10 are connected to the vertical line L 2, through shaded lights. The remaining terminals are connected to the vertical line L 2, through shaded circles. A label outputs is placed above the connection of the output module and the line L 2. A table labeled as station is shown above the ladder diagram. The table consists of a row with 16 columns. From left to right, the columns have labels from 16 through one. These labels are placed at the top of the columns. From left to right, the columns of the table are labeled as O : 4/1 5, O : 4/1 4, O : 4/1 3, O : 4/1 2, O : 4/1 1, O : 4/1 0, O : 4/9, O : 4/ 8, O : 4/ 7, O : 4/ 6, O : 4/ 5, O : 4/ 4, O : 4/ 3, O : 4/ 2, O : 4/ 1, and O : 4/ 0. A label O : 4 is placed at the right side of the table. The column labeled seven is highlighted in green and points at a label rework. The column labeled five is highlighted in red and points at a label inspection.

Return to Figure 12-30

OceanofPDF.com

Figure 12-31 Extended Description (Chapter 12)

Return to Figure 12-31

Question mark is placed at the right side of the last four labels. All the question marks are highlighted in blue. From top to bottom, the right side of the B S L instruction is connected to output energize E N, and output energize D N.

Return to Figure 12-31

OceanofPDF.com

Figure 12-32 Extended Description (Chapter 12)

Return to Figure 12-32

There are four vertical lines. From the left, the first vertical line is labeled as L 1 and the fourth vertical line is labeled as L 2. Three horizontal lines are connected at the right side of the vertical line L 1. From top to bottom, N O pushbutton labeled as reject_button, N O proximity switch labeled as product_sensor, and N O pushbutton labeled as reset_button are connected along the horizontal lines connected at the right side of the vertical line L 1. A label inputs is shown over the horizontal line connected to the L 1. There are four horizontal lines connected between the second and the third vertical lines, which are labeled from zero through three. From left to right, X I C open symbol labeled as reject_button and <local:1:i dot Data dot 1> (highlighted in blue), X I O open symbol labeled as control_status dot D N, and an output energize reject_source_bit are connected in series along the first horizontal line. An X I C open symbol labeled as reject_source_bit is connected across the X I C open symbol labeled as reject_button. From left to right, X I C open symbol labeled as product_sensor and <local:1:I dot data dot 0> (highlighted in blue), B S L instruction, and an output energize E N are connected in series along the second horizontal line. The right side of the B S L is connected to an output energize D N. From top to bottom, B S L instruction is labeled as B S L, bit shift left, array B S L_array, control control_status, source bit reject_source_bit, and length 4. The first M O V instruction is connected in series along the third horizontal line. From top to bottom, the first M O V instruction is labeled as M O V, move, source B S L_array, 0 (at the right side), dest zone_outputs, and 0 (at the right side). From left to right, X I C open symbol labeled as reset_button and <local:1:i dot data dot 2> (highlighted in blue), the second M O V instruction, and the third M O V instruction are connected in series along the fourth horizontal line. From top to bottom, the second M O V instruction is labeled as M O V, move, source 0, dest B S L_array, and 0 (at the right side). From top to bottom, the third M O V instruction is labeled as M O V, move, source 0, dest control_status_P O S, and 0 (at the right side). A label ladder logic program is placed over the first horizontal line connected between the second and the third vertical lines. There are four horizontal lines connected at the left side of the line L 2. From top to bottom, pilot light Z 1 labeled as <local:2:O dot data dot 0> (highlighted in blue), pilot light Z 2 labeled as <local:2:O dot data dot 1> (highlighted in blue), pilot light Z 3 labeled as <local:2:O dot data dot 2> (highlighted in blue), and pilot light Z 4 labeled as <local:2:O dot data dot 3> (highlighted in blue) are connected along the horizontal lines connected to the line L 2. A label outputs is placed over the first horizontal line connected to the line L 2. An illustration placed at the bottom-left side shows a square labeled as reject_source_bit is placed at the left side of an N O pushbutton labeled as reject_button (highlighted in yellow). The square points downwards at a row of four squares labeled as B S L_array. An arrow pointing left is placed over the row of four squares. From left, the first square in the row points downwards at another square labeled as unload_bit. A table of 12 rows and four columns is placed at the bottom-right side. From top to bottom, the first column is labeled as tag name, positive sign (enclosed in a square) B S L_array, positive sign (enclosed in a square) control_status, negative sign (enclosed in a square) zone_outputs, zone_outputs dot 0, zone_outputs dot 1, zone_outputs dot 2, zone_outputs dot 3, product_sensor, reject_button, reject_source_bit, and

reset_button. The first row of the second column is labeled as value. The third row of the second column is labeled as {dot dot dot}. The remaining rows of the second column are labeled as zero. The first row of the third column is labeled as style. The third row of the third column is unlabeled. The remaining rows of the third column are labeled as decimal. From top to bottom, the fourth column is labeled as data type, dint, control, dint, bool, bool, bool, bool, bool, bool, bool, and bool.

Return to Figure 12-32

OceanofPDF.com

Figure 12-34 Extended Description (Chapter 12)

Return to Figure 12-34

The toolbar consists of an arrow-head pointing left at the top-left side and an arrow-head pointing right at the top-right side. Nine labels enclosed in rectangles are placed relatively on the left side between the arrow-heads. From left to right, the labels are B S L, B S R, S Q C, S Q L, S Q O, F F L (highlighted in blue), F F U (highlighted in blue), L F L, and L F U. Five labels are shown at the bottom. From left to right, the labels are arrow-head facing left, arrow-head facing right, file or misc, file shift or sequencer (highlighted in blue), and program control. The right side of the F F L instruction has three terminals. From the top, the first, the second, and the third terminals are connected to an output energize E N, output energize D N, and output energize E M, respectively. From top to bottom, F F L instruction is labeled as F F L (highlighted in blue), fifo load (highlighted in blue), source, fifo, control, length, and position. A shaded horizontal rectangle is placed at the right side of the each last five labels.

Return to Figure 12-34

OceanofPDF.com

Figure 12-35 Extended Description (Chapter 12)

Return to Figure 12-35

The toolbar consists of an arrow-head pointing left at the top-left side and an arrow-head pointing right at the top-right side. Nine labels enclosed in rectangles are placed relatively on the left side between the arrow-heads. From left to right, the labels are B S L, B S R, S Q C, S Q L, S Q O, F F L (highlighted in blue), F F U (highlighted in blue), L F L, and L F U. Five labels are shown at the bottom. From left to right, the labels are arrow-head facing left, arrow-head facing right, file or misc, file shift or sequencer (highlighted in blue), and program control. The right side of the F F U instruction has three terminals. From the top, the first, the second, and the third terminals are connected to an output energize E U, output energize D N, and output energize E M, respectively. From top to bottom, F F U instruction is labeled as F F U (highlighted in blue), fifo unload (highlighted in blue), fifo, destination, control, length, and position. A shaded horizontal rectangle is placed at the right side of the last each five labels.

Return to Figure 12-35

OceanofPDF.com

Figure 12-36 Extended Description (Chapter 12)

Return to Figure 12-36

There are three vertical lines. From the left, the first vertical line is labeled as L 1. Two horizontal lines connected to the left side of the line L 1. From top to bottom, the N O pushbutton labeled input A, and the N O pushbutton labeled input B are connected along the horizontal lines connected at the right side of the vertical line L 1. A label inputs is shown over the first horizontal line connected to the L 1. Two horizontal lines are connected between the second and the third vertical lines. From left to right, X I C open symbol labeled as input A, F F L instruction, and an output energize E N are connected in series along the first horizontal line. From top to bottom, the right side of the F F L instruction is connected to an output energize D N and an output energize E M. From top to bottom, the F F L instruction is labeled as F F L, FIFO load, source N 7 : 1 0, FIFO # N 7 : 1 2, control R 6 : 0, length 10, and position 8. From left to right, X I C open symbol labeled as input B, F F U instruction, and an output energize E U are connected in series along the second horizontal line. From top to bottom, the right side of the F F U instruction is connected to an output energize D N and an output energize E M. From top to bottom, the F F U instruction is labeled as F F U, FIFO unload, FIFO # N 7 : 1 2, dest N 7 : 1 1, control R 6 : 0, length 10, and position 8. A label ladder logic program is placed over the first horizontal line connected between the second and the third vertical lines. The table placed at the right side of the ladder diagram is labeled as integer table. It consists of two columns and 13 rows. From top to bottom, the first row of first column is unlabeled. Except for the first row, the remaining rows of the first column are labeled as N 7 : 1 0, N 7 : 1 1, N 7 : 1 2, N 7 : 1 3, N 7 : 1 4, N 7 : 1 5, N 7 : 1 6, N 7 : 1 7, N 7 : 1 8, N 7 : 1 9, N 7 : 2 0, and N 7 : 2 1. From top to bottom, the rows of the second column are labeled as value, 23, 16, 31, 53, 146, 9,875, 125, 867, 5, 11, 0, and 0. A label radix: decimal (enclosed in a rectangle with a drop down button arrow at the right side) is placed below the table. The illustration is placed at the bottom and it shows a table and two rectangular blocks. Table on the right side is labeled as FIFO file 3 N 7 : 1 2. It consists of a column and 10 rows. From top to bottom, the column is labeled as 31 (highlighted in red), 153, 146, 9,875, 125, 867, 5, 11, 0 (highlighted in blue), and 0. From top to bottom, the labels at the left side of the table are from zero through nine. A label at the right side of the first row is N 7 : 1 2. A label at the right side of the last row is N 7 : 2 1. An upwards pointing arrow points from N 7 : 2 1 towards N 7 : 1 2 and is labeled as data index toward the starting address of the file, one word with each false-to-true transition of the F F U. A label position is placed at the bottom-left corner of the table. An arrow pointing left from the left side of the first row points towards the first rectangle labeled destination. The label within the first rectangle is 16 (highlighted in red) and a label at the left side of the rectangle is N 7 : 11. A label data exit from position 0 of the FIFO file on a false-to-true transition of the F F U and write over current data in the destination is placed below the first block. The second block labeled as source points at the ninth row of the table. The label within the second block is 23 (highlighted in blue) and a label placed at the left side of the second block is N 7 : 1 0. A label data enter the FIFO file on a false-to-true transition of the F F L at the position indicated in the instruction is placed below the second block.

Return to Figure 12-36

OceanofPDF.com

Figure 12-37 Extended Description (Chapter 12)

Return to Figure 12-37

For FIFO stack, three containers labeled three, two, and one (highlighted in red) are enclosed by a horizontal rectangle, from left to right. Another container labeled four is placed at the left side of the container labeled three. Another container labeled one (highlighted in red) is placed at the right side of the previous container labeled one. Each container has an arrow pointing right over it. A label in is present over the arrow above the container labeled four. A label out is present over the arrow above the second container labeled one. A label horizontal storage area is present below this illustration. For LIFO stack, three containers labeled three (highlighted in red), two, and one are enclosed by a vertical rectangle, from bottom to top. The second container labeled three (highlighted in red) is placed at the left side of the first container labeled three. The third container labeled three (highlighted in red) is placed at the right side of the first container labeled three. The second and the third container labeled three has an arrow pointing right over it. Arrow pointing right is present between the second and the first containers labeled three. Arrow pointing right is present between the first and the third containers labeled three. A label in is present over the arrow above the second container labeled three. A label out is present over the arrow above the third container labeled three. A label vertical storage area is present below this illustration.

Return to Figure 12-37

OceanofPDF.com

Figure 12-38 Extended Description (Chapter 12)

Return to Figure 12-38

The toolbar consists of an arrow-head pointing left at the top-left side and an arrow-head pointing right at the top-right side. Nine labels enclosed in rectangles are placed relatively on the left side between the arrow-heads. From left to right, the labels are B S L, B S R, S Q C, S Q L, S Q O, F F L (highlighted in blue), F F U (highlighted in blue), L F L, and L F U. Five labels are shown at the bottom. From left to right, the labels are arrow-head facing, arrow-head facing right, file or misc, file shift or sequencer (highlighted in blue), and program control. From left to right, L F L instruction and an output energize E N are connected in series along a horizontal line connected at the left side of a vertical line on the right side. The right side of the L F L instruction has three terminals. From the top, the second, and the third terminals are connected to an output energize D N, and output energize E M, respectively. From top to bottom, L F L instruction is labeled as L F L, LIFO load, source N 7 0 : 1 (highlighted in blue), LIFO # N 7 0 : 3, control R 6 : 6 1, length 64, and position 0. From left to right, L F U instruction and an output energize E N are connected in series along the second horizontal line connected at the left side of a vertical line on the right side. The right side of the L F U instruction has three terminals. From the top, the second, and the third terminals are connected to an output energize D N, and output energize E M, respectively. From top to bottom, L F U instruction is labeled as L F U, LIFO unload, LIFO # N 7 0 : 3, dest N 7 0 : 2 (highlighted in blue), control R 6 : 6 1, length 64, and position 0. The illustration shows a column of 11 rows labeled as file # N 7 0 : 3. The vertical lines of the column are extended from the top and the bottom. A curved line is placed at the endings of the vertical line. A horizontal line joins the two vertical lines near the endings. A marking above the horizontal line is shown as 63. From top to bottom, the right side of the column is labeled from three through 11. The right side of the column is labeled as 64 words allocated for LIFO stack at # N 7 0 : 3. A label word is placed at the top-left side of the column. A rectangular block labeled N 7 0 : 1 source placed at the left side of the column points at the left side of the column. A label LIFO load enters data into stack at next position is placed below the rectangular block. The right side of the column points at the left side of the second rectangular block labeled destination N 7 0 : 2 placed at the right side of the column. A label LIFO unload removes data from stack in reverse order is placed below the second rectangular block.

Return to Figure 12-38

OceanofPDF.com

Figure 12-39 Extended Description (Chapter 12)

Return to Figure 12-39

There are four vertical lines. From the left, the first vertical line is labeled as L 1 and the fourth vertical line is labeled as L 2. Three horizontal lines connected to the left side of the line L 1. From top to bottom, the N O pushbutton labeled data_stack_load, the N O pushbutton labeled data_stack_unload, and the thumbwheel labeled as data_stack_input <local : 5 : I dot data> are connected along the horizontal lines connected at the right side of the vertical line L 1. From left to right, the blocks of the thumbwheel switch is labeled as zero, zero, two, and three. A label inputs is shown over the first horizontal line connected to the L 1. Two horizontal lines are connected between the second and the third vertical lines. From left to right, X I C open symbol labeled as data_stack_load <local : 1 : I dot data dot zero>, F F L instruction, and an output energize E N are connected in series along the first horizontal line. From top to bottom, the right side of the F F L instruction is connected to an output energize D N and an output energize E M. From top to bottom, the F F L instruction is labeled as F F L, FIFO load, source data_stack_input, FIFO data_stack_array[0], control data_stack_status, length 5, and position 0. From left to right, X I C open symbol labeled as data_stack_unload <local : 1 : I dot data dot one>, F F U instruction, and an output energize E N are connected in series along the second horizontal line. From top to bottom, the right side of the F F U instruction is connected to an output energize D N and an output energize E M. From top to bottom, the F F U instruction is labeled as F F U, FIFO unload, FIFO data_stack_array[0], dest data_stack_output, control data_stack_status, length 5, and position 0. A label ladder logic program is placed over the first horizontal line connected between the second and the third vertical lines. The L E D display labeled as data_stack_output [0] <local : 6 : 0 dot data> is connected along the horizontal line connected at the left side of the vertical line L 2. A label output is placed over the symbol of L E D display.

Return to Figure 12-39

OceanofPDF.com

Figure 12-40 Extended Description (Chapter 12)

Return to Figure 12-40

A check pattern is placed over the cylinder with 16 columns. The cylinder rotates in anti-clockwise direction. Red circles labeled as peg are placed within boxes over the cylinder. A label N O switch, and steps are placed at the left side and the right side of the N O switches, respectively. From top to bottom, numbers one, two, three, and four are shown at the right side of the cylinder. The pegs are placed over the cylinder at the second, fourth, seventh, eighth, eleventh, twelfth, thirteenth, fifteenth, and sixteenth positions of the row labeled as one. The pegs are placed over the cylinder at the first, second, fourth, fifth, seventh, ninth, eleventh, thirteenth, and fifteenth positions of the row labeled as two. The pegs are placed over the cylinder at the first, eighth, and fourteenth positions of the row labeled as three. The pegs are placed over the cylinder at second, fourth, seventh, ninth, eleventh, twelfth, thirteenth, fourteenth, and fifteenth positions of the row labeled as four.

Return to Figure 12-40

OceanofPDF.com

Figure 12-41 Extended Description (Chapter 12)

Return to Figure 12-41

From left to right, the 16 columns have labels 15, 14, 13, 12, 11, 10, nine, eight, seven, six, five, four, three, two, one, and zero. These labels are placed the top of the columns of the first table. All the 16 columns are labeled as zero. A label output O : 2 is placed at the left side of the first table. A label position is placed at the top-right corner of the second table. From top to bottom, labels start, step one, step two, step three, and step four are placed at the right side of the second table. From left to right, the first, the second, fourth, fifth, seventh, eight, tenth, eleventh, fifteenth, and the sixteenth columns of the second row are labeled as one. From left to right, the third, sixth, ninth, twelfth, thirteenth, and the fourteenth columns of the third row are labeled as one. From left to right, the first, third, fifth, seventh, ninth, eleventh, thirteenth and the fifteenth columns of the fourth row are labeled as one. From left to right, the first, second, third, fourth, ninth, tenth, eleventh, and the twelfth columns of the fifth row are labeled one. The remaining cells are labeled as zero.

Return to Figure 12-41

OceanofPDF.com

Figure 12-42 Extended Description (Chapter 12)

Return to Figure 12-42

The ladder diagram consists of three vertical lines. From left, the third vertical line is labeled as L 2. Two horizontal lines are connected between the first and the second vertical line. A label ladder logic program is placed over the first horizontal line connected between the first and the second vertical line. From left to right, X I C open symbol labeled as time dot D N, the first S Q O instruction, and an output energize E N is connected in series along the first horizontal line. The right side of the first S Q O instruction is connected to an output energize D N. From top to bottom, the first S Q O instruction is labeled as S Q O, sequencer output, array output_array, mask 1 6 # 7 F, dest output_card, control control_tag, length 4, and position 0. The series connection of the second S Q O instruction, and the second output energize E N is parallelly connected across the series connection of the first S Q O instruction and the first output energize E N. The right side of the second S Q O instruction is connected to the second output energize D N. From top to bottom, the second S Q O instruction is labeled as S Q O, sequencer output, array timing_array, mask 1 6 # 7 F F F, dest time dot P R E, control control_tag, length 4, and position 0. From left to right, X I O open symbol (highlighted in green) labeled as time dot D N, T O N instruction, and the third output energize E N are connected in series along the second horizontal line. The right side of the T O N instruction is connected to the third output energize D N. From top to bottom, the T O N instruction is labeled as T O N, timer on delay, timer time, preset 0, and accumulated 0. Six horizontal lines are connected to the left side of the vertical line L 2. From top to bottom, red light labeled 0 enclosed within square brackets, yellow light labeled 1 enclosed within square brackets, green light labeled 2 enclosed within square brackets, red light labeled 3 enclosed within square brackets, yellow light labeled 4 enclosed within square brackets, and green light labeled as 5 enclosed within square brackets. An illustration of a traffic light labeled north or south is placed at the right side of the first three lights. An illustration of a traffic light labeled east or west is placed at the right side of the last three lights. A label output_card local : 2 : 0 dot data is placed below the sixth horizontal line connected to the line L 2. The first table consists of two columns and six rows. From top to bottom, the first column is labeled as [-] timing_array, [+] timing_array [0], [+] timing_array [1], [+] timing_array [2], [+] timing_array [3], and [+] timing_array [4]. From top to bottom, the second column is labeled as tag editor or monitor tags, 0, 25,000, 5,000, 25,000, and 5,000. From top to bottom, the labels at the right side of the first table are positions, zero, one, two, three, and four. The second table consists of two columns and six rows. From top to bottom, the first column is labeled as [-] output_array, [+] output_array [0], [+] output_array [1], [+] output_array [2], [+] output_array [3], and [+] output_array [4]. From top to bottom, the second column is labeled as tag editor or monitor tags, _0000_0000_0000, _0000_0010_0001, _0000_0001_0001, _0000_0000_1100, and _0000_0000_1010. From top to bottom, the labels at the right side of the second table are positions, zero, one, two, three, and four.

Return to Figure 12-42

OceanofPDF.com

Figure 12-43 Extended Description (Chapter 12)

Return to Figure 12-43

There are four vertical lines. From the left, the first vertical line is labeled as L 1 and the fourth vertical line is labeled as L 2. From top to bottom, the N O proximity switch labeled S 1, and the second NO proximity switch labeled as S 2 are connected along the horizontal lines connected at the right side of the vertical line L 1. A label inputs is shown over the first horizontal line connected to the L 1. There is a horizontal line connected between the second and the third vertical lines. From left to right, X I C open symbol labeled as S 1, S Q O instruction, and an output energize E N are connected in series along the horizontal line. The right side of the S Q O instruction is connected to an output energize D N. Another X I C open symbol labeled as S 2 is parallelly connected across the X I C open symbol labeled as S 1. From top to bottom, the S Q O instruction is labeled as S Q O, sequencer output, file # B 3 : 0 (highlighted in green), mask F 0 F F (highlighted in green), dest O : 2(highlighted in red), control R 6 : 0, length 8, and position 0. An output module consisting of 16 terminals labeled from zero through 15 is placed between the third and the fourth vertical lines. A label output module O : 2 is placed above the output module. Terminals labeled from zero through two are connected to the vertical line L 2, through shaded lights. Terminals labeled from three through seven and from 12 through 15 are connected to the vertical line L 2, through shaded circles. A label output is placed above the connection of the output module and the line L 2. An arrow pointing right is from a label current step points towards a label pos two and is placed below the output module. The first table with a row of 16 columns is placed below the ladder diagram. From left to right, the first, fifth, sixth, seventh, eighth, and sixteenth columns have labels 15, 11, 10, nine, eight, and zero. These labels are placed at the top of the columns. The fifth, sixth, seventh, and eighth columns are unlabeled. The last four columns are labeled as one and are highlighted in red. The remains columns are labeled as zero and highlighted in red. A label destination O : 2 is placed at the right side of the first table. The second table with a row of 16 columns is placed below the first table. The fifth, sixth, seventh, and eighth columns are labeled as zero. The remaining columns are labeled as one. All the columns are highlighted in green. A label mask is placed at the right side of the second table. Arrows from the first four columns and the last eight columns of the second table points at the bottom of the first four columns and the last eight columns of the first table. The third table placed below the second table consists of nine rows and 16 columns. From top to bottom, labels zero, one, pos two (highlighted in yellow), three, four, five, six, seven, and eight are placed at the left side of the third table. A label pos is placed at the top-left corner of the third table. A label file # B 3 : 0 is placed at the top-right corner of the third table. From left to right, the third, fourth, eleventh, twelfth, fifteenth, and sixteenth columns of the second row are labeled as one. From left to right, the thirteenth, fourteenth, fifteenth, and sixteenth columns of the third row are labeled as one. From left to right, the first, second, ninth, tenth, thirteenth and the fourteenth columns of the fourth row are labeled as one. From left to right, the last eight columns of the fifth row are labeled one. From left to right, the ninth, and the tenth columns of the sixth row are labeled as one. From left to right, the third, fourth, and the last eight columns of the seventh row are labeled one. From left to right, the fourth, and the last five columns of the eighth row

are labeled one. From left to right, second, fourth, tenth, twelfth, fourteenth, and the sixteenth columns of the ninth row are labeled as one. The remaining cells are labeled as zero. The last four columns of the third row are highlighted in yellow. All the cells are highlighted in blue, except for the last four columns of the third row. Arrows from all the columns of the first row of the third table points at the bottom of all the columns of the second table.

Return to Figure 12-43

OceanofPDF.com

Figure 12-44 Extended Description (Chapter 12)

Return to Figure 12-44

A proximity switch is connected at the bottom-left side of the conveyor belt, such that it is between the first and the second product, from the left. A label inspection zone is placed at the left side of the proximity switch. A dashed arrow from the fifth product points downwards at the products highlighted in red and placed in a bin. Four stations are shown over the conveyor belt. From the left, the station one shown over the second product, station two shown over the third product, station three shown over the fourth product, and station four shown over the fifth product. A symbol of light labeled as reject lamp status is placed below first three stations.

Return to Figure 12-44

OceanofPDF.com

An Illustration Shows Graphic Extended Description (Chapter 13)

Return to An Illustration Shows Graphic

The illustration also shows a ladder at the right side. Three blocks of different colors are stacked vertically at the right side of the ladder. These blocks are connected to the blocks similar to them, which are present on the left side of the ladder.

Return to An Illustration Shows Graphic

OceanofPDF.com

Figure 13-1 Extended Description (Chapter 13)

Return to Figure 13-1

From left to right, the lower region is labeled as seven and eight. The door of the panel is labeled as 10. A terminal at the door is labeled as nine.

Return to Figure 13-1

OceanofPDF.com

Figure 13-3 Extended Description (Chapter 13)

Return to Figure 13-3

From top, three horizontal lines are labeled as three-phase supply. An N O switch and a fuse are connected in series along the three horizontal lines. A node at the second horizontal line is connected to the left end of the primary of the transformer. A node at the third horizontal line is connected to the right end of the primary of the transformer. A label disconnect switch is placed over the first horizontal line. The transformer is labeled as step-down isolation transformer. The secondary of the transformer is connected across two terminals labeled as 120 V A C. The left terminal is connected to the fuse, through a label L 1 and the right terminal is connected to a terminal of the panel ground bus. A ladder diagram with three horizontal lines connected between two vertical lines. The left end of the fuse is connected to the top end of the vertical line on the left side. The right terminal is also connected to the top end of the vertical line on the right side. A block labeled processor power supply is connected along the first horizontal line of the ladder diagram. A label L 1 is placed at its left side and a label neutral placed at its right side. The bottom-right side of the processor power supply is connected to the second terminal of the panel ground bus. The top side of the block labeled as master control relay is connected along the second horizontal line of the ladder diagram. From left to right, two N C pushbuttons labeled as emergency stop, bottom side of the master control relay, N O pushbutton labeled as start, and a relay coil M C R are connected in series along the THIRD horizontal line. An N O relay contact labeled as M C R is connected across the N O pushbutton. From top to bottom, below the third horizontal line an N O relay contact labeled M C R and four nodes are connected at the right side of the vertical line on the left side. The first two nodes are connected to the left side of a block labeled to common (L 1) of 120 V A C input devices, through fuse. Both the fuses have three vertically arranged dots between them. The last two nodes are connected to the left side of a block labeled to L 1 of 120 V A C output modules, through fuse. Both the fuses have three vertically arranged dots between them. The bottom end of the vertical line on the left side is connected to the left side of a block labeled as D C power supply, through a label L 1. From top to bottom, below the third horizontal line four nodes are connected at the left side of the vertical line on the right side. The first two nodes are connected to the right side of a block labeled to neutral connection of 120 V A C input modules. Both the lines connected to first two nodes have three vertically arranged dots between them. The last two nodes are connected to the right side of a block labeled to neutral of 120 V A C output devices. Both the lines connected to last two nodes have three vertically arranged dots between them. The bottom end of the vertical line on the right side is connected to the right side of a block labeled as D C power supply, through a label neutral. The right side of the D C power supply is also connected to the third terminal of the panel ground bus, through a label ground. The fourth terminal of the panel ground bus is not connected. The fifth terminal of the panel ground bus is connected to a grounding electrode. The bottom side of the D C power supply is connected to the vertical line on the left, through a label V positive and four nodes. The first two nodes are connected to the left side of a block labeled to V positive of 120 V A C input devices, through fuse. Both the fuses have three vertically arranged dots between them. The last two nodes are connected to

the left side of a block labeled to V positive of 120 V A C output modules, through fuse. Both the fuses have three vertically arranged dots between them. The bottom side of the D C power supply is also connected to the vertical line on the right side, through a label D C common and four nodes. The first two nodes are connected to the right side of a block labeled to common of D C input modules. Both the lines connected to first two nodes have three vertically arranged dots between them. The last two nodes are connected to the right side of a block labeled to common of D C output devices. Both the lines connected to first two nodes have three vertically arranged dots between them.

Return to Figure 13-3

OceanofPDF.com

Figure 13-4 Extended Description (Chapter 13)

Return to Figure 13-4

The ladder diagram shows a horizontal line connected between two vertical lines. From the left, the first vertical line is labeled as L 1, and the second vertical line is labeled as L 2. From left to right, N C pushbutton labeled stop, N O pushbutton labeled s start, motor starter coil, and N C relay contact labeled O L are connected in series along the horizontal line. An N O relay contact labeled as M is parallelly connected across the N O pushbutton labeled s start. A noise suppressor is parallelly connected across the motor starter relay coil. An illustration of actual starter coil is placed above its symbol. An illustration of noise suppressor is placed below its symbol.

Return to Figure 13-4

OceanofPDF.com

Figure 13-6 Extended Description (Chapter 13)

Return to Figure 13-6

The ladder diagram shows a horizontal line connected between two vertical lines. From the left, the first vertical line is labeled as L 1, and the second vertical line is labeled as N. From left to right, the line L 1 is connected to the vertical line N, through the N O proximity switch, a node connected to the third terminal of the input module, bleeder resistor, and a node labeled common is connected to the sixth terminal of the input module. A node close to the top end of the line N is grounded. An illustration of two-wire proximity switch is placed over its symbol. Current labeled leakage current flows into the upper end of the resistor, and flows away from the lower end of the resistor.

Return to Figure 13-6

OceanofPDF.com

Figure 13-7 Extended Description (Chapter 13)

Return to Figure 13-7

The ladder diagram shows a horizontal line connected between two vertical lines. From the left, the first vertical line is labeled as L 1, and the second vertical line is labeled as N. From left to right, the line L 1 is connected to the first terminal of the output module. The third terminal of the output module is connected to the vertical line N, through the high-impedance load. The bleeder resistor is parallelly connected across the high-impedance load. A node close to the top end of the line N is grounded. Current labeled leakage current flows into the left end of the bleeder resistor.

Return to Figure 13-7

OceanofPDF.com

Figure 13-8 Extended Description (Chapter 13)

Return to Figure 13-8

At the left side of the vertical line of the enclosure, four C P U or I/O racks are connected to a ground bus placed at the top-right side. The ground bus has seven slots. From the left, the first slot is connected to the ground for slot power supply. The second slot is connected to the C P U placed at the bottom-left side. The third slot is connected to the C P U placed at the top-left side. The fourth slot is connected to the C P U placed at the bottom-right side. The fifth slot is connected to the C P U placed at the top-right side. The sixth slot is connected to a node at the vertical line. An upwards pointing arrow labeled as an equipment grounding conductor points at the connection between the sixth slot and the node. The seventh slot is connected to grounding electrode system, which is at the right side of the vertical line. A downwards pointing arrow labeled grounding electrode system points at the connection between the seventh slot and the grounding electrode system.

Return to Figure 13-8

OceanofPDF.com

Figure 13-9 Extended Description (Chapter 13)

Return to Figure 13-9

A line passes through the center of the tab. A ring-shaped structure labeled as a star washer is placed in front of the chassis mounting tab, along the line. Next is a ring-shaped structure, whose bottom side is attached to a wire is placed along the line. The wire is labeled as equipment grounding conductor (ground lug with (eight A W G) wire), and the end of the wire is labeled as to ground bus. After this, a screw is placed along the line, such that its head is away from the parts connected in between.

Return to Figure 13-9

OceanofPDF.com

Figure 13-10 Extended Description (Chapter 13)

Return to Figure 13-10

From the left, the first slot is connected to the first source, and the second slot is connected to the second source. A label no ground loops points at the connection between these two sources and the slots. The third slot is connected to the third source, through arrow heads pointing towards the third slot. The third slot then points at the fourth slot, which is grounded from the top, through an arrow head pointing upwards at the third source. The third source is also grounded from the bottom, through an arrow head pointing towards the third source. A label ground loop formed is placed at the right side of the grounded third source, and the grounded fourth slot.

Return to Figure 13-10

OceanofPDF.com

Figure 13-11 Extended Description (Chapter 13)

Return to Figure 13-11

The top ends of the vertical lines are connected, through the battery source. A node at the positive terminal of the battery, which is connected to the left vertical line is labeled as L 1 (positive sign). A node at the negative terminal of the battery is labeled as L 2 (negative sign). The left vertical line is connected to the first terminal of the output module. The second terminal is connected to the right vertical line, through a node connected to the cathode of the diode, solenoid coil, and a node connected to the anode of the diode. A positive sign enclosed in brackets is placed at the right side of the second terminal. A symbol of reversed bias diode is placed below the diode. The sixth terminal of the output module is connected to the vertical line at the right side.

Return to Figure 13-11

OceanofPDF.com

Figure 13-12 Extended Description (Chapter 13)

Return to Figure 13-12

The ladder diagram consists of two vertical lines. From the left, the first and the second vertical lines are labeled as L 1 and L 2, respectively. The top ends of the vertical lines are connected, through the a c source. The line L 1 is connected to the first terminal of the output module, labeled as L 1. The second terminal of the output module is connected to the line L 2, through a common node, motor starter coil, and a node connected to the right end of the resistor R. The N O pushbutton labeled as P B is connected across output module and the common node, from the top. From bottom, the common node is connected to the left plate of the capacitor C. The right plate of the capacitor C and the left end of the resistor R are connected. The sixth terminal of the output module labeled as L 2 is connected to the line L 2. A vertical rectangle with two terminals at the bottom is placed below the capacitor. A horizontal rectangle with a terminal at the left and the right side is placed below the resistor.

Return to Figure 13-12

OceanofPDF.com

Figure 13-13 Extended Description (Chapter 13)

Return to Figure 13-13

The table consists of a column with 10 rows. From top to bottom, it is labeled as V A C or D A C, OUT 0, OUT 1, OUT 2, OUT 3, OUT 4, OUT 5, OUT 6, OUT 7, and COM. The V A C or V D C is connected to a terminal labeled positive D C or L 1. The row labeled as COM is connected to an extended terminal labeled D C COM or L 2, through a node. The node is connected to the row labeled as OUT 2, through an inductive load, and N O pushbutton. The M O V surge suppressor is parallelly connected across the inductive load, from the top. An illustration of the M O V surge suppressor along with two Zener diodes connected back to back are placed over the symbol of M O V surge suppressor.

Return to Figure 13-13

OceanofPDF.com

Figure 13-14(a-b) Extended Description (Chapter 13)

Return to Figure 13-14(a-b)

In part a, the computer screen shows a ladder diagram, which points at the P L C placed at the bottom-right side. This arrow is labeled as program from the computer is loaded into the P L C. An arrow labeled program that was in the P L C is lost points away from the bottom side of the P L C towards the bottom-right side.

In part b, the P L C placed at the bottom points at a ladder diagram on computer's screen. This arrow is labeled as program from the P L C is loaded into the computer. The top side of the computer points upwards and towards a label unsaved program in the computer is lost. A label program that was in the P L C remains unchanged is placed below the P L C.

Return to Figure 13-14(a-b)

OceanofPDF.com

Figure 13-15 Extended Description (Chapter 13)

Return to Figure 13-15

From top to bottom, the table is labeled as data files, cross reference, O : 0-OUTPUT, I : 1-INPUT, S 2-STATUS, B 3-BINARY, T 4-TIMER, C 5-COUNTER, R 6-CONTROL, N 7-INTEGER, and F 8-FLOAT. A symbol of folder is present at the left side of the first label. A symbol of file id present at the left side of the remaining labels is shown. The data file window shows a table labeled as input table at the top-left side. A cross symbol enclosed in a red colored square is placed at the top-right side. From left to right, the table has headings as 15, 14, 13, 12, 11, 10, nine, eight, seven, six, five, four, three, two, one, and zero. From top to bottom, the headings for the rows are I : 1 dot 0 and I : 2 dot 0. The cell under the heading four and the right side of the heading I : 1 dot zero is labeled as one. The remaining cells are labeled zero. From top to bottom, an arrow head pointing upwards, a bolded dash, and an arrow head pointing downwards is placed at the right side of the table. Three labels are present at the bottom of the table. From left to right, the labels are radix: binary (enclosed in a rectangle with a drop down box at the right side), table: (a drop down list), and forces (enclosed in a rectangle and is faded). From top to bottom, the drop down list is labeled as O : 0 : Output, I : 1 : Input, S 2 : Status, B 3 : Binary, T 4 : Timer, C 5 : Counter, R 6 : Control, N 7 : Integer, and F 8 : Float. A label address along with a rectangle at its right side is placed below the label radix.

Return to Figure 13-15

OceanofPDF.com

Figure 13-16 Extended Description (Chapter 13)

Return to Figure 13-16

The ladder diagram consists of two horizontal lines connected between two vertical lines. From left to right, X I C open symbol labeled as stop P B I : 1/1, X I C open symbol labeled as start P B I : 1/2, and an output energize labeled as O : 2/1 are connected in series along the first horizontal line labeled as zero. A rectangle labeled as motor control relay encloses the output energize. An X I C open symbol labeled as motor control relay O : 2/1 is parallelly connected across the X I C open symbol labeled as start P B. From left to right, the second X I C open symbol labeled as motor control relay O : 2/1 and output energize labeled as run pilot light O : 2/3 are connected in series along the second horizontal line labeled as one. The window shows symbol of minimize, maximize, and close a window are placed at the top-right side of the window. From top to bottom, the window is labeled as O : 2/1 - motor control relay, O T E - File # 2 MAIN_PROG - 0 (indented), X T C - File # 2 MAIN_PROG - 0, 1 (indented), O : 2/3 - Run Pilot Light, O T E - File # 2 MAIN_PROG - 1 (indented), I : 1/1 - Stop P B, X T C - File # 2 MAIN_PROG - 0 (indented), I : 1/2 - Start P B, and X T C - File # 2 MAIN_PROG - 0 (indented). Three labels enclosed in a rectangle are placed at the bottom. From left to right the labels are sort by symbol, refresh, and help. The first letter of each label is underlined.

Return to Figure 13-16

OceanofPDF.com

Figure 13-17 Extended Description (Chapter 13)

Return to Figure 13-17

Wires from both the terminals are connected to a horizontal rectangle, such that the positive terminal is connected to the top side and the negative terminal is connected to the bottom side.

Return to Figure 13-17

OceanofPDF.com

Figure 13-18 Extended Description (Chapter 13)

Return to Figure 13-18

From top to bottom, the second part is labeled as R U N, F L T, and B A T T. A green colored rectangle is placed at the right side of the label R U N. A red colored rectangle is placed at the right side of the label F L T. A red colored rectangle is placed at the right side of the label B A T T.

Return to Figure 13-18

OceanofPDF.com

Figure 13-19 Extended Description (Chapter 13)

Return to Figure 13-19

The input module consists of pair of squares, such that one pair is over the another. The table is shown below the squares. From top to bottom, the table is labeled as IN 0, IN 1, IN 2, IN 3, IN 4, IN 5, IN 6, IN 7, D C COM, and D C COM. The right side of the last two rows are connected. A node labeled common is connected to the left side of the last row of the table. The first square at top-left side is highlighted and a label L E D indicator not illuminated points at the square. A node labeled positive 24 V D C is connected to the left side of the first row of the table. The block labeled check input voltage level has a question mark enclosed in a rectangle at the top and two terminals at the bottom. The left terminal is negative and the right terminal is positive. The negative terminal is pointing at the left side of the last row of the table. The positive terminal is pointing at the left side of the first row of the table. The first and the last rows of the table are highlighted in blue.

Return to Figure 13-19

OceanofPDF.com

Figure 13-20 Extended Description (Chapter 13)

Return to Figure 13-20

From left to right, the headings along the first row are input device condition, input module status indicator, monitor display status indicator, and possible fault (s). The symbols at the sub-divided columns are of X I C input device and X I O input device. On including the sub-divided columns, below the headings there are five columns and six rows. From left to right, the second row is labeled as (symbol of N C limit switch) closed - ON 24 V D C input; ON; true (X I C symbol highlighted in green); false (X I O symbol); and none - correct indications. From left to right, the third row is labeled as (symbol of N O limit switch) open - OFF 0 V D C input; OFF; false (X I C symbol); true (X I O symbol highlighted in green); and none - correct indications. From left to right, the fourth row is labeled as (symbol of N C limit switch) closed - ON 24 V D C input; ON; false (X I C symbol); true (X I O symbol highlighted in green); sensor condition, input voltage, status indicator are correct, ladder instructions have incorrect indications, input module or processor fault. From left to right, the fifth row is labeled as (symbol of N C limit switch) closed - ON 0 V D C input; OFF; false (X I C symbol); true (X I O symbol highlighted in green); sensor indicator and instructions agree but not with the sensor condition, open field device or wiring. From left to right, the sixth row is labeled as (symbol of N O limit switch) open - OFF 0 V D C input; OFF; true (X I C symbol highlighted in green); false (X I O symbol); sensor condition, input voltage, status indicator are correct, ladder instructions have incorrect indications, input module or processor fault. From left to right, the seventh row is labeled as (symbol of N O limit switch) open - OFF 24 V D C input; ON; true (X I C symbol highlighted in green); false (X I O symbol); input voltage, status indicator, and ladder instructions agree but not with sensor condition, short circuit in the field device or wiring. The last four rows of the second column are highlighted in blue. The third and the fourth column of the fourth and the sixth rows are highlighted in blue.

Return to Figure 13-20

OceanofPDF.com

Figure 13-21 Extended Description (Chapter 13)

Return to Figure 13-21

Four pairs of four rectangles are placed at the left side of the highlighted section, such that the pairs are placed one over the another. From top to bottom, the right side of the highlighted section is labeled as (symbol of red colored rectangle), E, F, U, S, and E. The left side of the pairs is labeled as status indicators. A label electronic protection L E D points at the red colored rectangle.

Return to Figure 13-21

OceanofPDF.com

Figure 13-22 Extended Description (Chapter 13)

Return to Figure 13-22

The table consists of four columns and five rows. From left to right, the first row is labeled as output device condition, output module status indicator, monitor display status indicator, and fault (s). From left to right, the second row is labeled as energized - ON, ON, true (symbol of output energize highlighted in green), and none - correct indication. From left to right, the third row is labeled as de-energized - OFF, OFF, false (symbol of output energize), and none - correct indication. From left to right, the fourth row is labeled as de-energized - OFF, ON, true (symbol of output energize highlighted in green), and output instruction and status indicator agree but the field device does not; open field device or wiring; module circuit or fuse. From left to right, the fifth row is labeled as de-energized - OFF, OFF, true (symbol of output energize highlighted in green), and field device status and status indicator agree but the output condition does not; module circuit or fuse. The output module consists of two pairs of four squares, such that the pairs are placed one over the another. The first square is shaded in red. A label output module status indicator ON points at the red square. A table with a column of 10 rows is placed below the squares. From top to bottom, the table is labeled as V A C, OUT 0, OUT 1, OUT 2, OUT 3, OUT 4, OUT 5, OUT 6, OUT 7, and A C COM. The right side of the row labeled V A C is connected to the label L 1. The right side of the row labeled as OUT 0 is connected to a common node, through an opening in wiring, and an output device. The right side of the row labeled as A C COM is connected to a label L 2, through the common node. An illustration of open in wiring is placed above its symbol in the circuit. An illustration of a device is placed at the right side of the output device. A double headed arrow points at the illustration and the output device. The left side of the fourth row of the table points at the left side of the output module, through a dashed arrow.

Return to Figure 13-22

OceanofPDF.com

Figure 13-23 Extended Description (Chapter 13)

Return to Figure 13-23

Indentation increases from top to bottom. A dashed sign is enclosed in a rectangle and is placed on the left side of the first three labels. At the bottom of the window, an arrowhead pointing left and an arrowhead pointing right are placed together on the left side. A label verify results is placed at the right side of the arrowhead pointing right.

Return to Figure 13-23

OceanofPDF.com

Figure 13-24 Extended Description (Chapter 13)

Return to Figure 13-24

From left to right, file, edit, view, search, comms, tools, and window are the labels along the first section. First letter of each word is underlines. An underlined letter H is also placed at the end. From left to right, a symbol of file, a symbol of folder, a symbol for saving the file, a symbol for print, a symbol of cut, a symbol of copy, a symbol of paste (faded), a symbol of undo (faded), a symbol of redo (faded), and a label B 3 : 1/0 (enclosed in a rectangle) are placed along the second section. From left to right, a rectangle labeled as offline along with a drop down button at the right side, a rectangle labeled no forces along with a drop down button at the right side, and an illustration of a ladder diagram labeled as node : 1o are placed along the third section. A drop down list labeled as go online, download, and upload is placed below the label offline. A rectangle labeled forces enabled (highlighted in yellow) with a drop down button at the right side is placed below the rectangle labeled no forces along with a drop down button at the right side. From left to right, a rectangle along with symbol of minimize, maximize, and closing a file is placed along the fourth section. A rectangle labeled LAD 2 is placed at the right side of the rectangle present along the fourth horizontal line.

Return to Figure 13-24

OceanofPDF.com

Figure 13-25 Extended Description (Chapter 13)

Return to Figure 13-25

The toolbar shows an arrow-head pointing left at the top-left side and an arrow-head pointing right at the top-right side. Eight labels enclosed in rectangles are placed relatively on the left side between the arrow-heads. From left to right, the labels are J M P, L B L, J S R, R E T, S B R, T N D (highlighted in blue), M C R, and S U S. From bottom-left to bottom-right, an arrow-head facing left, an arrow-head facing right, labels: program control (highlighted in blue), ascll control, ascll string, and micro, are shown. The ladder diagram consists of three vertical lines. From left, the first vertical line is labeled as L 1. The line L 1 shows three dots at the end and an arrow pointing downwards is connected to the third dot, from the top. The N O switch labeled as S W is connected along a horizontal line connected at the right side of the line L 1. A label input is placed above the symbol of the switch. Two horizontal lines are connected between the second and the third vertical lines. From left to right, X I C open symbol, and an output energize are connected in series along the first horizontal line. From left to right, X I C open symbol labeled as S W, and an output energize T N D (highlighted in blue) are connected in series along the second horizontal line. A label temporary end is placed below the first horizontal line. Two dots arranged vertically are labeled as main program and placed above the first horizontal line. A downwards pointing arrow is connected to the second dot. Two dots arranged vertically are labeled as remainder of main program and placed below the second horizontal line. A downwards pointing arrow is connected to the second dot.

Return to Figure 13-25

OceanofPDF.com

Figure 13-26 Extended Description (Chapter 13)

Return to Figure 13-26

The toolbar shows an arrow-head pointing left at the top-left side and an arrow-head pointing right at the top-right side. Eight labels enclosed in rectangles are placed relatively on the left side between the arrow-heads. From left to right, the labels are J M P, L B L, J S R, R E T, S B R, T N D, M C R, and S U S (highlighted in blue). From bottom-left to bottom-right, an arrow-head facing left, an arrow-head facing right, labels: program control (highlighted in blue), ascll control, ascll string, and micro, are shown. The ladder diagram consists of three vertical lines. From left, the first vertical line is labeled as L 1. The line L 1 shows three dots at the top end and an arrow pointing downwards is connected between the third dot and the top end of the line L 1. Three horizontal lines are connected at the right side of the vertical line L 1. From top to bottom, N C switch labeled as A, N C switch labeled as B, and N O switch labeled as C are connected along the horizontal lines connected at the right side of the line L 1. A label inputs is placed above the symbol of the N C switch A. One horizontal line is connected between the second and the third vertical lines. From left to right, X I C open symbol (highlighted in green) labeled as A, X I C open symbol (highlighted in green) labeled as B, X I O open symbol (highlighted in green) labeled as C, and S U S instruction are connected in series along the horizontal line. From top to bottom, the S U S instruction is labeled as S U S, suspend, and suspend I D 100. Two dots arranged vertically are labeled as main program and placed above the S U S instruction. A downwards pointing arrow is connected to the second dot.

Return to Figure 13-26

OceanofPDF.com

Figure 13-27 Extended Description (Chapter 13)

Return to Figure 13-27

From left, the first vertical line is labeled as L 1 and the fourth line is labeled as L 2. Two horizontal lines are connected at the right side of the line L 1. From top to bottom, N C switch labeled as I : 1/1, and N O switch labeled as I : 1/2 are connected along the horizontal lines. A label inputs is placed above the first horizontal line connected to the line L 1. Two horizontal lines are connected between the second and the third vertical lines with a label ladder logic program at the top. From left to right, X I C open symbol (highlighted in green) labeled as I : 1/1, and an output energize labeled as O : 2/1 are connected in series along the first horizontal line. From left to right, X I C open symbol labeled as I : 1/2, and the second output energize labeled as O : 2/1 are connected in series along the second horizontal line. A double headed arrow points at both the output energize and is labeled as same address. The light labeled as P L at the top and (OFF) at the bottom is placed along a horizontal line connected at the left side of the line L 2. A label output is placed over the symbol of the light.

Return to Figure 13-27

OceanofPDF.com

Figure 13-28 Extended Description (Chapter 13)

Return to Figure 13-28

A trapezium is placed at the top-right corner of both the input and the output modules. The 10 terminals for both the modules are enclosed in a vertical rectangle at the bottom. Nine circles are shows at the left side of the modules with a label status at the left side. The input module is labeled as hardware comparison test - compare state of I/O as stored in memory to actual state. The fourth circle is highlighted in blue and labeled as off for the input module. The fourth terminal of the input module is connected to the label L 1 at the left side, through an N O limit switch (encircled) labeled as input device and open-off. The tenth terminal of the input module is connected to the label L 2 at the left side. The fourth circle is highlighted in red and labeled as on for the output module. The first terminal of the output module is connected to the label L 1 at the right side. The tenth terminal of the output module is connected to the label L 2 at the right side, through a node. The node is connected to the fifth terminal of the output module, through a relay coil shaded red (encircled) and labeled as load energized on. The block labeled memory has a label logic observation - determine validity of decisions made by processor at the bottom. From top to bottom, the block labeled memory shows a window screen labeled as output image table, a window screen labeled as input image table, and a ladder diagram labeled as user program. The output image table shows a window screen with a label output table at the top. The input image table shows a window screen with a label input table at the top. For both the tables, headings for the columns are from 15 through zero, as observed from left to right. For the output table, headings for the rows are O : 0/, O : 1/, O : 2/, O : 3/, and O : 4/, as observed from top to bottom. For the input table, headings for the rows are I : 0/, I : 1/, I : 2/, I : 3/, and I : 4/, as observed from top to bottom. For both the tables, the labels are either zero or one, except for the fifth row of the input table as it is unlabeled. For the output table, the label at the third row of the thirteenth column is one, which is encircled. For the input table, the label at the third row of the thirteenth column is zero, which is encircled. The ladder diagram labeled as user program consists of five horizontal lines connected between two vertical lines. From left to right, three X I C symbols, and an output energize is connected in series along the first horizontal line. From left to right, an X I O symbol X I C symbol, the second X I O symbol, and an output energize are connected in series along the second horizontal line. A series connection of two X I C symbols are connected across the series connection of the first X I O symbol, and the X I C symbol connected along the second horizontal line. From left to right, four X I O symbols, and an output energize are connected in series along the third horizontal line. From left to right, an X I C symbol and an output energize are connected in series along the fourth horizontal line. From left to right, an X I O symbol (highlighted in green and encircled) and an output energize (highlighted in green and encircled) are connected in series along the fifth horizontal line.

Return to Figure 13-28

OceanofPDF.com

Figure 13-29 Extended Description (Chapter 13)

Return to Figure 13-29

The output module has two columns of nine terminals. A block labeled output is placed at the top of the output module. The block consists of 16 cells placed in four by four arrangement. From top to bottom, each column is labeled as zero, one, two, three; four, five, six, seven; eight, nine, 10, 11; and 12, 13, 14, 15. Three and eight are highlighted in blue. A block with two columns is placed at the right side of the output module. From top to bottom, the column on the left side is labeled as V A C (highlighted in blue), out one, out three (highlighted in blue), out five, out seven, out nine, out 11, out 13, and out 15. From top to bottom, the column on the right side is labeled as out zero, out two, out four, out six, out eight (highlighted in blue), out 10, out 12, put 14, A C COM (highlighted in blue). The first terminal of the left column of the output module is highlighted in red and connected to the label L 1. The third terminal of the left column of the output module is connected to the ninth terminal of the right column of the output module, through the red light labeled as O : 2/3, and a node. The node is labeled as L 2 and is connected to the fifth terminal of the right column of the output module, through a green light labeled as O : 2/8. All the terminals connected are highlighted in red.

The steps of the flow chart are as follows:

1. output module, if yes then move to second step.
2. output device would not.
3. If the output device would not turn off, then check for output status indicators.
4. If the output status indicators are OFF, then check I/O device wiring.
5. If after the third step, the output indicators are ON, then display rung on screen.
6. Check if the output instruction is true.
7. If yes, then
8. If no, then check I/O modules, swing arm, power, and rack.
9. If at the second step, the output device would not turn on, then check output status indicators.
10. If the output status indicators are ON, then check I/O device wiring.
11. If after the tenth step, the output indicators are OFF, then go to blown fuse indicator.
12. If the blown fuse indicator is on, then find out why repair or replace.
13. If the blow fuse indicator is OFF, then display rung on screen.
14. Check if the output instruction is true.
15. If no, then
16. If yes, then check I/O modules, swing arm, power, and rack.

Return to Figure 13-29

OceanofPDF.com

Figure 13-30 Extended Description (Chapter 13)

Return to Figure 13-30

In both the cases, number two is highlighted, whereas it is highlighted in red in the first row and highlighted in green in the second row. The table in the first row is labeled as ON at the top. The table in the second row is labeled as OFF at the top. The table provides various input troubleshooting guidelines for various situations.

1. If your input circuit L E D is on and your input device is on or closed or activated and your input device does not turn off, then the probable cause is that the device is shorted or damaged.
2. If your input circuit L E D is on and your input device is on or closed or activated and your program operates as though it is off, then the probable cause is either the input circuit wiring or module or the input is forced off in program.
3. If your input circuit L E D is on and your input device is off or open or deactivated and your program operates as though it is on and/or the input circuit will not turn off, then the probable cause is either the input device off-state leakage current exceeds input circuit specification or the input device is shorted or damaged or the input circuit wiring or module.
4. If your input circuit L E D is off and your input device is on or closed or activated and your program operates as though it is off and/or the input circuit will not turn on, then the probable cause is that the input circuit is incompatible or the low voltage across the input or the input circuit wiring or the module or the input signal turn-on time too fast for input circuit.
5. If your input circuit L E D is off and your input device is on or closed or activated and your input device will not turn on, then the probable cause is that the input device is shorted or damaged.
6. If your input circuit L E D is off and your input device is off or open or deactivated and your program operates as though it is on, then the probable cause is either the input is forced on in program or the input circuit wiring or module.

In the first and the second situations, an N C limit switch and a an X I C symbol (highlighted in green) are shown. In the third situation, an N O limit switch and a an X I C symbol are shown. In the third situation, an N C limit switch and a an X I C symbol (highlighted in green) are shown. In the fifth and the sixth situations, an N O limit switch and a an X I C symbol are shown.

Return to Figure 13-30

OceanofPDF.com

Figure 13-31 Extended Description (Chapter 13)

Return to Figure 13-31

In both the cases, number eight is highlighted, whereas it is highlighted in red in the first row and highlighted in green in the second row. The table in the first row is labeled as ON at the top. The table in the second row is labeled as OFF at the top. The table provides various output troubleshooting guidelines for various situations.

1. If your output circuit L E D is ON and your output device is on or energized and your program indicates that the output circuit is off or the output circuit will not turn off, then there are three probable causes. The first is programming problem, for that check for duplicate outputs and addresses or if using subroutines, outputs are left in their last state when not executing subroutines or use the force function to force output off. If using the force function does not force the output off, output circuit is damaged. If the output does force off, then check again for logic or programming problem. The second probable cause is that the output is forced on in program. The third probable cause is the output circuit wiring or module.
2. If your output circuit L E D is ON and your output device is off or de-energized and your output device will not turn on and the program indicates that it is on, then there are three probable causes. The first is the low or no voltage across the load. The second is the output device is incompatible for which check specifications and sink or source compatibility (if dc output). The third is the output circuit wiring or module.
3. If your output circuit L E D is OFF and your output device is on or energized and your output device will not turn off and the program indicates that it is off, then there are four probable cause. The first is that the output device is incompatible. The second is that the output circuit off-state leakage current may exceed output device specification. The third is the output circuit wiring or module. The fourth is that the output device is shorted or damaged.
4. If your output circuit L E D is OFF and your output device is off or de-energized and your program indicates that the output circuit is on or the output circuit will not turn on, then there are three probable causes. The first is the programming problem, for which check for duplicate outputs and addresses or if using subroutines, outputs are left in their last state when not executing subroutines or use the force function to force output on. If using the force function does not force the output on, then the output circuit is damaged. If the output does force on, then check again for logic or programming problem.

In case of the first and the third situations, a glowing light and an output energize are shown. In case of the second and the fourth situations, an off light and an output energize (highlighted in green) are shown.

Return to Figure 13-31

OceanofPDF.com

Figure 13-32 Extended Description (Chapter 13)

Return to Figure 13-32

From the left, the first section of the P L C is labeled as processor. A vertical rectangle with curved edges is placed at the bottom of the label processor. This rectangle is connected to the bottom terminal of the block labeled personal computer. The personal computer shows three horizontal rectangles, which are connected through double headed arrows. From top to bottom, the horizontal rectangles are labeled as R S logix software, R S linx software, and C O M port.

Return to Figure 13-32

OceanofPDF.com

Figure 13-33 Extended Description (Chapter 13)

Return to Figure 13-33

From top to bottom, the C P U is divided into four sections, The first section is labeled as S L C 5/0 4 C P U. The second section shows run, a rectangle, another rectangle, force; F L T, a rectangle, another rectangle, D H (with positive sign at the bottom); B A T T, a rectangle, another rectangle, R 3 2 3 2; and run, rem, prog. The third section shows a circle from which a rectangular region of height equal to the radius is removed. A line from the top-left side of the circle is connected to the label run in the above section. A line from the top side of the circle is connected to the label rem in the above section. A line from the top-right side of the circle is connected to the label prog in the above section. From top to bottom, the fourth section shows a distorted rectangle with three circles enclosed in it, two concentric circles enclosed in a square, and a jack shaped as a vertical trapezium enclosed in a vertical rectangle. The trapezium encloses four circles at the left and five circles at the right. A circle is at the top and another at the bottom of the trapezium. There are three lines connected to three points at the lower part of the computer. A red line connects the top point at the right side of the computer to the middle of the trapezium in the C P U. It is labeled as two send near the computer and receive three near the C P U. A blue dashed line connects an in between point at the right side of computer to the top of the trapezium in the C P U. It is labeled as three receive near the computer and send two near the C P U. A green line connects the bottom point at the right side of the computer to the bottom of the trapezium in the C P U. It is labeled as seven near the computer and near the C P U. It is labeled as ground (common) in the middle of the connection.

Return to Figure 13-33

OceanofPDF.com

Figure 13-34 Extended Description (Chapter 13)

Return to Figure 13-34

From left to right, file, view, communications, station, D D E / O P C, security, window, and help are the labels along the first horizontal section. From the left, a symbol of charts is labeled as R S who, a symbol of two plugs connected to each other, and another symbol are placed along the second horizontal section. The third horizontal section has a label R S who-1. From left to right, the fourth horizontal section shows a checked box, label auto browse, label refresh (enclosed in a rectangle and is faded), two symbols, and a label browsing node one found. In the left panel, there are five lines of labels. From left to right, the first line shows an negative sign enclosed in a rectangle connected to a symbol of computer through dashed line, workstation, and ADMIN-P C. From left to right, the second line shows a positive sign enclosed in a rectangle connected to a symbol of network tree through dashed line, linx gateways, and ethernet. From left to right, the third line shows an negative sign enclosed in a rectangle connected to a symbol of network tree with a block shaded through dashed line, and labels computer, D H-485 (highlighted in a shaded rectangle). From left to right, the fourth line shows a vertical line from the bottom of the tree network symbol in the third line connected to a symbol of computer through dashed line, 0 0, workstation, and D F 1-C O M 3. From left to right, the fifth line shows a vertical line from the bottom of the tree network symbol in the third line connected to a symbol of P L C through dashed line, 0 1, MicroLogix 1000, and P L C. From top to bottom, the indentation increases, such that the second and the third line are indented equally. The fourth and the fifth line are indented equally. From left to right, the right panel shows a symbol of computer labeled as 0 0 D F 1-C O M 3 workstation and symbol of P L C labeled as 0 1 P L C.

Return to Figure 13-34

OceanofPDF.com

Figure 13-35 Extended Description (Chapter 13)

Return to Figure 13-35

From left, the first vertical line is labeled as L 1 and the fourth line is labeled as L 2. Two horizontal lines are connected at the right side of the line L 1. From top to bottom, N O pushbutton labeled as P B 1, and N O pushbutton labeled as P B 2 are connected along the horizontal lines. A label inputs is placed above the first horizontal line connected to the line L 1. Three horizontal lines are connected between the second and the third vertical lines with a label ladder logic program at the top. From left to right, X I C open symbol labeled as P B 1, and an output energize labeled as pump 1 are connected in series along the first horizontal line. From left to right, X I C open symbol labeled as pump 1, and an output energize labeled as P B 1 are connected in series along the second horizontal line. From left to right, X I C open symbol labeled as P B 2, and an output energize labeled as pump 2 are connected in series along the third horizontal line. Three horizontal lines are connected at the left side of the line L 2. From top to bottom, motor starter coil M 1 labeled as pump 1, light labeled as P L 1, and motor starter coil labeled as pump 2 are connected along the horizontal lines connected to the line L 2. A label outputs is placed over the first horizontal line connected to the line L 2.

Return to Figure 13-35

OceanofPDF.com

Figure 13-36 Extended Description (Chapter 13)

Return to Figure 13-36

The ladder diagram consists of 4 vertical lines. From left, the first vertical line is labeled as L 1 and the fourth line is labeled as L 2. The N O switch labeled as input A is connected along the horizontal line connected at the right side of the line L 1. A label input is placed above the horizontal line connected to the line L 1. Four horizontal lines are connected between the second and the third vertical lines with a label ladder logic program at the top. From left to right, X I C open symbol labeled as input A, X I O open symbol (highlighted in green) labeled as B 3 : 1 at the top and zero at the bottom, T O N instruction, and an output energize labeled E N are connected in series along the first horizontal line. The right side of the T O N is connected to an output energize D N. From top to bottom, the T O N instruction is labeled as T O N, timer on delay, timer T 4 : 2, time base 0.1, preset 50, and accumulated 0. From left to right, X I C open symbol labeled as T 4 : 2 at the top and D N at the bottom, and an output energize labeled as P L 1 are connected in series along the second horizontal line. From left to right, the second X I C open symbol labeled as T 4 : 2 at the top and D N at the bottom, the second T O N instruction, and an output energize labeled E N are connected in series along the third horizontal line. The right side of the second T O N is connected to an output energize D N. From top to bottom, the second T O N instruction is labeled as T O N, timer on delay, timer T 4 : 3, time base 0.1, preset 100, and accumulated 0. From left to right, N O relay contact labeled as T 4 : 2 at the top and D N at the bottom, and an output energize labeled as B 3 : 1 at the top and 0 at the bottom are connected in series along the fourth horizontal line. The light labeled as P L 1 is placed along a horizontal line connected at the left side of the line L 2. A label output is placed over the symbol of the light.

Return to Figure 13-36

OceanofPDF.com

Figure 14-1 Extended Description (Chapter 14)

Return to Figure 14-1

A tray labeled engine block has two parts and it points at the bottom-left corner of the conveyor belt. An arrow points at the left side of the engine block. From left to right, a tray with a part, an arrow pointing right, two trays each carrying a part are placed over the station labeled automated machine assembly, and two trays each carrying a part are placed over the station labeled hand assembly, are shown along the bottom edge of the conveyor belt. An arrow pointing upwards is placed along the right edge of the conveyor belt. From right to left, an arrow pointing left, two trays carrying a part are placed over the station labeled adjustments checks, an arrow pointing left, and a tray with a part, are placed along the top side of the conveyor belt. The top-left corner of the conveyor belt points at a tray carrying two parts labeled as completed engines. An arrow points away from the left side of the tray labeled completed engines. From top to bottom, a downwards pointing arrow, an empty tray, and a downwards pointing arrow, are placed along the left edge of the conveyor belt.

Return to Figure 14-1

OceanofPDF.com

Figure 14-2 Extended Description (Chapter 14)

Return to Figure 14-2

Two green triangles joined, through a vertex are labeled as material input one. Two blue triangles joined, through a vertex are labeled as material input two. Two red triangles joined, through a vertex are labeled as material input three. From top to bottom, the material input one, the material input two, and the material input three are connected from the right side and are transferred to the first U-shaped container, through a flow meter and two triangles joined through a vertex. The first U-shaped container has a mixer motor, and a heater. The mixture in the first U-shaped container has labels heat batch and mix batch. From the first U-shaped container, the mixture is transferred to the second U-shaped container, through two triangles joined through a vertex. The second U-shaped container has a tube attached to it on the right side. A breakage in the tube is shown.

Return to Figure 14-2

OceanofPDF.com

Figure 14-3 Extended Description (Chapter 14)

Return to Figure 14-3

The sensor is connected to the right side of the P L C. The left side of the P L C is connected to the cutter, through a block labeled Opto module and the cut pieces. The right side of the P L C is also connected to a block over which the sensor is placed, through a motor and a leadscrew. The lead screw passes through the bottom region of the block carrying the sensor. A double-headed arrow is placed below the leadscrew. A device labeled operator interface is placed below the P L C.

Return to Figure 14-3

OceanofPDF.com

Figure 14-4 Extended Description (Chapter 14)

Return to Figure 14-4

A box is placed before the sensor on the conveyor belt. A box moves along the same conveyor belt after passing through the rotatory gate. Two boxes are placed on another conveyor belt joined at the junction, where the rotatory gate is placed.

Return to Figure 14-4

OceanofPDF.com

Figure 14-5 Extended Description (Chapter 14)

Return to Figure 14-5

From left to right, the host computer is connected to the human machine interface, through three nodes. The three nodes are highlighted in blue and labeled as communication network. All the three nodes are connected to three different machines, though a P L C.

Return to Figure 14-5

OceanofPDF.com

Figure 14-8 Extended Description (Chapter 14)

Return to Figure 14-8

The circle with a cross over it is labeled as error amplifier. The cross divides the circle into four regions. There is a positive sign in the left region of the circle and a negative sign in the bottom region. A label set-point (highlighted in yellow) points at the left side of the error amplifier. The right side of the error amplifier leads to the left side of a block labeled controller, through a label error signal. The block labeled controller leads to a block labeled output actuator, which results in a block labeled process. A node labeled as disturbance points at the process, from the top. The bottom side of the block labeled process leads to the right side of the block labeled input sensors, through a label feedback path. The left side of the input sensors lead to the bottom side of the error amplifier, through a label process variable signal.

Return to Figure 14-8

OceanofPDF.com

Figure 14-9 Extended Description (Chapter 14)

Return to Figure 14-9

The inverted container is filler with detergent.

Return to Figure 14-9

OceanofPDF.com

Figure 14-10 Extended Description (Chapter 14)

Return to Figure 14-10

The illustration shows a circle and three blocks. The circle is labeled as error amplifier. There is a positive sign at the top-left side of the circle and a negative sign at the bottom-left side of the circle. A label input desired weight (highlighted in yellow) points at the left side of the error amplifier. The right side of the error amplifier leads to the left side of a block labeled controller. The block labeled controller leads to a block labeled filling process, which points at a label filled container (highlighted in yellow). A node at the right side of the block labeled filling process points at a block labeled weight sensor. The left side of the weight sensor lead to the bottom side of the error amplifier, through a label feedback. For the block diagram a loop is formed. A label set-point (highlighted in yellow) points downwards at a block labeled controller. The left side of the controller points at the bottom side of the block labeled as final control element. The right side of the final control element points at the left side of the block labeled process. The right side of the block labeled process points at the top side of a block labeled measurement (sensor). A label solenoid is placed at the left side of the final control element. A label container filling is placed at the top of the block labeled process. A label scale is placed at the right side of the block labeled measurement (sensor).

Return to Figure 14-10

OceanofPDF.com

Figure 14-11 Extended Description (Chapter 14)

Return to Figure 14-11

The liquid enters the tank through a pipeline, which has valve at its end. A thumbwheel switch labeled as set-point temperature points at the left side of a block labeled on or off controller. The right side of the block labeled on or off controller points at the top side of the valve at the pipeline of steam. A temperature sensor it the tank points at the left side of the block labeled on or off controller. The tank is connected to a pipeline at the bottom-right side, through another valve. From left to right, the blocks in the thumbwheel switch are labeled as one, three, and six.

Return to Figure 14-11

OceanofPDF.com

Figure 14-12 Extended Description (Chapter 14)

Return to Figure 14-12

In the timing diagram at the top, a dashed horizontal line labeled as set-point is placed at a point on the vertical axis. The curve for the variable process gradually increases and decreases across the set-point, such that it shows three upward peaks and two downward peaks. In the timing diagram at the bottom, the vertical axis has two labels off and on, from bottom to top. A rectangular wave is shown with two rectangular peaks. The second peak is wider than the first peak. Vertical dashed lines from the peaks enters the timing diagram at the top and that region is highlighted. In the region highlighted in the first timing diagram, the downward peaks are shown.

Return to Figure 14-12

OceanofPDF.com

Figure 14-13 Extended Description (Chapter 14)

Return to Figure 14-13

From top to bottom, the illustration shows a trapezium labeled valve actuator, a vertical rectangle which shows actuator steam transferred along a vertical line from the top, and a T-shaped pipeline. The table has two rows. The first row is divided into two columns. From the left, the columns of the first row are labeled as actuator current (milli-ampere) and valve response (percent open). The second rows shows labels along nine lines. The labels along each line are separated through 12 dots. From top to bottom, the labels on the left side of the second row are four, six, eight, 10, 12, 14, 16, 18, and 20. From top to bottom, the labels on the right side of the second row are zero, 12.5, 25, 37.5, 50, 62.5, 75, 87.5, and 100. The label on the left side of the second row lies below the first column of the first row. The label on the right side of the second row lies below the second column of the first row.

Return to Figure 14-13

OceanofPDF.com

Figure 14-14 Extended Description (Chapter 14)

Return to Figure 14-14

All the timing diagram show rectangular wave. Except, for the one on the top, the remaining timing diagrams show two rectangular peaks. The timing diagram on the top shows one rectangular peak, which is the widest of all. From bottom to top, the width of the peaks increases. The rising edge of all the peaks in all the timing diagrams is shown at the same time. From bottom to top, the timing diagrams are labeled as 12.5 percent – 25 watts, 25 percent – 50 watts, 50 percent – 100 watts, and 100 percent – 200 watts.

Return to Figure 14-14

OceanofPDF.com

Figure 14-15 Extended Description (Chapter 14)

Return to Figure 14-15

From left to right, the first three columns of the second row are labeled as percent on, on time (seconds), and off time (seconds). The first two rows of the third column are merged and labeled as temp (degree Fahrenheit). The last two columns of the first row are merged and labeled as 4 through 20 milli-amperes proportional. From left to right, the last two columns of the second row are labeled as output level and percent output. From left to right, the third row is labeled as zero, zero, 20, over 540, four milli-amperes, and zero. From left to right, the fourth row is labeled as zero, zero, 20, 540, four milli-amperes, and zero. From left to right, the fifth row is labeled as 12.5, 2.5, 17.5, 530, six milli-amperes, and 12.5. From left to right, the sixth row is labeled as 25, five, 15, 520, eight milli-amperes, and 25. From left to right, the seventh row is labeled as 37.5, 7.5, 12.5, 510, 10 milli-amperes, and 37.5. From left to right, the eighth row (highlighted in red) is labeled as 50, 10, 10, 500, 12 milli-amperes, and 50. From left to right, the ninth row is labeled as 62.5, 12.5, 7.5, 490, 14 milli-amperes, and 62.5. From left to right, the tenth row is labeled as 75, 15, five, 480, 16 milli-amperes, and 75. From left to right, the eleventh row is labeled as 87.5, 17.5, 2.5, 470, 18 milli-amperes, and 87.5. From left to right, the twelfth row is labeled as 100, 20, zero, 460, 20 milli-amperes, and 100. From left to right, the thirteenth row is labeled as 100, 20, zero, under 460, 20 milli-amperes, and 100. The first three columns are highlighted in green, and the last two columns are highlighted in yellow. A block labeled as five, zero, and zero points at the left side of the eight row.

Return to Figure 14-15

OceanofPDF.com

Figure 14-16 Extended Description (Chapter 14)

Return to Figure 14-16

A horizontal dashed line is placed at the label set-point along the vertical axis. The are under the line is shaded. A curve increases and peaks at a point above the set-point, then decreases and finally increases to saturate at a point just below the set-point. This point of saturation is labeled as offset signal.

Return to Figure 14-16

OceanofPDF.com

Figure 14-17 Extended Description (Chapter 14)

Return to Figure 14-17

There is a positive sign at the top-left side of the circle and a negative sign at the bottom-left side of the circle. The right side of the circle points at a label error, which further points at the base of a triangle labeled as proportional. A triangle labeled as integral is connected across the triangle labeled as proportional, from the top. A triangle labeled as derivative is connected across the triangle labeled as proportional, from the bottom. The vertex opposite to base of all the three triangles points at another circle enclosing a positive sign. The right side of the second circle points at a block labeled process. The right side of the block labeled process points at the right, through a node pointing at the bottom side of the first circle.

Return to Figure 14-17

OceanofPDF.com

Figure 14-18 Extended Description (Chapter 14)

Return to Figure 14-18

The top-left side of the vessel is connected to a pipe at which an arrow labeled process supply points from the left side. The bottom-right side of the vessel is connected to a pipeline, through a valve. The right side of this pipeline points at a label process output. The top-right corner of the vessel points at the top side of the PL C, through a circle labeled as pressure sensor and transmitter enclosing a label P T, and through a label process variable. The column (highlighted in red) of the P L C at which the vessel points is labeled as analog input module. A column (highlighted in green) at the left side of the analog input module is labeled as C P U. A column (highlighted in red) at the right side of the analog input module is labeled as analog output module, which points at the top side of the valve connected to vessel. The P I D loop is a rectangle is divided into three vertical sections, such that the first section from the left is the widest. From left to right, the vertical sections are labeled as C P U, analog input module, and the analog output module. In the section labeled as C P U, an arrow labeled as set-point points at the left side of a circle enclosing the letter S. The right side of the circle points at the left side of a block labeled as P I D loop calculation (highlighted in green), through a label error. A terminal in the section labeled as analog input module points at the bottom of the circle in the section labeled C P U, through a label feedback. The right side of the block labeled P I D calculation points at the right side, through an arrow passing across the right edge of the rectangle. The arrow passes through a terminal in the section labeled as analog output module and is labeled as output. The section labeled as analog input module is highlighted in blue. The section labeled as analog output module is highlighted in red.

Return to Figure 14-18

OceanofPDF.com

Figure 14-19 Extended Description (Chapter 14)

Return to Figure 14-19

From top to bottom, the P I D instruction is labeled as P I D, P I D, control block, process variable, control variable, control block length, and setup screen. A green colored rectangle is placed at the right side of the label control block. A blue colored rectangle is placed at the right side of the label process variable. A red colored rectangle is placed at the right side of the label control variable. A black colored rectangle is placed at the right side of the label control block length. The right side of the P I D instruction is connected to a vertical line.

Return to Figure 14-19

OceanofPDF.com

Figure 14-21 Extended Description (Chapter 14)

Return to Figure 14-21

Sets of three bottles are placed over a conveyor belt, which has a conveyor servo motor on the left side. The bottles move towards the left side. The P L C consists of six columns. From the left, the six columns are labeled as P L C, processor, communication, motion, conveyor servo drive, and filler servo drive. The bottle filler servo motor is linked to filler servo drive. The conveyor servo motor is linked to conveyor servo drive.

Return to Figure 14-21

OceanofPDF.com

Figure 14-22 Extended Description (Chapter 14)

Return to Figure 14-22

Another cylinder labeled as two and shoulder swivel is placed over the base. The arm is placed over the second cylinder. From left to right, the arm has labels as three elbow extension, four pitch, five yaw, and six roll. The rotation of the arm sweep and the yaw are in the same plane. The rotation of the shoulder swivel, elbow extension, and pitch are in the same plane. The rotation of the roll is another plane. All these planes are perpendicular to each other.

Return to Figure 14-22

OceanofPDF.com

Figure 14-23 Extended Description (Chapter 14)

Return to Figure 14-23

The right side and the left side of a block labeled point-to-point communications points at the bottom-left side of the jack and the top side of a P L C, respectively. The left side and the bottom side of a block labeled serial data communications points at the bottom side of the jack and the top side of an operator interface, respectively. The left side and the right side of a block labeled point-to-point communications points at the bottom-right side of the jack and the bottom side of a printer, respectively.

Return to Figure 14-23

OceanofPDF.com

Figure 14-25 Extended Description (Chapter 14)

Return to Figure 14-25

From top to bottom, coaxial, twister pair, and fiber optic are shown on the left side. For wireless system, a computer connected to radio, through a P L C is shown. A double-headed zig-zag arrows points at the connection of computer, radio, and P L C and a connection of a radio and a P L C. Another double-headed zig-zag arrows points at the connection of computer, radio, and P L C and another connection of a radio and a P L C.

Return to Figure 14-25

OceanofPDF.com

Figure 14-26 Extended Description (Chapter 14)

Return to Figure 14-26

From the left, a P L C and robot controller are connected across the double-headed arrow labeled device level (bottom) and a double-headed arrow labeled as control level (top). From the left, a P L C and computer system are connected across the double-headed arrow labeled control level (bottom) and a double-headed arrow labeled as information level (top). At the top of the double-headed arrow labeled as information level, a block divided into four regions and another computer system are connected.

Return to Figure 14-26

OceanofPDF.com

Figure 14-28 Extended Description (Chapter 14)

Return to Figure 14-28

The end points of the main trunk line are labeled as termination. The top side of the inverted T-shaped blocks is connected to com label in a processor. A label network interface module points at the top side of all the four processors. An illustration of the actual media attachment is given at the right side. The vertical line is labeled as drop line and the horizontal line is labeled as trunk line. An inverted T-shaped illustration is labeled as tee tap. It has two cables along the trunk line, one from the left side and another from the right side. It also has a cable along the drop line. The joining ends of the cable and tee tap are labeled as connectors.

Return to Figure 14-28

OceanofPDF.com

Figure 14-29 Extended Description (Chapter 14)

Return to Figure 14-29

The first gateway at top is connected to siemens. The second gateway at top is connected to Omron. The first gateway at the bottom is connected to Allen-Bradley, which is further connected to five blocks labeled as data highway. The second gateway at the bottom is connected to Gould, which is further connected to five blocks labeled as Modbus.

Return to Figure 14-29

OceanofPDF.com

Figure 14-30 Extended Description (Chapter 14)

Return to Figure 14-30

A white token is placed at the top-left side of each node. An orange toke is placed at the top-right side of the node one. An arrow from the bottom-right side of the trunk line points at the first white token, through a label three-to-one. An arrow from the orange token points at the second white token, through a label one-to-two. An arrow from the top-right side of the node two points at the third white token, through a label two-to-three.

Return to Figure 14-30

OceanofPDF.com

Figure 14-31 Extended Description (Chapter 14)

Return to Figure 14-31

An upwards pointing arrow labeled as read command and a downwards pointing arrow labeled as write command is shown below the server controller and along the connection.

Return to Figure 14-31

OceanofPDF.com

Figure 14-32 Extended Description (Chapter 14)

Return to Figure 14-32

The fourth device at the bottom is a computer system. The second device at the top shows a digital screen.

Return to Figure 14-32

OceanofPDF.com

Figure 14-33 Extended Description (Chapter 14)

Return to Figure 14-33

Eight arrows from transmitting side points towards the receiving side. From top to bottom, arrows are labeled as 1, 0, 1, 0, 1, 1, 0, and 1.

Return to Figure 14-33

OceanofPDF.com

Figure 14-34 Extended Description (Chapter 14)

Return to Figure 14-34

An arrow from transmitting side points towards the receiving side. A rectangular wave is placed at the top of the arrow. From left to right, the region between rising and falling edge is labeled as 1, 0, 1, 0, 1, 1, 0, and 1.

Return to Figure 14-34

OceanofPDF.com

Figure 14-35 Extended Description (Chapter 14)

Return to Figure 14-35

A label D H plus network is placed at the bottom of the computer system. From top to bottom, the C P U is divided into four sections. The first section is labeled as S L C 5/0 4 C P U. The second section shows run, a rectangle, another rectangle, force; F L T, a rectangle, another rectangle, D H (with positive sign at the bottom); B A T T, a rectangle, another rectangle, R S 2 3 2; and run, rem, prog. The third section shows a circle from which a rectangular region of height equal to the radius is removed. A line from the top-left side of the circle is connected to the label run in the above section. A line from the top side of the circle is connected to the label rem in the above section. A line from the top-right side of the circle is connected to the label prog in the above section. From top to bottom, the fourth section shows a distorted rectangle with three circles enclosed in it, two concentric circles enclosed in a square, and a jack shaped as a vertical trapezium enclosed in a vertical rectangle. The trapezium encloses four ports at the left and five ports at the right. A circle is at the top and another at the bottom of the trapezium. The distorted rectangle, two concentric circles enclosed in a square, and the jack are labeled as communication ports.

Return to Figure 14-35

OceanofPDF.com

Figure 14-36 Extended Description (Chapter 14)

Return to Figure 14-36

There is one trapezium-shaped jack in the section labeled as processor. It is connected to an operator terminal. There are two trapezium-shaped jacks in the section labeled as R S-2 3 2 module. One of then is connected to a barcode reader and another is connected to a vision system.

Return to Figure 14-36

OceanofPDF.com

Figure 14-37 Extended Description (Chapter 14)

Return to Figure 14-37

The I or O module connected to 5 sets of devices, through different connection points. This is labeled as conventional system. The device net scanner module is connected to a device, through a connection point. This device is connected in series with other devices. This is labeled as device net system.

Return to Figure 14-37

OceanofPDF.com

Figure 14-38 Extended Description (Chapter 14)

Return to Figure 14-38

The device net scanner has ports labeled as device net port. The device net I or O bus network is connected to a sensor at the bottom, through the connections labeled as power and data.

Return to Figure 14-38

OceanofPDF.com

Figure 14-39 Extended Description (Chapter 14)

Return to Figure 14-39

The bottom-right port of the device box tap is connected to left side of an open style tap, through thick cable. The right side of the open style tap is connected to the left side of the T-port tap with a terminator at the right side, through a thick cable. The second generic sealed device is connected at the top of the T-port tap, through a thin cable. The top side of the open style tap is connected to a generic open-style device. Their connection is enclosed in an inverted L-shape. The top-left port of the device box tap is connected to the bottom side of the PowerTap tap, through thick cable. The top-left port of the PowerTap tap is connected to D C power supply. The top-right port of the PowerTap tap is connected to the second T-port tap, with a terminator at the opposite side, through a thick cable. The third port of the T-port tap is connected to top-left port of a device port tap (eight-port), whose bottom-right port is connected to the third generic sealed device.

Return to Figure 14-39

OceanofPDF.com

Figure 14-40 Extended Description (Chapter 14)

Return to Figure 14-40

From the left, the first terminal and the second terminal of the control net scanner are labeled B (highlighted in orange) and A (highlighted in blue). The terminal A is connected to the first tee tap (blue), through red cable. The first tee tap is connected to the second inverted tee tap (blue) from the right side. The left side of the first tee tap shows two dots. The right side of the second tee tap is connected to the third tee tap (blue), whose right end is connected to two dots. The bottom side of the second tee tap is connected to the bottom terminal of the device, through blue cable. The top side of the third tee tap is connected to the bottom terminal of the computer system. The terminal B is connected to the top side of the fourth tee tap (orange), through red cable. The fourth tee tap is connected to the fifth inverted tee tap (orange) from the right side. The left side of the fourth tee tap shows two dots. The right side of the fifth tee tap is connected to the sixth tee tap (orange), whose right end is connected to two dots. The bottom side of the fifth tee tap is connected to the top terminal of the device, through blue cable. The top side of the sixth tee tap is connected to the top terminal of the computer system.

Return to Figure 14-40

OceanofPDF.com

Figure 14-41 Extended Description (Chapter 14)

Return to Figure 14-41

The top-left side of the corporate network is connected to back-office mainframes and servers (E R P, M E S, etcetera). The bottom-left side of the corporate network is connected to office applications, internetworking, data servers, and storage. From the left, the first block and the second block are connected to each other. The left side of the first block is connected to a camera. From the left, the bottom side of the first block is connected to controller, I OR O, H M I, and robotics. The right side of the second block is connected to a phone. From the left, the bottom side of the second block is connected to supervisory control, motors drives actuators, industrial network, safety controller, and the third block. From the left, the third block is further connected to safety I or O, three sensors and other input or output devices, and H M I.

Return to Figure 14-41

OceanofPDF.com

Figure 14-42 Extended Description (Chapter 14)

Return to Figure 14-42

The top side of the first field device is connected with two terminals, through connectors. The left-most terminal is connected to the second field device. The top side of the field device is connected with two terminals, through connectors.

Return to Figure 14-42

OceanofPDF.com

Figure 14-43 Extended Description (Chapter 14)

Return to Figure 14-43

The S E R C O S interface module has a receiver on the left side and a transmitter is at the right side of the receiver. Two fiber optic cables are shown below the receiver and the transmitter. The second illustration shows a ControlLogix system labeled as 1 7 5 6-M 0 8 S E is connected to Kinetix 6000 multi-axis servo drives, through S E R C O S. The multi axis Serco drives are connected to an A C induction motor, and three M P series brushless servo motors. From the left, a conveyor belt is connected to the A C induction motor, and three M P series brushless servo motors. The other side of the conveyor belt is connected with two sensors. There are six products on the conveyor belt, which are randomly spaced. The second illustration has a label A-B quality at the top-right side.

Return to Figure 14-43

OceanofPDF.com

Figure 14-44 Extended Description (Chapter 14)

Return to Figure 14-44

For the line graph, the values in milli-amperes are along the vertical axis and have two labels four milli-amperes and 20 milli-amperes. The horizontal axis is labeled time. A straight line labeled analog signal increases gradually. A sine wave labeled H A R T digital signal varies across the straight line. The sine wave shows two types of peak, one of them is wider than the other. For each wider peak, a label one is shown and for every two less wider peaks, a label zero is shown. From left to right, the labels below the sine wave are 1, 1, 0, 1, 0, 0, and 1. The illustration shows a H A R T-enabled I or O (left) is connected to a smart H A R T instrument (right). A sine wave is shown across the line connecting both the devices. An arrow pointing left over the connection is labeled as analog four to 20 milli-amperes. A double-headed arrow is placed below the connection and is labeled as H A R T digital data.

Return to Figure 14-44

OceanofPDF.com

Figure 14-45 Extended Description (Chapter 14)

Return to Figure 14-45

The computer further points at spreadsheet and a hardcopy. A person is sitting in front of the computer. The printer further points at tables and graphs. The left side of the interface system is labeled as signal sources. From top to bottom, the left side of the interface system is connected to tools labeled as three temperature thermocouples and resistance temperature detectors, pressure, position, force, strain, and speed. The right side of the interface system is labeled as output loads. From top to bottom, the right side of the interface system points at labels, alarms, P L C relay controls, analog outputs, and P I D process control.

Return to Figure 14-45

OceanofPDF.com

Figure 14-46 Extended Description (Chapter 14)

Return to Figure 14-46

An arrow labeled S C A D A or H M I software points at the right side of the computer. Four double-headed arrows are shown along the connection of the computer and the P L Cs. The three arrows are placed near the P L Cs, while the fourth arrow is placed near the host computer. The arrow near the host computer is labeled as data transfer. The right side of all the P L Cs is connected to separate field devices, through I or O control.

Return to Figure 14-46

OceanofPDF.com

Figure 14-47(a-b) Extended Description (Chapter 14)

Return to Figure 14-47(a-b)

In part a, five steps are shown, where the steps are connected through double-headed arrows. The first step shows a person sitting in front of a computer and the first step is labeled as R S Logix 5000 alarm instructions are programmed and then downloaded into the controller. The second step shows a controller at the top-left side of a ladder diagram and this step is labeled as controller detects alarm conditions and notifies device server of changes in alarm states. The third step shows a computer and this step is labeled as device server notifies FactoryTalk Alarms and events services of alarm state changes. The fourth step shows a block labeled as FactoryTalk alarms and events services and this step is labeled as FactoryTalk alarms and events services routes alarm information to logs and to alarm and event objects embedded in graphic displays in FactoryTalk view. The block is also connected to a C P U labeled as alarm and event history log, through a double headed arrow. The fifth step shows another person sitting in front of a computer and this step is labeled as using FactoryTalk View SE Clients, operators interact with alarms (for example, acknowledge, suppress) from graphic displays. In part b, four steps are shown, where the steps are connected through double-headed arrows. The first step shows a person sitting in front of a computer along with a P L C at the right side and the first step is labeled as logic in a controller is programmed to detect problems and latch fault conditions in controller tags or addresses. The second step shows a computer and this step is labeled as tag alarm and event server polls controller to monitor alarms and then notifies users of alarm conditions. The third step shows a block labeled as FactoryTalk alarms and events services and this step is labeled as FactoryTalk alarms and events services routes alarm information to logs and to alarm and event objects embedded in graphic displays in FactoryTalk view. The block is also connected to a C P U labeled as alarm and event history log, through a double headed arrow. The fourth step shows another person sitting in front of a computer and this step is labeled as using FactoryTalk View SE Clients, operators interact with alarms (for example, acknowledge, suppress) from graphic displays.

Return to Figure 14-47(a-b)

OceanofPDF.com

Figure 14-48 Extended Description (Chapter 14)

Return to Figure 14-48

An arrow labeled as DeviceNet scanner points at the right side of the P L C. A downwards pointing arrow is labeled as P L C outputs, which points from the P L C towards the V F D. An upwards pointing arrow is labeled as V F D outputs, which points from the V F D towards the P L C. For the structure of the V F D, a vertical bock labeled as V F D is shown. From top to bottom, the left side has various terminals, which are labeled as start, stop, F w d, R e v, other, comms, and speed. The first four terminals are connected to a rectangle labeled as outputs from the P L C are viewed as inputs to the V F D. The label other is shown beside four terminals. The label speed is shown beside three terminals connected to a circle labeled as potentiometer or analog signal. From top to bottom, the right side of the V F D is connected to three terminals that are further connected to a circle, a terminal labeled alarm, and four terminals that are further connected to a rectangle. The circle has a label motor three times phi. The rectangle has a label outputs from the V F D are viewed as inputs to the P L C.

Return to Figure 14-48

OceanofPDF.com

Figure 14-49 Extended Description (Chapter 14)

Return to Figure 14-49

The table is labeled as start or stop code example. From left to right, the first row is labeled as bit address, command, and description. From left to right, the second row is labeled as <Local:6:O.Data[0].0, stop drive, and 1 = stop, 0 = not stop. From left to right, the third row is labeled as <Local:6:O.Data[0].1>, stop drive, and 1 = start, 0 = not start. The ladder diagram consists of two horizontal lines connected between three vertical lines. A label inputs is placed over the first horizontal line connected between the first and the second vertical line. From top to bottom, N O pushbutton labeled as start_V F D_P B, and N C pushbutton labeled as stop_V F D_P B are connected along the horizontal lines that are connected between the first and the second vertical line. From left to right, X I C symbol labeled as start_V F D_P B, and an output energize labeled as logic command bits V F D_start <Local:6:O.Data[0].1> are connected in series along the first horizontal line connected between the second and third vertical line. From left to right, X I O symbol (highlighted in green) labeled as stop_V F D_P B, and an output energize labeled as logic command bits V F D stop <Local:6:O.Data[0].0> are connected in series along the second horizontal line connected between the second and third vertical line.

Return to Figure 14-49

OceanofPDF.com

Figure 14-50 Extended Description (Chapter 14)

Return to Figure 14-50

There are fourth vertical lines. From top to bottom, N O pushbutton labeled as start_V F D_P B, and N C symbol labeled as stop_V F D_P B are connected along two horizontal lines connected at the right side of the first vertical line. A label inputs is placed over the first horizontal line connected at the right side of the first vertical line. There are five horizontal lines connected between the second and the third vertical lines. An output energize labeled as Local:6:O.CommandRegister.Run is connected along the first horizontal line labeled as run DeviceNet scanner. From left to right, X I C symbol labeled as start_V F D_P B and an output energize labeled as Logic Command Bits V F D_start <Local:6:O.Data[0].1> are connected in series along the second horizontal line labeled as start variable frequency drive. From left to right, X I O symbol (highlighted in green) labeled as stop_V F D_P B and an output energize labeled as Logic Command Bits V F D_stop <Local:6:O.Data[0].0> are connected in series along the third horizontal line labeled as stop variable frequency drive. The M D V instruction is connected along the fourth horizontal line labeled as set speed reference to 30 hertz. From top to bottom, the M D V instruction is labeled as M D V, move, source 300, dest V F D_SPEED_REFERENCE, and <Local:6:O.Data[1]> (at the right side). From left to right, X I C symbol labeled as Logic Command Bits V F D_active <Local:6:O.Data[0].1> and an output energize labeled as V F D_Active P L are connected in series along the fifth horizontal line labeled as variable frequency drive action indication. The light labeled as V F D active P L is connected along a horizontal line that is connected at the left side of the fourth vertical line. A label output is placed over the horizontal line that is connected at the left side of the fourth vertical line.

Return to Figure 14-50

OceanofPDF.com

Figure 14-51 Extended Description (Chapter 14)

Return to Figure 14-51

Top: A P L C local rack depicted at the middle with processor module and output module slot 2 terminal 0 slots labeled. The processor module slot is connected to H M I graphic display (on the left) via ethernet communication cable. The H M I graphic display has a green start button on the top left, a darker green running light on the bottom right, and a red stop light on the bottom left of the running light. The motor on the right is connected to the output module slot 2 terminal 0 via motor power cable.

Bottom left: A table titled P L C tag structure consisting of 3 rows and 5 columns. The column headers are labeled: tag name, style, data type, alias for (in bold), base tag (in bold). Row entries are as follows: Row 1: decimal, B O O L, blank, and blank. Row 2: stop, decimal, B O O L, blank, and blank. Row 3: motor, decimal, B O O L, 2:O.Data.0 (in bold), and 2:O.Data.0 (in bold).

Top right: The illustration titled P L C program consists of three vertical lines. A stop and start push buttons along with a motor in series is connected across the first and the second vertical lines. A motor is connected in parallel across the start button. On the right, a motor with an output in series is connected to line L 2 on the right.

Return to Figure 14-51

OceanofPDF.com

Figure 15-2 Extended Description (Chapter 15)

Return to Figure 15-2

The window screen is divided into three horizontal sections. From the left, the first section is labeled as file, edit, view, search, comms, tools, window, and help. All the labels are faded. From the left, the second section shows a symbol of file, a symbol of folder, symbol to save the file, symbol to print, symbol of cut, symbol to copy, symbol to paste, symbol of undo, symbol of redo, a rectangle with a drop-down button on the left, three symbols faded, a symbol of file with a green check, another symbol with a green check, a symbol of zoom in, a symbol of zoom out, symbol of a table, positive sign, and an arrowhead pointing right. Except for the first two, all the symbols are faded. The third horizontal section is divided into two vertical sections. On the left section, labeled no controller, no forces, no edits, and redundancy are enclosed in a rectangle with a symbol at the right side. Except for the first label, all are faded. These labels are placed on the left side of the labels run, ok, bat, and I or O. On the right section, a toolbar with an arrow-head pointing left at the top-left side and an arrow-head pointing right at the top-right side. Five symbols and a label enclosed in rectangles are placed relatively on the left side between the arrow-heads. From left to right, the symbols are of X I C, X I O, O T E, O T E with a label L enclosed in it, and O T E with a label U enclosed in it. Seven labels are shown at the bottom. From left to right, the labels are arrow-head facing, arrow-head facing right, favorites, bit, timer or counter, input or output, and compare. A rectangle labeled path: A B_D F 1-1\1 (enclosed in a rectangle with a drop-down button on the right side) along with a symbol of chart on the right side.

Return to Figure 15-2

OceanofPDF.com

Figure 15-3 Extended Description (Chapter 15)

Return to Figure 15-3

The hardware disk is labeled as Logix designer and placed at the top-left side of the illustration. The block with the ladder diagram is labeled as ladder logic program at the top and has a rectangle labeled as R S Logix is placed at the bottom of the ladder diagram. From the left, parallel connection of two X I C symbols, X I O symbol (highlighted in green), and an output energize are connected in series along a rung. The second block is placed below the first block. The symbol of chart in the second block is labeled as R S who. The second block has a rectangle labeled as R S Linx. A double-headed arrow points at the labels R S Logix and R S Linx. A double-headed arrow points at the label R S Linx and the control Logix Controller placed at the right side. The table labeled R S Linx is placed at the bottom-left side. The table consists of three rows and three columns. From left to right, the first row is labeled as file, view, and communications. From left to right, the second row shows a symbol of a flow chart, a symbol of two plugs joined together, and a label R S who. The first two columns of the last row are merged and highlighted in grey. The last column of the last row is labeled as configure drivers.

Return to Figure 15-3

OceanofPDF.com

Figure 15-4 Extended Description (Chapter 15)

Return to Figure 15-4

Both the dialogue boxes has a label general above the top-left side of the boxes. From the top, the dialogue box on the left side is labeled as vendor: Allen-Bradley (enclosed in a rectangle); type: 1756-L55 Control Logix 5555 Controller (enclosed in a rectangle); revision: 10.24 (enclosed in a rectangle); name: controller 1 (enclosed in a rectangle); description: prime controller (enclosed in a rectangle); chassis type: 1756-A7 7-slot chassis (enclosed in a rectangle); and slot: 1 (enclosed in a rectangle with an arrow-head pointing upwards above the arrow-head pointing downwards). From the top, the dialogue box on the right side is labeled as type: 1756-L55 Control Logix 5555 Controller (enclosed in a rectangle); vendor: Allen-Bradley (enclosed in a rectangle); parent: local (enclosed in a rectangle), slot 0 (enclosed in a rectangle with an arrow-head pointing upwards above the arrow-head pointing downwards); name: Digital_Input_16pt (enclosed in a rectangle); description: optional (enclosed in a rectangle); comm form: Input Data (enclosed in a rectangle); and revision: 1 (enclosed in a rectangle with an arrow-head pointing upwards above the arrow-head pointing downwards), electronic keying, exact match (enclosed in a rectangle).

Return to Figure 15-4

OceanofPDF.com

Figure 15-5 Extended Description (Chapter 15)

Return to Figure 15-5

The first block has two rectangles labeled as I or O data and system-shared data, which are placed one over the another. The second block labeled task has a rectangle labeled as program. The rectangle shows four rectangles aligned in a right slanting line, such that they cover certain area of the rectangle behind them. The firs two rectangles are labeled as main routine and other routine. Another rectangle labeled as program tags (local data) is placed on the right side of the four rectangles.

Return to Figure 15-5

OceanofPDF.com

Figure 15-6 Extended Description (Chapter 15)

Return to Figure 15-6

From the top, the labels are, controller R S Logix, (intended to level 1) controller tags, (intended to level 1) controller fault handler, (intended to level 1) power-up handler, tasks, (intended to level 1) continuous, (intended to level 2) program_01, (intended to level 3) program tags, (intended to level 3) main routine, (intended to level 1) periodic, (intended to level 2) program_02, (intended to level 3) program tags, (intended to level 3) main routine, (intended to level 1) unscheduled programs, (intended to level 2) H M I, (intended to level 3) program tags, (intended to level 3) main, motion groups, trends, data types, (intended to level 1) user-defined, (intended to level 1) strings, (intended to level 1) predefined, (intended to level 1) module-defined, and I or O configuration. A negative sign enclosed in a rectangle is placed on the left side of the labels controller R S Logix, tasks, continuous, program_01, periodic, program_02, unscheduled programs, H M I, and data types. A positive sign enclosed in a rectangle is placed on the left side of the labels motion groups, user-defined, strings, and predefined.

Return to Figure 15-6

OceanofPDF.com

Figure 15-7 Extended Description (Chapter 15)

Return to Figure 15-7

The folder is labeled as continuous task. The rectangle labeled as continuous task scan has a rectangle labeled as task auto-restart consists of dashed edges, except for the right side. The right side of the rectangle has a downwards pointing arrow. At the right side of the rectangle, three rectangles are shown. From the top, the rectangles are labeled as program 1, program 2, and program 32. There are three dots between the second and the third rectangles. A violet horizontal block is placed on the right side of the block with folder labeled as continuous task. The violet block is discontinuous, such that a blue horizontal block is placed above that discontinuous part. The left side of the blue horizontal block is connected to a block with a folder labeled as periodic task. A rectangle labeled periodic task scan is placed on the top-right side. It is similar to the continuous task scan, except for the rectangle labeled task auto-restart, there are four downwards pointing arrows. From the left, the arrows are labeled as five ms, 10 ms, and 15 ms. The last arrow has three dots instead of a label.

Return to Figure 15-7

OceanofPDF.com

Figure 15-8 Extended Description (Chapter 15)

Return to Figure 15-8

From top to bottom, the labels are main task, main program, program_A, program_B, and unscheduled programs. A negative sign enclosed in a rectangle is placed on the left side of the label main task. A positive sign enclosed in a rectangle is placed on the left side of the labels main program, program_A, and program_B.

Return to Figure 15-8

OceanofPDF.com

Figure 15-10 Extended Description (Chapter 15)

Return to Figure 15-10

The ladder diagram consists of four N O relay contacts and two O T E instructions. The three blocks labeled tags are arranged one over the other and are labeled as tag one, tag 2, and tag 3, from top to bottom. The three rows of the column labeled controller memory are labeled as tag one data, tag two data, and tag data three, from top to bottom. The right side of the column is labeled as memory locations. Three double-headed arrows are shown between the program and the tags. Three double-headed arrows are shown between the tags and the controller memory.

Return to Figure 15-10

OceanofPDF.com

Figure 15-11 Extended Description (Chapter 15)

Return to Figure 15-11

The first block is labeled as controller scope tags sensor_1 temp_1. A double-headed arrow points at the bottom-left side of the first block and the top side of the program A. A double-headed arrow points at the bottom-right side of the first block and the top side of the program B. A double headed arrow enclosed in a circle with a left slanting line over it points at the program A and program B. Both the blocks shows show four rectangles aligned in a right slanting line, such that they cover certain area of the rectangle behind them. The firs two rectangles are labeled as main routine and other routine. Another rectangle labeled as program scoped tags, tag_1, tag_2, and tag_3 is placed on the right side of the four rectangles.

Return to Figure 15-11

OceanofPDF.com

Figure 15-12 Extended Description (Chapter 15)

Return to Figure 15-12

From the top, the labels are controller R S Logix, (intended to level 1) controller tags (highlighted), (intended to level 1) controller fault handler, (intended to level 1) power-up handler, tasks, (intended to level 1) continuous, (intended to level 2) program_01, (intended to level 3) program tags (highlighted), (intended to level 3) main routine, (intended to level 1) periodic, (intended to level 2) program_02, (intended to level 3) program tags (highlighted), and (intended to level 3) main routine. An arrow pointing left side points at the first highlighted label controller tags. An arrow pointing left side points at the second highlighted label program tags. An arrow pointing left side points at the third highlighted label program tags.

Return to Figure 15-12

OceanofPDF.com

Figure 15-13 Extended Description (Chapter 15)

Return to Figure 15-13

From the left, the five blocks are labeled as location, :slot, :type, .Member, .SubMember, and .Bit. A label format on the left side of the block labeled location. From the left, the labels local, :2, :O, and .Data are placed below the first four blocks. A label optional is placed above the last two blocks. A label .4 is placed below the last block. At the bottom-left side, a double-headed arrowpoints at a block labeled base tag and the column labeled controller memory. The block labeled base tag has a label Local:2:O.Data.4 enclosed in it. The first row of the column has a label tag data. The screen has a label data type on the right side of the label tag name. A label Local:2:O.Data is placed below the label tag name. A label B O O L is placed below the label data type. A table consisting of four rows and eight columns is placed below the labels on the screen. From the left, the first row is labeled from zero through seven with number 4 in bold. From the left, the second row is labeled from eight through fifteen. From the left, the third row is labeled from 16 through 23. From the left, the fourth row is labeled from 24 through 31. A label controller scoped tags is placed at the bottom of the screen.

Return to Figure 15-13

OceanofPDF.com

Figure 15-14 Extended Description (Chapter 15)

Return to Figure 15-14

The alias tag labeled as fan_motor is connected to the base tag labeled as Local:2:O.Data.5. The base tag and the tag data labeled as controller memory are connected, through double-headed arrow. A block labeled Fan_Motor <Local:2:O.Data.5> has a symbol of O T E instruction and the block is placed below the alias and the base tag.

Return to Figure 15-14

OceanofPDF.com

Figure 15-15 Extended Description (Chapter 15)

Return to Figure 15-15

The controller labeled produced tag is connected to two controllers labeled as consumed tag. The controller labeled as producer or consumer I or O model has eight vertical sections, such that the first section is wider than the others. From the left, the second, third, and fourth sections are labeled as input modules, which have arrows pointing downwards at a left pointing arrow labeled commonly shared data. The fifth and the sixth section are labeled as Logix 5500 controllers, which have double-headed arrows. The last two sections are labeled as output modules and has arrows pointing downwards. An arrow points from the fifth towards eight section. An arrow points from the sixth towards seventh section.

Return to Figure 15-15

OceanofPDF.com

Figure 15-16 Extended Description (Chapter 15)

Return to Figure 15-16

All the blocks have a label 31 at the left end of the first row and a label zero at the right end of the first row. The second row of all the blocks are divided into 32 column. For the block labeled BOOL, the first 31 columns are merged and labeled as unused. For the block labeled SINT, the first 24 columns are merged and labeled as unused and a label 7 is shown above the next column. For the block labeled INT, the first 16 columns are merged and labeled as unused and a label 15 is shown above the next column.

Return to Figure 15-16

OceanofPDF.com

Figure 15-17 Extended Description (Chapter 15)

Return to Figure 15-17

The labels in the first column, except for the first row are labeled as members. From the top, the second column is labeled as DINT, DINT, BOOL, BOOL, BOOL, BOOL, BOOL, BOOL, and BOOL. The labels in the second column, except for the first row are labeled as data types. The first row of the third and the fourth columns are labeled as style and description. The second, third, fourth, seventh, eight, and the ninth rows of the third column have label decimal.

Return to Figure 15-17

OceanofPDF.com

Figure 15-18 Extended Description (Chapter 15)

Return to Figure 15-18

From the top, the labels are name, counter (enclosed in a rectangle); description, an empty rectangle with an upwards pointing arrowhead placed above a downwards pointing arrow on the right side); and members, data type size: 12 byte (s). From the top, the first column of the table is labeled as name, P R E, A C C, C U, C D, D N, O V, and U N. From the top, the second column is labeled as data type, DINT, DINT, BOOL, BOOL, BOOL, BOOL, and BOOL. From the top, the first row of the third column is labeled as style and the remaining rows are labeled as decimal. The first row of the fourth column is labeled as description.

Return to Figure 15-18

OceanofPDF.com

Figure 15-19 Extended Description (Chapter 15)

Return to Figure 15-19

The label controller three is enclosed in a rectangle with a drop-down button on the right side. The first table consists of three rows and five columns. From left to right, the first row is labeled as name, value (an arrow pointing left), force mask (an pointing left), style, and data type. From the top, the first column is labeled as name, (positive sign) Local:1:C, and (positive sign) Local:1:I. A label configuration data points at the label Local:1:C. A label input data points at the label Local:1:I. From the top, the fourth column is labeled as data type, AB:1756_DI_AC_..., and AB:1756_DI_AC_. At the bottom of the first table, an arrowhead pointing left, an arrowhead pointing right, a label monitor tags (highlighted), and edit tags are shown. From the left, the first row of the second table are labeled as name, value (an arrow pointing left), force mask (an pointing left), style, and data type. The second table consists of 16rows and five columns.

Return to Figure 15-19

OceanofPDF.com

Figure 15-20 Extended Description (Chapter 15)

Return to Figure 15-20

From the top, the labels are name, tank (enclosed in a rectangle), size: 16 (enclosed in a rectangle), byte (s); and description: generic storage tank data type (enclosed in a rectangle with an upwards pointing arrowhead placed above a downwards pointing arrow on the right side). The last row is unlabeled. From the top, the first column of the table is labeled as name, level, pressure, temp, and agitator_speed. From the top, the second column is labeled as data type, INT, DINT, REAL, and DINT. From the top, the third column is labeled as style, decimal, decimal, float, and decimal. From the top, the fourth column is labeled as description, stores the level in inches, stores the pressure in P S I G, the temperature in F, and speed in R P M.

Return to Figure 15-20

OceanofPDF.com

Figure 15-21 Extended Description (Chapter 15)

Return to Figure 15-21

The two labels are controller controller 1 and controller tags (highlighted). A box is placed over the highlighted label. From the top, the box are labeled as new tag Ctrl plus W, monitor tags, edit tags, verify, export tags, and print Ctrl plus P. From the top, the dialogue box labeled as new tag is labeled as name: all_stop (enclosed in a rectangle); description: this is the line stop pushbutton input (enclosed in a rectangle); tag type: base (selected); (indented to level 1) alias; (indented to level 1) produced, one (enclosed in a rectangle), consumers; (indented to level 1) consumed; data type: BOOL, configure (faded); scope: controller 1 (controller) (enclosed in a rectangle with a drop-down button); and style: binary (enclosed in a rectangle with a drop-down button). From the top, the labels ok, cancel, and help are placed on the right side of the labels name and description.

Return to Figure 15-21

OceanofPDF.com

Figure 15-22 Extended Description (Chapter 15)

Return to Figure 15-22

From the left, the labels at the top are scope: controller 1 (controller) (enclosed in a rectangle with a drop-down button on the right side), show: show all (enclosed in a rectangle with a drop-down button), and sort: base tag (enclosed in a rectangle). An arrowhead pointing right is placed on the second rectangle at the left side of the table. From left to right, the first row is labeled as tag name, value, force mask, and style. From the top, the first column is labeled as tag name, all_stop, (a positive sign) Local:2:C, (a positive sign) Local:2:I, and section_3_run. From the top, the second column is labeled as value, 0, {...}, {...}, and 2#0000_0000. The third and fourth rows of the third column are labeled as {...}. The second and the fifth row of the fourth column are labeled as decimal and binary, respectively. From the left, an arrowhead pointing left, an arrowhead pointing right, a label monitor tags (highlighted in blue), and a label edit tags are shown at the bottom. A vertical scrollbar and a horizontal bar are shown at the bottom and the right side of the screen.

Return to Figure 15-22

OceanofPDF.com

Figure 15-23 Extended Description (Chapter 15)

Return to Figure 15-23

From the left, the labels at the top are scope: controller 1 (controller) (enclosed in a rectangle with a drop-down button on the right side), show: show all (enclosed in a rectangle with a drop-down button), and sort: base tag (enclosed in a rectangle). An arrowhead pointing right is placed on the second rectangle at the left side of the table. An asterisk is placed on the fifth rectangle at the left side of the table. From left to right, the first row is labeled as tag name, value, force mask, and style. From the top, the first column is labeled as tag name, all_stop, (a positive sign) Local:2:C, (a positive sign) Local:2:I, and section_3_run. The fifth row of the second column and the third column are labeled as Local:2:I.Data.0. From the top, the fourth column is labeled as type, BOOL, AB:1756_DI:C:0, AB:1756_DI:C:0, and BOOL. From the left, an arrowhead pointing left, an arrowhead pointing right, a label monitor tags, and a label edit tags (highlighted in blue) are shown at the bottom. A vertical scrollbar and a horizontal bar are shown at the bottom and the right side of the screen.

Return to Figure 15-23

OceanofPDF.com

Figure 15-24 Extended Description (Chapter 15)

Return to Figure 15-24

The one-dimensional array with six vertically arranged elements is labeled as table six (in square brackets). The two-dimensional array with three columns of six vertically arranged elements is labeled as grid six, three (both the numbers are enclosed in square brackets). The three-dimensional array with two sets of two-dimensional array placed together is labeled as block six, three, two (all the numbers are enclosed in square brackets). The screen labeled as select data type has a block labeled as selection. The block shows a label DINT [6, 3, 2], and find match (enclosed in a rectangle). From the top, the labels ok, cancel, and help are placed on the right side of the block labeled selection. Another block labeled data types is placed below the block labeled selection. From the top, the second block is labeled as BOOL, CONTROL, COUNTER, DINT (highlighted in blue), INT, Load, Load_info, and MESSAGE. A vertical scroll bar is place on the left side of the second block. The third block labeled as array dimensions is placed at the bottom. From the left, the first row of the third block is labeled as Dim zero, Dim one, and Dim two. The numbers are underlined. From the left, the second row of the third block is labeled as six, three, and two. Each number in the second row is enclosed in a rectangle with a vertical scrollbar on the right side.

Return to Figure 15-24

OceanofPDF.com

Figure 15-26 Extended Description (Chapter 15)

Return to Figure 15-26

The table consists of three rows and 18 columns. The P L C processor is connected to the input tag data base, through an upwards pointing arrow and a downwards pointing arrow. The P L C processor is connected to the output tag data base, through a downwards pointing arrow. The bottom side of the output tag data base is connected to the left side of a P L C output module. The bottom side of the output module is connected to the top side of the field power supply. The right side of the output module is connected to the right side of the field power supply, through solenoid and three nodes. The first node is connected to the output module, through motor starter coil. The second node is connected to the output module, through second solenoid. The third node is connected to the output module, through a light. A label field output devices is placed over the first solenoid. P L C input module on the left side is connected to the P L C processor, through four arrows pointing right. The bottom side of the input module is connected to the top side of a field power supply. The left side of the input module is connected to the left side of a field power supply, through an N O pushbutton and three nodes. The first node is connected to the input module, through an N O switch. The second node is connected to the input module, through an N O limit switch. The third node is connected to the input module, through an N C pushbutton. A label field input devices is placed over the N O pushbutton. The left side of a block labeled ladder logic program is connected to the right side of the P L C processor, through an arrow pointing right and an arrow pointing left. From left to right, an X I C symbol, an X I O symbol, and an O T E instruction are connected in series along the first rung. From left to right, parallel connection of two X I C symbols, and the second O T E instruction are connected in series along the second rung. An arrow pointing right is shown over both the rungs. A downwards pointing arrow beside the vertical line on the right side.

Return to Figure 15-26

OceanofPDF.com

Figure 15-27 Extended Description (Chapter 15)

Return to Figure 15-27

From the left, an X I C symbol, an X I O symbol, and an output energize are connected in series along a rung. A label inputs is present between the X I C and X I O symbols. A label output is placed over the O T E instruction. For the first table, the first row shows a label if the data bit is: and X I C instruction is:, from the left. A symbol of X I C instruction is placed in the first row of the second column. From the left, the second row is labeled as logic zero and false. From the left, the third row is labeled as logic one and true. For the second table, the first row shows a label if the data bit is: and X I O instruction is:, from the left. A symbol of X I O instruction is placed in the first row of the second column. From the left, the second row is labeled as logic zero and true. From the left, the third row is labeled as logic one and false. For the third table, the first row shows a label if the data bit is: and O T E instruction is:, from the left. A symbol of O T E instruction is placed in the first row of the second column. From the left, the second row is labeled as logic zero and false. From the left, the third row is labeled as logic one and true.

Return to Figure 15-27

OceanofPDF.com

Figure 15-28 Extended Description (Chapter 15)

Return to Figure 15-28

From the left, X I C symbol labeled as tag A, X I O symbol labeled as tag B, and output energize labeled as tag 1, are connected in series along the rung of the first ladder diagram. An output energize labeled as tag 2 is connected across the output energize labeled as tag 1. A label parallel outputs is placed above the output energize labeled as tag 1. From the left, X I C symbol labeled as tag A, X I O symbol labeled as tag B, output energize labeled as tag 1, and output energize labeled as tag 2, are connected in series along the rung of the second ladder diagram. A label series outputs is placed above the output energize labeled as tag 1 and tag 2.

Return to Figure 15-28

OceanofPDF.com

Figure 15-29 Extended Description (Chapter 15)

Return to Figure 15-29

From left to right, X I O symbol (highlighted in green) labeled as tag A, X I C symbol (highlighted in green) labeled as tag B, output energize (highlighted in green) labeled as tag 1, X I O symbol labeled as tag C, and output energize labeled as tag 2 are connected in series along the rung.

Return to Figure 15-29

OceanofPDF.com

Figure 15-30 Extended Description (Chapter 15)

Return to Figure 15-30

Top: On the left, push buttons, P B 1, P B 2, P B 3, and P B 4 are connected to an input 120 volts of A C source. The middle portion consists of four rungs. The first rung shows a push button P B 1 connected to a pilot light P L 1 in series. The second rung shows a push button P B 2, a closed push button P B 3, and a pilot light P L 2 connected in series. The third rung shows a closed push button P B 4 connected to a pilot light P L 2 in series. On the right, pilot lights, P L 1, P L 2, and P L 3 are connected to an output line labeled N.

Bottom: A truth table consists of eight columns and six rows. The first two cells in column 1 are merged and labeled input device state. From column 2 through column 5, the top cells are merged and labeled input state. From column 6 through column 8, the top cells are merged and labeled output state. From row 2 through row 5, the row headers are No P Bs pressed, P B 1 pressed, P B 3 pressed, and P B 2, 3, 4 pressed, respectively. The row wise entries are as follows: Row 2: P B 4, P B 3, P B 2, P B 1, P L 3, P L 2, and P L 1. Row 3: 0, 0, 1, 0, 1, 1, and 0. Row 4: 0, 0, 1, 1,1, 1, and 1. Row 5: 0, 1, 1, 0, 1, 0, and 0. Row 6: 1, 1, 0, 0, 0, 0, and 0.

Return to Figure 15-30

OceanofPDF.com

Figure 15-31 Extended Description (Chapter 15)

Return to Figure 15-31

From the top, the first table is labeled as tag edit. The first table consists of three rows and five columns. From the left, the first row is labeled as tag name, alias for, base tag, type, and style. From the left, the second row is labeled as Limit_Switch, Local:1:I.data.2(C) (highlighted in blue), Local:1:I.data.2(C), BOOL, and decimal. From the left, the third row is labeled as High_limit_light, Local:2:O.data.4(C) (highlighted in blue), Local:2:O.data.4(C0, BOOL, and decimal. The P L C is labeled as seven-slot ControlLogix chassis has eight columns, where the first column is wider than the other columns. From the left, the columns are labeled as power supply, zero, one, two, three, four, five, and six. An input module labeled as eight-point discrete (digital) input module is at the top-left side of the P L C. From the top, the eight terminals of the input module are labeled from zero through seven. An output module labeled as eight-point discrete (digital) onput module is at the top-right side of the P L C. From the top, the eight terminals of the output module are labeled from zero through seven. The third terminal of the input module points at the column labeled one in the P L C, through an N C limit switch labeled as Limit_switch and switch-closed. The column labeled two in the P L C points at the fifth terminal of the output module, through a bulb labeled as High_limit_light and light-off. A label processor module points at the column labeled zero in the P L C. A double-headed blue arrow points at the highlighted label Local:1:I.data.2(C) in the first table and the limit switch. A double-headed red arrow points at the highlighted label Local:2:O.data.4(C) in the first table and the light bulb. The bottom side of the P L C points downwards at a block labeled program, which further points downwards at a block with a ladder diagram. From the left, an X I O symbol labeled as Limit_switch at the top and <Local:1:I.data.2(C)> at the bottom, and an output energize labeled as High_limit_light at the top and <Local:2:O.data.4(C)> at the bottom are connected in series along a rung. The second table at the bottom of the ladder diagram consists of three rows and four columns. From the left, the first row is labeled as tag name, value, style, and type. From the left, the second row is labeled as Limit_switch, one (highlighted in blue), decimal, and BOOL. From the left, the third row is labeled as High_limit_light, zero (highlighted in blue), decimal, and BOOL. A double-headed blue arrow points at the X I O symbol and label one in the second table. A double-headed red arrow points at the O T E and the label zero in the second table.

Return to Figure 15-31

OceanofPDF.com

Figure 15-32 Extended Description (Chapter 15)

Return to Figure 15-32

From left to right, N O pushbutton labeled motor starter, N C pushbutton labeled motor stop, and the motor starter coil labeled motor run are connected in series along the rung. The N O relay contact labeled as contactor auxiliary contact is connected across the N O pushbutton. The illustration of the N O pushbutton, N C pushbutton, and the motor starter coil is placed above their symbols.

Return to Figure 15-32

OceanofPDF.com

Figure 15-33 Extended Description (Chapter 15)

Return to Figure 15-33

The toolbar shows an arrowhead pointing left at the top-left side and an arrowhead pointing right at the top-right side. Five symbols and two labels enclosed in rectangles are placed relatively on the left side between the arrowheads. From left to right, the first three symbols are, an upper-case letter I rotated at ninety degrees labeled rung, same symbol as the first but includes a rectangle at the bottom of the middle line of the letter I labeled branch, and same symbol as the second but includes a horizontal line dividing the rectangle into half. From left, the last five symbols are of X I C, X I O, O T E, O T L, and O T U. From left to right, the labels are O N S and O S R. Two symbols and a label are shown at the bottom. From left to right, the labels are arrow-head facing, arrow-head facing right, and bit (highlighted in blue). At the bottom, the ladder diagram consists of three vertical lines. There are two rungs connected at the right side of the first vertical lines. From the left, N C pushbutton labeled stop is connected to a terminals four along the first rung. From the left, N O pushbutton labeled start is connected to a terminals three along the second rung. The bottom end of the first vertical line is connected to a label common, through a block labeled field device power. A positive sign is placed above the block. From left to right, parallel connection of two X I C symbols, another X I C symbol, and an output energize are connected in series along a rung between the second and the third vertical line. A label ladder logic program is placed over the rung between the second and the third vertical line. All the symbols used here are labeled as question mark. A rung is connected at the left side of the fourth vertical line. From the left, the terminal four is connected to a motor starter coil labeled as contactor. The bottom end of the fourth vertical line is connected to a label common, through a block labeled field device power. A positive sign is placed above the block. An illustration of input module labeled as D C input and S T 0 1 2 3 4 5 6 7 is placed below the first block labeled field device power. Numbers three and four are highlighted. An illustration of output module labeled as D C output and S T 0 1 2 3 4 5 6 7 is placed below the second block labeled field device power. Number four is highlighted.

Return to Figure 15-33

OceanofPDF.com

Figure 15-34 Extended Description (Chapter 15)

Return to Figure 15-34

From the top, the dialogue box labeled as new tag is labeled as name: motor_start (enclosed in a rectangle); description: start button for motor (enclosed in a rectangle); (indented to level 1) alias (enclosed in a rectangle with a drop-down button on the right side), (an empty rectangle); and (indented to level 1) Local:1:I.Data (enclosed in a rectangle with a drop-down button on the right side). A table of three rows and two columns is placed below these labels. From top, the first column is labeled as name, Local:1:C, Loacl:1:I, and Local:1:I.Fault. From top, the second column is labeled as data type, AB:1756_DO:C.0, AB:1756_DO:I.0,, and DINT. A label Local:1:I.Data (enclosed in a rectangle with a drop-down button) is placed at the bottom of the table. Another table is placed below the first table. The second table consists of four rows and eight column. From the left, the first row is labeled from zero through seven, where three is highlighted in blue and four is highlighted in grey. From the left, the second row is labeled from eight through 15. From the left, the third row is labeled from 16 through 23. From the left, the fourth row is labeled from 24 through 31.

Return to Figure 15-34

OceanofPDF.com

Figure 15-35 Extended Description (Chapter 15)

Return to Figure 15-35

From the left, 2 X I C symbols and an output energize are connected in series along the rung. From the top, the first X I C symbol is labeled as start button for motor (indicated as description), motor_start (indicated as tag name), and <Local:1:I.Data.3> (indicated as alias). From the top, the second X I C symbol is labeled as stop button for motor, motor_stop, and <Local:1:I.Data.4>. From the top, output energize is labeled as motor contactor coil, motor_run, and <Local:2:O.Data.4>. The third X I C symbol is connected across the first X I C symbol. From the top, the third X I C symbol is labeled as motor contactor coil, motor_run, and <Local:2:O.Data.4>.

Return to Figure 15-35

OceanofPDF.com

Figure 15-36 Extended Description (Chapter 15)

Return to Figure 15-36

The ladder diagram consists of 4 vertical lines. From the left, the first vertical line is labeled as L 1 and the fourth vertical line is labeled as L 2. Two rungs are connected on the right side of the line L 1. From the top, N C pushbutton (highlighted in red) labeled as motor_stop and N O pushbutton (highlighted in green) labeled as motor_start are connected along the rungs connected to the line L 1. An illustration of N O and N C pushbutton is shown below the second rung. A label inputs is placed over the first rung. A rung is connected between the second and the third vertical lines. From the left, an X I C symbol, the second X I C symbol (highlighted in green) and an output energize (highlighted in green) are connected in series along the rung. From the top, the first X I C symbol is labeled as start button for motor, motor_start, and <Local:1:I.Data.3>. From the top, the second X I C symbol is labeled as stop button for motor, motor_stop, and <Local:1:I.Data.4>. From the top, output energize is labeled as motor contactor coil, motor_run, and <Local:2:O.Data.4>. The third X I C symbol (highlighted in green) is connected across the first X I C symbol. From the top, the third X I C symbol is labeled as motor contactor coil, motor_run, and <Local:2:O.Data.4>. A label ladder logic program is placed over the first rung. A motor starter coil labeled as motor_run is connected along a rung connected at the left side of the line L 2. An illustration of the motor starter coil with a label contactor is shown below its symbol. A label output is placed over the rung connected to the line L 2. The table labeled as monitor tags window consists of four rows and five columns and is placed at the bottom. From the left, the first row is labeled as tag name, value, style, data type, and description. From the left, the second row is labeled as motor_start, zero, decimal, BOOL, and start button for motor. From the left, the third row is labeled as motor_stop, one, decimal, BOOL, and stop button for motor. From the left, the fourth row is labeled as motor_run, one, decimal, BOOL, and motor contactor coil.

Return to Figure 15-36

OceanofPDF.com

Figure 15-37 Extended Description (Chapter 15)

Return to Figure 15-37

At the top, a conveyor belt is connected to a three-phase reversing motor on the left side. A vertical rectangle placed over a horizontal rectangle and this setup is labeled as workpiece. The workpiece is placed on the left side over the conveyor bet. An arrow labeled forward points at the right side from the right side of the work piece. A dotted map of the workpiece is shown on the right side and an arrow labeled reverse points away from it towards the left side. Below the illustration, the ladder diagram is shown. The primary of the transformer has two parts. Terminals at the left and the right ends of the left primary transformer are labeled as H 1 and H 2. Terminals at the left and the right ends of the right primary transformer are labeled as H 3 and H 4. The terminal H 1 is connected to a label A on the left side and the terminal H 3. The terminal H 4 is connected to a label B on the left side and the terminal H 2. The secondary of the transformer has terminals X 1 at the left and X 2 at the right. A label 120 V is placed below the secondary of the transformer. The terminal X1 is connected to N C pushbutton labeled stop, through a fuse. The terminal X 2 is connected to an N C relay contact labeled as O L. The N C pushbutton and the N C relay contact labeled as O L are connected along a rung, through N O pushbutton labeled forward, N C relay contact labeled as R, and a relay coil F. An N O relay contact labeled as F is connected across the N O pushbutton labeled as forward. A series connection of an N O pushbutton labeled as reverse, N C relay contact labeled as F, and a relay coil R is connected across the series connection of the N O pushbutton labeled forward, N C relay contact labeled as R, and the relay coil F. An N O relay contact labeled as R is connected across the N O pushbutton labeled as R. The relay coils F and R are connected to the left end of a small horizontal red line on the right side, through dashed red lines. An illustration of the three pushbutton on the left side of the ladder diagram shows a red button labeled as stop, a blue button labeled as forwards, and a yellow button labeled as reverse.

Return to Figure 15-37

OceanofPDF.com

Figure 15-38 Extended Description (Chapter 15)

Return to Figure 15-38

From the left, the first vertical line is labeled as L 1 and the fourth vertical line is labeled as L 2. Three rungs are connected on the right side of the line L 1. From the top, N C pushbutton (highlighted in red) labeled as stop_button, N O pushbutton (highlighted in blue) labeled as forward_button, and N O pushbutton (highlighted in yellow) labeled as reverse_button are connected along the rungs connected to the line L 1. An illustration of pushbuttons is shown below the third rung. A label inputs is placed over the first rung. Two rungs are connected between the second and the third vertical lines. From the left, an X I C symbol (highlighted in green), the second X I C symbol, X I O symbol (highlighted in green), and an output energize are connected in series along the first rung. From the top, the first X I C symbol is labeled as start_button, and <Local:1:I.Data.0>. From the top, the second X I C symbol is labeled as forward_button, and <Local:1:I.Data.1>. From the top, the first X I O symbol is labeled as reverse_coil, and <Local:2:O.Data.1>. From the top, output energize is labeled as forward_coil, and <Local:2:O.Data.0>. The third X I C symbol is connected across the second X I C symbol. From the top, the third X I C symbol is labeled as forward_coil, and <Local:2:O.Data.0>. From the left, the fourth X I C symbol (highlighted in green), the fifth X I C symbol, the second X I O symbol (highlighted in green), and the second output energize are connected in series along the second rung. From the top, the fourth X I C symbol is labeled as stop_button, and <Local:1:I.Data.0>. From the top, the fifth X I C symbol is labeled as reverse_button, and <Local:1:I.Data.2>. From the top, the second X I O symbol is labeled as forward_coil, and <Local:2:O.Data.0>. From the top, the second output energize is labeled as reverse_coil, and <Local:2:O.Data.1>. The sixth X I C symbol is connected across the fifth X I C symbol. From the top, the sixth X I C symbol is labeled as reverse_coil, and <Local:2:O.Data.1>. A label ladder logic program is placed over the first rung. Two rungs are connected at the left side of the line L 2. From the top, motor starter coil F labeled as forward_run and motor starter coil R labeled as reverse_coil are connected along the rungs connected at the left side of the line L 2. An illustration of a motor starter coil with a label reverse and a motor starter coil with a label contactors are shown below their symbols. A label outputs is placed over the first rung connected to the line L 2.

Return to Figure 15-38

OceanofPDF.com

Figure 15-39 Extended Description (Chapter 15)

Return to Figure 15-39

The ladder diagram consists of 4 vertical lines. From the left, the first vertical line is labeled as L 1 and the fourth vertical line is labeled as L 2. Three rungs are connected on the right side of the line L 1. From the top, N O switch labeled as position_1_switch, N O switch labeled as position_2_switch, and N O switch labeled as position_3_switch are connected along the rungs connected to the line L 1. An illustration of a switch is shown below the third rung. A label inputs is placed over the first rung. Two rungs are connected between the second and the third vertical lines. From the left, an X I C symbol, X I O symbol (highlighted in green), and an output energize are connected in series along the first rung. From the top, the first X I C symbol is labeled as position_1_switch, and <Local:1:I.Data.1>. From the top, the first X I O symbol is labeled as position_2_switch, and <Local:1:I.Data.2>. From the top, output energize is labeled as internal_relay. The series combination of the second X I C symbol and the second X I O symbol (highlighted in green) is connected across the series combination of the first X I C symbol and the first X I O symbol. From the top, the second X I C symbol is labeled as position_2_switch, and <Local:1:I.Data.2>. From the top, the second X I O symbol is labeled as position_1_switch, and <Local:1:I.Data.1>. From the left, the third X I C symbol, the third X I O symbol (highlighted in green), and the second output energize are connected in series along the second rung. From the top, the third X I C symbol is labeled as internal_relay. From the top, the third X I O symbol is labeled as position_3_switch, and <Local:1:I.Data.3>. From the top, the second output energize is labeled as room_light, and <Local:2:O.Data.5>. The series combination of fourth X I C symbol and the fourth X I O symbol (highlighted in green) is connected across the series combination of the third X I C symbol and the third X I O symbol. From the top, the fourth X I C symbol is labeled as position_3_switch, and <Local:2:O.Data.3>. From the top, the fourth X I O symbol is labeled as internal_relay. A label ladder logic program is placed over the first rung. A rung is connected at the left side of the line L 2. A light labeled as room_light is connected along the rung connected at the left side of the line L 2. An illustration of a bulb is shown below the light symbol. A label output is placed over the rung connected to the line L 2. The table labeled as monitor tags window consists of six rows and five columns and is placed at the bottom. From the top, the first column is labeled as name, internal_relay, position_1_switch, position_2_switch, position_3_switch, and room_light. The first row of the second column is labeled as value (an arrow pointing at the label from the right side) and the remaining rows are labeled as zero. The first row of the third column is labeled as force mask (an arrow pointing at the label from the right side) and the remaining rows are unlabeled. The first row of the fourth column is labeled as style and the remaining rows are labeled as decimal. The first row of the fifth column is labeled as data type and the remaining rows are labeled as BOOL.

Return to Figure 15-39

OceanofPDF.com

Figure 15-40 Extended Description (Chapter 15)

Return to Figure 15-40

From the left, an N C pushbutton labeled as stop, N O pushbutton labeled as start, and a relay coil labeled as control relay are connected in series along the first rung. An N O relay contact labeled as C R-1 is connected across the N O pushbutton. From the left, an N C pushbutton labeled as C R-2 and a green light labeled as standby P L are connected in series along the second rung. From the left, an N O pushbutton labeled as C R-3 and a green light labeled as run P L are connected in series along the third rung. From the left, an N O pushbutton labeled as C R-4 and a motor starter coil labeled as motor are connected in series along the fourth rung.

Return to Figure 15-40

OceanofPDF.com

Figure 15-41 Extended Description (Chapter 15)

Return to Figure 15-41

From the left, the first vertical line is labeled as L 1 and the fourth vertical line is labeled as L 2. Two rungs are connected on the right side of the line L 1. From the top, N C pushbutton (highlighted in red) labeled as motor_stop, and N O pushbutton (highlighted in green) labeled as motor_start are connected along the rungs connected to the line L 1. An illustration of pushbuttons is shown below the second rung. A label inputs is placed over the first rung. Four rungs are connected between the second and the third vertical lines. From the left, an X I C symbol (highlighted in green), the second X I C symbol, and an output energize are connected in series along the first rung. From the top, the first X I C symbol is labeled as motor_stop, and <Local:1:I.Data.0>. From the top, the second X I C symbol is labeled as motor_start, and <Local:1:I.Data.1>. From the top, output energize is labeled as internal_relay_C R. The third X I C symbol is connected across the second X I C symbol. From the top, the third X I C symbol is labeled as internal_relay_C R. From the left, the first X I O symbol (highlighted in green) labeled as internal_relay_C R, and the second output energize labeled as standby_P L and <Local:2:O.Data.1> are connected in series along the second rung. From the left, the fourth X I C symbol labeled as internal_relay_C R, and the third output energize labeled as run_P L and <Local:2:O.Data.1> are connected in series along the third rung. From the left, the fifth X I C symbol labeled as internal_relay_C R, and the fourth output energize labeled as motor and <Local:2:O.Data.2> are connected in series along the fourth rung. A label ladder logic program is placed over the first rung. Three rungs are connected at the left side of the line L 2. From the top, green light labeled as standby_P L, red light labeled as run_P L, and a motor are connected along the rungs connected at the left side of the line L 2. A label outputs is placed over the first rung connected to the line L 2.

Return to Figure 15-41

OceanofPDF.com

Figure 15-42 Extended Description (Chapter 15)

Return to Figure 15-42

The ladder diagram consists of 4 vertical lines. From the left, the first vertical line is labeled as L 1 and the fourth vertical line is labeled as L 2. Two rungs are connected on the right side of the line L 1. From the top, N C pushbutton (highlighted in green) labeled as fan_on_button and N O pushbutton (highlighted in red) labeled as fan_off_button are connected along the rungs connected to the line L 1. An illustration of N O and N C pushbutton is shown below the second rung. A label inputs is placed over the first rung. Two rungs are connected between the second and the third vertical lines. From the left, an X I C symbol, and O T L (highlighted in green) are connected in series along the first rung. From the top, the first X I C symbol is labeled as fan_on_button, and <Local:1:I.Data.2>. From the top, O T L is labeled as vent_fan, and <Local:2:O.Data.4>. From the left, an X I C symbol, and O T U (highlighted in green) are connected in series along the second rung. From the top, the second X I C symbol is labeled as fan_off_button, and <Local:1:I.Data.3>. From the top, O T U is labeled as vent_fan, and <Local:2:O.Data.4>. A label ladder logic program is placed over the first rung. A motor starter coil labeled as vent_fan is connected along a rung connected at the left side of the line L 2. An illustration of the vent fan is shown below its symbol. A label output is placed over the rung connected to the line L 2. The table labeled as monitor tags window consists of four rows and four columns and is placed at the bottom. From the left, the first row is labeled as tag name, value, style, and data type. From the left, the second row is labeled as fan_on_button, zero, decimal, and BOOL. From the left, the third row is labeled as fan_off-button, zero, decimal, and BOOL. From the left, the fourth row is labeled as vent_fan, one, decimal, and BOOL.

Return to Figure 15-42

OceanofPDF.com

Figure 15-43 Extended Description (Chapter 15)

Return to Figure 15-43

The ladder diagram consists of four vertical lines. From the left, the first vertical line is labeled as L 1 and the fourth vertical line is labeled as L 2. Four rungs are connected on the right side of the line L 1. From the top, N C pushbutton (highlighted in red) labeled as motor_stop, N O pushbutton (highlighted in green) labeled as motor_start, N O limit switch labeled as enter_L S, and N O limit switch labeled as exit_L are connected along the rungs connected to the line L 1. Illustrations of pushbuttons and limit switches are shown below their symbols. A label inputs is placed over the first rung. Three rungs are connected between the second and the third vertical lines. From the left, an X I C symbol (highlighted in green), the second X I C symbol, and an output energize are connected in series along the first rung. From the top, the first X I C symbol is labeled as motor_stop, and <Local:1:I.Data.0>. From the top, the second X I C symbol is labeled as motor_start, and <Local:1:I.Data.1>. From the top, output energize is labeled as track_motor and <Local:2:O.Data.0>. The third X I C symbol labeled as track_motor and <Local:2:O.Data.0> is connected across the second X I C symbol. The series combination of the fourth X I C symbol and O T L labeled as wash_in_use is connected across the first O T E. From the top, the fourth X I C symbol is labeled as enter_L S and <Local:1:I.Data.2>. From the left, the fifth X I C symbol labeled as wash_in_use, and the second output energize labeled as do_not_enter_sign and <Local:2:O.Data.1> are connected in series along the second rung. From the left, the sixth X I C symbol labeled as exit_L S and <Local:1:I.Data.3>, and the O T U labeled as wash_in_use are connected in series along the third rung. A label ladder logic program is placed over the first rung. Two rungs are connected at the left side of the line L 2. From the top, a light labeled as do_not_enter_sign, and a motor labeled as track_motor are connected along the rungs connected at the left side of the line L 2. A label outputs is placed over the first rung connected to the line L 2. An illustration of a car passing through a washer with a board labeled as do not enter is shown below the second rung connected to the line L 2.

Return to Figure 15-43

OceanofPDF.com

Figure 15-44 Extended Description (Chapter 15)

Return to Figure 15-44

On the left, start push button P B and stop push button P B as input are connected to line L 1. The middle portion labeled program consists of 3 rungs. The first rung has a stop push button P B with address Local:1:I.Data.1, start push button P B with address Local:1:I.Data.0 and run pilot light P L with address Local:2:O.Data.2 connected in series. Run pilot light P L with address Local:2:O.Data.2 is connected across start push button P B. The second rung has a run pilot light P L with address Local:2:O.Data.2, a closed motor 2 with address Local:2:O.Data.1, a closed internal relay, and a motor 1 with address Local:2:O.Data.0 connected in series. Motor 1 with address Local:2:O.Data.0 is connected across closed internal relay. An internal relay L and a motor 1 pilot light P L with address Local:2:O.Data.2 in parallel combination is connected across motor 1. The third rung has a run pilot light P L with address Local:2:O.Data.2, a closed motor 1 with address Local:2:O.Data.0, an internal relay, and motor 2 with address Local:2:O.Data.1 connected in series. Motor 2 with address Local:2:O.Data.1 is connected across internal relay. An internal relay U and motor 2 pilot light with address Local:2:O.Data.4 in parallel combination are connected across motor 2. On the right, motor M 1 with closed O L in series, motor M 2 with a closed O L in series, a run pilot light P L, a motor 1 pilot light P L, and a motor 2 pilot light P L as output are connected to line L 2.

Return to Figure 15-44

OceanofPDF.com

Figure 15-45 Extended Description (Chapter 15)

Return to Figure 15-45

The ladder diagram consists of three vertical lines. From the left, the first vertical line is labeled as L 1. A rung is connected on the right side of the line L 1. An N O limit switch labeled as Limit_switch_1 is connected along the rung connected to the line L 1. Illustration of a limit switch is shown below its symbol. A label input is placed over the rung connected to the line L 1. A rung is connected between the second and the third vertical lines. From the left, an X I C symbol, O N S instruction labeled as storage_1, and A D D instruction are connected in series along the rung. From the top, the X I C symbol is labeled as limit_switch_1, and <Local:1:I.Data.6>. From the top, the A D D instruction is labeled as A D D, add, source A sum, zero (placed on the right side and an arrow points at it from the right side), source B one, dest sum, and zero (placed on the right side and an arrow points at it from the right side).

Return to Figure 15-45

OceanofPDF.com

Figure 15-46 Extended Description (Chapter 15)

Return to Figure 15-46

The right side of the O S R instruction is connected to an output energize O B and an output energize S B. From the top, the O S R instruction is labeled as O S R, one shot rising, and output bit. All three timing diagrams show a rectangular peak, where the rectangular peak has same width for rung-condition-in and storage bit. The rectangular peak of the output bit is less wider than the other two. The rising edge of all the timing diagrams is at the same time. The rising edge of the output bit is labeled as instruction is executed. The falling edge of the output bit is labeled as instruction resets during next scan execution. Three dots are shown in between the line of the rectangular peaks of the first two timing diagrams, while for the output bit, the dots are shown on the right side of the peak.

Return to Figure 15-46

OceanofPDF.com

Figure 15-47 Extended Description (Chapter 15)

Return to Figure 15-47

The right side of the O S F instruction is connected to an output energize O B and an output energize S B. From the top, the O S F instruction is labeled as O S F, one shot falling, and output bit. The first two timing diagrams show two rectangular peaks, where the rectangular peaks have same width for both the diagrams. The rectangular peak of the output bit is less wider than the other two. The rising edge of the first two timing diagrams is at the same time. The rising edge of the output bit is shown when the falling edge of the first rectangular peak of the first two timing diagrams is shown. The rising edge of the output bit is labeled as instruction is executed. The falling edge of the output bit is labeled as instruction resets during next scan execution. Three dots are shown in between the line of the rectangular peaks of the first two timing diagrams, while for the output bit, the dots are shown on the left side of the peak.

Return to Figure 15-47

OceanofPDF.com

Figure 15-48 Extended Description (Chapter 15)

Return to Figure 15-48

The ladder diagram consists of three vertical lines. From the left, the first vertical line is labeled as L 1. Four rungs are connected on the right side of the line L 1. From the top, N O proximity switch labeled as 4_sensor, N O proximity switch labeled as 8_sensor, N O proximity switch labeled as 12_sensor, and N O pushbutton labeled as clear_sums are connected along the rungs connected to the line L 1. Illustrations of proximity switches and the pushbutton are shown below their symbols. A label inputs is placed over the first rung connected to the line L 1. Six rungs are connected between the second and the third vertical lines. From the left, an X I C symbol labeled as 4_sensor and <Local:1:I.Data.0>, O N S instruction labeled as one_shot_bit, and A D D instruction are connected in series along the first rung. From the top, the A D D instruction is labeled as A D D, add, source A sum_O N S, zero (placed on the right side), source B four, dest sum_O N S, and zero (placed on the right side). From the left, an X I O symbol labeled as 8_sensor and <Local:1:I.Data.1>, O S R instruction, and output energize O B are connected in series along the second rung. The right side of the O S R instruction is also connected to the output energize S B. From the top, the O S R instruction is labeled as O S R, one shot rising, storage bit storage_O S R, and output bit output_bit_O S R. From the left, the second X I C symbol labeled as output_bit_O S R, and A D D instruction are connected in series along the third rung. From the top, the A D D instruction is labeled as A D D, add, source A sum_O S R, zero (placed on the right side), source B eight, dest sum_O S R, and zero (placed on the right side). From the left, the second X I O symbol labeled as 12_sensor and <Local:1:I.Data.2>, O S F instruction, and output energize O B are connected in series along the fourth rung. The right side of the O S F instruction is also connected to the output energize S B. From the top, the O S F instruction is labeled as O S F, one shot falling, storage bit storage_O S F, and output bit output_bit_O S F. From the left, the third X I C symbol labeled as output_bit_O S F, and A D D instruction are connected in series along the fifth rung. From the top, the A D D instruction is labeled as A D D, add, source A sum_O S F, zero (placed on the right side), source B 12, dest sum_O S F, and zero (placed on the right side). From the left, the fourth X I C symbol labeled as clear_sums_P B and <Local:1:I.Data.3>, and three C L R instructions are connected in series along the sixth rung. From the top, the first C L R instruction is labeled as C L R, clear, dest sum_O N S, and zero (placed on the right side). From the top, the second C L R instruction is labeled as C L R, clear, dest sum_O S R, and zero (placed on the right side). From the top, the third C L R instruction is labeled as C L R, clear, dest sum_O S F, and zero (placed on the right side).

Return to Figure 15-48

OceanofPDF.com

Figure 15-49 Extended Description (Chapter 15)

Return to Figure 15-49

On the left: On the left, push buttons, P B 1, P B 2, P B 3, and P B 4 are connected to an input 120 volts of A C source. The middle portion consists of three rungs. The first rung shows a push button P B 1 connected to a pilot light P L 1 in series, and a closed push button P B 4 is connected across push button P B 1. The second rung shows a push button P B 2, a closed push button P B 3, and a pilot light P L 2 connected in series. The third rung shows a push button P B 1, a closed push button P B 4, and a pilot light P L 3 connected in series. A closed push button P B 3 and a push button P B 2 are connected across P B 1 and closed P B 4 push buttons, respectively. On the right, pilot lights, P L 1, P L 2, and P L 3 are connected to an output line labeled N.

On the right: A truth table consists of eight columns and six rows. The first two cells in column 1 are merged and labeled input device state. From column 2 through column 5, the top cells are merged and labeled input state. From column 6 through column 8, the top cells are merged and labeled output state. From row 2 through row 5, the row headers are No P Bs pressed, P B 1 pressed, P B 3 pressed, and P B 2, 3, 4 pressed, respectively. The entries in Row 2 are as follows: P B 4, P B 3, P B 2, P B 1, P L 3, P L 2, and P L 1. The cells for rows 3 through row 6 are blank.

Return to Figure 15-49

OceanofPDF.com

Figure 15-50 Extended Description (Chapter 15)

Return to Figure 15-50

The positive terminal of batter labeled 24 V D C is connected to the left end of the latch coil, through an N O float switch labeled float S W 1 and an N O float switch labeled as float S W 2. The negative terminal of the battery is connected to the right side of the latch coil. An N O temperature switch labeled as temp S w is connected across the series connection of the two float switches, from the top. The parallel connection of N O proximity switches labeled as sensor S w 1 and sensor S w 2 is connected to an N C pressure labeled as pressure S w in series. This series connection is connected across the series connection of the two float switches, from the bottom. The series connection of an N O pushbutton labeled as reset button and unlatch coil is connected across the series connection of the two float switches and latch coil, from the bottom. The second circuit shows series connection of an a c voltage source labeled as 120 V A C, N O relay contact, and an alarm.

Return to Figure 15-50

OceanofPDF.com

Figure 15-51 Extended Description (Chapter 15)

Return to Figure 15-51

The illustration in the left shows a pipe with a valve at the opening labeled as solenoid A filling a tank with a fluid. A full tank sensor is placed at the right side and near the mouth of the tank. An empty tank sensor is placed at the bottom-left side of the tank. A pipe line is placed at the bottom of the tank with a valve labeled as solenoid B. A control panel with a red dot labeled as stop, a green dot labeled as fill, and a blue dot labeled as empty, on the left side of the solenoid B. Illustrations of the valves are shown next to their symbols. The ladder diagram consists of a rung connected along the vertical lines L 1 and L 2. From the left, an N C pushbutton labeled as stop, N O pushbutton labeled as fill, an N C proximity switch labeled as full tank sensor, and a relay coil labeled as 1 C R. An N O relay contact labeled as I C R sub 1 is connected across the N O pushbutton labeled as fill. The series connection of an N O relay contact labeled as 1 C R sub 2, and a solenoid valve labeled as solenoid A is connected across the series connection of the N O pushbutton labeled as fill, N C proximity switch labeled as full tank sensor, and relay coil labeled as 1 C R. The series connection of an N O pushbutton labeled as empty, N O proximity switch labeled as empty tank sensor, and relay coil labeled as 2 C R is connected across the series connection of the N O relay contact labeled as 1 C R sub 2, and the solenoid valve labeled as solenoid A. An N O relay contact labeled as 2 C R sub 2 is connected across the N O pushbutton labeled as empty. The series connection of the N O relay contact labeled as 2 C R sub 2, and the solenoid valve labeled as solenoid B is connected across the series connection of the N O pushbutton labeled as empty, N O proximity switch labeled as empty tank sensor, and relay coil labeled as 2 C R.

Return to Figure 15-51

OceanofPDF.com

Figure 15-52 Extended Description (Chapter 15)

Return to Figure 15-52

From the top, the labels are name, pump_timer (enclosed in a rectangle); description, an empty rectangle with an upwards pointing arrowhead placed above a downwards pointing arrow on the right side); and members, data type size: 12 byte (s). From the top, the first column of the table is labeled as name, P R E, A C C, E N, T T, and D N. From the top, the second column is labeled as data type, DINT, DINT, BOOL, BOOL, and BOOL. From the top, the first row of the third column is labeled as style and the remaining rows are labeled as decimal. The first row of the fourth column is labeled as description.

Return to Figure 15-52

OceanofPDF.com

Figure 15-53 Extended Description (Chapter 15)

Return to Figure 15-53

The toolbar shows an arrowhead pointing left at the top-left side and an arrowhead pointing right at the top-right side. Three symbols and six labels enclosed in rectangles are placed relatively on the left side between the arrowheads. From left to right, the symbols are, an upper-case letter I rotated at ninety degrees, same symbol as the first but includes a rectangle at the bottom of the middle line of the letter I, and same symbol as the second but includes a horizontal line dividing the rectangle into half. From left to right, the labels are T O N (highlighted in blue), T O F (highlighted in blue), R T O (highlighted in blue), C T U, C T D, and R E S (highlighted in blue). Seven labels are shown at the bottom. From left to right, the labels are arrow-head facing, arrow-head facing right, favorites, add-on, alarms, bit, and timer or counter (highlighted in blue). The left side of the T O N instruction has three dots labeled as input side of rung. The right side of the T O N instruction is connected to a vertical line on the left side, through an output energize E N. The right side of the T O N is also connected to an output energize D N, through second terminal. From top to bottom, the T O N instruction is labeled as T O N, timer on delay, timer solenoid_delay, preset 3000 (an arrow pointing at 3000 from the right side), and accum zero (an arrow pointing at zero from the right side). A downwards pointing arrow labeled as tag name points at the label solenoid_delay.

Return to Figure 15-53

OceanofPDF.com

Figure 15-54 Extended Description (Chapter 15)

Return to Figure 15-54

The T O N instruction has a label diverted gate solenoid delay timer placed over the instruction. The right side of the T O N instruction is connected to a vertical line on the left side, through an output energize E N. The right side of the T O N is also connected to an output energize D N, through second terminal. From top to bottom, the T O N instruction is labeled as T O N, timer on delay, timer solenoid_delay, preset 3000 (an arrow pointing at 3000 from the right side), and accum zero (an arrow pointing at zero from the right side). From top to bottom, the dialogue box new tag is labeled as name, solenoid_delay (on the right side and enclosed in a rectangle); description, diverted gate solenoid delay timer (on the right side and enclosed in a rectangle); tag type, base (on the right side and enclosed in a rectangle with drop-down button); data type, timer (on the right side and enclosed in a rectangle); and scope, controller (on the right side and enclosed in a rectangle).

Return to Figure 15-54

OceanofPDF.com

Figure 15-55 Extended Description (Chapter 15)

Return to Figure 15-55

There are four vertical lines in the ladder diagram. From the left, the firs vertical line is labeled as L 1 and the second vertical line is labeled as L 2. A label input is placed over a rung connected to the line L 1. An N C switch labeled as Timer_Sw is connected along the rung connected to the L 1. An illustration of the switch is placed below its symbol. Four rungs are connected between the second and the third vertical lines with a label ladder logic program over them. From the left, an X I C symbol (highlighted in green) labeled as Timer_Sw, T O N instruction, and output energize E N are connected in series along the first rung. The right side of the T O N is connected to an output energize D N. From the top, the T O N is labeled as T O N, timer on delay, timer Status_Timer, preset 10,000 (an arrow pointing at it from the right side), and accum 5,000 (an arrow pointing at it from the right side). From the left, X I C symbol (highlighted in green) labeled as Status_Timer.E N and an O T E (highlighted in green) labeled as E N_P L and <Local:2:O.Data.1> are connected in series along the second rung. From the left, X I C symbol (highlighted in green) labeled as Status_Timer.T T and an O T E (highlighted in green) labeled as T T_P L and <Local:2:O.Data.2> are connected in series along the third rung. From the left, X I C symbol labeled as Status_Timer.D N and an O T E labeled as D N_P L and <Local:2:O.Data.3> are connected in series along the fourth rung. Three rungs are connected at the left side of the line L 2 with a label outputs over them. From the top, relay coil E N labeled as E N_P L, relay coil T T labeled as T T_P L, and relay coil D N labeled as D N_P L are connected along the rungs at the left side of the line L 2. From the top, the first column of the table is labeled as tag name, Timer_Sw, E N_P L, T T_P L, D N_P L, (negative sign enclosed in a square) Status_Timer, (positive sign enclosed in a square and indented to level 1) Status_Timer.P R E, (positive sign enclosed in a square and indented to level 1) Status_Timer.A C C, (indented to level 1) Status_Timer.E N, (indented to level 1) Status_Timer.T T, and (indented to level 1) Status_Timer.D N. From the top, the second column is labeled as value, 1, 1, 1, 0, {...}, 10,000, 500, 1, 1, and 0. For the third column, the first row is labeled as style, the sixth row is unlabeled, and the remaining rows are labeled as decimal. From the top, the fourth column is labeled as data type, BOOL, BOOL, BOOL, BOOL, TIMER, DINT, DINT, BOOL, BOOL, and BOOL.

Return to Figure 15-55

OceanofPDF.com

Figure 15-56 Extended Description (Chapter 15)

Return to Figure 15-56

There are four vertical lines in the ladder diagram. From the left, the firs vertical line is labeled as L 1 and the second vertical line is labeled as L 2. A label input is placed over two rungs connected to the line L 1. From the top, an N O proximity switch labeled as SOL_Energize_Sensor and an N O proximity switch labeled as SOL_Deenergize_Sensor connected along the rung connected to the L 1. Two rungs are connected between the second and the third vertical lines with a label ladder logic program over them. From the left, an X I C symbol labeled as SOL_Energize_Sensor and <Local:1:I.Data.3>, an X I C symbol (highlighted in green) labeled as SOL_Deenergize_Sensor and <Local:1:I.Data.6>, T O N instruction, and output energize E N are connected in series along the first rung. The right side of the T O N is connected to an output energize D N. From the top, the T O N is labeled as T O N, timer on delay, timer T_SOL_Delay, preset 3,000 (an arrow pointing at it from the right side), and accum zero (an arrow pointing at it from the right side). An X I C symbol labeled as T_SOL_Delay.E N is connected across the X I C symbol labeled as SOL_Energize_Sensor. From the left, X I C symbol labeled as T_SOL_Delay.D N and an O T E labeled as SOL_Gate and <Local:2:O.Data.2> are connected in series along the second rung. A rung is connected at the left side of the line L 2 with a label output over it. A solenoid labeled as SOL_Gate is connected along the rung at the left side of the line L 2. An illustration od movable gate is shown below the solenoid. From the top, the first column of the table is labeled as tag name, SOL_Energize_Sensor, SOL_Deenergize_Sensor, SOL_Gate, (negative sign enclosed in a square) T_SOL_Delay, (positive sign enclosed in a square and indented to level 1) T_SOL_Delay.PRE, (positive sign enclosed in a square and indented to level 1) T_SOL_Delay.ACC, (indented to level 1) T_SOL_Delay.EN, (indented to level 1) T_SOL_Delay.T T, and (indented to level 1) T_SOL_Delay.D N. From the top, the second column is labeled as value, 0, 1, 0, {...}, 3,000, 0, 0, 0, and 0. For the third column, the first row is labeled as style, the fifth row is unlabeled, and the remaining rows are labeled as decimal. From the top, the fourth column is labeled as data type, BOOL, BOOL, BOOL, TIMER, DINT, DINT, BOOL, BOOL, and BOOL.

Return to Figure 15-56

OceanofPDF.com

Figure 15-57 Extended Description (Chapter 15)

Return to Figure 15-57

There four vertical lines. From the left, the first vertical line is labeled as L 1 and the fourth vertical line is labeled as L 2.An N O pushbutton labeled as Timer_Button is connected along a rung connected at the right side of the vertical line L 1. A label input is placed over the rung connected to the line L 1. There are two rungs connected between the second and the third vertical lines. From the left, X I C symbol labeled as Timer_Button and <Local:1:I.Data.0>, O T L labeled as Green_P L and <Local:2:O.Data.3>, T O N instruction, and an output energize E N (highlighted in green) are connected in series along the first rung. An X I C symbol labeled as Pilot_Light_Timer.E N is connected across the X I C symbol labeled as Timer_Button. The right side of the T O N instruction is connected to an output energize D N. From the top, T O N instruction is labeled as T O N, timer on delay, timer Pilot_Light_Timer, preset 20,000 (an arrow pointing at it from the right side), and accum zero (an arrow pointing at it from the right side). From the left, X I C symbol labeled as Pilot_Light_Timer.D N, R E S instruction labeled as Pilot_Light_Timer, and O T L labeled as Green_P L and <Local:2:O.Data.3> are connected in series along the second rung. A label ladder logic program is placed over the first rung. A green light labeled as Green_P L is connected along a rung that is connected at the left side of the vertical line L 2. A label output is placed over the rung connected to the line L 2.

Return to Figure 15-57

OceanofPDF.com

Figure 15-58 Extended Description (Chapter 15)

Return to Figure 15-58

The ladder diagram using three vertical lines. From the left, the third vertical line is labeled as L 2. Six rungs are connected between the first and the second vertical line. A label ladder logic program is placed over the first rung connected between the first vertical line and the second vertical line. From the left, X I O symbol (highlighted in green) labeled as Amber_Light_Timer.D N, T O N instruction, and an output energize E N (highlighted in green) are connected in series along the first rung. The right side of the first T O N instruction is connected to an output energize D N. From the top, the first T O N instruction is labeled as T O N, timer on delay, timer Red_Light_Timer, preset 30,000 (an arrow pointing at 30,000 from the right side), and accum zero (an arrow pointing at zero from the right side). From the left, X I C symbol labeled as Red_Light_Timer.D N, the second T O N instruction, and an output energize E N are connected in series along the second rung. The right side of the second T O N instruction is connected to an output energize D N. From the top, the second T O N instruction is labeled as T O N, timer on delay, timer Green_Light_Timer, preset 25,000 (an arrow pointing at 25,000 from the right side), and accum zero (an arrow pointing at zero from the right side). From the left, X I C symbol labeled as Green_Light_Timer.D N, the third T O N instruction, and an output energize E N are connected in series along the third rung. The right side of third T O N instruction is connected to an D N instruction. From the top, the third T O N instruction is labeled as T O N, timer on delay, timer Amber_Light_Timer, preset 50,000 (an arrow pointing at 50,000 from the right side), and accum zero (an arrow pointing at zero from the right side). From the left, X I C symbol (highlighted in green) labeled as Red_Light_Timer.E N, X I O symbol (highlighted in green) labeled as Red_Light_Timer.D N, and an O T E (highlighted in green) labeled as Red_Light and <Local:2:O.Data.0> are connected in series along the fourth rung. From the left, X I C symbol labeled as Green_Light_Timer.E N, X I O symbol (highlighted in green) labeled as Green_Light_Timer.D N, and an O T E labeled as Green_Light and <Local:2:O.Data.1> are connected in series along the fifth rung. From the left, X I C symbol labeled as Amber_Light_Timer.E N, X I O symbol (highlighted in green) labeled as Amber_Light_Timer.D N, and an O T E labeled as Amber_Light and <Local:2:O.Data.2> are connected in series along the sixth rung. Three rungs are connected at the left side of the vertical line L 2. From the top, lights labeled as Red_Light, Amber_Light, and Green_Light are connected along the rungs connected at the left side of the line L 2. A label outputs is placed over the first rung connected to the line L 2.

Return to Figure 15-58

OceanofPDF.com

Figure 15-59 Extended Description (Chapter 15)

Return to Figure 15-59

From the top, the first column is labeled as tag name, amber_light_timer, green_light_timer, red_light_timer, (indented to level 1) red_light_timer.P R E, (indented to level 1) red_light_timer.A C C, (indented to level 1) red_light_timer.E N, (indented to level 1) red_light_timer.T T, (indented to level 1) red_light_timer.D N, red_light, green_light, and amber_light. A positive symbol enclosed in a square is shown at the left side of the second, third, fifth, and the sixth label. A negative symbol enclosed in a square is shown at the left side of the fourth label. From the top, the second column is labeled as {...}, {...}, {...}, 30,000, 0, 1, 1, 0, 1, 0, and 0. From the top the first row of the third column is labeled as style. The last eight rows of the third column are labeled as decimal. From the top, the fouth column is labeled as data type, timer, timer, timer, dint, dint, bool, bool, bool, bool, bool, and bool.

Return to Figure 15-59

OceanofPDF.com

Figure 15-60 Extended Description (Chapter 15)

Return to Figure 15-60

The right side of the instruction is connected to the vertical line, through an output energize E N. The right side of the instruction is connected to an output energize D N, through the second terminal. From top to bottom, the T O F instruction is labeled as T O F, timer, off delay, timer sample_T O F, preset 5000 (an arrow pointing at 5000 from the right side), and accum zero (an arrow pointing at zero from the right side). A downwards pointing arrow labeled as tag name points at the label sample_T O F.

Return to Figure 15-60

OceanofPDF.com

Figure 15-61 Extended Description (Chapter 15)

Return to Figure 15-61

There four vertical lines. From the left, the first vertical line is labeled as L 1 and the fourth vertical line is labeled as L 2. An N O pushbutton labeled as Timer_Button is connected along a rung connected at the right side of the vertical line L 1. A label input is placed over the rung connected to the line L 1. There are two rungs connected between the second and the third vertical lines. From the left, X I C symbol labeled as Timer_Button and <Local:1:I.Data.0>, T O F instruction, and an output energize E N (highlighted in green) are connected in series along the first rung. The right side of the T O F instruction is connected to an output energize D N. From the top, T O F instruction is labeled as T O F, timer off delay, timer Pilot_Light_Timer, preset 20,000 (an arrow pointing at it from the right side), and accum zero (an arrow pointing at it from the right side). From the left, X I C symbol labeled as Pilot_Light_Timer.D N, and O T E labeled as Green_P L and <Local:2:O.Data.3> are connected in series along the second rung. A label ladder logic program is placed over the first rung. A green light labeled as Green_P L is connected along a rung that is connected at the left side of the vertical line L 2. A label output is placed over the rung connected to the line L 2.

Return to Figure 15-61

OceanofPDF.com

Figure 15-62 Extended Description (Chapter 15)

Return to Figure 15-62

There are four vertical lines. From the left, the first vertical line is labeled as L 1 and the fourth vertical line is labeled as L 2. Two rungs are connected at right side of the line L 1 with a label inputs placed over the first rung. From the top, N O pushbutton (highlighted in green) labeled as Over_On_Button, and N C pushbutton (highlighted in red) labeled as Oven_off_Button. An illustration of the buttons is placed below the second rung. Five rungs are connected between the second and the third vertical lines with a label ladder logic program placed over the first rung. From the left, X I C symbol labeled as Oven_On_Button and <Local:1:I.Data.1>, X I C symbol (highlighted in green) labeled as Oven_Off_Button and <Local:1:I.Data.2>, and an O T E labeled as Oven_On_P L and <Local:2:O.Data.2> are connected in series along the first rung. An X I C symbol labeled as Oven_On_P L and <Local:2:O.Data.2> is connected across the X I C symbol labeled as Oven_On_Button and <Local:1:I.Data.1>. From the left, X I C symbol labeled as Oven_On_P L and <Local:2:O.Data.2>, T O N instruction, and an output energize E N are connected in series along the second rung. The right side of the T O N is connected to an output energize D N. From the top, the first T O N is labeled as T O N, timer on delay, timer Timer_Heater, preset 10,000 (an arrow pointing at it from the right side), and accum zero (an arrow pointing at it from the right side). Series connection of the second T O N instruction and the second output energize is connected across the series connection of the first T O N and the first output energize E N. The right side of the second T O N is connected to an output energize D N. From the top, the second T O N is labeled as T O N, timer on delay, timer Timer_Cooling, preset 300,000 (an arrow pointing at it from the right side), and accum zero (an arrow pointing at it from the right side). From the left, an X I C symbol labeled as Timer_Heat.T T and an O T E labeled as Warning_Horn and <Local:2:O.data.3> are connected in series along the third rung. From the left, an X I C symbol labeled as Timer_Heat.D N and an O T E labeled as Heater_Contactor and <Local:2:O.data.4> are connected in series along the fourth rung. From the left, an X I C symbol labeled as Timer_Cooling.D N and an O T E labeled as Fan_Motor and <Local:2:O.data.5> are connected in series along the fifth rung. Four rungs are connected at the left side of the vertical line L 2 with a label outputs placed over the first rung. From the top, red light labeled as Oven_On_P L, horn labeled as Warning_Horn, heater labeled as Heater_Contactor, and fan labeled as Fan_Motor are connected along the rungs at the left side of the line L 2.

Return to Figure 15-62

OceanofPDF.com

Figure 15-63 Extended Description (Chapter 15)

Return to Figure 15-63

From the top, the first column is labeled as tag name, warning_horn, heater_contactor, fan_motor, oven_on_P L, oven_on_button, oven_off_button, (a positive sign enclosed in a square) timer_heat, and (a positive sign enclosed in a square) timer_cooling. The last two rows of the second, the third, and the fourth columns are unlabeled. From the top, the second column is labeled as alias for, Local:2:O.Data.3, Local:2:O.Data.4, Local:2:O.Data.5, Local:2:O.Data.2, Local:1:I.Data.1, and Local:1:I.Data.2. From the top, the third column is labeled as base tag, Local:2:O.Data.3, Local:2:O.Data.4, Local:2:O.Data.5, Local:2:O.Data.2, Local:1:I.Data.1, and Local:1:I.Data.2. From the top, the fourth column is labeled as data type, BOOL, BOOL, BOOL, BOOL, BOOL, BOOL, TIMER, and TIMER. The first row of the fifth column is labeled as style and the remaining rows except for the last two are labeled as decimal.

Return to Figure 15-63

OceanofPDF.com

Figure 15-64 Extended Description (Chapter 15)

Return to Figure 15-64

The right side of the R T O is also connected to an output energize D N, through second terminal. From top to bottom, the R T O instruction is labeled as RT O, retentive timer on, timer SOL_On_Timer, preset 10,000 (an arrow pointing at 10,000 from the right side), and accum zero (an arrow pointing at zero from the right side). An output energize R E S labeled as SOL_On_Timer is connected to the vertical line and is placed below the R T O instruction. The left side of the R E S instruction is connected to three dots. A label input side of rung is placed at the bottom-left side of the R T O instruction. A double-headed arrow points at the right side of the label SOL_On_Timer in the R T O instruction and the label SOL_On_Timer over the R E S instruction.

Return to Figure 15-64

OceanofPDF.com

Figure 15-65 Extended Description (Chapter 15)

Return to Figure 15-65

There are four vertical lines. From the left, the first vertical line is labeled as L 1 and the fourth vertical line is labeled as L 2. Two rungs are connected at right side of the line L 1 with a label inputs placed over the first rung. From the top, N O limit switch labeled as Limit_Switch, and N O pushbutton labeled as Reset_L S_Timer are connected along the rungs connected to the line L 1. Five rungs are connected between the second and the third vertical lines with a label ladder logic program placed over the first rung. From the left, X I C symbol labeled as Limit_Switch and <Local:1:I.Data.7>, R T O instruction, and output energize E N are connected in series along the first rung. The right side of the R T O is connected to an output energize D N. From the top, the R T O is labeled as R T O, retentive timer on, timer L S_Timer, preset 120,000 (an arrow pointing at it from the right side), and accum zero (an arrow pointing at it from the right side). From the left, X I C symbol labeled as L S_Timer.E N and an O T E labeled as L S_E N_P L and <Local:2:O.Data.0> are connected in series along the second rung. From the left, X I C symbol labeled as L S_Timer.T T and an O T E labeled as L S_T T_P L and <Local:2:O.Data.1> are connected in series along the third rung. From the left, X I C symbol labeled as L S_Timer.D N and an O T E labeled as L S_Alarm and <Local:2:O.Data.2> are connected in series along the fourth rung. From the left, X I C symbol labeled as Reset_L S_Timer and <Local:1:I.Data.2> is connected to an R E S labeled as L S_Timer in series along the fifth rung. Three rungs are connected at the left side of the line L 2 with a label outputs placed over the first rung. From the top, light labeled as L S_E N_P L, light labeled as L S_T T_P L, and alarm labeled as L S_Alarm are connected along the rungs at the left side of the line L 2.

Return to Figure 15-65

OceanofPDF.com

Figure 15-66 Extended Description (Chapter 15)

Return to Figure 15-66

From the top, the first column is labeled as tag name, (a negative sign enclosed in a square) L S_Timer, (a positive sign enclosed in a square and indented to level 1) L S_Timer.P R E, (a positive sign enclosed in a square and indented to level 1) L S_Timer.A C C, (indented to level 1) L S_Timer.E N, (indented to level 1) L S_Timer.T T, (indented to level 1) L S_Timer.D N, limit_switch, L S_E N_P L, L S_T T_P L, and L S_alarm. From the top, the second column is labeled as value, {...}, 120,000, 0, 0, 0, 0, 0, 0, 0, and 0. For the third column, the first row is labeled as style, the second row is unlabeled, and the remaining rows are labeled as decimal. From the top, the furth column is labeled as data type, TIMER, DINT, DINT, BOOL, BOOL, BOOL, BOOL, BOOL, BOOL, and BOOL.

Return to Figure 15-66

OceanofPDF.com

Figure 15-67 Extended Description (Chapter 15)

Return to Figure 15-67

There are four vertical lines. From the left, the first vertical line is labeled as L 1 and the fourth vertical line is labeled as L 2. Two rungs are connected at right side of the line L 1 with a label inputs placed over the first rung. From the top, N O pushbutton labeled as Stop_P B, and N O pushbutton labeled as Start_P B are connected along the rungs connected to the line L 1. Seven rungs are connected between the second and the third vertical lines with a label ladder logic program placed over the first rung. From the left, X I C symbol labeled as Stop_PB and <Local:1:I.Data.0>, X I C symbol labeled as Start_PB and <Local:1:I.Data.1>, and an O T E labeled as Internal_Relay are connected in series along the first rung. From the left, X I C symbol labeled as Internal_Relay, T O N instruction, and output energize E N are connected in series along the second rung. The right side of the first T O N is connected to an output energize D N. From the top, the first T O N is labeled as T O N, timer on delay, timer Timed_Event_1, preset 3,000, and accum zero. From the left, X I C symbol labeled as Timed_Event_1.D N, the second T O N instruction, and an output energize E N are connected in series along the third rung. The right side of the second T O N is connected to an output energize D N. From the top, the second T O N is labeled as T O N, timer on delay, timer Timed_Event_2, preset 10,000, and accum zero. From the left, X I C symbol labeled as Timed_Event_2.D N, the third T O N instruction, and an output energize E N are connected in series along the fourth rung. The right side of the third T O N is connected to an output energize D N. From the top, the third T O N is labeled as T O N, timer on delay, timer Timed_Event_3, preset 5,000, and accum zero. From the left, X I C symbol labeled as Timed_Event_1.D N and an O T E labeled as Event_1_Complete_P L and <Local:2:0.Data.0> are connected in series along the fifth rung. From the left, X I C symbol labeled as Timed_Event_2.D N and an O T E labeled as Event_2_Complete_P L and <Local:2:0.Data.1> are connected in series along the sixth rung. From the left, X I C symbol labeled as Timed_Event_3.D N and an O T E labeled as Process_Complete_P L and <Local:2:0.Data.2> are connected in series along the seventh rung. Three rungs are connected at the left side of the line L 2 with a label outputs placed over the first rung. From the top, green lights labeled as Event_1, Event_2, and red light labeled as complete are connected along the rungs at the left side of the line L 2.

Return to Figure 15-67

OceanofPDF.com

Figure 15-68 Extended Description (Chapter 15)

Return to Figure 15-68

There are four vertical lines. From the left, the first vertical line is labeled as L 1 and the fourth vertical line is labeled as L 2. A rung is connected at right side of the line L 1 with a label input placed over the first rung. An N O limit switch labeled as Limit_Switch is connected along the rung connected to the line L 1. Three rungs are connected between the second and the third vertical lines with a label ladder logic program placed over the first rung. From the left, X I C symbol labeled as Limit_Switch and <Local:1:I.Data.0>, X I O symbol labeled as Timer_Reciprocating_On.D N, the first T O N instruction, and output energize E N are connected in series along the first rung. The right side of the first T O N is connected to an output energize D N. From the top, the first T O N is labeled as T O N, timer on delay, timer Timer_Reciprocating_Off, preset 2,000, and accum zero. From the left, X I C symbol labeled as Timer_Reciprocating_Off.D N, the second T O N, and an output energize E N are connected in series along the second rung. The right side of the second T O N is connected to an output energize D N. From the top, the second T O N is labeled as T O N, timer on delay, timer Timer_Reciprocating_On, preset 2,000, and accum zero. From the left, X I C symbol labeled as Timer_Reciprocating_On.D N and an O T E labeled as Alarm_P L and <Local:2:0.Data.0> are connected in series along the third rung. A rung is connected at the left side of the line L 2 with a label output placed over the rung. A light labeled as Alarm_P L is connected along the rung at the left side of the line L 2.

Return to Figure 15-68

OceanofPDF.com

Figure 15-69 Extended Description (Chapter 15)

Return to Figure 15-69

The ladder diagram consists of three rungs connected between two vertical lines L 1 and L 2. From left to right, N O switch labeled as on or off switch and relay coil C R are connected in series along the first rung. From left to right, N O relay contact, N C pressure switch labeled as low pressure switch, and a relay coil T R labeled as on-delay timer are connected in series along the second rung. From left to right, N O temperature switch T R and alarm are connected in series along the third rung.

Return to Figure 15-69

OceanofPDF.com

Figure 15-70 Extended Description (Chapter 15)

Return to Figure 15-70

The toolbar shows an arrowhead pointing left at the top-left side and an arrowhead pointing right at the top-right side. Three symbols and six labels enclosed in rectangles are placed relatively on the left side between the arrowheads. From left to right, the symbols are, an upper-case letter I rotated at ninety degrees, same symbol as the first but includes a rectangle at the bottom of the middle line of the letter I, and same symbol as the second but includes a horizontal line dividing the rectangle into half. From left to right, the labels are T O N, T O F, R T O, C T U (highlighted in blue), C T D (highlighted in blue), and R E S (highlighted in blue). Seven labels are shown at the bottom. From left to right, the labels are arrow-head facing, arrow-head facing right, favorites, add-on, alarms, bit, and timer or counter (highlighted in blue). The left side of the C T U instruction has three dots. A rectangular wave varying between false and true is placed at the left side of the three dots. The right side of the C T U instruction is connected to a vertical line on the left side, through an output energize C U. The right side of the C T U is also connected to an output energize D N, through second terminal. From top to bottom, the C T U instruction is labeled as C T U, count up, counter package_counter, preset 24 (an arrow pointing at 24 from the right side), and accum zero (an arrow pointing at zero from the right side). An output energize R E S labeled as package_counter is connected to the vertical line and is placed below the C T U instruction. The left side of the R E S instruction is connected to three dots. A label input side of rung is placed at the bottom-left side of the C T U instruction. A downwards pointing arrow labeled as tag name points at the label package_counter. A double-headed arrow points at the right side of the C T U instruction and the label package_counter over the R E S instruction.

Return to Figure 15-70

OceanofPDF.com

Figure 15-71 Extended Description (Chapter 15)

Return to Figure 15-71

From the top, the first column is labeled as tag name, (a negative sign enclosed in a square) Part_Counter, (a positive sign enclosed in a square and indented to level 1) Part_Counter.P R E, (a positive sign enclosed in a square and indented to level 1) Part_Counter.A C C, (indented to level 1) Part_Counter.C U, (indented to level 1) Part_Counter.C D, (indented to level 1) Part_Counter.D N, (indented to level 1) Part_Counter.O V, and (indented to level 1) Part_Counter.U N. From the top, the second column is labeled as data type, COUNTER, DINT, DINT, BOOL, BOOL, BOOL, BOOL, and BOOL. For the third column, the first row is labeled as style and the remaining rows are labeled as decimal.

Return to Figure 15-71

OceanofPDF.com

Figure 15-72 Extended Description (Chapter 15)

Return to Figure 15-72

The C T U instruction has a label 24 bottle counter placed over the instruction. The right side of the C T U instruction is connected to a vertical line on the left side, through an output energize C U. The right side of the C T U is also connected to an output energize D N, through second terminal. From top to bottom, the C T U instruction is labeled as C T U, count up, counter package_counter, preset 24 (an arrow pointing at 24 from the right side), and accumulator zero (an arrow pointing at zero from the right side). From top to bottom, the dialogue box new tag is labeled as name, package_counter (on the right side and enclosed in a rectangle); description, 24 bottle counter (on the right side and enclosed in a rectangle); tag type, base (on the right side and enclosed in a rectangle with drop-down button); data type, counter (on the right side and enclosed in a rectangle); and scope, main program (on the right side and enclosed in a rectangle).

Return to Figure 15-72

OceanofPDF.com

Figure 15-73 Extended Description (Chapter 15)

Return to Figure 15-73

There four vertical lines. From the left, the first vertical line is labeled as L 1 and the fourth vertical line is labeled as L 2. Two rungs are connected to the right side of the line L 1. From the top, N O proximity switch labeled as Bottle_Sensor, and N O pushbutton labeled as Reset_Button is connected along the rungs connected at the right side of the vertical line L 1. A label inputs is placed over the first rung connected to the line L 1. Illustrations of the button and the bottle sensor is placed below their symbols. There are four rungs connected between the second and the third vertical lines. From the left, X I C symbol labeled as Bottle_Sensor and <Local:1:I.Data.1>, C T U instruction, and an output energize C U are connected in series along the first rung. The right side of the C T U instruction is connected to an output energize D N. From the top, C T U instruction is labeled as C T U, count up, counter Package_Counter, preset 24 asterisk, and accum zero asterisk. From the left, X I C symbol labeled as 24 bottle counter and Package_Counter.C U, and O T E labeled as Increment_P L and <Local:2:O.Data.4> are connected in series along the second rung. From the left, X I C symbol labeled as 24 bottle counter and Package_Counter.D N, and O T E labeled as Preset_Reached_P L and <Local:2:O.Data.5> are connected in series along the third rung. From the left, X I C symbol labeled as Reset_Button and <Local:1:I.Data.3>, and R E S labeled as 24 bottle counter and Package_Counter are connected in series along the fourth rung. A label ladder logic program is placed over the first rung. Two rungs are connected at the left side of the line L 2. From the top, lights labeled as Increment_P L and Preset_Reached_P L are connected along the rungs that are connected at the left side of the vertical line L 2. A label outputs is placed over the first rung connected to the line L 2.

Return to Figure 15-73

OceanofPDF.com

Figure 15-74 Extended Description (Chapter 15)

Return to Figure 15-74

There are four vertical lines. From the left, the first vertical line is labeled as L 1 and the fourth vertical line is labeled as L 2. A rung is connected at right side of the line L 1 with a label input placed over the first rung. An N O proximity switch labeled as Container_Sensor is connected along the rung connected to the line L 1. Three rungs are connected between the second and the third vertical lines with a label ladder logic program placed over the first rung. From the left, an X I C symbol labeled as Container_Sensor and <Local:1:I.Data.2>, C T U instruction, and output energize C U are connected in series along the first rung. The right side of the C T U is connected to an output energize D N. From the top, the first C T U is labeled as C T U, count up, counter Container_Counter_Counts, preset 6 (an arrow pointing at it from the right side), and accum zero (an arrow pointing at it from the right side). The series connection of the second C T U and an output energize C U is connected across the series connection of the first C T U and the first output energize C U. The right side of the second C T U is connected to an output energize D N. From the top, the second C T U is labeled as C T U, count up, counter Container_Counter_Max, preset 11 (an arrow pointing at it from the right side), and accum zero (an arrow pointing at it from the right side). From the left, an X I C symbol labeled as Container_Counter_Max.D N is connected to the parallel connection of the R E S instruction labeled as Container_Counter_Counts and R E S instruction labeled as Container_Counter_Max, in series along the second rung. From the left, an X I C symbol labeled as Container_Counter_Counts.D N, X I C symbol labeled as Container_Sensor and <Local:1:I.Data.2>, and an O T E labeled as SOL and <Local:2:O.Data.2>. A rung is connected at the right side of the line L 2 with a label output over it. A solenoid labeled as SOL is connected along the rung connected to the line L 2. The illustration at the bottom shows four containers on a conveyor belt with a movable gate on one side of third container. A sensor is also placed over the third container.

Return to Figure 15-74

OceanofPDF.com

Figure 15-75 Extended Description (Chapter 15)

Return to Figure 15-75

From the top, the first column is labeled as tag name, (a negative sign enclosed in a square) Part_Counter_Counts, (a positive sign enclosed in a square and indented to level 1) Part_Counter_Counts.P R E, (a positive sign enclosed in a square and indented to level 1) Part_Counter_Counts.A C C, (indented to level 1) Part_Counter_Counts.C U, (indented to level 1) Part_Counter_Counts.C D, (indented to level 1) Part_Counter_Counts.D N, (indented to level 1) Part_Counter_Counts.O V, (indented to level 1) Part_Counter_Counts.U N, (a negative sign enclosed in a square) Part_Counter_Max, (a positive sign enclosed in a square and indented to level 1) Part_Counter_Max.P R E, (a positive sign enclosed in a square and indented to level 1) Part_Counter_Max.A C C, (indented to level 1) Part_Counter_Max.C U, (indented to level 1) Part_Counter_Max.C D, (indented to level 1) Part_Counter_Max.D N, (indented to level 1) Part_Counter_Max.O V, (indented to level 1) Part_Counter_Max.U N, Container_Sensor, and SOL. From the top, the second column is labeled as value, {...}, 6, 0, 0, 0, 0, 0, 0, {...}, 11, 0, 0, 0, 0, 0, 0, 0, and 0. For the third column, the first row is labeled as style, the second and the tenth rows are unlabeled, and the remaining rows are labeled as decimal. From the top, the fourth column is labeled as data type, COUNTER, DINT, DINT, BOOL, BOOL, BOOL, BOOL, BOOL, COUNTER, DINT, DINT, BOOL, BOOL, BOOL, BOOL, BOOL, BOOL, and BOOL.

Return to Figure 15-75

OceanofPDF.com

Figure 15-76 Extended Description (Chapter 15)

Return to Figure 15-76

There are four vertical lines. From the left, the first vertical line is labeled L 1 and the fourth vertical line is labeled L 2. Timer switch as input is connected to L 1. Four rungs are connected between the second and the third vertical lines with a label program placed over the first rung. The first rung has timer switch with address Local:3:I.Data.0, base timer D N, a time on delay T O N, and E N, and connected in series. The right side of the T O N is connected to D N. From the top, the T O N is labeled as T O N, time on delay, timer with base_timer, present with 10000, and accumulated with 0. The second rung has base timer D N, C T U and C U, and connected in series. D N is connected to the right side of the C T U box. From top to left, C T U has count up, counter with drive counter, present with 5, and accumulated with 0. The third rung has timer switch with address Local:3:I.Data.0 and drive counter R E S, and connected in series. The fourth rung has drive counter D N and output pilot light P L with address Local:2:O.Data.0, and connected in series. On the right, output P L light is connected to L 2.

Return to Figure 15-76

OceanofPDF.com

Figure 15-77 Extended Description (Chapter 15)

Return to Figure 15-77

The right side of the C T D is also connected to an output energize D N. R E S instruction labeled as Counter_1 is connected to the same vertical line along the second rung. Three dots arranged horizontally are placed at the left side of the C T D instruction and the R E S instruction. A label input side of rung is placed between the dots at the top and the dots at the bottom. From the top, C T D is labeled as C T D, count down, counter Counter_1, preset, and accum. The label Counter_1 in the C T D instruction points at the label Counter_1 placed above the R E S. A table labeled as tag name consists of a column of eight rows. Arrows from a label tag name points at the label counter_1 in the C T D and the label tag name in the table. From the top, the table is labeled as (negative sign enclosed in a square) Counter_1, (a positive sign enclosed in a square and indented to level 1) Counter_1.P R E, (a positive sign enclosed in a square and indented to level 1) Counter_1.A C C, (indented to level 1) Counter_1.C U (highlighted in blue), (indented to level 1) Counter_1.C D (highlighted in blue), (indented to level 1) Counter_1.D N, (indented to level 1) Counter_1.O V, and (indented to level 1) Counter_1.U N.

Return to Figure 15-77

OceanofPDF.com

Figure 15-78 Extended Description (Chapter 15)

Return to Figure 15-78

There are four vertical lines. From the left, the first vertical line is labeled as L 1 and the fourth vertical line is labeled as L 2. Three rungs are connected at right side of the line L 1 with a label inputs placed over the first rung. From the top, N O pushbutton labeled as Restart_Button, N O limit switch labeled as Enter_Limit_Sw, and N O limit switch labeled as Exit_Limit_Sw are connected along the rungs connected to the line L 1. Four rungs are connected between the second and the third vertical lines with a label ladder logic program placed over the first rung. From the left, X I C symbol labeled as Restart_Button and <Local:1:I.Data.1>, and R ES labeled as Counter_1 are connected in series along the first rung. From the left, X I C symbol labeled as Enter_Limit_Sw and <Local:1:I.Data.3>, C T U instruction, and an output energize C U are connected in series along the second rung. The right side of the C T U is connected to an output energize D N. From the top, the C T U is labeled as C T U, count up, counter Counter_1, preset 50 (an arrow pointing at it from the right side), and accum zero (an arrow pointing at it from the right side). From the left, an X I C symbol labeled as Exit_Limit_Sw and <Local:1:I.Data.4>, C T D instruction, and an output energize C D are connected in series along the third rung. The right side of the C T D is connected to an output energize D N. From the top, the C T D is labeled as C T D, count down, counter Counter_1, preset 50 (an arrow pointing at it from the right side), and accum zero (an arrow pointing at it from the right side). From the left, an X I O symbol (highlighted in green) labeled as Counter_1.D N is connected to O T E (highlighted in green) labeled as Conveyor_Contactor and <Local:2:O.Data.2> in series along the fourth rung. A rung is connected at the left side of the line L 2 with a label output over it. A relay coil C labeled as Conveyor_Contactor is connected along the rung connected to the line L 2.

Return to Figure 15-78

OceanofPDF.com

Figure 15-79 Extended Description (Chapter 15)

Return to Figure 15-79

From the top, the first column is labeled as tag name, (a negative sign enclosed in a square) Counter_1, (a positive sign enclosed in a square and indented to level 1) Counter_1.P R E, (a positive sign enclosed in a square and indented to level 1) Counter_1.A C C, (indented to level 1) Counter_1.C U, (indented to level 1) Counter_1.C D, (indented to level 1) Counter_1.D N, (indented to level 1) Counter_1.O V, (indented to level 1) Counter_1.U N, Restart_Button, Enter_Limit_Sw, Exit_Limit_Sw, and Conveyor_Contactor. From the top, the second column is labeled as value, {...}, 50, 0, 0, 0, 0, 0, 0, 0, 0, 0, and 1. For the third column, the first row is labeled as style, the second row is unlabeled, and the remaining rows are labeled as decimal. From top to bottom, the fourth column is labeled as data type, COUNTER, DINT, DINT, BOOL, BOOL, BOOL, BOOL, BOOL, BOOL, BOOL, BOOL, and BOOL.

Return to Figure 15-79

OceanofPDF.com

Figure 15-80 Extended Description (Chapter 15)

Return to Figure 15-80

There are three vertical lines. From the left, the first vertical line is labeled as L 1. Two rungs are connected at right side of the line L 1 with a label inputs placed over the first rung. From the top, N O selector switch labeled as Start_Switch, and N O proximity switch labeled as Bottle_Sensor are connected along the rungs connected to the line L 1. Three rungs are connected between the second and the third vertical lines with a label ladder logic program placed over the first rung. From the left, X I C symbol labeled as Start_Switch and <Local:1:I.Data.0>, T O N instruction, and an output energize E N are connected in series along the first rung. The right side of the T O N is connected to an output energize D N. From the top, T O N is labeled as T O N, timer on delay, timer 1_Min_Timer, preset 60,000, and accum zero. From the left, X I C symbol labeled as Start_Switch and <Local:1:I.Data.0>, X I O symbol labeled as 1_Min_Timer.D N, X I C symbol labeled as Bottle_Sensor and <Local:1:I.Data.1>, C T U instruction, and output energize C U are connected in series along the second rung. The right side of the C T U is connected to an output energize D N. From the top, C T U is labeled as C T U, count up, counter Total_Bottles, preset zero, and accum zero. From the left, X I O symbol labeled as Start_Switch and <Local:1:I.Data.0> is connected in series with R E S instruction labeled as Total_Bottles. The illustration at the bottom shows five bottles on a conveyor belt with a sensor on one side.

Return to Figure 15-80

OceanofPDF.com

Figure 15-81 Extended Description (Chapter 15)

Return to Figure 15-81

Three symbols and seven labels enclosed in rectangles are placed relatively on the left side between the arrowheads. From left to right, the symbols are, an upper-case letter I rotated at ninety degrees, same symbol as the first but includes a rectangle at the bottom of the middle line of the letter I, and same symbol as the second but includes a horizontal line dividing the rectangle into half. From left to right, the labels highlighted in blue are C P T, A D D, S U B, M U L, D I V, M O D, and S Q R. Six labels are shown at the bottom. From left to right, the labels are arrow-head facing, arrow-head facing right, bit, compare, compute or math (highlighted in blue), move or logical, and file or misc.

Return to Figure 15-81

OceanofPDF.com

Figure 15-82 Extended Description (Chapter 15)

Return to Figure 15-82

There are three vertical lines. From the left, the first vertical line is labeled as L 1. A rung is connected at right side of the line L 1 with a label input placed over it. An N C switch labeled as Add_Sw is connected along the rung. A rung is connected between the second and the third vertical lines with a label ladder logic program over it. From the left, X I C symbol (highlighted in green) labeled as Add_Sw and <Local:1:I.Data.1> is connected to A D D instruction in series along the rung. From the top, A D D instruction is labeled as A D D, add value_A, source A 25 (an arrow pointing at it from the right side), value_B (on the right side), source B 50 (an arrow pointing at it from the right side), Total_Value (on the right side), and dest 75 (an arrow pointing at it from the right side). From the top, the first column is labeled as tag name, (a positive sign enclosed in a square) Total_Value, (a positive sign enclosed in a square) Value_A, (a positive sign enclosed in a square) Value_B, and Add_Sw. From the top, the second column is labeled as value, 75, 25, 50, and one. From the top, the third column is labeled as style, decimal, decimal, decimal, and decimal. From the top, the fourth column is labeled as data type, DINT, DINT, DINT, and BOOL.

Return to Figure 15-82

OceanofPDF.com

Figure 15-83 Extended Description (Chapter 15)

Return to Figure 15-83

There are three vertical lines. From the left, the first vertical line is labeled as L 1. A rung is connected at right side of the line L 1 with a label input placed over it. An N C switch labeled as Sub_Sw is connected along the rung. A rung is connected between the second and the third vertical lines with a label ladder logic program over it. From the left, X I C symbol (highlighted in green) labeled as Sub_Sw and <Local:1:I.Data.4> is connected to S U B instruction in series along the rung. From the top, S U B instruction is labeled as S U B, subtract, source A Parts_Stock (an arrow pointing at it from the right side), 900 (on the right side), source B Shipped_Parts (an arrow pointing at it from the right side), 200 (on the right side), dest Current_Inventory (an arrow pointing at it from the right side), and 700 (on the right side). An X I C symbol labeled Calculate is connected across the first X I C symbol. From the top, the first column is labeled as tag name, (a positive sign enclosed in a square) Parts_Stock, (a positive sign enclosed in a square) Shipped_Parts, (a positive sign enclosed in a square) Current_Inventory, Sub_Sw, and calculate. From the top, the second column is labeled as value, 900, 200, 700, one, and zero. From the top, the third column is labeled as style, decimal, decimal, decimal, decimal, and decimal. From the top, the fourth column is labeled as data type, DINT, DINT, DINT, BOOL, and BOOL.

Return to Figure 15-83

OceanofPDF.com

Figure 15-84 Extended Description (Chapter 15)

Return to Figure 15-84

There are three vertical lines. From the left, the first vertical line is labeled as L 1. Two rungs are connected at right side of the line L 1 with a label inputs placed over the first rung. From the top, N C switches labeled as Sw_1 and Sw_2 are connected along the rungs. A rung is connected between the second and the third vertical lines with a label ladder logic program over it. From the left, X I C symbol (highlighted in green) labeled as Sw_1 and <Local:1:I.Data.1>, X I C symbol (highlighted in green) labeled as Sw_2 and <Local:1:I.Data.2>, and M U L instruction are connected in series along the rung. From the top, M U L instruction is labeled as M U L, multiply, source A Cases_Produced (an arrow pointing at it from the right side), 60 (on the right side), source B Bottles_Per_Case (an arrow pointing at it from the right side), 12 (on the right side), dest Bottles_Produced, and 720 (an arrow pointing at it from the right side and it is placed on the right side). From the top, the first column is labeled as tag name, Sw_1, Sw_2. (a positive sign enclosed in a square) Cases_Produced, (a positive sign enclosed in a square) Bottles_Per_Case, and (a positive sign enclosed in a square) Bottles_Produced. From the top, the second column is labeled as value, 1, 1, 60, 12, and 720. From the top, the third column is labeled as style, decimal, decimal, decimal, decimal, and decimal. From the top, the fourth column is labeled as data type, BOOL, BOOL, DINT, DINT, and DINT.

Return to Figure 15-84

OceanofPDF.com

Figure 15-85 Extended Description (Chapter 15)

Return to Figure 15-85

From the left, X I C symbol (highlighted in green) and D I V instruction are connected in series along a rung. From top to bottom, the D I V instruction is labeled as D I V, divide, source A 5, source B3,dest answer_real, and 1.6666666 (placed on the right and an arrow is pointing at 1.6666666 from the right side). From left to right, the first row of the table is labeled as tag name, value, style, and data type. From left to right, the second row of the table is labeled as calculate, one, decimal, and BOOL. From left to right, the third row of the table is labeled as answer_real, 1.6666666, float, and real.

Return to Figure 15-85

OceanofPDF.com

Figure 15-86 Extended Description (Chapter 15)

Return to Figure 15-86

There are three vertical lines. From the left, the first vertical line is labeled as L 1. Three rungs are connected at right side of the line L 1 with a label inputs placed over the first rung. From the top, N O proximity switch labeled as Conveyor_1_Sensor, N O proximity switch labeled as Conveyor_2_Sensor, and N O pushbutton labeled as Manual_Conveyor_Reset are connected along the rungs. Four rungs are connected between the second and the third vertical lines with a label ladder logic program over it. From the left, X I C symbol labeled as Conveyor_1_Sensor and <Local:1:I.Data.4>, first C T U instruction, and an output energize C U are connected in series along the first rung. The right side of the first C T U is connected to an output energize D N. From the top, the first C T U instruction is labeled as C T U, count up, counter Counter_1_Parts, preset 250, and accum 30. From the left, X I C symbol labeled as Conveyor_2_Sensor and <Local:1:I.Data.5>, second C T U instruction, and an output energize C U are connected in series along the first rung. The right side of the second C T U is connected to an output energize D N. From the top, the second C T U instruction is labeled as C T U, count up, counter Counter_2_Parts, preset 250, and accum 70. A D D instruction is connected along the third rung. From the top, A D D is labeled as A D D, add, source A Conveyor_1_Parts.A C C, 30 (right), source B Conveyor_2_Parts.A C C, 70 (right), dest Conveyor_3_Parts, and 100 (right). From the left, X I C symbol labeled as Conveyor_1_Parts.D N, R E S labeled as Conveyor_1_Parts, and R E S labeled as Conveyor_2_Parts are connected in series along the fourth rung. The parallel connection of the X I C symbols labeled as Conveyor_2_Parts.D N and Manual_Conveyor_Reset <Local:1:I.Data.2> is connected across the X I C symbol labeled as Conveyor_1_Parts.D N. From the top, the table is labeled as Conveyor_1_Parts, Conveyor_1_Parts.P R E, Conveyor_1_Parts.A C C, Conveyor_1_Parts.C U, Conveyor_1_Parts.D N, Conveyor_2_Parts, Conveyor_2_Parts.P R E, Conveyor_2_Parts.A C C, Conveyor_2_Parts.C U, Conveyor_2_Parts.D N, Conveyor_1_Sensor, Conveyor_2_Sensor, Conveyor_3_Parts, and Manual_Conveyor_Reset. From the top, the second column is labeled as value, {...}, 250, 30, 0, 0, {...}, 250, 70, 0, 0, 0, 0, 100, and 0. For the third column, the first row is labeled as style, the second and the seventh rows are unlabeled and the remaining rows are labeled as decimal. From the top, the fourth column is labeled as data type, COUNTER, DINT, DINT, BOOL, BOOL, COUNTER, DINT, DINT, BOOL, BOOL, BOOL, BOOL, DINT, and BOOL.

Return to Figure 15-86

OceanofPDF.com

Figure 15-87 Extended Description (Chapter 15)

Return to Figure 15-87

Three symbols and seven labels enclosed in rectangles are placed relatively on the left side between the arrowheads. From left to right, the symbols are, an upper-case letter I rotated at ninety degrees, same symbol as the first but includes a rectangle at the bottom of the middle line of the letter I, and same symbol as the second but includes a horizontal line dividing the rectangle into half. From left to right, the labels highlighted in blue are C M P, L I M, M E Q, E Q U, N E Q, L E S, and G R T. Six labels are shown at the bottom. From left to right, the labels are arrow-head facing, arrow-head facing right, bit, timer or counter, input or output, and compare (highlighted).

Return to Figure 15-87

OceanofPDF.com

Figure 15-88 Extended Description (Chapter 15)

Return to Figure 15-88

The ladder diagram consists of three vertical lines. From the left, the third vertical line is labeled as L 2. From the left, E Q U instruction and the O T E instruction (highlighted in green) labeled as equal_P L <Local:2:O.Data.2> are connected in series along a rung. From the top, the E Q U instruction is labeled as E Q U, equal, source A value_1, 25 (at the right side and an arrow points at 25 from right side), source B value_2, and 25 (at the right side and an arrow points at 25 from right side). A label ladder logic program is placed over the rung. The red light labeled as equal_P L is connected along a rung connected at the left side of the line L 2. A label output is placed over the light. From the left, the first row of the table is labeled as tag name, value, style, and data type. From the left, the second row is labeled as equal_P L, one, decimal, and BOOL. From the left, the third row is labeled as (a positive sign) value_1, 25, decimal, and DINT. From the left, the fourth row is labeled as (a positive sign) value_2, 25, decimal, and DINT.

Return to Figure 15-88

OceanofPDF.com

Figure 15-89 Extended Description (Chapter 15)

Return to Figure 15-89

From the left, the third vertical line is labeled as L 2. From the left, N E Q instruction and the O T E instruction (highlighted in green) labeled as not_equal_P L <Local:2:O.Data.3> are connected in series along a rung. From the top, the N E Q instruction is labeled as N E Q, not equal, source A value_1, 10 (at the right side and an arrow points at 10 from right side), source B value_2, and 25 (at the right side and an arrow points at 25 from right side). A label ladder logic program is placed over the rung. The green light labeled as not_equal_P L is connected along a rung connected at the left side of the line L 2. A label output is placed over the light.

Return to Figure 15-89

OceanofPDF.com

Figure 15-90 Extended Description (Chapter 15)

Return to Figure 15-90

From the left, the third vertical line is labeled as L 2. From the left, L E S instruction and the O T E instruction (highlighted in green) labeled as less_than_P L <Local:2:O.Data.4> are connected in series along a rung. From the top, the L E S instruction is labeled as L E S, less than (A less than B), source A value_1, 100 (at the right side and an arrow points at 100 from right side), source B value_2, and 300 (at the right side and an arrow points at 300 from right side). A label ladder logic program is placed over the rung. The yellow light labeled as less_than_P L is connected along a rung connected at the left side of the line L 2. A label output is placed over the light.

Return to Figure 15-90

OceanofPDF.com

Figure 15-91 Extended Description (Chapter 15)

Return to Figure 15-91

From the left, the third vertical line is labeled as L 2. From the left, G R T instruction and the O T E instruction (highlighted in green) labeled as greater_than_P L <Local:2:O.Data.5> are connected in series along a rung. From the top, the G R T instruction is labeled as G R T, greater than (A greater than B), source A value_1, 420 (at the right side and an arrow points at 1420 from right side), source B value_2, and 1200 (at the right side and an arrow points at 1200 from right side). A label ladder logic program is placed over the rung. The blue light labeled as greater_than_P L is connected along a rung connected at the left side of the line L 2. A label output is placed over the light.

Return to Figure 15-91

OceanofPDF.com

Figure 15-92 Extended Description (Chapter 15)

Return to Figure 15-92

The ladder diagram consists of three vertical lines. From the left, the third vertical line is labeled as L 2. From the left, C M P instruction and the O T E instruction (highlighted in green) labeled as equal_P L <Local:2:O.Data.2> are connected in series along a rung. From the top, the C M P instruction is labeled as C M P, compare, and expression value_1 = value_2. A label ladder logic program is placed over the rung. The red light labeled as equal_P L is connected along a rung connected at the left side of the line L 2. A label output is placed over the light. From the left, the first row of the table is labeled as tag name, value, style, and data type. From the left, the second row is labeled as equal_P L, one, decimal, and BOOL. From the left, the third row is labeled as (a positive sign) value_1, 300, decimal, and DINT. From the left, the fourth row is labeled as (a positive sign) value_2, 300, decimal, and DINT.

Return to Figure 15-92

OceanofPDF.com

Figure 15-93 Extended Description (Chapter 15)

Return to Figure 15-93

There are four vertical lines. From the left, the first vertical line is labeled as L 1 and the fourth vertical line is labeled as L 2. From the top, N O pushbuttons labeled as Count_P B and Reset_P B are connected along two rungs connected at the right side of the line L 1 with a label inputs over the first rung. Five rungs are connected between the second and the third vertical lines with a label ladder logic program is placed over the first rung. From the left X I C symbol labeled as Count_PB <Local:1:I.Data.1>, C T U instruction and output energize E N are connected in series along the first rung. From the top, C T U is labeled as C T U, count up, counter C 1, preset 25, and accum zero. From the left, G R T, L E S, and an O T E labeled as PL_1 <Local:2:O.Data.1> are connected in series along the second rung. From the top, G R T is labeled as greater than (A greater than B), source A C1.A C C, zero, and source B 5. L E S is labeled as L E S, less than (A less than B), source A C1.A C C, zero, and source B 10. From the left E Q U and an O T E labeled as PL_2 <Local:2:O.Data.2> are connected in series along the third rung. E Q U is labeled as E Q U, equal, source A C1.A C C, zero, and source B 15. From the left, N E Q, and an O T E labeled as PL_3 <Local:2:O.Data.3> are connected in series along the fourth rung. N E Q is labeled as N E Q, not equal, source A C1.A C C, zero, and source B 20. From the left, X I C labeled as C1.D N and R E S labeled as C 1 are connected in series along the fifth rung. An X I C labeled as Reset_PB <Local:1:I.Data.2> is connected across the X I C labeled as C1.D N. From the top, lights labeled as P L_1, P L_2, and P L_3 are connected along three rungs connected at the left side of the line L 2 with a label outputs over the first rung.

Return to Figure 15-93

OceanofPDF.com

Figure 15-94 Extended Description (Chapter 15)

Return to Figure 15-94

Three symbols and seven labels enclosed in rectangles are placed relatively on the left side between the arrowheads. From left to right, the symbols are, an upper-case letter I rotated at ninety degrees, same symbol as the first but includes a rectangle at the bottom of the middle line of the letter I, and same symbol as the second but includes a horizontal line dividing the rectangle into half. From left to right, the labels are M O V (highlighted in blue), M V M (highlighted in blue), A N D, O R, X O R, S W P B, N O T, C L R, and B T D. Three labels are shown at the bottom. From left to right, the labels are arrow-head facing, arrow-head facing right, and move or logical (highlighted and is placed after three blank regions). The M O V instruction is placed below the toolbar. From top to bottom, the M O V instruction is labeled as M O V, move, source, and dest.

Return to Figure 15-94

OceanofPDF.com

Figure 15-95 Extended Description (Chapter 15)

Return to Figure 15-95

Three rungs are connected at the right side of the line L 1. From the top, N O pushbutton labeled as P B_10 S, N O pushbutton labeled as P B_15 S, and N O switch labeled as timer_start are connected along the three rungs. A label inputs is placed above the first rung. There are five rungs between the second and the third vertical lines. From the left, an X I C symbol labeled as P B_10 s <Local:1:I.Data.1> and M O V instruction are connected in series along the first rung. From the top, the first M O V instruction is labeled as M O V, move, source 10,000, dest T 1.P R E, and 10,000 (placed on the right side and an arrow points at 10,000 from the right side). From the left, an X I C symbol labeled as P B_15 s <Local:1:I.Data.2> and the second M O V instruction are connected in series along the second rung. From the top, the second M O V instruction is labeled as M O V, move, source 15,000, dest T 1.P R E, and 10,000 (placed on the right side and an arrow points at 10,000 from the right side). From the left, X I C symbol labeled as timer_start <Local:1:I.Data.3>, T O N instruction, and an output energize E N are connected in series along the third rung. The right side of the T O N is connected to an output energize D N. From the top, T O N instruction is labeled as T O N, timer on delay, timer T1, preset 10,000 (an arrow pointing at 10,000 from the right side), and accum zero (an arrow pointing at zero from the right side). From the left, an X I C symbol labeled as T 1.T T and an output energize labeled as P L_1 <Local:2:O.Data.1> are connected in series along the fourth rung. From the left, X I C symbol labeled as T 1.D N and an output energize labeled as P L_2 <Local:2:O.Data.2> are connected in series along the fifth rung. A label ladder logic program is placed over the first rung. Two rungs are connected at the left side of the line L 2. From the top, light labeled as P L_1 and light labeled P L_2 are connected along the rungs connected to the line L 2. A label outputs is placed above the first rung connected to the line L 2. An illustration of N O switch is placed below its symbol.

Return to Figure 15-95

OceanofPDF.com

Figure 15-96 Extended Description (Chapter 15)

Return to Figure 15-96

There are four vertical lines. From the left, the first vertical line is labeled as L 1 and the fourth vertical line is labeled as L 2. From the top, N O switch labeled as P L_Equal_Sw and the thumbwheel switch labeled as B C D_T W S (all 16 bits) are connected along the two rungs connected to the line L 1. A label inputs is placed over the first rung. Two rungs are connected between the second and the third vertical lines with a label ladder logic program is placed over the first rung. M O V instruction is connected along the first rung. From the top, the M O V instruction is labeled as M O V, move, source BCD_TWS, 2#0000_0000_0000_0000_0000_0000_0010_0000, dest BCD_Display, and 2#0000_0000_0000_0000_0000_0000_0010_0000. From the left, X I C symbol labeled as PL_Equal_Sw <Local:1:I.Data.0>, E Q U, and O T E labeled as Equal_PL <Local:2:O.Data.0> are connected in series along the second rung. From the top, E Q U is labeled as E Q U, equal, source A BCD_TWS, 2#0000_0000_0000_0000_0000_0000_0010_0000, source B source_B, and 2#0000_0000_0000_0000_0000_0000_0010_0000. From the top, light labeled as Equal_PL, and L E D display labeled as BCD_Display (all 16 bits) are connected along the two rungs at the left side of the line L 2. A label outputs is placed over the first rung. 20 Decimal = 100,000 B C D.

Return to Figure 15-96

OceanofPDF.com

Figure 15-97 Extended Description (Chapter 15)

Return to Figure 15-97

There are four vertical lines. From the left, the first vertical line is labeled as L 1 and the fourth vertical line is labeled as L 2. From the top, N C pushbutton labeled as conveyor_stop, N O pushbutton labeled as conveyor_start, and a load cell labeled as conveyor_load_cell are connected along the three rungs are connected to the line L 1. A label inputs is placed over the first rung. Four rungs are connected between the second and the third vertical lines with a label ladder logic program over the first rung. From the top, X I C symbol (highlighted in green) labeled as Conveyor_Stop <Local:1:I.Data.0>, X I C symbol labeled as Conveyor_Start<Local:1:I.Data.1>, and an O T E labeled as Internal_Relay are connected in series along the first rung. An X I C symbol labeled as Internal_Relay is connected across the X I C symbol labeled as Conveyor_Start<Local:1:I.Data.1>. From left, X I C symbol labeled as Internal_Relay, E Q U, and an O T E labeled as Weight_Normal <Local:2:0.Data.0> are connected in series along the second rung. From top, the E Q U is labeled as E Q U, equal, source A Conveyor_Load_Cell, 50.0, and source B 50. From the left, X I C symbol labeled as Internal_Relay, L E S, and an O T E labeled as Weight_Below_Normal <Local:2:0.Data.1> are connected in series along the third rung. From the top, L E S is labeled as L E S, less than (A less than B), source A Conveyor_Load_Cell, 50.0,and source B 50. From the left, X I C symbol labeled as Internal_Relay, G R T, and an O T E labeled as Weight_Above_Normal <Local:2:0.Data.2> are connected in series along the fourth rung. From the top, G R T is labeled as G R T, greater than (A greater than B), source A Conveyor_Load_Cell, 50.0,and source B 50. From the top, yellow light labeled as Weight_Normal, green light labeled as Weight_Below_Normal, and red light labeled as Weight_Above_Normal are connected along the three rungs connected to the right side of the line L 2. A label outputs is placed over the first rung. An illustration shows a block on a conveyor belt.

Return to Figure 15-97

OceanofPDF.com

Figure 15-98 Extended Description (Chapter 15)

Return to Figure 15-98

There are four vertical lines. From the left, the first vertical line is labeled as L 1 and the fourth vertical line is labeled as L 2. From the top, N O limit switch is labeled as Main_Conveyor_LS, N O limit switch labeled as Reject_Conveyor_LS, and N O pushbutton labeled as Reset_PB are connected along the three rungs connected to the line L 1. A label inputs is placed over the first rung. Five rungs are connected between the second and the third vertical lines with a label ladder logic program placed over the first rung. From left, X I C symbol labeled as Main_Conveyor_LS <Local:1:I.Data.0>, C T U, and an output energize C U are connected in series along the first rung. The right side of the first C T U is connected to an output energize D N. From the top, the first C T U is labeled as C T U, count up, counter Main_Conveyor_Counter, preset 5,000, and accum zero. From the left, X I C symbol labeled as Reject_Conveyor_LS <Local:1:I.Data.1>, the second C T U, and an output energize CU are connected in series along the second rung. The right side of the second C T U is connected to an output energize D N. From the top, the second C T U is labeled as C T U, count up, counter Reject_Conveyor_Counter, preset 5,000, and accum zero. From left, X I C symbol labeled as Main_Conveyor_LS <Local:1:I.Data.0>, and S U B instruction are connected in series along the third rung. An X I C symbol labeled as Reject_Conveyor_LS <Local:1:I.Data.1> is connected across the X I C symbol labeled as Main_Conveyor_LS <Local:1:I.Data.0>. From the top, S U B is labeled as S U B, subtract, source A Main_Conveyor_Counter.A C C, zero, source B Reject_Conveyor_Counter.A C C, zero, dest Data_Format_Conversion, and zero. T O D instruction labeled as T O D, to B C D, source Data_Format_Conversion, zero, dest B C D_output_Display, and zero is connected along the fourth rung. From the left, X I C symbol labeled as Reset_PB <Local:1:I.Data.2>, R E S labeled as Main_Conveyor_Counter, R E S labeled as Reject_Conveyor_Counter, and C L R instruction are connected in series along the fifth rung. From top, the C L R is labeled as C L R, clear, dest Data_Format_Conversion, and zero. An L E D display is connected along a rung that is connected to the vertical line L 2 with a label output over it. An illustration of the switches, pushbutton, and L E D display is shown below their symbols. An illustration at the bottom of the ladder diagram shows a reject conveyor joined in the middle of main conveyor. A diverter gate is placed at their junction. A reset button is shown below the illustration. A limit switch labeled as Main conveyor L S is placed at the beginning of the main conveyor. A limit switch labeled as Reject conveyor L S is placed at the beginning of the reject conveyor.

Return to Figure 15-98

OceanofPDF.com

Figure 15-99 Extended Description (Chapter 15)

Return to Figure 15-99

There are four vertical lines. From the left, the first vertical line is labeled L 1 and the fourth vertical line is labeled L 2. From the top, a switch as input is connected to line L 1. Four rungs are connected between the second and the third vertical lines with a label program placed over the first rung. From left, a switch, a T O N, and an E N are connected in series along the first rung. The right side of the T O N is connected to D N. From the top, T O N is labeled as C T U, time on delay, timer with time labeled on the right, present with 10000 labeled on the right, and accumulated with 0 labeled on the right is depicted. The second rung consists of only a subtract box. From top to bottom, S U B, subtract, source A with Time.PRE 10000 labeled on the right, source B with Time.ACC 0 labeled on the right, and destination with different 10000 labeled on the right, are depicted. The third rung consists of only a D I V box. From top to bottom, D I V, divide, source A with difference 10000 labeled on the right, source B with 1000 labeled on the right, and destination with quotient 10 labeled on the right, are depicted. The fourth rung consists of only an F R D box. From top to bottom, F R D, from B C D, source with quotient 10 labeled on the right, and destination with B C D_Display 0 labeled on the right, are depicted. On the right, a B C D display with three boxes in a column connected in series is linked to line L 2. B C D display with three cells, each containing a 0 is depicted below it.

Return to Figure 15-99

OceanofPDF.com

Figure 15-100 Extended Description (Chapter 15)

Return to Figure 15-100

There are four vertical lines. From the left, the first vertical line is labeled L 1 and the fourth vertical line is labeled L 2. From the top, a temperature transmitter as input is connected to line L 1. A sensor display is depicted below it with reading 0 0 0. 0 V. Five rungs are connected between the second and the third vertical lines with a label program placed over the first rung. The first rung consists of only a box M U L. From top to bottom, M U L, multiply, source A with transmitter_Volts 0 depicted on the right, source B with 10 labeled on the right, and destination with Simulated_Temperature 0.0 are labeled. The second rung consists of G R T box and light L connected in series. From top to bottom of the G R T box, G R T, greater than (A>B), source A with Simulated_Temperaure 0 labeled on the right, source B with 29 labeled on the right, are depicted. The third rung consists of L E S box and F H fan, U connected in series. From top to bottom of the L E S box, L E S, less than (A<B), source A with Simulated_Temperaure 0 labeled on the right, source B with 13 labeled on the right, are depicted. The fourth rung consists of G R T box and F H H fan, L connected in series. From top to bottom of the G R T box, G R T, greater than (A>B), source A with Simulated_Temperaure 0 labeled on the right, source B with 44 labeled on the right, are depicted. The fifth rung consists of L E S box and F H H fan, U connected in series. From top to bottom of the L E S box, L E S, less than (A<B), source A with Simulated_Temperaure 0 labeled on the right, source B with 36 labeled on the right, are depicted. On the right, fan F H and fan F H H, each with an O L light in series, are connected to line L 2.

On the bottom left, a schematic of a P L C operated two fan cooling tower is depicted. A temperature transmitter is situated on the bottom right of the tank, while an outlet for cold water is shown on the bottom left. Additionally, a horizontal pipe on the top of the tank is labeled as hot water in. The hot water from this pipe is dispersed onto a filter shown below. Above the hot water pipe, two outlet fans labeled F H and F H H are designated as air outlets.

Return to Figure 15-100

OceanofPDF.com

Figure 15-101 Extended Description (Chapter 15)

Return to Figure 15-101

On the extreme left, a column labeled source is depicted. On its right, slot 1 lists numbers from 00 through 15 vertically. Numbers 00, 04, 05, 06, 07, 10, 12, and 15 are highlighted in red, and the rest of the numbers in gray. On the left of slot 1 column, closed keys and open keys are shown beside the red and grey shaded numbers. On the extreme right, two columns jointly labeled destination is depicted. The first column is labeled before. On its right, slot 2 lists numbers from 00 through 15 vertically. Each number is joined by a dark indicator on its left. The second column is labeled after. On its right, slot 2 lists numbers from 00 through 15 vertically. Each number is joined by an indicator on its left. Numbers 00, 04, 05, 06, 07, 10, 12, and 15 have red indicators, while the rest of the numbers have dark indicators on their left. In the middle, there are two independent rungs. The first rung is labeled before enabled and consists of an input switch with address Local:3:I:Data.0 and a M O V box. From the top, this box lists M O V, move, source with Local:1:I.Data 1001_0100_1111_0001 labeled on its right, destination with Local:2:O.Data 0000_0000_0000_0000 labeled on its right. The second rung is labeled after enabled and consists of an input switch with address Local:3:I:Data.0 and a M O V box. From the top, this box lists M O V, move, source with Local:1:I.Data 1001_0100_1111_0001 labeled on its right, destination with Local:2:O.Data 1001_0100_1111_0001 labeled on its right. At the bottom of the second rung, two rows labeled Local:1:I.Data and Local:2:O.Data are depicted one over below. Both the rows have 16 cells. Each row has entries: 1, 0, 0, 1, 0, 1, 0, 0, 1, 1, 1, 1, 0, 0, 0, 1. Both the rows are mainly divided into four sections, where the first section contains numbers 15 and 12 over first and fourth cells, numbers 11 and 8 over fifth and eights cells, numbers 7 and 4 over ninth and twelfths cells, and numbers 3 and 0 over thirteenth and sixteenth cells. A downward arrow from the top row to bottom row for each cell is displayed.

Return to Figure 15-101

OceanofPDF.com

Figure 15-102 Extended Description (Chapter 15)

Return to Figure 15-102

On the extreme left, a column labeled source is depicted with slot 1. It lists the numbers 1, 3, 4, 7, 8, 11, 12, and 15 on the left. Next to numbers, 15 keys are arranged in columns. On its right, 15 indicator lights are shown. The first, fourth, fifth, sixth, seventh, tenth, twelfth, and fifteenth indicator lights are shown in red, and the rest are shown in grey. On the extreme right, a column labeled destination with slot 2. It lists the numbers 1, 3, 4, 7, 8, 11, 12, and 15 on the right,15 indicator lights are shown. Fourth, fifth, twelfth, and fifteenth lights are shown in red, and the rest are shown in grey. In the middle, there is one independent rung that consists of an input switch and a M V W box. From the top, this box lists M V M, masked move, source with Slot_1_Discrete_Input 2#0000_0000_0000_0000_1001_0100_1111_0001 labeled on its right, mask with 16#F034 labeled on its right, and destination with Slot_2_Discrete_Output 2#0000_0000_0000_0000_1001_0000_0011_0000 labeled on the right. At the bottom of the rung, three rows labeled Local:1:I.Data, 16#F034 Mask, and Local:2:O.Data are depicted one over below. All the three rows have 16 cells. Each row has entries: 1, 0, 0, 1, 0, 1, 0, 0, 1, 1, 1, 1, 0, 0, 0, 1. The first row is mainly divided into four sections, where the first section contains numbers 15 and 12 over first and fourth cells, numbers 11 and 8 over fifth and eights cells, numbers 7 and 4 over ninth and twelfths cells, and numbers 3 and 0 over thirteenth and sixteenth cells. A downward arrow from the top row to second row for each cell is displayed. In the second row, numbers 15, 12, and 11 are shown over the first, fourth, and fifth cells. In the third row, numbers 15, 12, 11, 4, and 3 are shown over the first, fourth, fifth, twelfth, and eleventh cells. A downward arrow between the second and the third rows, cells, 1, 2, 3, 4, 11, 12, 14 are displayed.

Return to Figure 15-102

OceanofPDF.com

Figure 15-103 Extended Description (Chapter 15)

Return to Figure 15-103

On the top: There is one independent rung that consists of an input switch and a B T D box. From the top, this box lists B T D, bit field distribution, source with Local:1:I.Data 2#0000_0000_0000_0000_1001_0110_1111_0000 labeled on its right, source bit with 8 labeled on its right, destination with Local:2:O.Data 2#0000_0000_0000_0000_0000_0000_1001_0110 labeled on the right, destination bit with 0 on its right, and length with 8 labeled on its right. At the bottom of the rung, two rows are depicted one over below. These rows have 16 cells. The first row has entries in cells: 1, 0, 0, 1, 0, 1, 1, 0, 1, 1, 1, 1, 0, 0, 0, 0. The second row has entries in cells: 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 1, 1, 0. Both the rows are mainly divided into four sections, where the first section contains numbers 15 and 12 over first and fourth cells, numbers 11 and 8 over fifth and eights cells, numbers 7 and 4 over ninth and twelfths cells, and numbers 3 and 0 over thirteenth and sixteenth cells. Eight diagonally aligned arrows from the first eight cells from the first row are pointed to last eight cells of the second row.

Return to Figure 15-103

OceanofPDF.com

Figure 15-104 Extended Description (Chapter 15)

Return to Figure 15-104

A block labeled input wire connector points downwards at a block labeled value with a pinon the right side. This pin is connected to the first input pin of the function block, through a label wire pointing at the connection. A label input reference points upwards at a block labeled value with a pin on the right side. The pin of the second block labeled as value is connect ted to the second input pin of the function block, through a dashed line and a label Boolean (zero or one). The function block shows labels value around each pin. The output pin of the function block is connected to the pin of a block labeled as value (red), through a label integer or real. A label output reference points upwards at the third block labeled value. The label value near the output pin of the function block is also highlighted in red.

Return to Figure 15-104

OceanofPDF.com

Figure 15-105 Extended Description (Chapter 15)

Return to Figure 15-105

The block has labels BAND and Boolean AND. It has four input terminals on the left side labeled as one from outside. From inside, the terminals are labeled as In 1, In 2, In 3, and In 4. The block has a terminal on the right side labeled as out from inside and zero from outside. A small rectangle with three dots is shown at the right side of the label BAND in the block. A label click to open properties box points at the small rectangle and the title of the window screen. The window screen has labels parameter and tag at the top-left side. It consists of a table with headings Vis, name, value, type, and description. It also has a check list on the left side.

Return to Figure 15-105

OceanofPDF.com

Figure 15-106 Extended Description (Chapter 15)

Return to Figure 15-106

The output pin of the first IREF is connected to input of the first block. The output of the first block is connected to the first input of the second block. The other two inputs of the second block are connected to the outputs of the second and the third IREF. A label input references points at all the three IREFs. The two outputs of the second block are connected to the inputs of the OREFs. A label output reference points at the two OREFs.

Return to Figure 15-106

OceanofPDF.com

Figure 15-107 Extended Description (Chapter 15)

Return to Figure 15-107

The output pin of an input wire connector is connected to the input pin of the first block, through a label wire pointing at the connection. The output pin of the first block is connected to the first input pin of the second block. The output pin of the second input wire connector is connected to the second input pin of the second block. The output pin of the second block is connected to the input pin of an output wire connector, through a label wire pointing at the connection. An illustration labeled as wiring elements at the top shows an input wire connector labeled as A connected to the input of a function block labeled as B.

Return to Figure 15-107

OceanofPDF.com

Figure 15-108 Extended Description (Chapter 15)

Return to Figure 15-108

In sheet one, the output wire connector is labeled as O C O N and has a label speed within it. In sheet two, the input wire connector is labeled as I C O N and has a label speed within it. Also, another input wire connector is connected to the second input of the block.

Return to Figure 15-108

OceanofPDF.com

Figure 15-109 Extended Description (Chapter 15)

Return to Figure 15-109

Each IREF has a label input reference above it. Each OREF has a label output reference above it. Each wire connector with a label O C O N is labeled as output wire connector. Each wire connector with a label I C O N is labeled as input wire connector. The first IREF is connected to the input pin of the first function block. The output pin of the first function block is connected to the first input of the second function block. The other two inputs of the second function block are connected to the other two IREFs. The first output pin of the second function block is connected to an OREF. An illustration labeled solenoid is placed on the right side of the OREF. The second output of the second function block is connected to O C O N. An illustration at the bottom-right side shows an I CO N connected to the input pin of the third function block.

Return to Figure 15-109

OceanofPDF.com

Figure 15-111 Extended Description (Chapter 15)

Return to Figure 15-111

The small rectangle is labeled as click this button to view block parameters. The parameters tab shows a table of four rows and four columns. From left, the first row is labeled as Vis, name, value, and type. From left, the second row shows an unchecked box, enable in, one, and BOOL. From left, the third row shows a checked box, source A, 0.0, and REAL. From left, the fourth row shows an unchecked box, source B, unlabeled cell, and REAL.

Return to Figure 15-111

OceanofPDF.com

Figure 15-112 Extended Description (Chapter 15)

Return to Figure 15-112

The first input of the block is connected to an input wire connector labeled as Timer_Enable_Bit. The second input of the block is connected to an input wire connector labeled as Preset_Value. The third input of the block is connected to its second output. The first output of the block is connected to an output wire connector labeled as Accumulated_Time. The block has a label TONR within it. The input terminals of the block are labeled from inside as TimerEnable, P R E, and reset. The output terminals of the block are labeled from inside as A C C, and D N.

Return to Figure 15-112

OceanofPDF.com

Figure 15-113 Extended Description (Chapter 15)

Return to Figure 15-113

The first input of the block 1 is connected to an input wire connector. The second input of the block 1 is connected to a node between the output terminal of the block 3 and an output wire connector. This connection is labeled as feedback loop. Two arrow-heads are placed at the second input of the block 1, which are labeled as assume data available indicator. The block 2 is connected between the output of the block 1 and the input of the block 3.

Return to Figure 15-113

OceanofPDF.com

Figure 15-114 Extended Description (Chapter 15)

Return to Figure 15-114

The tree organizer shows a label Main Program with a negative sign at its left. This label is connected to three labels at the bottom, which are indented to level 1. From the top, the three labels are Program Tags, Main Routine, and FBD_Sample. The new routine tab shows labels as name, FBD_Sample (in a rectangle); description, optional (in a rectangle); type, function block diagram (in a rectangle with a drop-down button); and in program, Main Program (in a rectangle with a drop-down button. The first toolbar shows an arrow-head pointing left at the top-left side and an arrow-head pointing right at the top-right side. Four symbols and nine labels enclosed in rectangles are placed relatively on the left side between the arrow-heads. From the left, the symbols are as follows a pentagon pointing right labeled as IREF, a pentagon pointing left labeled as OREF, a pentagon pointing right labeled as ICON, and pentagon pointing left labeled as OCON. From left to right, the labels are ADD, SUB, MUL, DIV, BAND, BOR, BXOR, and BNOT. These labels are labeled as function blocks. Nine labels are shown at the bottom. From left to right, the labels are arrow-head facing, arrow-head facing right, favorites (highlighted), process, drives, filters, select or limit, statistical, and exit. The second toolbar shows seven symbols on the top. The first three symbols on the left are faded. The fourth symbol is of a paper labeled as add sheet. The fifth symbol is of a paper with a right slanting line over it. The fifth symbol is a double-headed pointing left labeled as move. The sixth symbol is a double-headed pointing right labeled as sheet. the seventh symbol is of a table. Labels sheet, three (enclosed in a rectangle with drop-down button), and of four are placed below the symbols on the left side.

Return to Figure 15-114

OceanofPDF.com

Figure 15-115 Extended Description (Chapter 15)

Return to Figure 15-115

The vertical line L 1 is connected to N O proximity switches labeled sensor_1, sensor_2, and sensor_3 along three rungs. A label inputs is placed over the first rung. From the left, X I C symbols labeled Sensor_1 <Local:1:I.Data.1>, Sensor_2 <Local:1:I.Data.2>, Sensor_3 <Local:1:I.Data.3>, and O T E labeled as Caution_PL <Local:2:O.Data.4> are connected in series along a rung between the second and the third vertical lines. This rung has a label ladder logic program over it. The light labeled as Caution_P L is connected along a rung connected to the vertical line L 2 and has label output over it. The function block is connected to three input wire connectors labeled as Sensor_1, Sensor_2, and Sensor_3, through label zero. The output terminal of the function block is connected to an output wire connector labeled as Caution_P L, through a label zero. The block has a label BAND_01 over it. From within the block is labeled as BAND and Boolean AND. The input terminals from inside are labeled as In 1, In 2, and In 3. The output terminal from inside is labeled as out.

Return to Figure 15-115

OceanofPDF.com

Figure 15-116 Extended Description (Chapter 15)

Return to Figure 15-116

There are four vertical lines. From the left, the first vertical line is labeled as L 1 and the fourth vertical line is labeled as L 2. From the top, N O switches labeled as Sw_1 and S w_2 are connected along the two rungs that are connected to the line L 1. A label inputs is placed over the first rung. From the left, X I C symbol labeled as Sw_1 <Local:1:I.Data.3> and O T E labeled as SOL_1 <Local:2:O.Data.4> are connected in series along a rung connected between the second and the third vertical lines. An X I C symbol labeled as Sw_2 <Local:1:I.Data.4> is connected across the first X I C symbol. A label ladder logic is placed over the rung. A solenoid labeled as SOL_1 is connected along a rung connected to the line L 2. A label output is placed above the rung. The F B D equivalent shows a function block labeled as B O R_01. It has a label Boolean OR. The first input pin labeled as In 1 is connected to an input wire connector labeled as Sw_1, through a label zero. The second input pin labeled as In 2 is connected to an input wire connector labeled as Sw_2, through a label zero. The output pin labeled as out is connected to an output wire connector labeled as SOL_1, through a label zero.

Return to Figure 15-116

OceanofPDF.com

Figure 15-117 Extended Description (Chapter 15)

Return to Figure 15-117

There are four vertical lines. From the left, the first vertical line is labeled as L 1 and the fourth vertical line is labeled as L 2. From the top, N O temperature switch labeled as Temp_Sw, N O pressure switch labeled as Press_Sw, three N O proximity switches labeled as Sensor_1, Sensor_2, and Sensor_3 are connected along the five rungs connected at the right side of the line L 1. A label inputs is placed over the first rung. From left to right, X I C symbol labeled as Temp_Sw <Local:1:I.Data.1>, an X I O symbol (highlighted in green) labeled as Press_Sw <Local:1:I.Data.2>, and an O T E labeled as Alarm <Local:2:O.Data.4> are connected in series along a rung connected between the second and the third vertical lines. The series combination of X I C symbols labeled as Sensor_1 <Local:1:I.Data.3>, Sensor_2 <Local:1:I.Data.4>, and Sensor_3<Local:1:I.Data.5> is connected across the series connection of the X I C symbol labeled as Temp_Sw <Local:1:I.Data.1>, and the X I O symbol (highlighted in green) labeled as Press_Sw <Local:1:I.Data.2>. A label ladder logic is placed over the rung. An alarm is connected along a rung connected to the line L 2. A label output is placed over the rung connected to the line L 2. The F B D equivalent shows function blocks labeled as B N O T_01, B A N D_01, B A N D_02, and B O R_01. The first input pin of B A N D_01 is labeled as In 1 and is connected to an input wire connector labeled as Temp_Sw, through a label zero. The second pin of B A N D-01 is labeled as In 2 and is connected to output pin of the B N O T labeled as out. The input pin of the B N O T labeled as In is connected to an input wire connector labeled as Press_Sw, through a label zero. The output pin of the B A N D_01 labeled as out is connected to the first input pin of the B O R labeled as In 1. The second input pin of the B O R labeled as In 2 is connected to the output pin of the B A N D_02 labeled as out. The output pin of the B O R labeled as out is connected to an output wire connector labeled as alarm, through a label zero. The input pins of the B A N D_02 labeled as In 1, In 2, and In 3 are connected to input wire connectors labeled as Sensor_1, Sensor_2, and Sensor_3, through label zero respectively. Both the B A N D_01 and B A N D_02 have label Boolean AND. The B O R has a label Boolean OR. The B N O T has a label Boolean NOT.

Return to Figure 15-117

OceanofPDF.com

Figure 15-118 Extended Description (Chapter 15)

Return to Figure 15-118

The ladder diagram consists of 4 vertical lines. From the left, the first vertical line is labeled as L 1 and the fourth vertical line is labeled as L 2. Two rungs are connected on the right side of the line L 1. From the bottom, N C pushbutton labeled as motor_stop and N O pushbutton labeled as motor_start are connected along the rungs connected to the line L 1. An illustration of N O and N C pushbutton is shown below the second rung. A label inputs is placed over the first rung. A rung is connected between the second and the third vertical lines. From the left, an X I C symbol, the second X I C symbol (highlighted in green), and an output energize are connected in series along the rung. From the top, the first X I C symbol is labeled as start button for motor, motor_start, and <Local:1:I.Data.3>. From the top, the second X I C symbol is labeled as stop button for motor, motor_stop, and <Local:1:I.Data.4>. From the top, output energize is labeled as motor contactor coil, motor_run, and <Local:2:O.Data.4>. The third X I C symbol is connected across the first X I C symbol. From the top, the third X I C symbol is labeled as motor contactor coil, motor_run, and <Local:2:O.Data.4>. A label ladder logic program is placed over the first rung. A motor starter coil labeled as motor_run is connected along a rung connected at the left side of the line L 2. An illustration of the motor starter coil with a label contactor is shown below its symbol. A label output is placed over the rung connected to the line L 2. The F B D equivalent shows a function block labeled as B O R_01 and a function block labeled as B A N D_01. The first input pin labeled as In 1 of the B O R is connected to an input wire connector labeled as Motor_Start and start button for motor, through a label zero. The second input pin labeled as In 2 of the B O R is connected to node, The node is in between the connection of the output pin of the B A N D labeled out and an output wire connector labeled as Motor_Run and motor contactor coil, through a label zero. The output pin of the B O R labeled as out is connected to the second input terminal of the B A N D labeled as In 2, through a label zero. The second input pin of the B A N D labeled as In 1 is connected to an input wire connector labeled as Motor_Stop and stop button for motor, through a label one. The B O R has a label Boolean or within it. The B A N D has a label Boolean AND within it.

Return to Figure 15-118

OceanofPDF.com

Figure 15-119 Extended Description (Chapter 15)

Return to Figure 15-119

There are four vertical lines in the ladder diagram. From the left, the firs vertical line is labeled as L 1 and the second vertical line is labeled as L 2. A label input is placed over a rung connected to the line L 1. An N O switch labeled as Timer_Sw is connected along the rung connected to the L 1. An illustration of the switch is placed below its symbol. Four rungs are connected between the second and the third vertical lines with a label ladder logic program over them. From the left, an X I C symbol labeled as Timer_Sw, T O N instruction, and output energize E N are connected in series along the first rung. The right side of the T O N is connected to an output energize D N. From the top, the T O N is labeled as T O N, timer on delay, timer Status_Timer, preset 10,000 (an arrow pointing at it from the right side), and accum zero (an arrow pointing at it from the right side). From the left, X I C symbol labeled as Status_Timer.E N and an O T E labeled as E N_P L and <Local:2:O.Data.1> are connected in series along the second rung. From the left, X I C symbol labeled as Status_Timer.T T and an O T E labeled as T T_P L and <Local:2:O.Data.2> are connected in series along the third rung. From the left, X I C symbol labeled as Status_Timer.D N and an O T E labeled as D N_P L and <Local:2:O.Data.3> are connected in series along the fourth rung. Three rungs are connected at the left side of the line L 2 with a label outputs over them. From the top, lights labeled as E N_P L, T T_P L, and D N_P L are connected along the rungs at the left side of the line L 2. An illustration of traffic light is shown below the symbols of light. The F B D equivalent shows a function block labeled as T O N R_01. It has a label timer on delay with reset. The first input labeled as Timer enable is connect to an input wire connector labeled as Timer_Sw, through a label zero. The second input labeled as P R E is connect to an input wire connector labeled as 1,000,through a label 1,000. The first output labeled as A C C is connected to an output wire connector labeled as A C C_Value, through a label zero. The second output labeled as E N is connected to an output wire connector labeled as E N_P L, through a label zero. The third output labeled as T T is connected to an output wire connector labeled as T T_P L, through a label zero. The fourth output labeled as D N is connected to an output wire connector labeled as D N_P L, through a label zero.

Return to Figure 15-119

OceanofPDF.com

Figure 15-120 Extended Description (Chapter 15)

Return to Figure 15-120

There are four vertical lines. From the left, the first vertical line is labeled as L 1 and the fourth vertical line is labeled as L 2. Three rungs are connected at right side of the line L 1 with a label inputs placed over the first rung. From the top, N O pushbutton labeled as Restart_Button, N O limit switch labeled as Enter_Limit_Sw, and N O limit switch labeled as Exit_Limit_Sw are connected along the rungs connected to the line L 1. Four rungs are connected between the second and the third vertical lines with a label ladder logic program placed over the first rung. From the left, X I C symbol labeled as Restart_Button and <Local:1:I.Data.1>, and R ES labeled as Counter_1 are connected in series along the first rung. From the left, X I C symbol labeled as Enter_Limit_Sw and <Local:1:I.Data.3>, C T U instruction, and an output energize C U are connected in series along the second rung. The right side of the C T U is connected to an output energize D N. From the top, the C T U is labeled as C T U, count up, counter Counter_1, preset 50 (an arrow pointing at it from the right side), and accum zero (an arrow pointing at it from the right side). From the left, an X I C symbol labeled as Exit_Limit_Sw and <Local:1:I.Data.4>, C T D instruction, and an output energize C D are connected in series along the third rung. The right side of the C T D is connected to an output energize D N. From the top, the C T D is labeled as C T D, count down, counter Counter_1, preset 50 (an arrow pointing at it from the right side), and accum zero (an arrow pointing at it from the right side). From the left, an X I O symbol (highlighted in green) labeled as Counter_1.D N is connected to O T E (highlighted in green) labeled as Conveyor_Contactor and <Local:2:O.Data.2> in series along the fourth rung. A rung is connected at the left side of the line L 2 with a label output over it. A relay coil C labeled as Conveyor_Contactor is connected along the rung connected to the line L 2. The F B D equivalent shows a block labeled as C T U D_01 and B N O T_01. The first input labeled as C U enable of the C T U D is connected to an input wire connector labeled as Enter_Limit_Sw, through a label zero. The second input labeled as C D enable of the C T U D is connected to an input wire connector labeled as Exit_Limit_Sw, through a label zero. The third input labeled as P R E of the C T U D is connected to an input wire connector labeled as 50, through a label 50. The fourth input labeled as reset of the C T U D is connected to an input wire connector labeled as Restart_Button, through a label zero. The first output pin labeled as A C C of the C T U D is connected to an output wire connector labeled as A C C, through a label zero. The second output pin labeled as D N of the C T U D is connected to input pin labeled as In of the B N O T. The output pin of the B N O T labeled as out is connected to an output wire connector labeled as conveyor_contactor, through a label one.

Return to Figure 15-120

OceanofPDF.com

Figure 15-121 Extended Description (Chapter 15)

Return to Figure 15-121

There are four vertical lines. From the left, the first vertical line is labeled as L 1 and the fourth vertical line is labeled as L 2. From the top, N O pushbuttons labeled as Count_P B and Reset_P B are connected along two rungs connected at the right side of the line L 1 with a label inputs over the first rung. Five rungs are connected between the second and the third vertical lines with a label ladder logic program is placed over the first rung. From the left X I C symbol labeled as Count_PB <Local:1:I.Data.1>, C T U instruction and output energize E N are connected in series along the first rung. From the top, C T U is labeled as C T U, count up, counter C 1, preset 25, and accum zero. From the left, G R T, L E S, and an O T E labeled as PL_1 <Local:2:O.Data.1> are connected in series along the second rung. From the top, G R T is labeled as greater than (A greater than B), source A C1.A C C, zero, and source B 5. L E S is labeled as L E S, less than (A less than B), source A C1.A C C, zero, and source B 10. From the left E Q U and an O T E labeled as PL_2 <Local:2:O.Data.2> are connected in series along the third rung. E Q U is labeled as E Q U, equal, source A C1.A C C, zero, and source B 15. From the left, N E Q, and an O T E labeled as PL_3 <Local:2:O.Data.3> are connected in series along the fourth rung. N E Q is labeled as N E Q, not equal, source A C1.A C C, zero, and source B 20. From the left, X I C labeled as C1.D N and R E S labeled as C 1 are connected in series along the fifth rung. An X I C labeled as Reset_PB <Local:1:I.Data.2> is connected across the X I C labeled as C1.D N. From the top, lights labeled as P L_1, P L_2, and P L_3 are connected along three rungs connected at the left side of the line L 2 with a label outputs over the first rung. The F B D equivalent is shown in four sheets. The sheet one of four shows a function block labeled as CTUD_01 with four inputs and two outputs. The first input labeled as C U enable is connected to an input wire connector labeled as Count_P B, through a label zero. The second terminal labeled as C D enable is unconnected. The third input labeled as P R E is connected to an input wire connector labeled as 25, through a label 25. The fourth input labeled as reset is connected to the output of a block labeled BOR_01, through a label zero. The first output terminal labeled as A C C is connected to an output wire connector labeled as A C C, through a label zero. The first input labeled as In 1 of the BOR is connected to an input wire connector labeled as Reset_P B, through a label zero. The second input labeled In 2 of BOR is connected to the second output labeled as D N of the C T U D. The block labeled C T U D has a label count up or down. The block labeled as BOR has a label Boolean or. The sheet two of four shows three blocks labeled as G R T_02, B A N D_01, and L E S_02. The first input pin labeled as source A of the G R T is connected to an input wire connector labeled as A C C and 1-C2, through a label zero and a node connected to the first input labeled as source A of the L E S. The second input pin labeled as source B of the G R T is connected to an input wire connector labeled as five, through a label five. The second input pin labeled as source B of the L E S is connected to an input wire connector labeled as 10, through a label 10. The output pin labeled as dest of the G R T is connected to the first input pin labeled as In 1 of the B A N D, through a label zero. The output pin labeled as dest of the L E S is connected to the second input pin labeled as In 2 of the B A N D, through a label zero. The output pin labeled as out is connected to an output wire connector labeled as

PL_1, through a label zero. The G R T has a label greater than (A greater than B). The BA N D has a label Boolean AND. The L E S has a label less than (A less than B). The sheet three of four shows E Q U_01. It has a label equal in it. The first input pin labeled as source A is connected to an input wire connector labeled as A C C, through a label zero. The second input pin labeled as source B is connected to an input wire connector labeled as 15 and 1-C2, through a label 15. The output pin labeled as dest is connected to an output wire connector labeled as P L_2, through a label zero. The sheet four of four shows N E Q_01. It has a label not equal in it. The first input pin labeled as source A is connected to an input wire connector labeled as A C C, through a label zero. The second input pin labeled as source B is connected to an input wire connector labeled as 20 and 1-C2, through a label 20. The output pin labeled as dest is connected to an output wire connector labeled as P L_3, through a label zero.

Return to Figure 15-121

OceanofPDF.com

Table of Contents

1. Table of Contents and Preface
    1. Cover Page
    2. Title Page
    3. Copyright Information
    4. About the Author
    5. Contents
    6. Preface
    7. Chapter Updates to Note
    8. Acknowledgments
    9. Chapter Objectives
    10. Ancillaries that Work
    11. Digital Resources
    12. Connect
2. Chapter 1: Programmable Logic Controllers (PLCs): An Overview
    1. Chapter 1 Introduction
    2. 1.1 Programmable Logic Controllers
    3. 1.2 Parts of a PLC
    4. 1.3 Principles of Operation
    5. 1.4 Modifying the Operation
    6. 1.5 PLCs versus Computers
    7. 1.6 PLC Size and Application
    8. Chapter 1 Review Questions
    9. Chapter 1 Problems
3. Chapter 2: PLC Hardware Components
    1. Chapter 2 Introduction
    2. 2.1 The I/O Section
    3. 2.2 Discrete I/O Modules
    4. 2.3 Analog I/O Modules
    5. 2.4 Special I/O Modules
        1. High-Speed Counter Module
        2. Encoder-Counter Module
        3. Stepper-Motor Module
        4. Motion and Position Control Module
        5. Communication Modules
    6. 2.5 I/O Specifications
        1. Typical Discrete (Digital) I/O Module Specifications
        2. Typical Analog I/O Module Specifications
    7. 2.6 The Central Processing Unit (CPU)
        1. RUN Position
        2. PROG Position
        3. REM Position
    8. 2.7 Memory Design
    9. 2.8 Memory Types
    10. 2.9 Programming Terminal Devices
    11. 2.10 Recording and Retrieving Data

12. 2.11 Human Machine Interfaces (HMIs)
    1. HMI Monitor and Enclosure
    2. Alarms
    3. Event History
    4. Trend
    5. Graphics Library
13. Chapter 2 Review Questions
14. Chapter 2 Problems

4. Chapter 3: Number Systems and Codes
    1. Chapter 3 Introduction
    2. 3.1 Decimal System
    3. 3.2 Binary System
    4. 3.3 Negative Numbers
    5. 3.4 Octal System
    6. 3.5 Hexadecimal System
    7. 3.6 Binary Coded Decimal (BCD) System
    8. 3.7 Gray Code
    9. 3.8 ASCII Code
    10. 3.9 Parity Bit
    11. 3.10 Binary Arithmetic
    12. 3.11 Floating Point Arithmetic
    13. Chapter 3 Review Questions
    14. Chapter 3 Problems
5. Chapter 4: Fundamentals of Logic
    1. Chapter 4 Introduction
    2. 4.1 The Binary Concept
    3. 4.2 AND, OR, and NOT Functions
        1. The AND Function
        2. The OR Function
        3. The NOT Function
        4. The Exclusive-OR (XOR) Function
    4. 4.3 Boolean Algebra
    5. 4.4 Developing Logic Gate Circuits from Boolean Expressions
    6. 4.5 Producing the Boolean Equation for a Given Logic Gate Circuit
    7. 4.6 Hardwired Logic versus Programmed Logic
    8. 4.7 Programming Word Level Logic Instructions
    9. Chapter 4 Review Questions
    10. Chapter 4 Problems
6. Chapter 5: Basics of PLC Programming
    1. Chapter 5 Introduction
    2. 5.1 Processor Memory Organization
        1. Program Files
        2. Data Files
    3. 5.2 Program Scan
    4. 5.3 PLC Programming Languages
    5. 5.4 Bit-Level Logic Instructions
    6. 5.5 Instruction Addressing
    7. 5.6 Branch Instructions
    8. 5.7 Internal Relay Instructions
    9. 5.8 Programming Examine If Closed and Examine If Open Instructions

10. 5.9 Entering the Ladder Diagram
11. 5.10 Modes of Operation
12. 5.11 Connecting with Analog Devices
13. Chapter 5 Review Questions
14. Chapter 5 Problems

7. Chapter 6: Developing Fundamental PLC Wiring Diagrams and Ladder Logic Programs
    1. Chapter 6 Introduction
    2. 6.1 Electromagnetic Control Relays
    3. 6.2 Contactors
    4. 6.3 Motor Starters
    5. 6.4 Manually Operated Switches
    6. 6.5 Mechanically Operated Switches
    7. 6.6 Sensors
        1. Proximity Sensor
        2. Magnetic Reed Switch
        3. Light Sensors
        4. Ultrasonic Sensors
        5. Strain/Weight Sensors
        6. Temperature Sensors
        7. Flow Measurement
        8. Velocity and Position Sensors
    8. 6.7 Output Control Devices
    9. 6.8 Seal-In Circuits
    10. 6.9 Electrical Interlocking Circuits
    11. 6.10 Latching Relays
    12. 6.11 Converting Relay Schematics into PLC Ladder Programs
    13. 6.12 Writing a Ladder Logic Program Directly from a Narrative Description
    14. 6.13 Instrumentation
    15. Chapter 6 Review Questions
    16. Chapter 6 Problems
8. Chapter 7: Programming Timers
    1. Chapter 7 Introduction
    2. 7.1 Mechanical Timing Relays
    3. 7.2 Timer Instructions
    4. 7.3 On-Delay Timer Instruction
    5. 7.4 Off-Delay Timer Instruction
    6. 7.5 Retentive Timer
    7. 7.6 Cascading Timers
    8. Chapter 7 Review Questions
    9. Chapter 7 Problems
9. Chapter 8: Programming Counters
    1. Chapter 8 Introduction
    2. 8.1 Counter Instructions
    3. 8.2 Up-Counter
        1. One-Shot Instruction
    4. 8.3 Down-Counter
    5. 8.4 Cascading Counters
    6. 8.5 Incremental Encoder-Counter Applications
    7. 8.6 Combining Counter and Timer Functions
    8. 8.7 High-Speed Counters

    9. Chapter 8 Review Questions
    10. Chapter 8 Problems
10. Chapter 9: Program Control Instructions
    1. Chapter 9 Introduction
    2. 9.1 Program Control
    3. 9.2 Master Control Reset Instruction
    4. 9.3 Jump and Label Instructions
    5. 9.4 Subroutine Functions
    6. 9.5 Immediate Input and Immediate Output Instructions
    7. 9.6 Forcing External I/O Addresses
    8. 9.7 Safety Circuitry
    9. 9.8 Fault Routine
    10. 9.9 Temporary End Instruction
    11. Chapter 9 Review Questions
    12. Chapter 9 Problems
11. Chapter 10: Data Manipulation Instructions
    1. Chapter 10 Introduction
    2. 10.1 Data Manipulation
    3. 10.2 Data Transfer Operations
    4. 10.3 Data Compare Instructions
    5. 10.4 Data Manipulation Programs
    6. 10.5 Numerical Data I/O Interfaces
    7. 10.6 Closed-Loop Control
    8. Chapter 10 Review Questions
    9. Chapter 10 Problems
12. Chapter 11: Math Instructions
    1. Chapter 11 Introduction
    2. 11.1 Math Instructions
    3. 11.2 Addition Instruction
    4. 11.3 Subtraction Instruction
    5. 11.4 Multiplication Instruction
    6. 11.5 Division Instruction
    7. 11.6 Other Word-Level Math Instructions
    8. 11.7 File Arithmetic Operations
    9. Chapter 11 Review Questions
    10. Chapter 11 Problems
13. Chapter 12: Sequencer and Shift Register Instructions
    1. Chapter 12 Introduction
    2. 12.1 Mechanical Sequencers
    3. 12.2 Sequencer Instructions
    4. 12.3 Sequencer Programs
    5. 12.4 Bit Shift Registers
    6. 12.5 Word Shift Operations
    7. Chapter 12 Review Questions
    8. Chapter 12 Problems
14. Chapter 13: PLC Installation Practices, Editing, and Troubleshooting
    1. Chapter 13 Introduction
    2. 13.1 PLC Enclosures
    3. 13.2 Electrical Noise
    4. 13.3 Leaky Inputs and Outputs

5. 13.4 Grounding
6. 13.5 Voltage Variations and Surges
7. 13.6 Program Editing and Commissioning
8. 13.7 Programming and Monitoring
9. 13.8 Preventive Maintenance
10. 13.9 Troubleshooting
    1. Processor Module
    2. Input Malfunctions
    3. Output Malfunctions
    4. Ladder Logic Program
11. 13.10 PLC Programming Software
12. Chapter 13 Review Questions
13. Chapter 13 Problems

15. Chapter 14: Process Control, Network Systems, and SCADA
    1. Chapter 14 Introduction
    2. 14.1 Types of Processes
    3. 14.2 Structure of Control Systems
        1. Sensors
        2. Human Machine Interface (HMI)
        3. Signal Conditioning
        4. Actuators
        5. Controller
    4. 14.3 On/Off Control
    5. 14.4 PID Control
        1. Manual
        2. Semiautomatic or Autotune
        3. Fully Automatic or Intelligent
    6. 14.5 Motion Control
        1. Programmable Logic Controller
        2. Motion Module
        3. Servo Drive
        4. Servo Motor
    7. 14.6 Data Communications
        1. Data Highway
        2. Serial Communication
        3. DeviceNet
        4. ControlNet
        5. EtherNet/IP
        6. Modbus
        7. Fieldbus
        8. PROFIBUS-DP
    8. 14.7 Supervisory Control and Data Acquisition (SCADA)
    9. 14.8 Variable Frequency Drive (VFD) with PLC Control
        1. Drive Control
        2. Drive Status
    10. 14.9 PLC and HMI Programming
        1. PLC Programming
        2. Graphic Display
        3. HMI/PLC Communication
        4. Runtime Application

11. Chapter 14 Review Questions
12. Chapter 14 Problems

16. Chapter 15: ControlLogix Controllers
    1. Chapter 15 Introduction
    2. Part 1 Memory and Project Organization
        1. Memory Layout
        2. Configuration
        3. Project
        4. Tasks
        5. Programs
        6. Routines
        7. Tags
        8. Structures
        9. Creating Tags
        10. Monitoring and Editing Tags
        11. Array
        12. Part 1 Review Questions
    3. Part 2 Bit-Level Programming
        1. Program Scan
        2. Creating Ladder Logic
        3. Tag-Based Addressing
        4. Adding Ladder Logic to the Main Routine
        5. Internal Relay Instructions
        6. Latch and Unlatch Instructions
        7. One-Shot Instruction
        8. Part 2 Review Questions
        9. Part 2 Problems
    4. Part 3 Programming Timers
        1. Timer Predefined Structure
        2. On-Delay Timer (TON)
        3. Off-Delay Timer (TOF)
        4. Retentive Timer On (RTO)
        5. Cascading of Timers
        6. Part 3 Review Questions
        7. Part 3 Problems
    5. Part 4 Programming Counters
        1. Counters
        2. Count-Up (CTU) Counter
        3. Count-Down (CTD) Counter
        4. Combining Counter and Timer Functions
        5. Part 4 Review Questions
        6. Part 4 Problems
    6. Part 5 Math, Comparison, and Move Instructions
        1. Math Instructions
        2. Comparison Instructions
        3. Move Instructions
        4. Combining Math, Comparison, and Move Instructions
        5. Part 5 Review Questions
        6. Part 5 Problems
    7. Part 6 Function Block Programming

        1. Function Block Diagram (FBD)
        2. FBD Programming
        3. Part 6 Review Questions
        4. Part 6 Problems
17. Glossary
    1. Glossary
18. Index
    1. Index
19. Accessibility Content: Extended Descriptions for Images
    1. Book Cover Extended Description (Front Matter)
    2. The Image Shows Extended Description (Front Matter)
    3. A Ladder Diagram Extended Description (Front Matter)
    4. An Illustration Shows Extended Description (Front Matter)
    5. A Schematic Diagram Shows Extended Description (Front Matter)
    6. The Screenshot Extended Description (Front Matter)
    7. An Illustration Extended Description (Front Matter)
    8. The Book Page Extended Description (Front Matter)
    9. The Book Page Extended Description (Front Matter)
    10. Mcgraw Hill Connect and Proctorio Logos Extended Description (Front Matter)
    11. The Laptop Screen Showcases Extended Description (Front Matter)
    12. The EVERGREEN Logo Extended Description (Front Matter)
    13. A Line with 5 Turns Is Shown Extended Description (Front Matter)
    14. A Screenshot Extended Description (Chapter 1)
    15. Figure 1-2 Extended Description (Chapter 1)
    16. Figure 1-3 Extended Description (Chapter 1)
    17. Figure 1-6 Extended Description (Chapter 1)
    18. Figure 1-7 Extended Description (Chapter 1)
    19. Figure 1-8 Extended Description (Chapter 1)
    20. Figure 1-11 Extended Description (Chapter 1)
    21. Figure 1-12 Extended Description (Chapter 1)
    22. Figure 1-13 Extended Description (Chapter 1)
    23. Figure 1-14 Extended Description (Chapter 1)
    24. Figure 1-15 Extended Description (Chapter 1)
    25. Figure 1-16 Extended Description (Chapter 1)
    26. Figure 1-17 Extended Description (Chapter 1)
    27. Figure 1-18 Extended Description (Chapter 1)
    28. Figure 1-19 Extended Description (Chapter 1)
    29. Figure 1-20 Extended Description (Chapter 1)
    30. Figure 1-21 Extended Description (Chapter 1)
    31. Figure 1-23 Extended Description (Chapter 1)
    32. Figure 1-24 Extended Description (Chapter 1)
    33. Figure 1-25 Extended Description (Chapter 1)
    34. Figure 2-2 Extended Description (Chapter 2)
    35. Figure 2-3 Extended Description (Chapter 2)
    36. Figure 2-4 Extended Description (Chapter 2)
    37. Figure 2-5 Extended Description (Chapter 2)
    38. Figure 2-7 Extended Description (Chapter 2)
    39. Figure 2-9 Extended Description (Chapter 2)
    40. Figure 2-10 Extended Description (Chapter 2)
    41. Figure 2-11 Extended Description (Chapter 2)

42. Figure 2-12 Extended Description (Chapter 2)
43. Figure 2-13 Extended Description (Chapter 2)
44. Figure 2-14 Extended Description (Chapter 2)
45. Figure 2-15 Extended Description (Chapter 2)
46. Figure 2-16 Extended Description (Chapter 2)
47. Figure 2-17 Extended Description (Chapter 2)
48. Figure 2-18 Extended Description (Chapter 2)
49. Figure 2-19 Extended Description (Chapter 2)
50. Figure 2-20 Extended Description (Chapter 2)
51. Figure 2-21 Extended Description (Chapter 2)
52. Figure 2-22 Extended Description (Chapter 2)
53. Figure 2-23 Extended Description (Chapter 2)
54. Figure 2-24 Extended Description (Chapter 2)
55. Figure 2-25 Extended Description (Chapter 2)
56. Figure 2-27 Extended Description (Chapter 2)
57. Figure 2-28 Extended Description (Chapter 2)
58. Figure 2-29 Extended Description (Chapter 2)
59. Figure 2-32 Extended Description (Chapter 2)
60. Figure 2-34 Extended Description (Chapter 2)
61. Figure 2-39 Extended Description (Chapter 2)
62. Figure 2-41 Extended Description (Chapter 2)
63. Figure 2-42 Extended Description (Chapter 2)
64. A Block Diagram Extended Description (Chapter 3)
65. Figure 3-1 Extended Description (Chapter 3)
66. Figure 3-2 Extended Description (Chapter 3)
67. Figure 3-3 Extended Description (Chapter 3)
68. Figure 3-4 Extended Description (Chapter 3)
69. Figure 3-5 Extended Description (Chapter 3)
70. Figure 3-6 Extended Description (Chapter 3)
71. Table 3-2 Extended Description (Chapter 3)
72. Table 3-3 Extended Description (Chapter 3)
73. Figure 3-7 Extended Description (Chapter 3)
74. Figure 3-8 Extended Description (Chapter 3)
75. Figure 3-9 Extended Description (Chapter 3)
76. Figure 3-10 Extended Description (Chapter 3)
77. Figure 3-11 Extended Description (Chapter 3)
78. Figure 3-12 Extended Description (Chapter 3)
79. Figure 3-13 Extended Description (Chapter 3)
80. Three Steps Extended Description (Chapter 3)
81. Calculation for Subtracting Numbers Extended Description (Chapter 3)
82. Figure 3-14 Extended Description (Chapter 3)
83. Figure 3-15 Extended Description (Chapter 3)
84. Figure 3-16 Extended Description (Chapter 3)
85. Logic for Boolean Expression Extended Description (Chapter 4)
86. Figure 4-1 Extended Description (Chapter 4)
87. Figure 4-2 Extended Description (Chapter 4)
88. Figure 4-3 Extended Description (Chapter 4)
89. Figure 4-4 Extended Description (Chapter 4)
90. Figure 4-5 Extended Description (Chapter 4)
91. Figure 4-6 Extended Description (Chapter 4)

92. Figure 4-7 Extended Description (Chapter 4)
93. Figure 4-8 Extended Description (Chapter 4)
94. Figure 4-9 Extended Description (Chapter 4)
95. Figure 4-10 Extended Description (Chapter 4)
96. Figure 4-11 Extended Description (Chapter 4)
97. Figure 4-12 Extended Description (Chapter 4)
98. Figure 4-13 Extended Description (Chapter 4)
99. Figure 4-14 Extended Description (Chapter 4)
100. Figure 4-15 Extended Description (Chapter 4)
101. Figure 4-16 Extended Description (Chapter 4)
102. Figure 4-17 Extended Description (Chapter 4)
103. Figure 4-18 Extended Description (Chapter 4)
104. Figure 4-19 Extended Description (Chapter 4)
105. Figure 4-20 Extended Description (Chapter 4)
106. Figure 4-21 Extended Description (Chapter 4)
107. Figure 4-22 Extended Description (Chapter 4)
108. Figure 4-23 Extended Description (Chapter 4)
109. Example 4-1 Extended Description (Chapter 4)
110. Example 4-2 Extended Description (Chapter 4)
111. Example 4-3 Extended Description (Chapter 4)
112. Example 4-4 Extended Description (Chapter 4)
113. Example 4-5 Extended Description (Chapter 4)
114. Example 4-6 Extended Description (Chapter 4)
115. Example 4-7 Extended Description (Chapter 4)
116. Example 4-8 Extended Description (Chapter 4)
117. Example 4-9 Extended Description (Chapter 4)
118. Figure 4-24 Extended Description (Chapter 4)
119. Figure 4-25 Extended Description (Chapter 4)
120. Figure 4-26 Extended Description (Chapter 4)
121. Figure 4-27 Extended Description (Chapter 4)
122. Figure 4-28(a-b) Extended Description (Chapter 4)
123. Figure 4-29 Extended Description (Chapter 4)
124. Figure 4-30 Extended Description (Chapter 4)
125. Figure 4-31(a-f) Extended Description (Chapter 4)
126. Figure 4-32 Extended Description (Chapter 4)
127. A Table Extended Description (Chapter 5)
128. Figure 5-1 Extended Description (Chapter 5)
129. Figure 5-2 Extended Description (Chapter 5)
130. Figure 5-3 Extended Description (Chapter 5)
131. Figure 5-4 Extended Description (Chapter 5)
132. Figure 5-5 Extended Description (Chapter 5)
133. Figure 5-6 Extended Description (Chapter 5)
134. Figure 5-7 Extended Description (Chapter 5)
135. Figure 5-8 Extended Description (Chapter 5)
136. Figure 5-9 Extended Description (Chapter 5)
137. Figure 5-10 Extended Description (Chapter 5)
138. Figure 5-11 Extended Description (Chapter 5)
139. Figure 5-12 Extended Description (Chapter 5)
140. Figure 5-13 Extended Description (Chapter 5)
141. Figure 5-14(a-c) Extended Description (Chapter 15)

142. Figure 5-15 Extended Description (Chapter 5)
143. Figure 5-16 Extended Description (Chapter 5)
144. Figure 5-17 Extended Description (Chapter 5)
145. Figure 5-18 Extended Description (Chapter 5)
146. Figure 5-19 Extended Description (Chapter 5)
147. Figure 5-20 Extended Description (Chapter 5)
148. Figure 5-22 Extended Description (Chapter 5)
149. Figure 5-23 Extended Description (Chapter 5)
150. Figure 5-24 Extended Description (Chapter 5)
151. Figure 5-25 Extended Description (Chapter 5)
152. Figure 5-26 Extended Description (Chapter 5)
153. Figure 5-27 Extended Description (Chapter 5)
154. Figure 5-28 Extended Description (Chapter 5)
155. Figure 5-29 Extended Description (Chapter 5)
156. Figure 5-30 Extended Description (Chapter 5)
157. Figure 5-31 Extended Description (Chapter 5)
158. Figure 5-32 Extended Description (Chapter 5)
159. Figure 5-33 Extended Description (Chapter 5)
160. Figure 5-34 Extended Description (Chapter 5)
161. Figure 5-35 Extended Description (Chapter 5)
162. Figure 5-36 Extended Description (Chapter 5)
163. Figure 5-37 Extended Description (Chapter 5)
164. Figure 5-38 Extended Description (Chapter 5)
165. Figure 5-39 Extended Description (Chapter 5)
166. Figure 5-40 Extended Description (Chapter 5)
167. Figure 5-41 Extended Description (Chapter 5)
168. Figure 5-42 Extended Description (Chapter 5)
169. Figure 5-43 Extended Description (Chapter 5)
170. Figure 5-44 Extended Description (Chapter 5)
171. Figure 5-45 Extended Description (Chapter 5)
172. Figure 5-46 Extended Description (Chapter 5)
173. Figure 5-47 Extended Description (Chapter 5)
174. Figure 5-48 Extended Description (Chapter 5)
175. Figure 5-49 Extended Description (Chapter 5)
176. Figure 5-50(a-c) Extended Description (Chapter 15)
177. Figure 5-51 Extended Description (Chapter 5)
178. Figure 5-52 Extended Description (Chapter 5)
179. Figure 5-53 Extended Description (Chapter 5)
180. Figure 5-54 Extended Description (Chapter 5)
181. Figure 5-56 Extended Description (Chapter 5)
182. Figure 5-57 Extended Description (Chapter 5)
183. Figure 6-1 Extended Description (Chapter 6)
184. Figure 6-3 Extended Description (Chapter 6)
185. Figure 6-4 Extended Description (Chapter 6)
186. Figure 6-5 Extended Description (Chapter 6)
187. Figure 6-6 Extended Description (Chapter 6)
188. Figure 6-8 Extended Description (Chapter 6)
189. Figure 6-9 Extended Description (Chapter 6)
190. Figure 6-11 Extended Description (Chapter 6)
191. Figure 6-13 Extended Description (Chapter 6)

192. Figure 6-18 Extended Description (Chapter 6)
193. Figure 6-19 Extended Description (Chapter 6)
194. Figure 6-20 Extended Description (Chapter 6)
195. Figure 6-21 Extended Description (Chapter 6)
196. Figure 6-23 Extended Description (Chapter 6)
197. Figure 6-28 Extended Description (Chapter 6)
198. Figure 6-31 Extended Description (Chapter 6)
199. Figure 6-32 Extended Description (Chapter 6)
200. Figure 6-33 Extended Description (Chapter 6)
201. Figure 6-34 Extended Description (Chapter 6)
202. Figure 6-35 Extended Description (Chapter 6)
203. Figure 6-36 Extended Description (Chapter 6)
204. Figure 6-37 Extended Description (Chapter 6)
205. Figure 6-38 Extended Description (Chapter 6)
206. Figure 6-39 Extended Description (Chapter 6)
207. Figure 6-40 Extended Description (Chapter 6)
208. Figure 6-41 Extended Description (Chapter 6)
209. Figure 6-42 Extended Description (Chapter 6)
210. Figure 6-44 Extended Description (Chapter 6)
211. Figure 6-45 Extended Description (Chapter 6)
212. Figure 6-46 Extended Description (Chapter 6)
213. Figure 6-47 Extended Description (Chapter 6)
214. Figure 6-48 Extended Description (Chapter 6)
215. Figure 6-49 Extended Description (Chapter 6)
216. Figure 6-50 Extended Description (Chapter 6)
217. Figure 6-52 Extended Description (Chapter 6)
218. Figure 6-53 Extended Description (Chapter 6)
219. Figure 6-54 Extended Description (Chapter 6)
220. Figure 6-55 Extended Description (Chapter 6)
221. Figure 6-56 Extended Description (Chapter 6)
222. Figure 6-57 Extended Description (Chapter 6)
223. Figure 6-60 Extended Description (Chapter 6)
224. Figure 6-61 Extended Description (Chapter 6)
225. Figure 6-62 Extended Description (Chapter 6)
226. Figure 6-63 Extended Description (Chapter 6)
227. Figure 6-64 Extended Description (Chapter 6)
228. Figure 6-65 Extended Description (Chapter 6)
229. Figure 6-66 Extended Description (Chapter 6)
230. Figure 6-67 Extended Description (Chapter 6)
231. Figure 6-68 Extended Description (Chapter 6)
232. Figure 6-69 Extended Description (Chapter 6)
233. Figure 6-70 Extended Description (Chapter 6)
234. Figure 6-71 Extended Description (Chapter 6)
235. Figure 6-72 Extended Description (Chapter 6)
236. Figure 6-73 Extended Description (Chapter 6)
237. Figure 6-74 Extended Description (Chapter 6)
238. Figure 6-75 Extended Description (Chapter 6)
239. Figure 6-76 Extended Description (Chapter 6)
240. Figure 6-77 Extended Description (Chapter 6)
241. A Toolbar Window Extended Description (Chapter 7)

242. Figure 7-2 Extended Description (Chapter 7)
243. Figure 7-3 Extended Description (Chapter 7)
244. Figure 7-4 Extended Description (Chapter 7)
245. Figure 7-5 Extended Description (Chapter 7)
246. Figure 7-6 Extended Description (Chapter 7)
247. Figure 7-7 Extended Description (Chapter 7)
248. Figure 7-8 Extended Description (Chapter 7)
249. Figure 7-9 Extended Description (Chapter 7)
250. Figure 7-10 Extended Description (Chapter 7)
251. Figure 7-11 Extended Description (Chapter 7)
252. Figure 7-12 Extended Description (Chapter 7)
253. Figure 7-13 Extended Description (Chapter 7)
254. Figure 7-14 Extended Description (Chapter 7)
255. Figure 7-15 Extended Description (Chapter 7)
256. Figure 7-16 Extended Description (Chapter 7)
257. Figure 7-17 Extended Description (Chapter 7)
258. Figure 7-18 Extended Description (Chapter 7)
259. Figure 7-19 Extended Description (Chapter 7)
260. Figure 7-20 Extended Description (Chapter 7)
261. Figure 7-21 Extended Description (Chapter 7)
262. Figure 7-22 Extended Description (Chapter 7)
263. Figure 7-23 Extended Description (Chapter 7)
264. Figure 7-25 Extended Description (Chapter 7)
265. Figure 7-26 Extended Description (Chapter 7)
266. Figure 7-27 Extended Description (Chapter 7)
267. Figure 7-28 Extended Description (Chapter 7)
268. Figure 7-29 Extended Description (Chapter 7)
269. Figure 7-30 Extended Description (Chapter 7)
270. Figure 7-31 Extended Description (Chapter 7)
271. Figure 7-32 Extended Description (Chapter 7)
272. Figure 7-33 Extended Description (Chapter 7)
273. Figure 7-34 Extended Description (Chapter 7)
274. Figure 7-35 Extended Description (Chapter 7)
275. Figure 7-36 Extended Description (Chapter 7)
276. Figure 7-37 Extended Description (Chapter 7)
277. Figure 7-38 Extended Description (Chapter 7)
278. Figure 7-39 Extended Description (Chapter 7)
279. Figure 7-40 Extended Description (Chapter 7)
280. Figure 7-41 Extended Description (Chapter 7)
281. Figure 7-42 Extended Description (Chapter 7)
282. Figure 7-43 Extended Description (Chapter 7)
283. Figure 7-44 Extended Description (Chapter 7)
284. A Toolbar Window Extended Description (Chapter 8)
285. Figure 8-1 Extended Description (Chapter 8)
286. Figure 8-2 Extended Description (Chapter 8)
287. Figure 8-3 Extended Description (Chapter 8)
288. Figure 8-4 Extended Description (Chapter 8)
289. Figure 8-5(a-b) Extended Description (Chapter 8)
290. Figure 8-6 Extended Description (Chapter 8)
291. Figure 8-7 Extended Description (Chapter 8)

292. Figure 8-8 Extended Description (Chapter 8)
293. Figure 8-9 Extended Description (Chapter 8)
294. Figure 8-10 Extended Description (Chapter 8)
295. Figure 8-11 Extended Description (Chapter 8)
296. Figure 8-12 Extended Description (Chapter 8)
297. Figure 8-13 Extended Description (Chapter 8)
298. Figure 8-14 Extended Description (Chapter 8)
299. Figure 8-15 Extended Description (Chapter 8)
300. Figure 8-16 Extended Description (Chapter 8)
301. Figure 8-17 Extended Description (Chapter 8)
302. Figure 8-18 Extended Description (Chapter 8)
303. Figure 8-19(a-b) Extended Description (Chapter 8)
304. Figure 8-20 Extended Description (Chapter 8)
305. Figure 8-21 Extended Description (Chapter 8)
306. Figure 8-22 Extended Description (Chapter 8)
307. Figure 8-23(a-b) Extended Description (Chapter 8)
308. Figure 8-24 Extended Description (Chapter 8)
309. Figure 8-25 Extended Description (Chapter 8)
310. Figure 8-26 Extended Description (Chapter 8)
311. Figure 8-27 Extended Description (Chapter 8)
312. Figure 8-28 Extended Description (Chapter 8)
313. Figure 8-29 Extended Description (Chapter 8)
314. Figure 8-30(a-b) Extended Description (Chapter 8)
315. Figure 8-31(a-b) Extended Description (Chapter 8)
316. Figure 8-32 Extended Description (Chapter 8)
317. Figure 8-33 Extended Description (Chapter 8)
318. Figure 8-34 Extended Description (Chapter 8)
319. Figure 8-35 Extended Description (Chapter 8)
320. Figure 8-36 Extended Description (Chapter 8)
321. Figure 8-37 Extended Description (Chapter 8)
322. Figure 8-38 Extended Description (Chapter 8)
323. Figure 8-39 Extended Description (Chapter 8)
324. Figure 8-40 Extended Description (Chapter 8)
325. Figure 8-41 Extended Description (Chapter 8)
326. Figure 8-42 Extended Description (Chapter 8)
327. Ladder Diagram Extended Description (Chapter 9)
328. Figure 9-1 Extended Description (Chapter 9)
329. Figure 9-2 Extended Description (Chapter 9)
330. Figure 9-3 Extended Description (Chapter 9)
331. Figure 9-4 Extended Description (Chapter 9)
332. Figure 9-5 Extended Description (Chapter 9)
333. Figure 9-6 Extended Description (Chapter 9)
334. Figure 9-7 Extended Description (Chapter 9)
335. Figure 9-8 Extended Description (Chapter 9)
336. Figure 9-9 Extended Description (Chapter 9)
337. Figure 9-10 Extended Description (Chapter 9)
338. Figure 9-11(a-b) Extended Description (Chapter 9)
339. Figure 9-12 Extended Description (Chapter 9)
340. Figure 9-13 Extended Description (Chapter 9)
341. Figure 9-14 Extended Description (Chapter 9)

342. Figure 9-15 Extended Description (Chapter 9)
343. Figure 9-16 Extended Description (Chapter 9)
344. Figure 9-17 Extended Description (Chapter 9)
345. Figure 9-18 Extended Description (Chapter 9)
346. Figure 9-19 Extended Description (Chapter 9)
347. Figure 9-20 Extended Description (Chapter 9)
348. Figure 9-21 Extended Description (Chapter 9)
349. Figure 9-22 Extended Description (Chapter 9)
350. Figure 9-23 Extended Description (Chapter 9)
351. Figure 9-24 Extended Description (Chapter 9)
352. Figure 9-25 Extended Description (Chapter 9)
353. Figure 9-26 Extended Description (Chapter 9)
354. Figure 9-27 Extended Description (Chapter 9)
355. Figure 9-28 Extended Description (Chapter 9)
356. A Ladder Diagram Extended Description (Chapter 10)
357. Figure 10-1 Extended Description (Chapter 10)
358. Figure 10-2(a-b) Extended Description (Chapter 10)
359. Figure 10-3 Extended Description (Chapter 10)
360. Figure 10-4 Extended Description (Chapter 10)
361. Figure 10-5 Extended Description (Chapter 10)
362. Figure 10-6 Extended Description (Chapter 10)
363. Figure 10-7 Extended Description (Chapter 10)
364. Figure 10-8 Extended Description (Chapter 10)
365. Figure 10-9 Extended Description (Chapter 10)
366. Figure 10-10 Extended Description (Chapter 10)
367. Figure 10-11 Extended Description (Chapter 10)
368. Figure 10-12 Extended Description (Chapter 10)
369. Figure 10-13 Extended Description (Chapter 10)
370. Figure 10-14 Extended Description (Chapter 10)
371. Figure 10-15 Extended Description (Chapter 10)
372. Figure 10-16 Extended Description (Chapter 10)
373. Figure 10-17 Extended Description (Chapter 10)
374. Figure 10-19 Extended Description (Chapter 10)
375. Figure 10-20 Extended Description (Chapter 10)
376. Figure 10-21 Extended Description (Chapter 10)
377. Figure 10-22 Extended Description (Chapter 10)
378. Figure 10-23 Extended Description (Chapter 10)
379. Figure 10-24 Extended Description (Chapter 10)
380. Figure 10-25 Extended Description (Chapter 10)
381. Figure 10-26 Extended Description (Chapter 10)
382. Figure 10-27 Extended Description (Chapter 10)
383. Figure 10-28 Extended Description (Chapter 10)
384. Figure 10-29 Extended Description (Chapter 10)
385. Figure 10-31 Extended Description (Chapter 10)
386. Figure 10-32 Extended Description (Chapter 10)
387. Figure 10-33 Extended Description (Chapter 10)
388. Figure 10-34 Extended Description (Chapter 10)
389. Figure 10-35 Extended Description (Chapter 10)
390. Figure 10-36 Extended Description (Chapter 10)
391. Figure 10-37 Extended Description (Chapter 10)

392. The Binary Number Extended Description (Chapter 10)
393. The Binary Number Extended Description (Chapter 10)
394. Figure 10-38 Extended Description (Chapter 10)
395. Figure 10-39 Extended Description (Chapter 10)
396. Figure 10-40 Extended Description (Chapter 10)
397. Figure 10-41 Extended Description (Chapter 10)
398. Figure 10-42 Extended Description (Chapter 10)
399. Figure 10-43 Extended Description (Chapter 10)
400. Figure 10-44 Extended Description (Chapter 10)
401. Figure 10-45 Extended Description (Chapter 10)
402. Figure 10-46(a-c) Extended Description (Chapter 10)
403. Figure 10-47 Extended Description (Chapter 10)
404. Figure 10-48 Extended Description (Chapter 10)
405. Figure 10-49 Extended Description (Chapter 10)
406. Figure 10-50 Extended Description (Chapter 10)
407. A Ladder Diagram Extended Description (Chapter 11)
408. Figure 11-1 Extended Description (Chapter 11)
409. Figure 11-3 Extended Description (Chapter 11)
410. Figure 11-4 Extended Description (Chapter 11)
411. Figure 11-5 Extended Description (Chapter 11)
412. Figure 11-6 Extended Description (Chapter 11)
413. Figure 11-7 Extended Description (Chapter 11)
414. Figure 11-8 Extended Description (Chapter 11)
415. Figure 11-9 Extended Description (Chapter 11)
416. Figure 11-10 Extended Description (Chapter 11)
417. Figure 11-11 Extended Description (Chapter 11)
418. Figure 11-12 Extended Description (Chapter 11)
419. Figure 11-13 Extended Description (Chapter 11)
420. Figure 11-14 Extended Description (Chapter 11)
421. Figure 11-15 Extended Description (Chapter 11)
422. Figure 11-16 Extended Description (Chapter 11)
423. Figure 11-17 Extended Description (Chapter 11)
424. Figure 11-18 Extended Description (Chapter 11)
425. Figure 11-19 Extended Description (Chapter 11)
426. Figure 11-20 Extended Description (Chapter 11)
427. Figure 11-21 Extended Description (Chapter 11)
428. Figure 11-22 Extended Description (Chapter 11)
429. Figure 11-23 Extended Description (Chapter 11)
430. Figure 11-24 Extended Description (Chapter 11)
431. Figure 11-25 Extended Description (Chapter 11)
432. Figure 11-26 Extended Description (Chapter 11)
433. Figure 11-27 Extended Description (Chapter 11)
434. Figure 11-35 Extended Description (Chapter 11)
435. Figure 11-36 Extended Description (Chapter 11)
436. Figure 11-37 Extended Description (Chapter 11)
437. Figure 11-38 Extended Description (Chapter 11)
438. Figure 11-39 Extended Description (Chapter 11)
439. Figure 11-40 Extended Description (Chapter 11)
440. Figure 11-41 Extended Description (Chapter 11)
441. Figure 12-1 Extended Description (Chapter 12)

442. Figure 12-2 Extended Description (Chapter 12)
443. Figure 12-3 Extended Description (Chapter 12)
444. Figure 12-5 Extended Description (Chapter 12)
445. Figure 12-6 Extended Description (Chapter 12)
446. Figure 12-7 Extended Description (Chapter 12)
447. Figure 12-8 Extended Description (Chapter 12)
448. Figure 12-9 Extended Description (Chapter 12)
449. Figure 12-10 Extended Description (Chapter 12)
450. Figure 12-11 Extended Description (Chapter 12)
451. Figure 12-12 Extended Description (Chapter 12)
452. Figure 12-13 Extended Description (Chapter 12)
453. Figure 12-14 Extended Description (Chapter 12)
454. Figure 12-15 Extended Description (Chapter 12)
455. Figure 12-16 Extended Description (Chapter 12)
456. Figure 12-17 Extended Description (Chapter 12)
457. Figure 12-18 Extended Description (Chapter 12)
458. Figure 12-19 Extended Description (Chapter 12)
459. Figure 12-20 Extended Description (Chapter 12)
460. Figure 12-21 Extended Description (Chapter 12)
461. Figure 12-22 Extended Description (Chapter 12)
462. Figure 12-23 Extended Description (Chapter 12)
463. Figure 12-24 Extended Description (Chapter 12)
464. Figure 12-25 Extended Description (Chapter 12)
465. Figure 12-26 Extended Description (Chapter 12)
466. Figure 12-28 Extended Description (Chapter 12)
467. Figure 12-29 Extended Description (Chapter 12)
468. Figure 12-30 Extended Description (Chapter 12)
469. Figure 12-31 Extended Description (Chapter 12)
470. Figure 12-32 Extended Description (Chapter 12)
471. Figure 12-34 Extended Description (Chapter 12)
472. Figure 12-35 Extended Description (Chapter 12)
473. Figure 12-36 Extended Description (Chapter 12)
474. Figure 12-37 Extended Description (Chapter 12)
475. Figure 12-38 Extended Description (Chapter 12)
476. Figure 12-39 Extended Description (Chapter 12)
477. Figure 12-40 Extended Description (Chapter 12)
478. Figure 12-41 Extended Description (Chapter 12)
479. Figure 12-42 Extended Description (Chapter 12)
480. Figure 12-43 Extended Description (Chapter 12)
481. Figure 12-44 Extended Description (Chapter 12)
482. An Illustration Shows Extended Description (Chapter 13)
483. Figure 13-1 Extended Description (Chapter 13)
484. Figure 13-3 Extended Description (Chapter 13)
485. Figure 13-4 Extended Description (Chapter 13)
486. Figure 13-6 Extended Description (Chapter 13)
487. Figure 13-7 Extended Description (Chapter 13)
488. Figure 13-8 Extended Description (Chapter 13)
489. Figure 13-9 Extended Description (Chapter 13)
490. Figure 13-10 Extended Description (Chapter 13)
491. Figure 13-11 Extended Description (Chapter 13)

492. Figure 13-12 Extended Description (Chapter 13)
493. Figure 13-13 Extended Description (Chapter 13)
494. Figure 13-14(a-b) Extended Description (Chapter 13)
495. Figure 13-15 Extended Description (Chapter 13)
496. Figure 13-16 Extended Description (Chapter 13)
497. Figure 13-17 Extended Description (Chapter 13)
498. Figure 13-18 Extended Description (Chapter 13)
499. Figure 13-19 Extended Description (Chapter 13)
500. Figure 13-20 Extended Description (Chapter 13)
501. Figure 13-21 Extended Description (Chapter 13)
502. Figure 13-22 Extended Description (Chapter 13)
503. Figure 13-23 Extended Description (Chapter 13)
504. Figure 13-24 Extended Description (Chapter 13)
505. Figure 13-25 Extended Description (Chapter 13)
506. Figure 13-26 Extended Description (Chapter 13)
507. Figure 13-27 Extended Description (Chapter 13)
508. Figure 13-28 Extended Description (Chapter 13)
509. Figure 13-29 Extended Description (Chapter 13)
510. Figure 13-30 Extended Description (Chapter 13)
511. Figure 13-31 Extended Description (Chapter 13)
512. Figure 13-32 Extended Description (Chapter 13)
513. Figure 13-33 Extended Description (Chapter 13)
514. Figure 13-34 Extended Description (Chapter 13)
515. Figure 13-35 Extended Description (Chapter 13)
516. Figure 13-36 Extended Description (Chapter 13)
517. Figure 14-1 Extended Description (Chapter 14)
518. Figure 14-2 Extended Description (Chapter 14)
519. Figure 14-3 Extended Description (Chapter 14)
520. Figure 14-4 Extended Description (Chapter 14)
521. Figure 14-5 Extended Description (Chapter 14)
522. Figure 14-8 Extended Description (Chapter 14)
523. Figure 14-9 Extended Description (Chapter 14)
524. Figure 14-10 Extended Description (Chapter 14)
525. Figure 14-11 Extended Description (Chapter 14)
526. Figure 14-12 Extended Description (Chapter 14)
527. Figure 14-13 Extended Description (Chapter 14)
528. Figure 14-14 Extended Description (Chapter 14)
529. Figure 14-15 Extended Description (Chapter 14)
530. Figure 14-16 Extended Description (Chapter 14)
531. Figure 14-17 Extended Description (Chapter 14)
532. Figure 14-18 Extended Description (Chapter 14)
533. Figure 14-19 Extended Description (Chapter 14)
534. Figure 14-21 Extended Description (Chapter 14)
535. Figure 14-22 Extended Description (Chapter 14)
536. Figure 14-23 Extended Description (Chapter 14)
537. Figure 14-25 Extended Description (Chapter 14)
538. Figure 14-26 Extended Description (Chapter 14)
539. Figure 14-28 Extended Description (Chapter 14)
540. Figure 14-29 Extended Description (Chapter 14)
541. Figure 14-30 Extended Description (Chapter 14)

542. Figure 14-31 Extended Description (Chapter 14)
543. Figure 14-32 Extended Description (Chapter 14)
544. Figure 14-33 Extended Description (Chapter 14)
545. Figure 14-34 Extended Description (Chapter 14)
546. Figure 14-35 Extended Description (Chapter 14)
547. Figure 14-36 Extended Description (Chapter 14)
548. Figure 14-37 Extended Description (Chapter 14)
549. Figure 14-38 Extended Description (Chapter 14)
550. Figure 14-39 Extended Description (Chapter 14)
551. Figure 14-40 Extended Description (Chapter 14)
552. Figure 14-41 Extended Description (Chapter 14)
553. Figure 14-42 Extended Description (Chapter 14)
554. Figure 14-43 Extended Description (Chapter 14)
555. Figure 14-44 Extended Description (Chapter 14)
556. Figure 14-45 Extended Description (Chapter 14)
557. Figure 14-46 Extended Description (Chapter 14)
558. Figure 14-47(a-b) Extended Description (Chapter 14)
559. Figure 14-48 Extended Description (Chapter 14)
560. Figure 14-49 Extended Description (Chapter 14)
561. Figure 14-50 Extended Description (Chapter 14)
562. Figure 14-51 Extended Description (Chapter 14)
563. Figure 15-2 Extended Description (Chapter 15)
564. Figure 15-3 Extended Description (Chapter 15)
565. Figure 15-4 Extended Description (Chapter 15)
566. Figure 15-5 Extended Description (Chapter 15)
567. Figure 15-6 Extended Description (Chapter 15)
568. Figure 15-7 Extended Description (Chapter 15)
569. Figure 15-8 Extended Description (Chapter 15)
570. Figure 15-10 Extended Description (Chapter 15)
571. Figure 15-11 Extended Description (Chapter 15)
572. Figure 15-12 Extended Description (Chapter 15)
573. Figure 15-13 Extended Description (Chapter 15)
574. Figure 15-14 Extended Description (Chapter 15)
575. Figure 15-15 Extended Description (Chapter 15)
576. Figure 15-16 Extended Description (Chapter 15)
577. Figure 15-17 Extended Description (Chapter 15)
578. Figure 15-18 Extended Description (Chapter 15)
579. Figure 15-19 Extended Description (Chapter 15)
580. Figure 15-20 Extended Description (Chapter 15)
581. Figure 15-21 Extended Description (Chapter 15)
582. Figure 15-22 Extended Description (Chapter 15)
583. Figure 15-23 Extended Description (Chapter 15)
584. Figure 15-24 Extended Description (Chapter 15)
585. Figure 15-26 Extended Description (Chapter 15)
586. Figure 15-27 Extended Description (Chapter 15)
587. Figure 15-28 Extended Description (Chapter 15)
588. Figure 15-29 Extended Description (Chapter 15)
589. Figure 15-30 Extended Description (Chapter 15)
590. Figure 15-31 Extended Description (Chapter 15)
591. Figure 15-32 Extended Description (Chapter 15)

592. Figure 15-33 Extended Description (Chapter 15)
593. Figure 15-34 Extended Description (Chapter 15)
594. Figure 15-35 Extended Description (Chapter 15)
595. Figure 15-36 Extended Description (Chapter 15)
596. Figure 15-37 Extended Description (Chapter 15)
597. Figure 15-38 Extended Description (Chapter 15)
598. Figure 15-39 Extended Description (Chapter 15)
599. Figure 15-40 Extended Description (Chapter 15)
600. Figure 15-41 Extended Description (Chapter 15)
601. Figure 15-42 Extended Description (Chapter 15)
602. Figure 15-43 Extended Description (Chapter 15)
603. Figure 15-44 Extended Description (Chapter 15)
604. Figure 15-45 Extended Description (Chapter 15)
605. Figure 15-46 Extended Description (Chapter 15)
606. Figure 15-47 Extended Description (Chapter 15)
607. Figure 15-48 Extended Description (Chapter 15)
608. Figure 15-49 Extended Description (Chapter 15)
609. Figure 15-50 Extended Description (Chapter 15)
610. Figure 15-51 Extended Description (Chapter 15)
611. Figure 15-52 Extended Description (Chapter 15)
612. Figure 15-53 Extended Description (Chapter 15)
613. Figure 15-54 Extended Description (Chapter 15)
614. Figure 15-55 Extended Description (Chapter 15)
615. Figure 15-56 Extended Description (Chapter 15)
616. Figure 15-57 Extended Description (Chapter 15)
617. Figure 15-58 Extended Description (Chapter 15)
618. Figure 15-59 Extended Description (Chapter 15)
619. Figure 15-60 Extended Description (Chapter 15)
620. Figure 15-61 Extended Description (Chapter 15)
621. Figure 15-62 Extended Description (Chapter 15)
622. Figure 15-63 Extended Description (Chapter 15)
623. Figure 15-64 Extended Description (Chapter 15)
624. Figure 15-65 Extended Description (Chapter 15)
625. Figure 15-66 Extended Description (Chapter 15)
626. Figure 15-67 Extended Description (Chapter 15)
627. Figure 15-68 Extended Description (Chapter 15)
628. Figure 15-69 Extended Description (Chapter 15)
629. Figure 15-70 Extended Description (Chapter 15)
630. Figure 15-71 Extended Description (Chapter 15)
631. Figure 15-72 Extended Description (Chapter 15)
632. Figure 15-73 Extended Description (Chapter 15)
633. Figure 15-74 Extended Description (Chapter 15)
634. Figure 15-75 Extended Description (Chapter 15)
635. Figure 15-76 Extended Description (Chapter 15)
636. Figure 15-77 Extended Description (Chapter 15)
637. Figure 15-78 Extended Description (Chapter 15)
638. Figure 15-79 Extended Description (Chapter 15)
639. Figure 15-80 Extended Description (Chapter 15)
640. Figure 15-81 Extended Description (Chapter 15)
641. Figure 15-82 Extended Description (Chapter 15)

642. Figure 15-83 Extended Description (Chapter 15)
643. Figure 15-84 Extended Description (Chapter 15)
644. Figure 15-85 Extended Description (Chapter 15)
645. Figure 15-86 Extended Description (Chapter 15)
646. Figure 15-87 Extended Description (Chapter 15)
647. Figure 15-88 Extended Description (Chapter 15)
648. Figure 15-89 Extended Description (Chapter 15)
649. Figure 15-90 Extended Description (Chapter 15)
650. Figure 15-91 Extended Description (Chapter 15)
651. Figure 15-92 Extended Description (Chapter 15)
652. Figure 15-93 Extended Description (Chapter 15)
653. Figure 15-94 Extended Description (Chapter 15)
654. Figure 15-95 Extended Description (Chapter 15)
655. Figure 15-96 Extended Description (Chapter 15)
656. Figure 15-97 Extended Description (Chapter 15)
657. Figure 15-98 Extended Description (Chapter 15)
658. Figure 15-99 Extended Description (Chapter 15)
659. Figure 15-100 Extended Description (Chapter 15)
660. Figure 15-101 Extended Description (Chapter 15)
661. Figure 15-102 Extended Description (Chapter 15)
662. Figure 15-103 Extended Description (Chapter 15)
663. Figure 15-104 Extended Description (Chapter 15)
664. Figure 15-105 Extended Description (Chapter 15)
665. Figure 15-106 Extended Description (Chapter 15)
666. Figure 15-107 Extended Description (Chapter 15)
667. Figure 15-108 Extended Description (Chapter 15)
668. Figure 15-109 Extended Description (Chapter 15)
669. Figure 15-111 Extended Description (Chapter 15)
670. Figure 15-112 Extended Description (Chapter 15)
671. Figure 15-113 Extended Description (Chapter 15)
672. Figure 15-114 Extended Description (Chapter 15)
673. Figure 15-115 Extended Description (Chapter 15)
674. Figure 15-116 Extended Description (Chapter 15)
675. Figure 15-117 Extended Description (Chapter 15)
676. Figure 15-118 Extended Description (Chapter 15)
677. Figure 15-119 Extended Description (Chapter 15)
678. Figure 15-120 Extended Description (Chapter 15)
679. Figure 15-121 Extended Description (Chapter 15)

Page List

1. i
2. ii
3. iii
4. iv
5. v
6. vi
7. vii
8. viii

9. ix
10. x
11. xi
12. xii
13. xiii
14. xiv
15. xv
16. xvi
17. xvii
18. xviii
19. 1
20. 2
21. 3
22. 4
23. 5
24. 6
25. 7
26. 8
27. 9
28. 10
29. 11
30. 12
31. 13
32. 14
33. 15
34. 16
35. 17
36. 18
37. 19
38. 20
39. 21
40. 22
41. 23
42. 24
43. 25
44. 26
45. 27
46. 28
47. 29
48. 30
49. 31
50. 32
51. 33
52. 34
53. 35
54. 36
55. 37
56. 38
57. 39
58. 40

59. 41
60. 42
61. 43
62. 44
63. 45
64. 46
65. 47
66. 48
67. 49
68. 50
69. 51
70. 52
71. 53
72. 54
73. 55
74. 56
75. 57
76. 58
77. 59
78. 60
79. 61
80. 62
81. 63
82. 64
83. 65
84. 66
85. 67
86. 68
87. 69
88. 70
89. 71
90. 72
91. 73
92. 74
93. 75
94. 76
95. 77
96. 78
97. 79
98. 80
99. 81
100. 82
101. 83
102. 84
103. 85
104. 86
105. 87
106. 88
107. 89
108. 90

109. 91
110. 92
111. 93
112. 94
113. 95
114. 96
115. 97
116. 98
117. 99
118. 100
119. 101
120. 102
121. 103
122. 104
123. 105
124. 106
125. 107
126. 108
127. 109
128. 110
129. 111
130. 112
131. 113
132. 114
133. 115
134. 116
135. 117
136. 118
137. 119
138. 120
139. 121
140. 122
141. 123
142. 124
143. 125
144. 126
145. 127
146. 128
147. 129
148. 130
149. 131
150. 132
151. 133
152. 134
153. 135
154. 136
155. 137
156. 138
157. 139
158. 140

159. 141
160. 142
161. 143
162. 144
163. 145
164. 146
165. 147
166. 148
167. 149
168. 150
169. 151
170. 152
171. 153
172. 154
173. 155
174. 156
175. 157
176. 158
177. 159
178. 160
179. 161
180. 162
181. 163
182. 164
183. 165
184. 166
185. 167
186. 168
187. 169
188. 170
189. 171
190. 172
191. 173
192. 174
193. 175
194. 176
195. 177
196. 178
197. 179
198. 180
199. 181
200. 182
201. 183
202. 184
203. 185
204. 186
205. 187
206. 188
207. 189
208. 190

209. 191
210. 192
211. 193
212. 194
213. 195
214. 196
215. 197
216. 198
217. 199
218. 200
219. 201
220. 202
221. 203
222. 204
223. 205
224. 206
225. 207
226. 208
227. 209
228. 210
229. 211
230. 212
231. 213
232. 214
233. 215
234. 216
235. 217
236. 218
237. 219
238. 220
239. 221
240. 222
241. 223
242. 224
243. 225
244. 226
245. 227
246. 228
247. 229
248. 230
249. 231
250. 232
251. 233
252. 234
253. 235
254. 236
255. 237
256. 238
257. 239
258. 240

259. 241
260. 242
261. 243
262. 244
263. 245
264. 246
265. 247
266. 248
267. 249
268. 250
269. 251
270. 252
271. 253
272. 254
273. 255
274. 256
275. 257
276. 258
277. 259
278. 260
279. 261
280. 262
281. 263
282. 264
283. 265
284. 266
285. 267
286. 268
287. 269
288. 270
289. 271
290. 272
291. 273
292. 274
293. 275
294. 276
295. 277
296. 278
297. 279
298. 280
299. 281
300. 282
301. 283
302. 284
303. 285
304. 286
305. 287
306. 288
307. 289
308. 290

309. 291
310. 292
311. 293
312. 294
313. 295
314. 296
315. 297
316. 298
317. 299
318. 300
319. 301
320. 302
321. 303
322. 304
323. 305
324. 306
325. 307
326. 308
327. 309
328. 310
329. 311
330. 312
331. 313
332. 314
333. 315
334. 316
335. 317
336. 318
337. 319
338. 320
339. 321
340. 322
341. 323
342. 324
343. 325
344. 326
345. 327
346. 328
347. 329
348. 330
349. 331
350. 332
351. 333
352. 334
353. 335
354. 336
355. 337
356. 338
357. 339
358. 340

359. 341
360. 342
361. 343
362. 344
363. 345
364. 346
365. 347
366. 348
367. 349
368. 350
369. 351
370. 352
371. 353
372. 354
373. 355
374. 356
375. 357
376. 358
377. 359
378. 360
379. 361
380. 362
381. 363
382. 364
383. 365
384. 366
385. 367
386. 368
387. 369
388. 370
389. 371
390. 372
391. 373
392. 374
393. 375
394. 376
395. 377
396. 378
397. 379
398. 380
399. 381
400. 382
401. 383
402. 384
403. 385
404. 386
405. 387
406. 388
407. 389
408. 390

409. 391
410. 392
411. 393
412. 394
413. 395
414. 396
415. 397
416. 398
417. 399
418. 400
419. 401
420. 402
421. 403
422. 404
423. 405
424. 406
425. 407
426. 408
427. 409
428. 410
429. 411
430. 412
431. 413
432. 414
433. 415
434. 416
435. 417
436. 418
437. 419

Guide

1. Cover Page
2. Title Page
3. Copyright Information
4. About the Author
5. Contents
6. Preface
7. Chapter Updates to Note
8. Acknowledgments
9. Chapter Objectives
10. Ancillaries that Work
11. Digital Resources
12. Connect
13. Chapter 1 - Start of Content
14. Glossary
15. Index

OceanofPDF.com

Remarks

OceanofPDF.com

Glossary

2’s complement
: A numbering system used to express positive and negative binary numbers.

Accumulated value
: The number of elapsed timed intervals or counted events.

Actuator
: An output device normally connected to an output module. Examples are an air valve and cylinder.

Address
: A code that indicates the location of data to be used by a program, or the location of additional program instructions.

Alias tag
: References a memory location that has been defined by another tag.

American National Standard Code for Information Interchange (ASCII)
: An 8-bit (7 bits plus parity) code that represents all characters of a standard typewriter keyboard, both uppercase and lowercase, as well as a group of special characters used for control purposes.

Analog device
: Apparatus that measures continuous information (e.g., voltage or current). The measured analog signal has an infinite number of possible values. The only limitation on resolution is the accuracy of the measuring device.

Analog input module
: An input circuit that employs an analog-to-digital converter to convert an analog value, measured by an analog measuring device, to a digital value that can be used by the processor.

Analog output module
: An output circuit that employs a digital-to-analog converter to convert a digital value, sent from the processor, to an analog value that will control a connected analog device.

Analog signal
: Signal having the characteristic of being continuous and changing smoothly over a given range rather than switching suddenly between certain levels, as with discrete signals.

AND (logic)
: A Boolean operation that yields a logic 1 output if all inputs are 1, and a logic 0 if any input is 0.

Array
: A group of data, of the same type, under a common name.

Asynchronous
: Recurrent or repeated operations that occur in unrelated patterns over time.

Automatic control
: A process in which the output is kept at a desired level by using feedback from the output to control the input.

Backplane
: A printed circuit board, located in the back of a chassis, that contains a data bus, power bus, and mating connectors for modules to be inserted in the chassis.

Base tag
: A definition of the memory location at which a data element is stored.

Baud

A unit of signaling speed equal to the number of discrete conditions or signal events per second; often defined as the number of binary digits transmitted per second.

Binary

A number system using 2 as a base. The binary number system requires only two digits, zero (0) and one (1), to express any alphanumeric quantity desired by the user.

Binary word

A related group of 1s and 0s that has meaning assigned by position or by numerical value in the binary system of numbers.

Binary-coded decimal (BCD)

A system of numbering that expresses each individual decimal digit (0–9) of a number as a series of 4-bit binary notations. The binary-coded decimal system is often referred to as *8421 code.*

Bit

An abbreviated term for the words *binary digit.* The bit is the smallest unit of information in the binary numbering system. It represents a decision between one of two possible and equally likely values or states. It is often used to represent an off or on state as well as a true or false condition.

Block format

Format that uses a box shape to display instructions.

BOOL

A data type that stores the state of a single bit, where 0 equals off and 1 equals on.

Boolean algebra

A mathematical shorthand notation that expresses logic functions, such as AND, OR, EXCLUSIVE OR, NAND, NOR, and NOT.

Bus topology

A network configuration in which all stations are connected in parallel with the communication medium and all stations can receive information from any other station on the network.

Cascading

In the programming of timers and counters, a technique used to extend the timing or counting range beyond what would normally be available. This technique involves the driving of one timer or counter instruction from the output of another, similar instruction.

Central processing unit (CPU)

The electronic circuitry that controls all the data activity of the PLC, performs calculations, and makes decisions with its operation controlled by a sequence of instructions. The central processing unit is also referred to as the *processor* or the *CPU.*

Clear

An instruction or a sequence of instructions that removes all current information from a programmable logic controller’s memory.

Closed loop

A control system that uses feedback from the process to maintain outputs at a desired level.

Communication module

Allows the user to connect the PLC to high-speed local networks that may differ from the network communication provided with the PLC.

Compare

An instruction that compares the contents of two designated data memory locations of a programmable logic controller for equality or inequality.

Contact histogram

An instruction sequence that monitors a designated memory bit or a designated input or output point for a change of state. A listing is generated by the instruction sequence that displays how

quickly the monitored point is changing state.

Contactor
: A special-purpose relay designed to establish and interrupt the power flow of high-current electric circuits.

Counter
: An electromechanical device in relay-based control systems that counts numbers of events for the purpose of controlling other devices based on the current number of counts recorded; a programmable logic controller instruction that performs the functions of its electromechanical counterpart.

Cross reference
: In ladder diagrams, letters or numbers to the right of coils or functions. The letters or numbers indicate where on other ladder lines contacts of the coil or function are located.

Current sinking
: Refers to an output device (typically an NPN transistor) that allows current flow from the load through the output to ground.

Current sourcing
: Output device (typically a PNP transistor) that allows current flow from the output through the load and then to ground.

Data file
: A group of data memory words acted on as a group rather than singly.

Data latching
: A technique used to read the value of the input data that will be operated on by the instructions with a function block.

Data manipulation
: The process of exchanging, altering, or moving data within a programmable logic controller or between programmable logic controllers.

Data server
: Used to store, send, and receive data.

Data transfer
: The process of moving information from one location to another, in other words, from register to register, from device to device, and so forth.

Decimal number system
: A number system that uses ten numeral digits (decimal digits): 0, 1, 2, 3, 4, 5, 6, 7, 8, 9. Each digit position has a place value of 1, 10, 100, 1000, and so on, beginning with the least significant (rightmost) digit; base 10.

Decrement
: The act of reducing the contents of a storage location or value in varying increments.

Determinism
: The ability to reliably predict when data will be delivered.

Diagnostics
: The detection and isolation of an error or malfunction.

Digital device
: One that processes discrete electric signals.

Digital-to-analog converter
: An electrical circuit that converts binary bits to a representative, continuous, analog signal.

DINT
: A data type that stores a 32-bit (4-byte) signed integer value.

DIP switch
: A group of small, in-line on-off switches. From *d*ual *i*n-line *p*ackage.

Discrete I/O

A group of input and/or output modules that operate with on/off signals, as contrasted to analog modules that operate with continuously variable signals.

Divide

A programmable logic controller instruction that performs a numerical division of one number by another.

Done bit (DN)

Bit that is set to 1 when the instruction has completed its task, such as reaching its preset value.

Double precision

The system of using two addresses or registers to display a number too large for one address or register; allows the display of more significant figures because twice as many bits are used.

Down-counter

A counter that starts from a specified number and increments down to zero.

Edit

The act of modifying a programmable logic controller program to eliminate mistakes and/or simplify or change system operation.

Electrically erasable programmable read-only memory (EEPROM)

A type of programmable read-only memory that is programmed and erased by electrical pulses.

Electromagnetic interference (EMI)

A phenomenon responsible for noise in electric circuits.

Enclosure

A steel box with a removable cover or hinged door used to house electric equipment.

Encoder

A rotary or linear device that transmits position information; a device that transmits a fixed number of pulses for each revolution.

Erasable programmable read-only memory (EPROM)

A programmable read-only memory that can be erased with ultraviolet light, then reprogrammed with electrical pulses.

Examine if closed (XIC)

Refers to a normally open contact instruction in a logic ladder program. An examine if closed instruction is true if its addressed bit is on (1). It is false if the bit is off (0).

Examine if open (XIO)

Refers to a normally closed contact instruction in a logic ladder program. An examine if open instruction is true if its addressed bit is off (0). It is false if the bit is on (1).

Exclusive-OR gate

A logic device requiring one or the other, but not both, of its inputs to be satisfied before activating its output.

False

As related to programmable logic controller instructions, a disabling logic state; a bit that is off or zero.

Fault-routine file

A special subroutine that, if assigned, executes when the processor has a major fault.

Fiber optic cable

Transmits information via light pulses down optical fibers.

File

A formatted block of data treated as a unit.

Fixed I/O

Input/output terminals on a programmable logic controller that are built into the unit and are not changeable. A fixed I/O PLC has no removable modules.

Floating-point data file
: Used to store real numerical values that cannot be stored in an integer file.

Full duplex
: A mode of data communications in which data may be transmitted and received simultaneously.

Function block
: Rectangular block with inputs entering from the left and outputs exiting on the right.

Function block diagram (FBD)
: Graphical language where the basic programming elements appear as blocks.

Gateway
: A device or pair of devices that connects two or more communication networks. This device may act as a host to each network and may transfer messages between the networks by translating their protocols.

Graphics library
: An aid in creating computer graphics for an HMI display screen.

Gray code
: A binary coding scheme that allows only 1 bit in the data word to change state at each increment of the code sequence.

Ground
: A conducting connection between an electric circuit or equipment chassis and the earth ground.

Ground loop
: A condition in which two or more electrical paths exist within a ground line.

Hardware
: The mechanical, electric, and electronic devices that make up a programmable logic controller and its application.

Hardwired
: The physical interconnection of electric and electronic components with wire.

High-speed counter encoder module
: A module that enables you to count and encode faster than you could with a regular control program written on a PLC in which the control program’s execution is too slow.

HMI terminal
: An interactive screen that provides information, data, and control of a given process to a human user operating the HMI screen.

Human machine interface (HMI)
: Graphical display hardware in which machine status, alarms, messages, diagnostics, and data entry are available to the operator in graphical display format.

Image table
: An area in programmable logic controller memory dedicated to input/output data. Ones and zeros (1s and 0s) represent on and off conditions, respectively. During every input/output scan, each input controls a bit in the input image table file; each output is controlled by a bit in the output image table file.

Immediate input instruction
: A programmable logic controller instruction that temporarily halts the user program scan so that the processor can update the input image table file with the current status of one or more user-specified input points.

Immediate output instruction
: A programmable logic controller instruction that temporarily halts the user program scan so that the current status of one or more user-specified output points can be updated to current output image table file status by the processor.

Increment
: The act of increasing the contents of a storage location or value in varying amounts.

Input
: Information transmitted from a peripheral device to the input module and then to the data table.

Input devices
: Devices such as limit switches, pressure switches, pushbuttons, and analog and/or digital devices that supply data to a programmable logic controller.

Input/output (I/O) address
: A unique number assigned to each input and output. The address number is used when programming, monitoring, or modifying a specific input or output.

Instruction set
: The set of general-purpose instructions available with a given controller. In general, different machines have different instruction sets.

INT
: Two-byte integer.

Integer
: A positive or negative whole number.

Inverter
: The digital circuit that performs inversion.

Label instruction
: A programmable logic controller instruction that assigns an alphanumeric designation to a particular location in a program. This location is used as the target of a jump, skip, or jump to subroutine instruction.

Ladder diagram
: An industry standard for representing relay logic control systems. The diagram resembles a ladder because the vertical supports of the ladder appear as power feed and return buses and the horizontal rungs of the ladder appear as series and/or parallel circuits connected across the power lines.

Language
: A set of symbols and rules for representing and communicating information among people or between people and machines; the method used to instruct a programmable device to perform various operations.

Latching relay
: A relay that maintains a given position by mechanical or electrical means until released mechanically or electrically.

Leakage
: The small amount of current that flows in a semiconductor device when it is in the off state.

Limit switch
: An electric switch actuated by some part and/or motion of a machine or equipment.

Limit test
: A test that determines if a value is inside or outside a specified range.

Logic gate
: A device in which the output state is based on the state of its inputs.

Master control relay (MCR)
: A mandatory hardwired relay that can be de-energized by any series-connected emergency stop switch. Whenever the master control relay is de-energized, its contacts open to de-energize all application input and output devices.

Memory

That part of the programmable logic controller in which data and instructions are stored either temporarily or semi-permanently. The control program is stored in memory.

Metal-oxide varistor (MOV)
: Used for suppressing electrical power surges.

Mode
: A term used to refer to the selected operating method, such as automatic, manual, TEST, PROGRAM, or diagnostic.

Motor starter
: A special relay designed to provide power to motors; it has both a contactor relay and an overload relay connected in series and prewired so that, if the overload operates, the contactor is de-energized.

Move instruction
: A programmable logic controller instruction that moves data from one location to another. Although a move instruction typically places the data in a new location, the original data still reside in their original location.

Multiply instruction
: A programmable logic controller instruction that provides for the mathematical multiplication of two numbers.

National Electrical Code (NEC)
: A set of regulations developed by the National Fire Protection Association that governs the construction and installation of electric wiring and electric devices. The National Electrical Code is recognized by many governmental bodies, and compliance is mandatory in much of the United States.

Nested branches
: A branch that begins or ends within another branch.

Node
: In hardware, a connection point on the network; in programming, the smallest possible increment in a ladder diagram.

NOR
: The logic gate that results in zero unless both inputs are zero.

Normally closed contact (NC)
: A contact that is conductive when its operating coil is not energized.

Normally open contact (NO)
: A contact that is nonconductive when its operating coil is not energized.

NOT
: A logical operation that yields a logic 1 at the output if a logic 0 is entered at the input, and a logic 0 at the output if a logic 1 is entered at the input. The NOT, also called the *inverter,* is normally used in conjunction with the AND and OR functions.

Octal number system
: A base eight numbering system that uses numbers 0–7, 10–17, 20–27, and so on. There are no 8s or 9s in the octal number system.

Off-delay timer
: An electromechanical relay with contacts that change state a predetermined time period after power is removed from its coil; on re-energization of the coil, the contacts return to their shelf state immediately; also, a programmable logic controller instruction that emulates the operation of the electromechanical off-delay relay.

Offline programming and/or offline editing
: A method of programmable logic controller programming and/or editing in which the operation of the processor is stopped and all output devices are switched off. Offline programming is the safest way to develop or edit a programmable logic controller program

since the entry of instructions does not affect operating hardware until the program can be verified for accuracy of entry.

On-delay timer
: An electromechanical relay with contacts that change state a predetermined time period after the coil is energized; the contacts return to their shelf state immediately on de-energization of the coil; also, a programmable logic controller instruction that emulates the operation of the electromechanical on-delay timer.

One-shot
: A programmed technique that sets a storage bit or output for only one program scan.

Online programming and/or online editing
: The ability of a processor and programming terminal to make joint user-directed additions, deletions, or changes to a user program while the processor is actively solving and executing the commands of the existing user program. Extreme care should be exercised when performing online programming to ensure that erroneous system operation does not result.

OPC UA
: An open platform communications united architecture used for transferring information between servers.

Open loop
: A system that has no feedback or auto correction.

OR
: A logical operation that yields a logic 1 output if one of any number of inputs is 1, and a logic 0 if all inputs are 0.

Output
: Information sent from the processor to a connected device via some interface. The information could be in the form of control data that will signal some device such as a motor to switch on or off or to vary the speed of a drive.

Output image table file
: A portion of a processor’s data memory reserved for the storage of output device statuses. A 1, on, or true state in an output image table file storage location is used to switch on the corresponding output point.

Overload relay
: A special-purpose relay designed so that its contacts transfer whenever its current exceeds a predetermined value. Overload relays are used with electric motors to prevent motor burnout due to mechanical overload.

Parallel transmission
: A computer operation in which two or more bits of information are transmitted simultaneously.

Peer-to-peer network
: A network in which nodes are given an equal chance of initiating and controlling communications.

Power supply
: The unit that supplies the necessary voltage and current to a system’s circuitry.

Preset value (PRE)
: In a counter instruction, the number of time intervals or events to be counted.

Pressure switch
: A switch activated at a specified pressure.

Program
: A sequence of instructions to be executed by the processor to control a machine or process.

Program files
: The area of processor memory in which the ladder logic programming is stored.

Program scan

One of three parts of the PLC scan. During the program scan, the CPU scans each rung of the user program.

Programmable controller

A computer that has been hardened to work in an industrial environment and is equipped with special I/O and a control programming language.

Project file

Contains all data associated with the PLC project. A project comprises five major pieces: help folder, controller folder, ladder folder, data folder, and data base folder.

Protocol

A formal definition of criteria for receiving and transmitting data through communications channels.

Proximity switch

An input device that senses the presence or absence of a target without physical contact.

Rack

A housing or framework used to hold assemblies; a plastic and/or metal assembly that supports input/output modules and provides a means of supplying power and signals to each input/output module or card.

Random-access memory (RAM)

A memory system that permits the random accessing of any storage location for the purpose of either storing (writing) or retrieving (reading) information. Random-access memory systems allow the data to be retrieved and stored at speeds independent of the storage locations being accessed.

Read-only memory (ROM)

A permanent memory structure in which data are placed at time of fabrication or by the user at a speed much slower than it will be read. Information entered in a read-only memory is usually not changed once it is entered.

Register

A memory word or area for the temporary storage of data used within mathematical, logical, or transfer functions.

Relay logic

A representation of the program or other logic in a form normally used for relays.

Remote input/output (I/O) system

Any input/output system that permits communication between the processor and input/output hardware over a coaxial or twin axial cable. Remote input/output systems permit the placement of input/output hardware at any distance from the processor.

Resolution

The smallest distinguishable increment into which a quantity is divided.

Retentive timer

An electromechanical relay that accumulates time whenever the device receives power and maintains the current time should power be removed from the device. Loss of power to the device after reaching its preset value does not affect the state of the contacts.

Routine

A series of instructions that perform a specific function or task.

RS-232

An Electronic Industries Association (EIA) standard for data transfer and communication for serial binary communication circuits.

RSLinx

A communications driver used to communicate, via EtherNet/IP, between the PLC Program and its HMI application.

Rung

A group of programmable logic controller instructions that controls an output or storage bit, or performs other control functions such as file moves, arithmetic, and/or sequencer instructions. A rung is represented as one section of a ladder logic diagram.

Runtime application
: Compiles all of the necessary application information into a single file required to run the application.

SCADA
: An acronym for supervisory control and data acquisition.

Scan time
: The time required to read all inputs, execute the control program, and update local and remote input and output statuses. Scan time is, in effect, the time required to activate an output controlled by programmed logic.

Sensor
: A device used to gather information by the conversion of a physical occurrence to an electric signal.

Sequencer
: A mechanical, electric, or electronic device that can be programmed so that a predetermined set of events occurs repeatedly.

Sequential control
: A process that dictates the correct order of events and allows one event to occur only after the completion of another.

Sequential Function Chart (SFC)
: Graphical language whose basic language elements are steps or states with associated actions and transitions with associated conditions used to move from the current state to the next.

Servo module
: The device whose feedback is used to accomplish closed-loop control. Though programmed through a PLC, once programmed it can control a device independently without interfering with the PLC's normal operation.

Set-point
: The value that the process value is to be held to by the automatic control function.

Shift register
: A PLC function capable of storing and shifting binary data.

Single-scan function
: A supervisory instruction that causes the control program to be executed for one scan, including input/output update. This troubleshooting function allows step-by-step inspection of occurrences while the machine is stopped.

SINT
: A data type that stores an 8-bit (1-byte) signed integer value.

Star topology
: A network architecture in which all network nodes are connected to a central device that routes the nodes' messages.

Station
: Any programmable logic controller, computer, or data terminal connected to, and communicating by means of, a data highway.

Status indicators
: LEDs that indicate the on-off status of an input or output point and are visible on the outside of the PLC.

Stepper-motor module
: Provides pulse trains to a stepper-motor translator that enables control of a stepper motor.

Structure

A data type that is composed from multiple data types.

Structured Text (ST)
: High-level, text-based language with commands that support a highly structured program development and the ability to evaluate complex mathematical expressions.

Subroutines
: Program files that are scanned only when called on by logic and can be used to break the program into smaller segments.

Subtract
: A programmable logic controller instruction that performs the mathematical subtraction of one number from another.

Suppression device
: A unit that attenuates the magnitude of electrical noise.

Synchronous shift register
: A shift register in which only one change of state occurs per control pulse.

Tag
: A text-based name for an area of the controller's memory where data are stored.

Tag database
: Used for storing, deleting, and editing of tags.

Task
: It holds the information necessary to schedule the program's execution and sets the execution priority for one or more programs.

Time base
: A unit of time generated by a microprocessor's clock circuit and used by PLC timer instructions. Typical time bases are 0.001, 0.01, 0.1, and 1.0 second.

Timer
: In relay-panel hardware, an electromechanical device that can be wired and preset to control the operating interval of other devices. In a programmable logic controller, a timer is internal to the *processor;* that is, it is controlled by a user-programmed instruction.

Token passing
: A technique in which tokens are circulated among nodes in a communication network.

Transducer
: A device used to convert physical parameters such as temperature, pressure, and weight into electric signals.

Transitional contact
: A contact that, depending on how it is programmed, will be on for one program scan every 0 to 1 transition, or every 1 to 0 transition, of the referenced coil.

True
: As related to programmable logic controller instructions, an on, enabled, or 1 state.

Truth table
: A table listing that shows the state of a given output as a function of all possible input combinations.

Unlatch instruction
: One-half of a programmable logic controller instruction pair that emulates the unlatching action of a latching relay. The unlatch instruction de-energizes a specified output point or internal coil until re-energized by a latch instruction. The output point or internal coil remains de-energized regardless of whether or not the unlatch instruction is energized.

Up-counter
: An event that starts from 0 and increments up to the preset value.

Watchdog timer

Monitors logic circuits controlling the processor. If the watchdog timer, which is reset every scan, ever times out, the processor is assumed to be faulty and is disconnected from the process.

OceanofPDF.com