GE Fanuc Automation

Programmable Control Products

GE Fanuc
Micro PLC
Programmer's Guide

GFK-0804B *April 1994*

Download from Www.Somanuals.com. All Manuals Search And Download.

Warnings, Cautions, and Notes as Used in this Publication

Warning

Warning notices are used in this publication to emphasize that hazardous voltages, currents, temperatures, or other conditions that could cause personal injury exist in this equipment or may be associated with its use.

In situations where inattention could cause either personal injury or damage to equipment, a Warning notice is used.

Caution

Caution notices are used where equipment might be damaged if care is not taken.

Note

Notes merely call attention to information that is especially significant to understanding and operating the equipment.

This document is based on information available at the time of its publication. While efforts have been made to be accurate, the information contained herein does not purport to cover all details or variations in hardware or software, nor to provide for every possible contingency in connection with installation, operation, or maintenance. Features may be described herein which are not present in all hardware and software systems. GE Fanuc Automation assumes no obligation of notice to holders of this document with respect to changes subsequently made.

GE Fanuc Automation makes no representation or warranty, expressed, implied, or statutory with respect to, and assumes no responsibility for the accuracy, completeness, sufficiency, or usefulness of the information contained herein. No warranties of merchantability or fitness for purpose shall apply.

The following are trademarks of GE Fanuc Automation North America, Inc.

Alarm Master	CIMSTAR	Helpmate	PROMACRO	Series Six
CIMPLICITY	Field Control	GEnet	Logicmaster	Series One
Series 90	CIMPLICITY 90–ADS	Genius	Modelmaster	Series Three
VuMaster	CIMPLICITY PowerTRAC	Genius PowerTRAC	ProLoop	Series Five
Workmaster				

Copyright 1994 GE Fanuc Automation North America, Inc.
All Rights Reserved

Download from Www.Somanuals.com. All Manuals Search And Download.

Preface

This book is the reference guide to programming the GE Fanuc Micro PLC.

Content of this Manual

Chapter 1. Programming for the Micro PLC: describes programming basics, the Micro PLC instruction set, programming devices and formats, memory types and addresses, constants and register values in a program, and special coils.

Chapter 2. Programming with the Programming Software: explains how to create and edit programs using the programming software.

Chapter 3. Programming with a Hand-held Programmer: explains how to create and edit programs using a Hand-held Programmer.

Chapter 4. The Micro PLC Instruction Set: explains in detail the instructions that can be incorporated into an application program for the Micro PLC.

Appendix A. Using Directories: gives advice on organizing the Micro PLC directory structure on your hard disk.

Appendix B. Micro PLC Protocol: the information in this appendix is for advanced users only. It explains programming to set up communications between the Micro PLC and a host system.

Appendix C. RTU Protocol: describes the Remote Terminal Unit (RTU) serial communications protocol, which can be used to provide communications between the Micro PLC or other remote device and a host computer.

Appendix D. Communications Using Windows DDE: describes an available software product that can be used to connect DDE-compliant Microsoft™ Windows programs with data in a Micro PLC.

Appendix E. Data Acquisition, Logging, and Display Program: describes the Data Acquisition, Logging, and Display Program software, which is provided on the Micro PLC software diskettes.

Appendix F. Programming Applications: describes simple programming for: a flip-flop, a powerup one-shot, cascading counters, and an industrial "starting circuit".

Related Publications

GE Fanuc Micro PLC User's Guide (GFK-0803): contains product specifications, installation instructions, and general information needed to set up and use a Micro PLC.

GE Fanuc Micro PLC Self-Teach Manual (GFK–0811): a quick-start guide to understanding and using the Micro PLC.

Download from Www.Somanuals.com. All Manuals Search And Download.

Technical Assistance

If you should have a problem installing or programming your GE Fanuc Micro PLC, and the information you need is not in this book or the *Micro PLC User's Guide*, you can call GE Fanuc Field Service at 1-800-828-5747.

We Welcome Your Comments and Suggestions

At GE Fanuc automation, we strive to produce quality technical documentation. After you have used this manual, please take a few moments to complete and return the Reader's Comment Card located on the next page.

Jeanne L. Grimsby
Senior Technical Writer

Download from Www.Somanuals.com. All Manuals Search And Download.

Chapter 1 Programming for the Micro PLC 1-1

Programming Basics 1-2
PLC Programs 1-3
The Micro PLC Instruction Set 1-4
Programming Devices and Formats 1-5
Memory Types and Addresses 1-6
Constants and Register Values in a Program 1-6
Special Coils 1-7
Programming for an Analog Expander Unit 1-8

Chapter 2 Programming with the Programming Software 2-1

Using the Programming Functions 2-2
Creating a Program Rung 2-4
Running the Programming Software 2-5
Editing Basics 2-6
Horizontal and Vertical Lines in a Rung 2-7
Element Labels and Rung Labels 2-8
Editing a Completed Rung 2-9
Deleting Rungs 2-13
Moving Rungs 2-13
Copying Rungs 2-14
Searching for a Rung or Program Element 2-15

Chapter 3 Programming with a Hand-held Programmer 3-1

Program Listing 3-2
Program Transfer 3-2
Entering Program Logic 3-3
Inserting a Rung Element 3-4
Deleting a Rung Element, Rung or Program In Memory 3-5
Searching 3-7
Programming Examples Using the HHP 3-8

Chapter 4 The Micro PLC Instruction Set 4-1

Instruction Set Summary 4-2
Contacts 4-4
Normally-Open Contact 4-5
Normally-Closed Contact 4-7
Positive Transition Contact 4-9
Negative Transition Contact 4-10

Download from Www.Somanuals.com. All Manuals Search And Download.

Coils 4-11
Output Coil 4-12
Set / Reset Coil Pair 4-13
Master Control Relay / End Coil Pair 4-14
Skip / End Coil Pair 4-15
Timers 4-16
On Timer 4-18
Off Timer 4-19
Counters 4-20
Up Counter 4-22
Down Counter 4-23
Math Functions 4-24
Addition (ADD) 4-24
Subtraction (SUB) 4-26
Multiplication (MUL) 4-28
Division (DIV) 4-30
Move Functions 4-32
Move 4-32
Block Move 4-34
Indirect Move 4-36
Compare Functions 4-38
Logic Operations 4-40
Word AND 4-41
Inclusive OR (IOR) 4-42
Exclusive OR (XOR) 4-43
Shift Register Right 4-44
Shift Register Left 4-45
NOT 4-46

Download from Www.Somanuals.com. All Manuals Search And Download.

Appendix A Using Directories A-1

Appendix B Micro PLC Protocol B-1

Communications Files B-2
Communications Memory Types and Addresses B-3
Communications Parameters B-3
Communications Protocol B-4
Data Format B-4
Communications Functions B-8

Appendix C RTU Protocol C-1

Introduction C-1
Message Types C-2
Transmission Sequence C-2
Message Fields C-3
Character Format C-4
Message Termination C-4
Timeout Usage C-4
Cyclic Redundancy Check (CRC) C-5
RTU Message Length C-8
Message Descriptions C-9
Message (01): Read Output Table C-9
Message (02): Read Input Table C-10
Message (03): Read Registers C-11
Message (04): Read Analog Inputs C-12
Message (05): Force Single Output C-13
Message (06): Preset Single Register C-14
Message (07): Read Exception Status C-15
Message (16): Preset Multiple Registers C-16
Message (17): Report Device Type C-17
Communication Errors C-18
Invalid Query Message C-18
Serial Link Timeout C-19
Invalid Transactions C-19

Appendix D Communications Using Windows DDE D-1

Features of the Micro PLC DDE Driver Software D-2
Simple Demonstration using Microsoft Word D-3
Demonstration using Microsoft Excel D-3
Viewing PLC Data in Windows D-4
Viewing PLC Data in another DDE-compliant Application D-5
Writing Values to the PLC from another Application D-5
Ordering Information D-5

Download from Www.Somanuals.com. All Manuals Search And Download.

Appendix E Data Acquisition, Logging, and Display Program E-1

Features E-1
Using the Display Software with Micro PLC Net E-1
Overview E-2
Equipment Required E-4
Startup E-5
Changing the Screen Colors E-7
Editing Summary E-8
Manual Mode E-9
Creating or Editing Autopolling Screens E-13
System Messages E-19
Auto-Polling During System Operation E-21
Data Logging E-23
Error Messages During Operation E-25

Appendix F Programming Applications F-1

Application #1: FLIP / FLOP (Toggle Operation) F-2
Application #2: Power Up One Shot (Start–up Protection) F-3
Application #3: Cascading Counters F-4
Application #4: Industrial “Starting Circuit” F-5

Download from Www.Somanuals.com. All Manuals Search And Download.

Chapter 1 Programming for the Micro PLC

This chapter is an introduction to programming the Micro PLC.

- Programming Basics
- PLC Programs
  - Power Flow in a Program
- The Micro PLC Instruction Set
- Programming Devices and Formats
  - Programming with the Programming Software
  - Programming on a Hand-held Programmer
- Memory Types and Addresses
  - Memory Map
  - Non-retentive and Retentive Registers
  - Reserved Registers
  - Constants and Register Values in a Program
- Special Coils
  - 0.1 Sec Clock (C1018)
  - Start-up Scan Coil (C1019)
  - Hold Output Coil (C1021)
- Programming for an Analog Expander Unit
  - Analog Scaling
  - Analog References
  - Programming Examples

Download from Www.Somanuals.com. All Manuals Search And Download.

Programming Basics

The most important ingredients in creating a successful PLC program are a thorough understanding of the application itself, and a good measure of common sense.

The first step in creating a PLC application program is planning.

- □ The desired sequence of program actions is determined.
- □ All of the required inputs and outputs are identified and listed.
- □ Each input and output is associated with a PLC memory location. *For example:*

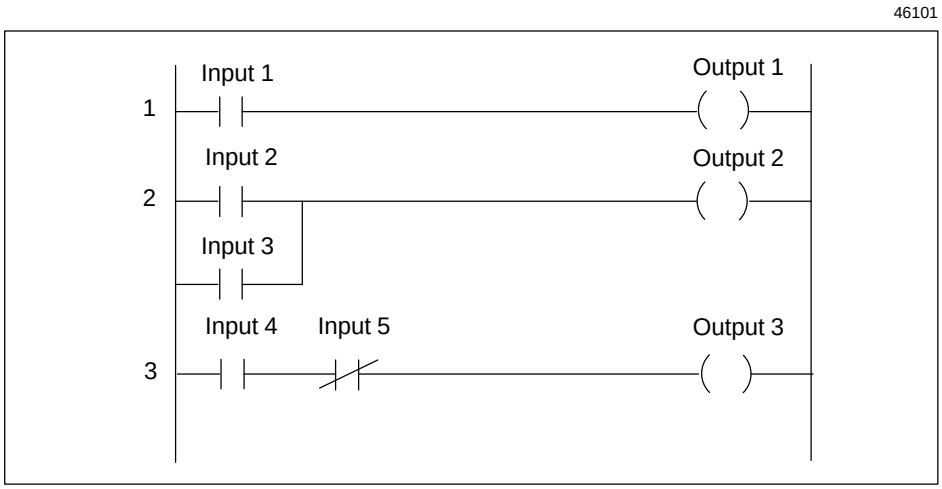
Device	Designation	Memory Location
Start switch	Input 1	Discrete Input Table 1
Limit switch on conveyor line	Input 2	" 2
Syrup tank #1, level detector	Input 3	" 3
Syrup tank #2, level detector	Input 4	" 4
Conveyor line optical sensor	Input 5	" 5
Conveyor line motor starter	Output 1	Discrete Output Table 1
Operator warning light	Output 2	" 2
Signal to bottle capper	Output 3	" 3

- □ The program (like the short example program on the facing page) is created with a programming device and transferred to the PLC.
- □ Before the system begins full operation, the program is tested and any corrections that are needed are made.
- □ The final version of the program is transferred to the PLC, and the application is ready to go.

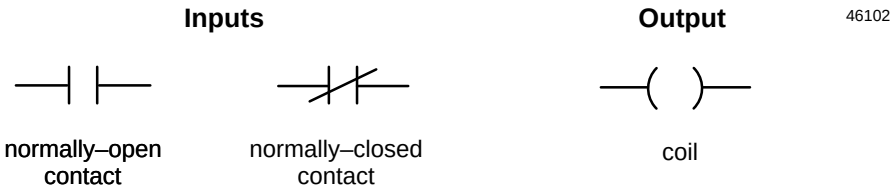
Download from Www.Somanuals.com. All Manuals Search And Download.

PLC Programs

A typical PLC **application program** is created in a format called **ladder logic**.



Each symbol in the ladder logic represents a type of input, output, or other program action. There are many types of symbols. The three symbols shown above are:



In the ladder logic, each line or group of lines that ends in an action being performed, such as an output being sent, is called a **rung**. In the example above, there are three rungs.

Power Flow in a Program

The PLC executes the logic in the ladder from top to bottom, one rung at a time. Within each rung, the execution flows from left to right. This movement of program execution through the ladder can also be thought of as power flow. *In the example:*

Rung 1: Input 1 represents a switch. It is shown in the ladder logic program as a "normally–open" contact. When the switch is turned on, the input 1 contact closes and power flows across rung 1 to the coil labelled Output 1.

Rung 2: Rung 2 begins at the left side with two lines of logic that lead to the same output on the right. In this type of rung, which can have several lines beginning on the left, the output is ON if *any* of the input lines can be completed. In this rung, if either Input 2 or Input 3 is closed, Output 2 is turned ON.

Rung 3: Rung 3 illustrates the use of multiple inputs in the same line of logic. *All* of these inputs must be completed for the output to be ON. In this example, Input 4 must be closed (active), and Input 5 must be closed (inactive) for Output 3 to be set to ON.

Download from Www.Somanuals.com. All Manuals Search And Download.

The Micro PLC Instruction Set

Programs for a PLC are created from the elements provided in its Instruction Set. The Instruction Set for the GE Fanuc Micro PLC includes both basic relay-replacement contacts and many advanced program functions:

- Contacts
  - Normally-open Contact
  - Normally-closed Contact
  - Positive Transition Contact
  - Negative Transition Contact
- Outputs
  - Output coil
  - Set coil
  - Reset coil
  - Master Control Relay
  - Skip / Jump
- Timers
  - On Timer
  - Off Timer
- Counters
  - Up Counter
  - Down Counter
- Math functions
  - Addition
  - Subtraction
  - Multiplication
  - Division
- Move functions
  - Move
  - Block Move
  - Indirect Move
- Comparison functions
  - Equal
  - Not Equal
  - Greater Than
  - Less Than
  - Greater Than or Equal to
  - Less Than or Equal to
- Logical operation functions
  - AND
  - Inclusive OR
  - Exclusive OR
  - Shift Right
  - Shift Left
  - Not

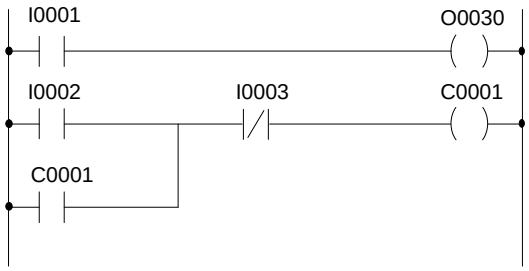
Download from Www.Somanuals.com. All Manuals Search And Download.

Programming Devices and Formats

Programs for the Micro PLC can be created using a computer that is equipped with the programming software, or using a Hand-held Programmer.

Programming with the Programming Software

Programs created with the programming software are in traditional ladder logic format:



Chapter 2 describes programming with the programming software.

Programming on a Hand-held Programmer

Equivalent programs are easily created on the Hand-held Programmer. For example:

Key Operations	HHP Displays
START I 1 ENTER	STA I001 Empty location
OUT O 3 0 ENTER	OUT O030 Empty location
START I 2 ENTER	STA I002 Empty location
OR C 1 ENTER	OR C0001 Empty location
AND F3 I 3 ENTER	AND NOT I003 Empty location
OUT C 1 ENTER	OUT C0001 Empty location

Chapter 3 describes programming with a Hand-held Programmer.

Download from Www.Somanuals.com. All Manuals Search And Download.

Memory Types and Addresses

Memory Map

Type	Total	Non Retentive	Retentive	Use for Timer or Counter Coil?	General Purpose Internal Coil?	Use as General Purpose Register?	Use as Indirect Register Reference?
I	256	1 – 256	none	no	yes	no	no
O	256	1 – 256	none	yes	yes	no	no
R	512	1 – 384	385 – 512*	no	yes	yes	yes
IR	256	1 – 256	no	not applicable	not applicable	yes	no
OR	256	1 – 256	no	not applicable	not applicable	yes	no
C	1017	1 – 768	769 – 1017	yes	yes	no	no
C	1	1018		0.1 sec clock for use as input in application program (read only).			
C	1	1019		Startup scan coil for use as input in application program. (read only)			
C	1		1021	Hold output coil for use in application program (read only).			

Non-retentive and Retentive Registers

Data assigned to retentive memory is saved if power is removed from the Micro PLC.

Non-retentive registers are cleared to zero when power is removed, and when the Micro PLC is switched from Stop mode to Run mode.

* Reserved Registers

Retentive Registers 501 through 512 should not be used in your application program; they are reserved.

Constants and Register Values in a Program

Many program functions for the Micro PLC use either constants, or variables in registers. Constant values are limited to 32757 maximum. Single-register variables are limited to 65535 maximum.

Download from Www.Somanuals.com. All Manuals Search And Download.

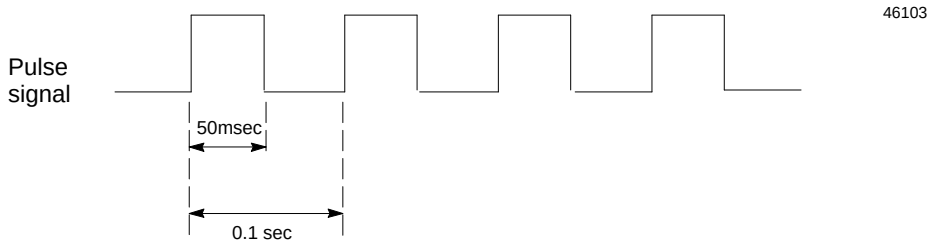
Special Coils

The Micro PLC provides three special-purpose coils:

- **0.1 sec clock**
- **start–up scan coil**
- **hold output coil**

0.1 Sec Clock (C1018)

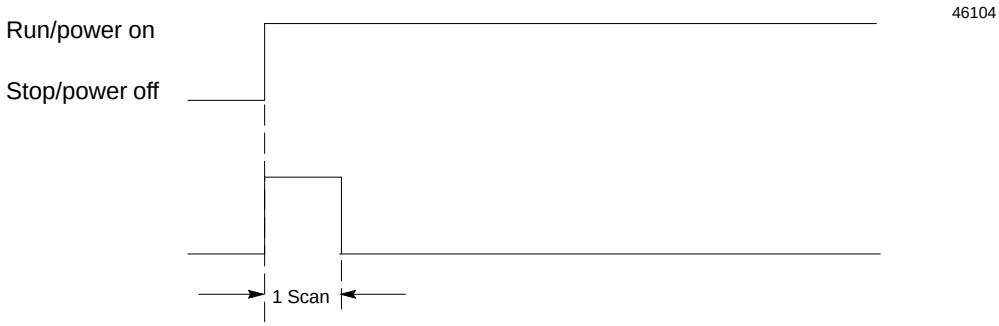
Coil C1018 is a pulse generator. The pulse width is shown below. This coil is a read-only coil. It can only be used as a program input, not as an output.



This coil should be used as a one-shot contact that feeds a regular coil.

Start-up Scan Coil (C1019)

When the controller starts operating, this coil goes ON for one scan. It is a read-only coil. It can only be used as a program input, not an output.



Hold Output Coil (C1021)

This coil can be used to control the state of the program outputs when the PLC is put in stop mode. As a group, all of the outputs can either hold their last state, or be set to OFF.

If all outputs should hold their last state, set coil C1021 to ON (1).

If all outputs should be set to OFF, set coil C1021 to OFF (0).

Note: This function takes precedence over the Clear Data function of the programming software (key **F7** in the Online menu).

Download from Www.Somanuals.com. All Manuals Search And Download.

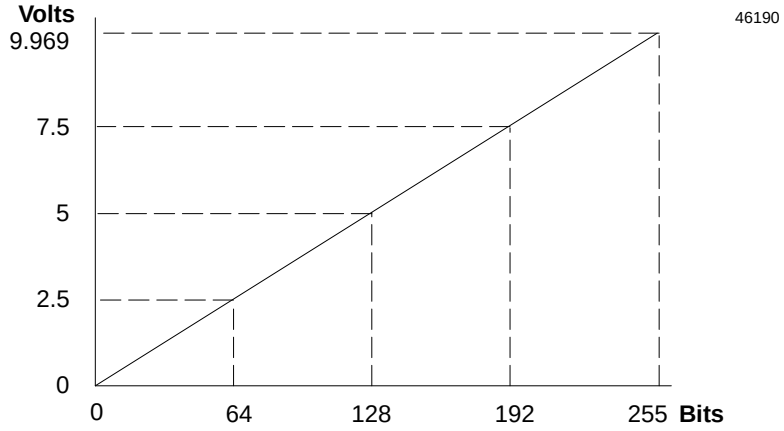
Programming for an Analog Expander Unit

Analog Scaling

The Analog Expander Unit provides two 8-bit analog inputs and one 8-bit analog output. Scaling for an analog input or output is:

Minimum: 0 volts = 0mA = 0 bits
Maximum: 9.969 volts = 19.922mA = 255 bits

Some examples of equivalent values are:



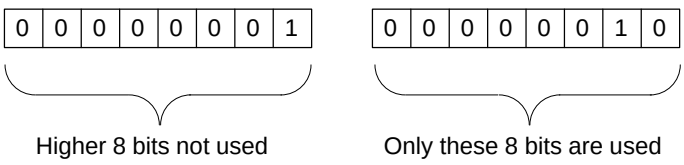
Analog References

In a program, input 1 uses reference IR1. Input 2 uses reference IR2. The analog output uses reference OR1.

Maximum Values

The program references (IR1, IR2, and OR) used to store analog data are 16 bits each. However, the module utilizes only the lower 8 bits. Therefore, it is important not to inadvertently program a value greater than 255, which would cause incorrect results.

For example, suppose you programmed an output value of 258. This is shown below in bits. Because the module uses only the lower 8 bits, it would interpret the value incorrectly.



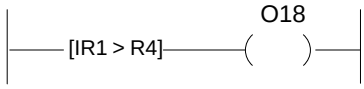
Download from Www.Somanuals.com. All Manuals Search And Download.

Programming Examples

Two simple programming examples are shown below.

Example #1:

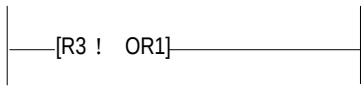
For an analog input, the program might read the input value and turn on a discrete output when the analog input reaches a specific value. In this example, the program compares the value of the first analog input (IR1) with a value stored in register R4. If the analog input is greater than that value, then a discrete output (O18) is turned on.



The output that is turned on might represent an actual output device such as a switch, or a logical output that is used elsewhere in the program.

Example #2:

The logic below might be used for an analog output.



In this example, each program scan, the Move function copies the content of register R3 to reference OR1, which is the reference used by the analog output.

Download from Www.Somanuals.com. All Manuals Search And Download.

Chapter 2 Programming with the Programming Software

This chapter explains how to create and edit a program using the Micro PLC programming software.

- Using the Programming Functions
- Creating a Program Rung
- Running the Programming Software
- Editing Basics
- Horizontal and Vertical Lines in a Rung
- Element Labels and Rung Labels
- Editing a Completed Rung
- Deleting Rungs
- Moving Rungs
- Copying Rungs
- Searching for a Rung or Program Element

See the *Micro PLC User's Guide* if you need information about:

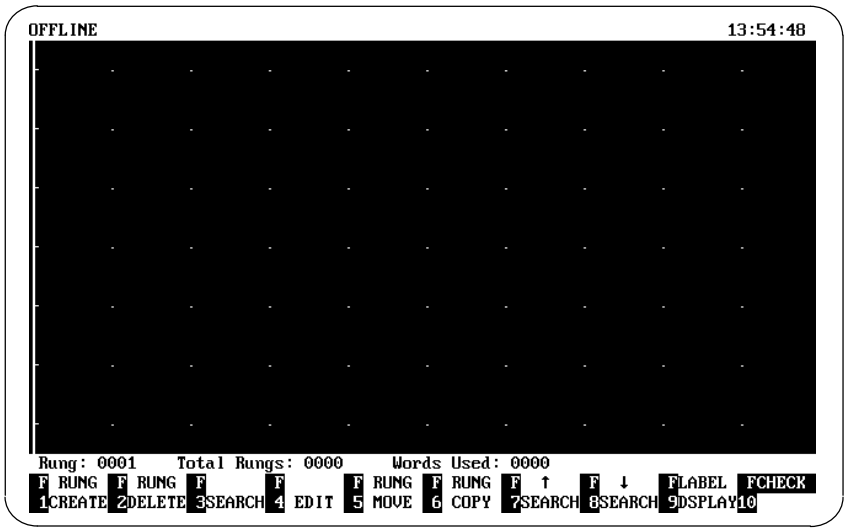
- Software installation
- Software functions
- Changing to another directory
- Loading a program file
- Saving a Program File
- Clearing a program from RAM memory
- Printing an application program
- Exiting the programming software
- Setup parameters
- Files and file-handling
- Monitoring a program online in the PLC

Download from Www.Somanuals.com. All Manuals Search And Download.

Using the Programming Functions

When you select **Offline (F3)** from the Main menu, the application program currently in the computer's RAM memory appears.

If there is no program currently in RAM memory, the screen looks like this:



The window shows the current rung, total number of rungs in the program, and program size in words. If you want to quit the Programming window, use the ESC key.

Programming Operations

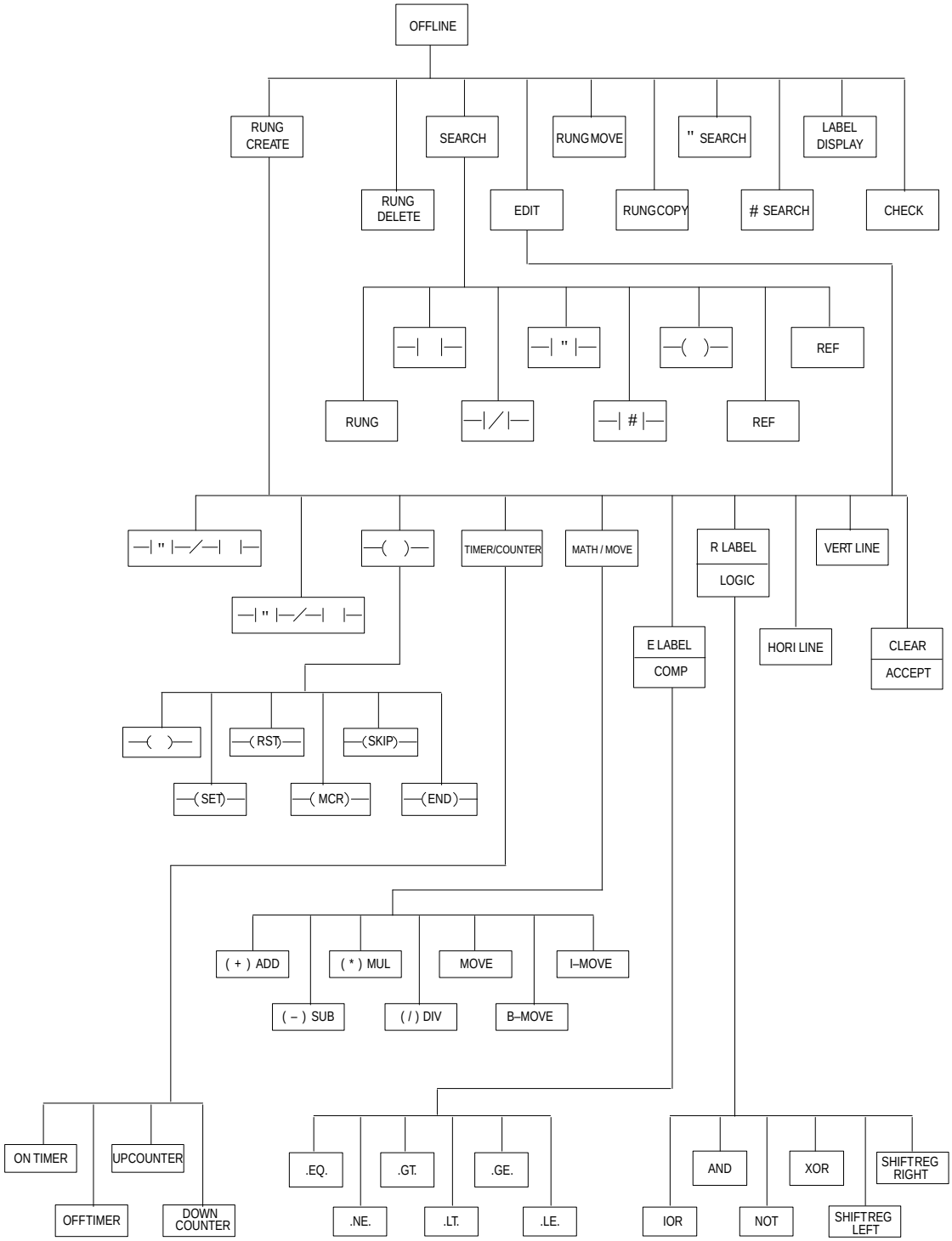
In the Offline window, use the function keys to select a programming operation.

Rung Create (F1)	to create a new program rung
Rung Delete (F2)	to delete a program rung
Search (F3)	to search for a type of function or operand
Edit (F4)	to edit the rung at the top of the screen. After pressing F4, – moves the cursor within the rung. PgUp, PgDn moves from rung to rung.
Rung Move (F5)	to move a program rung
Rung Copy (F6)	to duplicate a program rung
" **Search (F7)**	to search previous rungs (search backward)
# **Search (F8)**	to search next program rungs (search forward)
Label Display (F9)	to toggle between absolute and symbolic display
Check (F10)	to check program syntax.

Download from Www.Somanuals.com. All Manuals Search And Download.

Programming Functions

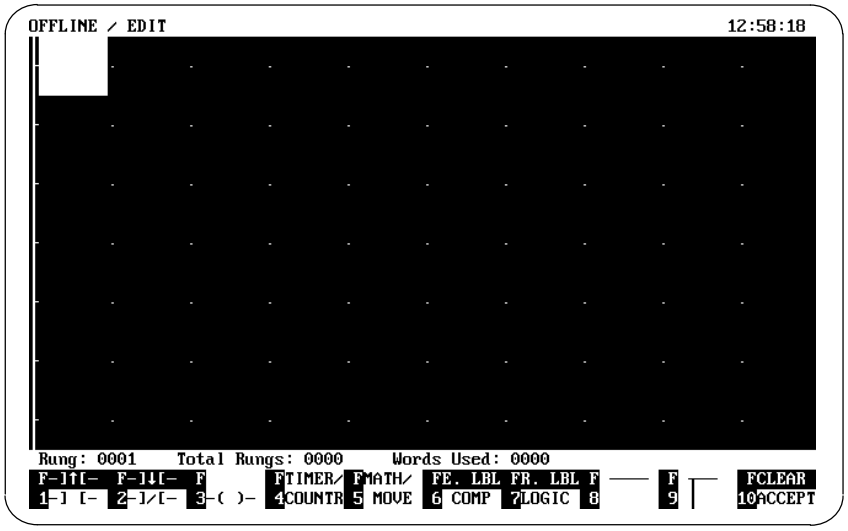
46105



Download from Www.Somanuals.com. All Manuals Search And Download.

Creating a Program Rung

Select **Rung Create (F1)** to create a new program rung. If there are already rungs in the program, the new rung will appear at the top of the page.



You can now enter a program element in the highlighted location. Use the ESC key if you want to return to a previous menu.

In the Edit window, use the function keys to select a program element.

–| |– (F1)	Normally-open contact. See page 4-4.
–| " |– (Shift, F1)	Positive transition contact. See page 4-4.
–| / |– (F2)	Normally-closed contact. See page 4-4.
–| # |– (Shift, F2)	Negative transition contact. See page 4-4.
–( )– (F3)	Output. See page 4-11.
Timer/Counter (F4)	Timer or Counter. See pages 4-16 and 4-19.
Math/Move (F5)	Math or Move function. See pages 4-22 and 4-30.
Comp (F6)	Compare function. See page 4-36.
E. Lbl (Shift, F6)	Element label. See page 2-8.
Logic (F7)	Logic function. See page 4-38.
R. Lbl (Shift, F7)	Rung label. See page 2-8.
_____ (F8)	Draw horizontal (serial) line. See page 2-7.
┌ (F9)	Draw or erase vertical (parallel) line. See page 2-7.
Accept (F10)	Exit editing a rung. See page 2-6.
Clear (Shift, F10)	Delete (clear) the rung being edited. Be careful. **Note**: the Delete key on your keyboard will remove the element at the current position in either Rung Edit or Rung Create mode.

Download from Www.Somanuals.com. All Manuals Search And Download.

Running the Programming Software

The programming software can be run from diskette, or installed on a hard disk. For installation instructions and information about the files on the software diskette, refer to the *Micro PLC User's Guide* (GFK-0803).

Running the Programming Software from a Hard Disk

1. **Go to the directory where you placed the MICRO.EXE file.** For example:

   C:>CD MICRO (Press the Enter key)

2. **To run the programming software, type:**

   C:\MICRO>MICRO (Press the Enter key)

Running the Programming Software Directly from a Diskette

1. **Go to the DOS prompt if it is not already displayed:**

   A:>

2. **Place the diskette containing the programming software into the appropriate diskette drive (for example, drive A).** To run the programming software, type:

   A:>MICRO (Press the Enter key)

3. **Place a formatted diskette into drive B. You can use this diskette for storing configuration files.**

Download from Www.Somanuals.com. All Manuals Search And Download.

Editing Basics

After selecting **Rung Create (F1)**:

- use the function keys to create an element and enter a reference value. For example, “I2”. (Note that the element reference cannot be entered as “2I”).
- use the cursor keys to move to another position in the rung being edited
- use your keyboard Delete key to delete a rung element.
- use your keyboard ESC key to quit a function or to return to the previous menu.
- To create another rung, press **Rung Create (F1)**. The new rung appears.

Initial display:

Enter a function at the highlighted location. (Use the function keys)

Move the highlight box. (Use the cursor keys)

Enter the next function:

When you enter a coil, the highlight box automatically goes to the end of the rung:

After entering all the logic for the rung, use the **Accept (F10)** key to add it to the program. The disappearance of the highlight box shows that the software is not presently in Edit mode.

Download from Www.Somanuals.com. All Manuals Search And Download.

Horizontal and Vertical Lines in a Rung

Horizontal and vertical lines are used to connect elements of a multi-line rung.

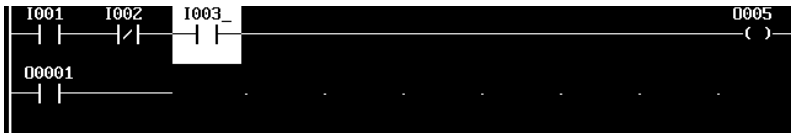
If a rung has more than one line of logic, move the highlight box down to the start of the next line. Enter the first element on that line.

To add a horizontal line to the logic, move the highlight box to the location for the line.

Use the —— **(F8)** key to add the horizontal line.

Use the Delete key on your keyboard to remove a horizontal line.

To add a vertical line to the logic, move the highlight box to the end location for the line.



Use the ⌐ **(F9)** key to add the vertical line.

You must use the ⌐ **(F9)** key to remove a vertical line.

Download from Www.Somanuals.com. All Manuals Search And Download.

Element Labels and Rung Labels

Element labels and rung labels are text that can be added to a program and viewed in **Offline** mode by selecting the **Label Display** function. Element and rung labels will also appear in their entirety in a hard copy printout or a print to disk. The first 5 characters of an element label can also be viewed in Online mode. (Think about this when selecting the element label.) To *create* an element label or rung label, follow the instructions below, in **Edit** or **Create Rung** mode.

To create a label for an individual element, move the highlight box to that element. Press Enter to clear element edit. Then, press **Shift, F6**.

Enter the text for the element label. Pressing the Enter (return) key ends text entry. Enter spaces if you want the label to have more than one line.

After you press the Enter key, the element label disappears.

To create a label for a rung, press **Shift, F7**.

Enter the text for the rung label. Pressing the Enter (return) key ends text entry. Enter spaces if you want the label to have more than one line.

After you press the Enter key, the element label disappears.

Download from Www.Somanuals.com. All Manuals Search And Download.

Editing a Completed Rung

After using the **Accept (F10)** key to save a rung, the rung can be changed by selecting **Edit (F4**) with the Offline function keys. The same basic programming features are available in both Create mode and Edit mode. Refer to the previous descriptions of program functions and editing techniques.

While editing a rung, you can delete, add, or change rung elements within the rung as described below and on the following pages.

If you want to delete or move an entire rung, see page 2-13.

Selecting a Rung to Edit

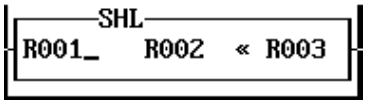
To select a rung for display or editing, use the cursor keys to scroll the program up or down on the screen. Or you can use the Search function to locate a specific rung, element, or reference.

Bring the rung you want to edit to the top of the screen.

To edit the rung, use the **Edit (F4)** key. In Edit mode, you can add elements to the rung, or edit the elements that are already there. You can then use the cursor keys to move the cursor within a rung, and the PgUp, PgDn keys to move from rung to rung.

Editing a Rung Element

In Edit mode, the highlight box selects the rung element that can be edited.



Use the cursor keys to move to the highlighted box.

Enter the reference and address, then press the Enter key or a cursor key to go to the next rung element.

Deleting a Rung Element

Delete the highlighted element by pressing the Delete key on your keyboard. (If the small edit cursor is displayed in the highlight box, first press the Enter key to remove it).

This creates an empty space in the rung. If you do not want to add an element at that location, insert a horizontal line **(F8)**.

Download from Www.Somanuals.com. All Manuals Search And Download.

Adding a Contact to a Rung

To add a contact to a rung, move the highlight box to the location for the new contact.

Add the contact using its function key. Enter a reference for the contact and press the Enter key, or use the cursor keys to move to a new position.

Adding a Program Function to a Rung

If you want to add something other than a contact (for example, an Equal function) to a rung, first create a space for it. Position the highlight box where the function should begin, then press your keyboard Delete key. If the element to be added requires 2 or 3 consecutive spaces, move the highlight box, then press the Delete key again.

Move the highlight cursor back to the leftmost empty space:

Add the function using its function key. Edit the function as necessary.

Download from Www.Somanuals.com. All Manuals Search And Download.

Replacing a Rung Element with a Similar Element

To replace a rung element with a similar element (for example, to replace a normally-open contact with a normally-closed contact), select the element with the highlight box.

Press the function key that corresponds to the new element type. Edit it if necessary.

Replacing a Rung Element with a Horizontal Line

To replace a rung element with a horizontal line, select it with the highlight box:

Then , select the horizontal line **(F8)**.

Download from Www.Somanuals.com. All Manuals Search And Download.

Replacing a Rung Element with a Dissimilar Element

To replace an element with an element of a different type (for example, to change a normally-open contact to an Equals function), select the element with the highlight box.

First, delete the element by pressing the Delete key on your keyboard. (If the small edit cursor is displayed in the highlight box, first press the Enter key to remove it).

This creates an empty space in the rung. If you do not want to add an element at that location, insert a horizontal line **(F8)**.

If you want to add an element at that location, use the appropriate function key to enter it.

You may need to delete more than one existing element to insert a new element if the new element is wider than the element it is replacing.

Download from Www.Somanuals.com. All Manuals Search And Download.

Deleting Rungs

To delete one or more rungs, use the **Rung Delete (F2)** key. The software prompts:

The first number that appears is the number of the rung that is now at the top of the screen. If you want to delete just that rung, press the Enter key.

If you want to delete a different rung, or a group of rungs, enter their rung numbers then press the Enter key.

If you want to quit the Delete function without deleting any rungs, press the ESC key.

Moving Rungs

You might want to move rungs to make a program more understandable, or to group rungs together that work with each other.

To move one or more rungs, use the **Rung Move (F5)** key. The software prompts:

The first number that appears is the number of the rung that is now at the top of the screen.

Enter the numbers of the rungs to be moved, and the number of the rung you want to insert them in front of. Press the Enter key.

If you want to quit the Move function without moving any rungs, press the ESC key.

Download from Www.Somanuals.com. All Manuals Search And Download.

Copying Rungs

You might want to copy rungs and then make simple changes to the rungs rather than enter new rungs.

To copy one or more rungs, use the **Rung Copy (F6)** key. The software prompts:

The first number that appears is the number of the rung that is now at the top of the screen.

Enter the numbers of the rungs to be copied, and the number of the rung you want to insert the copied rungs in front of. Press the Enter key.

If you want to quit the Copy function without copying any rungs, press the ESC key.

Download from Www.Somanuals.com. All Manuals Search And Download.

Searching for a Rung or Program Element

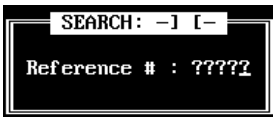
You can search for a specific program rung, contact or coil, reference address, number, or program function. Select **Search (F3)** in Offline mode, then use the function keys described below to select the target of the search. Finally, use the " **Search (F7)** and # **Search (F8)** keys to specify the search direction.

Rung (F1) rung number

Type in a rung number and press the Enter key.

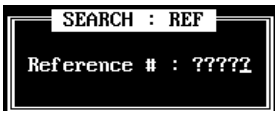
–] [– (F2) normally-open contact
–] / [– (F3) normally-closed contact
–] " [– (F4) positive transition contact
–] # [– (F5) negative transition contact
–( )– (F6) coil

Type in a reference address and press the Enter key.



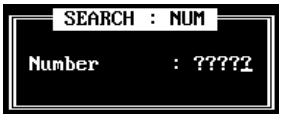
Ref (F7) reference

Type in a reference address and press the Enter key.



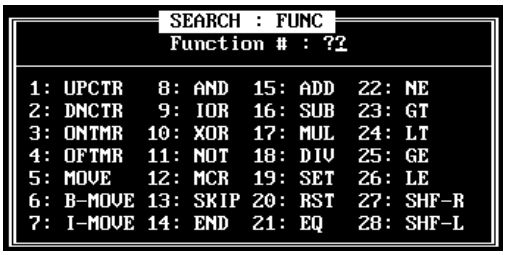
Number (F8) number

Type in a number and press the Enter key.



Func (F10) program function

Type in a number and press the Enter key.



Download from Www.Somanuals.com. All Manuals Search And Download.

Chapter 3 Programming with a Hand-held Programmer

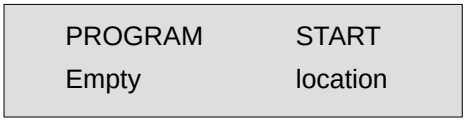
This chapter explains how use a Hand-held Programmer for programming the GE Fanuc Micro PLC.

- Program Listing
- Program Transfer
- Entering Program Logic
- Inserting a Rung Element
- Deleting a Rung Element, Rung or Program In Memory
- Searching
- Programming Examples Using the HHP

Download from Www.Somanuals.com. All Manuals Search And Download.

Program Listing

To display the program listing, press the ENTER key from the powerup screen.



Program Transfer

To transfer a program, press the XFER key from the program listing screen.

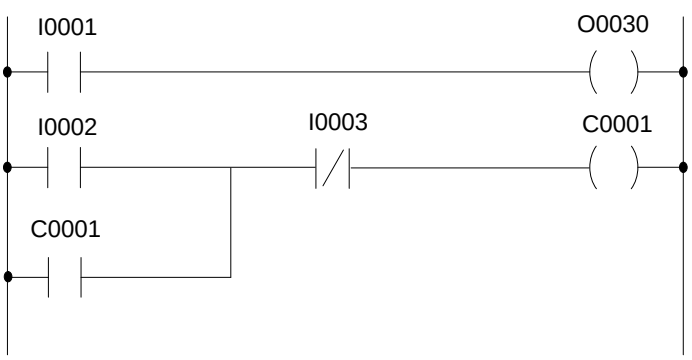
The Hand-held Programmer will transfer the program to or from the Micro PLC, EEPROM, or the programming software in the computer.

Download from Www.Somanuals.com. All Manuals Search And Download.

Entering Program Logic

The steps below show how to create a simple example program.

Example Program



Hand-Held Programmer Key Operation and Displays

- use the ENTER key to accept a command.
- On the display, “empty location” refers to the current unoccupied program line.

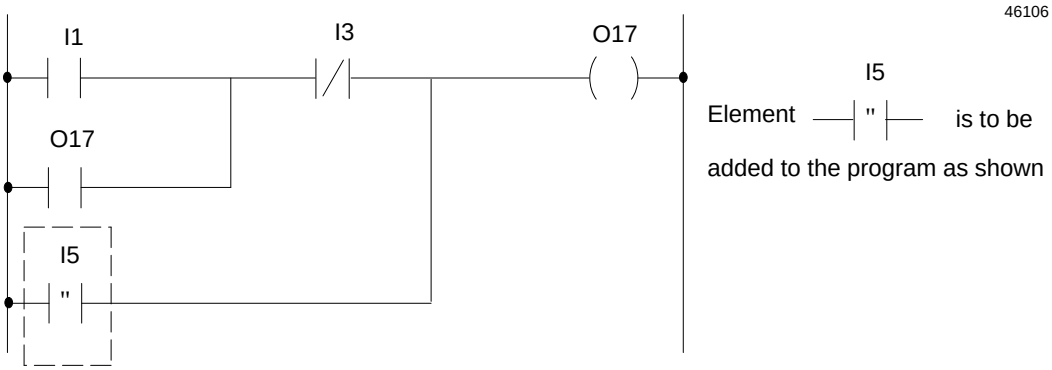
Key Operations	HHP Displays
START I 1 ENTER	STA I001 Empty location
OUT O 3 0 ENTER	OUT O030 Empty location
START I 2 ENTER	STA I002 Empty location
OR C 1 ENTER	OR C0001 Empty location
AND F3 I 3 ENTER	AND NOT I003 Empty location
OUT C 1 ENTER	OUT C0001 Empty location

Download from Www.Somanuals.com. All Manuals Search And Download.

Inserting a Rung Element

The steps below show how to insert an element in a simple example program.

Example Program



In this example, the statement OR PTRAN I005 is inserted between the existing statements:

AND NOT I003

OR PTRAN I005
OUT O017

Hand-Held Programmer Key Operation and Displays

- use the SRCH key to locate the ladder logic rung to be edited. The display selects the last element in the selected rung.
- Use the PREV key to select the previous element. In this example, it is:

  AND NOT I003 (see below).

- use the PREV key to scroll the program display backward.
- use the NEXT key to scroll the program display forward.

Key Operations	HHP Displays
SRCH 1 ENTER	OUT O017 Empty location
PREV	AND NOT I003 OUT O017
OR F1 I 5 ENTER	OR PTRAN I005 OUT O017

Download from Www.Somanuals.com. All Manuals Search And Download.

Deleting a Rung Element, Rung or Program In Memory

The steps below show how to use the HHP's delete function key to delete a program or part of a program in memory.

Hand-Held Programmer Key Operation and Displays

- use the SRCH key to locate the ladder logic rung to be edited. The display selects the last element of the rung. Use the PREV key to select a previous element.
- use the PREV key to scroll the program display backward.
- use the NEXT key to scroll the program display forward.
- use the DEL key and F1 key to delete a program element.
- use the DEL key and F2 key to delete a rung.
- use the DEL key and F3 key to delete a program.

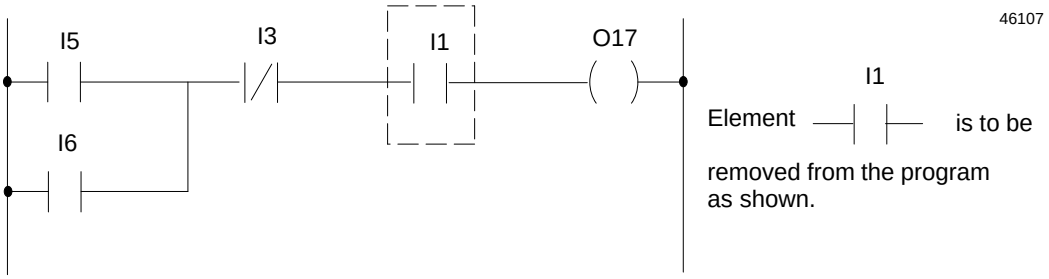
Key Operations	HHP Displays
DEL F1 (Elem F1 Rung F2 Prog F3) Delete the program element currently displayed.	DELETE Elem Rung Prog
DEL F2 Delete the program rung currently displayed.	DELETE Elem Rung Prog
DEL F3 Delete the current program.	DELETE PROGRAM ENTER=YES, ESC=No

Download from Www.Somanuals.com. All Manuals Search And Download.

Deleting a Rung Element

The steps below show how to delete an element of a simple example program.

Example Program



AND NOT I003

OR PTRAN I005
OUT O017

Hand-Held Programmer Key Operation and Displays

- use the SRCH key to locate the ladder logic rung to be edited. The display selects the last element of the rung. Use the PREV key to select a previous element.
- use the DEL key and F1 key to delete a program element.
- use the PREV key to scroll the program display backward.
- use the NEXT key to scroll the program display forward.

Key Operations	HHP Displays
SRCH 1 ENTER	OUT O017 Empty location
PREV	AND I1 OUT O017
DEL F1	AND NOT I003 OUT O017

Download from Www.Somanuals.com. All Manuals Search And Download.

Searching

Use the SRCH key to locate a rung, element, or operand in a program. The steps below show how to search for an operand, element, or rung umber, or the start or end of the program.

Hand-Held Programmer Key Operation and Displays

Key Operations	HHP Displays
SRCH I/O enter number ENTER The search operand may be I, O, IR, OR, C, or R.	AND I001 OUT O017
SRCH CNTR F1 or F2 ENTER This example searches for a counter (either up or down).	C R001 To 100 Empty location
SRCH 1 0 ENTER This example searches for rung number 10.	OUT O040 Empty location
SRCH 0 To locate the start of the program, search for rung 0.	PROGRAM START STA I001
SRCH ENTER To locate the end of the program, search with no rung number.	OUT O040 Empty location

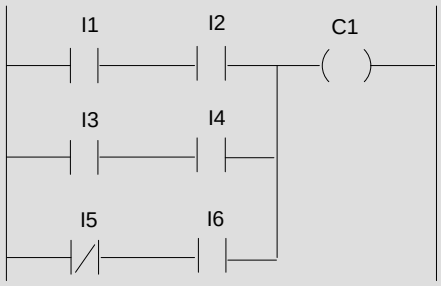
PLC TRANSFER Open Rung	An open rung or other program error was encountered.
PLC TRANSFER PLC in RUN Mode	The PLC was in Run mode when the transfer was attempted.

Download from Www.Somanuals.com. All Manuals Search And Download.

Programming Examples Using the HHP

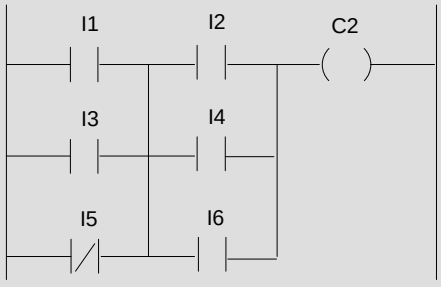
Example 1

46108

Ladder Diagram	Key Operations		
	START		I1 ENTER
	AND		I2 ENTER
	START		I3 ENTER
	AND		I4 ENTER
	OR		ENTER
	START	F3 (NOT)	I5 ENTER
	AND		I6 ENTER
	OR		ENTER
	OUT		C1 ENTER

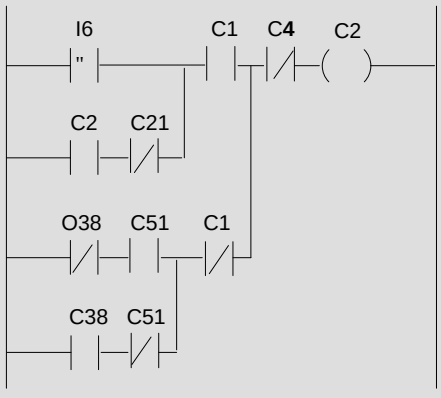
Example 2

46109

Ladder Diagram	Key Operations		
	START		I1 ENTER
	OR		I3 ENTER
	OR	F3 (NOT)	I5 ENTER
	START		I2 ENTER
	OR		I4 ENTER
	OR		I6 ENTER
	AND		ENTER
	OUT		C2 ENTER

Example 3

46110

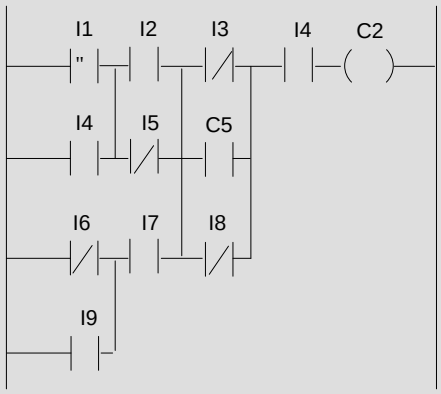
Ladder Diagram	Key Operations		
	START	F1 (PTRAN)	I6 ENTER
	START		C2 ENTER
	AND	F3 (NOT)	C21 ENTER
	OR		ENTER
	AND		C1 ENTER
	START	F3 (NOT)	O38 ENTER
	AND		C51 ENTER
	START		C38 ENTER
	AND	F3 (NOT)	C51 ENTER
	OR		ENTER
	AND	F3 (NOT)	C1 ENTER
	OR		ENTER
	AND	F3 (NOT)	C4 ENTER
	OUT		C2 ENTER

Download from Www.Somanuals.com. All Manuals Search And Download.

Example 4

46111

Ladder Diagram



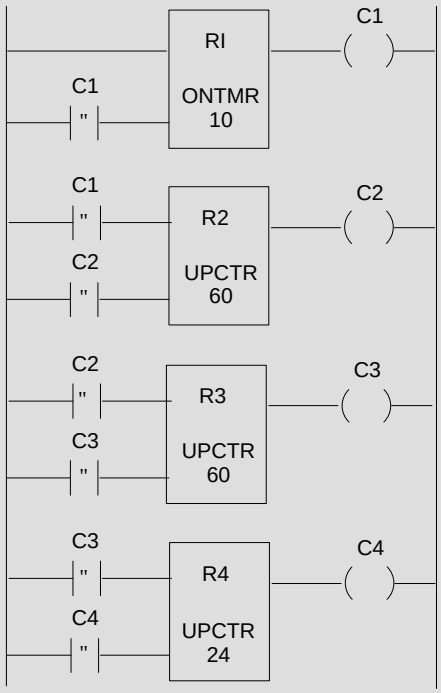
Key Operations

START	F1 (PTRAN)	I1	ENTER
OR		I4	ENTER
START		I2	ENTER
OR	F3 (NOT)	I5	ENTER
AND			ENTER
START	F3 (NOT)	I6	ENTER
OR		I9	ENTER
AND		I7	ENTER
OR			ENTER
START	F3 (NOT)	I3	ENTER
OR		C5	ENTER
OR	F3 (NOT)	I8	ENTER
AND		I4	ENTER
AND			ENTER
OUT		C2	ENTER

Example 5

46112

Ladder Diagram



Key Operations

START			ENTER
START	F1 (PTRAN)	C1	ENTER
TIMER	F1 (ONTIM)	R1	ENTER
		10	ENTER
OUT		C1	ENTER
			ENTER
START	F1 (PTRAN)	C1	ENTER
START	F1 (PTRAN)	C2	ENTER
CNTR	F1 (UPCNT)	R2	ENTER
		60	ENTER
OUT		C2	ENTER
			ENTER
START	F1 (PTRAN)	C2	ENTER
START	F1 (PTRAN)	C3	ENTER
CNTR	(F1 (UPCNT)	R3	ENTER
		60	ENTER
OUT		C3	ENTER
			ENTER
START	F1 (PTRAN)	C3	ENTER
START	F1 (PTRAN)	C4	ENTER
CNTR	F1 (UPCNT)	R4	ENTER
		24	ENTER
OUT		C4	ENTER

Download from Www.Somanuals.com. All Manuals Search And Download.

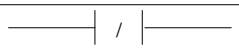
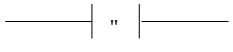
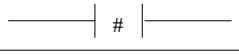
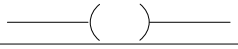
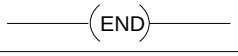
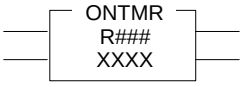
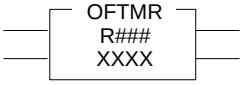
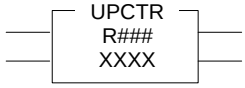
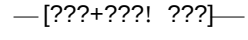
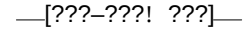
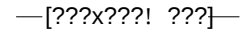
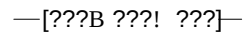
Chapter 4 The Micro PLC Instruction Set

This chapter defines the individual logic elements that can be combined to make a program for the Micro PLC.

- Instruction Set Summary
- Contacts
  - Normally-Open Contact
  - Normally-Closed Contact
  - Positive Transition Contact
  - Negative Transition Contact
- Coils
  - Output Coil
  - Set / Reset Coil Pair
  - Master Control Relay / End Coil Pair
  - Skip / End Coil Pair
- Timers
  - On Timer
  - Off Timer
- Counters
  - Up Counter
  - Down Counter
- Math Functions
  - Addition (ADD)
  - Subtraction (SUB)
  - Multiplication (MUL)
  - Division (DIV)
- Move Functions
  - Move
  - Block Move
  - Indirect Move
- Compare Functions
- Logic Operations
  - Word AND
  - Inclusive OR (IOR)
  - Exclusive OR (XOR)
  - Shift Register Right
  - Shift Register Left
  - NOT

Download from Www.Somanuals.com. All Manuals Search And Download.

Instruction Set Summary

Operation	Ladder Symbol	Description
Contact		Normally-open contact
		Normally-closed contact
		Positive transition
		Negative transition
Output		Output coil
		Set coil
		Reset coil
		Master Control Relay
		Skip/jump operation
		Ending operation for a skip or jump
Timer		On Timer Variable Register R### (R1 – R500) Timer Setting XXXX (Timebase 0.1S)
		Off Timer Variable Register R### (R1 – R500) Timer Setting XXXX (Timebase 0.1S)
Counter		Up Count Variable Register R### (R1 – R500) Timer Setting XXXX
		Down Counter Variable Register R### (R1 – R500) Timer Setting XXXX
Math Functions		Addition
		Subtraction
		Multiplication
		Division

Download from Www.Somanuals.com. All Manuals Search And Download.

Operation	Ladder Symbol	Description
Move Operations	——[???! ???]——	Move
	L —[???E???! ???]— N	Block Move
	——[???! @???]——	Indirect Move
Compare Function	———[S1 = S2]———	Continue when S1 = S2
	———[S1 > S2]———	Continue when S1 > S2
	———[S1 < S2]———	Continue when S1 < S2
	———[S1 ≥ S2]———	Continue when S1 ≥ S2
	———[S1 ≤ S2]———	Continue when S1 ≤ S2
	———[S1 0 S2]———	Continue when S1 0 S2
Logic Operation	A —[???N???! ???]— D	AND X Y O 1 1 1 1 0 0 0 1 0 0 0 0
	I —[???O???! ???]— R	IOR X Y O 1 1 1 1 0 1 0 1 1 0 0 0
	X —[???O???! ???]— R	XOR X Y O 1 1 0 1 0 1 0 1 1 0 0 0
	S —[???H???! ???]— R	Shift Right
	S —[???H???! ???]— L	Shift Left
	—[??? ! ???] —	Not

Download from Www.Somanuals.com. All Manuals Search And Download.

Contacts

The Micro PLC instruction set includes the following contacts:

–| |–	**Normally-open contact.** Passes power flow to the right when its associated reference is = 1.
–| " |–	**Positive transition contact**. Passes power flow to the right for one program cycle when its associated reference changes from 0 to 1.
–| / |–	**Normally-closed contact**. Passes power flow to the right while its associated reference is = 0.
–| # |–	**Negative transition contact**. Passes power flow to the right for one program cycle when its associated reference changes from 1 to 0.

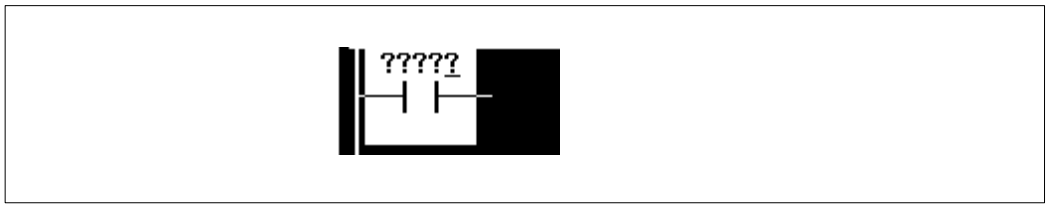
Each of these contact types is described more fully on the pages that follow. If you are using the programming software, refer to the programming instructions below. If you are using the Hand-held Programmer, refer to the instructions for each contact type.

General Programming Software Instructions for Contacts

1. **Select the contact type using the function keys:**

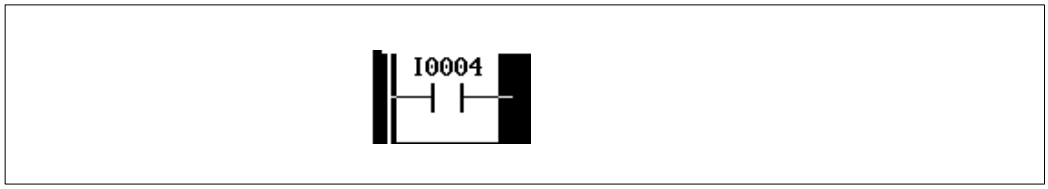
–| |–	**(F1)**	Normally-open contact.
–| " |–	**(Shift, F1)**	Positive transition contact.
–| / |–	**(F2)**	Normally-closed contact.
–| # |–	**(Shift, F2)**	Negative transition contact.

The selected contact type is displayed:



2. **Enter an appropriate address.** For a contact, enter an address in input (I), output (O), or internal (C) memory. For example: I4. You don't need to enter leading zeros. Note that you cannot enter the reference as "4I".

3. **Press the Enter key**. The memory type and address appear:



If you want to change the memory type and address, press the Enter key. A cursor appears next to the address. Type in the new memory type and address. Press the Enter key again to accept the new entry.

Download from Www.Somanuals.com. All Manuals Search And Download.

Normally-Open Contact

A Normally-Open contact passes power flow to the right when the associated reference is equal to 1.

Programming Software Instructions

If you are using the programming software, refer to the instructions on the opposite page.

Examples and HHP Instructions

The HHP programming keystrokes to create a Normally-Open contact depend on its position in a rung, as explained below.

Programming Normally-Open Contact at the Start of a Rung

When a Normally-Open contact is the first element of the rung, program a START, as shown below.

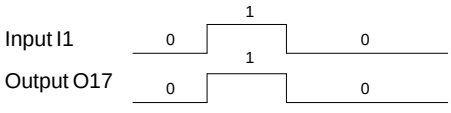
46114

Ladder Diagram

Hand-held Programmer

START	I1
OUT	O17

Timing Diagram



The output O17 status follows the signal of input I1, when I1–1, output O17=1.

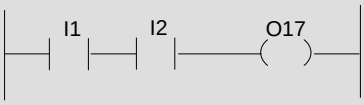
Download from Www.Somanuals.com. All Manuals Search And Download.

Programming a Normally-Open Contact in Series

When a Normally-Open contact is not the first element of the rung, program an AND with the HHP, as shown below. The example shows how to use the HHP to program a Normally-Open contact at the first and second positions in a rung.

46115

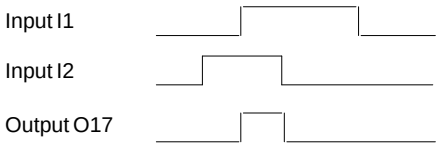
Ladder Diagram



Hand-held Programmer

START	I1
AND	I2
OUT	O17

Timing Diagram



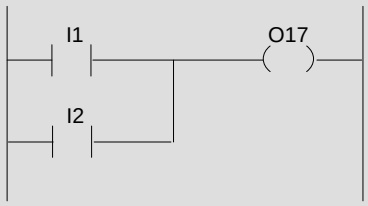
AND operation of inputs I1 and I2; output O17 is 1 when I1 and I2 are both 1.

Programming a Normally-Open Contact in Parallel

To enter a Normally-Open contact in parallel, as shown by I2 in the following example, program an OR with the HHP.

46116

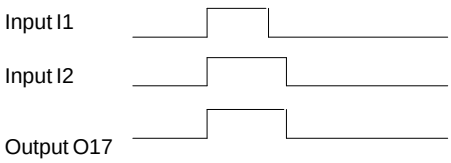
Ladder Diagram



Hand-held Programmer

START	I1
OR	I2
OUT	O17

Timing Diagram



OR operation of inputs I1 and I2. The output O17 is 1 when either I1 or I2 is 1.

Download from Www.Somanuals.com. All Manuals Search And Download.

Normally- Closed Contact

A Normally-Closed contact passes power flow to the right when the associated reference is equal to 0.

Programming Software Instructions

If you are using the programming software, refer to the instructions on page 4-4.

Examples and HHP Instructions

The HHP programming keystrokes to create a Normally-Closed contact depend on its position in a rung, as explained below.

Programming Normally-Closed Contact at the Start of a Rung

When a Normally-Closed contact is the first element of the rung, program a START NOT, as shown below.

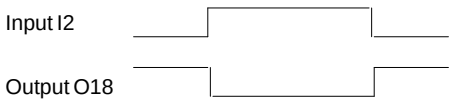
46017

Ladder Diagram

Hand-held Programmer

START	F3 (NOT)	II2
OUT		O18

Timing Diagram



The output O18 status is the reverse signal of input I2, when I2=1, output O18=0.

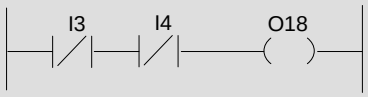
Download from Www.Somanuals.com. All Manuals Search And Download.

Programming a Normally-Closed Contact in Series

When a Normally-Closed contact is not the first element of the rung, program an AND NOT with the HHP, as shown below. The example shows how to use the HHP to program a Normally-Closed contact at the first and second positions in a rung.

46018

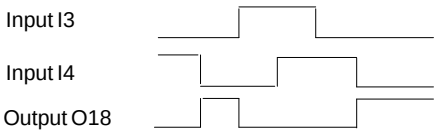
Ladder Diagram



Hand-held Programmer

START	F3 (NOT)	I3
AND	F3 (NOT)	I4
OUT		O18

Timing Diagram



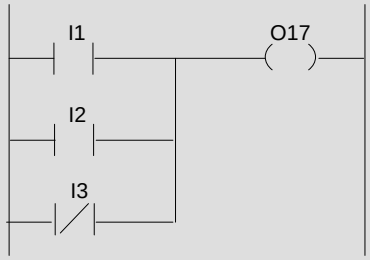
NAND operation of inputs I3 and I4; output O18 is 1 when I3 and I4 are both 0.

Programming a Normally-Closed Contact in Parallel

To enter a Normally-closed contact in parallel, as shown by I3 in the following example, program an OR NOT with the HHP.

46019

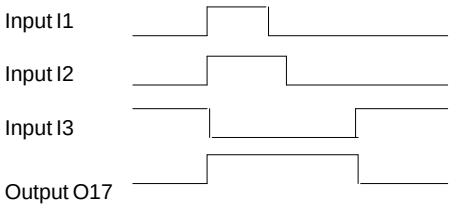
Ladder Diagram



Hand-held Programmer

START		I1
OR		I2
OR	F3 (NOT)	I3
OUT		O17

Timing Diagram



OR operation of inputs I1 and I2 and the reverse signal of I3. The output O17 is 1 when either I1 or I2 is 1 or I3 is 0.

Download from Www.Somanuals.com. All Manuals Search And Download.

Positive Transition Contact

The Positive Transition contact passes power flow to the right for one program cycle when its reference changes from 0 to 1.

Programming Software Instructions

If you are using the programming software, refer to the instructions on page 4-4.

Example and HHP Instructions

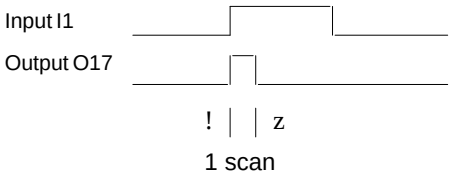
46020

Ladder Diagram

Hand-held Programmer

START	F1(PTRAN)	I1
OUT		O17

Timing Diagram



Upon the rising edge signal of input I1, output O17 is activated. The status of O17 remains 1 for only one program cycle.

Download from Www.Somanuals.com. All Manuals Search And Download.

Negative Transition Contact

The Negative Transition contact passes power flow to the right for one program cycle when its reference changes from 1 to 0.

Programming Software Instructions

If you are using the programming software, refer to the instructions on page 4-4.

Example and HHP Instructions

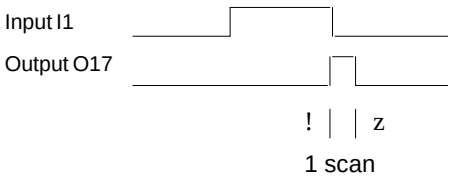
46021

Ladder Diagram

Hand-held Programmer

START	F2(NTRAN)	I1
OUT		O17

Timing Diagram



Upon the falling edge signal of input I1, output O17 is activated. The status of O17 remains 1 for only one program cycle.

Download from Www.Somanuals.com. All Manuals Search And Download.

Coils

The Micro PLC instruction set includes the following coils:

–( )–	**Coil.** The basic type of program output.
–(SET)–	**Set coil.** Used to set a specified reference to 1.
–(RST)–	**Reset coil.** Used to reset the same specified reference to 0.
–(MCR)–	**Master Control Relay.** Used to set a group of outputs to 0.
–(SKIP)–	**Skip.** Used to cause a group of outputs to hold last state.
–(END)–	**End** of MCR or Skip

Using Coil Pairs

The following types of coils must always be used in pairs. If one of these instructions is used in the program, the paired instruction must also exist somewhere in the program.

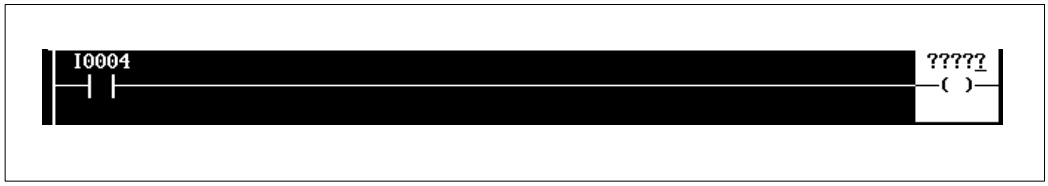
SET	**RST**	Set and Reset
MCR	**END**	Master Control Relay and End MCR
SKIP	**END**	Skip (rungs) to rung containing End

General Programming Software Instructions for Coils

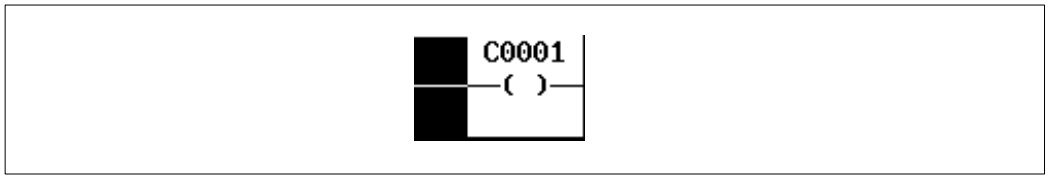
1. **Select –( )– (F3), then use the assigned function key(s) to select an output type:**

–( )–	**(F1)**	Coil
–(SET)–	**(F2)**	Set coil
–(RST)–	**(F3)**	Reset coil
–(MCR)–	**(F4)**	Master Control Relay
–(SKIP)–	**(F5)**	Skip
–(END)–	**(F6)**	End of MCR or Skip

The selected coil type appears at the end of the rung:



2. **Enter an appropriate address.** For a coil, enter an address in output (O) or internal coil (C) memory. For example, C0001. You don't need to enter leading zeros.
3. **Press the Enter key**. The memory type and address appear:



If you want to change the memory type and address, press the Enter key. A cursor appears next to the address. Type in the new memory type and address. Press the Enter key again to accept the new entry.

Download from Www.Somanuals.com. All Manuals Search And Download.

Output Coil

The basic type of output coil is controlled by conditional logic within the same rung. When the coil receives power flow from logic to its left in the rung, the reference associated with that coil is set to 1. When no power flow is received, the reference is 0.

The state of output coils may also be affected if they are between an MCR / End MCR coil pair or a Skip / End coil pair, as explained on the following pages.

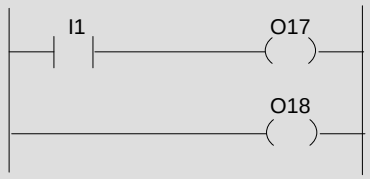
Programming Software Instructions

If you are using the programming software, refer to the instructions on page 4-11.

Examples and HHP Instructions

46022

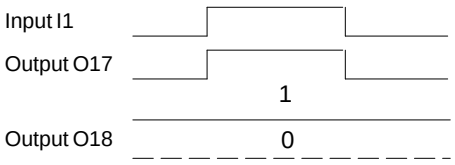
Ladder Diagram



Hand-held Programmer

START	I1
OUT	O17
START	
OUT	O18

Timing Diagram



Output O17 follows the status of input I1. Output O18 is unconditional; it always remains 1.

Download from Www.Somanuals.com. All Manuals Search And Download.

Set/Reset Coil Pair

The Set/Reset coil pair can be used to control the state of an associated reference. The Set coil sets the reference to 1. The reference (O17 in the example below) STAYS set to 1 until it is reset to 0 by the Reset coil. Notice that the Set and Reset coil are programmed with the same associated reference; that of reference being controlled.

In the example below, the Set coil and Reset coil are shown in adjoining rungs of logic. That is not necessary, however.

Programming Software Instructions

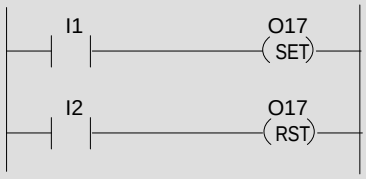
If you are using the programming software, refer to the instructions on page 4-11.

Example and HHP Instructions

Set (—(SET)—) **Reset** (—(RST)—)

46023

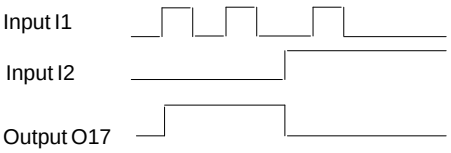
Ladder Diagram



Hand-held Programmer

START		I1
OUT	F2 (SET)	O17
START		I2
OUT	F2 (RESET)	O17

Timing Diagram



This is a flip-flop operation. Input I1 sets output O17 to 1. Output O17 remains 1 even if I1 changes to 0. Output O17 is only reset to 0 when I2 is activated.

When both I1 and I2 are ON, the last occurrence in the scan (RST in this case) takes precedence.

Download from Www.Somanuals.com. All Manuals Search And Download.

Master Control Relay/End Coil Pair

The MCR / End coil pair can be used to turn off one or more outputs in the program, *regardless of the state of any inputs or other conditional logic to those outputs*. The rungs with the outputs to be controlled must be located between the Master Control Relay coil and its paired End coil in the program.

In the example below, the MCR and End MCR coil control the states of outputs C1 and O17. While the Master Control Relay coil receives power flow from its conditional logic (in the example, it is contact I1), all output coils within the MCR / End MCR pair go to 0.

Programming Software Instructions

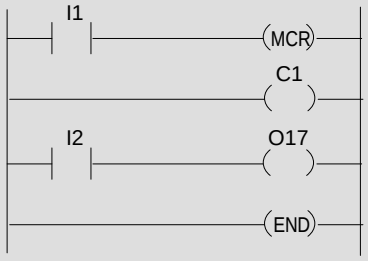
If you are using the programming software, refer to the instructions on page 4-11.

Example and HHP Instructions

MCR (—(MCR)— |) **End** (—(END)— |)

46024

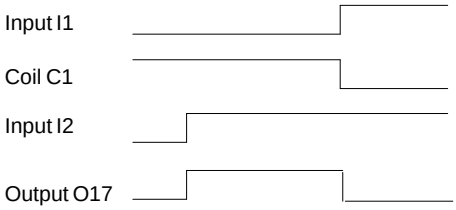
Ladder Diagram



Hand-held Programmer

START		I1
OUT	More MENU	F1 (MCR)
START		
OUT	C1	
START	I2	
OUT	O17	
OUT	More MENU	F3 (END MCR)

Timing Diagram



This is a Master Control Relay operation. When I1 is 0, the program till the END command operates normally.

When input I1 is 1, all outputs before the END are disabled.

Download from Www.Somanuals.com. All Manuals Search And Download.

Skip/End Coil Pair

The Skip / End coil pair can be used to cause one or more outputs in the program to stay in their current state (1 or 0), *regardless of the state of any inputs or other conditional logic to those outputs*. The rungs with the outputs to be controlled must be located between the Skip coil and its paired End coil in the program.

In the example below, the Skip and End coil control the states of outputs O17 and O18. While the Skip coil receives power flow from its conditional logic (in the example, it is contact I1), all output coils within the Skip / End pair hold their last states.

Programming Software Instructions

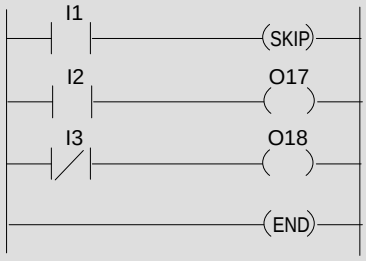
If you are using the programming software, refer to the instructions on page 4-11.

Example and HHP Instructions

Skip (—(SKIP)— |) **End** (—(END)— |)

46025

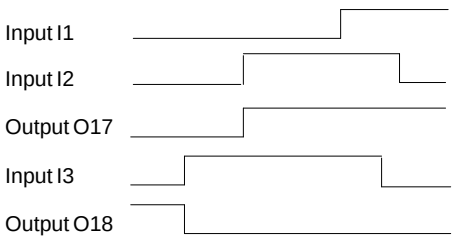
Ladder Diagram



Hand-held Programmer

START		I1
OUT		More MENU F2 (SKIP)
START		I2
OUT		O17
START	F3 (NOT)	I3
OUT		O18
START		
OUT		More MENU F3 (END)

Timing Diagram



This is a Skip to END operation. When input I1 is 0, the Skip command is not performed.

When input I1 is 1, all outputs till END are on hold, and are not affected by input conditions.

Download from Www.Somanuals.com. All Manuals Search And Download.

Timers

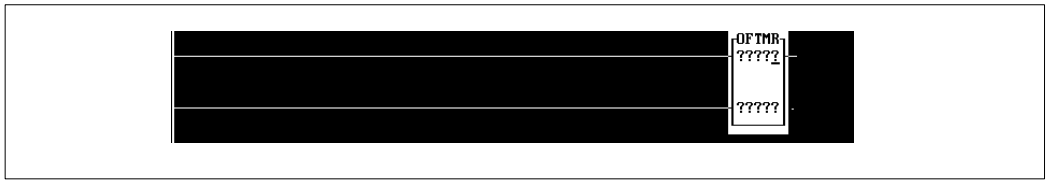
The Micro PLC instruction set provides two Timers:

On Timer	Sets an output to 1 after a specified time period. See page 4-17.
Off Timer	Sets an output to 0 after a specified time period. See page 4-18.

Programming Software Instructions for Timers

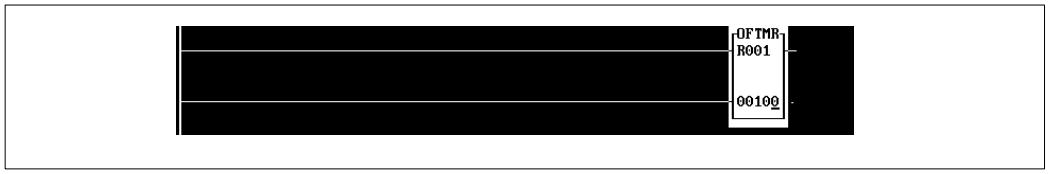
Timers require input logic in the same rung. This can be entered either before selecting the Timer function, or after as described below. Note that the enable line for a timer CANNOT contain parallel branches or instructions. If this type is logic is desired, place it in another rung, using it to drive an internal coil which then drives the timer.

1. **To enter a Timer, select Timer/Counter (F4), then On Timer (F1) or Off Timer (F2).**



2. **For the top parameter, enter a register location for the PLC to store the current value of the timer as it increments.** For example: R001. Then press the Enter key.

3. **For the other parameter, enter the timer length in intervals of 1/10 second.** For example, for a 10-second timer, enter 100.



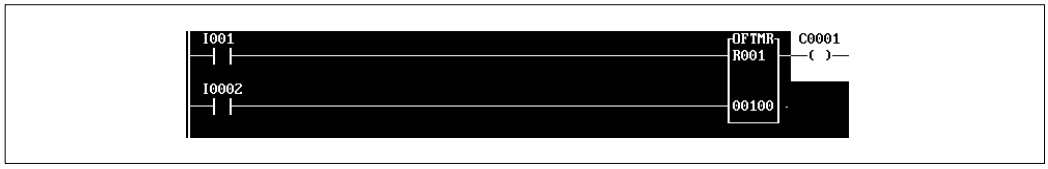
If you want to change a parameter, press the Enter key. Type in the new value and press the Enter key again to accept the new entry.

4. **Enter the other logic required by the Timer:**

   A. Logic in the top line to enable timing. Timing only takes place while this line passes power flow to the Timer.

   B. Logic in the bottom line that will clear the current Timer value to 0 when it passes power flow to the Timer.

   C. An output that will be turned On (for an On Timer) or Off (for a Down Timer) when the Timer reaches the desired value (in this example, 10 seconds).



5. **Use the Accept (F10) key to add the rung to the program.**

Download from Www.Somanuals.com. All Manuals Search And Download.

On Timer

The On Timer turns ON an output after a specified time period. The time period can be from 0 to 6553.5 seconds. However, if a constant is used for the Preset value, the time period can be from 0 to 3276.7 seconds. To program a timer function, enter two inputs (in the example below, I1, and I2), and two parameters (R1 and 100 in the example).

The first input controls operation of the timer. Timing only occurs while I1 is 1. The second input to the timer clears the current timer value to 0.

The first timer parameter is a register memory location for PLC to store the current timer value.

The second timer parameter is the timer length, *in intervals of 0.1 second.* If this number should always be the same, you can enter it directly, as shown in the example. Or, if you want to be able to change the timer length using the Hand-held Programmer, instead enter a register from R385 to R500 as the second parameter for the timer. The current timer value (in R1 below) will not increment beyond its programmed length (time period), even if I1 remains closed and I2 remains open.

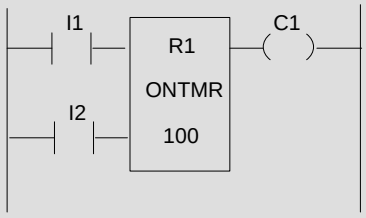
Programming Software Instructions

If you are using the programming software, refer to the instructions on page 4-16.

Example and HHP Instructions

46026

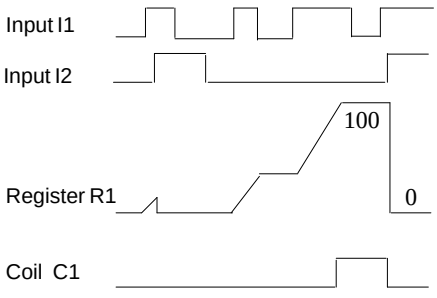
Ladder Diagram



Hand-held Programmer

START		I1
START		I2
TIMER	F1	R1
		100
OUT		C1

Timing Diagram



When I1 is 1, the timer is enabled. The timer setting is:

100 x 0.1 = 10 Sec.

During the operation of the timer, if I1 switches to 0, the timer is on hold. When I1 is 1 again, the timer resumes timing.

When the timer reaches the set value (10 seconds), output coil C1 activates.

Input I2 is used to reset the timer. When both I1 and I2 are on, the timer remains reset.

Download from Www.Somanuals.com. All Manuals Search And Download.

Off Timer

The Off Timer turns OFF an output after a specified time period. The time period can be from 0 to 6553.5 seconds. However, if a constant is used for the Preset value, the time period can be from 0 to 3276.7 seconds.

To program a timer function, enter two inputs (in the example below, I1, and I2), and two parameters (R1 and R385 in the example). In this example, the timer length is not specified directly. Instead, this example uses the register R385 as the second parameter. In this case, register R385 contains the value 100. The timer length can be placed in that register either by program logic or from the Hand-held Progammer or programming software.

When the program encounters a register location as the second parameter of the timer, it looks in that register for the value it contains, and uses that value as the timer length. So for this example, timing is the same as for the previous example.

The current timer value (in R1 above) will not increment beyond its programmed length (time period), even if I1 remains closed and I2 remains open.

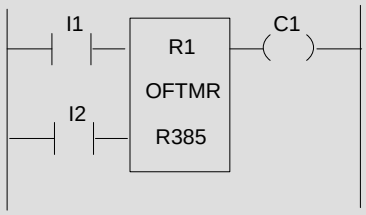
Programming Software Instructions

If you are using the programming software, refer to the instructions on page 4-16.

Example and HHP Instructions

46027

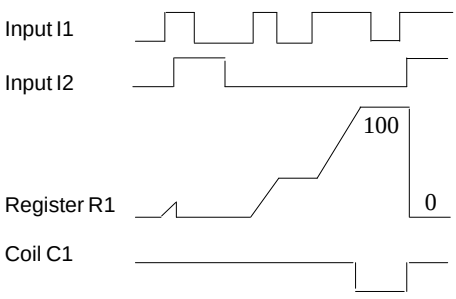
Ladder Diagram



Hand-held Programmer

START		I1
START		I2
TIMER	F1	R1
		R385
OUT		C1

Timing Diagram



When I1 is 1, the timer is started. The timer setting is:

100 x 0.1 = 10 Sec.

During the operation of the timer, if I1 switches to 0, the timer is on hold. When I1 is 1 again, the timer resumes timing,

When the timer reaches the set value (10 seconds), output coil C1 *deactivates.*

Input I2 is used to reset the timer. When both I1 and I2 are ON, the timer remains reset.

Download from Www.Somanuals.com. All Manuals Search And Download.

Counters

The Micro PLC instruction set includes two Counters:

Up Counter	Sets an output to 1 when the counter is equal to a specified value. See page 4-20.
Down Counter	Sets an output to 1 when the counter reaches 0. See page 4-21.

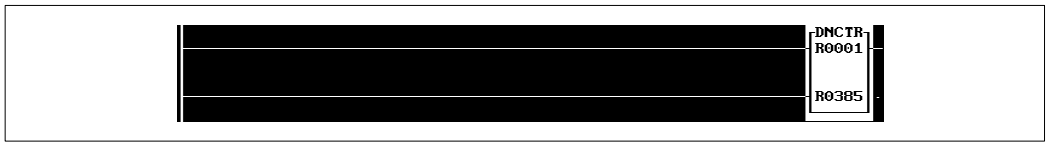
Programming Software Instructions for Counters

Counters require input logic in the same rung. This can be entered either before selecting the Counter function, or after as described below. Note that the count line for a counter CANNOT contain parallel branches or instructions. If this type is logic is desired, place it in another rung, using it to drive an internal coil which then drives the timer.

1. Select Timer/Counter (F4), then Up Counter (F3) or Down Counter (F4).

2. For the top parameter, enter a register location for the PLC to store the current value of the counter as it increments. For example: R001. Then press the Enter key. Register R001, used in this example, is not a “retentive” register; it is cleared if power goes off. If the current count value should be saved with the power off, use a retentive register instead. (R385 - R500)

3. For the other parameter, enter the counter total. If this should always be the same, enter the number. If you want to be able to change the counter total, instead enter a register location. The counter total can then be placed in that register by program logic, from the Hand-held Programmer or by the programming software (if a retentive register is used).



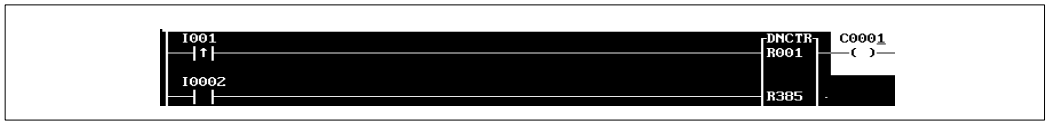
If you want to change a parameter, press the Enter key. Type in the new value and press the Enter key again to accept the new entry.

4. Enter the other logic required by the Counter:

A. Logic in the top line to either increment (Up Counter) or decrement (Down Counter) the count value. The counter counts each scan that this line has power flow.

B. Logic in the bottom line to reset to the count total (to 0 for an Up Counter, to the maximum value for a Down Counter). If both top and bottom lines are active, the counter will be reset.

C. An output that will be turned On when the Timer reaches the desired value.



5. Use the Accept (F10) key to add the rung to the program.

Download from Www.Somanuals.com. All Manuals Search And Download.

Up Counter

The Up Counter turns on an output when the count reaches a specified value (from 0 to 65535 if a register is used to load the counter, or 0 to 32676 if a constant is used). The output remains on only as long as the count is equal to the specified value. If the count increases past the specified value, the output goes back to 0.

To program a counter function, enter two inputs (in the example below, I1, and I2), and two parameters (R1 and 5 in the example).

The first input increments the count value. In this example, each time the Positive Transition contact transitions on, the count total stored in R1 goes up by 1. The second input to the counter resets the count total (here, in R1) to 0. It also deactivates the output coil.

The first parameter of the Up Counter is a register containing a value that is being incremented (usually, by an input in the application program). Register R1, used in this example, is not a “retentive” register; it is cleared if power goes off. If the current count value should be saved with the power off, use a register from R385 to R500.

The second parameter is the counter total. If this number should always be the same, you can enter it directly, as shown in the example. Or, if you want to be able to change the counter total, instead enter a register from R385 to R500 as the second parameter for the timer (see the example Down Counter).

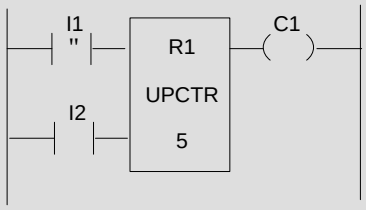
Programming Software Instructions

If you are using the programming software, refer to the instructions on page 4-19.

Example and HHP Instructions

46128

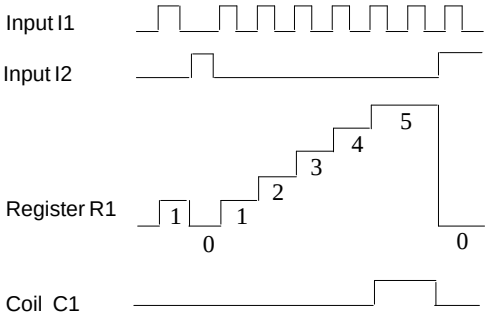
Ladder Diagram



Hand-held Programmer

START	F1	I1
START		I2
CNTR	F1	R1
		5
OUT		C1

Timing Diagram



Each time input I1 transitions on, the content of register R1 is incremented. When it reaches 5, output coil C1 activates.

Input I2 is used to reset counter R1 to 0. It also deactivates coil C1.

Download from Www.Somanuals.com. All Manuals Search And Download.

Down Counter

The Down Counter turns on an output when the count reaches zero (from 65535 to 0 if a register is used to load the counter, or 32767 to 0 if a constant is used). To program a counter function, enter two inputs (in the example below, I1, and I2), and two parameters (R2 and R386 in the example).

The first input decrements the count value. In this example, each time the Positive Transition contact transitions on, the count total stored in R1 goes down by 1. The second input to the counter resets the count total (here, in R1) to its maximum value. It also deactivates the output coil.

The first parameter of the Down Counter is a register containing the value that is being incremented (usually, by an input in the application program). Register R1, used in this example, is not a "retentive" register; it is cleared if power goes off. If the current count value should be saved with the power off, use a register from R385 to R500.

The second counter parameter is the counter total. If this number should always be the same, you can enter it directly, as shown in the example Up Counter. Or, if you want to be able to change the counter total, instead enter a register from R385 to R500 as the second parameter for the counter (see the example). The counter total can then be placed in that register, either by program logic, or from the Hand-held Progammer or programming software.

When the program encounters a register location as the second parameter of the timer, it looks in that register for the value it contains, and uses that value as the timer length. So for this example, timing is the same as for the previous example.

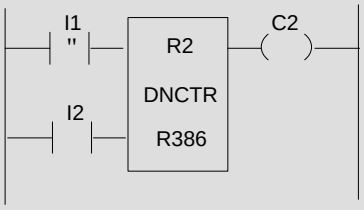
Programming Software Instructions

If you are using the programming software, refer to the instructions on page 4-19.

Example and HHP Instructions

46129

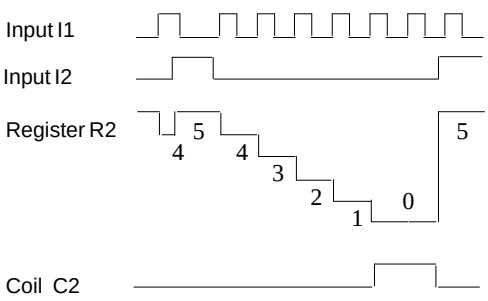
Ladder Diagram



Hand-held Programmer

START	F1	I1
START		I2
CNTR	F2	R2
		R386
OUT		C2

Timing Diagram



Each time input I1 transitions on, the content of register R1 is *decremented.* When it reaches 0, output coil C2 activates.

Input I2 is used to reset counter R2 to 5. It also deactivates coil C2.

Download from Www.Somanuals.com. All Manuals Search And Download.

Math Functions

The Micro PLC instruction set includes these Math functions:

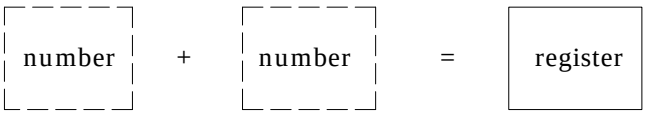
Addition	Adds together two numbers and places the total in a specified register.
Subtraction	The Subtraction function subtracts one number from another and places the result in a specified register.
Multiplication	The Multiply function multiplies one number by another and places the result in two consecutive registers.
Division	The Division function divides one number by another, and places the result in two consecutive registers.

For the Math functions, all numbers are assumed to be positive. There are no negative numbers.

Power flow from a Math function is always true.

Addition (ADD)

The Addition function adds together two numbers and places the total in a specified register.



46130

The numbers added can be constants, as in the second addition shown on the facing page, or the contents of register locations in memory. For example, if R1 contains the value 23 and you add the constant 16 to it, then register R3 will contain the value 39.

Maximum Total

The maximum total for the Addition function is 65535. If the total exceeds 65535, only the excess is placed in the specified register. For example:

1st number		2nd number		Total in register
50,000	+	50,000	=	34464 (excess above 65536)

The program can compare the total with the two numbers that were added. If the total is less than either number, a “rollover” has occurred.

Download from Www.Somanuals.com. All Manuals Search And Download.

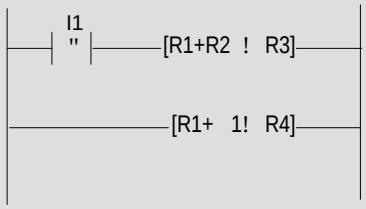
Programming Software Instructions

1. **Select Math/Move (F5).**
2. **Select +ADD (F1) from the Math/Move function keys**.
3. **Enter the first number to be added.** This can be either a constant, or a register location that will contain the number to be added. For example: R001. Then press the Enter key.
4. **Enter the second number to be added**. This can also be a constant or a register location. Press the Enter key.
5. **Enter a register location to contain the total.** Press the Enter key.

Examples and HHP Instructions

46131

Ladder Diagram



Hand-held Programmer

START	F1	I1
MATH	F1	R1
		R2
		R3
OUT	F1 (DROP)	
START		
MATH	F1	R1
		1
		R4
OUT	F1 (DROP)	

In the first rung of the example above, the Positive Transition contact I1 is used as conditional logic to the Addition function. When I1 transitions on, the value in register R2 is added to the value in register R1 and the total is placed in register R3.

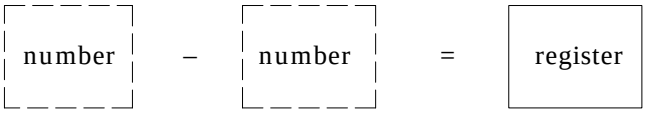
In the second rung of the example, there is no contact placed before the Addition function. That means it is executed unconditionally. Each program scan, 1 is added to the value in R1.

In this example, since R1 is also one of the parameters of the first Addition function, these two rungs of logic work together. Each program scan, the second rung increments the value in R1 that is being added to the contents of R2.

Download from Www.Somanuals.com. All Manuals Search And Download.

Subtraction (SUB)

The Subtraction function subtracts one number from another and places the result in a specified register.



The numbers can be constants, as in the first subtraction shown on the facing page, or the contents of register locations in memory. For example, if R1 contains the value 100 and R4 contains the value 57, then register R3 will contain the value 43.

The second number (the number being subtracted) must be smaller than the first. Other wise the content of the specified register will "roll over" as shown by these examples:

1st number		2nd number		Total in register
100	–	10	=	90
0	–	1	=	65535
0	–	2	=	65534

The program should check to be sure that the second number is smaller than the first before subtracting.

Programming Software Instructions

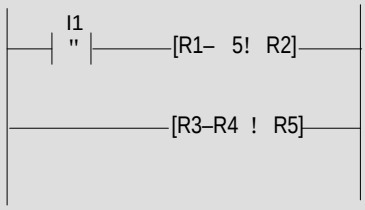
1. **Select Math/Move (F5).**
2. **Select –SUB (F2) from the Math/Move function keys.**
3. **First, enter the number to be subtracted from.** This can be either a constant, or a register location that will contain the number. For example: R001. Then press the Enter key.
4. **Enter the number to be subtracted**. This can also be a constant or a register location. Press the Enter key.
5. **Enter a register location to contain the result.** Press the Enter key.

Download from Www.Somanuals.com. All Manuals Search And Download.

Examples and HHP Instructions

46133

Ladder Diagram



Hand-held Programmer

START		I1
MATH	F2	R1
		5
		R2
OUT	F1	
START		
MATH	F2	R3
		R4
		R5
OUT	F1	

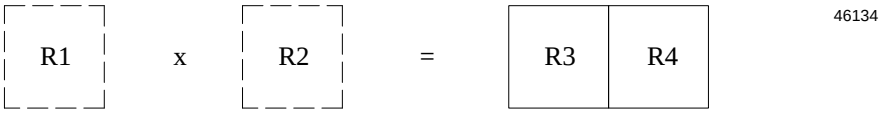
In the first rung of the example above, the Positive Transition contact I1 is used as conditional logic to the Subtraction function. When I1 transitions on, 5 is subtracted from the current value in register R1. The result is placed in register R2.

In the second rung of the example, there is no contact placed before the Subtraction function. That means it is executed unconditionally. Each program scan, the value in R4 is subtracted from the value in R3. The result is placed in R5.

Download from Www.Somanuals.com. All Manuals Search And Download.

Multiplication (MUL)

The Multiply function multiplies one number by another and places the result in *two consecutive registers.*



Two registers are needed to accommodate the larger numbers that might possibly result from the multiplication. Only the first register (R3 above) is actually specified although R3 and R4 will contain the result. Be sure not to use the second register for any other purpose in the program.

The multiplication can be performed on constants or the contents of register locations in memory.

If the result of the multiplication is 65535 or less, it is placed in the *higher numbered register* and the lower numbered register is not used. If your application does not use numbers above 65535 for multiplication or division, you do not need to understand how 32-bit numbers are handled.

Examples:

R1		R2		R3	R4	
10	*	10	=	00000	00100	(0*65536 + 100)
100	*	1000	=	00001	34464	(1*65536 + 34464)
1000	*	1000	=	00015	16960	(15*65536 + 16960)

If you expect the result of the multiplication to be too large to use easily, you can use the Division function on the result. See page 4-29.

Programming Software Instructions

1. **Select Math/Move (F5).**
2. **Select *MUL (F3) from the Math/Move function keys.**
3. **Enter the first number to be multiplied.** This can be either a constant, or a register location that will contain the number to be multiplied. For example: R001. Then press the Enter key.
4. **Enter the second number to be multiplied**. This can also be a constant or a register location. Press the Enter key.
5. **Enter the first register of a two register location to contain the result.** For example, if you entered R003 as shown above, the result of the multiplication would be be located in R003 and R004. Press the Enter key.

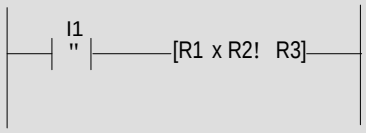
   If the result of the multiplication is 65535 or less, it is placed in the *higher numbered register*, and the lower numbered register (R003 in the example) is set to 0.

Download from Www.Somanuals.com. All Manuals Search And Download.

Example and HHP Instructions

46135

Ladder Diagram



Hand-held Programmer

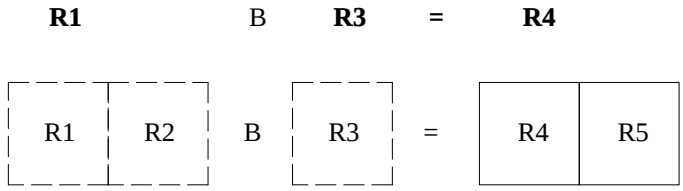
START	F1	I1
MATH	F3	R1
		R2
		R3
OUT	F1	

In the example above, the Positive Transition contact I1 is used as conditional logic to the Multiplication function. When I1 transitions on, the value in register R1 is multiplied by the value in register R2. The result is placed in registers R3 (specified) and R4.

Download from Www.Somanuals.com. All Manuals Search And Download.

Division (DIV)

The Division function divides one number by another, and places the result in *two consecutive registers.*



The numbers can be either constants or the contents of register locations.

When this function is programmed, only the first register of a pair is programmed; that is, the first register of the pair where the result of the division will be placed, and the first register of the number to be divided, (unless it is a constant). Although only the first register of a pair is actually specified, it is important not to use the second register for any other purpose in the program.

Examples:

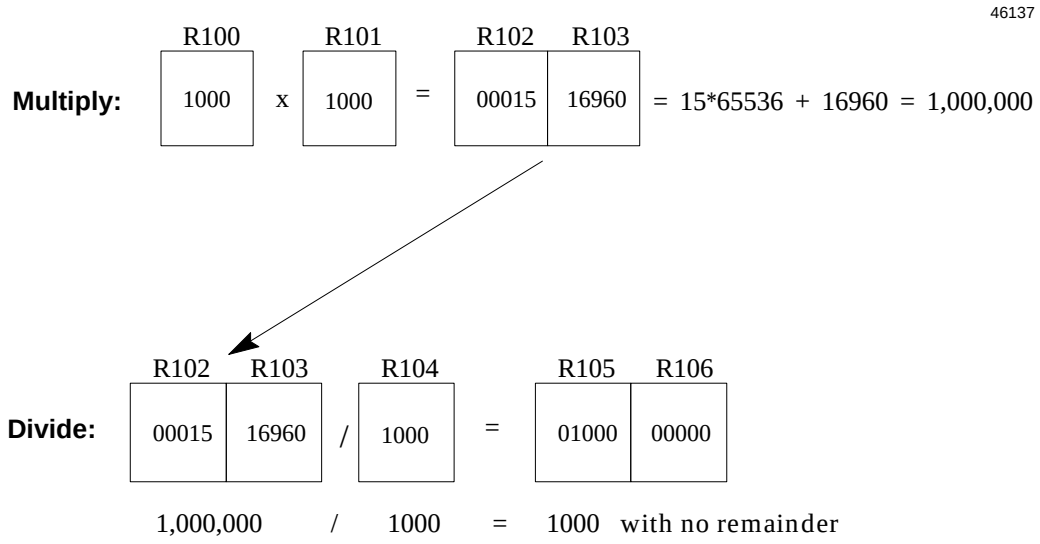
						Whole #	**Remainder**	
R1		**R2**		**R3**		**R4**	**R5**	
00000	01000	/	00010	=	00100	00000		(100 plus no remainder)
00000	01000	/	00012	=	00083	00004		(83 plus 4 remainder)
00001	01000	/	00002	=	33268	00000		(66536 / 2 = 33268, no remainder)
00010	01000	/	01000	=	00656	00360		((655360+1000) / 1000) = 656 plus remainder of 360)
00000	01000	/	03000	=	00000	01000		(1000 / 3000 = 0, remainder 1000)
10000	00000	/	00010	=	00000	00000		(number is too large to represent as a whole number plus a remainder)
00000	01000	/	00000	=	LAST RESULT (but C1023 turns on to indicate that a divide by 0 was incorectly attempted.			

Download from Www.Somanuals.com. All Manuals Search And Download.

Using Division and Multiplication Functions Together

If a Multiplication produces a result that is too large to use easily (the results are greater than 65535, and occupy two registers), you can divide the Multiplication result to scale the result to use smaller numbers.

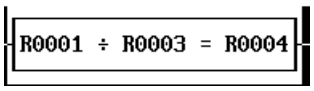
For example:



In this example, R105 would contain the whole number result of the division, and R106 would contain the remainder.

Programming Software Instructions

1. **Select Math/Move (F5).**
2. **Select /DIV (F4) from the Math/Move function keys.**



3. **First, enter the number to be divided.** This can be either a constant, or a *two-register* location that will contain the number to be added. For example: R001 above represents the two-register location R001 and R002. Press the Enter key.
4. **Enter the number to divide by**. This can also be a constant or a *one-register* location. Press the Enter key.
5. **Enter the first register of a two register location to contain the result.** For example, if you entered R004 as shown above, the result of the division would be be located in R004 and R005. Press the Enter key.

Download from Www.Somanuals.com. All Manuals Search And Download.

Move Functions

The Micro PLC instruction set includes these Math functions:

Move	The Move function copies 1 word (16 bits) of data to a specified memory location.
Block Move	The Block Move function copies a selectable amount of data to a specified memory location. It copies more than 16 bits at a time.
Indirect Move	The Indirect Move function copies a constant or the content of register to a *variable* register location.

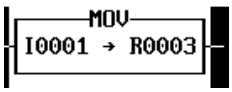
Move

The Move function copies 1 word (16 bits) of data to a specified memory location. The possible choices are:

Register > Register
Input (I) > Register
Register > Coil (C)
Coil (C) > Register
Register > Output (O)

Programming Software Instructions

1. **Select Move (F5) from the Math/Move function keys**.



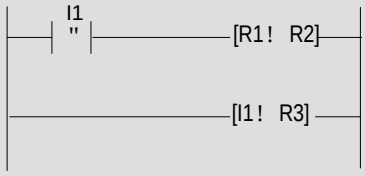
2. **Enter the number to be moved.** This can be a constant, or the contents of a specified bit or register memory location. In the example shown, the Move function copies 16 bits located from I001 through I016. Then press the Enter key.
3. **Enter a memory location for the data to be copied to**. The memory type (bit or register) must be compatible with the data being moved. Press the Enter key. The software will not let you select an incompatible memory type.

Download from Www.Somanuals.com. All Manuals Search And Download.

Examples and HHP Instructions

46138

Ladder Diagram



Hand-held Programmer

START	F1	I1
MOVE	F1	R1
		R2
OUT	F1	
START		
MOVE	F1	
		I1R3
OUT	F1	

In the first example rung above, a transitional contact controls execution of a Move function. That Move function copies the contents of register R1 to register R2.

The second example rung has no conditional logic; it executes each program scan. That Move function copies 16 bits from I1 to I16 into register R3.

Download from Www.Somanuals.com. All Manuals Search And Download.

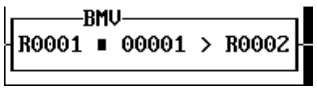
Block Move

The Block Move function copies a selectable amount of data to a specified memory location. It works like a Move function, except that it copies more than 16 bits at a time. The options are explained below:

Option	Example	Description
Register > Register	**R001 0002 > R004**	Copies the contents of R001–R002 into R004–R005.
	R010 R020 > R100	Copies the contents of R010 to R100 in incremental order, according to the length specified in R020. For example, if R020 contained the number 2, then the contents of R010–R011 would be copied onto R100–R101.
Coil (C) > Coil (C)	**C001 0010 > C014**	Copies Internal Relay status coils C001–C010 into C014–C023.
Output (O) > Output (O)	**O001 0016 >O033**	Copies 16 output bits from O001–O016 into O033–O048.
Input (I) > Output (O)	**I004 0016 > O033**	Copies 16 input bits (I001–I019) into outputs O033–O048.
Input (I) > Coil (C)	**I016 0006 > C020**	Copies 6 input bits (I016–I011) into internal coils C010–C025.
Coil (C) > Output (O)	**C001 0010 > O003**	Copies 10 bits from internal coils C001–C010 into outputs O003–O012.

Programming Software Instructions

1. **Select B–Move (F6) from the Math/Move function keys.**



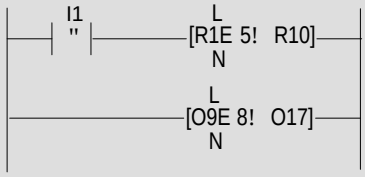
2. **Enter the number to be moved.** This can be a constant, or the contents of a specified memory location (see above). For example: R001. Then press the Enter key.
3. **Enter the data length**. This can be a constant or the contents of a specified register location. The number may represent either bits or words of data, according to the data type being moved.
4. **Enter a memory location to contain the result.** Press the Enter key.

Download from Www.Somanuals.com. All Manuals Search And Download.

Examples and HHP Instructions

46139

Ladder Diagram



Hand-held Programmer

START		I1
MOVE	F2	R1
		5
		R10
OUT	F1	
START		
MOVE	F2	O9
		8
		O17
OUT	F1	

In the first example rung above, when transitional contact I1 goes on, 5 registers (R1 – R5) are copied to locations R10 – R14.

In the second rung, each program scan 128 bits (8 x 16) are copied from O9 – O137 to locations O17 – O124. Since the locations being copied from overlap the locations being copied into, this function actually has the effect of shifting 128 bits eight bits to the “left” (that is, to higher locations) in memory.

Download from Www.Somanuals.com. All Manuals Search And Download.

Indirect Move

The Indirect Move function copies a constant or the content of register to a *variable* register location. The first part of this function specifies the data to be copied. The second part is a register that points to the actual register where the data will be placed. This allows the register where the data is placed to be changed as required for the program.

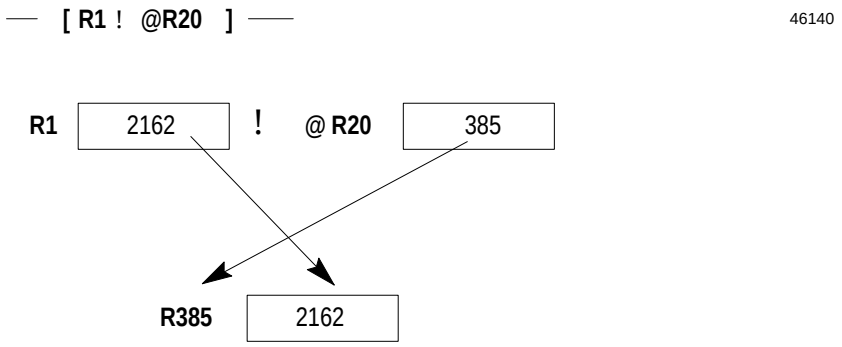
In the example on the facing page, an Indirect Move function copies the content of register R002 into a register which is *pointed to* by register R003. During one program scan, R003 might point to register R020. In another scan, it might point to register R021.

Specifying a Location in the Pointer Register

Ordinarily, the variable register for an Indirect Move function is placed into the pointer register by a Move instruction in the application program. It is also possible to enter a location from the programmer, however.

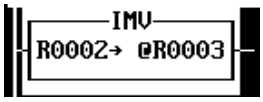
In either case, enter the number of any register from R001 to R500.

Example:



Programming Software Instructions

1. **Select I–Move (F7) from the Math/Move function keys.**



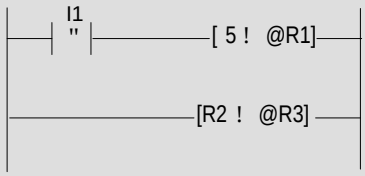
2. **Enter the number to be moved.** This can be a constant, or the contents of a specified memory location (see above). For example: R001. Then press the Enter key.
3. **Enter a memory location to contain the result.** Press the Enter key.

Download from Www.Somanuals.com. All Manuals Search And Download.

Examples and HHP Instructions

46141

Ladder Diagram



Hand-held Programmer

START	F1	I1
MOVE	F3	5
		R1
OUT	F1	
START		
MOVE	F3	R2
		R3
OUT	F1	

In the first example rung above, an Indirect Move function copies the constant 5 into a register *pointed to* by R1. During one program scan, R1 might point to register R20. In another scan, it might point to register R21.

The second example rung has no conditional logic; each program scan, it copies the content of register R2 to a register which is pointed to by register R3.

Download from Www.Somanuals.com. All Manuals Search And Download.

Compare Functions

The Compare functions are used to compare the contents of two registers. They can be used to activate an output, or as conditional logic to other program functions.

The Micro PLC Instruction Set includes these Compare functions:

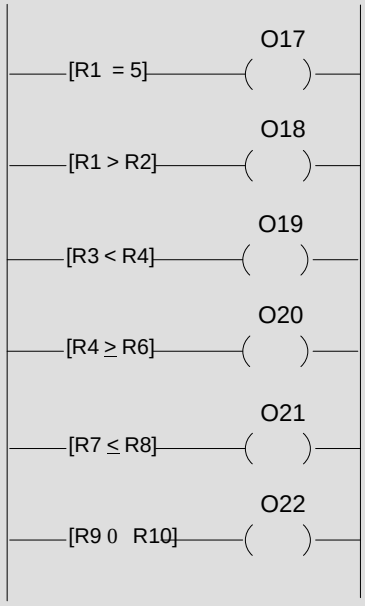
Equal	Passes power flow to the right in the rung if the contents of the two registers are equal.
Greater	Passes power flow to the right in the rung if the number in the first register is greater than the number in the second.
Less	Passes power flow to the right in the rung if the number in the first register is less than the number in the second.
Greater or Equal	Passes power flow to the right in the rung if the number in the first register is greater than or equal to the number in the second.
Less or Equal	Passes power flow to the right in the rung if the number in the first register is less than or equal to the number in the second.
Not Equal	Passes power flow to the right in the rung if the number in the first register is not the same as the number in the second.

An output in the same rung will be set to 1 when the condition being tested for is met.

Examples

46142

Ladder Diagram

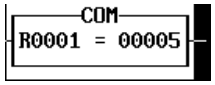
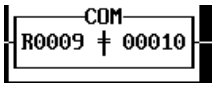
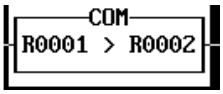
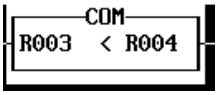
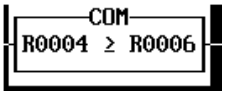
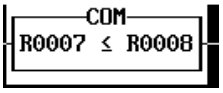


Equal	Output O17 is activated when contents of R1 = 5.
Greater	Output O18 activated when content of R1 is greater than content of R2.
Less	Output O19 activated when content of R3 is less than content of R4.
Greater or Equal	Output O20 activated when content of R4 is greater than or equal to content of R6.
Less or Equal	Output O21 activated when content of R7 is less than or equal to content of R8.
Not Equal	Output O22 activated when content of R9 is not equal to content of R10.

Download from Www.Somanuals.com. All Manuals Search And Download.

Programming Software Instructions

Select **Compare (F6)**, then use the assigned function key(s) to select a Compare function:

= .EQ. (F1)	Equal	
, .NE. (F2)	Not Equal	
> .GT. (F3)	Greater Than	
< .LT. (F4)	Less Than	
≥ .GE. (F5)	Greater Than or Equal	
≤ .LE. (F6)	Less Than or Equal	

1. **Enter the first value to be compared.** It can be a constant or the contents of a register. Then press the Enter key.
2. **Enter the second value to be compared.** This also can be a constant or the contents of a register. Press the Enter key.
3. **Press Enter to add the function to the program.**

Compare functions often are used in combination with output coils as shown below.

In this example, Output O022 is turned on when the content of register R009 is not equal to the constant 10.

Download from Www.Somanuals.com. All Manuals Search And Download.

Logic Operations

The Micro PLC Instruction Set includes these Logic operations:

Word AND	The AND function compares *each bit* in one register against each bit in another. If both bits are 1, it places a 1 in the corresponding bit of a third register.
Inclusive OR	The Inclusive OR function compares each bit in one register against each bit in another. If *either or both* bits are 1, it places a 1 in the corresponding bit of a third register.
Exclusive OR	The XOR function compares each bit in one register against each bit in another. If both bits are the same (1 or 0), it places a 0 in the corresponding bit of a third register
Shift Register Right	Shift Register Right shifts all the bits in a register to the right (toward the least significant bits) by a specified number of positions.
Shift Register Left	Shift Register Left shifts all the bits in a register to the rleft (toward the most significant bits) by a specified number of positions.
NOT	The NOT function takes the *opposite* state of each bit in a register and places it into a second register.

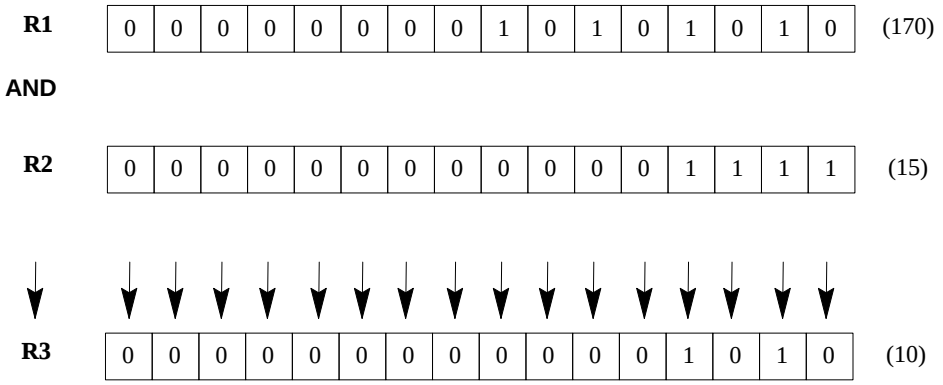
Download from Www.Somanuals.com. All Manuals Search And Download.

Word AND

The AND function compares *each bit* in the first specified register against each corresponding bit in the second specified register. If both bits are 1, it places a 1 in the corresponding bit of the third register. If either or both of the corresponding bits is 0, the AND function places a 0 in the corresponding location in the third register.

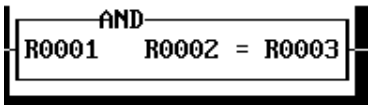
Example:

46143



Programming Software Instructions

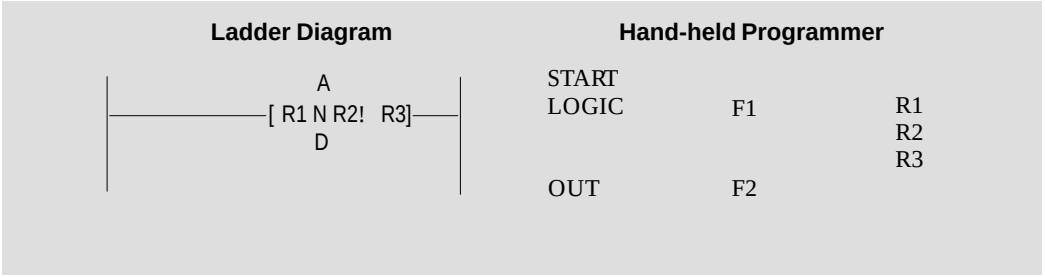
1. **Select Logic (F7)**
2. **Select AND (F1) from the Logic function keys.**



3. **Enter the register location of the first number to be ANDed.** Press the Enter key.
4. **Enter the register location of the second number to be ANDed.** Press the Enter key.
5. **Enter a register for the result AND to be placed in**. Press the Enter key.

Example and HHP Instructions

46144



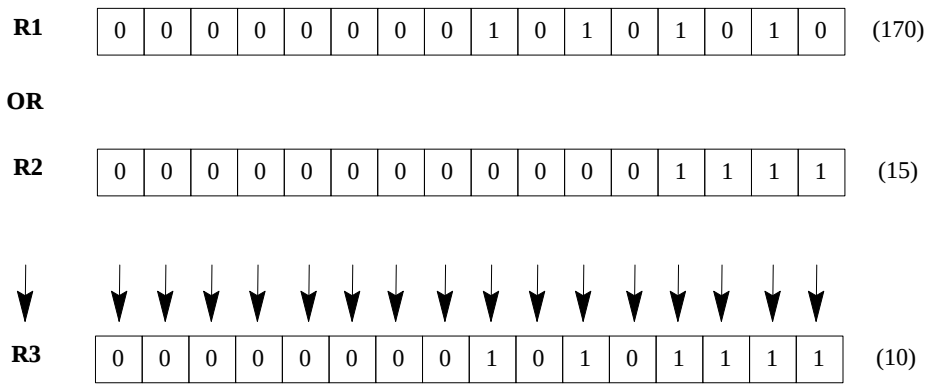
Download from Www.Somanuals.com. All Manuals Search And Download.

Inclusive OR (IOR)

The Inclusive OR function compares each bit in the first specified register against each corresponding bit in the second specified register. If *either or both* bits are 1, it places a 1 in the corresponding bit of the third register. If both of the corresponding bits is 0, the OR function places a 0 in the corresponding location in the third register.

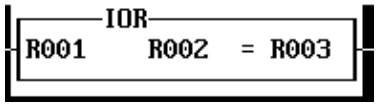
Example:

46145



Programming Software Instructions

1. **Select Logic (F7)**
2. **Select IOR (F2) from the Logic function keys.**

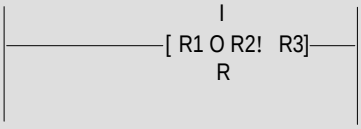


3. **Enter the register location of the first number to be IORed.** Press the Enter key.
4. **Enter the register location of the second number to be IORed.** Press the Enter key.
5. **Enter a register for the result to be placed in**. Press the Enter key.

Example and HHP Instructions

46146

Ladder Diagram



Hand-held Programmer

START LOGIC	F2	R1 R2 R3
OUT	F1	

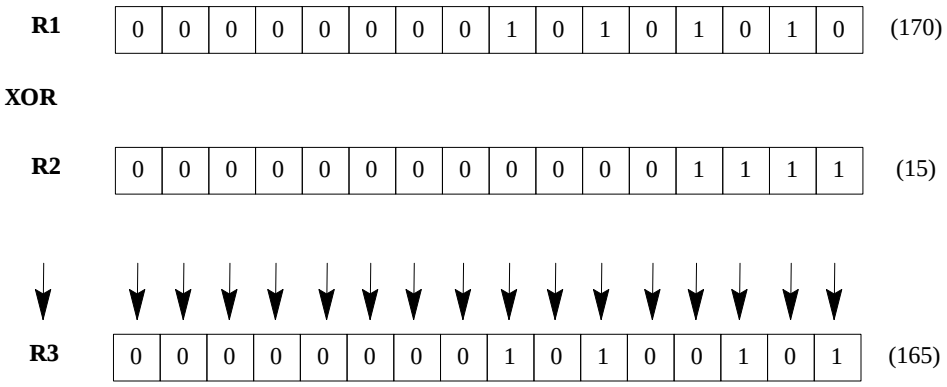
Download from Www.Somanuals.com. All Manuals Search And Download.

Exclusive OR (XOR)

The XOR function compares each bit in the first specified register against each corresponding bit in the second specified register. If both bits are the same (1 or 0), it places a 0 in the corresponding bit of the third register. If the bits are different, it places a 1 in the corresponding location in the third register.

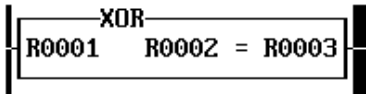
Example:

46147



Programming Software Instructions

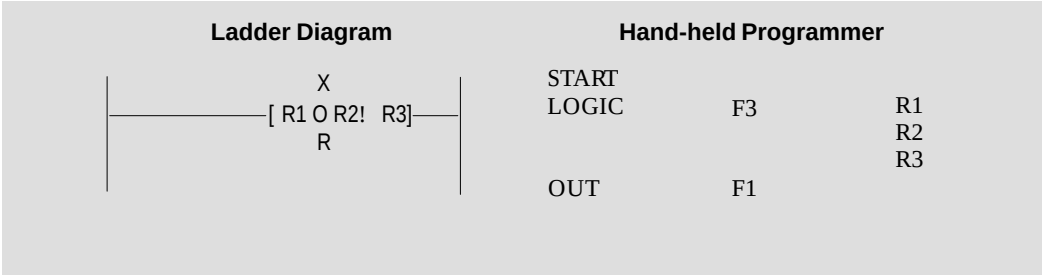
1. **Select Logic (F7)**
2. **Select XOR (F3) from the Logic function keys.**



3. **Enter the register location of the first number to be XORed.** Press the Enter key.
4. **Enter the register location of the second number to be XORed.** Press the Enter key.
5. **Enter a register for the result to be placed in**. Press the Enter key.

Example and HHP Instructions

46148



Download from Www.Somanuals.com. All Manuals Search And Download.

Shift Register Right

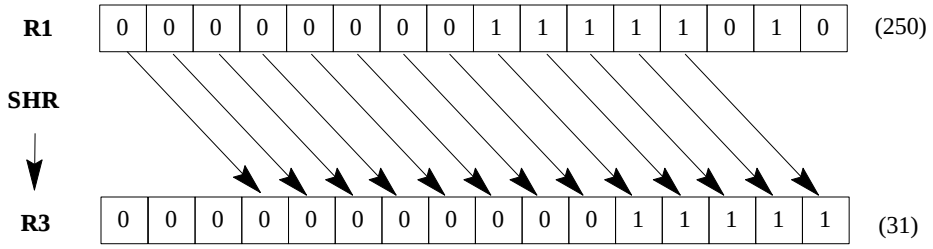
Each scan that power flow is received, the Shift Register Right function shifts all the bits in a register to the right (toward the least significant bits). The number of positions to be shifted is provided in the function's second register (R2 in the example above).

The data is copied to the register specified (R3 in the example above). The original data is not altered. In the copied data, the specified number of bits is "shifted out" on the right, and a corresponding number of zeros is moved in on the left.

Example:

For this example, if the second register (R2) contained the value 3, the bits would be shifted right by 3 positions:

46149

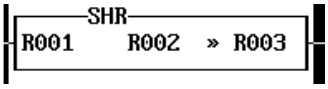


Entering the Number of Positions to Shift

The best way to enter the number of positions for the shift into the second register is with a Move command in the application program. Alternatively, you could use the programmer to manually place the number into a retentive register (R385 – R500).

Programming Software Instructions

1. **Select Logic (F7)**
2. **Select SH–REG RIGHT (F5) from the Logic function keys.**

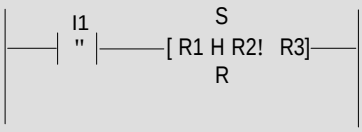


3. **Enter the register location of the data to be shifted right.** Press the Enter key.
4. **Enter the register location for the number of bits to shift.** Press the Enter key.
5. **Enter a register for shifted data to be placed in**. Press the Enter key.

Example and HHP Instructions

4615(

Ladder Diagram



Hand-held Programmer

START	F1	I1
LOGIC	More MENU	
	F2	R1
		R2
		R3
OUT	F1	

Download from Www.Somanuals.com. All Manuals Search And Download.

Shift Register Left

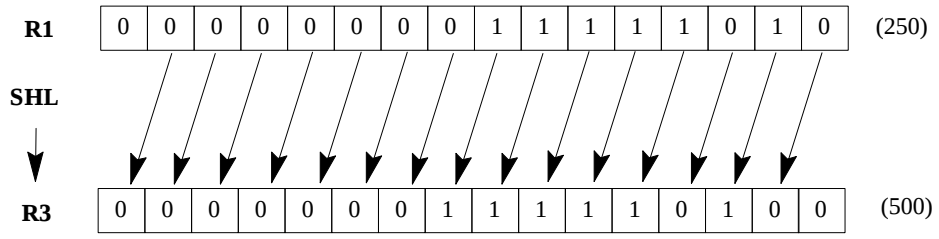
Each scan, the Shift Register Left function shifts all the bits in a register to the left (that is, toward the most significant bit). The number of positions to be shifted is provided in the function's second register (R2 in the example above).

The data is copied to the register specified (R3 in the example above). The original data is not altered. In the copied data, the specified number of bits is "shifted out" on the left, and a corresponding number of zeros is moved in on the right.

Example

For this example, if the second register (R2) contained the value 1, the bits would be shifted left by 1 position:

46151

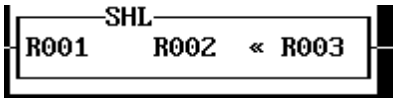


Entering the Number of Positions to Shift

The best way to enter the number of positions for the shift into the second register is with a Move command in the application program. Alternatively, you could use the programmer to manually place the number into a retentive register.

Programming Software Instructions

1. **Select Logic (F7)**
2. **Select SH–REG LEFT (F6) from the Logic function keys.**

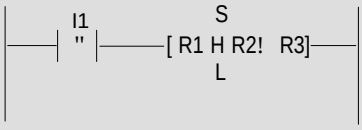


3. **Enter the register location of the data to be shifted left.** Press the Enter key.
4. **Enter the register location for the number of bits to shift.** Press the Enter key.
5. **Enter a register for shifted data to be placed in**. Press the Enter key.

Example and HHP Instructions

46152

Ladder Diagram



Hand-held Programmer

START	F1	I1
LOGIC	More MENU	
	F3	R1
		R2
		R3
OUT	F1	

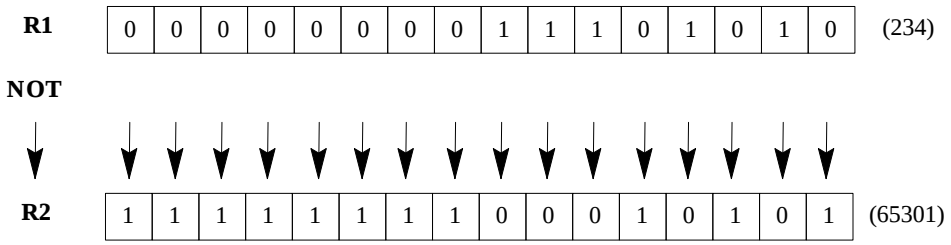
Download from Www.Somanuals.com. All Manuals Search And Download.

NOT

The NOT function takes the *opposite* state of each bit in the first specified register and places it into the second register.

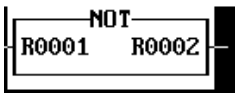
Example:

46153



Programming Software Instructions

1. **Select Logic (F7)**
2. **Select NOT (F4) from the Logic function keys.**

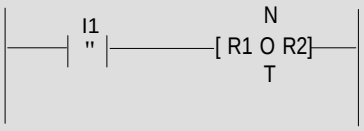


3. **Enter the register location of the first number to be NOTed.** Press the Enter key.
4. **Enter the register location of the second number to be NOTed.** Press the Enter key.
5. **Enter a register for the result to be placed in**. Press the Enter key.

Example and HHP Instructions

46154

Ladder Diagram



Hand-held Programmer

START	F1	I1
LOGIC	More MENU	
	F1	R1
		R2
OUT	F1	

Download from Www.Somanuals.com. All Manuals Search And Download.

Appendix A Using Directories

This section gives some advice on organizing the Micro PLC programming directory structure on your hard disk.

If you installed the programming software using the default values, the programming files are in a directory named **\MICRO** on your disk. You also have the following directories:

\MICRO
MICRO.EXE the programming software
MICRO.CFG a configuration file that contains information about your serial port, display, and data tables format.
\COMM
\DDE

Loading and Saving Files

When using the Load or Save command, you need to enter a filename. You can either:

A. Enter a filename without a path:

PROG1

In this case, **PROG1** is assumed to be in the present directory. If you started running MICRO.EXE in the \MICRO directory, the file PROG1 is really assumed to be C:\MICRO\PROG1, etc...

OR:

B. Enter a path with the filename:

C:\MICRO\RIVETER\PROG1

When you specify the entire filename including drive, directories, and filename, it doesn't make any difference which directory you are in when you Load and Save.

Download from Www.Somanuals.com. All Manuals Search And Download.

Using the Change Directory Function

Most of the time, you won't need to use the Change Directory function. The Change Directory function is normally used to change the default directory for Load and Save file commands. This is useful if you repeatedly need to load or save from a diskette, or if you need to repeatedly load or save from a directory that you didn't start running MICRO from.

Even if you don't use the Change Directory function, you can still load and save from other directories. You will need to type in the entire path name for the file.

For example, if you have "changed directories" to C:\MISC\OTHER\STUFF, when you save a file, the typed filename would be:

TESTFILE.PLC

If you have not changed directories, the typed filename would be:

C:\MISC\OTHER\STUFF\TESTFILE.PLC

The MICRO.CFG File

Whenever you Save configuration in the Setup menu, a MICRO.CFG file is written to the current directory. If you are in the \MICRO directory, MICRO.CFG is written there. If you are in another directory, then MICRO.CFG is written there instead. You can have as many MICRO.CFG files as you want, but each must be in a separate directory.

The MICRO.CFG file contains information about the video characteristics of your computer and your serial port. It also contains the format of your online data displays. This file is manually loaded from the current directory when you use the Restore function from the Setup menu. It is automatically loaded when you start up the programming software from a directory which has a resident MICRO.CFG file.

Hints for a Basic Application

If you will not be creating a large number of programs, or if you don't want to use directories, start up the software by entering:

C:>CD \MICRO
C:\MICRO>MICRO

When you start the software in this way, the MICRO.CFG file (configuration file) that is already saved in the \MICRO directory is used to set the video mode, serial port selection, and online data display format. You can change the format at any time, then Save the new format.

When in the Disk menu, you can use Load and Save to the present directory(\MICRO), and not need to change directories. If you want to save a file to diskette or load a file from diskette, enter the diskette drive name as part of the filename. For example:

A:1STPROG.PLC

Download from Www.Somanuals.com. All Manuals Search And Download.

Hints for an Advanced Application

If you plan to create a large number of Micro PLC programs, it makes sense to create separate directories with a separate program or group of related programs in each. This is a convenient way to keep track of your files.

Also, you then have the capability of saving a different online data display format for each directory. This can be useful if you have a specific online data display related to a program. For example, you may want to use multiple online windows, with a specific reference at the start of each window. The procedure would be:

1. From DOS, create directories to the \MICRO directory using the MD command. For example:

   C:>MD\MICRO\PALLET
   C:>MD\MICRO\RIVETER
   C:>MD\MICRO\TRANSFER

2. Then, go to the directory which has the program to be edited *before* starting the programming software.

   C:>CD\MICRO\PALLET
   C\MICRO\PALLET> MICRO

3. Create the new program (for example: PALLET1.PLC) and save it to disk.
4. Set up the online data display format for the program.
5. Go to the Setup menu and Save this configuration. This puts a MICRO.CFG file with the present online data display format in the directory you are now in (C:\MICRO\PALLET in this example).
6. Now, every time you start up the software as shown in step 2 above, the online data display you created for that program is used automatically.
7. Similarly, you could create unique data displays for each directory.

If you set up your system in this manner, you can change from one directory to another in either of the following ways:

A. Exit from the programming software, change to the \MICRO\RIVETER directory and type MICRO. This will automatically load the \RIVETER online data display format, and allow you to directly Load and Save the RIVETER files.

B. Use the Change Directory function from the Disk menu to get to the \RIVETER directory. This allows you to Load and Save the RIVETER files, but you need to separately Restore the online data display format from the Setup menu.

Download from Www.Somanuals.com. All Manuals Search And Download.

Appendix B Micro PLC Protocol

The information in this appendix is provided only for advanced users requiring communications between the Micro PLC and a host system.

- Communications Files
- Memory Types and Addresses
  - I/OMemory Addresses
  - Work Memory
  - Program and I/O Address Map
- Communications Protocol
- Data Format
  - Read Discretes
  - Read Analogs
  - Read Program Memory
  - Write Discretes
  - Write Analogs
  - Write Program Memory
  - Read Status
  - Start Program
  - Stop Program
  - Status Word
  - Error Reply
- Communications Functions
  - Microsoft C (Large Model: Compile w/ -AL Option)
  - Microsoft C (Small Model: Compile w/ -AS Option)
  - Turbo C (Large Model: Compile w/ -ml Option)
  - Turbo C (Small Model: Compile w/ -ms Option)
  - IBM Compiler BASIC
  - Sample Programs

Download from Www.Somanuals.com. All Manuals Search And Download.

Communications Files

The programming software diskette contains the following communications drivers and demonstration programs. If you used the automated install routine for the Programming software, these files were placed in the \comm subdirectory under the main / micro directory.

These files are provided "as is". Please do not call for technical support on these files.

MCROCOMM.C	This is driver code in Turbo C / C++, which can be used to communicate with the Micro PLC. This file should generally be included in other C files that perform the data acquisition.
DEMO1.C	This is source code which shows a simple application of the MCROCOMM.C driver written in Turbo C.
DEMO1.EXE	The executable form of the DEMO1.C file. Try this as is before modifying and recompiling.
COMPILE1.BAT	This is a simple BAT file to compile the DEMO1.C file under Turbo C. You may need to modify the TCC command line to account for differences in your hard drive and compiler file locations.
MICROBAR.C	This is source code for a more complicated application. This file also "includes" MCROCOMM.C at compile time.
MICROBAR.EXE	This is the executable file for MICROBAR.C. Try this before making changes, and recompiling.
COMPBAR.BAT	This batch file compiles MICROBAR.C. It has two command line switches. Print out the file to see. The TCC command line may need to be modified to account for your hard drive and compiler file locations.
OPT??.RES	These files are used by the COMPBAR.BAT file.
EGAVGA.OBJ	This is a video driver for EGA and VGA modes. This is used by the COMPBAR batch file during compilation.

Download from Www.Somanuals.com. All Manuals Search And Download.

Communications Memory Types and Addresses

I/O Memory Addresses

Type	Code	Max	Address Range (hex)
Discrete Input	I	256	0000 – 00FF
Discrete Output	O	256	0100 – 01FF
Forced Discrete Input	FI	256	0200 – 02FF
Forced Discrete Output	FO	256	0300 – 03FF
Internal Coil	C	1024	0400 – 07FF
Registers (includes the next two items)			
Analog Input	IR	256	0000 – 01FE *
Analog Output	OR		0200 – 03FE *
Internal Registers	R	512	0400 – 07FE *

* Byte address with Least Significant Bit always 0.

Special Registers

Registers 501 to 512 are reserved. Registers 510 to 512 are used for communications setup.

Register	Function	Description
510	Protocol	0 = Micro PLC (default) 1 = RTU
511	Station Address	1 (default) to 247
512	Baud rate	0 = 300 1 = 600 2 = 1200 3 = 2400 4 = 4800 5 = 9600

Communications Parameters

Baud Rate	9600 (default)
Data Bits	8
Stop Bit	1
Parity	None

Download from Www.Somanuals.com. All Manuals Search And Download.

Communications Protocol

This is the low level definition of the serial communications Protocol. A driver for the most commonly used portions of the protocol has already been written for C, and compiler Basic. This driver is contained in the MCROCOM.C file from the distribution diskette. Sample C programs which use this driver are provided. Refer to the MCROCOMM.C file and examples in the \COMM subdirectory of your disk.

DLE STX data1 data2 ... DLE ETX cc

DLE = 16
STX = 2
ETX = 3

data N – Data byte. If the data is equal to DLE (16), two DLE bytes will be transmitted. Maximum of 255 data bytes (excluding DLE bytes that were inserted).

cc – Checksum byte. 2's complement of 8 bit sum of data bytes only (excluding DLE bytes that were inserted and the DLE / STX DLE / ETX).

Data Format

NOTES:

- All addresses are in the range of 11 bits. Most significant 5 bits are 0s.
- Bit values are packed into bytes (byte 0 bit 0, byte 0 bit 1, ..., byte 1 bit 0, etc.).
- Analog values, addresses, and program words are 2 bytes each, MSB first.
- PLC id value is ignored by the programming port, but it shouldn't be dropped.
- Program write can be done only when the PLC is in Program mode (stopped).
- When downloading a program in more than one block, the blocks must be in ascending address order.

Read Discretes

ID 01 count addr_high addr_low

Reply:

ID 01 count addr_high addr_low data1 data2 ...

ID	– PLC ID number
count	– number of bits (discretes) to read
addr_high	– High byte of address of the first discrete to read
addr_low	– Low byte of address of the first discrete to read
dataN	– Data read

Download from Www.Somanuals.com. All Manuals Search And Download.

Read Analogs

ID 02 count addr_high addr_low

Reply:

ID 02 count addr_high addr_low data1 data2 ...

ID	– PLC ID number
count	– number of words (registers) to read
addr_high	– High byte of address of the first register to read
addr_low	– Low byte of address of the first register to read
dataN	– Data read

Read Program Memory

ID 03 count addr_high addr_low

Reply:

ID 03 count addr_high addr_low data1 data2 ...

ID	– PLC ID number
count	– number of instruction words to read
addr_high	– High byte of address of the first instruction to read
addr_low	– Low byte of address of the first instruction to read
dataN	– Data read

Write Discretes

ID 04 count addr_high addr_low data1 data2 ...

Reply:

ID 04 count addr_high addr_low

ID	– PLC ID number
count	– number of bits (discretes) to write
addr_high	– High byte of address of the first discrete to write
addr_low	– Low byte of address of the first discrete to write
dataN	– Data to write

Write Analogs

ID 05 count addr_high addr_low data1 data2 ...

Reply:

ID 05 count addr_high addr_low

ID	– PLC ID number
count	– number of words (registers) to write
addr_high	– High byte of address of the first register to write
addr_low	– Low byte of address of the first register to write
dataN	– Data to write

Download from Www.Somanuals.com. All Manuals Search And Download.

Write Program Memory

ID 06 count addr_high addr_low data1 data2 ...

Reply:

ID 06 count addr_high addr_low

ID	– PLC ID number
count	– number of instruction words to write
addr_high	– High byte of address of the first instruction to write
addr_low	– Low byte of address of the first instruction to write
dataN	– Data to write

Read Status

ID 07

Reply:

ID 07 stat_high stat_low pgm_siz_h pgm_siz_l mod0 mod1 ... mod13

ID	– PLC ID number
stat_high	– High byte of status word
stat_low	– Low byte of status word
pgm_siz_h	– High byte of program size (in words)
pgm_siz_l	– Low byte of program size
modN	– Type of module in slot N. It may be:

0	– not installed
1	– 24VDC Input 16 pts
2	– 115VAC Relay Output 8 pts

Status Word:

bits 0 – 7 are for fatal conditions (stopping the PLC)
bits 8 – 11 are for non-fatal conditions
bit 15 is informational

0	memory is corrupted
1	invalid program
2	invalid instruction encountered
3	(runtime error (such as Indirect Move to invalid address)
4 – 6	reserved
7	forced stop (PLC stopped by error or by programmer)
8	I/Oproblem
9	system timer malfunction
10 – 14	reserved
15	Program mode (1 if in Program mode)

Download from Www.Somanuals.com. All Manuals Search And Download.

Start Program

ID 08

Reply:

ID 08 stat_high stat_low

stat_high	– High byte of status word
stat_low	– Low byte of status word

Status Word:

bits 0 – 7 are for fatal conditions (stopping the PLC)
bits 8 – 11 are for non-fatal conditions
bit 15 is informational

0	memory is corrupted
1	invalid program
2	invalid instruction encountered
3	(runtime error (such as Indirect Move to invalid address)
4 – 6	reserved
7	forced stop (PLC stopped by error or by programmer)
8	I/Oproblem
9	system timer malfunction
10 – 14	reserved
15	Program mode (1 if in Program mode)

Stop Program

ID 09

Reply:

ID 09 stat_high stat_low

stat_high	– High byte of status word
stat_low	– Low byte of status word

Error Reply

ID 8x error_code

8x is the command code to which this is the reply plus 128 (80h).

error codes:

1	Receive buffer overflow
2	Checksum error
3	Illegal command code
4	Illegal command format
5	Block size too big or zero
6	Cannot modify program while running
7	Cannot run while switch is in PROG position
8	Address out of range

Download from Www.Somanuals.com. All Manuals Search And Download.

Communications Functions

The MCROCOMM.C file contains a driver that implements the following functions. Note that a driver function is not available for every feature supported by the underlying protocol.

Microsoft C (Large Model: Compile w/ -AL Option)

- Function : Read Register
  Call : int MCL_Rreg (port, memtype, addr, count, buf);
  Inputs : int port; /* 0–COM1, 1–COM2 */
  int memtype; /* 0–IR, 1–OR, 2–R */
  int addr; /* Adrs. of the 1st register to read */
  int count; /* Number of words to read */
  Return : int *buf; /* Data read */
  True / False

- Function : Write Register
  Call : int MCL_Wreg (port, memtype, addr, count, buf);
  Inputs : int port; /* 0–COM1, 1–COM2 */
  int memtype; /* 1–OR, 2–R */
  int addr; /* Adrs. of the 1st register to write */
  int count; /* Number of words to write */
  : int *buf; /* Data to write */
  Return True / False

- Function : Read Bit
  Call : int MCL_Rbit (port, memtype, addr, count, buf);
  Inputs : int port; /* 0–COM1, 1–COM2 */
  int memtype; /* 0–DI, 1–DO, 4–C */
  int addr; /* Adrs. of the 1st discrete to read */
  int count; /* Number of bits to read */
  Return : char *buf; /* Data read */
  True / False

- Function : Write Bit
  Call : int MCL_Wbit (port, memtype, addr, count, buf);
  Inputs : int port; /* 0–COM1, 1–COM2 */
  int memtype; /* 1–DO, 4–C */
  int addr; /* Adrs. of the 1st discrete to write */
  int count; /* Number of bits to write */
  char *buf; /* Data to write */
  Return True / False

Download from Www.Somanuals.com. All Manuals Search And Download.

Microsoft C (Small Model: Compile w/ -AS Option)

- Function : Read Register

```
Call    : int   MCS_Rreg (port, memtype, addr, count, buf);
Inputs  : int   port;       /* 0–COM1, 1–COM2                */
          int   memtype;    /* 0–IR, 1–OR, 2–R               */
          int   addr;       /* Adrs. of the 1st register to read */
          int   count;      /* Number of words to read       */
Return  : int   *buf;       /* Data read                     */
          True / False
```

- Function : Write Register

```
Call    : int   MCS_Wreg (port, memtype, addr, count, buf);
Inputs  : int   port;       /* 0–COM1, 1–COM2                */
          int   memtype;    /* 1–OR, 2–R                     */
          int   addr;       /* Adrs. of the 1st register to write */
          int   count;      /* Number of words to write      */
        : int   *buf;       /* Data to write                 */
Return    True / False
```

- Function : Read Bit

```
Call    : int   MCS_Rbit (port, memtype, addr, count, buf);
Inputs  : int   port;       /* 0–COM1, 1–COM2                */
          int   memtype;    /* 0–DI, 1–DO, 4–C               */
          int   addr;       /* Adrs. of the 1st discrete to read */
          int   count;      /* Number of bits to read        */
Return  : char  *buf;       /* Data read                     */
          True / False
```

- Function : Write Bit

```
Call    : int   MCS_Wbit (port, memtype, addr, count, buf);
Inputs  : int   port;       /* 0–COM1, 1–COM2                */
          int   memtype;    /* 1–DO, 4–C                     */
          int   addr;       /* Adrs. of the 1st discrete to write */
          int   count;      /* Number of bits to write       */
          char  *buf;       /* Data to write                 */
Return    True / False
```


Download from Www.Somanuals.com. All Manuals Search And Download.

Turbo C (Large Model: Compile w/ -ml Option)

- Function : Read Register
  Call : int TCL_Rreg (port, memtype, addr, count, buf);
  Inputs : int port; /* 0–COM1, 1–COM2 */
  int memtype; /* 0–IR, 1–OR, 2–R */
  int addr; /* Adrs. of the 1st register to read */
  int count; /* Number of words to read */
  Return : int *buf; /* Data read */
  True / False

- Function : Write Register
  Call : int TCL_Wreg (port, memtype, addr, count, buf);
  Inputs : int port; /* 0–COM1, 1–COM2 */
  int memtype; /* 1–OR, 2–R */
  int addr; /* Adrs. of the 1st register to write */
  int count; /* Number of words to write */
  : int *buf; /* Data to write */
  Return True / False

- Function : Read Bit
  Call : int TCL_Rbit (port, memtype, addr, count, buf);
  Inputs : int port; /* 0–COM1, 1–COM2 */
  int memtype; /* 0–DI, 1–DO, 4–C */
  int addr; /* Adrs. of the 1st discrete to read */
  int count; /* Number of bits to read */
  Return : char *buf; /* Data read */
  True / False

- Function : Write Bit
  Call : int TCL_Wbit (port, memtype, addr, count, buf);
  Inputs : int port; /* 0–COM1, 1–COM2 */
  int memtype; /* 1–DO, 4–C */
  int addr; /* Adrs. of the 1st discrete to write */
  int count; /* Number of bits to write */
  char *buf; /* Data to write */
  Return True / False

Download from Www.Somanuals.com. All Manuals Search And Download.

Turbo C (Small Model: Compile w/ -ms Option)

- Function : Read Register

```
Call    : int    TCS_Rreg (port, memtype, addr, count, buf);
Inputs  : int    port;      /* 0–COM1, 1–COM2                    */
          int    memtype;   /* 0–IR, 1–OR, 2–R                   */
          int    addr;      /* Adrs. of the 1st register to read */
          int    count;     /* Number of words to read           */
Return  : int    *buf;      /* Data read                         */
          True / False
```

- Function : Write Register

```
Call    : int    TCS_Wreg (port, memtype, addr, count, buf);
Inputs  : int    port;      /* 0–COM1, 1–COM2                    */
          int    memtype;   /* 1–OR, 2–R                         */
          int    addr;      /* Adrs. of the 1st register to write*/
          int    count;     /* Number of words to write          */
        : int    *buf;      /* Data to write                     */
Return    True / False
```

- Function : Read Bit

```
Call    : int    TCS_Rbit (port, memtype, addr, count, buf);
Inputs  : int    port;      /* 0–COM1, 1–COM2                    */
          int    memtype;   /* 0–DI, 1–DO, 4–C                   */
          int    addr;      /* Adrs. of the 1st discrete to read */
          int    count;     /* Number of bits to read            */
Return  : char   *buf;      /* Data read                         */
          True / False
```

- Function : Write Bit

```
Call    : int    TCS_Wbit (port, memtype, addr, count, buf);
Inputs  : int    port;      /* 0–COM1, 1–COM2                    */
          int    memtype;   /* 1–DO, 4–C                         */
          int    addr;      /* Adrs. of the 1st discrete to write*/
          int    count;     /* Number of bits to write           */
          char   *buf;      /* Data to write                     */
Return    True / False
```


Download from Www.Somanuals.com. All Manuals Search And Download.

IBM Compiler BASIC

- Function : Read Register

```
Call    : int   BASRreg (port%, memtype%, addr%, count%, buf%(0),
                status%);
Inputs  : int   port;       /* 0–COM1, 1–COM2                  */
          int   memtype;    /* 0–IR, 1–OR, 2–R                 */
          int   addr;       /* Adrs. of the 1st register to read */
          int   count;      /* Number of words to read         */
Return  : int   *buf;       /* Data read                       */
          int   status;     /* Error status–0: Normal          */
```

- Function : Write Register

```
Call    : int   BASWreg (port%, memtype%, addr%, count%, buf%(0),
                status%);
Inputs  : int   port;       /* 0–COM1, 1–COM2                  */
          int   memtype;    /* 1–OR, 2–R                       */
          int   addr;       /* Adrs. of the 1st register to write */
          int   count;      /* Number of words to write        */
        : int   *buf;       /* Data to write                   */
Return    int   status;     /* Error status–0: Normal          */
```

- Function : Read Bit

```
Call    : int   BASRbit (port%, memtype%, addr%, count%, buf$,
                status%);
Inputs  : int   port;       /* 0–COM1, 1–COM2                  */
          int   memtype;    /* 0–DI, 1–DO, 4–C                 */
          int   addr;       /* Adrs. of the 1st discrete to read */
          int   count;      /* Number of bits to read          */
Return  : char  *buf;       /* Data read                       */
          int   status;     /* Error status–0: Normal          */
```

- Function : Write Bit

```
Call    : int   BASWbit) (port%, memtype%, addr%, count%, buf$,
                status%);
Inputs  : int   port;       /* 0–COM1, 1–COM2                  */
          int   memtype;    /* 1–DO, 4–C                       */
          int   addr;       /* Adrs. of the 1st discrete to write */
          int   count;      /* Number of bits to write         */
          char  *buf;       /* Data to write                   */
Return    int   status;     /* Error status–0: Normal          */
```


Download from Www.Somanuals.com. All Manuals Search And Download.

Sample Programs

- **Sample C Program for Turbo C/C++ DEMO1.C**

  This program is provided "as-is". Do not call for technical support for this free program.

```
#include <dos.h>
#include <stdio.h>
#include "MCROCOMM.C"

int  port;
int  memtype;
int  addr;
int  count;
char  buf [16];
int  temp;
int  z,x,y,value1, value2 = 0;

main ( )
{
while (z == 0)
       {
          port = 2 ; /* COM3 */
          memtype = 2;
          addr = 1;
          count = 16;
          temp = TCL_Rreg(port,memtype,addr,count,buf);
          x = buf[0];
          y = buf[1];
          value1 = (x & 0x00ff) + ((y & 0x00ff) <<8) ;
          x = buf[2];
          y = buf[3];
          value2 = (x & 0x00ff) + ((y & 0x00ff) <<8) ;
          printf("Register 1, 2 values are %d %d \n", value1, value2);
           }
return (0);
}
```


Download from Www.Somanuals.com. All Manuals Search And Download.

- **Sample BASIC Program**

```
100    DIM BUF% (16)

200    PORT%     = 1    REM Port–COM2
210    MEMTYPE%  = 0    REM Analog Input
220    ADDR%     =49    REM Start Address–49
230    COUNT%    =16    REM Length–16 Registers
240    CALL BASRreg (PORT%, MEMTYPE%, ADDR%, COUNT%, BUF%(1), ERRSTAT%)
250    IF ERRSTAT% <>0 THEN GOTO 900

300    PORT%     = 0    REM Port–COM1
310    MEMTYPE%  = 0    REM Discrete Input
320    ADDR%     =33    REM Start Address–33
330    COUNT%    =16    REM Length–16 Bits
340    BUF$      =SPACE$ (16)
350    CALL BASRbit (PORT%, MEMTYPE%, ADDR%, COUNT%, BUF$, ERRSTAT%)
360    IF ERRSTAT% <>0 THEN GOTO 900
370    FOR I% = 1 TO 16
380           TEMP$=MID$ (BUF$, I%, 1)
390           IF TEMP$="1" THEN PRINT "ON" ELSE PRINT "OFF"
400    NEXT I%

900
```


Download from Www.Somanuals.com. All Manuals Search And Download.

Appendix C RTU Protocol

This appendix describes the Remote Terminal Unit (RTU) serial communications protocol, which can be used to provide communications between the Micro PLC or other remote device and a host computer.

- Message Types
- Transmission Sequence
- Message Fields
- Character Format
- Message Termination
- Timeout Usage
- Cyclic Redundancy Check (CRC)
- RTU Message Length
- RTU Table Addresses
- Message Descriptions
- Communication Errors

Refer to appendix E for a sample RTU master program.

Introduction

One of the software Setup parameters for the Micro PLC is the selection of either Micro PLC protocol or RTU protocol. RTU protocol must be selected to use the features described here.

RTU protocol is a query–response protocol used for communication between devices, such as the Micro PLC and a host computer. The host computer operates as the master device; the Micro PLC is always a slave.

The data transferred consists of 8–bit binary characters with a parity bit. No control characters are added to the data block; however, an error check (Cyclic Redundancy Check) included as the final field of each query and response to ensure accurate transmission of data.

Download from Www.Somanuals.com. All Manuals Search And Download.

Message Types

There are four message types: **query**, normal response, error response, and **broadcast**.

Query

The master sends a message addressed to a single slave, such as a Micro PLC.

Normal Response

After the slave performs the function requested by the query, it sends back a normal response for that function. This indicates that the request was successful.

Error Response

The slave receives the query, but for some reason it cannot perform the requested function. The slave sends back an error response which indicates the reason the request could not be processed. (No error message will be sent for certain types of errors. For more information see *Communication Errors*, page C-18).

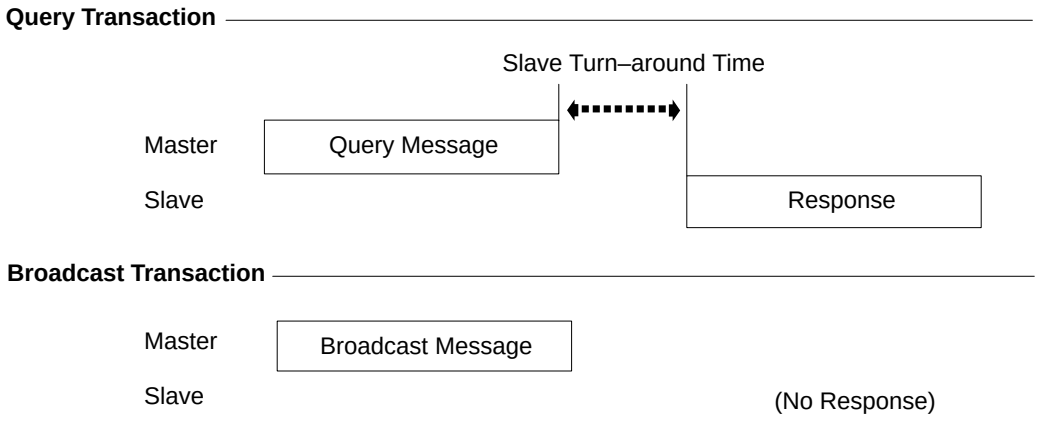
Broadcast

The master sends a message addressed to all of the slaves by using address 0. All slaves that receive the broadcast message perform the requested function. This transaction is ended by a time–out within the master.

Transmission Sequence

The master begins a transmission by sending a query or broadcast request message. If the master sent a query, the slave sends a normal or error response.

If master sends a broadcast request, the slaves do not send responses.

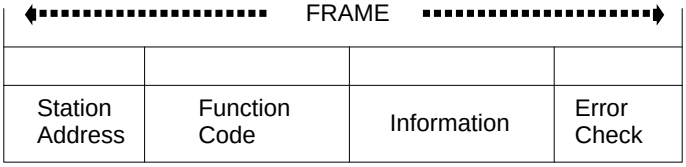


The time between the end of a query and the beginning of the response is called the slave turn–around time (see above). This varies, depending on the query and the activity of the Micro PLC application program. 500mS is a reasonable worst–case estimate.

Download from Www.Somanuals.com. All Manuals Search And Download.

Message Fields

A typical message has the fields shown below:



Station Address

The station address is the address of the slave selected for the data transfer. It is one byte in length and has a value from 0 to 247 inclusive. An address of 0 selects all slave stations, and indicates that this is a broadcast message. An address from 1 to 247 selects a slave station with that station address. The station address for a Micro PLC is one of its software Setup parameters.

Function Code

The function code identifies the command being sent. This field is one byte in length and may have a value from 0 to 255:

Function Code	Description	Function Code	Description
0	Illegal Function	8	Loopback Maintenance
1	Read Output Table	9–14	Unsupported Function
2	Read Input Table	15	Force multiple outputs
3	Read Registers	16	Reset multiple registers
4	Read Analog Input	17	Report device type
5	Force Single Output	18–127	Unsupported function
6	Preset Single Register	128–255	Reserved for Exception Responses
7	Read Exception Status		

Information Field

The information field contains all of the other information required to further specify or respond to a requested function. Detailed specification of the contents of the information field for each function code is found in the *Message Descriptions* that start on page C-9.

Error Check Field

The error check field is two bytes in length. It contains a cyclic redundancy check (CRC–16) code. Its content depends on the station address, function code, and information field. For information about generating the CRC– 16 code, see *Cyclic Redundancy Check (CRC)* beginning on page C-5. Note that the information field is variable in length. To properly generate the CRC– 16 code, the length of frame must be determined. See *RTU Message Length* (page C-8) to calculate the length of a frame for each of the defined function codes.

Download from Www.Somanuals.com. All Manuals Search And Download.

Character Format

A message is sent as a series of characters. Each byte in a message is transmitted as a character. The illustration below shows the character format. A character consists of a start bit (0), eight data bits, an optional parity bit, and one stop bit (1). Between characters the line is held in the 1 state.

		MSB			Data Bits				LSB		
10	9	8	7	6	5	4	3	2	1	0	
Stop	Parity (optional)										Start

Message Termination

Each station monitors the time between characters. When a period of three character times elapses without the reception of a character, the end of a message is assumed. The reception of the next character is assumed to be the beginning of a new message.

The end of a frame occurs when the first of the following two events occurs:

- The number of characters received for the frame is equal to the calculated length of the frame.
- A length of 3 character times elapses without the reception of a character.

Timeout Usage

Timeouts are used on the serial link for error detection, error recovery, and to prevent missing the end of messages and message sequences.

After sending a query message, the master should wait approximately 500 milliseconds before assuming that the slave did not respond to its request.

Download from Www.Somanuals.com. All Manuals Search And Download.

Cyclic Redundancy Check (CRC)

The Cyclic Redundancy Check (CRC) consists of 2 check characters generated at the transmitter and added at the end of the transmitted data characters. The receiver generates its own CRC for the incoming data and compares it to the CRC sent by the transmitter to ensure proper transmission.

The transmitter calculates the CRC. The essential steps are:

1. Multiply the data bits that make up the message by the number of bits in the CRC.
2. Divide the result by the generating polynomial (using modulo 2 with no carries). The CRC is the remainder of this division.
3. Discard the quotient.
4. Add the remainder (CRC) to the data bits.
5. Transmit the message with CRC.

The receiver divides the message plus CRC by the generating polynomial. If the remainder is 0, the transmission was transmitted without error.

The Generating Polynomial

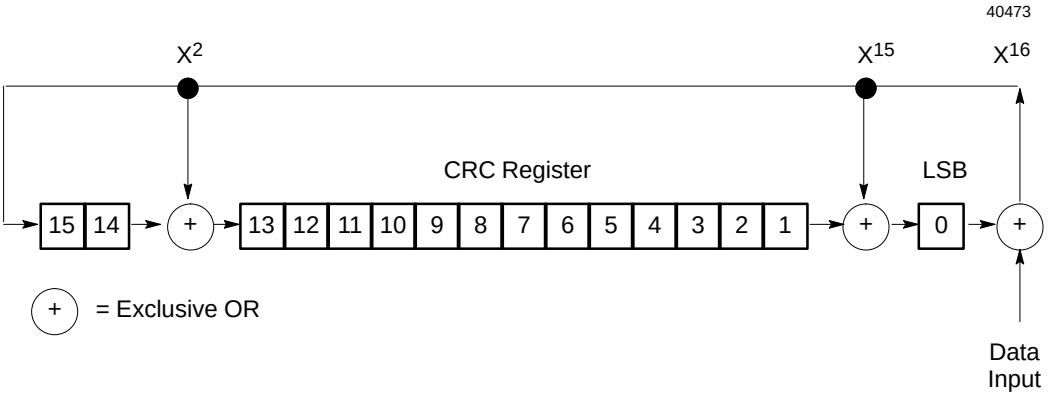
A generating polynomial is expressed as a string of terms in powers of X such as $X^3 + X^2 + X^0$ (or 1). It can also be expressed as a binary number. RTU protocol uses the polynomial $X^{16} + X^{15} + X^2 + 1$ which in binary is 1 1000 0000 0000 0101. The CRC this polynomial generates is known as CRC–16.

It can be implemented in hardware or software.

Download from Www.Somanuals.com. All Manuals Search And Download.

Hardware Implementation of CRC-16

CRC-16 can be implemented using a multi–section shift register (which is based on the generating polynomial):



The message data bits are fed to the Shift Register one at a time. The CRC register contains a preset value. As each data bit is presented to the Shift Register, the bits are shifted to the right. The LSB is XORed with the data bit and the result is: XORed with the old contents of bit 1 (the result placed in bit 0), XORed with the old contents of bit 14 (and the result placed in bit 13), and finally, it is shifted into bit 15. This process is repeated until all data bits in a message have been processed.

Software Calculation of CRC–16

The pseudo code for software calculation of the CRC–16 is given below.

```
                Preset byte count for data to be sent.
                Initialize the 16–bit remainder (CRC) register to all ones.
                XOR the first 8–bit data byte with the high order byte of the
                    16–bit CRC register. The result is the current CRC.
INIT SHIFT:     Initialize the shift counter to 0.
SHIFT:          Shift the current CRC register 1 bit to the right.
                Increment shift count.
                Is the bit shifted out to the right (flag) a 1 or a 0?
                    If it is a 1, XOR the generating polynomial with the current CRC.
                    If it is a 0, continue.
                Is shift counter equal to 8?
                    If NO, return to SHIFT.
                    If YES, increment byte count.
                Is byte count greater than the data length?
                    If NO, XOR the next 8–bit data byte with the current CRC
                        and go to INIT SHIFT.
                    If YES, add current CRC to end of data message
                        for transmission and exit.
```

When the message is transmitted, the receiver will perform the same CRC operation on all the data bits and the transmitted CRC. If the information is received correctly the resulting remainder (receiver CRC) will be 0.

Download from Www.Somanuals.com. All Manuals Search And Download.

Example CRC–16 Calculation

As an example we will calculate the CRC–16 for RTU message, Read Exception Status 07). The message format is:

Address	Function	CRC–16
01	07	

The Micro PLC transmits the rightmost byte (of registers or discrete data) first. The first bit of the CRC– 16 transmitted is the MSB. Therefore, in this example the MSB of the CRC polynomial is to the extreme right. The X^{16} term is dropped because it affects only the quotient (which is discarded) and not the remainder (the CRC characters). The generating polynomial is therefore 1010 0000 0000 0001. The remainder is initialized to all 1s.

In this example we are querying device number 1 (address 01). We need to know the amount of data to be transmitted. This information can be found for every message type on page C-8, *RTU Message Length*. For this message the data length is 2 bytes.

```
                TRANSMITTER
            CRC–16 ALGORITHM
MSB[2]          LSB[2] Flag MSB[2]
Initial Remainder    1111 1111 1111 1111
XOR 1st data byte    0000 0000 0000 0001
Current CRC          1111 1111 1111 1110
Shift 1               0111 1111 1111 1111  0
Shift 2               0011 1111 1111 1111  1
XOR Gen. Polynomial  1010 0000 0000 0001
Current CRC          1001 1111 1111 1110
Shift 3               0100 1111 1111 1111  0
Shift 4               0010 0111 1111 1111  1
XOR Gen. Polynomial  1010 0000 0000 0001
Current CRC          1000 0111 1111 1110
Shift 5               0100 0011 1111 1111  0
Shift 6               0010 0001 1111 1111  1
XOR Gen. Polynomial  1010 0000 0000 0001
Current CRC          1000 0001 1111 1110
Shift 7               0100 0000 1111 1111  0
Shift 8               0010 0000 0111 1111  1
XOR Gen. Polynomial  1010 0000 0000 0001
Current CRC          1000 0000 0111 1110
XOR 2nd data byte    0000 0000 0000 0111
Current CRC          1000 0000 0111 1001
Shift 1               0100 0000 0011 1100  1
XOR Gen. Polynomial  1010 0000 0000 0001
Current CRC          1110 0000 0011 1101
Shift 2               0111 0000 0001 1110  1
XOR Gen. Polynomial  1010 0000 0000 0001
Current CRC          1101 0000 0001 1111
Shift 3               0110 1000 0000 1111  1
XOR Gen. Polynomial  1010 0000 0000 0001
Current CRC          1100 1000 0000 1110
Shift 4               0110 0100 0000 0111  0
Shift 5               0011 0010 0000 0011  1
XOR Gen. Polynomial  1010 0000 0000 0001
Current CRC          1001 0010 0000 0010
Shift 6               0100 1001 0000 0001  0
Shift 7               0010 0100 1000 0000  1
XOR Gen. Polynomial  1010 0000 0000 0001
Current CRC          1000 0100 1000 0001
Shift 8               0100 0010 0100 0000  1
XOR Gen. Polynomial  1010 0000 0000 0001
Transmitted CRC      1110 0010 0100 0001
                       E    2    4    1
```

```
                RECEIVER[1]
            CRC–16 ALGORITHM
                LSB[2] Flag
Rcvr CRC after data   1110 0010 0100 0001
XOR 1st byte Trns CRC 0000 0000 0100 0001
Current CRC            1110 0010 0000 0000
Shift 1                 0111 0001 0000 0000  0
Shift 2                 0011 1000 1000 0000  0
Shift 3                0001 1100 0100 0000  0
Shift 4                 0000 1110 0010 0000  0
Shift 5                 0000 0111 0001 0000  0
Shift 6                 0000 0011 1000 1000  0
Shift 7                0000 0001 1100 0100  0
Shift 8                 0000 0000 1110 0010  0
XOR 2nd byte trns CRC 0000 0000 1110 0010
Current CRC            0000 0000 0000 0000
Shift 1–8 yields       0000 0000 0000 0000
                       ALL ZEROES FOR RECEIVER
                       FINAL CRC–16 INDICATES
                       TRANSMISSION CORRECT!
```

[1] Remember, the receiver processes incoming data the same way as the transmitter. The example for the receiver begins when all data bits but not the transmitted CRC have been received correctly. Therefore, the receiver CRC should be equal to the transmitted CRC at this point. When this occurs, the output of the CRC algorithm will be zero indicating that the transmission is correct.

The transmitted message with CRC would then be:

Address	Function	CRC–16	
01	07	41	E2

[2] The MSB and LSB references are to the data bytes only, *not* the CRC bytes. The CRC MSB and LSB order are the reverse of the data byte order.

Download from Www.Somanuals.com. All Manuals Search And Download.

RTU Message Length

To generate the CRC–16 for any message, the message length must be known. The length for all types of messages can be determined from the table below.

Function Code And Name		Query or Broadcast Message Length Less CRC Code	Response Message Length Less CRC Code
0		Not Defined	Not Defined
1	Read Output Table	6	3 + 3rd byte [1]
2	Read Input Table	6	3 + 3rd byte [1]
3	ReadRegisters	6	3 + 3rd byte [1]
4	Read Analog Input	6	3 + 3rd byte [1]
5	Force Single Output	6	6
6	Preset Single Register	6	6
7	Read Exception Status	2	3
8–14	Loopback / Maintenance	Not Defined	Not Defined
16	Preset Multiple Registers	7 + 7th byte [1]	6
17	Report Device Type	2	8
18–127		Not Defined	Not Defined
128–255		Not Defined	3

[1] The value of this byte is the number of bytes contained in the data being transmitted.

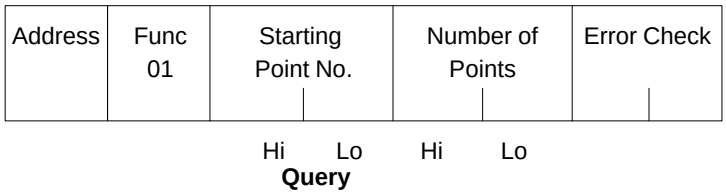
Download from Www.Somanuals.com. All Manuals Search And Download.

Message Descriptions

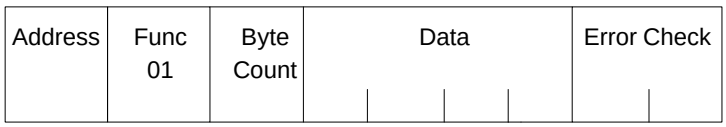
The following pages explain the format and fields for each RTU message.

Message (01): Read Output Table

Format:



Query



Normal Response

Query:

- An address of 0 is not allowed as this cannot be a broadcast request.
- The function code is 01.
- The starting point number is two bytes in length. It may be any value less than the highest output point number available in the Micro PLC. The starting point number is equal to one less than the number of the first output point returned in the normal response to this request.
- The number of points value is two bytes in length. It specifies the number of output points returned in the normal response. The sum of the starting point value and the number of points value must be less than or equal to the highest output point number available in the Micro PLC. The high order byte of the starting point number and number of bytes fields is sent as the first byte. The low order byte is the second byte in each of these fields.

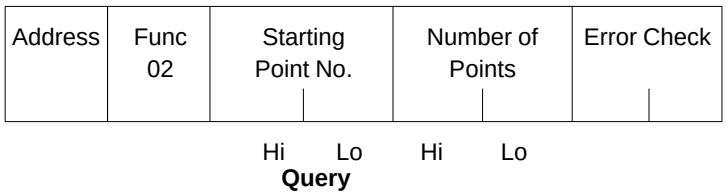
Response:

- The byte count is a binary number from 1 to 256 (0 = 256). It is the number of bytes in the normal response following the byte count and preceding the error check.
- The data field of the normal response is packed output status data. Each byte contains 8 output point values. The least significant bit (LSB) of the first byte contains the value of the output point whose number is equal to the starting point number plus one. The values of the output points are ordered by number starting with the LSB of the first byte of the data field and ending with the most significant bit (MSB) of the last byte of the data field. If the number of points is not a multiple of 8, then the last data byte contains zeros in one to seven of its highest order bits.

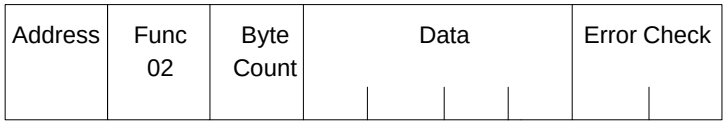
Download from Www.Somanuals.com. All Manuals Search And Download.

Message (02): Read Input Table

Format:



Query



Normal Response

Query:

- An address of 0 is not allowed as this cannot be a broadcast request.
- The function code is 02.
- The starting point number is two bytes in length. It may be any value less than the highest input point number available in the Micro PLC. The starting point number is equal to one less than the number of the first input point returned in the normal response to this request.
- The number of points value is two bytes in length. It specifies the number of input points returned in the normal response. The sum of the starting point value and the number of points value must be less than or equal to the highest input point number available in the Micro PLC. The high order byte of the starting point number and number of bytes fields is sent as the first byte. The low order byte is the second byte in each of these fields.

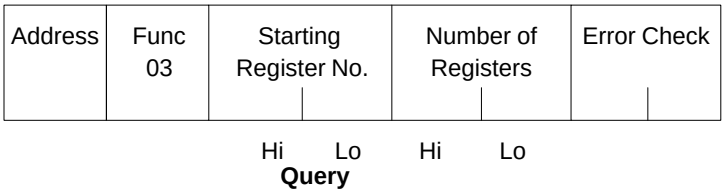
Response:

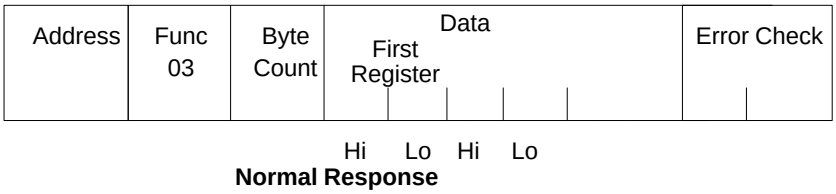
- The byte count is a binary number from 1 to 256 (0 = 256). It is the number of bytes in the normal response following the byte count and preceding the error check.
- The data field of the normal response is packed input status data. Each byte contains 8 input point values. The least significant bit (LSB) of the first byte contains the value of the input point whose number is equal to the starting point number plus one. The values of the input points are ordered by number starting with the LSB of the first byte of the data field and ending with the most significant bit (MSB) of the last byte of the data field. If the number of points is not a multiple of 8, then the last data byte contains zeros in one to seven of its highest order bits.

Download from Www.Somanuals.com. All Manuals Search And Download.

Message (03): Read Registers

Format:





Query:

- An address of 0 is not allowed as this request cannot be a broadcast request.
- The function code is equal to 03.
- The starting register number is two bytes in length. The starting register number may be any value less than the highest register number available in the Micro PLC. It is equal to one less than the number of the first register returned in the normal response to this request.
- The number of registers value is two bytes in length. It must contain a value from 1 to 125 inclusive. The sum of the starting register value and the number of registers value must be less than or equal to the highest register number available in the Micro PLC. The high order byte of the starting register number and number of registers fields is sent as the first byte in each of these fields. The low order byte is the second byte in each of these fields.

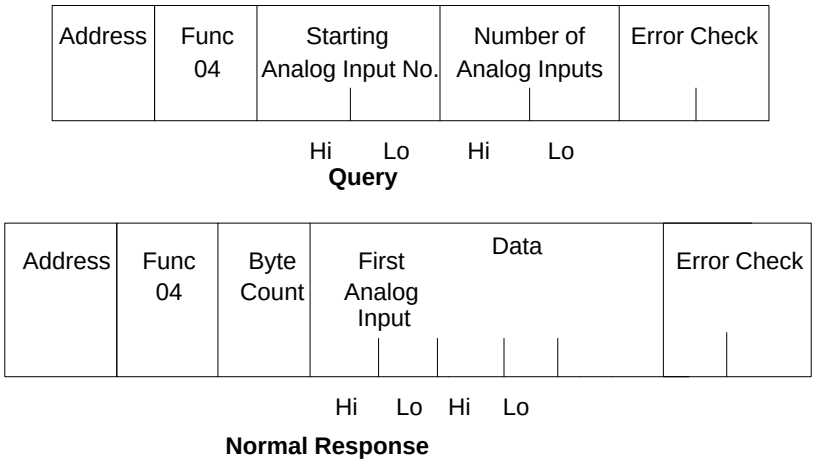
Response:

- The byte count is a binary number from 2 to 250 inclusive. It is the number of bytes in the normal response following the byte count and preceding the error check. Note that the byte count is equal to two times the number of registers returned in the response. A maximum of 250 bytes (125) registers is set so that the entire response can fit into one 256 byte data block.
- The registers are returned in the data field in order of number with the lowest number register in the first two bytes and the highest number register in the last two bytes of the data field. The number of the first register in the data field is equal to the starting register number plus one. The high order byte is sent before the low order byte of each register.

Download from Www.Somanuals.com. All Manuals Search And Download.

Message (04): Read Analog Inputs

Format:



Query:

- An address of 0 is not allowed as this request cannot be a broadcast request.
- The function code is equal to 4.
- The starting analog input number is two bytes in length. The starting analog input number may be any value less than the highest analog input number available in the Micro PLC. It is equal to one less than the number of the first analog input returned in the normal response to this request.
- The number of analog inputs value is two bytes in length. It must contain a value from 1 to 125 inclusive. The sum of the starting analog input value and the number of analog inputs value must be less than or equal to the highest analog input number available in the Micro PLC. The high order byte of the starting analog input number and number of analog input fields is sent as the first byte in each of these fields. The low order byte is the second byte in each of these fields.

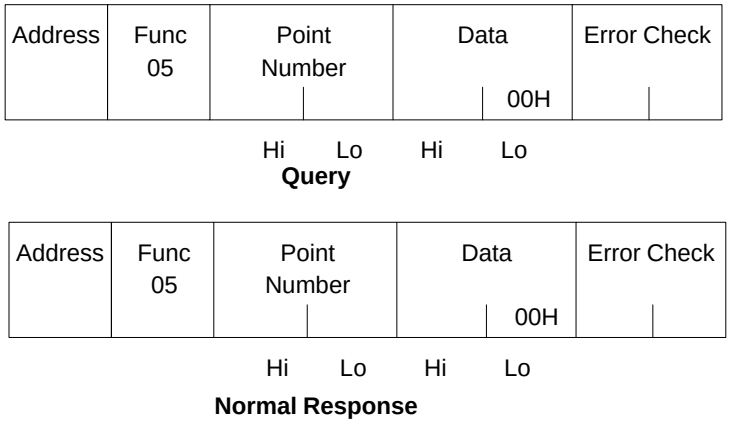
Response:

- The byte count is a binary number from 2 to 250 inclusive. It is the number of bytes in the normal response following the byte count and preceeding the error check. Note that the byte count is equal to two times the number of analog inputs returned in the response. A maximum of 250 bytes (125) analog inputs is set so that the entire response can fit into one 256 byte data block.
- The analog inputs are returned in the data field in order of number with the lowest number analog input in the first two bytes and the highest number analog input in the last two bytes of the data field. The number of the first analog input in the data field is equal to the starting analog input number plus one. The high order byte is sent before the low order byte of each analog input.

Download from Www.Somanuals.com. All Manuals Search And Download.

Message (05): Force Single Output

Format:



Query:

- An address of 0 indicates a broadcast request. All slave stations process a broadcast request and no response is sent.
- The function code is equal to 5.
- The point number field is two bytes in length. It may be any value less than the highest output point number available in the Micro PLC. It is equal to one less than the number of the output point to be forced on or off.
- The first byte of the data field is equal to either 0 or 255 (FFH). The output point specified in the point number field is to be forced off if the first data field byte is equal to 0. It is to be forced on if the first data field byte is equal to 255 (FFH). The second byte of the data field is always equal to zero.

Response:

- The normal response to a force single output query is identical to the query.

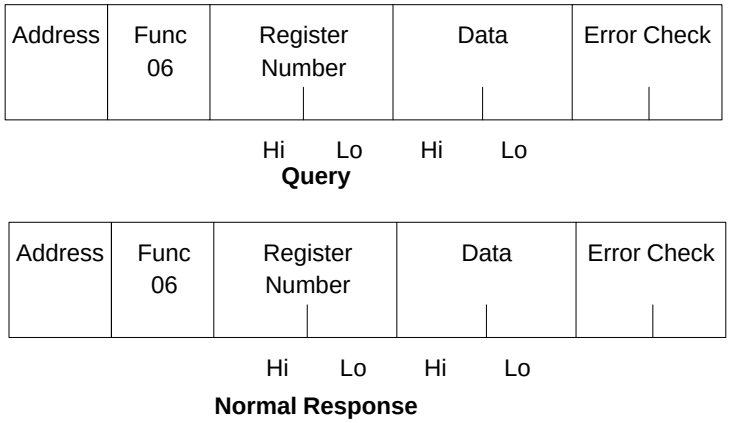
Note

The force single output request is not an output override command. The output specified in this request is ensured to be forced to the value specified only at the beginning of one sweep of the Micro PLC user logic.

Download from Www.Somanuals.com. All Manuals Search And Download.

Message (06): Preset Single Register

Format:



Query:

- An address 0 indicates a broadcast request. All slave stations process a broadcast request and no response is sent.
- The function code is equal to 06.
- The register number field is two bytes in length. It may be any value less than the highest register available in the Micro PLC. It is equal to one less than the number of the register to be preset.
- The data field is two bytes in length and contains the value that the register specified by the register number field is to be preset to. The first byte in the data field contains the high order byte of the preset value. The second byte in the data field contains the low order byte.

Response:

- The normal response to a preset single register query is identical to the query.

Download from Www.Somanuals.com. All Manuals Search And Download.

Message (07): Read Exception Status

Format:

Address	Func 07	Error Check

Query

Address	Func 07	Data	Error Check

Normal Response

Query:

This query is a short form of request for the purpose of reading the first eight output points.

- An address of zero is not allowed as this cannot be a broadcast request.
- The function code is equal to 07.

Response:

- The data field of the normal response is one byte in length and contains the states of output points 01 through 08. The output states are packed in order of number with output point one's state in the least significant bit and output point eight's state in the most significant bit.

Download from Www.Somanuals.com. All Manuals Search And Download.

Message (16): Preset Multiple Registers

Format:

Address	Func 16	Starting Register No.	Number of Registers	Byte Count	Data	Error Check

Query

Address	Func 16	Starting Register No.	Number of Registers	Error Check

Normal Response

Query:

- An address of 0 indicates a broadcast request. All slave stations process a broadcast request and no response is sent.
- The value of the function code is 16.
- The starting register number is two bytes in length. The starting register number may be any value less than the highest register number available in the Micro PLC. It is equal to one less than the number of the first register preset by this request.
- The number of registers value is two bytes in length. It must contain a value from 1 to 125 inclusive. The sum of the starting register number and the number of registers value must be less than or equal to the highest register number available in the Micro PLC. The high order byte of the starting register number and number of registers fields is sent as the first byte in each of these fields. The low order byte is the second byte in each of these fields.
- The byte count field is one byte in length. It is a binary number from 2 to 250 inclusive. It is equal to the number of bytes in the data field of the preset multiple registers request. Note that the byte count is equal to twice the value of the number of registers.
- The registers are returned in the data field in order of number with the lowest number register in the first two bytes and the highest number register in the last two bytes of the data field. The number of the first register in the data field is equal to the starting register number plus one. The high order byte is sent before the low order byte of each register.

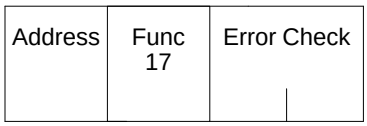
Response:

- The description of the fields in the response are covered in the query description.

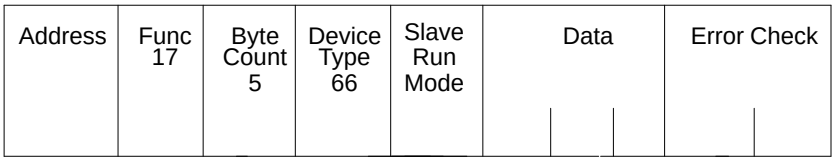
Download from Www.Somanuals.com. All Manuals Search And Download.

Message (17): Report Device Type

Format:



Query



Normal Response

Query:

The Report Device Type query is sent by the master to a slave in order to learn what type of programmable control or other computer it is.

- An address of zero is not allowed as this cannot be a broadcast request.
- The function code is equal to 17.

Response:

- The byte count field is one byte in length and is equal to 5.
- The device type field is one byte in length and is equal to 66.
- The slave run mode field is one byte in length. The slave run light byte is equal to OFFH if the Micro PLC is running. It is equal to 0 if the Micro PLC is not running.
- The data field contains three bytes.

  1st Byte = 30H if the Micro PLC is a 14-point or 16-point unit
  32H if the Micro PLC is a 28-point unit

  2nd Byte = 30H if the Micro PLC Expander is a 14-point unit
  32H if the Micro PLC Expander is a 28-point unit
  0H if the Micro PLC Expander is not present
  33H if the Micro PLC Expander is an analog unit

  3rd Byte = 0

Download from Www.Somanuals.com. All Manuals Search And Download.

Communication Errors

Communication errors are divided into three groups:

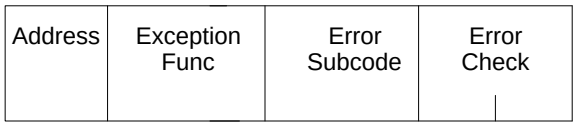
- Invalid Query Message
- Serial Link Time Outs
- Invalid Transaction

Invalid Query Message

When the Micro PLC receives a query addressed to itself, but cannot process the query, it sends one of the following error responses:

	Subcode
Invalid Function Code	1
Invalid Address Field	2
Invalid Data Field	3
Query Processing Failure	4

The format for an error response to a query is as follows:



The address reflects the address provided on the original request. The exception function code is equal to the sum of the function code of the query plus 128. The error subcode is equal to 1, 2, 3, or 4. The value of the subcode indicates the reason the query could not be processed.

Invalid Function Code Error Response (1)

An error response with a subcode of 1 is called an invalid function code error response. This response is sent by the Micro PLC if it receives a query whose function code is not equal to 1 through 8, 15, 16, or 17.

Invalid Address Error Response (2)

An error response with a subcode of 2 is called an invalid address error response. This error response is sent in the following cases:

1. The starting point number and number of points fields specify output points or input points that are not available in the Micro PLC (returned for function codes 1, 2, 15).
2. The starting register number and number of registers fields specify registers that are not available in the Micro PLC (returned for function codes 3, 4, 16).
3. The starting analog input number and analog input number fields specify analog inputs that are not available in the Micro PLC (returned for function code 4).
4. The point number field specifies an output point not available in the Micro PLC (returned for function code 5).

Download from Www.Somanuals.com. All Manuals Search And Download.

5. The register number field specifies a register not available in theMicro PLC (returned for function code 6).
6. The analog input number field specifies an analog input number not available in the Micro PLC (returned for function code 3).

Invalid Data Value Error Response (3)

An error response with a subcode of 3 is called an invalid data value error response. This response is sent in the following cases:

The first byte of the data field is not equal to 0 or 255 (FFh) or the second byte of the data field is not equal to 0 for the Force Single Output Request (Function Code 5).

This response is also sent when the data length specified by the memory address field is longer than the data received.

Query Processing Failure Error Response (4)

An error response with a subcode of 4 is called a query processing failure response. This error response is sent by the Micro PLC if it properly receives a query but communication between it and the computer fail.

Serial Link Timeout

The only cause for a RTU device to time out is if an interruption to a data stream of 3 character times occurs while a message is being received. If this occurs the message is considered to have terminated and no response will be sent to the master. There are certain timing considerations due to the characteristics of the slave that should be taken into account by the master.

After sending a query message, the master should wait approximately 500 milliseconds before assuming that the slave did not respond to its request.

Invalid Transactions

If an error occurs during transmission that does not fall into the category of an invalid query message or a serial link time–out, it is known as an invalid transaction. Types of errors causing an invalid transaction include:

- Bad CRC.
- The data length specified by the memory address field is longer than the data received.
- Framing or overrun errors.
- Parity errors.

If an error in this category occurs when a message is received by the Micro PLC, theMirco PLC does not return an error message. The Micro PLC treats the incoming message as though it was not intended for it.

Download from Www.Somanuals.com. All Manuals Search And Download.

Appendix D Communications Using Windows DDE

This appendix explains how to use the demonstration program MICROWIN.EXE that is included with the Micro PLC software. This demonstration program is provided to acquaint you with an available software product that can be used to connect DDE-compliant Microsoft™ Windows programs with data in a Micro PLC.

To run the demonstration program, you need:

- Microsoft Windows version 3.1 or later.
- (optional) Microsoft Word™ (version 2.0 or 6.0) if you want to read the links that are supported in the demonstation software.
- (optional) Microsoft Excel™ (version 4.0 or 5.0) if you want to demonstrate reading / writing Micro PLC data using an Excel spreadsheet (sample provided).

The Micro PLC must be set up to use Micro PLC protocol (not RTU protocol).

Limits of the Demonstration Software

A. The demonstration program will time out after 15 minutes.

B. The demonstration program has only four preconfigured topic / items; in the product itself, the number of links is only limited by the user's available memory.

C. The demonstration software does not support the modem features that are available in the product software.

D. The demonstration software does not support the configuration file features that are available in the product software.

Download from Www.Somanuals.com. All Manuals Search And Download.

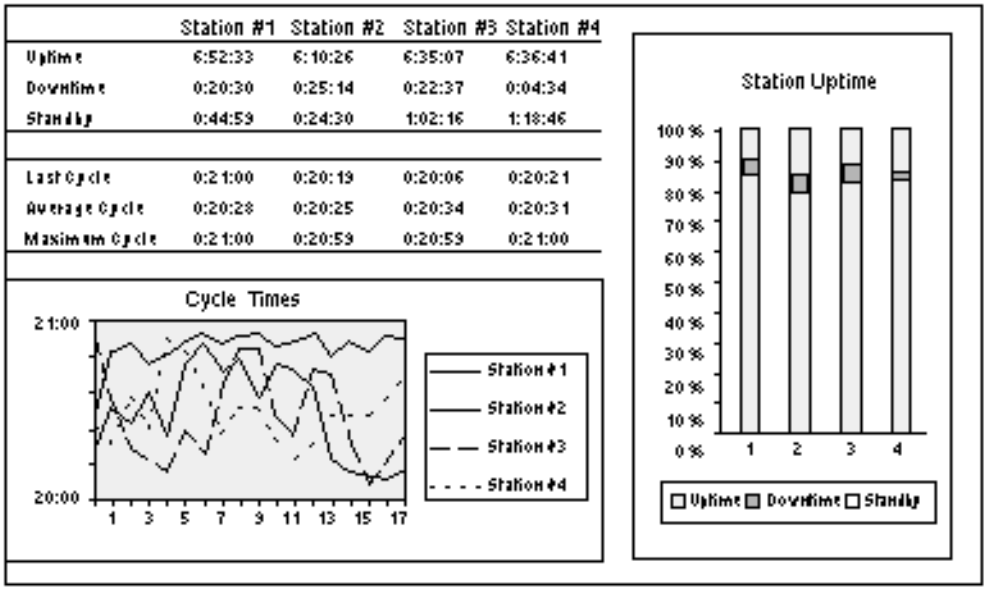
Features of the Micro PLC DDE Driver Software

The software product, called the Micro PLC DDE Driver, allows linking real-time data from the plant floor into applications for display, logging, or trending. It also allows setting individual parameters or downloading recipes to a programmable controller from a supervisory computer. Features include:

- Support for direct point-to-point port connections.
- Access to all Micro PLC data memory types.
- Support for multiple data formats.
- Independent polling rates for each data point.
- Optimizes reads.
- Easy configuration.
- Support for multiple configurations.
- Ability to change driver configuration and actions via DDE links.
- Remote access through modems with automatic dialing.
- Ability to monitor DDE conversion status.
- Comprehensive error messages.
- On-line help.

Windows DDE compliant applications such as Microsoft Excel allow data from the Micro PLC to be displayed in tabular or graphical form (see below).

46189



In addition to spreadsheet applications, data can be brought into word processing, database, and other applications.

Download from Www.Somanuals.com. All Manuals Search And Download.

Simple Demonstration using Microsoft Word

The Micro PLC software includes a Microsoft Word document named WORD.DOC. It can be used to read data links provided by the demonstration software.

To try it, follow these steps:

1. First, use the Micro PLC programming software to download the file DDE1.LAD from the \MICRO\DDE subdirectory to the Micro PLC. Then start the Micro PLC.
2. Return to DOS and call up Windows.
3. From the Program Managers menu bar, select "File/Run".
4. Enter the path\filename of the MICROWIN.EXE program (i.e. C:\MICRO\DDE\MICROWIN.EXE) and click OK.
5. Select "Settings/Serial Port Settings" from the MICROWIN menu bar.
6. Change the COM Port to reflect the serial port where the Micro PLC is connected.
7. Select "Actions/Toggle StopRun Driver" from the MICROWIN menu bar. This will start the driver communication with the PLC.
8. After the driver is started and communicating, start Microsoft Word.
9. Open the Word document (C:\MICRO\DDE\WORD.DOC) and click "YES" to re–establish the links with the MICROWIN driver.

The values shown in the document will be "real time" data values being read from the Micro PLC.

Demonstration using Microsoft Excel

An Excel sheet (EXCEL.XLS) and an Excel macro (EXCEL.XLM) are provided with the demonstration software. They can be used to demonstrate how the software can read and write Micro PLC data.

1. Start the DDE driver.
2. Begin communication with the Micro PLC as described above.
3. After the driver is started and communicating, start Microsoft Excel.
4. Open the excel sheet (EXCEL.XLS) and click "YES" to re–establish the links with the MICROWIN driver.

The values shown in the spreadsheet will be "real time" data values being read from the Micro PLC.

To change the values shown in the spreadsheet open the Excel macro (EXCEL.XLM) and click the "Run Macro" button. This macro will poke new data values to register 1, register 2, and output 1.

Download from Www.Somanuals.com. All Manuals Search And Download.

Viewing PLC Data in Windows

Follow these steps:

1. First, use the Micro PLC programming software to download the file DDE1.LAD from the \MICRO\DDE subdirectory to the Micro PLC.
2. Return to DOS and call up Windows.
3. From the Program Managers menu bar, select "File/Run".
4. Enter the path\filename of the MICROWIN.EXE program (i.e. C:\MICRO\DDE\MICROWIN.EXE) and click OK.
5. Select "Settings/Serial Port Settings" from the MICROWIN menu bar.
6. Change the COM Port to reflect the serial port where the Micro PLC is connected.
7. Select "Actions/Toggle Stop Run Driver" from the MICROWIN menu bar. This will start the driver communication with the PLC.
8. Select "Options/Driver Messages" from the MICROWIN menu bar.
9. Select one of the links in the "Topic/Item List". The Data associated with this item will be displayed in the selected item editbox. The demonstration program lists these four links:

   CPUID/R,1 – #convs=0,#Blk=3,A=0,L=2

   CPUID/R,2 – #convs=0,#Blk=3,A=0,L=2

   CPUID/I,1 – #convs=0,#Blk=2,A=0,L=1

   CPUID/Q,1 – #convs=0,#Blk=1,A=0,L=1

   Each line above displays the link information in the form:

   [TOPIC] / [ITEM] – #conv=[c],#Blk=[b],A=[a],L=[l] where,

   [TOPIC} = "CPUID" and is the same for all four links.

   [ITEM] = the item of each one is different and this is what determines what data value in the PLC is being addressed. The item of the first one is "R,1" which represents Register 1. The second one is "R,2" which represents Register 2. The third one is "I,1" which represents the state of Input 1. The fourth is "O,1" which represents the state of Output 1.

   [c] = refer to the driver help file.

   [b] = refer to the driver help file.

   [a] = refer to the driver help file.

   [l] = refer to the driver help file.

Download from Www.Somanuals.com. All Manuals Search And Download.

Viewing PLC Data in another DDE-compliant Application

You can view the PLC data in another DDE-compliant application by copying the data link to the Window's clipboard and then pasting the link into the application.

To copy the link to the clipboard, from the "Options / Driver Messages" screen select the desired link in the "Topic / Item List" by clicking on it and then clicking the Copy button. To paste the link in another application, switch to the other application and paste the link (this is typically done with the "Paste Special" or "Paste Link" option from the "Edit" selection on the menu bar, but the exact procedure may vary depending on the application).

Writing Values to the PLC from another Application

To write data to a PLC address, the client DDE application must POKE the desired value to the server link that represents the address. Note: in this demonstration software, the only topic that is supported is "CPUID" and the only items supported are the following: "R,1", "R,2", "I,1", "O,1".

The following Microsoft Excel macro (in cells A1 through A4) writes the value in cell B1 to register 1.

cell A1 – INITIATE("MICROWIN","CPUID")

cell A2 – POKE(A1,"R,1",B1) cell A3 – TERMINATE(A1)

cell A4 – RETURN()

cell B1 – Data for Register 1

This specific macro would execute the following steps:

1. The "INITIATE" function opens a DDE channel to Server "MICROWIN" and Topic "CPUID".
2. The "POKE" function writes the data value in spreadsheet cell B1 to register 1 ("R,1").
3. The "TERMINATE" function closes the DDE channel that was opened in cell A1.
4. The "RETURN" function ends the macro.

Ordering Information

For information about ordering this product, and for technical support, contact:

Engineering Specialists Inc.
12805 W. Burleigh Rd.
Brookfield, WI 53005–3119
414–782–3050

Download from Www.Somanuals.com. All Manuals Search And Download.

Appendix E Data Acquisition, Logging, and Display Program

This appendix describes the RTU-based Data Acquisition, Logging, and Display Program software, which is provided on the Micro PLC software diskettes. Note that not all of the capabilities of the Display software described here are available when it is used with multiple Micro PLCs on a Micro PLC Net (see below).

Features

The Data Acquisition, Logging, and Display Program software provides the following capabilities for your Micro PLC system:

- Data acquisition and display from Micro PLCs.
- Easy to make screen displays (no programming) with embedded / updating data.
- Multiple screens can be created, and called up with a single keystroke for display.
- Logging of "out of range" data to disk.
- Writing of banks of registers (recipes) to the PLC from diskette. The register bank data is created with a text editor, and may be documented as part of the file.
- Manual interrogation of remote devices allowing debug of the communications network, and / or RTU driver devices.
- Display of pre–programmed messages (stored in the computer) based on register contents in a remote device.

Note

This free software is provided "as– is". GE Fanuc makes no warranty of any kind with respect to this software.

GE Fanuc does not expect to provide enhancements or modifications to this program.

You are free to use this program in any way you wish, except that you should not use this program in a critical application, and you are not authorized to distribute copies.

Do not call requesting support for this program.

Using the Display Software with Micro PLC Net

Creating a network with multiple Micro PLCs requires the Micro PLC Net product described in the *Micro PLC User's Guide*. The MICRONET software establishes a point-to-point communications link with only one device at a time. Therefore, a system using the Display software on a Micro PLC Net can only communicate with one device at a time. To establish communications with different devices, it is necessary to return to DOS and re-invoke the MICRONET software to make each new connection.

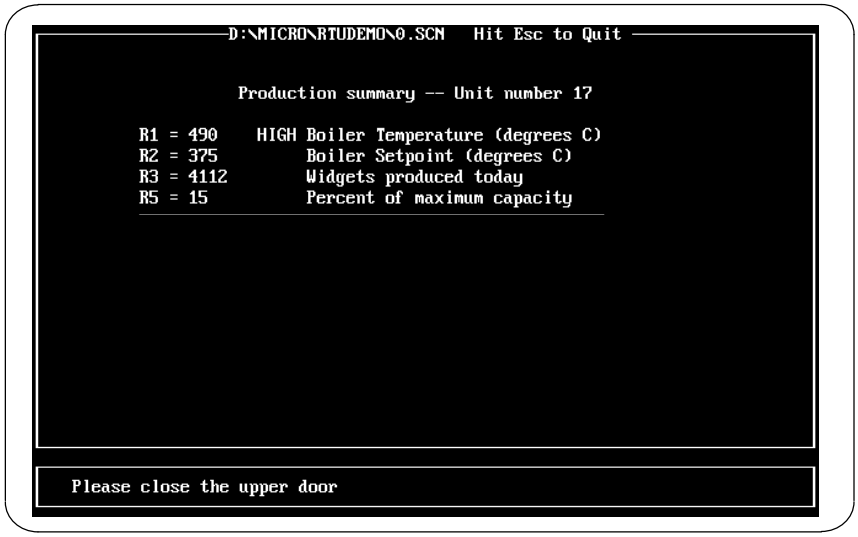
Download from Www.Somanuals.com. All Manuals Search And Download.

Overview

DISPLAY is an easy to use program that runs on an IBM PC, XT, or AT-compatible computer. It allows monitoring of Micro PLCs using the RTU protocol. You don't need to know how to use RTU protocol to use the Display software. However, you do need to set up the Micro PLC to operate using RTU protocol. Setup instructions are in chapter 7 of this manual. If you want to learn about RTU protocol, see appendix C.

The main features of the Display software are:

- Auto-polling screens that display actual data from Micro PLCs and other remote RTU devices. The example Auto-polling screen shown below displays the contents of four registers in a Micro PLC.



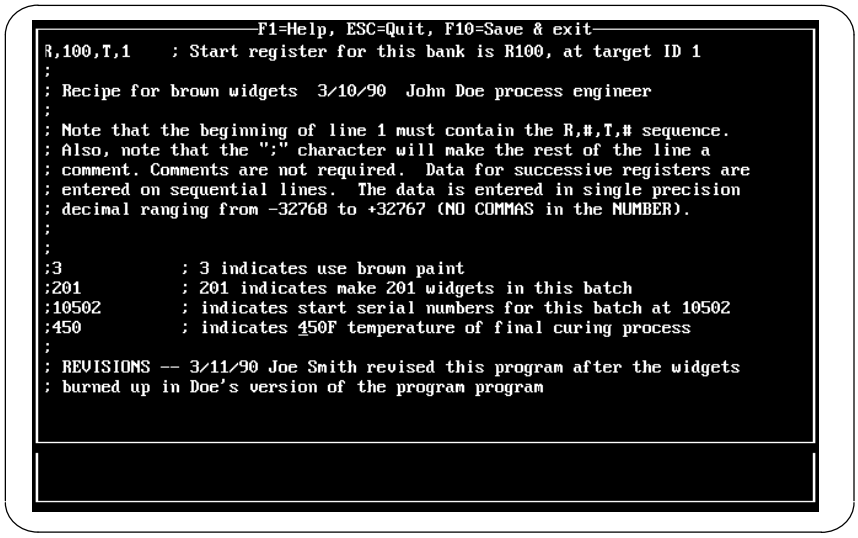
  Multiple screens can be created ahead of time (using a built-in text editor), then displayed with a single keystroke while the system is running. The data on the screen will be continuously updated. No programming is required, except for typing in the display screens as you wish them to appear.

- Normal operating limits can be set for any data items. "Out of range" data can be visually flagged on the screen and optionally logged to disk for later evaluation. In the example above, the boiler temperature has exceeded the limit in the HIGH direction. The value and corresponding HIGH (or LOW) indication blink.
- Display of system messages in an independent area on the bottom of all Auto-polling screens. In the above example, the message "Please close the upper door" appears. Messages with negative numbers blink. System messages are created ahead of time and saved in a file like this:

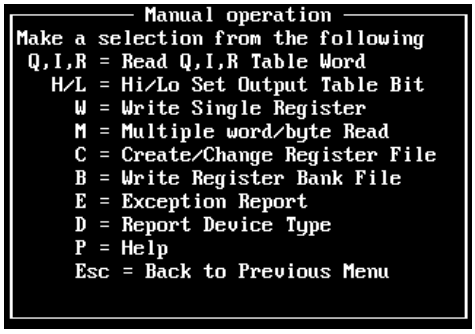
  Display of system messages is controlled by a Micro PLC on the network.

Download from Www.Somanuals.com. All Manuals Search And Download.

- Ability to download register data such as recipes to a Micro PLC. Comments can be used in the file, as in the following example, to describe the data being downloaded. In this example, data is sent to registers R100, R101, R102, and R103. To the application program in the Micro PLC, these registers represent paint color, number of widgets, starting serial number, and oven temperature.



- Finally, a Manual mode is available which allows you to poll a Micro PLC, and observe both the transmitted data (including header and checksum information) and the received data. This is valuable when you are trying to determine that the Micro PLC is actually sending the appropriate register or I/O data.



In Manual mode, you can also send data to the Micro PLC's register or I/O tables as individual words, and to the register table as a bank of up to 30 (typical) registers.

Download from Www.Somanuals.com. All Manuals Search And Download.

Equipment Required

Computer:	IBM PC / XT or AT or equivalent.
Micro PLC CPU:	MDR014C, MDR114C, MDR028B, MDR128B, MAA014B, MDD016A.
Micro PLC Cable	
Programming Software (optional):	Version 2.42
Micro PLC NET (optional):	
Master interface module:	HE485MST232
Slave interface module:	HE485SLV232
Software:	part number not available at time of publication; contact Horner Electric for information.

Download from Www.Somanuals.com. All Manuals Search And Download.

Startup

The DOS files COMMAND.COM and MODE.COM (or MODE.EXE) need to be in the search path. If you are unsure how to set up your path, you may copy these files to the same directory where Display is located.

To try a quick experiment, connect your PC to Micro PLC cable to COM1 of your computer, and to a compatible Micro PLC (MDR014C, MDR114C, MDR028B, MDR128B, MAA014B, MDD016A). Use version 2.42 of the programming software to set the Micro PLC communications mode to "RTU". Then exit the programming software package. Then go to the subdirectory \Micro\RTUDEMO, and type TRY–IT.

Use ESC to return to DOS.

Invoking DISPLAY

After booting with your diskette or going to the DISPLAY subdirectory, you can invoke the Display software in two different ways:

A. If you want to start the Display software and immediately begin executing an existing Auto-polling screen, type: **`DISPLAY filename.SCN baudrate`**.

   Use the filename of the Auto-polling screen you want to execute and enter the baud rate. The baud rate can be 300, 1200, 2400, 4800, or 9600. 9600 baud requires a very fast computer. For a slower computer, such as a 286, 2400 baud is probably the maximum. (The Micro PLC setup must also specify the same baud rate).

   This startup method is useful to automatically restart Display after a power cycle.

   Your AUTOEXEC.BAT file could include the line DISPLAY FILENAME.SCN BAUDRATE (e.g. Display Demo.scn 4800).

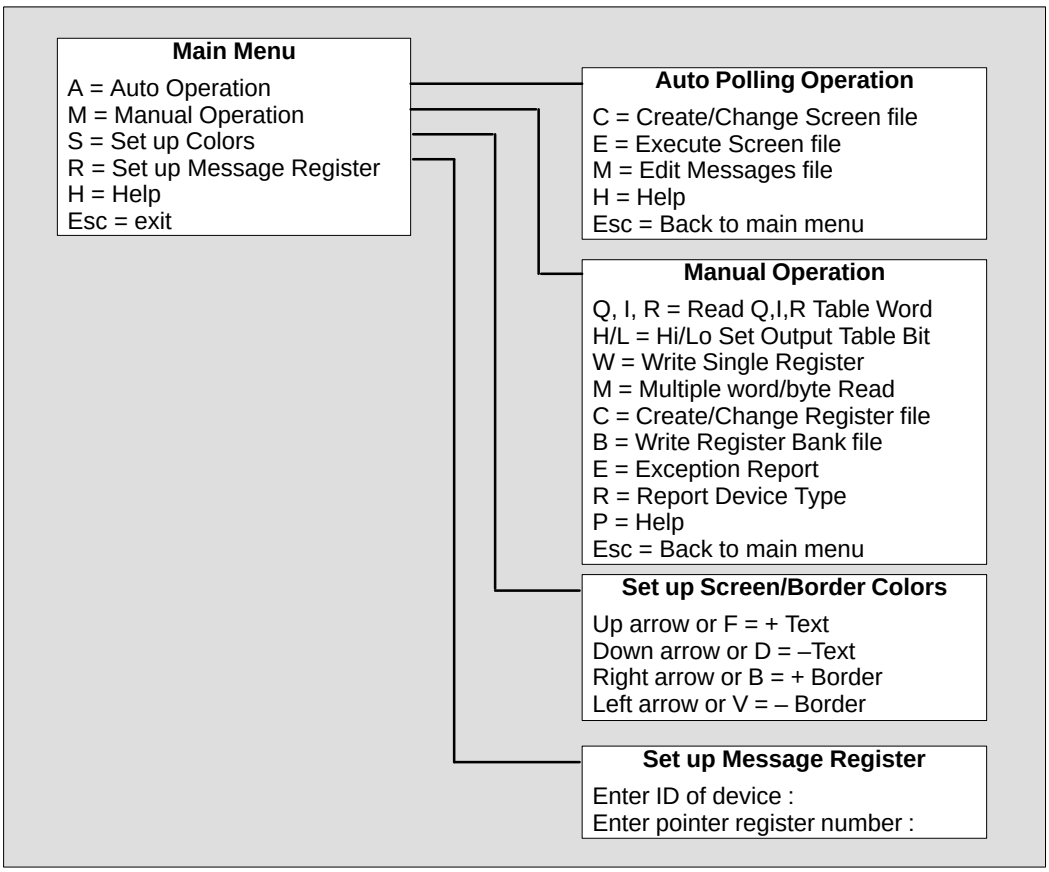
B. If you want to go to the Display software main menu, type: **`DISPLAY baudrate`**

   For example, DISPLAY 4800.

   A startup screen appears. From there, press any key to go to the Main menu:

Download from Www.Somanuals.com. All Manuals Search And Download.

The Display Software Menus



To Create or Execute Auto-polling Screens: Select A from the Main menu. For creating Auto-polling screens, see the instructions on page E-13. For executing Auto-polling screens, see the instructions on page E-21.

To Manually Poll a Micro PLC: Select M from the Main menu. See page E-9 for instructions.

To Change Data in a Micro PLC: Select M from the Main menu. See page E-9 for instructions.

To Download Registers to a Micro PLC: Select M from the Main menu. See page E-11.

To Change the Software Screen Colors: Select S from the Main menu. See page E-7 for instructions.

To Create or Edit System Messages: Select A from the Main menu. See page E-19 for instructions.

To Set Up a Micro PLC (or Other Device) to Control System Messages: Select R from the Main menu. See page E-20 for instructions.

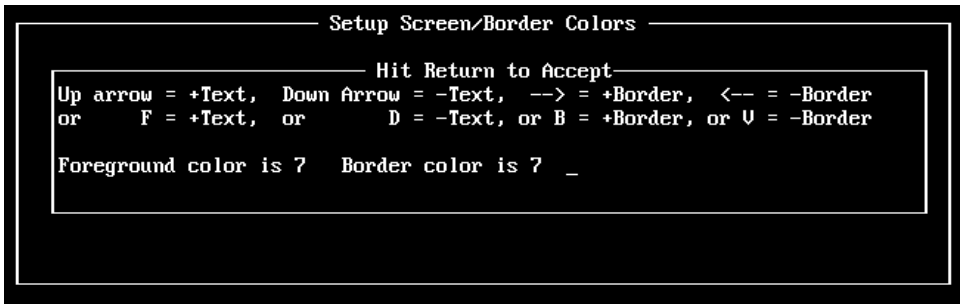
To Exit to DOS: You can exit to DOS from this menu by pressing the ESC key. You can also exit to DOS at any time by pressing Control–Break.

Download from Www.Somanuals.com. All Manuals Search And Download.

Changing the Screen Colors

If you want to change the screen colors for using the Display software, select S from the Main menu. (Colors for the Auto-polling screens are set up separately; see page E-18).

When you select S, this setup window appears:



Use the keys indicated to change the foreground (text and background) and border colors. The current colors are indicated by numbers in the bottom line of text.

The 7,7 defaults work best for monochrome displays. If you change the defaults, you may not be able to see the cursor when you are looking at a directory screen in either the auto or manual mode.

Download from Www.Somanuals.com. All Manuals Search And Download.

Editing Summary

Basic editing functions for the screen, message, and register files are listed below.

Aborting a Function: Use the ESC key to stop most functions. Pressing ESC with no function started exits the editor. ALT F7 will undo most of the block level commands, if necessary.

Cursor Movement: Use the page up and page down keys to move the text up or down one page at a time. Control page up, and control page down move to the beginning or end of the file. Home positions the cursor in column 1, and end puts the cursor at the end of the line.

String Search: To search for a string (case sensitive), press control F3, then enter the string. Press F3 again. The cursor will move to the first occurrence of that string. To find the next occurrence, push F3 again.

Block Copy: To copy a block of text, position the cursor at the beginning of the block and press Control F5. Then move the cursor to the end of the block, and press Control F5 again. Finally, move the cursor to the desired position for the copy to be placed, and press Control F5 again. To make another copy elsewhere, move the cursor and press Shift F5.

Block Move: Use the ALT F6 keys to move a block of text (similar to the block copy, above).

Copy From Disk File: To copy a file from disk into your working file, position the cursor at the desired copy location, and use the F7 key. Enter the filename that contains the text to be copied. Position the cursor at the beginning of the text to be copied and press F10. Then move the cursor to the end of the text to be copied and press F10 again.

Save Part Of The File To Disk: To save part of a file to disk, place the cursor at the start of the text to be saved, then press ALT F5. Move the cursor to the end of the text and press ALT F5 again. Enter the filename for the save. This file can be later read using the F7 key.

Misc: The Ins / 0 key toggles the insert / overwrite mode. To delete a line, use control Y.

Another set of editing commands can also be used. See the Help screen for details

Download from Www.Somanuals.com. All Manuals Search And Download.

Manual Mode

This section explains how to use Manual mode to manually poll the selected Micro PLC:

- Reading one word of input, output, or register data
- Setting an output low or high
- Writing a single register
- Creating or changing a register file
- Writing data to a register file
- Creating a report of exceptions
- Reporting the device type

Manual mode is useful to determine the mapping of the Micro PLC, the integrity of the interconnects, and other factors before creating an auto polling screen.

Data Format

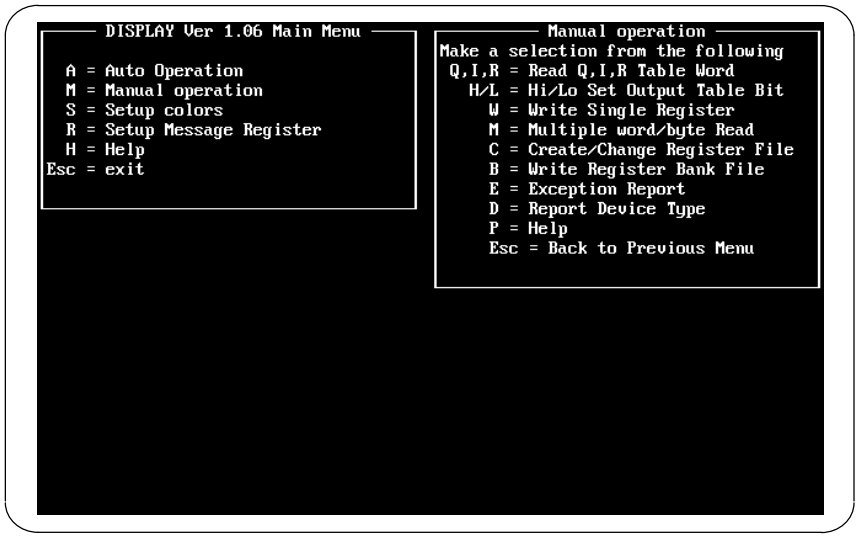
In Manual mode, both transmitted data and received data are displayed in hex format. In addition, both transmitted and received data displayed show header and checksum information. Received data also explicitly shows the "data" from the response. All returned data is 16 bits.

If the response has a bad checksum, wrong target ID, or wrong function number, this is displayed before the data is shown. If no response is forthcoming (for example, if there is an improperly wired cable), then "Bad Checksum" is shown along with no actual received data.

Download from Www.Somanuals.com. All Manuals Search And Download.

Entering Manual Mode

To go to Manual mode, press the M key at the main menu. The menu of Manual mode selections appears to the right of the Main menu.

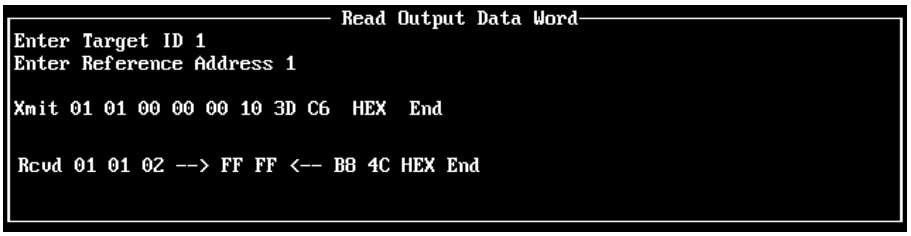


You can press ESC to return to the main menu, or press P for help.

Reading One Word of Input, Output, or Register Data

The R,Q, and I options each read a word of data from the register, output, or input table.

In the example below, Q was selected. Then target ID = 1 and address 1 were designated as the data to fetch.



The data returned was FF FF (hex) which indicates that outputs Q1– Q16 were all high (1).

Reading Multiple Bytes of Input, Output, or Register Data

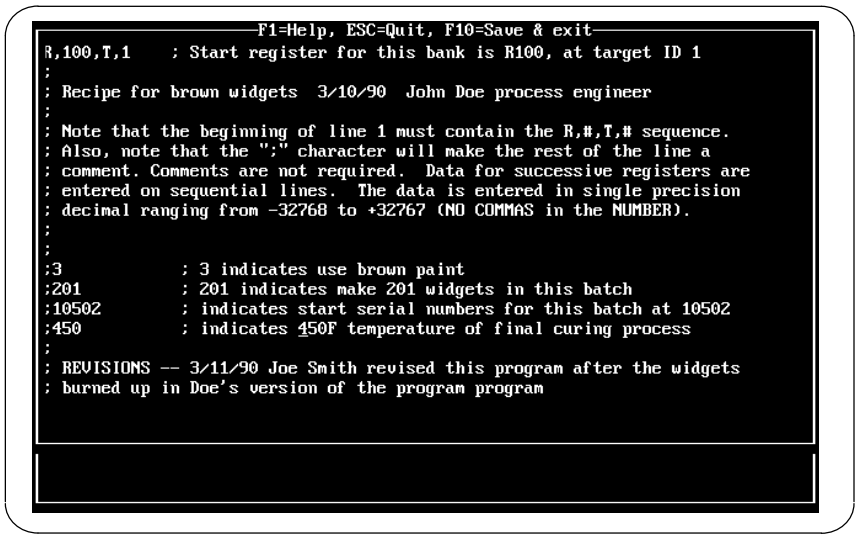
The M option allows reading of multiple bytes from any of the tables. The data which is being transmitted is shown as hex bytes from left to right in the order they are transmitted. Generally the first byte is the target ID, and the second byte is the function number. The last 2 bytes are the checksum. The bytes in between are the start addresses, lengths, and data if applicable. The received data is shown in the same format. The data bytes are bracketed by —> DATA BYTES <— left and right arrows.

Download from Www.Somanuals.com. All Manuals Search And Download.

Creating a Bank of Registers for Downloading Data

In Manual mode, you can create files of register data to be downloard to a Micro PLC. With this option, it is possible to store recipes on diskette which can then be downloaded to the PLC when appropriate.

A sample of this register bank file is shown below:



Select C. The screen displays:

Enter a name for the file that will contain the information to be downloaded to the target registers. The filename must end with the suffix .REG. Press the Enter key.

A blank screen appears to enter the information.

Start the first line by entering the starting register and the target ID of the Micro PLC (or other device) that will receive the information. Separate the items using commas. The required format is:

R,#,T,#

For example:

```
R,15,T,1   ; Begin at register 15, Micro PLC Target ID 1
```

Register (15) Target ID (1) Comment (starts with ;)

You can also include a comment that starts with a semicolon character, as shown above.

Go to the next line of the file and start entering the values (decimal numbers) to be downloaded to consecutive registers in the Micro PLC. The data values may be in the range –32768 to +32767.

Download from Www.Somanuals.com. All Manuals Search And Download.

Data values can be entered on separate lines, or on the same line but separated by spaces. For example, here are two ways to format the same data:

R,15,T,1 ; This specifier must be on line 1, and be as shown.
156 ; This is register 15, the setpoint temperature for the donut ovens.
–945 ; R16 contains the number of donuts we are ahead of schedule with
4467 ; This will be R17 data.

OR

R,15,T,1
156 –945 4467 ; R15–R17 contains the donut specific information

In the first example, each value is on a separate line, and each line includes a comment.

In the second example, the second line of the file contains all there values to be downloaded, separated by spaces.

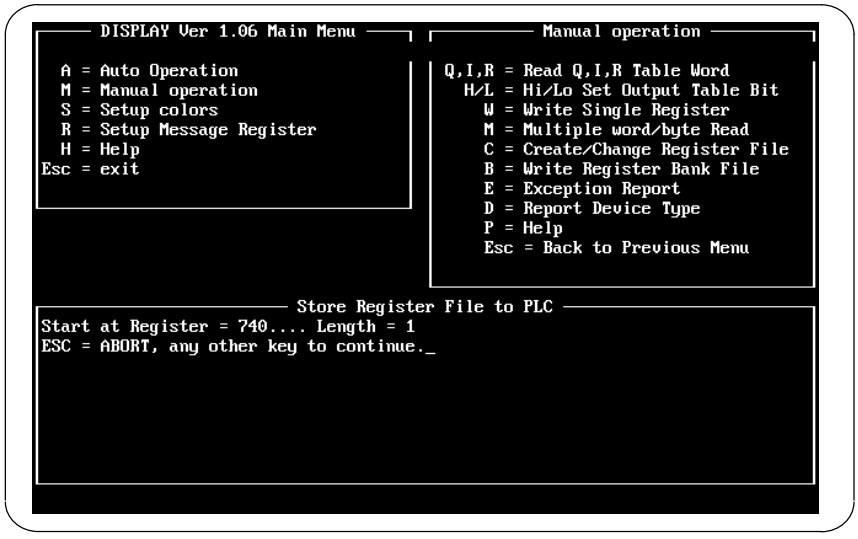
When entering numerical data do not use commas. For example, for the number ten–thousand, you must enter 10000, not 10,000.

Anything on a line followed by the ";" character is interpreted as a comment. This allows you to document the contents of your recipe files. The first item on the line must be the register contents.

After completing the file, press F10 to save the file and return to the menu.

Writing Data to the Registers

To download a completed register file to the selected Micro PLC, select B from the Manual Operation menu. The screen will show a list of files ending in the suffix: .REG. Select a file to download by highlighting the filename and pressing the Enter key. The file will appear for you to review. If you want to download the file, press the F10 key. A prompt like the example shown below appears.



The Start register, and length appear as a double check that the data was entered into the register bank file. Press any key to download the file.

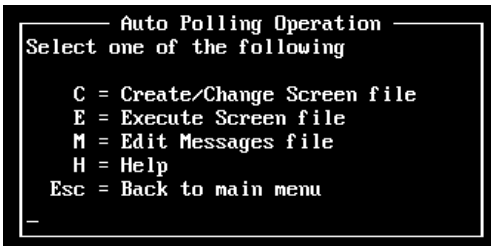
Download from Www.Somanuals.com. All Manuals Search And Download.

Creating or Editing Autopolling Screens

This section explains how to create or edit display screens that can be used for:

- Displaying selected input, output, and register data from multiple Micro PLCs or other remote RTU devices.
- Writing a Value to a Register
- Forcing an Output
- Obtaining Register Data from a Disk File

To create a screen, or change an existing one, from the main menu type A (Auto Polling). You will see the menu shown below.



To create or edit a screen file, press C.

Naming an Auto-polling Screen File

At the prompt, enter the file name you wish the screen to have. During operation, it will be possible to display (execute) any screen through the menus provided. However, it will also be possible to change from one screen to another with a single keystroke if you give the screens the following names: 0.SCN, 1.SCN, etc. up to 9.SCN. These screens can be loaded and executed with a single keystroke ("0" thru "9" keys).

You can create the screens with these names, or rename them later using the DOS rename command (COPY may also be used – for example, COPY DEMO.SCN 1.SCN will replace the). present 1.SCN with the DEMO.SCN screen).Press the Enter key.

You should always name your screens with a .SCN extension. This is required to execute screens from within the DISPLAY environment or from the DOS command line.

After entering the filename, press the Enter key to continue.

Specifying an Existing File to Edit

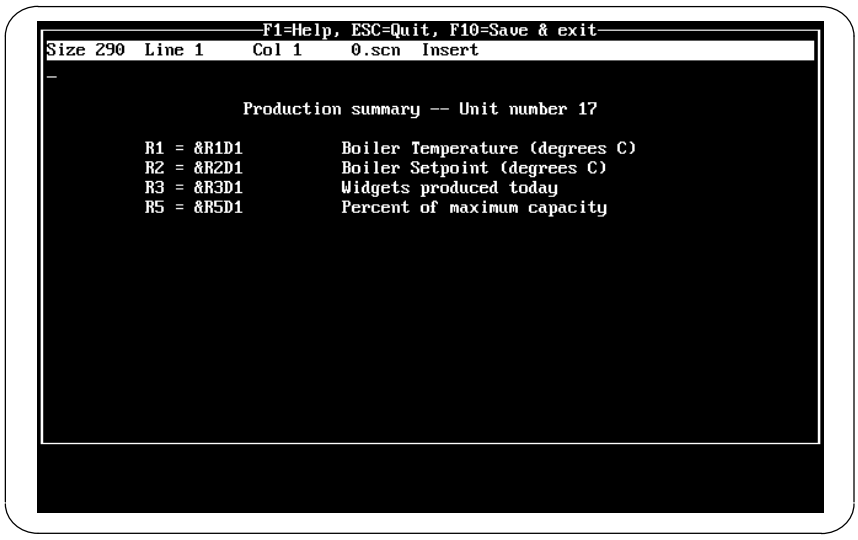
If you want to change an existing Auto-polling file, follow the steps above. When prompted for a filename, enter the name of the file you want to change. Then press the Enter key.

Download from Www.Somanuals.com. All Manuals Search And Download.

Creating an Auto-polling Screen

For a new screen, the following display appears:

During operation, an Auto-pollng screen can display up to 21 lines of text and data items. A very simple screen is shown below.



To create a screen, enter the following on the first 21 lines of the file:

- Text.
- A Data Display string for each data item that will appear on the Auto-polling screen.

These optional items are usually placed below the 21st line, so they will not be displayed:

- A Data Limits string for each item that will be assigned high / low limits.
- An optional Color Setup Data and Logging Interval string.

The following pages explain how to enter formatting strings. Instructions for using the text editor are on page E-8.

Note: Don't use tabs while in the editor or your remote data will end up at the wrong place on the screen. Use spaces instead of the TAB key.

Download from Www.Somanuals.com. All Manuals Search And Download.

Formatting Strings for Auto-polling Screens

There are 3 types of formatting strings that set up the display and operating parameters for the auto-polling display screens. The 3 setup strings are:

1. Data display formatting string (e.g. &R32H1). See page E-16.
2. Optional data limits formatting string (e.g. &L,R,32,–100,100). See page E-17.
3. Optional color setup / logging formatting string (e.g. &#,10,10,0). See page E-18.

In general, all the required information must be included in the string as shown, including commas. None of the items in the string are optional, they must be included.

If any item is missing in a string, the entire string is treated as normal text, and has no effect on operation. In the third item above, for example, if your string was &#,10,10 (missing the data logging time interval) then the entire string would be ignored, and the defaults of 7,7,0 would be used.

Note that while commas are required in the limits string and the colors / logging string, commas must not be used in the data display string.

Text for Auto-polling Screens

In the file for an Auto-polling screen, anything that is not specifically formatted as one of the strings listed above is treated as regular text. For example, the following line has both a formatting string and texxt:

```
&L,R,5,350,450    This sets the normal limits of R5 to 350 to 450.
```

Setup String — Display Text

Setup for Data Logging During Auto-Polling

It is sometimes desirable to save the displayed Auto-polling data to disk, especially if the data is out of range, indicating some type of error condition.

Saving data to disk requires the following setup:

1. A non–zero logging time must be entered in a Color / Setup Formatting string (see page E-18).
2. A data limit must be set for at least one of the values on the screen (see page E-17.). When one or more of these limits are exceeded, in either the positive or negative direction, logging will begin.

Data logging can use a lot of disk space, so be sure you have enough space available if you plan to log data.

Download from Www.Somanuals.com. All Manuals Search And Download.

Data Display String Format – &XYYYYYZN

Use this setup string format to:

- Specify one input, output, and / or register to include on the Auto-polling screen.
- Specify the target ID of the Micro PLC that is the source of the data.
- Specify how the data for that item should be displayed.

Enter a Data Display String for each data item to be displayed on the screen.

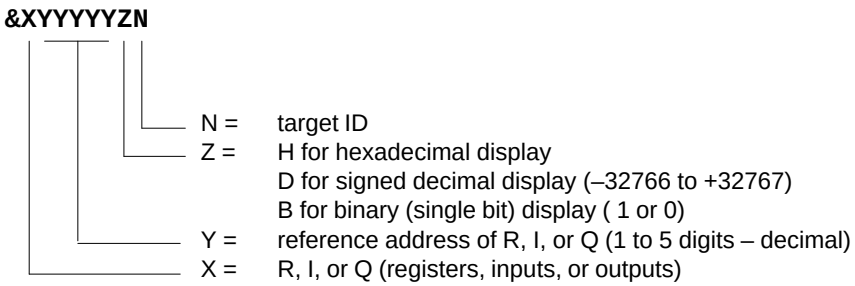
This string can be located anywhere in the first 21 lines of the screen. It is normally mixed with text to explain what the data is. Do not use commas in this string

The & character must start on the screen to the left of column 70, or there will not be room to display the resulting data.

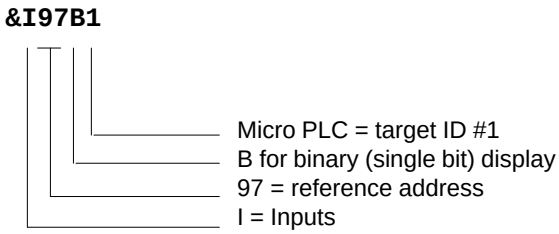
Lower case R,I,Q,H,D,B are not allowed, caps must be used.

If you specify Input (I) or Output (Q) data, and hex (H) or decimal (D) format, the actual Auto-polling screen will display 16 consecutive bits starting with the indicated reference.

Format:



Example:



Download from Www.Somanuals.com. All Manuals Search And Download.

Data Limits Format – &L,X,YYYYY,Low_Limit,Hi_limit

Use this setup string format to specify a high and low limit for a register, input, or output.

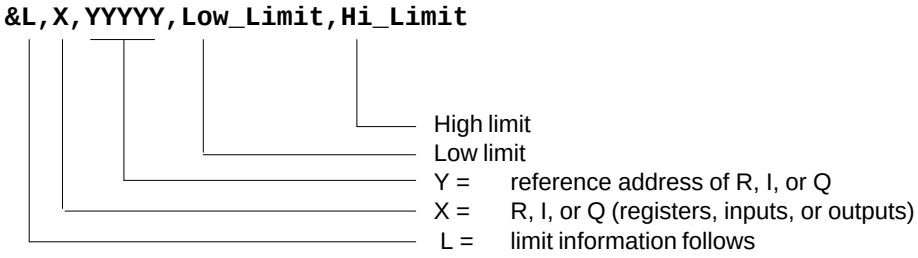
If the specified item exceeds either limit, the word HIGH or LOW appears on the Auto-polling display. In addition, the item blinks.

Limits are also used to control data logging to disk. Data logging occurs only if a data item is beyond one of its limits, AND if the data logging interval (see the next page) is set to some non-zero value. If you want to log everything on a screen regardless of status, then set the limit(s) on some data item so that it will always be out of range. For example, if you wanted to trigger on output bit Q27, you could set the limits of Q27 as &L,Q,27,0,0. Logging would occur whenever Q27 was set to 1. When Q27 was equal to 0, logging would stop. You could then manually control the logging with a hardwired switch in the PLC, or with the ^H and ^L commands available in the auto–screen and manual modes.

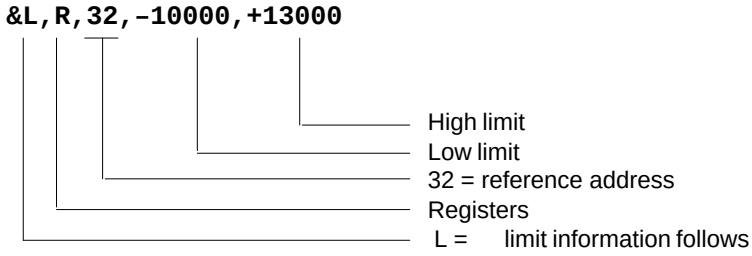
This formatting string is normally located below the first 21 lines of the screen, to de–clutter the screen and allow for more space for other items. It can be located in the first 21 lines if desired, however.

Enter a Data Limits String for each data item to be assigned limits.

Format:



Example:



Use comma to separate the individual items. Do NOT use commas within a limit, however (e.g. don't enter "–10,000").

&L	indicates the data to follow is limit information
X	R,Q, or I (capitals only)
YYYYY	Reference to which the limit will be applied – 1 to 5 decimal digits.
Low_limit	Any number within the available display range. Enter this number in signed decimal, regardless of the format in which the actual data will be displayed (see the previous page).
Hi_limit	Any number within the available display range. Enter this number in signed decimal, regardless of the format in which the actual data will be displayed.

Download from Www.Somanuals.com. All Manuals Search And Download.

Colors/logtime setup format – &#,SC,BC,LT

Use this setup format to:

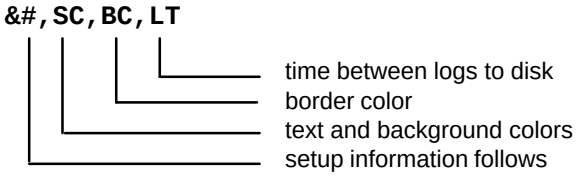
- Change screen colors for the Auto-polling screen you are working on (other Auto-polling screens can use different colors).

  The screen and border colors default to 7,7 which is recommended for monochrome monitors. However, you can specify any colors your computer can generate. If you want to try out a color combination before using it here, go back to the Main menu and select Setup Colors (S).

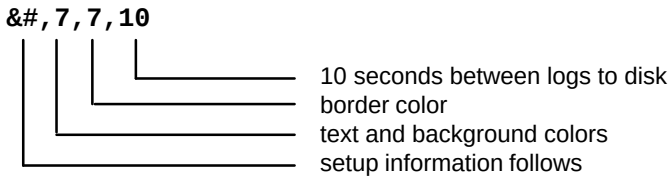
- Specify a time interval for logging screen data to disk.

This formatting string is normally located below the first 21 lines of the screen, to de–clutter the screen and allow for more space for other items. It can be located in the first 21 lines if desired, however.

Format:



Example:



&# indicates that setup information follows

SC Text / background color combinations. 7 is recommended for monochrome.

BC Border color. 7 is recommended for monochrome.

LT Time between data logs to disk. 0 turns the data logging off.

To turn logging off, make the log time parameter in the setup string equal to 0.

To turn logging on, enter a logging time interval of up to 32000 seconds (8.89 hours). If any data on the screen is out of range, the data will be saved to disk at this time interval (10 seconds in this example). This time does not affect the normal screen update time.

The logging time cannot be faster than the screen update time. If you are running at 300 baud, it may take 10–20 seconds to update a screen. It is not possible to log the disk data any faster than this, no matter what value is put in the formatting string.

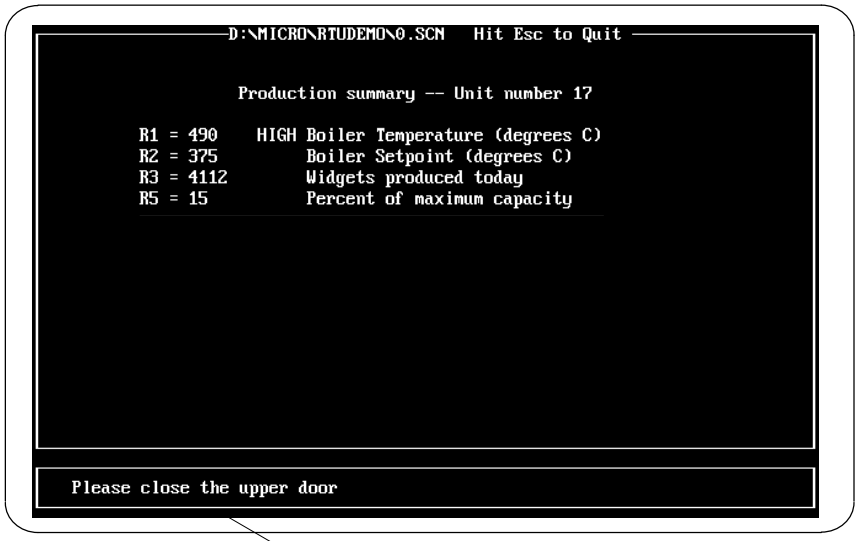
Download from Www.Somanuals.com. All Manuals Search And Download.

System Messages

This section explains how to:

- Create or edit system messages that can be displayed on the bottom of the Auto-polling screens.
- Specify a Micro PLC to control system messages
- Trigger system message displays during Auto-polling operation

System messages are activated by one (and one only) Micro PLC. However, the device that controls the system messages must be the one currently selected using the MICRONET software. With that device selected, the messages appear no matter what Auto-polling screen is executing.

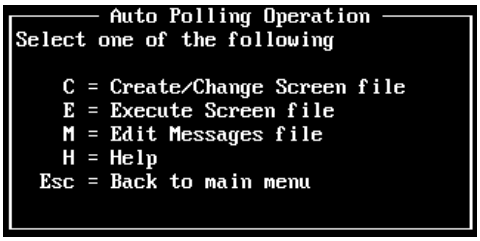


System Message Window on the Auto-polling Screen

Important messages can easily be set up to blink on the display.

Creating or Editing System Messages

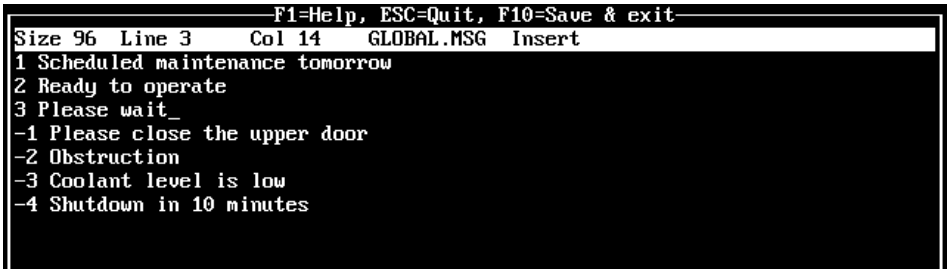
To create or edit system messages, enter "A" from the Main menu to display the Auto-Polling screen:



1. Enter "M" from the Auto-polling screen. The system messages file appears. At first, this file has no contents.

Download from Www.Somanuals.com. All Manuals Search And Download.

An example completed message file could be:



2. Enter the system messages you want to appear during system operation. Each message must fit on one line.
   - Start each message with a number in the range of –32768 to 32767.
   - If you want a message to blink on the display, give it a negative number.
   - Do not place a comma in the message number. The software interprets any numbers after a comma as part of the actual message.
   - Do not leave empty lines; messages beyond that point will not be used.

Specifying the Device to Control System Message Display

During system operation, display of the system messages is controlled by the contents of a single register at a Micro PLC.

1. Go to the main menu and press the R key for “Setup Message Register”.

2. Enter the ID of the Micro PLC that will control the execution of the system messages. Press the Enter key.
3. Enter the number of the register (in that PLC) that will contain the message pointer.

Triggering Display of a System Message During Auto-polling Operation

To trigger a message, the Micro PLC specified as the Message Register Source must place the number of that message in its Message Register.

For example, if you specified Target ID = 1 and Message Register 2, the Micro PLC with target ID = 1 would be the message controller, and the contents of register 2 would contain the number of the message.

If the register contains a number for which no message has been programmed, NO MESSAGE is displayed in the message window. NO MESSAGE may also be displayed if a transmission error from the message pointer register caused the data to be suspect.

Download from Www.Somanuals.com. All Manuals Search And Download.

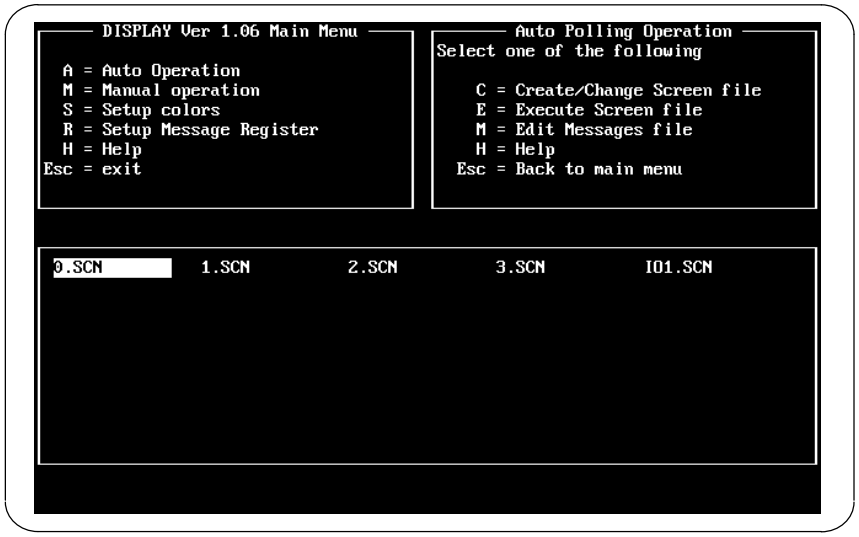
Auto-Polling During System Operation

This section explains how to:

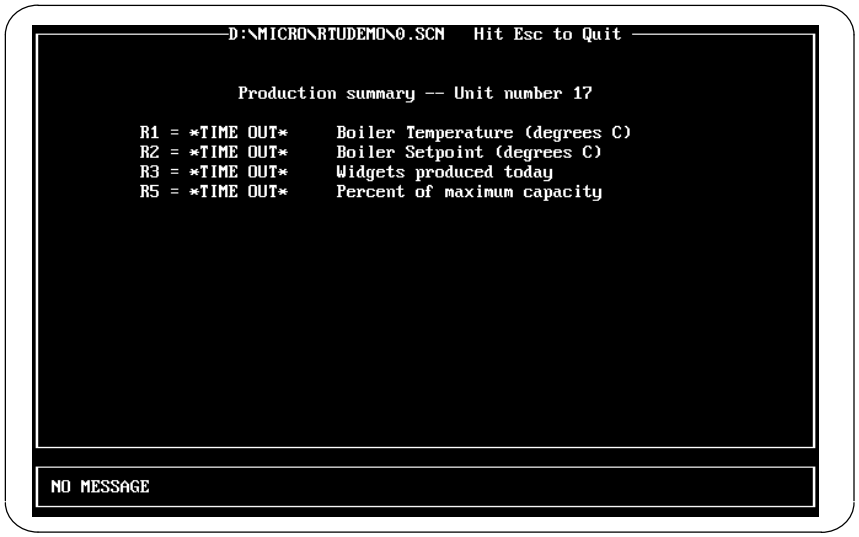
- Execute an Auto-polling screen
- Print the current screen
- Write a data value to a register
- Force an output
- Obtain register data from a disk file

Executing an Auto-polling Screen

To execute an Auto-polling screen, select E from the Auto-polling Operation menu You will see a directory listing of the available Auto-polling (.SCN) files which can be executed.



Move the cursor to the one you want and press the Enter key. If communications are properly established, the screen appears and the data begins to fill in. If communications have not been established, as in the example below, it will be indicated on the screen.



Download from Www.Somanuals.com. All Manuals Search And Download.

Changing the Display

If you want to return to the Auto mode menu, press the ESC key.

If you want to display another screen (and screens are named 0.SCN to 0.SCN), press the appropriate number key 0 through 9. The new screen will take a few seconds to start up.

Printing the Screen

If you want to print a copy of the screen, use the Shift–PrtSc keys. Note that the background data screen stops updating while the screen is printing.

Note

Be sure that you have a printer connected, or the system will hang up.

When you press Shift–PrtSc, the display freezes. That means the data on the screen will be partly old data and partly new data.

Writing a Value to a Register

If you want to write a register value (for example, for testing), press Control–W. Note that the background data screen stops updating during this function.

Note: if the CPU writes to the same register during its logic scan, the CPU's value will overwrite the value you entered manually.

Forcing an Output

If you want to force an output point high, press Control–H.

If you want to force an output point low, press Control–L. Note that the background data screen stops updating during this function.

Note: if the CPU writes to the same output during its logic scan, the CPU's point state will overwrite the state you entered manually.

Obtaining Register Data from a Disk File

This option could be used to preset a bank of registers for test purposes, or to download a new "recipe" to the PLC. Note that the background data screen stops updating during this function.

If you want to obtain a bank of registers from disk file, press Control–B.

Note: if the CPU writes to the same registers during its logic scan, the CPU's data will overwrite the data you obtained from the disk file.

Download from Www.Somanuals.com. All Manuals Search And Download.

Data Logging

This section describes:

- How Data Logging Works
- The .LOG File
- The Format of Logged Data

How Data Logging Works

During Auto-pollingn, data logging occurs at a selected time interval if some data on the screen is out of range. If all values are in range, no logging is done at the specified interval.

If any of the displayed values are out of range (indicated by a blinking data value followed by HIGH or LOW), and a non–zero logging time has been specified, all the screen data is saved to disk.

Data logging starts at the end of the screen update, immediately after a data item goes out of range. Subsequent data logs occur at the end of the first screen update that exceeds the specified logging time. For example, if you have specified an update time of 20 seconds, and your first log occurs at time=0, and the screen update time is 8 seconds, then the next log will occur at 24 seconds since this is the first ”end of screen update” beyond the 20 seconds specified.

If data logging is already occurring when a new item goes out of range, the screen data is logged at the next opportunity and the logging timer is set to 0. Subsequent data logs then occur at the scheduled intervals.

If the same item goes out of range, then in range, then out of range during the logging time period, the data logging records each in to out of range transition, and restarts the log timer for each “in to out of range” transition.

The .LOG File

Logged data is stored in a file with the same name as the current Auto-polling screen, anf the extension .LOG. For example, the file 2.LOG would contain the logged data for Auto-polling screen 2.SCN.

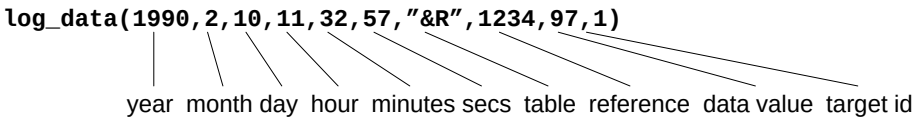
Renaming the .LOG File to Prevent Overwriting Data

When a screen starts executing, the utility automatically clears the .LOG file for that screen, even if no data is being saved yet. If there are contents in the file, they are cleared also. Therefore, if you want to save the .LOG file for future reference, be sure not to execute a screen with the same name before you have renamed the .LOG file using DOS.

Download from Www.Somanuals.com. All Manuals Search And Download.

The Format of Logged Data

Each data item on the Auto-polling screen at the time the log to disk starts is saved as a line in the .LOG file. Each line in the .LOG file has the following format:



These items are explained below.

Time Stamp (Year, Month, Day, Hour, Minutes, Seconds)

The time stamp represents the time logging occurred (not the time the actual data collection occurred). Therefore, the time stamp for each item on a screen is the same. If there are a lot of items on the screen and communications are running a low baud rate, it may take 10–20 seconds to update the screen, so the actual data collection times may vary by that amount.

Also, due to the time required to update the screen, the intervals between data logs to disk may not correspond exactly to the time interval specified in the setup string.

The time values used in the time stamp originate in the computer which is running the logging software, and not in the remote Micro PLC.

Data Type (Table, Reference, Data Value)

Although the logged data may represent inputs (&I), outputs (&O), or registers (&R), the data format is always decimal. In the example above, the value of Register 1234 was 97 decimal at the time logging occurred.

Target ID

The last part of a logged data value is the ID number of the Micro PLC that provided the data. This ID number is assigned (using the programming software) when configuring the Micro PLC.

Note that this ID may be different from the ID assigned with the DIP switches on the interface bos that connects the Micro PLC to the network. This is discussed in more detail in the appendix "Related Products" in the *Micro PLC Programmer's Manual.*

Download from Www.Somanuals.com. All Manuals Search And Download.

Error Messages During Operation

While you are executing a screen, the following errors may occur:

ERROR WRITING TO DEVICE COM1

You can test for the presence of COM 1 by typing >DIR >COM1. If you get a message "ERROR WRITING TO DEVICE COM1.... ", you probably need something else jumpered in your COM1 serial port, or you don't have a COM1 port defined, or physically available.

TIME OUT This message means that a response was not received from the Micro PLC. The problem could be lack of proper COM1 configuration, bad cabling, or an improperly set target ID in the Micro PLC.

CHECKSUM This message indicates that the received data had a checksum error. The integrity of the data is suspect. Usually, a checksum error is caused by a noise burst. It can also be caused by running at the wrong baud rate, or a baud rate higher than your computer can properly support when running DISPLAY (4800 baud for an AT, 2400 baud for an XT).

If this message appears only occasionally, you may need to lower the baud rate to the next lower value.

A checksum can also occur if too much data is requested from the Micro PLC in when using the software in manual mode. The ability of DISPLAY to handle large blocks of data depends on the baud rate, computer speed, and other factors.

WRONG ID This message means that the wrong remote device answered your polling request. This is not likely to occur, but the check is in there anyway.

WRONG HEADER / BAD REQUEST

Since the DISPLAY software uses only RTU functions 1–3 in auto mode, this is unlikely in automatic mode. In manual mode, you may get this error while experimenting. It will also occur if you ask for data beyond the end of a table, e.g. register 17K in a 16K system.

NO MESSAGE

This message appears in the message window if the message pointer register contains a value for which no message has been programmed. it also occurs if an error is detected in the data from the message pointer register.

Download from Www.Somanuals.com. All Manuals Search And Download.

Appendix F Programming Applications

This appendix describes some simple programming applications:

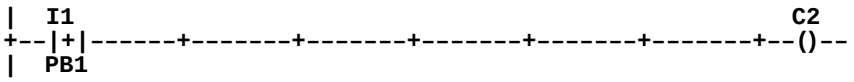
- Application #1: FLIP / FLOP (Toggle Operation)
- Application #2: Power Up One Shot (Start–up Protection)
- Application #3: Cascading Counters
- Application #4: Industrial “Starting Circuit”

Download from Www.Somanuals.com. All Manuals Search And Download.

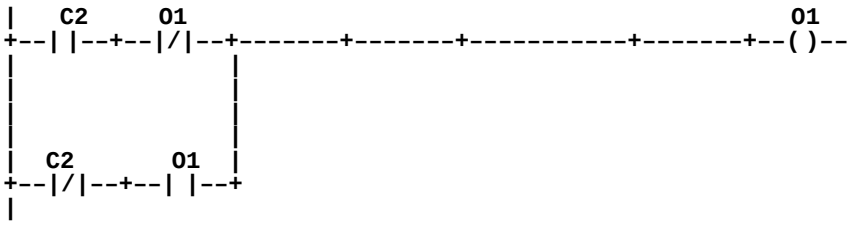
Application #1: FLIP / FLOP (Toggle Operation)

This logic reverses the state of an output each time a control signal is energized. Flip Flops can be used to toggle an output based on the presence of an input. A typical ladder logic is shown below.

Rung 1



Rung 2



In the logic shown above, push button 1 (PB1) is a control signal connected into input 1 of the Micro PLC. PB1 is a positive transition (one shot) contact. This means that each time PB1 is pushed, the PLC will "see" a positive signal for only one program cycle (about 6 micro sec) regardless of the duration of contact.

If PB1 is pushed at initial conditions (all inputs and outputs in their base state), output coil C2 comes on in rung 1. In the top branch of rung 2, contact C2 then closes, which in connection with the normally closed (N.C.) contact O1 activates output coil 1 (O1). During the next program cycle, C2 turns off **regardless of whether PB1 is still pressed.** With that, contact C2 in the top branch of rung 2 returns to its base normally open (N.O.) state. In addition, the normally closed contact O1 in the top branch is now open (reversed state) because of the activation of output coil O1. Therefore, the top branch of rung 2 no longer supports the activation of output coil 1. In the bottom branch, however, the N.O. contact O1 is closed because of the activation of output coil O1, while contact C2 returns to its base N.C. state. Therefore, the bottom branch now supports output coil 1.

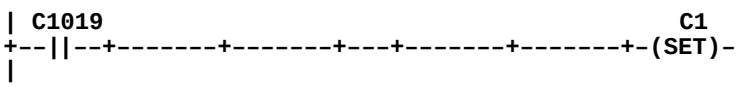
If PB1 is pushed when output coil 1 is on, N.C. contact C2 in the bottom branch opens and no longer supports the output coil. In the top rung, N.O. C2 closed, but N.C. O1 is open because output coil 1 is on. Therefore, neither branch supports output coil 1 and the coil goes off. The ladder has now returned to its initial state.

Download from Www.Somanuals.com. All Manuals Search And Download.

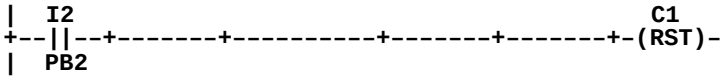
Application #2: Power Up One Shot (Start–up Protection)

This logic uses a special purpose coil called the Start–Up Scan Coil to provide protection in the event of power loss or stoppage in a program. The FLIP / FLOP program shown in Application 1 has been modified for this application. A typical ladder logic is shown below.

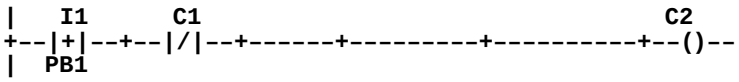
Rung 1



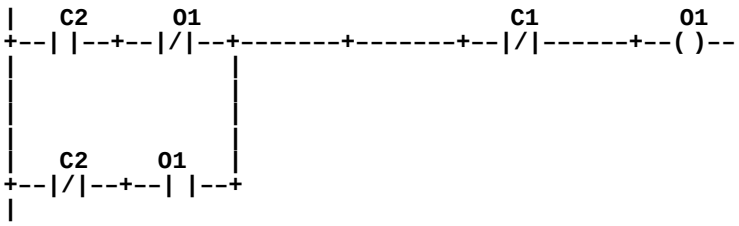
Rung 2



Rung 3



Rung 4



In the logic above, the start–up scan coil C1019 is used to set coil C1 in rung 1. Coil C1019 produces a positive output for a single program cycle when power is first applied or when the PLC is first put into RUN mode from a STOP / PROGRAM condition. At all other times, C1019 is off. Because C1 in rung 1 is defined as being in a set condition, C1 turns on when C1019 turns on and will remain on until a reset condition is activated. Push button 2 (PB2) in rung 2 activates the reset condition in this ladder.

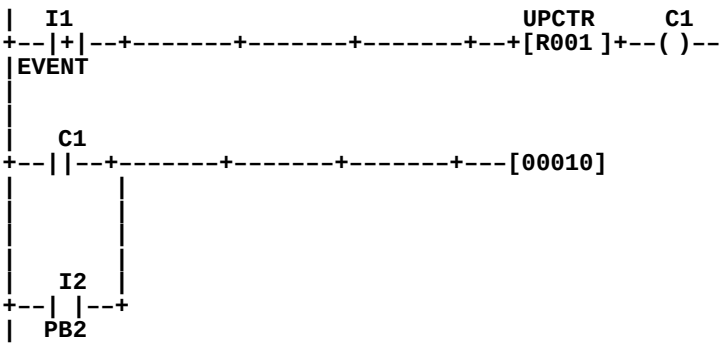
In rungs 3 and 4, N.C. contact C1 is placed immediately before the output coils. These contacts are open when C1 is set, which prevents either rung from being executed until PB2 is pushed. Once C1 is reset, both N.C. contacts return to their base states and the program performs normally.

Download from Www.Somanuals.com. All Manuals Search And Download.

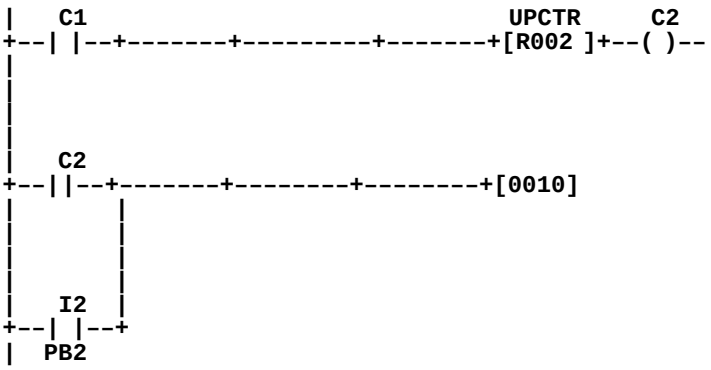
Application #3: Cascading Counters

This logic provides a technique for cascading counters. Cascading allows for counting more events than a single counter is able to count. A typical ladder logic is shown below.

Rung 1



Rung 2



In the logic above, I1 represents the event to be measured through input 1 on the Micro PLC. I1 is programmed as a positive transition contact so the counter only counts each activation of I1. The second input to the counter resets the counter. When an event occurs, it is counted in register R001. When the count reaches 10, output coil C1 is activated. Contact C1 in the bottom branch of rung 1 turns on and resets the counter. Output coil C1 also turns on the first input for the counter in rung 2. Therefore, the counter in rung 2 places one count in register R002 for every ten counts in register R1. Input I2 resets both counters.

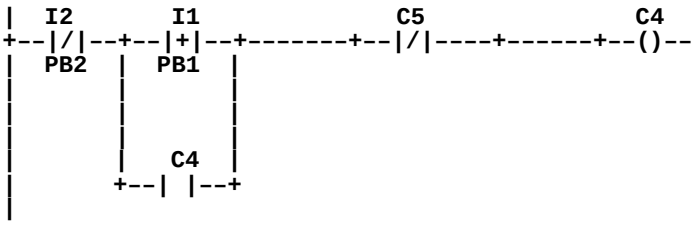
The counters in the Micro PLC can count from 0 to 32767. It may be advantageous, however, to structure the counters in an arrangement where the first counter records values up to ten thousand. The second counter would then record higher order values (tens of thousands, hundreds of thousands, millions, etc.).

Download from Www.Somanuals.com. All Manuals Search And Download.

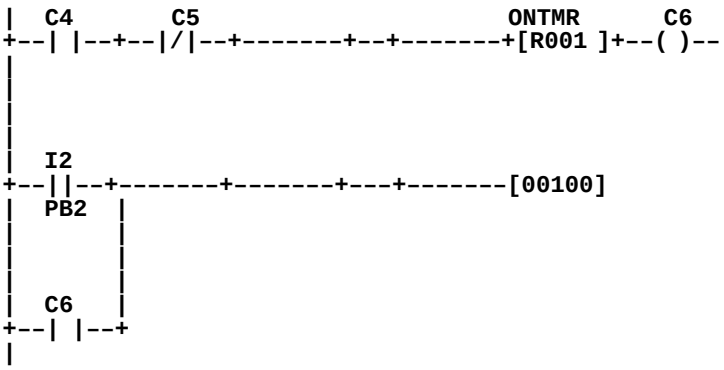
Application #4: Industrial "Starting Circuit"

This logic provides a sample starting circuit for an industrial application. Included in the circuit is a simulation of a time delay relay and an emergency stop feature. In your application other safety interlocks may need to be included in the logic. A typical ladder logic is shown below.

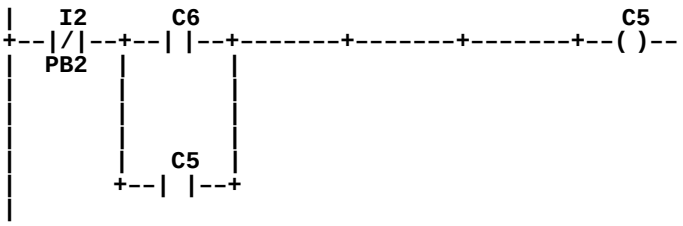
Rung 1



Rung 2



Rung 0003



Download from Www.Somanuals.com. All Manuals Search And Download.

Rung 4

```
|   C4     C5                                  O1
+--| |--+--|/|--+----------+-------+-------+--( )--
|
```

Rung 5

```
|  C5                                           O2
+--| |--+-------+-----------+-------+-------+--( )--
|
```

In the logic listed above, push button 1 (PB1), connected to input 1 of the Micro PLC, is the start button for an industrial process, while push button 2 (PB2) is the emergency stop button connected to input 2. If PB2 is pressed at any time, the process will terminate. When PB1 is pressed, Coil C4 is activated, which in turn closes contact C4 in rung 1, latching the circuit. C4 initiates a ten second timer sequence in rung 2. The parameter 100 in the timer R001 specifies the delay in tenth of a seconds (100 x 0.1 sec = 10 sec). Once the time has reached the ten second mark, output coil C6 is turned on. C6 also automatically resets the timer by closing contact C6 on the third branch of rung 2. In rung 3, contact C6 closes, activating output coil C5. Normally closed contact C5 opens in both rung 1 and rung 2, disabling the initial start mechanism and the timer.

Output coil O1 in rung 4, which corresponds to output 1 on the Micro PLC, is activated during the 10 second time delay. The output could be used to drive a ”start indicator” light or to begin a process that occurs during the time delay. Output coil O2 in rung 5, which corresponds to output 2, is activated at the end of the start–up process. This output could be used either to start the industrial process itself or to drive a ”run indicator” light.

Download from Www.Somanuals.com. All Manuals Search And Download.

A

Addition function, 4-22

Addresses, 1-6
  communications, B-3

Analog Expander, scaling and references, 1-8

B

Block Move function, 4-32

C

Change directory, A-2

Coils
  Clock (C1018), 1-7
  creating with programming software, 4-11
  Hold Output (C1021), 1-7
  internal, 1-6
  MCR / End MCR pair, 4-14
  output coil, 4-12
  Set / Reset pair, 4-13
  Skip / End pair, 4-15
  Startup Scan (C1019), 1-7
  types of, 4-11
  used in pairs, 4-11

Communication Errors (RTU), C-18

Communications, drivers and demo programs, B-2

Communications data format, B-4

Communications functions
  IBM Compiler BASIC, B-12
  Microsoft C
    large model, B-8
    small model, B-9
  sample programs, B-13
  Turbo C
    large model, B-10
    small model, B-11

Communications protocol, B-4

Compare functions, types of, 4-36

Constants in a program, 1-6

Contacts
  edit with programming software, 2-10
  enter with programming software, 4-4
  Negative Transition, 4-10
  Normally-closed, 4-7
  Normally-open, 4-5
  Positive Transition, 4-9
  types of, 4-4

Counters
  cascading, F-4
  Down Counter, 4-21
  registers, 1-6
  types of, 4-19
  Up Counter, 4-20

Cyclic Redundancy Check (CRC), C-5
  Calculating the CRC–16, C-6

D

Data acquisition, E-1

Directories, using, A-1

Display, software program, E-1

Division function, 4-28

Down Counter, 4-21

E

Emergency stop example, F-5

Error Codes, RTU Error Responses, C-18

Exclusive OR function, 4-41

F

Field Service assistance, iv

Flip-flop, F-2

Force Single Output (RTU message), C-13

H

Horizontal line in a rung, 2-7 , 2-11

I

Inclusive OR function, 4-40

Indirect Move function, 4-34

Instruction set, 1-4

Download from Www.Somanuals.com. All Manuals Search And Download.

L

Label display, 2-2 , 2-8

Labels, create with programming software, 2-8

Ladder logic format, 1-3

Loading files, A-1

Logging, E-1

Logic Operations
- Exclusive OR, 4-41
- Inclusive OR, 4-40
- NOT, 4-44
- Shift Register Left, 4-43
- Shift Register Right, 4-42
- types of, 4-38
- Word AND, 4-39

M

Master Control Relay coil, 4-14

Math functions
- Addition, 4-22
- Division, 4-28
- Multiplication, 4-26
- Subtraction, 4-24
- types of, 4-22

Maximum constant or register value, 1-6

MCROCOMM.C file, B-8

Memory addresses, 1-6
- communications, B-3

MICRO.CFG file, A-2

Move function, 4-30

Move functions
- Block Move, 4-32
- enter with programming software, 4-30
- Indirect Move, 4-34
- Move, 4-30
- types of, 4-30

Multiplication function, 4-26

N

NOT function, 4-44

O

Off Timer, 4-18

Offline functions of programming software, 2-2

On Timer, 4-17

One-shot, F-3

Output
- control reference with Set / Reset coils, 4-13
- control with Output Coil, 4-12

Output coil, 4-12

Outputs
- hold last state with Skip coil, 4-15
- On or Off in Stop mode, 1-7
- set to 0 in program, 4-14

P

Power flow in a program, 1-3

Powerup one-shot, F-3

Preset Multiple Registers (RTU message), C-16

Preset Single Register (RTU message), C-14

Program, search, programming software, 2-2

Program entry, HHP, 3-3
- deleting program elements, 3-5
- examples, 3-8
- insert a rung, 3-4
- searching, 3-7

Program format, overview, 1-5

Program listing, HHP, 3-2

Program syntax, check with programming software, 2-2

Program transfer, HHP, 3-2

Programming
- basics, 1-2
- Instruction set, 1-4

Programming functions
- map of, 2-3
- on HHP, 4-2

Protocol Definition, RTU Protocol, C-1

Pulse generator coil, 1-7

Download from Www.Somanuals.com. All Manuals Search And Download.

Q

Query–Response, C-1

R

Read Analog Inputs (RTU message), C-12
Read Exception Status (RTU message), C-15
Read Input Table (RTU message), C-10
Read Output Table (RTU message), C-9
Read Registers (RTU message), C-11
Reference, enter with programming software, 4-4
Registers, 1-6
Report Device Type (RTU Message), C-17
RTU Character Format, C-4
RTU Communication Errors, C-18
RTU Message Descriptions, C-9
  Force Single Output, C-13
  Preset Multiple Registers, C-16
  Preset Single Register, C-14
  Read Analog Inputs, C-12
  Read Exception Status, C-15
  Read Input Table, C-10
  Read Output Table, C-9
  Read Registers, C-11
  Report Device Type, C-17
RTU Message Fields, C-3
  Error Check Field, C-3
  Function Code, C-3
  Information Field, C-3
  Station Address, C-3
RTU Message Length, C-8
RTU Message Termination, C-4
RTU Message Types, C-2
  Broadcast, C-2
  Error Response, C-2
  Normal Response, C-2
  Query, C-2
RTU Protocol, C-1 , E-2
RTU Timeout Usage, C-4
Rung
  accept with programming software, 2-6
  copy with programming software, 2-14
  create with programming software, 2-2 , 2-4 , 2-6
  delete with HHP, 3-5
  delete with programming software, 2-2 , 2-13
  edit with HHP, 3-4 , 3-6
  edit with programming software, 2-2 , 2-6 , 2-9
  move with programming software, 2-13
  search with HHP, 3-7
  search with programming software, 2-15

S

Saving files, A-1
Search
  with HHP, 3-7
  with programming software, 2-15
Set / Reset coils, 4-13
Shift Register Left function, 4-43
Shift Register Right function, 4-42
Skip / End coils, 4-15
Startup scan coil, 1-7
Subtraction function, 4-24

T

Technical assistance, iv
Time delay relay example, F-5
Timeouts (RTU), C-4 , C-19
Timers
  creating with programming software, 4-16
  Off Timer, 4-18
  On Timer, 4-17
  registers, 1-6
  types of, 4-16

U

Up Counter, 4-20

V

Vertical line in a rung, 2-7

W

Word AND function, 4-39

Download from Www.Somanuals.com. All Manuals Search And Download.

Free Manuals Download Website

http://myh66.com

http://usermanuals.us

http://www.somanuals.com

http://www.4manuals.cc

http://www.manual-lib.com

http://www.404manual.com

http://www.luxmanual.com

http://aubethermostatmanual.com

Golf course search by state

http://golfingnear.com

Email search by domain

http://emailbydomain.com

Auto manuals search

http://auto.somanuals.com

TV manuals search

http://tv.somanuals.com