Allen-Bradley

SLC 500™ Instruction Set

Catalog Numbers 1747-L20x, 1747-L30x, 1747-L40x, 1747-L511, 1747-L514, 1747-L524, 1747-L531, 1747-L532, 1747-L541, 1747-L542, 1747-L543, 1747-L551, 1747-L552, 1747-L553

Reference Manual

Rockwell Automation

Important User Information

Because of the variety of uses for the products described in this publication, those responsible for the application and use of this control equipment must satisfy themselves that all necessary steps have been taken to assure that each application and use meets all performance and safety requirements, including any applicable laws, regulations, codes and standards.

The illustrations, charts, sample programs and layout examples shown in this guide are intended solely for purposes of example. Since there are many variables and requirements associated with any particular installation, Allen-Bradley does not assume responsibility or liability (to include intellectual property liability) for actual use based upon the examples shown in this publication.

Allen-Bradley publication SGI-1.1, *Safety Guidelines for the Application, Installation and Maintenance of Solid-State Control* (available from your local Allen-Bradley office), describes some important differences between solid-state equipment and electromechanical devices that should be taken into consideration when applying products such as those described in this publication.

Throughout this manual we use notes to make you aware of safety considerations:

ATTENTION

Identifies information about practices or circumstances that can lead to personal injury or death, property damage or economic loss

Attention statements help you to:

- identify a hazard
- avoid a hazard
- recognize the consequences

IMPORTANT

Identifies information that is critical for successful application and understanding of the product.

The information below summarizes the changes to this manual since the last printing.

To help you find new and updated information in this release of the manual, we have included change bars as shown next to this paragraph.

The table below lists the sections that document new features and additional or updated information about existing features.

For this information	See
Removed references to MicroLogix 1000.	Throughout
Created section explaining Data Files and Program Files	Chapter 1
Removed Processor Files information. Revisions to the Basic Instructions.	Chapter 2
Revisions to the Comparison Instructions.	Chapter 3
Revisions to the Math Instructions. Added Ramp Instruction (RMP).	Chapter 4
Revisions to the Data Handling Instructins. Added Encode 1 to 16 to 4 Instruction (ENC).	Chapter 5
Revisions to the Program Flow Instructions	Chapter 6
Revisions to the Application Specific Instructions. Added the following instructions: Read High-Speed Clock Instruction (RHC), Compute Time Difference Instruction (TDF), File Bit Comparison Instruction (FBC) and Diagnostic Detect (DDT).	Chapter 7
Added Remote I/O Block Transfer Instructions (BTR and BTW).	Chapter 8
Revisions to the PID Instruction.	Chapter 9
Revisions to the ASCII Instructions.	Chapter 10
Revisions to the Interrupt Routines.	Chapter 11
Revisions to the Communication Instructions.	Chapter 12
Created section for Communication Channels.	Chapter 13
Created section for Messaging Examples.	Chapter 14
Revisions to the Troubleshooting section.	Chapter 15
Removed execution times, combined the memory usage tables for SLC 5/03, SLC 5/04 and SLC 5/05. Added memory usage for new instructions.	Appendix C
Removed the estimating scan time section. Added new instructions to the Programming Instruction References section.	Appendix D
Updated ladder diagrams for the application examples.	Appendix G
Alphabetical list of all instructions.	Inside Back Cover

Table of Contents

Chapter 5

Data Handling Instructions

Chapter 10

ASCII Instructions

Chapter 11

Understanding Interrupt Routines

Chapter 12

SLC Communication Instructions

Chapter 13

SLC Communication Channels

Chapter 14

Messaging Examples

Chapter 15

Troubleshooting Faults

Appendix A

SLC 5/03 (OS30x), SLC 5/04 (OS40x) and SLC 5/05 (OS50x) Firmware History

Read this preface to familiarize yourself with the rest of the manual. It provides information concerning:

- who should use this manual
- the purpose of this manual
- related documentation
- conventions used in this manual
- Rockwell Automation support

Who Should Use this Manual

Use this manual if you are responsible for designing, installing, programming, or troubleshooting control systems that use SLC 500 processors.

You should have a basic understanding of electrical circuitry and familiarity with relay logic. If you do not, obtain the proper training before using this product.

Purpose of this Manual

This manual is a reference guide for SLC 500 processors. This manual:

- provides status file functions
- provides the instructions used in your ladder logic programs
- compliments the online help available at the terminal

Common Techniques Used in this Manual

The following conventions are used throughout this manual:

- Bulleted lists such as this one provide information, not procedural steps.
- Numbered lists provide sequential steps or hierarchical information.
- *Italic* type is used for emphasis.

Rockwell Automation Support

Rockwell Automation offers support services worldwide, with over 75 Sales/Support Offices, 512 authorized Distributors and 260 authorized Systems Integrators located throughout the United States alone, plus Rockwell Automation representatives in every major country in the world.

Local Product Support

Contact your local Rockwell Automation representative for:

- sales and order support
- product technical training
- warranty support
- support service agreements

Technical Product Assistance

If you need to contact Rockwell Automation for technical assistance, please review the *Troubleshooting* chapter on page 15-1 in this manual first. Then call your local Rockwell Automation representative.

Your Questions or Comments on this Manual

If you find a problem with this manual, or you have any suggestions for how this manual could be made more useful to you, please contact us at the address below:

Rockwell Automation
Automation Control and Information Group
Technical Communication, Dept. A602V
P.O. Box 2086
Milwaukee, WI 53201-2086

or visit our internet page at:
http://www.ab.com or http://www.rockwellautomation.com

Read this preface to familiarize yourself with the rest of the manual. It provides information concerning:

- who should use this manual
- purpose of this manual
- conventions used in this manual

Related Documentation

The following documents contain additional information concerning Allen-Bradley SLC products. To obtain a copy, contact your local Allen-Bradley office or distributor.

For	Read this Document
An overview of the SLC 500 family of products	SLC 500 System Overview, Publication Number 1747-SO001B-EN-P.
A description on how to install and use your Fixed SLC 500 programmable controller	Installation and Operation Manual for Fixed Hardware Style Programmable Controllers, Publication Number 1747-6.1
A description on how to install and use your Modular SLC 500 programmable controller	Installation and Operation Manual for Modular Hardware Style Programmable Controllers, Publication Number 1747-6.2
A glossary of industrial automation terms and abbreviations	Allen-Bradley Industrial Automation Glossary, Publication Number AG-7.1

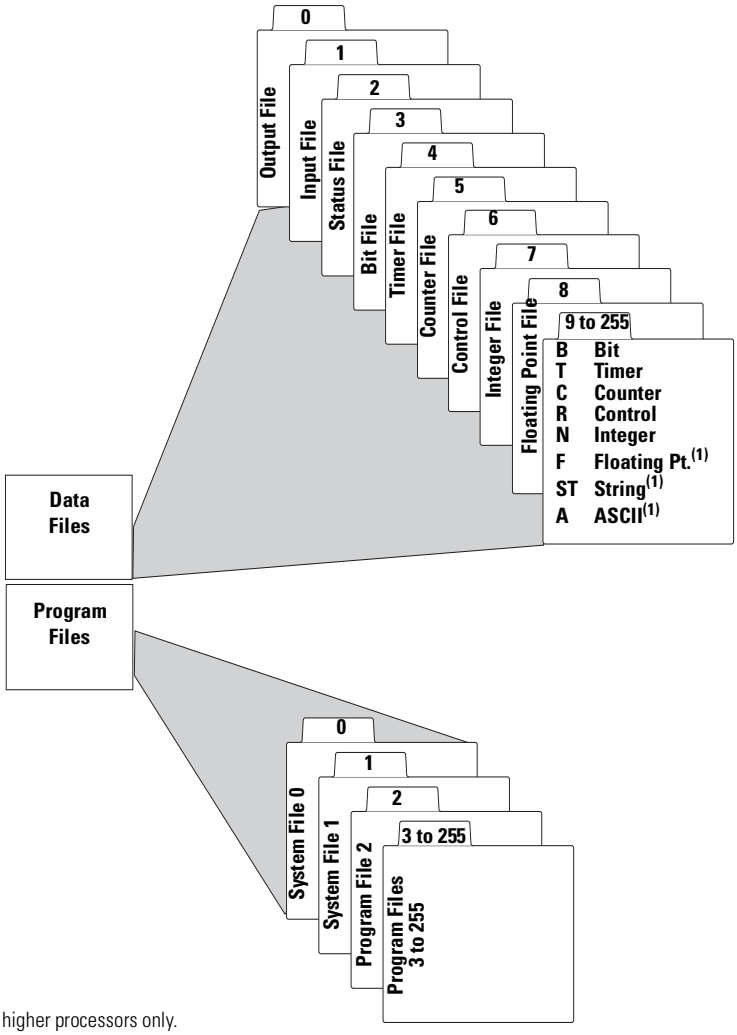
Processor Files

File Structure

SLC 500 user memory is comprised of Data Files and Program Files.

TIP

The file types shown below for data files 3 through 8 are the default values. Files 9 to 255 can be configured to be bit, timer, counter, control, integer, floating point, ASCII, or String files.



(1) SLC 5/03 and higher processors only.

Output and Input Data Files (Files O0: and I1:)

Data Files 0 and 1 represent external outputs and inputs, respectively. Bits in file 1 are used to represent external inputs. In most cases, a single 16-bit word in these files will correspond to a slot location in your controller, with bit numbers corresponding to input or output terminal numbers. Unused bits of the word are not available for use.

Table 1.1 explains the addressing format for outputs and inputs. Note that the format specifies **e** as the slot number and **s** as the word number. When referencing output and input data file words, refer to the element as **e.s** (slot and word), taken together.

Table 1.1 Output and Input Data File Addressing Formats

Format	Explanation		
O:e.s/b **I:e.s/b**	**O**	Output	
	I	Input	
	:	Element delimiter	
	e	Slot number (decimal)	Slot 0, adjacent to the power supply in the first chassis, applies to the processor module (CPU). Succeeding slots are I/O slots, numbered from 1 to a maximum of 30.
	.	Word delimiter.	Required only if a word number is necessary as noted below.
	s	Word number	Required if the number of inputs or outputs exceeds 16 for the slot. Range: 0-255 (range accommodates multi-word "specialty cards")
	/	Bit delimiter	
	b	Terminal number	Inputs: 0-15 Outputs: 0-15
Examples:			
O:3/15 **O:5/0** **O:10/11** **I:7/8** **I:2.1/3**	Output 15, slot3 Output 0, slot 5 Output 11, slot 10 Input 8, slot 7 Input 3, slot 2, word 1		
O:5 **O:5.1** **I:8**	Output word 0, slot 5 Output word 1, slot 5 Input word 0, slot 8		
Default Values: Your programming device will display an address more formally. For example, when you assign the address O:5/0, the programming device will show it as O:5.0/0 (Output file, slot 5, word 0, terminal 0).			

Status File (File S2:)

You cannot add to or delete from the status file. See Table 1.2 to understand how to address various bits and words within the status file. You can address various bits and words as follows:

Table 1.2 Status File Addressing Format

Format	Explanation		
S:e/b	**S**	Status file	
	:	Element delimiter	
	e	Element number	Ranges from 0 to 15 in a fixed or SLC 5/01 controller, 0 to 32 in an SLC 5/02, 0 to 82 in an SLC 5/03 and 0 to 82 in an SLC 5/05, 0 to 96 in an SLC 5/04 OS400, and 0 to 163 in an SLC 5/04 OS401 processors. These are 1-word elements. 16 bits per element.
	/	Bit delimiter	
	b	Bit number	Bit location within the element. Ranges from 0 to 15.
Examples:			
S:1/15	Element 1, bit 15. This is the “first pass” bit, which you can use to initialize instructions in your program.		
S:3	Element 3. The lower byte of this element is the current scan time. The upper byte is the watchdog scan time.		

Bit Data File (B3:)

File 3 is the bit file, used primarily for bit (relay logic) instructions, shift registers, and sequencers. The maximum size of the file is 256 1-word elements, a total of 4096 bits. You can address bits by specifying the element number (0 to 255) and the bit number (0 to 15) within the element. You can also address bits by numbering them in sequence, 0 to 4095.

You can also address elements of this file. See Table 1.3 for a detailed format description. Note the two different possible formats that can be used.

Table 1.3 Bit File Addressing Format

Format	Explanation		
Bf:e/b	**B**	Bit type file	
	f	File number. Number 3 is the default file. A file number between 9-255 can be used if additional storage is required.	
	:	Element delimiter	
	e	Element number	Ranges from 0-255. These are 1-word elements. 16 bits per element.
	/	Bit delimiter	
	b	Bit number	Bit location within the element. Ranges from 0-15.
Bf/b	**B** **f** **/**	Same as above. Same as above. Same as above.	
	b	Bit number	Numerical position of the bit within the file. Ranges from 0-4095.
Examples:			
B3:3/14	Bit 14, element 3		
B3:252/00	Bit 0, element 252		
B3:9	Bit 62		
B3/62	Bit 62		
B3/4032	Bit 4032		

Timer Data File (T4:)

TIP

Timing could be inaccurate if Jump (JMP), Label (LBL), Jump to Subroutine (JSR), or Subroutine (SBR) instructions skip over the rung containing a timer instruction while the timer is timing. If the skip duration is less than 2.5 seconds, no time will be lost; if the skip duration exceeds 2.5 seconds, an undetectable timing error occurs. When using subroutines, a timer must be executed at least every 2.5 seconds to prevent a timing error.

Timer instructions use various control bits. These are 3-word elements, used with Bit, TON, TOF and RTO instructions. Word 0 is the status word, word 1 indicates the preset value, and word 2 indicates accumulator value. This is shown in Table 1.4.

Table 1.4 Timer Control Fields

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Word
EN	TT	DN							Internal Use(1)							0
Preset Value (PRE)																1
Accumulator Value (ACC)																2

(1) Bits labeled "Internal Use" are not addressable.

Addressable Bits	Addressable Words
EN = Enable (Bit 15)	PRE = Preset Value
TT = Timer Timing (Bit 14)	ACC = Accumulated Value
DN = Done (Bit 13)	

Addressing Structure

Address bits and words using the format Tf:e.s/b

Table 1.5 Timer Addressing Format

Explanation		
T	Timer file	
f	File number. For SLC 500 processors the default is 4. A file number between 9 to 255 can be used for additional storage.	
:	Element delimiter	
e	Element number	These are 3-word elements. The range is 0 to 255.
.	Word Delimiter	Range 0 to 2
s	Word Number	
/	Bit delimiter	
b	Bit Number	Range 0 to 15
Examples		
T4:0/15 or T4:0/EN	Enable bit	
T4:0/14 or T4:0/TT	Timer timing bit	
T4:0/13 or T4:0/DN	Done bit	
T4:0.1 or T4:0.PRE	Preset value of the timer	
T4:0.2 or T4:0.ACC	Accumulated value of the timer	
T4:0.1/0 or T4:0.PRE/0	Bit 0 of the preset value	
T4:0.2/0 or T4:0.ACC/0	Bit 0 of the accumulated value	

Counter Data File Elements (C5:)

Each Counter address is made of a 3-word data file element. Word 0 is the control word, containing the status bits of the instruction. Word 1 is the preset value. Word 2 is the accumulated value.

The control word for counter instructions includes five status bits, as indicated below.

Table 1.6 Counter Control Fields

15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00	Word
CU	CD	DN	OV	UN	UA(1)				Internal Use(2)							0
Preset Value (PRE)																1
Accumulator Value (ACC)																2

(1) Fixed SLC 500 only.

(2) Bits labeled "Internal Use" are not addressable.

Addressable Bits	Addressable Words
CU = Count up enable (Bit 15)	PRE = Preset
CD = Count down enable (Bit 14)	ACC = Accum
DN = Done bit (Bit 13)	
OV = Overflow bit (Bit 12)	
UN = Underflow bit (Bit 11)	
UA = Update Accumulator bit (Bit 10) (Fixed Controller Only)	

Entering Parameters

There are several parameters associated with Counter instructions. The following parameters detail the operations of the counter.

Accumulator Value (ACC)

This is the number of false-to-true transitions that have occurred since the counter was last reset.

Preset Value (PRE)

Specifies the value which the counter must reach before the controller sets the done bit (DN). When the accumulator value becomes equal to or greater than the preset value, the done status bit is set. You can use the done bit (DN) to control an output device.

Preset and accumulated values for counters range from -32,768 to +32,767, and are stored as signed integers. Negative values are stored in two's complement form.

Addressing Structure

Assign counter addresses using the format **Cf:e.s/b**

Table 1.7 Counter File Addressing Format

Explanation		
C	Counter	
f	File number. For SLC 500 processors the default is 5. A file number between 9 to 255 can be used for additional storage.	
:	Element delimiter	
e	Element number	These are 3-word elements. The range is 0 to 255.
.	Word Delimiter	
s	Word Element	0 to 2
/	Bit delimiter	
b	Bit Number	0 to 15
Examples		
C5:0/15 or C5:0/CU	Count up enable bit	
C5:0/14 or C5:0/CD	Count down enable bit	
C5:0/13 or C5:0/DN	Done bit	
C5:0/12 or C5:0/OV	Overflow bit	

Table 1.7 Counter File Addressing Format

Explanation	
C5:0/11 or C5:0/UN	Underflow bit
C5:0/10 or C5:0/UA	Update accum. bit (use with HSC in fixed controller only)
C5:0.1 or C5:0.PRE	Preset value of the counter
C5:0.2 or C5:0.ACC	Accumulated value of the counter
C5:0.1/0 or C5:0.PRE/0	Bit 0 of the preset value
C5:0.2/0 or C5:0.ACC/0	Bit 0 of the accumulated value

Control Data File (R6:)

These instructions use various control bits. These are 3-word elements, used with bit shift, FIFO, LIFO, sequencer instructions, and ASCII instructions ABL, ACB, AHL, ARD, ARL, AWA, and AWT. Word 0 is the status word, word 1 indicates the length of stored data, and word 2 indicates position. This is shown in Table 1.8.

In the control element, there are eight status bits and an error code byte. A fixed controller and an SLC 5/01 control element has six bits. Bits EU and EM are not used by the processor.

Table 1.8 Three Word Element Structure

15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00	Word
EN	EU(1)	DN	EM(1)	ER	UL	IN	FD	Error Code								0
Length of Bit Array or File (LEN)																1
Bit Pointer or Position (POS)																2

(1) Not used in Fixed or SLC 5/01.

Addressable Bits	Addressable Words
EN = Enable	LEN = Length
EU = Update Enable	POS = Position
DN = Done	
EM = Stack Empty	
ER = Error	
UL = Unload	
IN = Inhibit	
FD = Found	

Assign control addresses as follows:

Table 1.9 Control File Addressing Format

Format	Explanation		
Rf:e	**R**	Control file	
	f	File number. Number 6 is the default file. A file number between 9 and 255 can be used if additional storage is required.	
	:	Element delimiter	
	e	Element number	Ranges from 0 to 255. These are 3-word elements. See figure above.
Rf:e.s/b	**Rf:e**	Explained above.	
	.	Word delimiter	
	s	Indicates word	
	/	Bit delimiter	
	b	Bit	
Examples:			
R6:2	Element 2, control file 6 Address bits and words by using the format Rf:e.s/b		
R6:2/15 or R6:2/EN **R6:2/14 or R6:2/EU** **R6:2/13 or R6:2/DN** **R6:2/12 or R6:2/EM** **R6:2/11 or R6:2/ER** **R6:2/10 or R6:2/UL** **R6:2/9 or R6:2/IN** **R6:2/8 or R6:2/FD** **R6:2.1 or R6:2.LEN** **R6:2.2 or R6:2.POS**	Enable bit Unload Enable bit Done bit Stack Empty bit Error bit Unload bit Inhibit bit Found bit Length value Position value		
R6:2.1/0 **R6:2.2/0**	**Bit 0 of length value** **Bit 0 of position value**		

Integer Data File (N7:)

Use these addresses as your program requires. These are 1-word elements, addressable at the element and bit level.

Assign integer addresses as follows:

Table 1.10 Integer File Addressing Format

Format	Explanation		
Nf:e/b	**N**	Integer file	
	f	File number. Number 7 is the default file. A file number between 9 to255 can be used if additional storage is required.	
	:	Element delimiter	
	e	Element number	Ranges from 0 to 255. These are 1-word elements. 16 bits per element.
	/	Bit delimiter	
	b	Bit number	Bit location within the element. Ranges from 0 to 15.
Examples:			
N7:2 **N7:2/8** **N10:36**	Element 2, integer file 7 Bit 8 in element 2, integer file 7 Element 36, integer file 10 (file 10 designated as an integer file by the user)		

Float Data File (F8:)

Use these addresses as your program requires. These are 2-word elements, addressable at the element and bit level.

Assign float addresses as follows:

Table 1.11 Float File Addressing Format

Format	Explanation		
Ff:e	**F**	Integer file	
	f	File number. Number 8 is the default file. A file number between 9 to 255 can be used if additional storage is required.	
	:	Element delimiter	
	e	Element number	Ranges from 0 to 255. These are 2-word elements. 32 bits per element.
Examples:			
F8:2	Element 2, float file 8		

TIP Float data type cannot be accessed at the bit level.

Basic Instructions

This chapter contains general information about the basic instructions and explains how they function in your application program. Each of the basic instructions includes information on:

- the instruction symbol
- the instruction format
- the instruction usage

The Basic Instructions detailed in this chapter are listed in Table 2.1

Table 2.1 Basic Instructions

Instruction Mnemonic	Instruction Name	Purpose	Page
XIC	Examine if Closed	Examines a bit for an On condition.	2-3
XIO	Examine if Open	Examines a bit for an Off condition.	2-3
OTE	Output Energize	Turns a bit On or Off.	2-4
OTL and **OTU**	Output Latch and Output Unlatch	OTL turns a bit on when the rung is executed, and this bit retains its state when the rung is not executed or a power cycle occurs. OTU turns a bit off when the rung is executed, and this bit retains its state when the rung is not executed or when power cycle occurs.	2-4
OSR	One-Shot Rising	Triggers a one-time event.	2-5
TON	Timer On-Delay	Counts timebase intervals when the instruction is true.	2-9
TOF	Timer Off-Delay	Counts timebase intervals when the instruction is false.	2-10
RTO	Retentive Timer	Counts timebase intervals when the instruction is true and retains the accumulated value when the instruction goes false or when power cycle occurs.	2-11
CTU	Count Up	Increments the accumulated value at each false-to-true transition and retains the accumulated value when the instruction goes false or when power cycle occurs.	2-13
CTD	Count Down	Decrements the accumulated value at each false-to-true transition and retains the accumulated value when the instruction goes false or when power cycle occurs.	2-14
HSC	High-Speed Counter	Counts high-speed pulses from a fixed controller high-speed input.	2-15
RES	Reset	Resets the accumulated value and status bits of a timer or counter. Do not use with TOF timers.	2-20

About the Basic Instructions

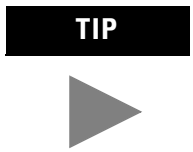
Basic instructions, when used in ladder programs, represent hardwired logic circuits used for the control of a machine or equipment.

The basic instructions are separated into three groups: bit, timer, and counter. Before you learn about the instructions in each of these groups, we suggest that you read the overviews that follow:

- Bit Instructions Overview
- Timer Instructions Overview
- Counter Instructions Overview

Bit Instructions Overview

Bit instructions operate on a single bit of data. During operation, the processor may set or reset the bit, based on logical continuity of ladder rungs. You can address a bit as many times as your program requires.



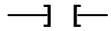
Using the same address with multiple output instructions is not recommended.

Bit instructions are used with the following data files:

- Output/Input Files
- Status File
- Bit File
- Timer File

Examine if Closed (XIC)

Use the XIC instruction in your ladder program to determine if a bit is On. When the instruction is executed, if the bit addressed is on (1), then the instruction is evaluated as true. When the instruction is executed, if the bit addressed is off (0), then the instruction is evaluated as false.



Fixed	SLC 5/01	SLC 5/02	SLC 5/03	SLC 5/04	SLC 5/05
•	•	•	•	•	•

Input Instruction

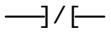
Bit Address State	XIC Instruction
0	False
1	True

Examples of devices that turn on or off include:

- a push button wired to an input (addressed as I:0/4)
- an output wired to a pilot light (addressed as O:0/2)
- a timer controlling a light (addressed as T4:3/DN)

Examine if Open (XIO)

Use the XIO instruction in your ladder program to determine if a bit is Off. When the instruction is executed, if the bit addressed is off (0), then the instruction is evaluated as true. When the instruction is executed, if the bit addressed is on (1), then the instruction is evaluated as false.



Fixed	SLC 5/01	SLC 5/02	SLC 5/03	SLC 5/04	SLC 5/05
•	•	•	•	•	•

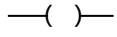
Input Instruction

Bit Address State	XIO Instruction
0	True
1	False

Examples of devices that turn on or off include:

- motor overload normally closed (N.C.) wired to an input (I:0/10)
- an output wired to a pilot light (addressed as O:0/4)
- a timer controlling a light (addressed as T4:3/DN)

Output Energize (OTE)



Fixed	SLC 5/01	SLC 5/02	SLC 5/03	SLC 5/04	SLC 5/05
•	•	•	•	•	•

Output Instruction

Use the OTE instruction in your ladder program to turn on a bit when rung conditions are evaluated as true.

An example of a device that turns on or off is an output wired to a pilot light (addressed as O:0/4).

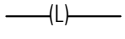
OTE instructions are reset when:

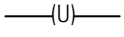
- The SLC enters or returns to the REM Run or REM Test mode or power is restored.
- The OTE is programmed within an inactive or false Master Control Reset (MCR) zone.

TIP

A bit that is set within a subroutine using an OTE instruction remains set until the subroutine is scanned again.

Output Latch (OTL) and Output Unlatch (OTU)





Fixed	SLC 5/01	SLC 5/02	SLC 5/03	SLC 5/04	SLC 5/05
•	•	•	•	•	•

Output Instructions

OTL and OTU are retentive output instructions. OTL can only turn on a bit, while OTU can only turn off a bit. These instructions are usually used in pairs, with both instructions addressing the same bit.

Your program can examine a bit controlled by OTL and OTU instructions as often as necessary.

ATTENTION

Under fatal error conditions, physical outputs are turned off. Once the error conditions are cleared, the controller resumes operation using the data table value of the operand.

Using OTL

When you assign an address to the OTL instruction that corresponds to the address of a physical output, the output device wired to this screw terminal is energized when the bit is set (turned on or enabled).

When rung conditions become false (after being true), the bit remains set and the corresponding output device remains energized.

When enabled, the latch instruction tells the controller to turn on the addressed bit. Thereafter, the bit remains on, regardless of the rung condition, until the bit is turned off (typically by a OTU instruction in another rung).

Using OTU

When you assign an address to the OTU instruction that corresponds to the address of a physical output, the output device wired to this screw terminal is de-energized when the bit is cleared (turned off or disabled).

The unlatch instruction tells the controller to turn off the addressed bit. Thereafter, the bit remains off, regardless of the rung condition, until it is turned on (typically by a OTL instruction in another rung).

One-Shot Rising (OSR)

Fixed	SLC 5/01	SLC 5/02	SLC 5/03	SLC 5/04	SLC 5/05
•	•	•	•	•	•

Input Instruction

The OSR instruction is a retentive input instruction that triggers an event to occur one time. Use the OSR instruction when an event must start based on the change of state of the rung from false-to-true.

When the rung conditions preceding the OSR instruction go from false-to-true, the OSR instruction will be true for one scan. After one scan is complete, the OSR instruction becomes false, even if the rung conditions preceding it remain true. The OSR instruction will only become true again if the rung conditions preceding it transition from false-to-true.

The SLC 500 and SLC 5/01 processors allow you to use one OSR instruction per output in a rung; the OSR cannot be within a branch. The SLC 5/02 and higher processors allow you to use one OSR instruction per output in a rung; putting the OSR within a branch is permitted.

Entering Parameters

The address assigned to the OSR instruction is *not* the one-shot address referenced by your program, nor does it indicate the state of the OSR instruction. This address allows the OSR instruction to remember its previous rung state.

Use a bit address from either the bit or integer data file. The addressed bit is set (1) for one scan when rung conditions preceding the OSR instruction are true (even if the OSR instruction becomes false); the bit is reset (0) when rung conditions preceding the OSR instruction are false.

TIP

The bit address you use for this instruction must be unique. Do not use it elsewhere in the program.

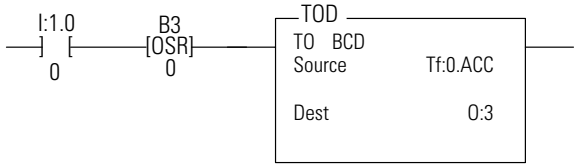
Do not use an input or output address to program the address parameter of the OSR instruction.

Examples

The following rungs illustrate the use of the OSR instruction. The first four rungs apply to SLC 500 and SLC 5/01 processors. The fifth rung involves output branching and applies to the SLC 5/02 and higher processors.

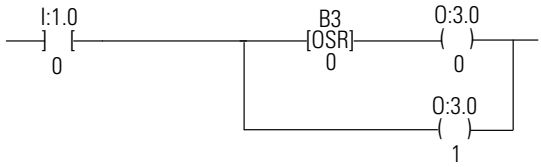
SLC 500 and SLC 5/01 Processors

When the input instruction goes from false-to-true, the OSR instruction conditions the rung so that the output goes true for one program scan. The output goes false and remains false for successive scans until the input makes another false-to-true transition.

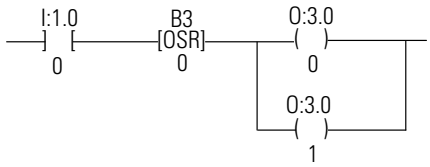


In this case, the accumulated value of a timer is converted to BCD and moved to an output word where an LED display is connected. When the timer is running, the accumulated value is changing rapidly. This value can be frozen and displayed for each false-to-true transition of the input condition of the rung.

Using an OSR Instruction in a Branch (SLC 500 and SLC 5/01 Processors)



In the above rung, the OSR instruction is not permitted inside a branch.



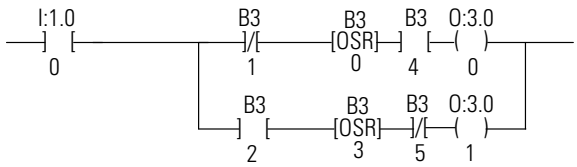
In this case, the OSR instruction is not in the branch so the rung is legal.

The SLC 500 and SLC 5/01 processors allow you to use only one OSR instruction per rung.

ATTENTION

When using a SLC 500 or SLC 5/01 processor, do not place input conditions after the OSR instruction in a rung. Unexpected operation may occur.

SLC 5/02 (and higher) Processors



The SLC 5/02 and higher processors allow you to use one OSR instruction per output in a rung. They also allow input conditions after the OSR instruction. Input branching around an OSR instruction is not allowed.

Timer Instructions Overview

Entering Parameters

These are several parameters associated with Timer instructions. The following paragraphs detail the operation of the timer instruction.

Accumulator Value (.ACC)

This is the time elapsed since the timer was last reset. When enabled, the timer updates this continually.

Preset Value (.PRE)

This specifies the value which the timer must reach before the controller sets the done bit. When the accumulated value becomes equal to or greater than the preset value, the done (DN) bit is set. You can use this bit to control an output device.

Preset and accumulated values for timers range from 0 to +32,767. If a timer preset or accumulated value is a negative number, a runtime error occurs.

Timebase

The timebase determines the duration of each timebase interval. For Fixed and SLC 5/01 processors, the timebase is set at 0.01 second.

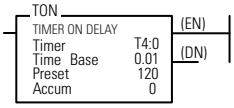
EXAMPLE If the timer base is set to 0.01, it would take 100 counts as the preset value (PRE) to equal 1 seconds worth of timing.

Timer Accuracy

Timer accuracy refers to the length of time between the moment a timer instruction is enabled and the moment the timed interval is complete. Inaccuracy caused by the program scan can be greater than the timer timebase. You must also consider the time required to energize the output device.

Timing accuracy is -0.01 to +0 seconds, with a program scan of up to 2.5 seconds. The 1-second timer maintains accuracy with a program scan of up to 1.5 seconds. If your programs can exceed 1.5 or 2.5 seconds, repeat the timer instruction rung so that the rung is scanned within these limits.

Timer On-Delay (TON)



Fixed	SLC 5/01	SLC 5/02	SLC 5/03	SLC 5/04	SLC 5/05
•	•	•	•	•	•

Output Instruction

Use the TON instruction to turn an output on or off after the timer has been on for a preset time interval. The TON instruction begins to count timebase intervals when rung conditions become true. As long as rung conditions remain true, the timer adjusts its accumulated value (ACC) each evaluation until it reaches the preset value (PRE). The accumulated value is reset when rung conditions go false, regardless of whether the timer has timed out.

Using Status Bits

This Bit	Is Set When	And Remains Set Until One of the Following
Timer Done Bit DN (bit 13)	accumulated value is equal to or greater than the preset value	rung conditions go false
Timer Timing Bit TT (bit 14)	rung conditions are true and the accumulated value is less than the preset value	rung conditions go false or when the done bit is set
Timer Enable Bit EN (bit 15)	rung conditions are true	rung conditions go false

When the processor changes from the REM Run or REM Test mode to the REM Program mode or user power is lost while the instruction is timing but has not reached its preset value, the following occurs:

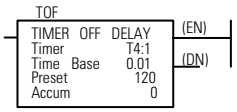
- Timer Enable (EN) bit remains set.
- Timer Timing (TT) bit remains set.
- Accumulated value (ACC) remains the same.

On returning to the REM Run or REM Test mode, the following can happen:

Condition	Result
If the rung is true:	EN bit remains set. TT bit remains set. ACC value is reset.
If the rung is false:	EN bit is reset. TT bit is reset. ACC value is reset.

Timer Off-Delay (TOF)

Use the TOF instruction to turn an output on or off after its rung has been off for a preset time interval. The TOF instruction begins to count timebase intervals when the rung makes a true-to-false transition. As long as rung conditions remain false, the timer increments its accumulated value (ACC) based on the timebase for each scan until it reaches the preset value (PRE). The accumulated value is reset when rung conditions go true regardless of whether the timer has timed out.



Fixed	SLC 5/01	SLC 5/02	SLC 5/03	SLC 5/04	SLC 5/05
•	•	•	•	•	•

Output Instruction

Using Status Bits

This Bit	Is Set When	And Remains Set Until One of the Following
Timer Done Bit DN (Bit 13)	rung conditions are true	rung conditions go false and the accumulated value is greater than or equal to the preset value
Timer Timing Bit TT (Bit 14)	rung conditions are false and the accumulated value is less than the preset value	rung conditions go true or when the done bit is reset
Timer Enable Bit EN (Bit 15)	rung conditions are true	rung conditions go false

When processor operation changes from the REM Run or REM Test mode to the REM Program mode or user power is lost while a timer off-delay instruction is timing but has not reached its preset value, the following occurs:

- Timer Enable (EN) bit remains set.
- Timer Timing (TT) bit remains set.
- Timer Done (DN) bit remains set.
- Accumulated value (ACC) remains the same.

On returning to the REM Run or REM Test mode, the following can happen:

Condition	Result
If the rung is true:	TT bit is reset DN bit remains set EN bit is set ACC value is reset.
If the rung is false:	TT bit is reset DN bit is reset EN bit is reset ACC value is set equal to the preset value.

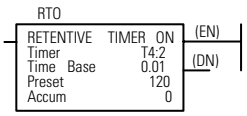
ATTENTION

The Reset (RES) instruction cannot be used with the TOF instruction because RES always clears the status bits as well as the accumulated value. (See 2-20)

TIP

The TOF timer times inside an inactive MCR Pair.

Retentive Timer (RTO)



Fixed	SLC 5/01	SLC 5/02	SLC 5/03	SLC 5/04	SLC 5/05
•	•	•	•	•	•

Output Instruction

Use the RTO instruction to turn an output on or off after its timer has been on for a preset time interval. The RTO instruction is a retentive instruction that begins to count timebase intervals when rung conditions become true.

The RTO instruction retains its accumulated value when any of the following occurs:

- Rung conditions become false.
- You change processor operation from the REM Run or REM Test mode to the REM Program mode.
- The processor loses power (provided that battery backup is maintained).
- A fault occurs.

When you return the processor to the REM Run or REM Test mode and/or rung conditions go true, timing continues from the retained accumulated value. By retaining its accumulated value, retentive timers measure the cumulative period during which rung conditions are true.

Using Status Bits

This Bit	Is Set When	And Remains Set Until One of the Following
Timer Done Bit DN (Bit 13)	accumulated value is equal to or greater than the preset value	the appropriate RES instruction is enabled
Timer Timing Bit TT (Bit 14)	rung conditions are true and the accumulated value is less than the preset value	Rung conditions go false or when the done bit is set
Timer Enable Bit EN (Bit 15)	rung conditions are true	rung conditions go false or if the timer is reset with the RES instruction

TIP

To reset the retentive timer's accumulated value and status bits after the RTO rung goes false, you must program a reset (RES) instruction with the same address in another rung.

When the processor changes from the REM Run or REM Test mode to the REM Program or REM Fault mode, or user power is lost while the timer is timing but not yet at the preset value, the following occurs:

- Timer Enable (EN) bit remains set.
- Timer Timing (TT) bit remains set.
- Accumulated value (ACC) remains the same.

On returning to the REM Run or REM Test mode or when power is restored, the following can happen:

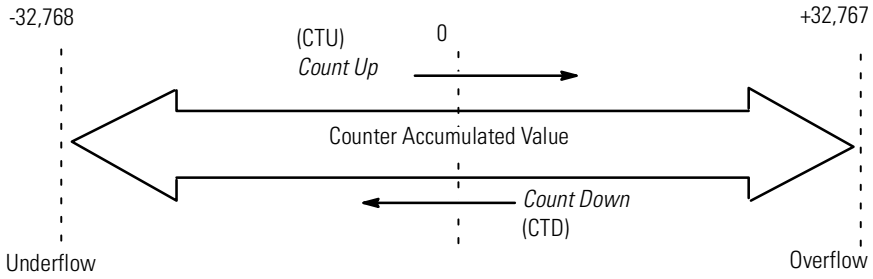
Condition	Results
If the rung is true:	TT bit remains set. EN bit remains set. ACC value remains the same and resumes incrementing.
If the rung is false:	TT bit is reset. DN bit remains in its last state. EN bit is reset. ACC value remains in its last state.

Counter Instructions Overview

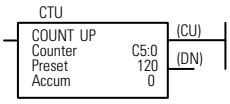
How Counters Work

The figure below demonstrates how a counter works. The count value must remain in the range of –32768 to –32767. If the count value goes above –32767 or below –32768, the counter status overflow (OV) or underflow (UN) bit is set.

A counter can be reset to zero using the reset (RES) instruction.



Count Up (CTU)



Fixed	SLC 5/01	SLC 5/02	SLC 5/03	SLC 5/04	SLC 5/05
•	•	•	•	•	•

Output Instruction

The CTU is an instruction that counts false-to-true rung transitions. Rung transitions can be caused by events occurring in the program (from internal logic or by external field devices) such as parts traveling past a detector or actuating a limit switch.

When rung conditions for a CTU instruction have made a false-to-true transition, the accumulated value is incremented by one count, provided that the rung containing the CTU instruction is evaluated between these transitions. The ability of the counter to detect false-to-true transitions depends on the speed (frequency) of the incoming signal.

TIP

The on and off duration of an incoming signal must not be faster than the scan time x2 (assuming a 50% duty cycle).

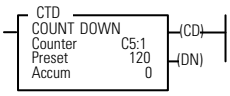
The accumulated value is retained when the rung conditions again become false. The accumulated count is retained until cleared by a reset (RES) instruction that has the same address as the counter reset.

Using Status Bits

This Bit	Is Set When	And Remains Set Until One of the Following
Count Up Overflow Bit OV (Bit 12)	accumulated value wraps around to -32,768 (from +32,767) and continues counting up from there	a RES instruction having the same address as the CTU instruction is executed OR the count is decremented less than or equal to +32,767 with a CTD instruction
Done Bit DN (Bit 13)	accumulated value is equal to or greater than the preset value	the accumulated value becomes less than the preset value
Count Up Enable Bit CU (Bit 15)	rung conditions are true	rung conditions go false OR a RES instruction having the same address as the CTU instruction is enabled

The accumulated value is retained after the CTU instruction goes false, or when power is removed from and then restored to the controller. Also, the on or off status of counter done, overflow, and underflow bits is retentive. The accumulated value and control bits are reset when the appropriate RES instruction is enabled. The CU bits are always set prior to entering the REM Run or REM Test modes.

Count Down (CTD)



Fixed	SLC 5/01	SLC 5/02	SLC 5/03	SLC 5/04	SLC 5/05
•	•	•	•	•	•

Output Instruction

The CTD is an instruction that counts false-to-true rung transitions. Rung transitions can be caused by events occurring in the program such as parts traveling past a detector or actuating a limit switch.

When rung conditions for a CTD instruction have made a false-to-true transition, the accumulated value is decremented by one count, provided that the rung containing the CTD instruction is evaluated between these transitions.

The accumulated counts are retained when the rung conditions again become false. The accumulated count is retained until cleared by a reset (RES) instruction that has the same address as the counter reset.

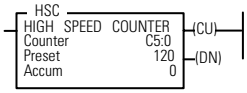
Using Status Bits

This Bit	Is Set When	And Remains Set Until One of the Following
Count Down Underflow Bit UN (Bit 11)	accumulated value wraps around to +32,767 (from -32,768) and continues counting down from there	a RES instruction having the same address as the CTD instruction is enabled. OR the count is incremented greater than or equal to +32,767 with a CTU instruction
Done Bit DN (Bit 13)	accumulated value is equal to or greater than the preset value	the accumulated value becomes less than the preset
Count Down Enable Bit CD (Bit 14)	rung conditions are true	rung conditions go false OR a RES instruction having the same address as the CTD instruction is enabled

The accumulated value is retained after the CTD instruction goes false, or when power is removed from and then restored to the controller. Also, the on or off status of counter done, overflow, and underflow bits is retentive. The accumulated value and control bits are reset when the appropriate RES instruction is executed. The CD bits are always set prior to entering the REM Run or REM Test modes.

High-Speed Counter (HSC)

The High-Speed Counter is a variation of the CTU counter. The HSC instruction is enabled when the rung logic is true and disabled when the rung logic is false.



Fixed	SLC 5/01	SLC 5/02	SLC 5/03	SLC 5/04	SLC 5/05
•					

Output Instruction

IMPORTANT Do not place the XIC instruction with address I:0/0 in series with the HSC instruction because counts will be lost.

The HSC instruction counts transitions that occur at input terminal I:0/0. The HSC instruction does not count rung transitions. You enable or disable the HSC rung to enable or disable the counting of transitions occurring at input terminal I:0/0. We recommend placing the HSC instruction in an unconditional rung.

The HSC is a special CTU counter for use with 24 VDC SLC fixed controllers. The HSC's status bits and accumulated values are non-retentive.

This instruction provides high-speed counting for fixed I/O controllers with 24 VDC inputs. One HSC instruction is allowed per controller. To

use the instruction, you must cut the jumper as shown below. A shielded cable is recommended to reduce noise to the input.

High-Speed Counter Data Elements

Address C5:0 is the HSC counter 3-word element.

Table 2.2 High Speed Counter Structure

15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00	Word
CU	CD	DN	OV	UN	UA				Not Used							0
Preset Value																1
Accumulator Value																2

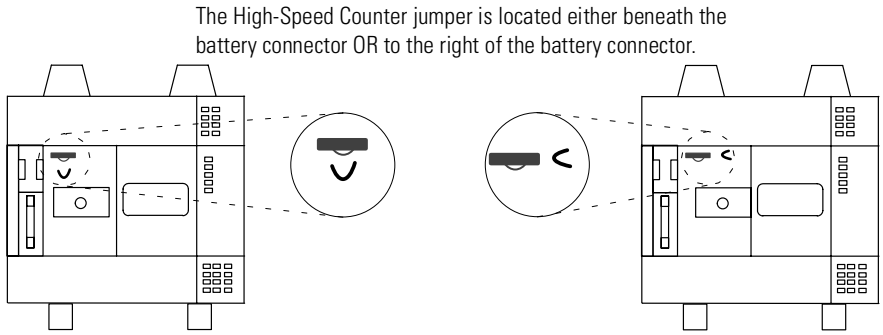
CU	Count up enable (Bit 15)
CD	Count down enable (Bit 14)
DN	Done bit (Bit 13)
OV	Overflow bit (Bit 12)
UN	Underflow bit (Bit 11)
UA	Update accumulator (HSC only) (Bit 10)

- Word 0 contains the following status bits of the HSC instruction:
  - Bit 10 (UA) updates the accumulator word of the HSC to reflect the immediate state of the HSC when true.
  - Bit 12 (OV) indicates if a HSC overflow has occurred.
  - Bit 13 (DN) indicates if the HSC preset value has been reached.
  - Bit 15 (CU) shows the Enable/Disable state of the HSC instruction.
- Word 1 contains the preset value that is loaded into the HSC when either the RES instruction is executed, when the Done bit is set, or when powerup takes place. The valid range is -1 to -32767.
- Word 2 contains the HSC accumulator value. This word is updated each time the HSC instruction is evaluated and when the update accumulator bit is set using an OTE instruction. This accumulator is read only. Any value written to the accumulator is overwritten by the actual high-speed counter on instruction evaluation, reset, or REM Run mode entry.

High-Speed Counter Operation

For high-speed counter operation you must do the following:

1. Turn off power to the fixed controller.
2. Remove the SLC 500 cover.
3. Locate and cut jumper wire J2. Do not remove completely but make certain that the ends of the cut jumper wire are not touching each other.



4. Replace the cover.

Input I:0/0 then operates in the high-speed mode. The address of the high-speed counter enable bit is C5:0/CU. When rung conditions are true, C5:0/CU is set and transitions occurring at input I:0/0 are counted.

To begin high-speed counting, load a preset value into C5:0.PRE and enable the counter rung. To load a preset value, do one of the following:

- Change to the REM Run or REM Test mode from another mode.
- Power up the processor in the REM Run mode.
- Reset the HSC using the RES instruction.

Automatic reloading occurs when the HSC itself sets the DN bit on interrupt.

Each input transition that occurs at input I:0/0 causes the HSC accumulated value to increment. When the accumulated value equals the preset value, the Done bit (C5:0/DN) is set, the accumulated value is cleared, and the preset value (C5:0.PRE) is loaded into the HSC in preparation for the next high-speed transition at input I:0/0.

Your ladder program should poll the Done bit (C5:0/DN) to determine the state of the HSC. Once the Done bit has been detected as set, the ladder program should clear bit C5:0/DN (using the unlatch OTU instruction) before the HSC accumulated again reaches the preset value, or the overflow bit (C5:0/OV) will be set.

The HSC differs from the CTU and CTD counters. The CTU and CTD are software counters. The HSC is a hardware counter and operates asynchronously to the ladder program scan. The HSC accumulated value (C5:0.ACC) is normally updated each time the HSC rung is evaluated in the ladder program. This means that the HSC hardware accumulator value is transferred to the HSC software accumulator. Only use the OTE instruction to transfer this value. The HSC instruction immediately clears bit C5:0/UA following the accumulated update.

Many HSC counts may occur between HSC evaluations, which would make C5:0.ACC inaccurate when used throughout a ladder program. To allow for an accurate HSC accumulated value, the update accumulator bit (C5:0/UA) causes C5:0.ACC to be immediately updated to the state of the hardware accumulator when set.

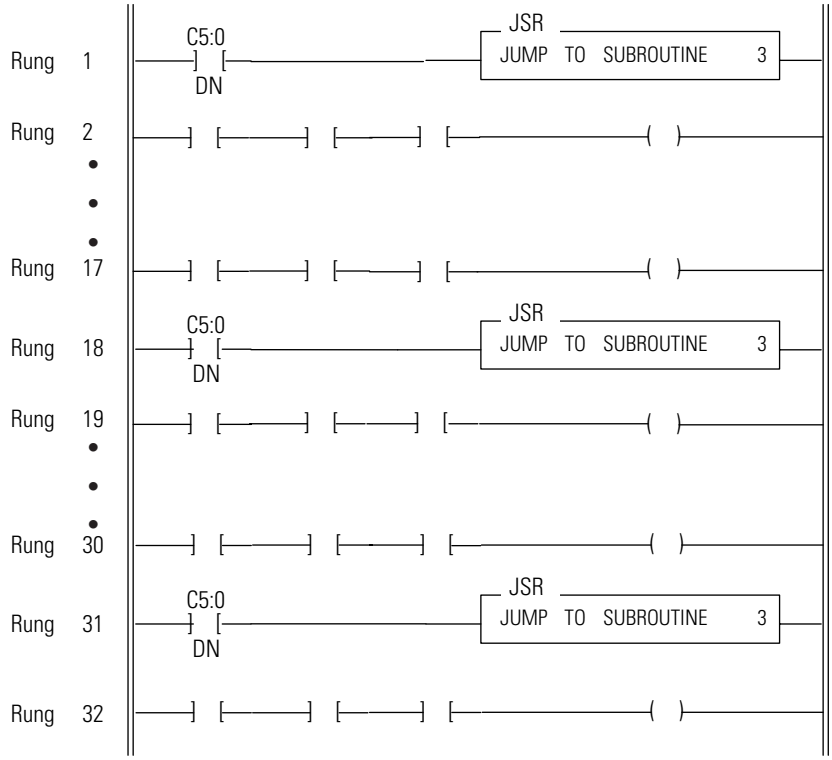
Use the RES instruction to reset the high-speed counter at address C5:0. The HSC instruction clears the status bits, the accumulator, and loads the preset value during:

- power up
- entry into the REM Run mode
- a reset

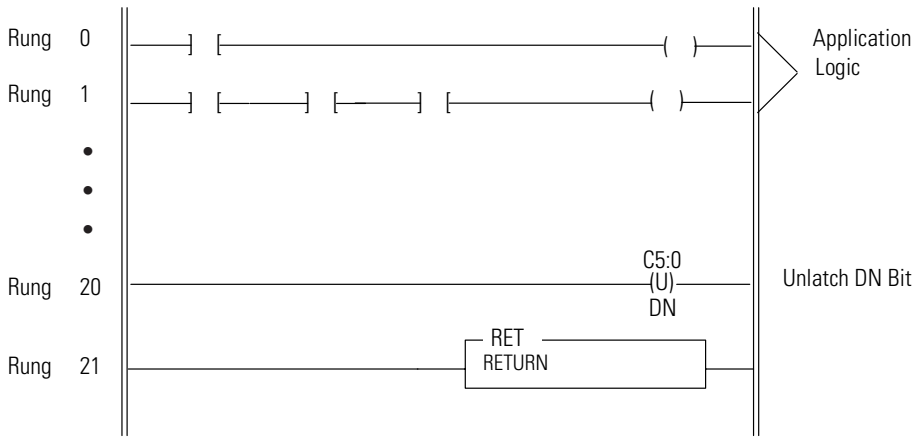
Application Example

In the following figures, rungs 1, 18, and 31 of the main program file each consist of an XIC instruction addressed to the HSC done bit and a JSR instruction. These rungs poll the status of the HSC done bit. When the Done bit is set at any of these poll points, program execution moves to subroutine file 3, executing the HSC logic. After the HSC logic is executed, the Done bit is reset by an unlatch instruction, and program execution returns to the main program file.

Application Example - File 2 (Poll for DN Bit in Main Program)



Application Example - File 3 (Execute HSC Logic)



Reset (RES)

Use a RES instruction to reset a timer or counter. When the RES instruction is enabled, it resets the Timer On Delay (TON), Retentive Timer (RTO), Count Up (CTU), or Count Down (CTD) instruction having the same address as the RES instruction.

—(RES)—

Fixed	SLC 5/01	SLC 5/02	SLC 5/03	SLC 5/04	SLC 5/05
•	•	•	•	•	•

Output Instruction

Using a RES instruction for a:	The processor resets the:
Timer (Do not use a RES instruction with a TOF.)	ACC value to 0 DN bit TT bit EN bit
Counter	ACC value to 0 OV bit UN bit DN bit CU bit CD bit
Control	POS value to 0 EN bit EU bit DN bit EM bit ER bit UL bit IN and FD go to last state

When resetting a counter, if the RES instruction is enabled and the counter rung is enabled, the CU or CD bit is reset.

If the counter preset value is negative, the RES instruction sets the accumulated value to zero. This in turn causes the done bit to be set by a count down or count up instruction.

ATTENTION

Because the RES instruction resets the accumulated value, and the done, timing, and enabled bits, do not use the RES instruction to reset a timer address used in a TOF instruction. Otherwise, unpredictable machine operation or injury to personnel may occur.

Comparison Instructions

This chapter contains general information about comparison instructions and explains how they function in your application program. Each of the comparison instructions includes information on:

- the instruction symbol
- instruction usage

Table 3.1 Comparison Instructions

Instruction Mnemonic	Instruction Name	Purpose	Page
EQU	Equal	Test whether two values are equal.	3-2
NEQ	Not Equal	Test whether one value is not equal to a second value.	3-2
LES	Less Than	Test whether one value is less than a second value.	3-3
LEQ	Less Than or Equal	Test whether one value is less than or equal to a second value.	3-3
GRT	Greater Than	Test whether one value is greater than another.	3-3
GEQ	Greater Than or Equal	Test whether one value is greater than or equal to a second value.	3-4
MEQ	Masked Comparison for Equal	Test portions of two values to see whether they are equal. Compares 16-bit data of a source address to 16-bit data at a reference address through a mask.	3-4
LIM	Limit Test	Test whether one value is within the limit range of two other values.	3-4

About the Comparison Instructions

Comparison instructions are used to test pairs of values to condition the logical continuity of a rung. As an example, suppose a LES instruction is presented with two values. If the first value is less than the second, then the comparison instruction is true.

To learn more about the compare instructions, we suggest that you read the Compare Instructions Overview that follows.

Comparison Instructions Overview

The following general information applies to comparison instructions.

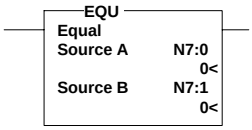
Using Indexed Word Addresses

When using comparison instructions, you have the option of using indexed word addresses for instruction parameters specifying word addresses. Indexed addressing is discussed in Appendix E of this manual.

Using Indirect Word Addresses

You have the option of using indirect word-level and bit-level addresses for instructions specifying word addresses when using an SLC 5/03 OS302, SLC 5/04 OS401, or SLC 5/05 processors. See Appendix E for more information.

Equal (EQU)



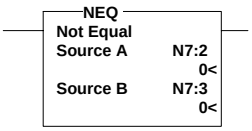
Fixed	SLC 5/01	SLC 5/02	SLC 5/03	SLC 5/04	SLC 5/05
•	•	•	•	•	•

Input Instruction

Use the EQU instruction to test whether two values are equal. If source A and source B are equal, the instruction is logically true. If these values are not equal, the instruction is logically false.

Source A must be an address. Source B can either be a program constant or a address. Negative integers are stored in two's complement form.

Not Equal (NEQ)



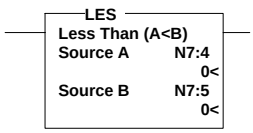
Fixed	SLC 5/01	SLC 5/02	SLC 5/03	SLC 5/04	SLC 5/05
•	•	•	•	•	•

Input Instruction

Use the NEQ instruction to test whether two values are not equal. If source A and source B are not equal, the instruction is logically true. If the two values are equal, the instruction is logically false.

Source A must be an address. Source B can be either a program constant or an address. Negative integers are stored in two's complement form.

Less Than (LES)



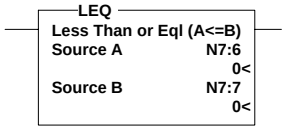
Fixed	SLC 5/01	SLC 5/02	SLC 5/03	SLC 5/04	SLC 5/05
•	•	•	•	•	•

Input Instruction

Use the LES instruction to test whether one value (source A) is less than another (source B). If source A is less than the value at source B, the instruction is logically true. If the value at source A is greater than or equal to the value at source B, the instruction is logically false.

Source A must be an address. Source B can either be a program constant or an address. Negative integers are stored in two's complement form.

Less Than or Equal (LEQ)



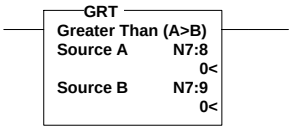
Fixed	SLC 5/01	SLC 5/02	SLC 5/03	SLC 5/04	SLC 5/05
•	•	•	•	•	•

Input Instruction

Use the LEQ instruction to test whether one value (source A) is less than or equal to another (source B). If the value at source A is less than or equal to the value at source B, the instruction is logically true. If the value at source A is greater than the value at source B, the instruction is logically false.

Source A must be an address. Source B can either be a program constant or an address. Negative integers are stored in two's complement form.

Greater Than (GRT)



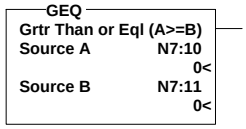
Fixed	SLC 5/01	SLC 5/02	SLC 5/03	SLC 5/04	SLC 5/05
•	•	•	•	•	•

Input Instruction

Use the GRT instruction to test whether one value (source A) is greater than another (source B). If the value at source A is greater than the value at source B, the instruction is logically true. If the value at source A is less than or equal to the value at source B, the instruction is logically false.

Source A must be an address. Source B can either be a program constant or an address. Negative integers are stored in two's complement form.

Greater Than or Equal (GEQ)



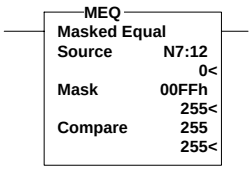
Fixed	SLC 5/01	SLC 5/02	SLC 5/03	SLC 5/04	SLC 5/05
•	•	•	•	•	•

Input Instruction

Use the GEQ instruction to test whether one value (source A) is greater than or equal to another (source B). If the value at source A is greater than or equal to the value at source B, the instruction is logically true. If the value at source A is less than the value at source B, the instruction is logically false.

Source A must be an address. Source B can either be a program constant or an address. Negative integers are stored in two's complement form.

Masked Comparison for Equal (MEQ)



Fixed	SLC 5/01	SLC 5/02	SLC 5/03	SLC 5/04	SLC 5/05
•	•	•	•	•	•

Input Instruction

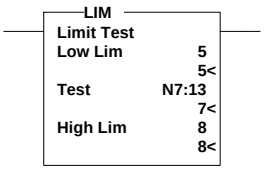
Use the MEQ instruction to compare data at a source address with data at a compare address. Use of this instruction allows portions of the data to be masked by a separate word.

Entering Parameters

- Source is the address of the value you want to compare.
- Mask is the address of the mask through which the instruction moves data. The mask can also be a hexadecimal value (constant).
- Compare is an integer value or the address of the reference.

If the 16 bits of data at the source address are equal to the 16 bits of data at the compare address (less masked bits), the instruction is true. The instruction becomes false as soon as it detects a mismatch. Bits in the mask word mask data when reset; they pass data when set.

Limit Test (LIM)



Fixed	SLC 5/01	SLC 5/02	SLC 5/03	SLC 5/04	SLC 5/05
		•	•	•	•

Input Instruction

Use the LIM instruction to test for values within or outside a specified range, depending on how you set the limits.

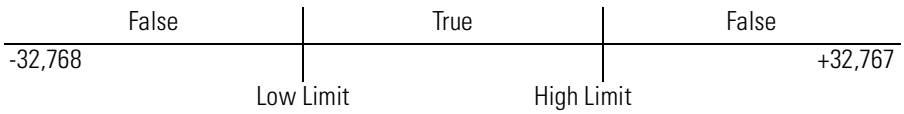
Entering Parameters

The Low Limit, Test, and High Limit values can be word addresses or constants, restricted to the following combinations:

- If the Test parameter is a program constant, both the Low Limit and High Limit parameters must be word addresses.
- If the Test parameter is a word address, the Low Limit and High Limit parameters can be either a program constant or a word address.

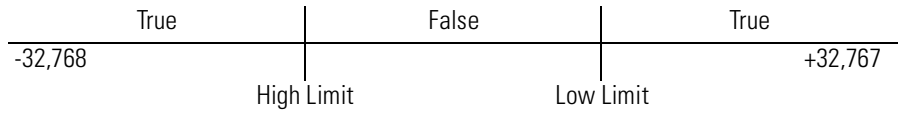
True/False Status of the Instruction

If the Low Limit has a value equal to or less than the High Limit, the instruction is true when the Test value is between the limits or is equal to either limit. If the Test value is outside the limits, the instruction is false, as shown below.



Example, low limit less than high limit:			
Low Limit	**High Limit**	**Instruction is True when Test value is**	**Instruction is False when Test value is**
5	8	5 through 8	-32,768 through 4 and 9 through 32,767

If the Low Limit has a value greater than the High Limit, the instruction is false when the Test value is between the limits. If the Test value is equal to either limit or outside the limits, the instruction is true, as shown below.



Example, low limit greater than high limit:			
Low Limit	**High Limit**	**Instruction is True when Test value is**	**Instruction is False when Test value is**
8	5	-32,768 through 5 and 8 through 32,767	6 and 7

Math Instructions

This chapter contains general information about math instructions and explains how they function in your logic program. Each of the math instructions includes information on:

- instruction symbol
- instruction usage

Table 4.1 Math Instructions

Instruction		Purpose	Page
Mnemonic	**Name**		
ADD	Add	Adds source A to source B and stores the result in the destination.	4-5
SUB	Subtract	Subtracts source B from source A and stores the result in the destination.	4-5
MUL	Multiply	Multiplies source A by source B and stores the result in the destination.	4-8
DIV	Divide	Divides source A by source B and stores the result in the destination and the math register.	4-9
DDV	Double Divide	Divides the contents of the math register by the source and stores the result in the destination and the math register.	4-11
CLR	Clear	Sets all bits of a word to zero.	4-12
SQR	Square Root	Calculates the square root of the source and places the integer result in the destination.	4-12
SCP	Scale with Parameters	Produces a scaled output value that has a linear relationship between the input and scaled values.	4-13
SCL	Scale Data	Multiplies the source by a specified rate, adds to an offset value, and stores the result in the destination.	4-15
RMP	Ramp	Provides the ability to create linear, acceleration, deceleration, and "S" curve ramp output data wave forms.	4-20
ABS	Absolute	Calculates the absolute value of the source and places the result in the destination.	4-24
CPT	Compute	Evaluates an expression and stores the result in the destination.	4-25
SWP	Swap	Swaps the low and high bytes of a specified number of words in a bit, integer, ASCII, or string file.	4-27
ASN	Arc Sine	Takes the arc sine of a number and stores the result (in radians) in the destination.	4-28
ACS	Arc Cosine	Takes the arc cosine of a number and stores the result (in radians) in the destination.	4-29
ATN	Arc Tangent	Takes the arc tangent of a number and stores the result (in radians) in the destination.	4-29
COS	Cosine	Takes the cosine of a number and stores the result in the destination.	4-30
LN	Natural Log	Takes the natural log of the value in the source and stores it in the destination.	4-30
LOG	Log to the Base 10	Takes the log base 10 of the value in the source and stores the result in the destination.	4-31
SIN	Sine	Takes the sine of a number and stores the result in the destination.	4-31
TAN	Tangent	Takes the tangent of a number and stores the result in the destination.	4-32
XPY	X to the Power of Y	Raise a value to a power and stores the result in the destination.	4-32

About the Math Instructions

The majority of the instructions take two input values, perform the specified arithmetic function, and output the result to an assigned memory location.

For example, both the ADD and SUB instructions take a pair of input values, add or subtract them, and place the result in the specified destination. If the result of the operation exceeds the allowable value, an overflow or underflow bit is set.

To learn more about the math instructions, we suggest that you read the Math Instructions Overview that follows.

Math Instructions Overview

The following general information applies to math instructions.

Entering Parameters

- Source is the address(es) of the value(s) on which the mathematical, logical, or move operation is to be performed. This can be word addresses or program constants. An instruction that has two source operands does not accept program constants in both operands.
- Destination is the address of the result of the operation. Signed integers are stored in two's complement form and apply to both source and destination parameters.

When using either an SLC 5/03 (OS301 and higher), SLC 5/04, or SLC 5/05 processor; floating point and string values (specified at the word level) are supported. Refer to Appendix D for additional valid addressing modes.

Using Indexed Word Addresses

You have the option of using indexed word addresses for instruction parameters specifying word addresses (except for fixed and SLC 5/01 processors). Indexed addressing is discussed in Appendix E.

Using Indirect Word Addresses

Refer to Appendix A for compatible firmware/processors. You have the option of using indirect word-level and bit-level addresses for instructions specifying word addresses when using an SLC 5/03

(OS302), SLC 5/04 (OS401), or SLC 5/05 processors.
Refer to Appendix C for more information.

Updates to Arithmetic Status Bits

The arithmetic status bits are found in Word 0, bits 0 to 3 in the controller status file. After an instruction is executed, the arithmetic status bits in the status file are updated:

With this Bit:		The Controller:
S:0/0	Carry (C)	sets if carry is generated; otherwise cleared.
S:0/1	Overflow (V)	indicates that the actual result of a math instruction does not fit in the designated destination.
S:0/2	Zero (Z)	indicates a 0 value after a math, move, or logic instruction.
S:0/3	Sign (S)	indicates a negative (less than 0) value after a math, move, or logic instruction.

Overflow Trap Bit, S:5/0

Minor error bit (S:5/0) is set upon detection of a mathematical overflow or division by zero. If this bit is set upon execution of an END statement, a Temporary End (TND) instruction, or an I/O Refresh (REF), the recoverable major error code 0020 is declared.

In applications where a math overflow or divide by zero occurs, you can avoid a CPU fault by using an unlatch (OTU) instruction with address S:5/0 in your program. The rung must be between the overflow point and the END, TND, or REF statement.

Updates to the Math Register, S:13 and S:14

Status word S:13 contains the *least* significant word of the 32-bit value of the MUL instruction. It contains the remainder for DIV and DDV instructions. It also contains the first four BCD digits for the Convert from BCD (FRD) and Convert to BCD (TOD) instructions.

Status word S:14 contains the *most* significant word of the 32-bit value of the MUL instruction. It contains the unrounded quotient for DIV and DDV instructions. It also contains the most significant digit (digit 5) for TOD and FRD instructions.

TIP When using floating point, S:13 and S:14 are not used.

Using Floating Point Data File (F8:)

This file type is valid for SLC 5/03 (OS301 and higher), SLC 5/04, and SLC 5/05 processors. These are 2-word elements and addressable only at the element level.

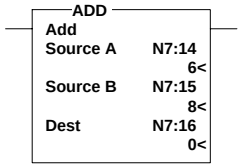
Assign floating point addresses as follows:

Table 4.2 Addressing Format

Format	Explanation		
Ff:e	**F**	Floating Point file	
	f	File number. Number 8 is the default file. A file number between 9- 255 can be used if additional storage is required.	
	:	Element delimiter	
	e	Element number	Ranges from 0- 255. These are 2-word elements. Non-extended 32-bit numbers
Examples:	**F8:2** **F10:36**	Element 2, floating point file 8 Element 36, floating point file 10 (file 10 designated as a floating point file by the user)	

Add (ADD)

Use the ADD instruction to add one value (source A) to another value (source B) and place the result in the destination.



Fixed	SLC 5/01	SLC 5/02	SLC 5/03	SLC 5/04	SLC 5/05
•	•	•	•	•	•

Output Instruction

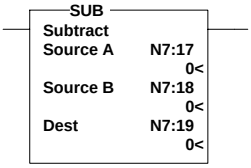
Updates to Arithmetic Status Bits

The arithmetic status bits are found in Word 0, bits 0 to 3 in the status file. After an instruction is executed, the arithmetic status bits in the status file are updated:

With this Bit:		The Processor:
S:0/0	Carry (C)	sets if carry is generated; otherwise resets (integer). For floating point, it is cleared.
S:0/1	Overflow (V)	sets if overflow is detected at destination; otherwise resets. On overflow, the minor error flag is also set. For floating point, the overflow value is placed in the destination. For an integer, the value -32,768 or 32,767 is placed in the destination. Exception: If you are using an SLC 5/02 or higher processor and have S:2/14 (math overflow selection bit) set, then the unsigned, truncated overflow remains in the destination.
S:0/2	Zero (Z)	sets if result is zero; otherwise resets.
S:0/3	Sign (S)	sets if result is negative; otherwise resets.

Subtract (SUB)

Use the SUB instruction to subtract one value (source B) from another (source A) and place the result in the destination.



Fixed	SLC 5/01	SLC 5/02	SLC 5/03	SLC 5/04	SLC 5/05
•	•	•	•	•	•

Output Instruction

Updates to Arithmetic Status Bits

The arithmetic status bits are found in Word 0, bits 0 to 3 in the status file. After an instruction is executed, the arithmetic status bits in the status file are updated:

With this Bit:		The Processor:
S:0/0	Carry (C)	sets if borrow is generated; otherwise resets (integer). For floating point it is cleared.
S:0/1	Overflow (V)	sets if underflow; otherwise reset. On underflow, the minor error flag is also set. For floating point, the overflow value is placed in the destination. For an integer, the value -32,768 or 32,767 is placed in the destination. Exception: If you are using an SLC 5/02 or higher processor and have S:2/14 (math overflow selection bit) set, then the unsigned, truncated overflow remains in the destination.
S:0/2	Zero (Z)	sets if result is zero; otherwise resets.
S:0/3	Sign (S)	sets if result is negative; otherwise resets.

32-Bit Addition and Subtraction

Fixed	SLC 5/01	SLC 5/02	SLC 5/03	SLC 5/04	SLC 5/05
		•	•	•	•

You have the option of performing 16-bit or 32-bit signed integer addition and subtraction. This is facilitated by status file bit S:2/14 (math overflow selection bit).

Math Overflow Selection Bit S:2/14

Set this bit when you intend to use 32-bit addition and subtraction. When S:2/14 is set, and the result of an ADD, SUB, MUL, DIV, or NEG instruction cannot be represented in the destination address (due to math underflow or overflow):

- The overflow bit S:0/1 is set.
- The overflow trap bit S:5/0 is set.
- The destination address contains the unsigned, truncated, least significant 16 bits of the result.

TIP

For MUL, DIV, integer, and all floating point instructions with an integer destination, when S:2/14 is set, the state change takes effect immediately.

When S:2/14 is reset (default condition), and the result of an ADD, SUB, MUL, DIV, or NEG instruction cannot be represented in the destination address (due to math underflow or overflow):

- The overflow bit S:0/1 is set.
- The overflow trap bit S:5/0 is set.
- The destination address contains 32767 if the result is positive or -32768 if the result is negative.

TIP

Additionally, the SLC 5/03 and higher processors only assert the state of bit S:2/14 at the end of scan for the ADD, SUB, and NEG instructions.

Note that the status of bit S:2/14 has no effect on the DDV instruction. Also, it has no effect on the math register content when using MUL and DIV instructions.

TIP

The SLC 5/03 and higher processors only interrogate the S:2/14 bit upon going to the Run mode and end-of-scan. Use the Data Monitor function to make this selection prior to entering the Run mode.

Example of 32-bit Addition

The following example shows how a 16-bit signed integer is added to a 32-bit signed integer. Remember that S:2/14 must be set for 32-bit addition.

Note that the value of the most significant 16 bits (B3:3) of the 32-bit number is increased by 1 if the carry bit S:0/0 is set and it is decreased by 1 if the number being added (B3:1) is negative.

TIP

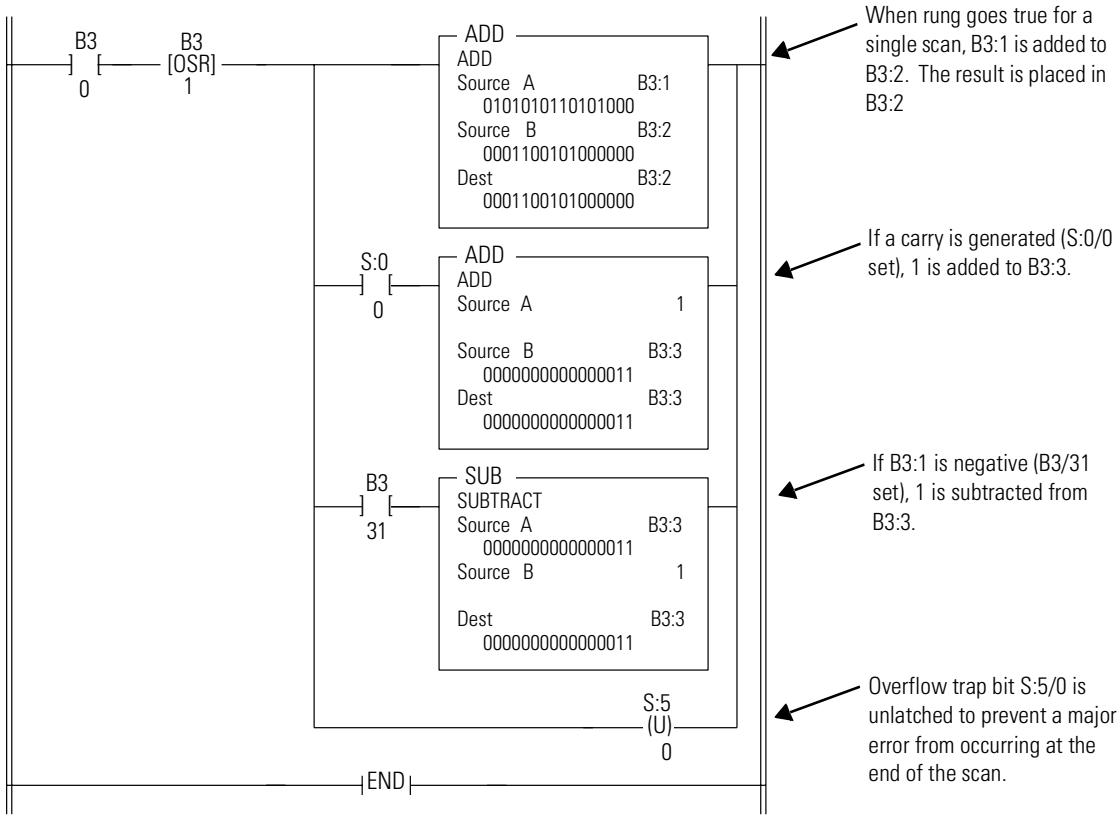
The largest possible number is 2,147,483,647 (7FFF FFFF)h.

To avoid a major error from occurring at the end of the scan, you must unlatch overflow trap bit S:5/0 as shown in the example ladder diagram to follow.

Table 4.3 Add 16-bit value B3:1 to 32-bit value B3:3 B3:2

Add Operation		Binary	Hex	Decimal(1)
Addend Addend	B3:3 B3:2 B3:1	0000 0000 0000 0011 0001 1001 0100 0000 0101 0101 1010 1000	0003 1940 55A8	203,072 21,928
Sum	B3:3 B3:2	0000 0000 0000 0011 0110 1110 1110 1000	0003 6EE8	225,000

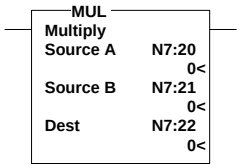
(1) The programming device displays 16-bit decimal values only. The decimal value of a 32-bit integer is derived from the displayed binary or hex value. For example, 0003 1940 Hex is 16^4x3 +16^3x1 + 16^2x9 + 16^1x4 + 16^0x0 = 203,072.



TIP You can use the rung above with a DDV instruction and a counter to find the average value of B3:1.

Multiply (MUL)

Use the MUL instruction to multiply one value (source A) by another (source B) and place the result in the destination.



Fixed	SLC 5/01	SLC 5/02	SLC 5/03	SLC 5/04	SLC 5/05
•	•	•	•	•	•

Output Instruction

Updates to Arithmetic Status Bits

The arithmetic status bits are found in Word 0, bits 0 to 3 in the status file. After an instruction is executed, the arithmetic status bits in the status file are updated:

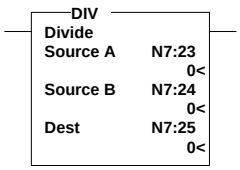
With this Bit:		The Processor:
S:0/0	Carry (C)	always resets.
S:0/1	Overflow (V)	sets if overflow is detected at destination; otherwise resets. On overflow, the minor error flag is also set. The value -32,768 or 32,767 is placed in the destination. Exception: If you are using an SLC 5/02 or higher processor and have S:2/14 (math overflow selection bit) set, then the unsigned, truncated least significant 16-bits of the result remains in the destination. For floating point destinations, the overflow result remains in the destination.
S:0/2	Zero (Z)	sets if result is zero; otherwise resets.
S:0/3	Sign (S)	sets if result is negative; otherwise resets.

Updates to the Math Register, S:13 and S:14

During integer operation, S:13 and S:14 contain the 32-bit signed result of the multiply instruction. This result is valid at overflow.

TIP For floating point operation, the math register does not change.

Divide (DIV)



Fixed	SLC 5/01	SLC 5/02	SLC 5/03	SLC 5/04	SLC 5/05
•	•	•	•	•	•

Output Instruction

Use the DIV instruction to divide one value (source A) by another (source B). The rounded quotient is then placed in the destination. If the remainder is 0.5 or greater, round up occurs in the destination. The unrounded quotient is stored in the most significant word of the math register. The remainder is placed in the least significant word of the math register.

Updates to Arithmetic Status Bits

The arithmetic status bits are found in Word 0, bits 0 to 3 in the status file. After an instruction is executed, the arithmetic status bits in the status file are updated:

With this Bit:		The Processor:
S:0/0	Carry (C)	always resets.
S:0/1	Overflow (V)	sets if division by zero or overflow is detected; otherwise resets. On overflow, the minor error flag is also set. The value 32,767 is placed in the destination. Exception: If you are using an SLC1 5/02 or higher processor and have S:2/14 (math overflow selection bit) set, then the unsigned, truncated overflow remains in the destination. For floating point destinations, the overflow result remains in the destination.
S:0/2	Zero (Z)	sets if result is zero; otherwise resets; undefined if overflow is set.
S:0/3	Sign (S)	sets if result is negative; otherwise resets; undefined if overflow is set.

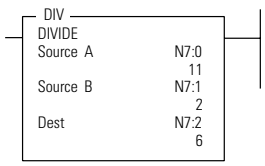
Updates to the Math Registers, S:13 and S:14

During integer operation, the unrounded quotient is placed in the most significant word (S:14), the remainder is placed in the least significant word (S:13).

TIP For floating point operation, the math register does not change.

Example

The remainder of 11/2 is 0.5, so the quotient is rounded up to 6 and is stored in the destination. The unrounded quotient, which is 5, is stored in S:14 and the remainder, which is 1, is stored at S:13.

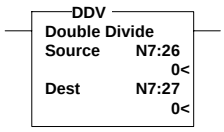


where: N7:0 = 11
N7:1 = 2
N7:2 = 6

result: S:14 = 5
S:13 = 1

Double Divide (DDV)

The 32-bit content of the math register is divided by the 16-bit source value and the rounded quotient is placed in the destination. If the remainder is 0.5 or greater, the destination is rounded up.



Fixed	SLC 5/01	SLC 5/02	SLC 5/03	SLC 5/04	SLC 5/05
•	•	•	•	•	•

Output Instruction

TIP This instruction typically follows a MUL instruction that creates a 32-bit result.

Updates to Arithmetic Status Bits

The arithmetic status bits are found in Word 0, bits 0 to 3 in the status file. After an instruction is executed, the arithmetic status bits in the status file are updated:

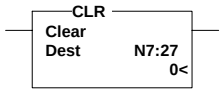
With this Bit:		The Processor:
S:0/0	Carry (C)	always resets.
S:0/1	Overflow (V)	sets if division by zero or if result is greater than 32,767 or less than -32,768; otherwise resets. On overflow, the minor error flag is also set. The value 32,767 is placed in the destination.
S:0/2	Zero (Z)	sets if result is zero; otherwise resets.
S:0/3	Sign (S)	sets if result is negative; otherwise resets; undefined if overflow is set.

Updates to the Math Registers, S:13 and S:14

Initially contains the dividend of the DDV operation. Upon instruction execution, the unrounded quotient is placed in the most significant word of the math register. The remainder is placed in the least significant word of the math register.

Clear (CLR)

Use the CLR instruction to set the destination value of a word to zero.



Fixed	SLC 5/01	SLC 5/02	SLC 5/03	SLC 5/04	SLC 5/05
•	•	•	•	•	•

Output Instruction

Updates to Arithmetic Status Bits

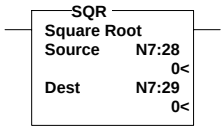
The arithmetic status bits are found in Word 0, bits 0 to 3 in the status file. After an instruction is executed, the arithmetic status bits in the status file are updated:

With this Bit:		The Processor:
S:0/0	Carry (C)	always resets.
S:0/1	Overflow (V)	always resets.
S:0/2	Zero (Z)	always sets.
S:0/3	Sign (S)	always resets.

Square Root (SQR)

When this instruction is evaluated as true, the square root of the absolute value of the source is calculated and the rounded result is placed in the destination.

The instruction calculates the square root of a negative number without overflow or faults. In applications where the source value may be negative, use a comparison instruction to evaluate the source value to determine if the destination may be invalid.



Fixed	SLC 5/01	SLC 5/02	SLC 5/03	SLC 5/04	SLC 5/05
		•	•	•	•

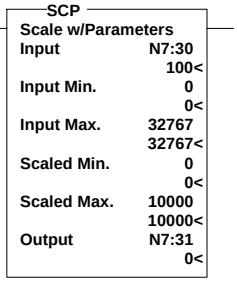
Output Instruction

Updates to Arithmetic Status Bits

The arithmetic status bits are found in Word 0, bits 0 to 3 in the status file. After an instruction is executed, the arithmetic status bits in the status file are updated:

With this Bit:		The Processor:
S:0/0	Carry (C)	is reserved (integer). For floating point, it is always cleared.
S:0/1	Overflow (V)	always resets.
S:0/2	Zero (Z)	sets when destination value is zero.
S:0/3	Sign (S)	always resets.

Scale with Parameters (SCP)



Fixed	SLC 5/01	SLC 5/02	SLC 5/03	SLC 5/04	SLC 5/05
			•	•	•

Output Instruction

Use the SCP instruction to produce a scaled output value that has a linear relationship between the input and scaled values. This instruction supports integer and floating point values. Use this instruction with SLC 5/03 (OS302), SLC 5/04 (OS401), and SLC 5/05 processors.

Use the following formula to convert analog input data to engineering units:

$$y = mx + b$$

Where:
y = scaled output
m = slope = (scaled MAX. - scaled MIN.) / (input MAX. - input MIN.)
x = input value
b = offset (y intercept) = scaled MIN - (input MIN. x m)

TIP

The **Input Minimum**, **Input Maximum**, **Scaled Minimum**, and **Scaled Maximum** are used to determine the slope and offset values. The input value can go outside of the specified input limits and no ordering is required. For example, the scaled output value is not necessarily clamped between the scaled minimum and scaled maximum values.

Entering Parameters

Enter the following parameters when programming this instruction:

- **Input value** can be a word address or an address of floating point data elements.
- **Input Minimum** and **Input Maximum** values determine the range of data that appears in the Input Value parameter. The value can be a word address, an integer constant, floating point data element, or a floating point constant.
- **Scaled Minimum** and **Scaled Maximum** values determine the range of data that appears in the Scaled Output parameter. The value can be a word address, an integer constant, floating point data element, or a floating point constant.
- **Scaled Output** value can be a word address or an address of floating point data elements.

Updates to Arithmetic Status Bits

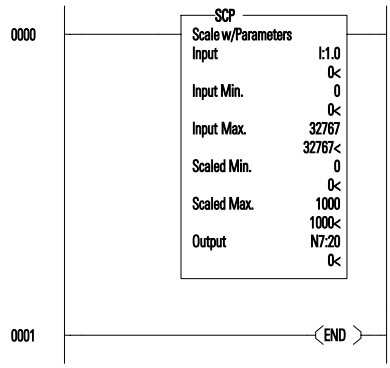
The arithmetic status bits are found in Word 0, bits 0 to 3 in the status file. After an instruction is executed, the arithmetic status bits in the status file are updated:

With this Bit:		The Processor:
S:0/0	Carry (C)	always resets.
S:0/1	Overflow (V)	sets if overflow generated or an unsupported input is detected; otherwise resets.
S:0/2	Zero (Z)	sets when destination value is zero; otherwise resets.
S:0/3	Sign (S)	sets if the destination value is negative; otherwise resets.

Application Examples

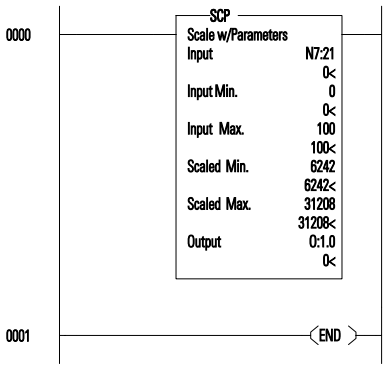
Example 1

In the first example, an analog I/O combination module (1746-NIO4I) is in slot 1 of the chassis. A pressure transducer is connected to input 0 and we want to read the value in engineering units. The pressure transducer measures pressures from 0 to 1000 psi and provides a 0 to 10V signal to the analog module. For a 0 to 10V signal, the analog module provides a range between 0 to 32,767. The following program rung places a number between 0 and 1000 into N7:20 based on the input signal coming from the pressure transducer into the analog module.



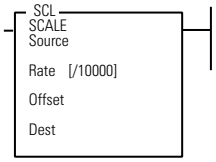
Example 2

In the second example, an analog I/O combination module (1746-NIO4I) is in slot 1 of the chassis. We want to control the proportional valve connected to output 0. The valve takes a 4 to 20 mA signal to control how far it opens (0 to 100%). (Assume that additional logic is present in the program that calculates how far to open the valve in percent and places a number between 0 and 100 into N7:21.) The analog module provides a 4 to 20mA output signal for a number between 6242 to 31,208. The following program rung directs an analog output to provide a 4 to 20 mA signal to the proportional valve (N7:21), based on a number between 0 and 100.



Scale Data (SCL)

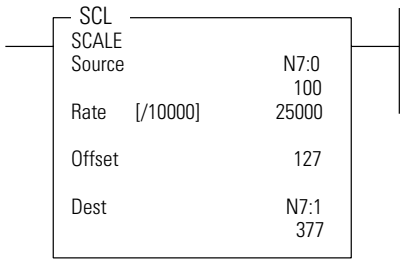
When this instruction is true, the value at the source address is multiplied by the rate value. The rounded result is added to the offset value and placed in the destination.



Fixed	SLC 5/01	SLC 5/02	SLC 5/03	SLC 5/04	SLC 5/05
		•	•	•	•

Output Instruction

Example



The source 100 is multiplied by 25000 and divided by 10000 and added to 127. The result, 377, is placed in the destination.

TIP

Anytime an underflow or overflow occurs in the destination file, minor error bit S:5/0 must be reset by the program. This must occur before the end of the current scan to prevent major error code 0020 from being declared. This instruction can overflow before the offset is added.

Note that the term rate is sometimes referred to as *slope*. The rate function is limited to the range -3.2768 to 3.2767. For example, -32768/10000 to +32767/10000.

Entering Parameters

The value for the following parameters is between -32,768 to 32,767.

- Source can be either a constant or a word address.
- Rate (or slope) is the positive or negative value you enter divided by 10,000. It can be either a constant or a word address.
- Offset can be either a constant or a word address.

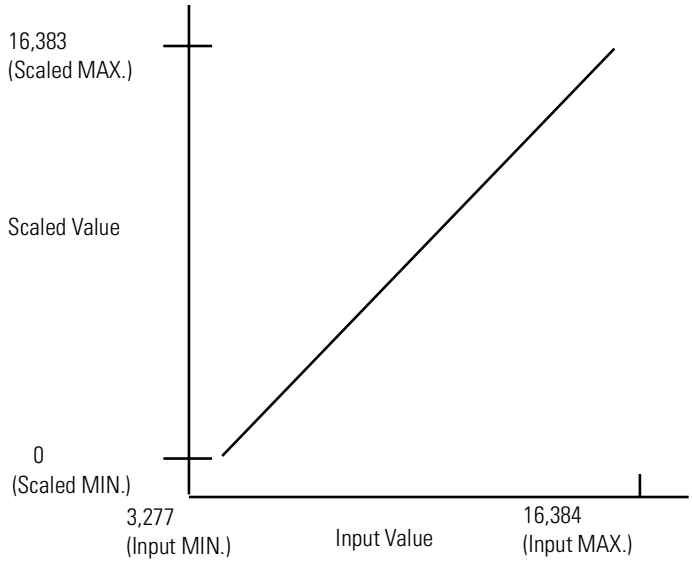
Updates to Arithmetic Status Bits

The arithmetic status bits are found in Word 0, bits 0 to 3 in the status file. After an instruction is executed, the arithmetic status bits in the status file are updated:

With this Bit:		The Processor:
S:0/0	Carry (C)	is reserved.
S:0/1	Overflow (V)	sets if an overflow is detected; otherwise resets. On overflow, minor error bit S:5/0 is also set and the value -32,768 or 32,767 is placed in the destination. The presence of an overflow is checked before and after the offset value is applied.(1)
S:0/2	Zero (Z)	sets when destination value is zero.
S:0/3	Sign (S)	sets if the destination value is negative; otherwise resets.

(1) If the result of the Source times the Rate, divided by 10000, is greater than 32767, the SCL instruction overflows, causing error 0020 (Minor Error Bit), and places 32767 in the Destination. This occurs regardless of the current offset.

Application Example 1 - Converting 4 to 20 mA Analog Input Signal to PID Process Variable



Calculating the Linear Relationship

Use the following equations to express the linear relationship between the input value and the resulting scaled value:

Scaled value = (Input Value X Rate) + Offset

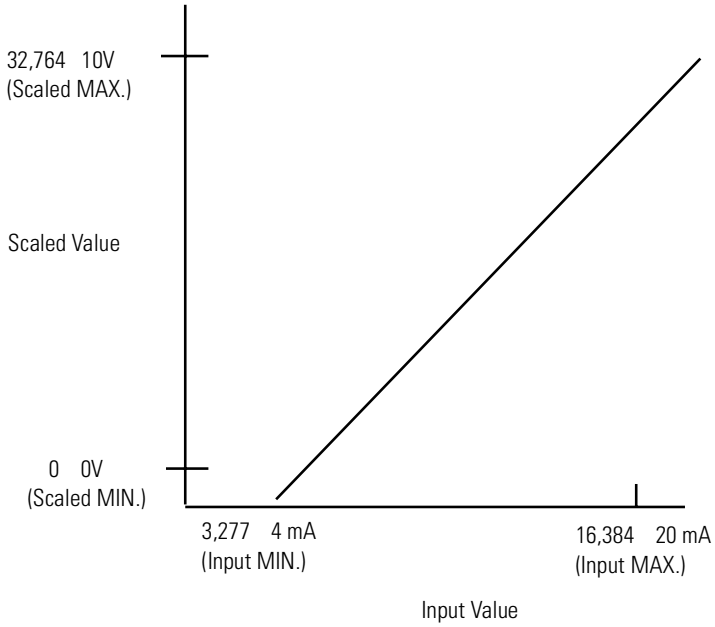
Rate = (Scaled MAX. - Scaled MIN.) / (Input MAX. - Input MIN.)

(16,383 - 0)/(16,384 - 3277) = 1.249 (or 12,490/10000)

Offset = Scaled MIN. - (Input MIN. X Rate)

0 - (3277 x 1.249) = -4093

Application Example 2 - Scaling an Analog Input to Control an Analog Output



Calculating the Linear Relationship

Use the following equations to calculate the scaled units:

Scaled value = (Input Value X Rate) + Offset

Rate = (Scaled MAX. - Scaled MIN.) / (Input MAX. - Input MIN.)

(32,764 - 0) / (16,384 - 3277) = 2.4997 (or 24,997/10000)

Offset = Scaled MIN. - (Input MIN. x rate)

0 - (3277 x 2.4997) = - 8192

The above offset and rate values are correct for the SCL instruction. However, if the input exceeds 13,107, the instruction overflows and sets S:5/0 math overflow bit. For example:

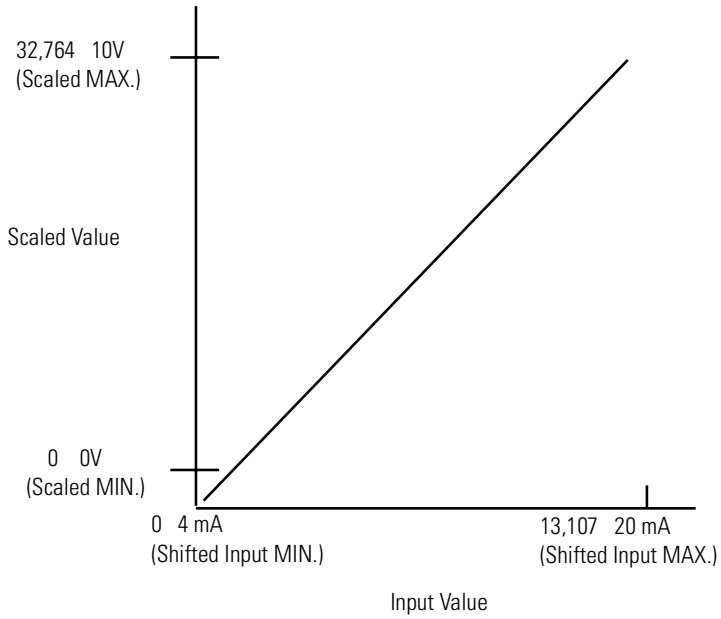
17 mA = 13,926 x 2.2997 = 34,810 (actual overflow)

34,810 - 8192 = 26,618

To avoid an overflow, we recommend shifting the linear relationship along the input value axis and reduce the values.

Notice that an overflow occurred even though the final value was correct. This happens because the overflow condition occurred during the rate calculation.

The following graph shows the shifted linear relationship. The input minimum value of 3,277 is subtracted from the input maximum value of 16,384, resulting in the value of 13,107.



Calculating the Shifted Linear Relationship

Use the following equations to calculate the scaled units:

$$\text{Scaled value} = (\text{Input Value} \times \text{Rate}) + \text{offset}$$

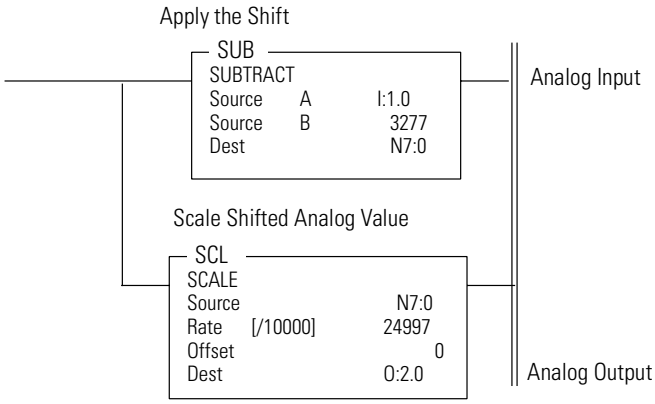
$$\text{Rate} = \frac{\text{Scaled MAX.} - \text{Scaled MIN.}}{\text{Input MAX.} - \text{Input MIN.}}$$

$$\frac{32,764 - 0}{13,107 - 0} = 2.4997 \text{ (or } 24,997/10000$$

$$\text{Offset} = \text{Scaled MIN.} - (\text{Input MIN.} \times \text{Rate})$$

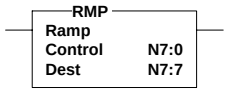
$$0 - (0 \times 2.4997) = 0$$

In this example, the SCL instruction is entered in the ladder logic program as follows:



Ramp Instruction (RMP)

mThe Ramp (RMP) instruction provides the ability to create linear, acceleration, deceleration, and "S" curve ramp output data wave forms. This instruction provides a means to ramp analog outputs when using them to control devices such as valves.



Fixed	SLC 5/01	SLC 5/02	SLC 5/03	SLC 5/04	SLC 5/05
			•	•	•

Output Instruction

The instruction has the following parameters:

- Control - Control Block is an integer file address that is user selectable. It is a 7-element file, which consists of the following bits and registers:
  - Word 0 Bit 15 - Enable bit, follows rung state of ramp instruction.
  - Word 0 Bit 14 - Ramping bit, when set, RMP function is working.
  - Word 0 Bit 13 - Done bit, set once the RMP function completes
    (current time = desired time).
  - Word 0 Bit 12 - Error bit, set if invalid parameters are specified.
  - Word 0 Bits 0 to 7 - Ramp Algorithm Type.
  - Word 1 Desired Time - This word defines the time duration of the ramp, in timebase units (1 second or 10 milliseconds). (integer value, valid range = +1 to +32767)
  - Word 2 Current Time - This word is the current time position of ramp, in timebase units (1 second or 10 milliseconds). The instruction updates the current time when the rungstate is true.
    (integer value, valid range = 0 to +32767)

- Word 3 Beginning Output Value - Starting point of ramp (integer value, valid range = -32768 to +32767).
- Word 4 Ending Output Value - Ending point of ramp (integer value, valid range = -32768 to +32767)
- Words 5 and 6 - These words are for internal use only.

• Destination - The Destination is any user defined integer word.

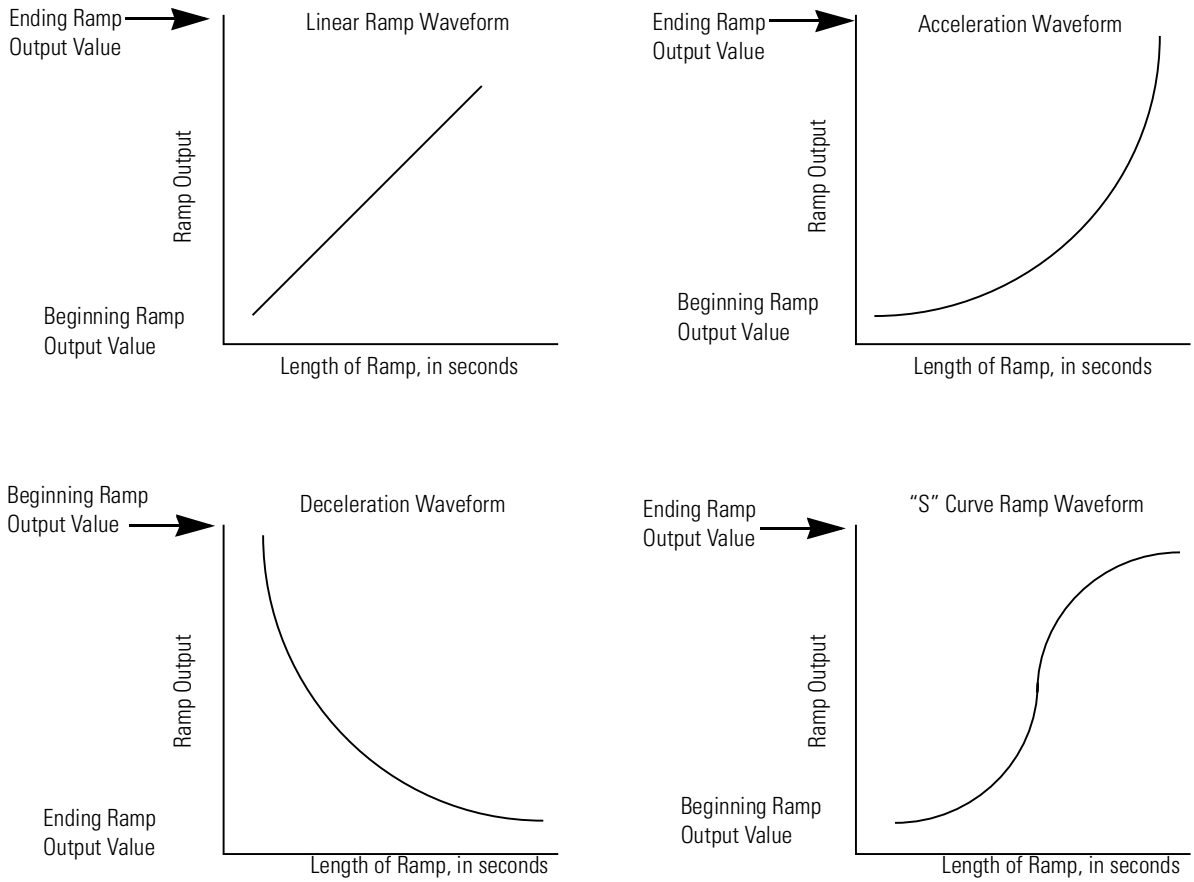
Table 4.4 Ramp Instruction Control Structure

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Word 0	EN	RMP	DN	ER					Ramp Algorithm Type							
									0	0	0	TB(1)	0	0	Waveform(2)	
Word 1	Desired Time															
Word 2	Current Time															
Word 3	Beginning Output Value															
Word 4	Ending Output Value															
Words 5 and 6	Internal use only															

(1) TB = 0, Timebase = 0.01 seconds
TB = 1, Timebase = 1.0 second

(2) Waveform = 00 Linear
Waveform = 01 Acceleration
Waveform = 10 Deceleration
Waveform = 11 "S" Curve

The following illustrations show Linear Ramp, Acceleration, Deceleration and "S" Curve Ramp waveforms.



Instruction Operation

When the rung state is true all parameters are validated to be in range. If the parameters are valid, the ramp function places the calculated output value in the destination register. The parameters are validated for every scan when the rung state is true. When the Ramp instruction is scanned and the rung state is true, the current time is updated, the destination value is calculated, and done condition is checked.

ATTENTION

Changing words 1 through 4 during instruction execution causes loss of resolution of one unit of measurement (1 second or 0.01 seconds depending on ramp). If these values are modified during execution the destination value automatically recalculates for the new value on the next scan.

ATTENTION

Modification of words 5 and 6 could result in unpredictable operation, possibly causing equipment damage and/or injury to personnel.

When the rung state is false, the current time is not updated and the destination value is not calculated. When the rung state sees another false to true change the current time is determined from the last updated position. An accuracy of +/- one unit (1 second or 0.01 seconds depending on ramp) or one scan, whichever is larger, can be expected for a false-to-true or true-to-false rung transition.

The Ramp instruction is retentive. If a Ramp instruction is executing when a power cycle occurs, the instruction continues to operate starting with the last updated position. The accuracy is limited to one unit or one scan, whichever is larger. Ramp instructions can be cascaded.

RMP Equation

The Ramp instruction is defined based on the following equations.

Linear: $Output = (E - S) \times \frac{P}{L} + S$

Acceleration: $Output = (E - S) \times \frac{P}{L} \times \frac{P}{L} + S$

Deceleration: $Output = (E - S) \times \left(1 - \frac{L - P}{L} \times \frac{L - P}{L}\right) + S$

"S" Curve:

$While \quad 0 \leq P \leq L / 2$

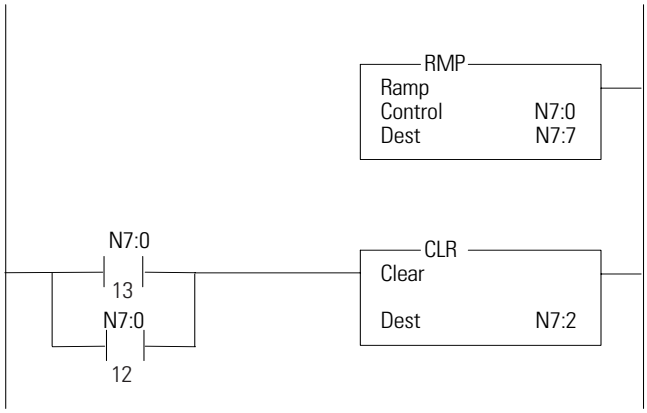
$Output = (E - S) \times 2 \times \frac{P}{L} \times \frac{P}{L} + S$

$While \quad (L / 2) < P < L$

$Output = (E - S) \times \left(1 - 2 \times \frac{L - P}{L} \times \frac{L - P}{L}\right) + S$

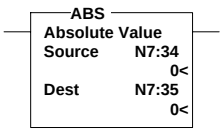
- S = the Beginning Output value
- E = the Ending Output value
- P = the Current Time
- L = the Overall Time
- Output = the RMP output value

Continuous Operation



Absolute (ABS)

Use the ABS instruction to calculate the absolute value of the Source and place the result in the Destination. This instruction supports integer and floating point values. Use this instruction with SLC 5/03 (OS302), SLC 5/04 (OS401), and SLC 5/05 processors.



Fixed	SLC 5/01	SLC 5/02	SLC 5/03	SLC 5/04	SLC 5/05
			•	•	•

Output Instruction

Entering Parameters

Enter the following parameters when programming this instruction:

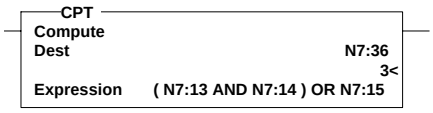
- Source can be a word address, an integer constant, floating point data element, or a floating point constant.
- Destination can only be a word address or a floating point data element.

Updates to Arithmetic Status Bits

The arithmetic status bits are found in Word 0, bits 0 to 3 in the status file. After an instruction is executed, the arithmetic status bits in the status file are updated:

With this Bit:		The Processor:
S:0/0	Carry (C)	always resets.
S:0/1	Overflow (V)	always resets with a floating point value; sets if the input is -32,768 (integer value).
S:0/2	Zero (Z)	sets when destination value is zero; otherwise resets.
S:0/3	Sign (S)	always resets.

Compute (CPT)



Fixed	SLC 5/01	SLC 5/02	SLC 5/03	SLC 5/04	SLC 5/05
			•	•	•

Output Instruction

The CPT instruction performs copy, arithmetic, logical, and conversion operations. You define the operation in the Expression and the result is written in the Destination. The CPT uses functions to operate on one or more values in the Expression to perform operations such as:

- converting from one number format to another
- manipulating numbers
- performing trigonometric functions

Use this instruction with SLC 5/03 (OS302), SLC 5/04 (OS401), and SLC 5/05 processors.

Instructions that can be used in the Expression include:
+, -, *, | (DIV), SQR, - (NEG), NOT, XOR, OR, AND, TOD, FRD, LN, TAN, ABS, DEG, RAD, SIN, COS, ATN, ASN, ACS, LOG, and ** (XPY).

TIP

The execution time of a CPT instruction is longer than a single arithmetic operation and uses more instruction words.

Entering Parameters

Enter the following parameters when programming this instruction:

- Destination can be a word address or the address of a floating-point data element.
- Expression is zero or more lines, with up to 28 characters per line, up to 255 characters.

Updates to Arithmetic Status Bits

The arithmetic status bits are found in Word 0, bits 0 to 3 in the status file. After an instruction is executed, the arithmetic status bits in the status file are updated:

With this Bit:		The Processor:
S:0/0	Carry (C)	sets based on the result of the last instruction in the Expression.
S:0/1	Overflow (V)	sets any time an overflow occurs during the evaluation of the Expression.
S:0/2	Zero (Z)	sets based on the result of the last instruction in the Expression.
S:0/3	Sign (S)	sets based on the result of the last instruction in the Expression.

The above bits are cleared at the start of the CPT instruction.Refer to Appendix B, status file bit S:34/2, for special handling of the math status bits when using floating point.

Application Example

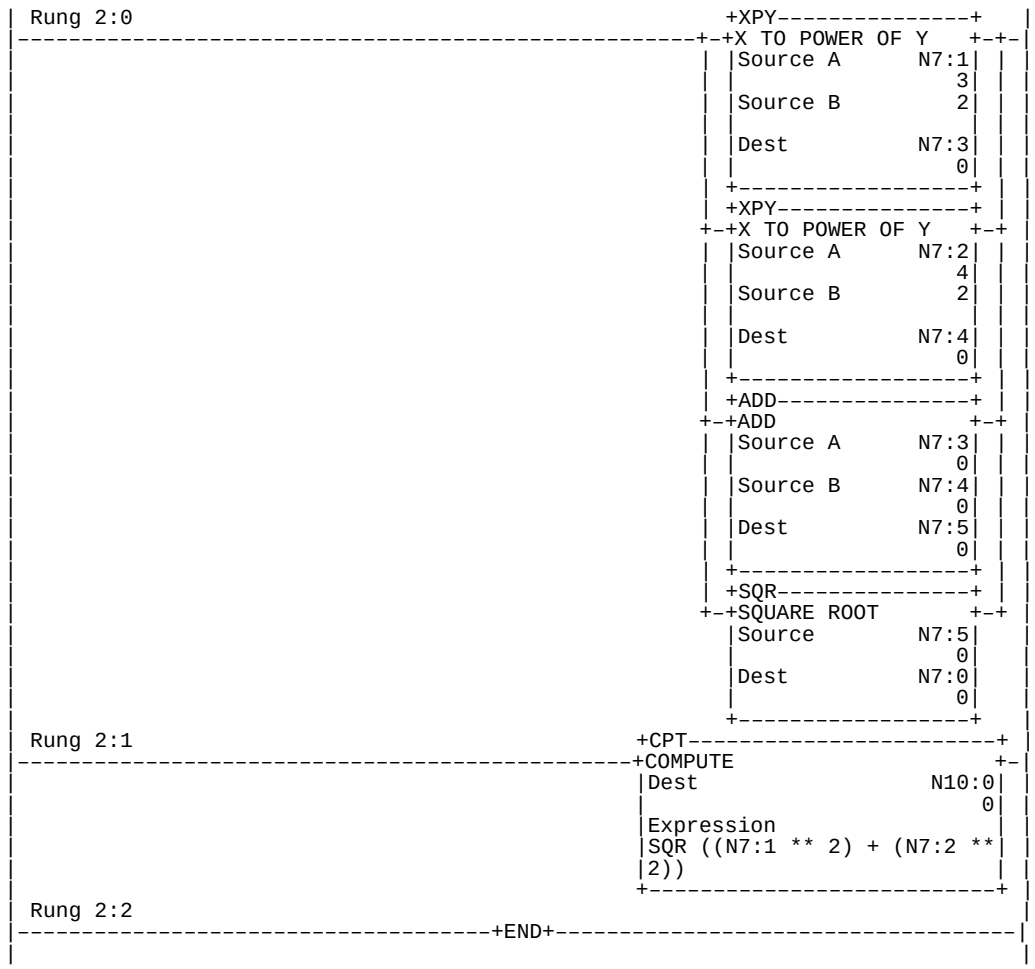
This application example uses Pythagorean's theorem to find the length of the long leg of a triangle, knowing the two other leg lengths. Use the following equation:

$$c^2 = a^2 + b^2$$

$$\text{where } c = \sqrt{(a^2 + b^2)}$$

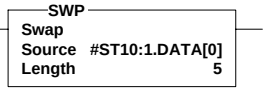
$$N10:0 = \sqrt{(N7:1)^2 + (N7:2)^2}$$

Rung 2:0 uses standard math instructions to implement Pythagorean's theorem. Rung 2:1 uses the CPT instruction to obtain the same calculation.



Swap (SWP)

Use this instruction to swap the low and high bytes of a specified number of words in a bit, integer, ASCII, or string file. Use this instruction with SLC 5/03 (OS302), SLC 5/04 (OS401), and SLC 5/05 processors.



Fixed	SLC 5/01	SLC 5/02	SLC 5/03	SLC 5/04	SLC 5/05
			•	•	•

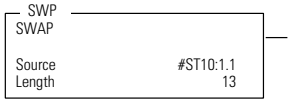
Output Instruction

Entering Parameters

Enter the following parameters when programming this instruction:

- Source can only be an indexed word address.
- Length refers to the number of words to be swapped, regardless of the file type. The address is limited to integer constants. For bit, integer, and ASCII file types, the length range is 1 to 128. For the string file type, the length range is 1 to 41. Note that this instruction is restricted to a single string element and cannot cross a string element boundary.

The following example shows how the SWP instruction works.

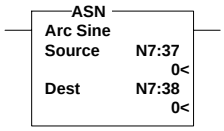


Before:
ST10:1 = abcdefghijklmnopqrstuvwxyz

After
ST10:1 = badcfehgjilknmporqtsvuxwzy

Arc Sine (ASN)

Use the ASN instruction to take the arc sine of a number and store the result (in radians) in the destination. The source must be greater than or equal to -1 and less than or equal to 1. The resulting value in the destination is always greater than or equal to -Pi/2 and less than or equal to Pi/2, where Pi = 3.141592. Use this instruction with SLC 5/03 (OS302), SLC 5/04 (OS401), and SLC 5/05 processors.



Fixed	SLC 5/01	SLC 5/02	SLC 5/03	SLC 5/04	SLC 5/05
			•	•	•

Output Instruction

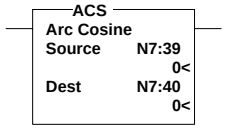
Updates to Arithmetic Status Bits

The arithmetic status bits are found in Word 0, bits 0 to 3 in the status file. After an instruction is executed, the arithmetic status bits in the status file are updated:

With this Bit:		The Processor:
S:0/0	Carry (C)	always resets.
S:0/1	Overflow (V)	sets if an overflow is generated or an unsupported input is detected; otherwise resets.
S:0/2	Zero (Z)	sets if the result is zero; otherwise resets.
S:0/3	Sign (S)	sets if the result is negative; otherwise resets.

Arc Cosine (ACS)

Use the ACS instruction to take the arc cosine of a number (source in radians) and store the result (in radians) in the destination. The source must be greater than or equal to -1 and less than or equal to 1. The resulting value in the destination is always greater than or equal to 0 and less than or equal to Pi, where Pi = 3.141592. Use this instruction with SLC 5/03 (OS302), SLC 5/04 (OS401), and SLC 5/05 processors.



Fixed	SLC 5/01	SLC 5/02	SLC 5/03	SLC 5/04	SLC 5/05
			•	•	•

Output Instruction

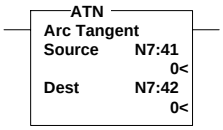
Updates to Arithmetic Status Bits

The arithmetic status bits are found in Word 0, bits 0 to 3 in the status file. After an instruction is executed, the arithmetic status bits in the status file are updated:

With this Bit:		The Processor:
S:0/0	Carry (C)	always resets.
S:0/1	Overflow (V)	sets if an overflow is generated or an unsupported input is detected; otherwise resets.
S:0/2	Zero (Z)	sets if the result is zero; otherwise resets.
S:0/3	Sign (S)	always resets.

Arc Tangent (ATN)

Use the ATN instruction to take the arc tangent of a number (source) and store the result (in radians) in the destination. The resulting value in the destination is always greater than or equal to -Pi/2 and less than or equal to Pi/2, where Pi = 3.141592. Use this instruction with SLC 5/03 (OS302), SLC 5/04 (OS401), and SLC 5/05 processors.



Fixed	SLC 5/01	SLC 5/02	SLC 5/03	SLC 5/04	SLC 5/05
			•	•	•

Output Instruction

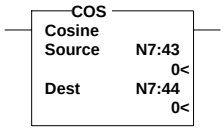
Updates to Arithmetic Status Bits

The arithmetic status bits are found in Word 0, bits 0 to 3 in the status file. After an instruction is executed, the arithmetic status bits in the status file are updated:

With this Bit:		The Processor:
S:0/0	Carry (C)	always resets.
S:0/1	Overflow (V)	sets if an overflow is generated or an unsupported input is detected; otherwise resets.
S:0/2	Zero (Z)	sets if the result is zero; otherwise resets.
S:0/3	Sign (S)	sets if the result is negative; otherwise resets.

Cosine (COS)

Use the COS instruction to take the cosine of a number (source in radians) and store the result in the destination. The source must be greater than or equal to -205887.4 and less than or equal to 205887.4. The greatest accuracy is achieved when the source is greater than -2 Pi and less than 2 Pi, where Pi = 3.141592. The resulting value in the destination is always greater than or equal to -1 and less than or equal to 1. Use this instruction with SLC 5/03 (OS302), SLC 5/04 (OS401), and SLC 5/05 processors.



Fixed	SLC 5/01	SLC 5/02	SLC 5/03	SLC 5/04	SLC 5/05
			•	•	•

Output Instruction

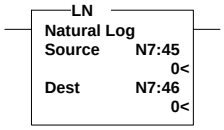
Updates to Arithmetic Status Bits

The arithmetic status bits are found in Word 0, bits 0 to 3 in the status file. After an instruction is executed, the arithmetic status bits in the status file are updated:

With this Bit:		The Processor:
S:0/0	Carry (C)	always resets.
S:0/1	Overflow (V)	sets if an overflow is generated or an unsupported input is detected; otherwise resets.
S:0/2	Zero (Z)	sets if the result is zero; otherwise resets.
S:0/3	Sign (S	sets if the result is negative; otherwise resets.

Natural Log (LN)

Use the LN instruction to take the natural log of the value in the source and store the result in the destination. The source must be greater than zero. The resulting value in the destination is always greater than or equal to -87.33654 and less than or equal to 88.72284. Use this instruction with SLC 5/03 (OS302), SLC 5/04 (OS401), and SLC 5/05 processors.



Fixed	SLC 5/01	SLC 5/02	SLC 5/03	SLC 5/04	SLC 5/05
			•	•	•

Output Instruction

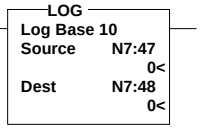
Updates to Arithmetic Status Bits

The arithmetic status bits are found in Word 0, bits 0 to 3 in the status file. After an instruction is executed, the arithmetic status bits in the status file are updated:

With this Bit:		The Processor:
S:0/0	Carry (C)	always resets.
S:0/1	Overflow (V)	sets if an overflow is generated or an unsupported input is detected; otherwise resets.
S:0/2	Zero (Z)	sets if the result is zero; otherwise resets.
S:0/3	Sign (S)	sets if the result is negative; otherwise resets.

Log to the Base 10 (LOG)

Use the LOG instruction to take the log base 10 of the value in the source and store the result in the destination. The source must be greater than zero. The resulting value in the destination is always greater than or equal to -37.92978 and less than or equal to 38.53184. Use this instruction with SLC 5/03 (OS302), SLC 5/04 (OS401), and SLC 5/05 processors.



Fixed	SLC 5/01	SLC 5/02	SLC 5/03	SLC 5/04	SLC 5/05
			•	•	•

Output Instruction

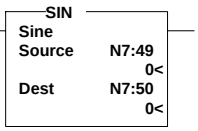
Updates to Arithmetic Status Bits

The arithmetic status bits are found in Word 0, bits 0 to 3 in the status file. After an instruction is executed, the arithmetic status bits in the status file are updated:

With this Bit:		The Processor:
S:0/0	Carry (C)	always resets.
S:0/1	Overflow (V)	sets if an overflow is generated or an unsupported input is detected; otherwise resets.
S:0/2	Zero (Z)	sets if the result is zero; otherwise resets.
S:0/3	Sign (S)	sets if the result is negative; otherwise resets.

Sine (SIN)

Use the SIN instruction to take the sine of a number (source in radians) and store the result in the destination. The source must be greater than or equal to -205887.4 and less than or equal to 205887.4. The greatest accuracy is achieved when the source is greater than -2 Pi and less than 2 Pi, where Pi = 3.141592. The resulting value in the destination is always greater than or equal to -1 and less than or equal to 1. Use this instruction with SLC 5/03 (OS302), SLC 5/04 (OS401), and SLC 5/05 processors.



Fixed	SLC 5/01	SLC 5/02	SLC 5/03	SLC 5/04	SLC 5/05
			•	•	•

Output Instruction

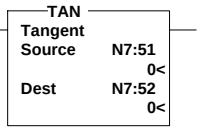
Updates to Arithmetic Status Bits

The arithmetic status bits are found in Word 0, bits 0 to 3 in the status file. After an instruction is executed, the arithmetic status bits in the status file are updated:

With this Bit:		The Processor:
S:0/0	Carry (C)	always resets.
S:0/1	Overflow (V)	sets if an overflow is generated or an unsupported input is detected; otherwise resets.
S:0/2	Zero (Z)	sets if the result is zero; otherwise resets.
S:0/3	Sign (S)	sets if the result is negative; otherwise resets.

Tangent (TAN)

Use the TAN instruction to take the tangent of a number (source in radians) and store the result in the destination. The value in the source must be greater than or equal to -102943.7 and less than or equal to 102943.7. The greatest accuracy is achieved when the source is greater than -2 Pi and less than 2 Pi, where Pi = 3.141592. The resulting value in the destination is either a real number or infinity. Use this instruction with SLC 5/03 (OS302), SLC 5/04 (OS401), and SLC 5/05 processors.



Fixed	SLC 5/01	SLC 5/02	SLC 5/03	SLC 5/04	SLC 5/05
			•	•	•

Output Instruction

Updates to Arithmetic Status Bits

The arithmetic status bits are found in Word 0, bits 0 to 3 in the status file. After an instruction is executed, the arithmetic status bits in the status file are updated:

With this Bit:		The Processor:
S:0/0	Carry (C)	always resets.
S:0/1	Overflow (V)	sets if an overflow is generated or an unsupported input is detected; otherwise resets.
S:0/2	Zero (Z)	sets if the result is zero; otherwise resets.
S:0/3	Sign (S)	sets if the result is negative; otherwise resets.

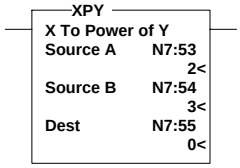
X to the Power of Y (XPY)

Use the XPY instruction to raise a value (source A) to a power (source B) and store the result in the destination. If the value in source A is negative, the exponent (source B) should be a whole number. If it is not a whole number, the overflow bit is set and the absolute value of the base is used in the calculation. Use this instruction with SLC 5/03 (OS302), SLC 5/04 (OS401), and SLC 5/05 processors.

The XPY instruction uses the following algorithm:

$$XPY = 2 ** (Y * \log_2 (X))$$

If any of the intermediate operations in this algorithm produce an overflow, the Arithmetic Overflow Status bit (S:0/1) is set.



Fixed	SLC 5/01	SLC 5/02	SLC 5/03	SLC 5/04	SLC 5/05
			•	•	•

Output Instruction

Updates to Arithmetic Status Bits

The arithmetic status bits are found in Word 0, bits 0 to 3 in the status file. After an instruction is executed, the arithmetic status bits in the status file are updated:

With this Bit:		The Processor:
S:0/0	Carry (C)	always resets.
S:0/1	Overflow (V)	sets if an overflow is generated or an unsupported input is detected; otherwise resets.
S:0/2	Zero (Z)	sets if the result is zero; otherwise resets.
S:0/3	Sign (S)	sets if the result is negative; otherwise resets.

Data Handling Instructions

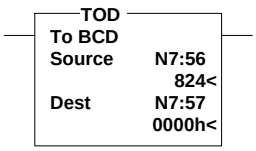
This chapter contains general information about the data handling instructions and explains how they function in your application program. Each of the instructions includes information on:

- what the instruction symbol looks like
- how to use the instruction

Table 5.1 Data Handling Instructions

Instruction		Purpose	Page
Mnemonic	**Name**		
TOD	Convert to BCD	Converts the integer source value to BCD format and stores it in the destination.	**5-2**
FRD	Convert from BCD	Converts the BCD source value to an integer and stores it in the destination.	**5-2**
DEG	Convert from Radians to Degrees	Converts radians (source) to degrees and stores the result in the destination.	**5-8**
RAD	Convert from Degrees to Radians	Converts degrees (source) to radians and stores the result in the destination.	**5-9**
DCD	Decode 4 to 1 of 16	Decodes a 4-bit value (0 to 15), turning on the corresponding bit in the 16-bit destination.	**5-10**
ENC	Encode 1 of 16 to 4	Encodes a 16-bit source to a 4-bit value. Searches the source from the lowest to the highest bit, and looks for the first set bit. The corresponding bit position is written to the destination as an integer.	**5-11**
COP and FLL	Copy File and Fill File	The COP instruction copies data from the source file to the destination file The FLL instruction loads a source value into each position in the destination file.	**5-12**
MOV	Move	Moves the source value to the destination.	**5-17**
MVM	Masked Move	Moves data from a source location to a selected portion of the destination.	**5-18**
AND	And	Performs a bitwise AND operation.	**5-20**
OR	Or	Performs a bitwise inclusive OR operation.	**5-21**
XOR	Exclusive Or	Performs a bitwise exclusive OR operation.	**5-22**
NOT	Not	Performs a NOT operation.	**5-23**
NEG	Negate	Changes the sign of the source and stores it in the destination.	**5-24**
FFL and FFU	FIFO Load and FIFO Unload	The FFL instruction loads a word into a FIFO stack on successive false-to-true transitions. The FFU unloads a word from the stack on successive false- to-true transitions. The first word loaded is the first to be unloaded.	**5-26**
LFL and LFU	LIFO Load and LIFO Unload	The LFL instruction loads a word into a LIFO stack on successive false-to-true transitions. The LFU unloads a word from the stack on successive false-to-true transitions. The last word loaded is the first to be unloaded.	**5-28**

Convert to BCD (TOD)



Fixed	SLC 5/01	SLC 5/02	SLC 5/03	SLC 5/04	SLC 5/05
•	•	•	•	•	•

Output Instruction

Use this instruction to convert 16-bit integers into BCD values.

With Fixed and SLC 5/01 processors, the destination can only be the math register. With SLC 5/02 and higher processors, the destination parameter can be a word address in any data file, or it can be the math register, S:13 and S:14.

If the integer value you enter is negative, the absolute value of the number is used for conversion.

Updates to Arithmetic Status Bits

The arithmetic status bits are found in Word 0, bits 0 to 3 in the status file. After an instruction is executed, the arithmetic status bits in the status file are updated:

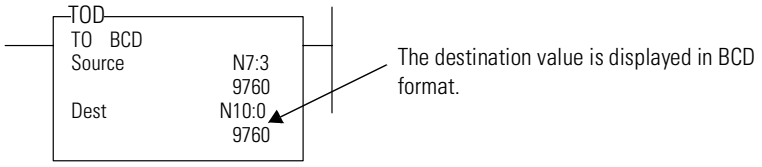
With this Bit:		The Processor:
S:0/0	Carry (C)	always resets.
S:0/1	Overflow (V)	sets if the BCD result is larger than 9999. Overflow results in a minor error.
S:0/2	Zero (Z)	sets if destination value is zero.
S:0/3	Sign (S)	sets if the source word is negative; otherwise resets.

Updates to the Math Register, S:13 and S:14

Contains the 5-digit BCD result of the conversion. This result is valid at overflow.

Example 1

The integer value 9760 stored at N7:3 is converted to BCD and the BCD equivalent is stored in N10:0. The maximum BCD value possible is 9999.



9	**7**	**6**	**0**	**N7:3**	**Decimal**	0010	0110	0010	0000
9	**7**	**6**	**0**	**N10:0**	**4-digit BCD**	1001	0111	0110	0000

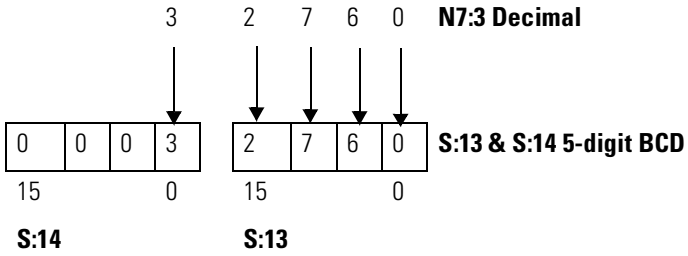
Example 2

The integer value 32760 stored at N7:3 is converted to BCD. The 5-digit BCD value is stored in the math register. The lower 4 digits of the BCD value is moved to output word O:2 and the remaining digit is moved through a mask to output word O:3.

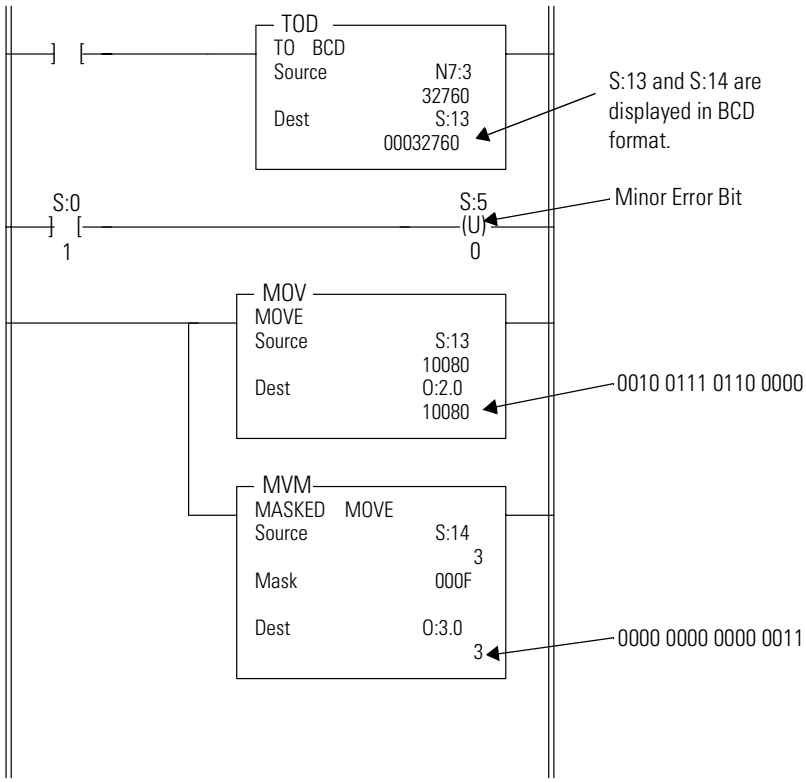
When using the math register as the destination parameter in the TOD instruction, the maximum BCD value possible is 32767.

TIP

However, for BCD values above 9999, the overflow bit is set, resulting in minor error bit S:5/0 also being set. Your ladder program can unlatch S:5/0 before the end of the scan to avoid major error 0020, as done in this example.

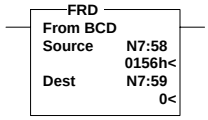


This example will output the absolute value (0 to 32767) contained in N7:3 as 5 BCD digits in output slots 2 and 3.



Convert from BCD (FRD)

Use this instruction to convert BCD values to integer values. With Fixed and SLC 5/01 processors, the source can only be the math register. With SLC 5/02 and higher processors, the source parameter can be a word address in any data file, or it can be the math register, S:13.



Fixed	SLC 5/01	SLC 5/02	SLC 5/03	SLC 5/04	SLC 5/05
•	•	•	•	•	•

Output Instruction

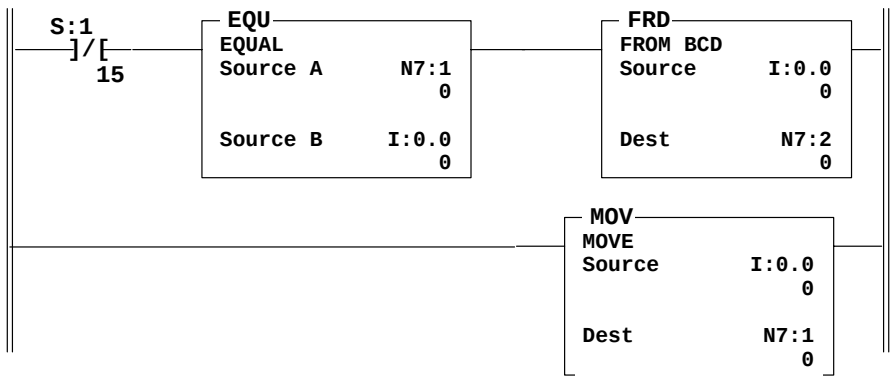
Updates to Arithmetic Status Bits

The arithmetic status bits are found in Word 0, bits 0 to 3 in the status file. After an instruction is executed, the arithmetic status bits in the status file are updated:

With this Bit:		The Processor:
S:0/0	Carry (C)	always resets.
S:0/1	Overflow (V)	sets if non-BCD value is contained at the source or the value to be converted is greater than 32,767; otherwise reset. Overflow results in a minor error.
S:0/2	Zero (Z)	sets if destination value is zero.
S:0/3	Sign (S)	always resets.

TIP

We recommend that you always provide ladder logic filtering of all BCD input devices prior to performing the FRD instruction. The slightest difference in point–to–point input filter delay can cause the FRD instruction to overflow due to the conversion of a non-BCD digit.



In the above example, the two rungs cause the processor to verify that the value at I:0.0 remains the same for two consecutive scans before it executes the FRD. This prevents the FRD from converting a non–BCD value during an input value change.

TIP

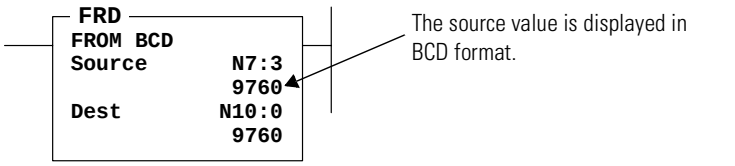
To convert numbers larger than 9999 BCD, the source must be the Math Register (S:13). You must reset the Minor Error bit (S:5/0) to prevent an error.

Changes to the Math Register, S:13 and S:14

Used as the source for converting the entire number range of a register.

Example 1

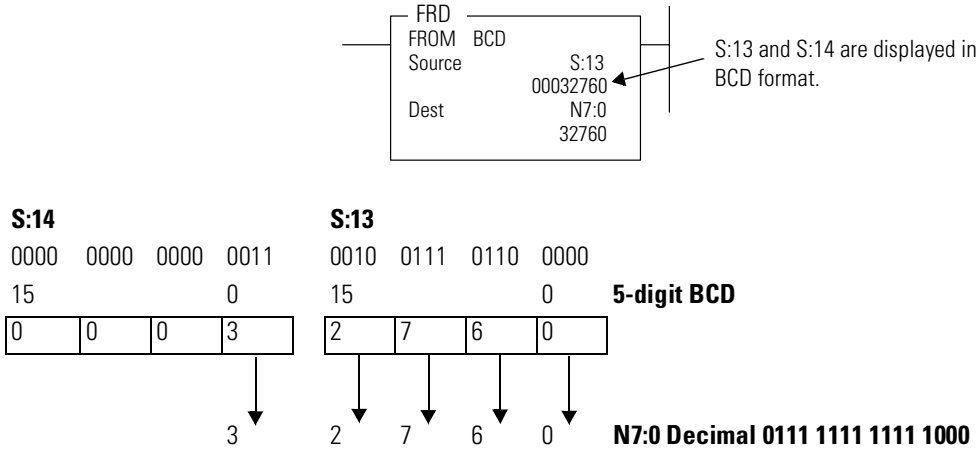
The BCD value 9760 at source N7:3 is converted and stored in N10:0. The maximum source value is 9999, BCD.



9	**7**	**6**	**0**	**N7:3**	**4-digit BCD**	1001	0111	0110	0000
9	**7**	**6**	**0**	**N10:0**	**Decimal**	0010	0110	0010	0000

Example 2

The BCD value 32760 in the math register is converted and stored in N7:0. The maximum source value is 32767, BCD.

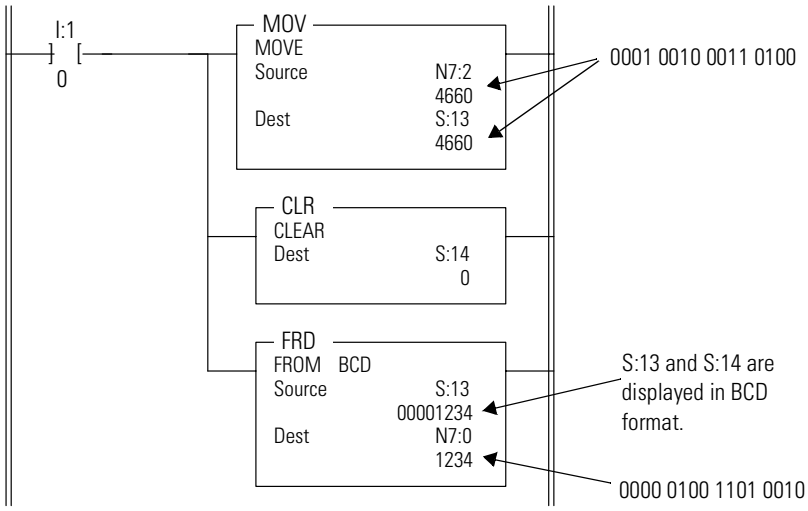


You should convert BCD values to integer before you manipulate them in your ladder program. If you do not convert the values, the processor manipulates them as integers and their value is lost.

TIP

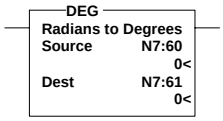
If the math register (S:13 and S:14) is used as the source for the FRD instruction and the BCD value does not exceed 4 digits, be sure to clear word S:14 before executing the FRD instruction. If S:14 is not cleared and a value is contained in this word from another math instruction located elsewhere in the program, an incorrect decimal value will be placed in the destination word.

Clearing S:14 before executing the FRD instruction is shown below:



When the input condition is set (1), a BCD value (transferred from a 4-digit thumb wheel switch for example) is moved from word N7:2 into the math register. Status word S:14 is then cleared to make certain that unwanted data is not present when the FRD instruction is executed.

Radian to Degrees (DEG)



Fixed	SLC 5/01	SLC 5/02	SLC 5/03	SLC 5/04	SLC 5/05
			•	•	•

Output Instruction

Use this instruction to convert radians (source) to degrees and store the result in the destination. The following formula applies:

Source * 180/Π
where Π = 3.141592

Use this instruction with SLC 5/03 (OS302), SLC 5/04 (OS401), and SLC 5/05 processors.

Example: Convert 3 radians to degrees. Using ratio where 180° = Π in radians, we have:

$$\frac{180}{\Pi} \longrightarrow \frac{x}{3} = \frac{180}{\Pi} \longrightarrow x = 3\left(\frac{180}{\Pi}\right) \longrightarrow x = 171.89 \text{ degrees}$$

Entering Parameters

- **Source** is the integer and/or floating point values.
- **Destination** is the address of the word where the data is to be stored.

Updates to Arithmetic Status Bits

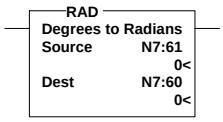
The arithmetic status bits are found in Word 0, bits 0 to 3 in the status file. After an instruction is executed, the arithmetic status bits in the status file are updated:

With this Bit:		The Processor:
S:0/0	Carry (C)	always resets.
S:0/1	Overflow (V)	sets if overflow generated or an unsupported input is detected; otherwise resets
S:0/2	Zero (Z)	sets if the result is zero; otherwise resets
S:0/3	Sign (S)	sets if the result is negative; otherwise resets

Degrees to Radians (RAD)

Use this instruction to convert degrees (source) to radians and store the result in the destination. The following formula applies:

Source * Π/ 180
where Π = 3.141592



Use this instruction with SLC 5/03 (OS302), SLC 5/04 (OS401), and SLC 5/05 processors.

Fixed	SLC 5/01	SLC 5/02	SLC 5/03	SLC 5/04	SLC 5/05
			•	•	•

Output Instruction

Example: Convert 135 degrees to radians.

$$\frac{\Pi}{180} \longrightarrow \frac{x}{135} = \frac{\Pi}{180} \longrightarrow x = 135\left(\frac{\Pi}{180}\right) \longrightarrow x = 2.356 \text{ Radians}$$

Entering Parameters

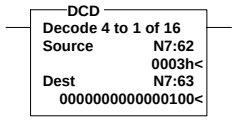
- **Source** is the integer and/or floating point values.
- **Destination** is the address of the word where the data is to be stored.

Updates to Arithmetic Status Bits

The arithmetic status bits are found in Word 0, bits 0 to 3 in the status file. After an instruction is executed, the arithmetic status bits in the status file are updated:

With this Bit:		The Processor:
S:0/0	Carry (C)	always resets.
S:0/1	Overflow (V)	sets if overflow generated or an unsupported input is detected; otherwise resets
S:0/2	Zero (Z)	sets if the result is zero; otherwise resets
S:0/3	Sign (S)	sets if the result is negative; otherwise resets

Decode 4 to 1 of 16 (DCD)



Fixed	SLC 5/01	SLC 5/02	SLC 5/03	SLC 5/04	SLC 5/05
•	•	•	•	•	•

Output Instruction

When executed, this instruction sets one bit of the destination word. The particular bit that is turned on depends on the value of the first four bits of the source word. See the table below.

Use this instruction to multiplex data in applications such as rotary switches, keypads, and bank switching.

Source						Destination															
Bit	15-04	03	02	01	00	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
	x	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
	x	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
	x	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
	x	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
	x	0	1	0	1	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
	x	0	1	1	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0
	x	0	1	1	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
	x	1	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
	x	1	0	0	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
	x	1	0	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
	x	1	0	1	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
	x	1	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
	x	1	1	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
	x	1	1	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	x	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

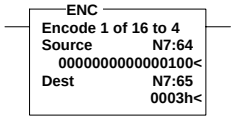
Entering Parameters

- Source is the address that contains the bit decode information. Only the first four bits (0 to 3) are used by the DCD instruction. The remaining bits may be used for other application specific needs. Change the value of the first four bits of this word to select one bit of the destination word.
- Destination is the address of the word where the data is to be stored.

Updates to Arithmetic Status Bits

Unaffected.

Encode 1 of 16 to 4 (ENC)



Fixed	SLC 5/01	SLC 5/02	SLC 5/03	SLC 5/04	SLC 5/05
			•	•	•

Output Instruction

When the rung is true, this output instruction searches the source from the lowest to the highest bit, and looks for the first set bit. The corresponding bit position is written to the destination as an integer as shown in the table below.

Use this instruction to multiplex data in applications such as rotary switches, keypads, and bank switching.

	Source																Destination				
Bit	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00	15 to 04	03	02	01	00
	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	1	x	0	0	0	0
	x	x	x	x	x	x	x	x	x	x	x	x	x	x	1	0	x	0	0	0	1
	x	x	x	x	x	x	x	x	x	x	x	x	x	1	0	0	x	0	0	1	0
	x	x	x	x	x	x	x	x	x	x	x	x	1	0	0	0	x	0	0	1	1
	x	x	x	x	x	x	x	x	x	x	x	1	0	0	0	0	x	0	1	0	0
	x	x	x	x	x	x	x	x	x	x	1	0	0	0	0	0	x	0	1	0	1
	x	x	x	x	x	x	x	x	x	1	0	0	0	0	0	0	x	0	1	1	0
	x	x	x	x	x	x	x	x	1	0	0	0	0	0	0	0	x	0	1	1	1
	x	x	x	x	x	x	x	1	0	0	0	0	0	0	0	0	x	1	0	0	0
	x	x	x	x	x	x	1	0	0	0	0	0	0	0	0	0	x	1	0	0	1
	x	x	x	x	x	1	0	0	0	0	0	0	0	0	0	0	x	1	0	1	0
	x	x	x	x	1	0	0	0	0	0	0	0	0	0	0	0	x	1	0	1	1
	x	x	x	1	0	0	0	0	0	0	0	0	0	0	0	0	x	1	1	0	0
	x	x	1	0	0	0	0	0	0	0	0	0	0	0	0	0	x	1	1	0	1
	x	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	x	1	1	1	0
	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	x	1	1	1	1

Entering Parameters

- Source is the address of the word to be encoded. Only one bit of this word should be on at any time. If more than one bit in the source is set, the destination bits are set based on the least significant bit that is set. If a source of zero is used, all of the destination bits are reset and the arithmetic status zero bit (S:0/2) is set.
- Destination is the address that contains the bit encode information. Bits 4 to 15 of the destination are reset by the ENC instruction.

Updates to Arithmetic Status Bits

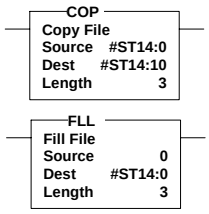
The arithmetic status bits are found in Word 0, bits 0 to 3 in the status file. After an instruction is executed, the arithmetic status bits in the status file are updated:

Table 5.2

With this Bit:		The Controller:
S:0/0	Carry (C)	always resets.
S:0/1	Overflow (V)	sets if more than one bit in the source is set; otherwise reset. The math overflow bit (s:5/0) is *not* set.
S:0/2	Zero (Z)	sets if destination value zero.
S:0/3	Sign (S)	always resets.

Copy File (COP) and Fill File (FLL) Instructions

The destination file type determines the number of words that an instruction transfers. For example, if the destination file type is a counter and the source file type is an integer, three integer words are transferred for each element in the counter-type file.



Fixed	SLC 5/01	SLC 5/02	SLC 5/03	SLC 5/04	SLC 5/05
•	•	•	•	•	•

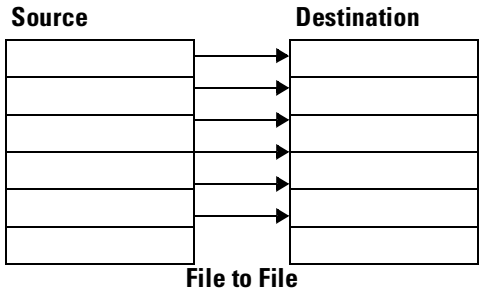
Output Instruction

After a COP or FLL instruction is executed, index register S:24 is cleared to zero.

Using COP

This instruction copies blocks of data from one location into another. It uses no status bits. If you need an enable bit, program an output instruction (OTE) in parallel using an internal bit as the output address. The following figure shows how file instruction data is manipulated.

.



Entering Parameters

Enter the following parameters when programming this instruction:

- Source is the address of the file you want to copy. You must use the file indicator (#) in the address.
- Destination is the starting address where the instruction stores the copy. You must use the file indicator (#) in the address.
- Length is the number of *elements* in the file you want to copy.
  - Maximum length is based on destination file type. If the destination file type is 3 words per element (Timer or Counter), you can specify a maximum length of 42. If the destination file type is 1 word per element, you can specify a maximum length of 128 words.

TIP

Refer to Appendix A for a listing of which processors support floating point and string values.

Refer to Appendix E for Indexed Addressing.

TIP The maximum lengths are based on destination file type.

All elements are copied from the source file into the destination file each time the instruction is executed. Elements are copied in ascending order.

If your destination file type is a timer, counter, or control file, be sure that the source words corresponding to the status words of your destination file contains zeros.

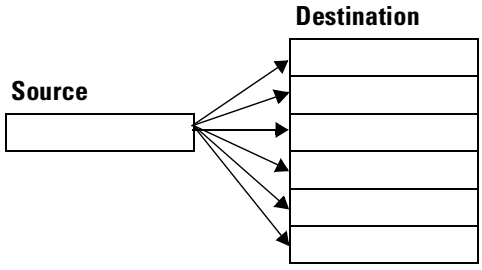
Be sure that you accurately specify the starting address and length of the data block you are copying. The instruction will not write over a file boundary (such as between files N16 and N17) at the destination. An error occurs if a write is attempted over a file boundary.

You can perform file shifts by specifying a source element address one or more elements greater than the destination element address within the same file. This shifts data to lower element addresses.

Using FLL

This instruction loads elements of a file with either a program constant or value from an element address.

The instruction fills the words of a file with a source value. It uses no status bits. If you need an enable bit, program a parallel output that uses a storage address. The following figure shows how file instruction data is manipulated.



Word to File

Entering Parameters

Enter the following parameters when programming this instruction:

- Source is the program constant or element address. The file indicator (#) is not required for an element address. When using either an SLC 5/03 (OS301 or higher), SLC 5/04 (OS401), or SLC 5/05 processor, floating point and string values are supported.
- Destination is the destination starting address of the file you want to fill. You must use the file indicator (#) in the address. When using either an SLC 5/03 (OS301 or higher), SLC 5/04 (OS401), or SLC 5/05 processor, floating point and string values are supported.
- Length is the number of *elements* in the file you want filled.
  - Maximum length is based on destination file type. If the destination file type is 3 words per element (Timer or Counter), you can specify a maximum length of 42. If the destination file type is 1 word per element, you can specify a maximum length of 128 words.

TIP

The maximum lengths are based on destination file type.

TIP

All elements are filled from the source value (typically a constant) into the specified destination file each scan the rung is true. Elements are filled in ascending order.

The instruction will not write over a file boundary (such as between files N16 and N17) at the destination. An error is declared if a write is attempted over a file boundary.

Move and Logical Instructions Overview

The following general information applies to move and logical instructions.

Entering Parameters

- Source is the address of the value on which the logical or move operation is to be performed. The source can be a word address or a program constant, unless otherwise described. If the instruction has two source operands, it does not accept program constants in both operands.

  When using either an SLC 5/03 (OS301 or higher), SLC 5/04 (OS401), or
  SLC 5/05 processor, floating point and string values are supported.

- Destination is the result address of a move or logical operation. It must be a word address.

Using Indexed Word Addresses

You have the option of using indexed word addresses for instruction parameters specifying word addresses. Refer to Specifying Indexed Addresses on page E-10 for more information.

Updates to Arithmetic Status Bits

The arithmetic status bits are found in Word 0, bits 0 to 3 in the controller status file. After an instruction is executed, the arithmetic status bits in the status file are updated.

Using Indirect Word Addresses

You have the option of using indirect word-level and bit-level addresses for instructions specifying word addresses when using an SLC 5/03 (OS302), SLC 5/04 (OS401), or SLC 5/05 processors. Refer to Specifying an Indirect Address on page E-14 for more information.

Updates to the Math Register, S:13 and S:14

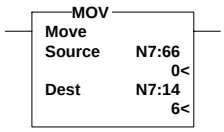
Move and logical instructions do not affect the math register.

Entering Mask Values

TIP

When entering constants, you can use “b” or “h” to change the radix of your entry. For example, instead of entering -1 as a constant, you could enter 1111111111111111b or FFFFh.

Move (MOV)



Fixed	SLC 5/01	SLC 5/02	SLC 5/03	SLC 5/04	SLC 5/05
•	•	•	•	•	•

Output Instruction

This output instruction moves the source value to the destination location. As long as the rung remains true, the instruction moves the data each scan.

Entering Parameters

Enter the following parameters when programming this instruction:

- Source is the address or constant of the data you want to move.
- Destination is the address where the instruction moves the data.

TIP

If you wish to move one word of data without affecting the arithmetic bits, use a copy (COP) instruction with a length of 1 word instead of the MOV instruction.

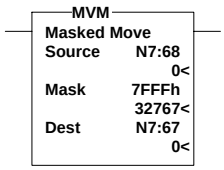
Updates to Arithmetic Status Bits

The arithmetic status bits are found in Word 0, bits 0 to 3 in the controller status file. After an instruction is executed, the arithmetic status bits in the status file are updated.

With this Bit:		The Controller:
S:0/0	Carry (C)	always resets.
S:0/1	Overflow (V)	always resets.
S:0/2	Zero (Z)	sets if result is zero; otherwise resets.
S:0/3	Sign (S)	sets if result is negative (most significant bit is set); otherwise resets.

Masked Move (MVM)

The MVM instruction is a word instruction that moves data from a source location to a destination, and allows portions of the destination data to be masked by a separate word. As long as the rung remains true, the instruction moves the data each scan.



Fixed	SLC 5/01	SLC 5/02	SLC 5/03	SLC 5/04	SLC 5/05
•	•	•	•	•	•

Output Instruction

Entering Parameters

Enter the following parameters when programming this instruction:

- **Source** is the address of the data you want to move.
- **Mask** is the address of the mask through which the instruction moves data; the mask can also be a hexadecimal value (constant).
- **Destination** is the address where the instruction moves the data.

Updates to Arithmetic Status Bits

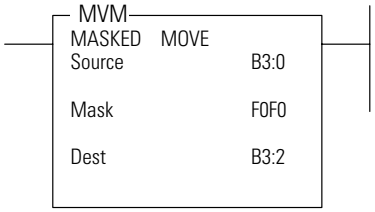
The arithmetic status bits are found in Word 0, bits 0 to 3 in the controller status file. After an instruction is executed, the arithmetic status bits in the status file are updated.

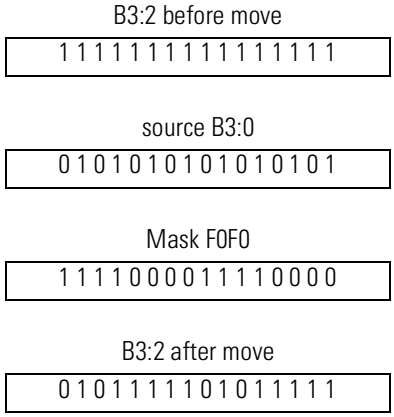
Table 5.3

With this Bit:		The Controller:
S:0/0	Carry (C)	always resets.
S:0/1	Overflow (V)	always resets.
S:0/2	Zero (Z)	sets if result is zero; otherwise resets.
S:0/3	Sign (S)	sets if result is negative; otherwise resets.

Operation

When the rung containing this instruction is true, data at the source address passes through the mask to the destination address. See the figure below.





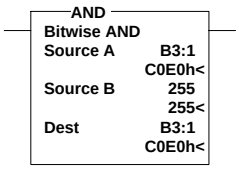
Mask data by resetting bits in the mask; pass data by setting bits in the mask to one. The bits of the mask can be fixed by a constant value, or you can vary them by assigning the mask a direct address.

TIP

Bits in the destination that correspond to zeros in the mask are not altered.

And (AND)

This instruction performs a bit-by-bit logical AND. The operation is performed using the value at source A and the value at source B. The result is stored in the destination.



Fixed	SLC 5/01	SLC 5/02	SLC 5/03	SLC 5/04	SLC 5/05
•	•	•	•	•	•

Output Instruction

Table 5.4 Truth Table for A AND B = Dest

A	B	Dest
0	0	0
1	0	0
0	1	0
1	1	1

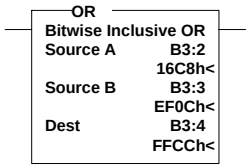
Source A and B can either be a word address or a constant; however, both sources cannot be a constant. The destination must be a word address.

Updates to Arithmetic Status Bits

The arithmetic status bits are found in Word 0, bits 0 to 3 in the controller status file. After an instruction is executed, the arithmetic status bits in the status file are updated.

With this Bit:		The Controller:
S:0/0	Carry (C)	always resets.
S:0/1	Overflow (V)	always resets.
S:0/2	Zero (Z)	sets if result is zero; otherwise resets.
S:0/3	Sign (S)	sets if most significant bit is set; otherwise resets.

Or (OR)



Fixed	SLC 5/01	SLC 5/02	SLC 5/03	SLC 5/04	SLC 5/05
•	•	•	•	•	•

Output Instruction

This instruction performs a bit-by-bit logical OR. The operation is performed using the value at source A and the value at source B. The result is stored in the destination.

Table 5.5 Truth Table for A OR B = Dest

A	B	Dest
0	0	0
1	0	1
0	1	1
1	1	1

Source A and B can either be a word address or a constant; however, both sources cannot be a constant. The destination must be a word address.

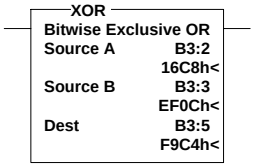
Updates to Arithmetic Status Bits

The arithmetic status bits are found in Word 0, bits 0 to 3 in the controller status file. After an instruction is executed, the arithmetic status bits in the status file are updated.

With this Bit:		The Controller:
S:0/0	Carry (C)	always resets.
S:0/1	Overflow (V)	always resets.
S:0/2	Zero (Z)	sets if result is zero; otherwise resets.
S:0/3	Sign (S)	sets if result is negative (most significant bit is set) otherwise resets.

Exclusive Or (XOR)

This instruction performs a bit-by-bit logical XOR. The operation is performed using the value at source A and the value at source B. The result is stored in the destination.



Fixed	SLC 5/01	SLC 5/02	SLC 5/03	SLC 5/04	SLC 5/05
•	•	•	•	•	•

Output Instruction

Table 5.6 Truth Table for A XOR B = Dest

A	B	Dest
0	0	0
1	0	1
0	1	1
1	1	0

Source A and B can either be a word address or a constant; however, both sources cannot be a constant. The destination must be a word address.

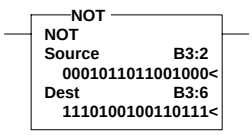
Updates to Arithmetic Status Bits

The arithmetic status bits are found in Word 0, bits 0 to 3 in the controller status file. After an instruction is executed, the arithmetic status bits in the status file are updated.

With this Bit:		The Controller:
S:0/0	Carry (C)	always resets.
S:0/1	Overflow (V)	always resets.
S:0/2	Zero (Z)	sets if result is zero; otherwise resets
S:0/3	Sign (S)	sets if result is negative (most significant bit is set); otherwise resets.

Not (NOT)

This instruction performs a bit-by-bit logical NOT. The operation is performed using the value at source A. The result (one's complement of A) is stored in the destination.



Fixed	SLC 5/01	SLC 5/02	SLC 5/03	SLC 5/04	SLC 5/05
•	•	•	•	•	•

Output Instruction

Table 5.7 Truth Table for A Not = Dest

A	Dest
0	1
1	0

The source and destination must be word addresses.

Updates to Arithmetic Status Bits

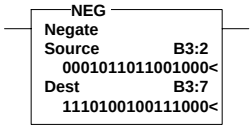
The arithmetic status bits are found in Word 0, bits 0 to 3 in the controller status file. After an instruction is executed, the arithmetic status bits in the status file are updated.

With this Bit:		The Controller:
S:0/0	Carry (C)	always resets.
S:0/1	Overflow (V)	always resets.
S:0/2	Zero (Z)	sets if result is zero; otherwise resets.
S:0/3	Sign (S)	sets if result is negative (most significant bit is set); otherwise resets.

Negate (NEG)

Use the NEG instruction to change the sign of the source and then place it in the destination. The destination contains the two's complement of the source. For example, if the source is 5, the destination would be -5.

The source and destination must be word addresses.



Fixed	SLC 5/01	SLC 5/02	SLC 5/03	SLC 5/04	SLC 5/05
•	•	•	•	•	•

Output Instruction

Updates to Arithmetic Status Bits

The arithmetic status bits are found in Word 0, bits 0 to 3 in the controller status file. After an instruction is executed, the arithmetic status bits in the status file are updated.

Table 5.8

With this Bit:		The Controller:
S:0/0	Carry (C)	clears if 0 or overflow, otherwise sets.
S:0/1	Overflow (V)	sets if overflow, otherwise reset. Overflow occurs only if -32,768 is the source. On overflow, the minor error flag is also set. The value 32,767 is placed in the destination. If S:2/14 is set, then the unsigned, truncated overflow remains in the destination. For floating point destinations, the overflow result remains in the destination.
S:0/2	Zero (Z)	sets if result is zero; otherwise resets.
S:0/3	Sign (S)	sets if result is negative; otherwise resets.

FIFO and LIFO Instructions Overview

FIFO (First in First out) instructions load words into a file and unload them in the same order as they were loaded. The first word in is the first word out.

LIFO (Last in First out) instructions load words into a file and unload them in the opposite order as they were loaded. The last word in is the first word out.

Entering Parameters

Enter the following parameters when programming these instructions:

- **Source** is a word address or constant (-32,768 to 32,767) that becomes the next value in the stack.
- **Destination** is a word address that stores the value that exits from the stack.

This Instruction:	Unloads the Value from:
FIFO's FFU	First word
LIFO's LFU	The last word entered

- FIFO/LIFO is the address of the stack. It must be an indexed word address in the bit, input, output, or integer file.
- Length specifies the maximum number of words in the stack. This is 128 words. Address the length value by mnemonic (LEN).
- Position is the next available location where the instruction loads data into the stack. This value changes after each load or unload operation. Address the position value by mnemonic (POS).
- Control is a control file address. The status bits, the stack length, and the position value are stored in this element. Use the same control file address for the associated FFL and FFU instructions; use the same control file address for the associated LFL and LFU instructions. Do not use the control file address for any other instruction.

Status bits of the control structure are addressed by mnemonic. These include:

- Empty Bit EM (bit 12) is set by the processor to indicate the stack is empty.
- Done Bit DN (bit 13) is set by the processor to indicate the stack is full. This inhibits loading the stack.
- FFU/LFU Enable Bit EU (bit 14) is set on a false-to-true transition of the FFU/LFU rung and is reset on a true-to-false transition.
- FFL/LFL Enable Bit EN (bit 15) is set on a false-to-true transition of the FFL/LFL rung and is reset on a true-to-false transition.

Effects on Index Register S:24

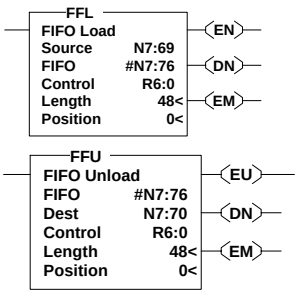
The value present in S:24 is overwritten with the position value when a false-to-true transition of the FFL/FFU or LFL/LFU rung occurs. For the FFL/LFL, the position value determined at instruction entry is placed in S:24. For the FFU/LFU, the position value determined at instruction exit is placed in S:24.

When the DN bit is set, a false-to-true transition of the FFL/LFL rung does not change the position value or the index register value. When the EM bit is set, a false-to-true transition of the FFU/LFU rung does not change the position value or the index register value.

FIFO Load (FFL) and FIFO Unload (FFU)

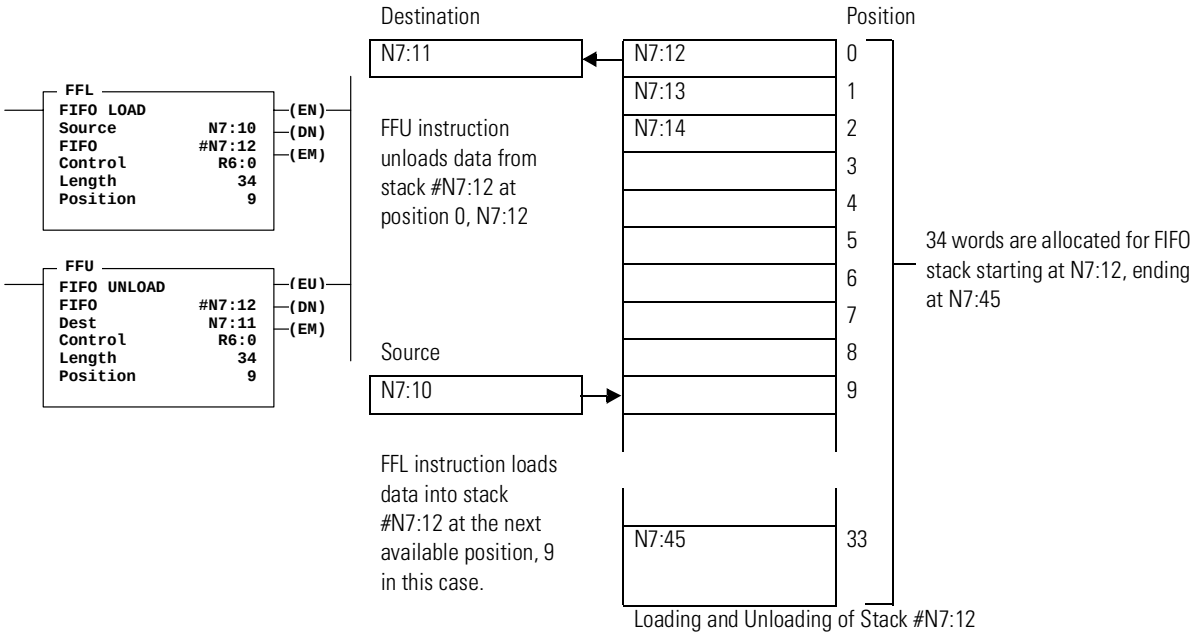
FFL and FFU instructions are used in pairs. The FFL instruction loads words into a user-created file called a FIFO stack. The FFU instruction unloads words from the FIFO stack, in the same order as they were entered.

Instruction parameters have been programmed in the FFL-FFU instruction pair shown on page 5-27.



Fixed	SLC 5/01	SLC 5/02	SLC 5/03	SLC 5/04	SLC 5/05
		•	•	•	•

Output Instructions



FFL Instruction Operation: When rung conditions change from false-to-true, the FFL enable bit (EN) is set. This loads the contents of the source, N7:10, into the stack element indicated by the position number, 9. The position value then increments.

The FFL instruction loads an element at each false-to-true transition of the rung, until the stack is filled (34 elements). The processor then sets the done bit (DN), inhibiting further loading.

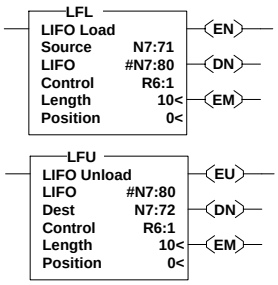
FFU Instruction Operation: When rung conditions change from false-to-true, the FFU enable bit (EU) is set. This unloads the contents of the element at stack position 0 into the destination, N7:11. All data in the stack is shifted one element toward position zero, and the highest numbered element is zeroed. The position value then decrements.

The FFU instruction unloads an element at each false-to-true transition of the rung, until the stack is empty. The processor then sets the empty bit (EM).

LIFO Load (LFL) and LIFO Unload (LFU)

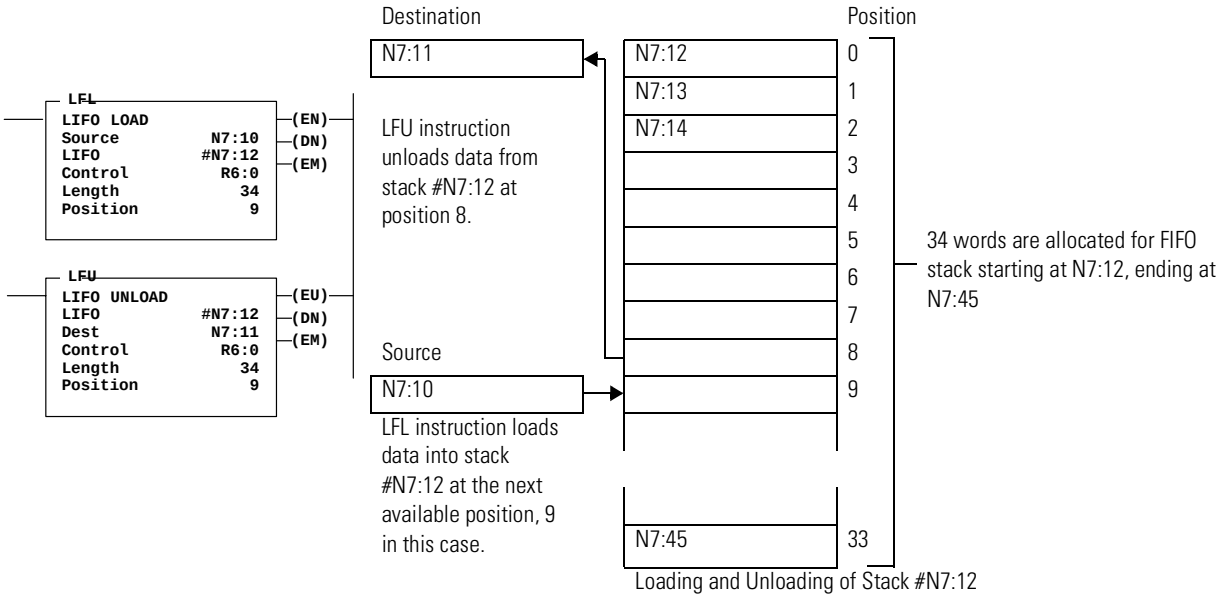
LFL and LFU instructions are used in pairs. The LFL instruction loads words into a user-created file called a LIFO stack. The LFU instruction unloads words from the LIFO stack in the opposite order as they were entered.

Instruction parameters have been programmed in the LFL - LFU instruction pair shown below.



Fixed	SLC 5/01	SLC 5/02	SLC 5/03	SLC 5/04	SLC 5/05
		•	•	•	•

Output Instruction



Loading and Unloading of Stack #N7:12

LFL Instruction Operation: When rung conditions change from false-to-true, the LFL enable bit (EN) is set. This loads the contents of the source, N7:10, into the stack element indicated by the position number, 9. The position value then increments.

The LFL instruction loads an element at each false-to-true transition of the rung, until the stack is filled (34 elements). The processor then sets the done bit (DN), inhibiting further loading.

LFU Instruction Operation: When rung conditions change from false-to-true, the LFU enable bit (EU) is set. This unloads data from the last element loaded into the stack (at the position value minus 1), placing it in the destination, N7:11. The position value then decrements.

The LFU instruction unloads one element at each false-to-true transition of the rung, until the stack is empty. The processor then sets the empty bit (EM).

Program Flow Instructions

This chapter contains general information about the program flow instructions and explains how they function in your application program. Each of the instructions includes information on:

- what the instruction symbol looks like
- how to use the instruction

Table 6.1 Program Flow Instructions

Instruction Mnemonic	Instruction Name	Purpose	Page
JMP and LBL	Jump to Label and Label	Jump forward or backward to the specified label instruction.	6-2
JSR, SBR, and RET	Jump to Subroutine, Subroutine, and Return from Subroutine	Jump to a designated subroutine and return.	6-3
MCR	Master Control Reset	Turn off all non-retentive outputs in a section of ladder program.	6-6
TND	Temporary End	Mark a temporary end that halts program execution.	6-7
SUS	Suspend	Identifies specific conditions for program debugging and system troubleshooting.	6-8
IIM	Immediate Input with Mask	Program an Immediate Input with Mask.	6-8
IOM	Immediate Output with Mask	Program an Immediate Output with Mask.	6-9
REF	Refresh	Interrupt the program scan to execute the I/O scan and service communications.	6-10

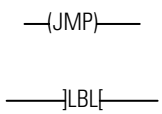
About the Program Flow Control Instructions

Use these instructions to control the sequence in which your program is executed.

Control instructions allow you to change the order in which the processor scans a ladder program. Typically, these instructions are used to minimize scan time, create a more efficient program, and troubleshoot a ladder program.

Jump (JMP) and Label (LBL)

Use these instructions in pairs to skip portions of the ladder program.



Fixed	SLC 5/01	SLC 5/02	SLC 5/03	SLC 5/04	SLC 5/05
•	•	•	•	•	•

If the Rung Containing the Jump Instruction is:	Then the Program:
True	Skips from the rung containing the JMP instruction to the rung containing the designated LBL instruction and continues executing. You can jump forward or backward.
False	Does not execute the JMP instruction.

Jumping forward to a label saves program scan time by omitting a program segment until needed. Jumping backward lets the controller execute program segments repeatedly.

TIP

Be careful not to jump backwards an excessive number of times. The watchdog timer could time out and fault the controller. Use a counter, timer, or the "program scan" register (system status register, word S:3, bits 0 to 7) to limit the amount of time you spend looping inside of JMP/LBL instructions.

Entering Parameters

Enter a decimal label number from 0 to 255 in each subroutine file.

Using JMP

The JMP instruction causes the controller to skip rungs. You can jump to the same label from one or more JMP instructions.

Using LBL

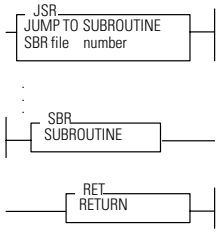
This input instruction is the target of JMP instructions having the same label number. You must program this instruction as the first instruction of a rung. This instruction has no control bits.

You can program multiple jumps to the same label by assigning the same label number to multiple JMP instructions. However, label numbers must be unique.

TIP

Do not jump (JMP) into an MCR zone. Instructions that are programmed within the MCR zone starting at the LBL instruction and ending at the "END MCR" instruction are always evaluated as though the MCR zone is true, regardless of the true state of the "Start MCR" instruction.

Jump to Subroutine (JSR), Subroutine (SBR), and Return (RET)



Fixed	SLC 5/01	SLC 5/02	SLC 5/03	SLC 5/04	SLC 5/05
•	•	•	•	•	•

The JSR, SBR, and RET instructions are used to direct the controller to execute a separate subroutine file within the ladder program and return to the instruction following the JSR instruction.

TIP

If you use the SBR instruction, the SBR instruction must be the first instruction on the first rung in the program file that contains the subroutine.

Use a subroutine to store recurring sections of program logic that must be executed from several points within your application program. A subroutine saves memory because you program it only once.

Update critical I/O within subroutines using immediate input and/or output instructions (IIM, IOM), especially if your application calls for nested or relatively long subroutines. Otherwise, the controller does not update I/O until it reaches the end of the main program (after executing all subroutines).

ATTENTION

Outputs controlled within a subroutine remain in their last state until the subroutine is executed again.

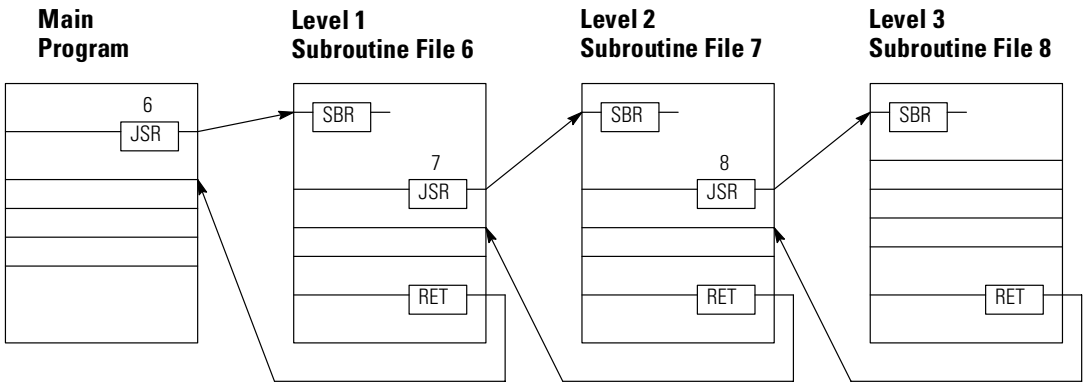
Nesting Subroutine Files

Nesting subroutines allows you to direct program flow from the main program to a subroutine and then on to another subroutine. The following rules apply when nesting subroutines:

- With Fixed and SLC 5/01 processors, you can nest subroutines up to four levels.
- With SLC 5/02 and higher processors, you can nest subroutines up to eight levels. If you are using an STI subroutine, I/O event-driven interrupt subroutine, user fault routine, or HSC interrupt subroutine, you can nest subroutines up to three levels from each subroutine.

The following figure illustrates how subroutines may be nested.

Figure 6.1 Example of Nesting Subroutines to Level 3



An error occurs if more than the allowable levels of subroutines are called (subroutine stack overflow) or if more returns are executed than there are call levels (subroutine stack underflow).

Using JSR

When the JSR instruction is executed, the controller jumps to the subroutine instruction (SBR) at the beginning of the target subroutine file and resumes execution at that point. You cannot jump into any part of a subroutine except the first instruction in that file.

You must program each subroutine in its own program file by assigning a unique file number (3 to 255)

IMPORTANT *Fixed and SLC 5/01 specific* - The JSR instruction cannot be programmed in nested output branches. A compiler error will occur if a rung containing multiple outputs with conditional logic and a JSR instruction is encountered.

Using SBR

The target subroutine is identified by the file number that you entered in the JSR instruction. This instruction serves as a label or identifier for a program file as a regular subroutine file.

This instruction has no control bits. It is always evaluated as true. The instruction must be programmed as the first instruction of the first rung of a subroutine. Use of this instruction is optional; however, we recommend using it for clarity.

Using RET

This output instruction marks the end of subroutine execution or the end of the subroutine file. It causes the controller to resume execution at the instruction following the JSR instruction. If a sequence of nested subroutines is involved, the instruction causes the processor to return program execution to the previous subroutine.

The rung containing the RET instruction may be conditional if this rung precedes the end of the subroutine. In this way, the controller omits the balance of a subroutine only if its rung condition is true.

Without an RET instruction, the END instruction (always present in the subroutine) automatically returns program execution to the instruction following the JSR instruction in your calling ladder file.

TIP The RET instruction terminates execution of the DII subroutine (SLC 5/03 and higher processors), STI subroutine, I/O event-driven interrupt subroutine, and the user error handler when an SLC 5/02 or higher processor is used.

Master Control Reset (MCR)

Use MCR instructions in pairs to create program zones that turn off all the non-retentive outputs in the zone. Rungs within the MCR zone are still scanned, but scan time is reduced due to the false state of non-retentive outputs.

—(MCR)—

Fixed	SLC 5/01	SLC 5/02	SLC 5/03	SLC 5/04	SLC 5/05
•	•	•	•	•	•

If the MCR Rung that Starts the Zone is:	Then the Controller:
True	Executes the rungs in the MCR zone based on each rung's individual input condition (as if the zone did not exist).
False	Resets all non-retentive output instructions in the MCR zone regardless of each rung's individual input conditions.

MCR zones let you enable or inhibit segments of your program, such as for recipe applications.

When you program MCR instructions, note that:

- You must end the zone with an unconditional MCR instruction.
- You cannot nest one MCR zone within another.
- Do not jump into an MCR zone. If the zone is false, jumping into it activates the zone.
- Always place the MCR instruction as the last instruction in a rung.

Processor Operation

Do not jump (JMP) into an MCR zone. Instructions that are programmed within the MCR zone starting at the LBL instruction and ending at the 'END MCR' instruction are always evaluated as though the MCR zone is true, regardless of the true state of the "Start MCR" instruction. If the zone is false, jumping into it activates the zone from the LBL to the end of the zone.

ATTENTION

If you start instructions such as timers or counters in an MCR zone, instruction operation ceases when the zone is disabled. Re-program critical operations outside the zone if necessary.

The TOF timer activates when placed inside of a false MCR zone.

The MCR instruction is not a substitute for a hard-wired master control relay. We recommend that your programmable controller system include a hard-wired master control relay and emergency stop switches to provide I/O power shut down. Emergency stop switches can be monitored but should not be controlled by the ladder program. Wire these devices as described in the installation manual.

SLC 5/03 and higher processors - When online and an unmatched MCR instruction exists in your program, the END instruction acts as the second unconditional MCR instruction and all of the rungs following the first MCR instruction execute via the current MCR instruction state.

You can save the program while online if unattended MCR instructions exist. However, if you are offline and unattended MCR instructions exist, an error will occur.

Temporary End (TND)

This instruction, when its rung is true, stops the processor from scanning the rest of the program file, updates the I/O, and resumes scanning at rung 0 of the main program (file 2). If this instruction's rung is false, the processor continues the scan until the next TND instruction or the END statement. Use this instruction to progressively debug a program, or conditionally omit the balance of your current program file or subroutines.

—(TND)—

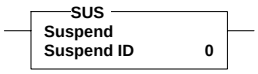
Fixed	SLC 5/01	SLC 5/02	SLC 5/03	SLC 5/04	SLC 5/05
•	•	•	•	•	•

Output Instruction

TIP

If you use this instruction inside a nested subroutine, execution of all nested subroutines is terminated.

Suspend (SUS)



Fixed	SLC 5/01	SLC 5/02	SLC 5/03	SLC 5/04	SLC 5/05
•	•	•	•	•	•

Output Instruction

When this instruction is executed, it causes the processor to enter the Suspend Idle mode and stores the Suspend ID in word 7 (S:7) of the status file. All outputs are de-energized.

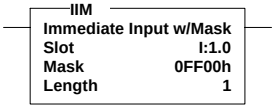
Use this instruction to trap and identify specific conditions for program debugging and system troubleshooting.

Entering Parameters

Enter a suspend ID number from -32,768 to +32,767 when you program the instruction.

When the SUS instruction is executed, the programmed suspend ID, S:7 (word 7), as well as the program file ID, S:8 (word 8), from which the SUS instruction executed is placed in the system status file.

Immediate Input with Mask (IIM)



Fixed	SLC 5/01	SLC 5/02	SLC 5/03	SLC 5/04	SLC 5/05
•	•	•	•	•	•

Input Instruction

This instruction allows you to update data prior to the normal input scan. When the IIM instruction is enabled, the program scan is interrupted. Data from a specified I/O slot is transferred through a mask to the input data file, making the data available to instructions following the IIM instruction in the ladder program.

Entering Parameters

Slot - Specify the input slot number and the word number pertaining to the slot. Word 0 of a slot need not be specified. Fixed and SLC 5/01 processors can have up to 8 words associated with the slot. The SLC 5/02 and higher processors can have up to 32 words associated with the slot (0 to 31).

For 16 I/O controllers, I:0/0 to 9 are valid and I:0/10 to 15 are considered unused inputs. (They do not physically exist.) For 32 I/O controllers, I:0/0 to 15 and I:1/0 to 3 are valid. Specify I:1 if you want to immediately update the last four input bits.

For the mask, a 1 in an input's bit position passes data from the source to the destination. A 0 inhibits data from passing from the source to the destination.

Example

I:2	Inputs of slot 2, word 0
I:2.1	Inputs of slot 2, word 1
I:1	Inputs of slot 1, word 0

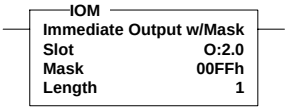
Mask - Specify a hexadecimal constant or register address.

Refer to Entering Mask Values on page 5-17 for information about entering mask.

Length - For SLC 5/03 and higher processors, this parameter is used to transfer more than one word per slot. Valid value is from 1 to 32.

Immediate Output with Mask (IOM)

This instruction allows you to update the outputs prior to the normal output scan. When the IOM instruction is enabled, the program scan is interrupted to transfer data to a specified I/O slot through a mask. The program scan then resumes.



Fixed	SLC 5/01	SLC 5/02	SLC 5/03	SLC 5/04	SLC 5/05
•	•	•	•	•	•

Output Instruction

Entering Parameters

Slot - Specify the slot number and the word number pertaining to the slot. Word 0 of a slot need not be specified. Fixed and SLC 5/01 processors can have up to 8 words associated with the slot. The SLC 5/02 and higher processors can have up to 32 words associated with the slot (0 to 30).

For 16 I/O controllers, O:0/0 to 5 are valid and O:0/6 to 15 are considered unused outputs. (They do not physically exist.) For 32 I/O controllers, O:0/0 to 11 are valid and O:0/12 to 15 are considered unused outputs.

Example

O:2	Outputs of slot 2, word 0
O:1	Outputs of slot 1, word 0
O:2.1	Outputs of slot 2, word 1

Mask - Specify a hexadecimal constant or register address.

For the mask, a 1 in the output bit position passes data from the source to the destination. A 0 inhibits the data from passing from the source to the destination.

Refer to Entering Mask Values on page 5-17 for information about entering mask.

Length - For SLC 5/03 and higher processors, this parameter is used to transfer more than one word per slot. Valid value is from 1 to 32.

I/O Refresh (REF)

—(REF)—

SLC 5/02 Processor

Fixed	SLC 5/01	SLC 5/02	SLC 5/03	SLC 5/04	SLC 5/05
		•	•	•	•

Output Instruction

Using an SLC 5/02 Processor

The REF instruction has no programming parameters. When it is evaluated as true, the program scan is interrupted to execute the I/O scan and service communication portions of the operating cycle (write outputs, service comms, read inputs). The scan then resumes at the instruction following the REF instruction.

You are not allowed to place a REF instruction in a DII subroutine, STI subroutine, I/O subroutine, or user fault subroutine.

ATTENTION

The watchdog and scan timers are reset when executing the REF instruction. You must insure that an REF instruction is not placed inside a non-terminating program loop. Do not place an REF instruction inside a program loop unless the program is thoroughly analyzed.

Using SLC 5/03 and Higher Processors

Operation of the REF instruction in the SLC 5/03 and higher processors is the same as the SLC 5/02 processor. However, when using the SLC 5/03 and higher processors, you can also select a specific communication channel to be serviced.

- SLC 5/03 processor
  - channel 0 is RS-232/DF1 Full-Duplex or Half-Duplex (master or slave), DH-485, or ASCII
  - channel 1 is DH-485
- SLC 5/04 processor
  - channel 0 is RS-232/DF1 Full-Duplex or Half-Duplex (master or slave), DH-485, or ASCII
  - channel 1 is DH+
- SLC 5/05 processor
  - channel 0 is RS-232/DF1 Full-Duplex or Half-Duplex (master or slave), DH-485, or ASCII
  - channel 1 is Ethernet

Application Specific Instructions

This chapter contains general information about the application specific instructions and explains how they function in your application program. Each of the instructions includes information on:

- what the instruction symbol looks like
- how to use the instruction

Table 7.1 Application Specific Instructions

Instruction Mnemonic	Instruction Name	Purpose	Page
BSL and BSR	Bit Shift Left and Bit Shift Right	Loads a bit of data into a bit array, shifts the pattern of data through the array, and unloads the last bit of data in the array. The BSL shifts data to the left and the BSR shifts data to the right.	**7-4**
SQO and SQC	Sequencer Output and Sequencer Compare	Controls sequential machine operations by transferring 16-bit data through a mask to image addresses.	**7-6**
SQL	Sequencer Load	Captures referenced conditions by manually stepping the machine through its operating sequences.	**7-12**
RHC	Read High Speed Clock	Provides a high performance time-stamp for performance diagnostics and preforming calculations such as velocity.	**7-17**
TDF	Compute Time Difference	Calculates the number of 10 µs "ticks" between any two time-stamps captured using the RHC instruction.	**7-17**
FBC	File Bit Comparison	Used to monitor machine or process operations to detect malfunctions.	**7-18**
DDT	Diagnostic Detect		**7-18**

About the Application Specific Instructions

These instructions simplify your ladder program by allowing you to use a single instruction or pair of instructions to perform common complex operations.

In this chapter you will find a general overview preceding groups of instructions. Before you learn about the instructions in each of these groups, we suggest that you read the overview that precedes each section. This chapter contains the following overviews:

- Bit Shift Instructions Overview
- Sequencer Instructions Overview
- RHC/TDF Instructions Overview

Bit Shift Instructions Overview

The following general information applies to bit shift instructions.

Entering Parameters

Enter the following parameters when programming these instructions:

- **File** is the address of the bit array you want to manipulate. You must use the file indicator (#) in the bit array address.
- **Control** is the control element that stores the status byte of the instruction and the size of the array (in number of bits). Note that the control address should not be used for any other instruction.

  The control element is shown below.

.

Table 7.2 Control File Structure

	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
Word 0	EN		DN		ER	UL			Not Used							
Word 1	Size of bit array (number of bits)															
Word 2	Reserved															

Status bits of the control element may be addressed by mnemonic. They include:

- Unload Bit UL (bit 10) stores the status of the bit exited from the array each time the instruction is enabled.

- Error Bit ER (bit 11), when set, indicates the instruction detected an error such as entering a negative number for the length or position. Avoid using the output bit when this bit is set.
- Done Bit DN (bit 13), when set, indicates the bit array has shifted one position.
- Enable Bit EN (bit 15) is set on a false-to-true transition of the rung and indicates the instruction is enabled.

When the register shifts and input conditions go false, the enable, done, and error bits are reset.

- **Bit Address** is the address of the source bit that the instruction inserts in the first (lowest) bit position (BSL) or the last (highest) bit position (BSR).
- **Length** (size of bit array) is the number of bits in the bit array, up to 2048 bits. A length value of 0 causes the input bit to be transferred to the UL bit.

  A length value that points past the end of the programmed file causes a runtime major error to occur.

TIP If you alter a length value with your ladder program, make certain that the altered value is valid.

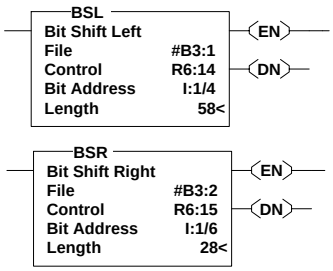
The instruction invalidates all bits beyond the last bit in the array (as defined by the length) up to the next word boundary

TIP If a String element address is used for the file parameter, the maximum length for SLC 5/03 and higher processors is 672 bits. Additionally, String element boundaries cannot be crossed.

Effects on Index Register S:24

The shift operation clears the index register S:24 to zero.

Bit Shift Left (BSL) Bit Shift Right (BSR)



Fixed	SLC 5/01	SLC 5/02	SLC 5/03	SLC 5/04	SLC 5/05
•	•	•	•	•	•

Output Instructions

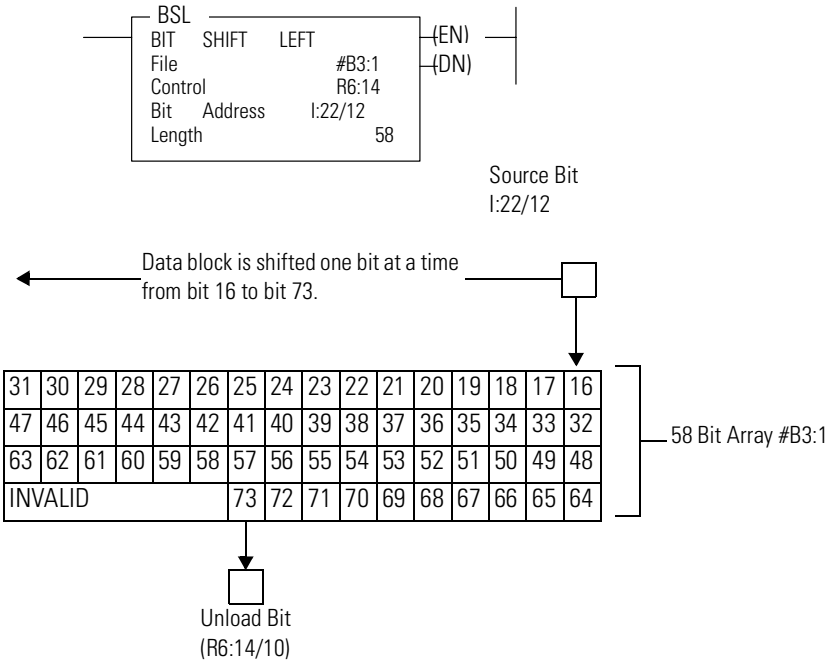
BSL and BSR are output instructions that load data into a bit array one bit at a time. The data is shifted through the array, then unloaded one bit at a time.

Using BSL

When the rung goes from false-to-true, the processor sets the enable bit (EN bit 15) and the data block is shifted to the left (to a higher bit number) one bit position. The specified bit at the bit address is shifted into the first bit position. The last bit is shifted out of the array and stored in the unload bit (UL bit 10). The shift is completed immediately.

For wraparound operation, set the position of the bit address to the last bit of the array or to the UL bit, whichever applies.

The figure below illustrates how the Bit Shift Left instruction works.



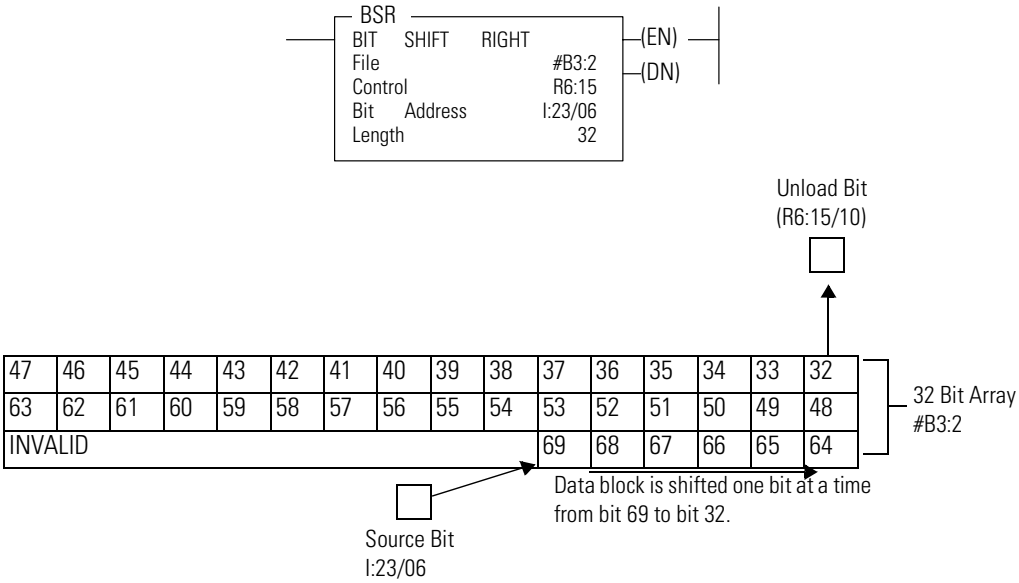
If you wish to shift more than one bit per scan, you must create a loop in your application using the JMP, LBL, and CTU instructions.

Using BSR

When the rung goes from false-to-true, the enable bit (EN bit 15) is set and the data block is shifted to the right (to a lower bit number) one bit position. The specified bit at the bit address is shifted into the last bit position. The first bit is shifted out of the array and stored in the unload bit (UL bit 10) in the status byte of the control element. The shift is completed immediately.

For wraparound operation, set the position of the bit address to the first bit of the array or to the UL bit, whichever applies.

The figure below illustrates how the Bit Shift Right instruction works.



If you wish to shift more than one bit per scan, you must create a loop in your application using the JMP, LBL, and CTU instructions.

Sequencer Instructions Overview

The following general information applies to sequencer instructions.

Effects on Index Register S:24

The value present in the index register S:24 is overwritten when the sequencer instruction is true. The index register value will equal the position value of the instruction.

Applications Requiring More than 16-Bits

When your application requires more than 16-bits, use parallel multiple sequencer instructions.

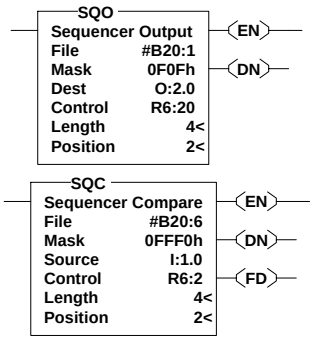
TIP

Refer to Appendix G for application examples using the sequencer instructions.

If a String element address is used for the file parameter, the maximum length for SLC 5/03 and higher processors is 41 words. Additionally, String element boundaries cannot be crossed.

Sequencer Output (SQO) Sequencer Compare (SQC)

These instructions transfer 16-bit data to word addresses for the control of sequential machine operations.



Fixed	SLC 5/01	SLC 5/02	SLC 5/03	SLC 5/04	SLC 5/05
•	•	•	•	•	•

Output Instructions

Entering Parameters

Enter the following parameters when programming these instructions:

- **File** is the address of the sequencer file. You must use the file indicator (#) for this address.

  Sequencer file data is used as follows:

Instruction	Sequencer File Stores
SQO	Data for controlling outputs
SQC	Reference data for monitoring inputs

- **Mask** (SQO, SQC) is a hexadecimal code or the address of the mask word or file through which the instruction moves data. Set mask bits to pass data and reset mask bits to mask data. Use a mask word or file if you want to change the mask according to application requirements.

  If the mask is a file, its length will be equal to the length of the sequencer file. The two files track automatically.

- **Source** is the address of the input word or file for a SQC from which the instruction obtains data for comparison to its sequencer file.

- **Destination** is the address of the output word or file for a SQO to which the instruction moves data from its sequencer file.

TIP You can address the mask, source, or destination of a sequencer instruction as a word or file. If you address it as a file, the instruction automatically steps through the source, mask, or destination file.

- **Control** (SQO, SQC) is the control structure that stores the status byte of the instruction, the length of the sequencer file, and the instantaneous position in the file. You should not use the control address for any other instruction.

Table 7.3 Control File Structure

	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
Word 0	EN		DN		ER			FD	Not Used							
Word 1	Length of sequencer file															
Word 2	Position															

Status bits of the control structure include:

- Found Bit FD (bit 08) - SQC only. When the status of all non-masked bits in the source address match those of the corresponding reference word, the FD bit is set. This bit is assessed each time the SQC instruction is evaluated while the rung is true.
- Error Bit ER (bit 11) is set when the processor detects a negative position value, or a negative or zero length value. This results in a major error if not cleared before the END or TND instruction is executed.
- Done Bit DN (bit 13) is set by the SQO or SQC instruction after it has operated on the last word in the sequencer file. It is reset on the next false-to-true rung transition after the rung goes false.
- Enable EN (bit 15) is set by a false-to-true rung transition and indicates the SQO or SQC instruction is enabled.

- **Length** is the number of steps of the sequencer file starting at position 1. The maximum number you can enter is 255 words. Position 0 is the startup position. The instruction resets (wraps) to position 1 at each cycle completion.

The address assigned for a sequencer file is step zero. Sequencer instructions use length +1 word of data table files for each file referenced in the instruction. This applies to the source, mask, and/or destination if addressed as files.

A length value that points past the end of the programmed file causes a runtime major error to occur.

TIP If you alter a length value with your ladder program, make certain that the altered value is valid.

- **Position** is the word location or step in the sequencer file from/to which the instruction moves data.

  A position value that points past the end of the programmed file causes a runtime major error to occur.

TIP You may use the reset (RES) instruction to reset a sequencer. All control bits (except FD) will be reset to zero. The Position will also be set to zero. Program the address of your control register in the RES (e.g.,R6:0).

Using SQO

This output instruction steps through the sequencer file whose bits have been set to control various output devices.

When the rung goes from false-to-true, the instruction increments to the next step (word) in the sequencer file. Data stored there is transferred through a mask to the destination address specified in the instruction. Current data is written to the corresponding destination word every scan that the rung remains true.

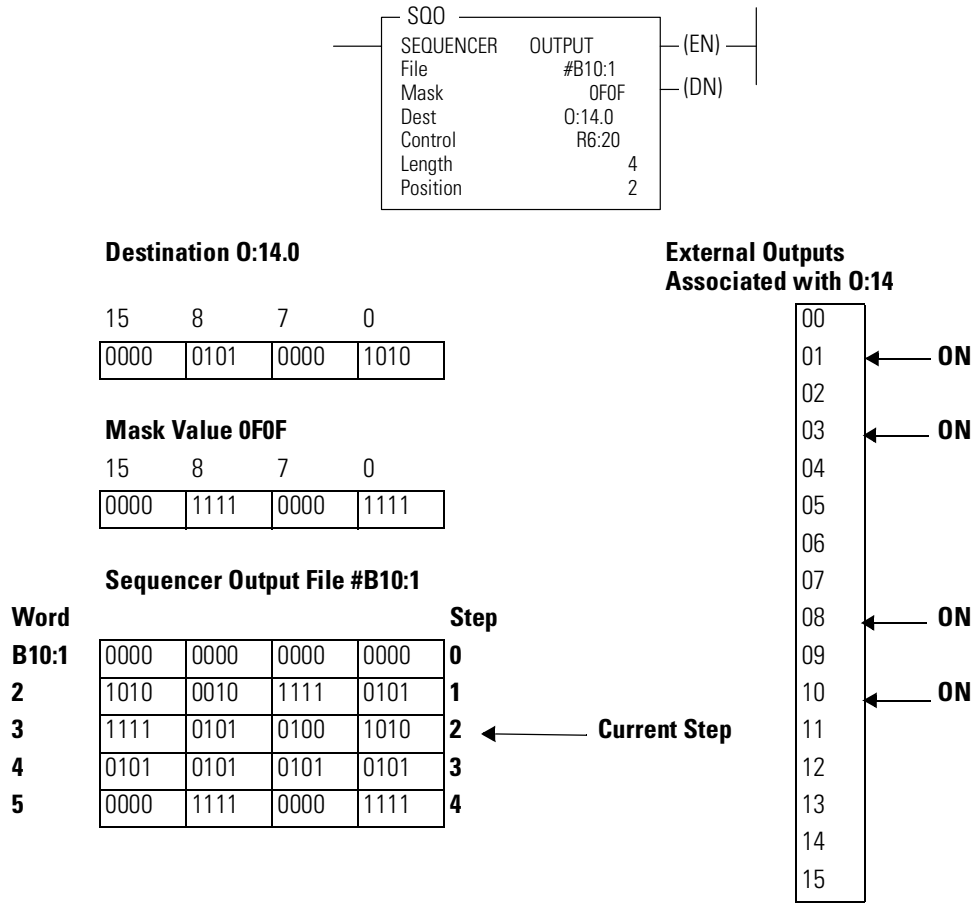
The done bit is set when the last word of the sequencer file is transferred. On the next false-to-true rung transition, the instruction resets the position to step one.

If the position is equal to zero at startup, when you switch the processor from the program mode to the run mode instruction operation depends on whether the rung is true or false on the first scan.

- If true, the instruction transfers the value in step zero.
- If false, the instruction waits for the first rung transition from false-to-true and transfers the value in step one.

The bits mask data when reset and pass data when set. The instruction will not change the value in the destination word unless you set mask bits. The mask can be fixed or variable. If you enter a hexadecimal code, it is fixed. If you enter an element address or a file address for changing the mask with each step, it is variable.

The following figure indicates how the SQO instruction works.



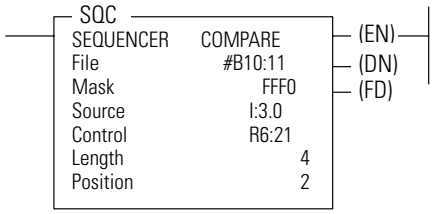
Using SQC

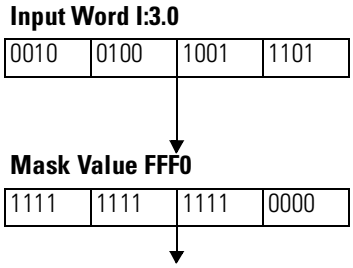
When the status of all non-masked bits in the source word match those of the corresponding reference word, the instruction sets the found bit (FD) in the control word. Otherwise, the found bit (FD) is cleared.

The bits mask data when reset and pass data when set. The instruction will not change the value in the destination word unless you set the mask bits. The mask can be fixed or variable. If you enter a hexadecimal code, it is fixed. If you enter an element address or a file address for changing the mask with each step, it is variable.

When the rung goes from false-to-true, the instruction increments to the next step (word) in the sequencer file. Data stored there is transferred through a mask and compared against the source data for equality. If the source data equals the reference data, the FD bit is set in the SQC's control counter. Current data is compared against the source every scan that the rung evaluates as true.

Applications of the SQC instruction include machine diagnostics. The following figure explains how the SQC instruction works.





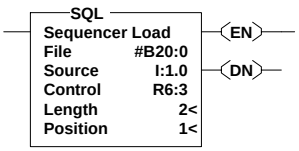
Sequencer Ref File #B10:11

Word		Step
B10:11		0
12		1
13	0010 0100 1001 0000	2
14		3
15		4

SQC FD bit is set when the instruction detects that an input word matches (through mask) its corresponding reference word.

The FD bit R6:21/FD is set in the example, since the input word matches the sequencer reference value using the mask value.

Sequencer Load (SQL)



Fixed	SLC 5/01	SLC 5/02	SLC 5/03	SLC 5/04	SLC 5/05
•	•	•	•	•	•

Output Instruction

The SQL instruction stores 16-bit data into a sequencer load file at each step of sequencer operation. The source of this data can be an I/O or storage word address, a file address, or a constant.

Entering Parameters

Enter the following parameters when programming this instruction:

- **File** is the address of the sequencer file. You must use the indexed file indicator (#) for this address.
- **Source** can be a word address, file address, or a constant (-32768 to 32767).

  If the source is a file address, the file length equals the length of the sequencer load file. The two files will step automatically, per the position value.

- **Length** is the number of steps of the sequencer load file (and also of the source if the source is a file address), starting at position 1. The maximum number you can enter is 255 words. Position 0 is the startup position. The instruction resets (wraps) to position 1 at each cycle completion.

  The position address assigned for a sequencer file is step zero. Sequencer instructions use length plus one word of data for each file referenced in the instruction. This applies to the source if addressed as a file.

  A length value that points past the end of the programmed file causes a runtime major error to occur.

TIP If you alter a length value with your ladder program, make certain that the altered value is valid.

- **Position** is the word location or step in the sequencer file to which data is moved.

  A position value that points past the end of the programmed file causes a runtime major error to occur.

TIP If you alter a length value with your ladder program, make certain that the altered value is valid.

- **Control** is a control file address. The status bits, length value, and position value are stored in this element. Do not use the control file address for any other instruction.

  The control element is shown below:

Table 7.4 Control File Structure

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Word 0	EN		DN		ER				Not Used							
Word 1	Length															
Word 2	Position															

Status bits of the control structure include:

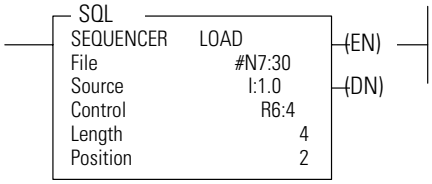
- Error Bit ER (bit 11) is set when the processor detects a negative position value, or a negative or zero length value.

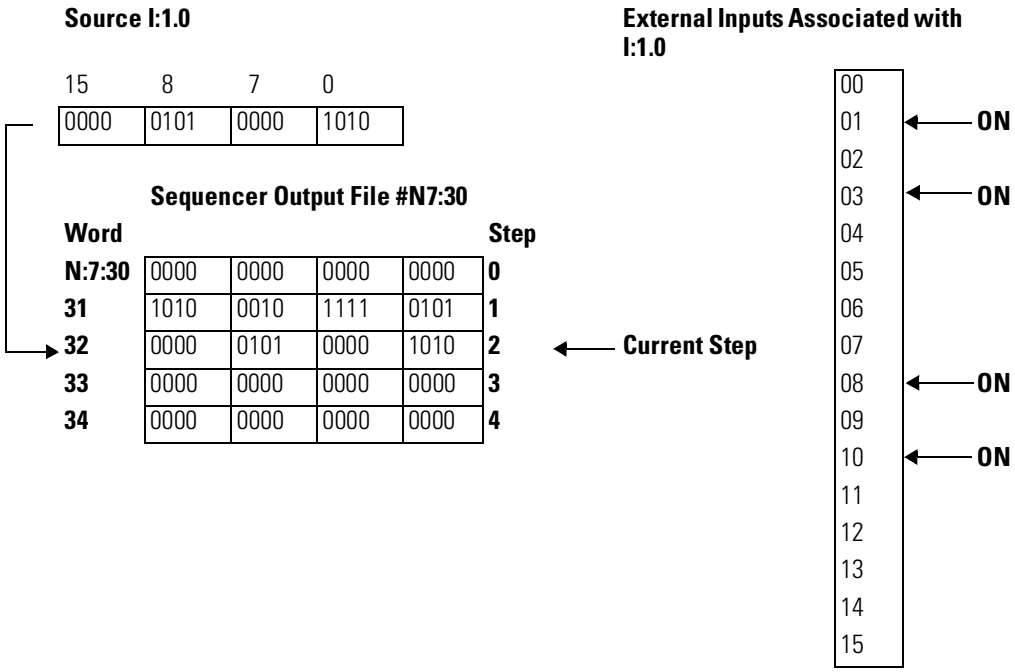
  For SLC processors, this results in a major error if not cleared before the END or TND instruction is executed.

- Done Bit DN (bit 13) is set after the instruction has operated on the last word in the sequencer load file. It is reset on the next false-to-true rung transition after the rung goes false.
- Enable Bit EN (bit 15) is set on a false-to-true transition of the SQL rung and reset on a true-to-false transition.

Operation

Instruction parameters have been programmed in the SQL instruction shown below. Input word I:1.0 is the source. Data in this word is loaded into integer file #N7:30 by the sequencer load instruction.





When rung conditions change from false-to-true, the SQL enable bit (EN) is set. The control element R6:4 increments to the next position in the sequencer file, and loads the contents of source I:1.0 into this location. The SQL instruction continues to load the current data into this location each scan that the rung remains true. When the rung goes false, the enable bit (EN) is reset.

The instruction loads data into a new file element at each false-to-true transition of the rung. When step 4 is completed, the done bit (DN) is set. Operation cycles to position 1 at the next false-to-true transition of the rung after position 4.

If the source were a file address such as #N7:40, files #N7:40 and #N7:30 would both have a length of 5 (0 to 4) and would track through the steps together per the position value.

Read High-Speed Clock and Compute Time Difference Overview

TDF and RHC instructions are used together. The RHC is used to record the start and stop time of an event. The TDF is used to calculate the time difference between the recorded start and stop times from the RHC instruction.

RHC Instruction Operation

SLC 500 maintains a 20-bit integer free running clock. This 20-bit value increments every 10 µs. The free running clock is non-retentive, a power cycle resets the free running clock to 0. It is accessed using the RHC instruction. When the RHC rung is true, the instruction moves the current value of the 10 µs free running clock into the destination address. If the destination is an integer address, the RHC moves the first 16 least significant bits to the destination address. If the destination is a float address, the instruction converts the 20-bit free running clock integer value into a float and moves this value to the destination address. Once the free running clock reaches 0x000F FFFF (10.48575 seconds), it wraps around to 0 and continues incrementing.

TIP

The RHC instruction does have an inherent latency due to execution time. The 20-bit float and 16-bit integer do not have the same amount of latency. A 20-bit float destination has additional latency due to the integer to float conversion. The accuracy of this instruction is based on the latency of the RHC instruction and potential hardware interrupts. See Table 7.5, "Accuracy (in counts: 1 count = 10 µs)," for more information.

Table 7.5 Accuracy (in counts: 1 count = 10 µs)

Processor		Best Case	Worst Case	Typical
SLC 5/05	Integer	1	26	1
	Float	1	29	2
SLC 5/04	Integer	1	26	1
	Float	1	29	2
SLC 5/03	Integer	1	53	2
	Float	2	62	3

TIP

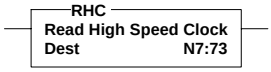
Measurements were calculated with both communication channels active and no devices connected to the processor. Worst case accuracy is improved by shutting down an unused communication channel.

TDF Instruction Operation

When the TDF is evaluated with a true rung state, the instruction calculates the number of 10 µs "ticks" that have elapsed from the Start value to the Stop value and places the result into the Destination location. The TDF instruction with float addresses accurately computes the time difference between the Start and Stop timestamps captured within 10.48575 seconds of each other (1048575 10 µs ticks). The TDF with float addresses calculates an invalid result if more than 10.48575 seconds have elapsed between the Start and Stop timestamps. The TDF with integer addresses accurately computes the time difference between the Start and Stop timestamps captured within 655.36 ms of each other (65536 10 µs ticks). The TDF with integer address calculates an invalid result if more than 655.36 ms have elapsed between the Start and Stop timestamps. It is up to the user to assure that the timestamps are captured within the valid time difference range.

Read High-Speed Clock Instruction (RHC)

The Read High-Speed Clock Instruction (RHC) provides a high performance timestamp for performance diagnostics and performing calculations such as velocity.



Fixed	SLC 5/01	SLC 5/02	SLC 5/03	SLC 5/04	SLC 5/05
			•	•	•

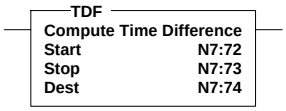
Entering Parameters

- **Destination** - The address to store the current value of the 10 µs free running clock. It can be an integer address (Nx:x) or Float address (Fx:x). The integer address supports 16 bits range time (0 to 655.36 ms). The float address gives the exact time for the free running clock value with 20 bits range time (0 to 10.48575 seconds).

Compute Time Difference Instruction (TDF)

The compute Time Difference Instruction (TDF) is used to calculate the number of 10 µs "ticks" between any two time-stamps captured using the RHC instruction. This allows the user program to determine the time difference between any two events using a 10 µs timebase.

Entering Parameters



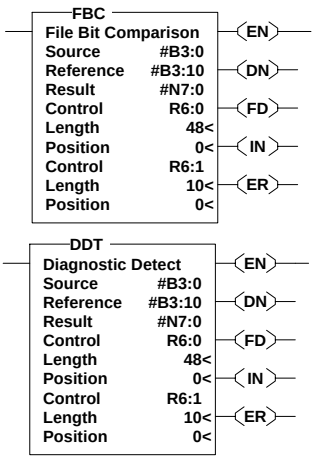
Fixed	SLC 5/01	SLC 5/02	SLC 5/03	SLC 5/04	SLC 5/05
			•	•	•

This instruction has three parameters. All of these parameters should be of the same data type (Nx:x or Fx:x).

- **Start** - The address of the earliest value previously captured using the RHC instruction.
- **Stop** - The address of a later value captured using the RHC instruction.
- **Destination** - The address to store the result of the time difference calculation.

File Bit Comparison (FBC) and Diagnostic Detect (DDT)

The FBC and DDT diagnostic instructions are output instructions that you use to monitor machine or process operations to detect malfunctions.



Fixed	SLC 5/01	SLC 5/02	SLC 5/03	SLC 5/04	SLC 5/05
			•	•	•

Table 7.6 Available Diagnostic Instructions

If You Want to Detect Malfunctions By:	Use this Instruction:
Comparing bits in a file of real-time inputs with a reference bit file that represents correct operation	FBC
Change-of-state diagnostics	DDT

Both the FBC and DDT instructions compare bits in a file of real-time machine or process values (input file) with bits in a reference file, detect deviations, and record mismatched bit numbers. These instructions record the position of each mismatch found and place this information in the result file. If no mismatches are found, the DN bit is set and the result file remains unchanged.

The difference between the DDT and FBC instruction is that each time the DDT instruction finds a mismatch, the processor changes the reference bit to match the source bit. The FBC instruction does not change the reference bit. Use the DDT instruction to update your reference file to reflect changing machine or process conditions.

Selecting the Search Mode

Select whether the diagnostic instruction searches for a mismatch one bit at a time or whether it searches for all mismatches during one program scan.

One Bit at a Time

With each false-to-true rung transition, the instruction compares the next bit between the input and reference files. If a mismatch is detected, the instruction stops and sets the found FD bit. Then the instruction enters the position number of the mismatch into the result file.

The DDT instruction also changes the status of the reference bit to match the status of the corresponding input bit. The instructions resets the found bit when the rung goes false.

After the instruction compares the last bit in two files, the done bit (bit 13 DN on the compare control element) is set. Then, when the rung goes false, the instruction resets:

- enable bit
- found bit (if set)
- compare done bit
- result done bit (if set)

The control position counters are reset on the next false-to-true rung transition. To enable this mode of operation, set the inhibit bit (IN=1) either by ladder program or manually before program execution.

All Per Scan

After a false-to-true rung transition, the instruction searches for all mismatches between the input and reference files in one program scan. Upon finding mismatches, the instruction enters the position numbers of mismatched bits into the result file in the order it finds them. After reaching the end of the input and reference files, the instruction sets the FD bit if it finds at least one mismatch. Then the instruction sets the DN bit.

If you use a result file that cannot hold all detected mismatches (if the result file fills), the instruction stops and requires another false-to-true rung transition to continue operation. The instruction wraps the new mismatched bit positions into the beginning of the result file writing over the old.

TIP To detect one mismatch at a time, set the result length value to one.

After completing the comparison and when the rung goes false, the instruction resets:

- enable bit
- found bit (if set)
- compare done bit
- result done bit (if set)

The control position counters are reset on the next false-to-true rung transition. To enable this mode of operation, reset the inhibit bit (IN=0) by ladder program or manually before program execution.

Entering Parameters

To program these instructions, you need to provide the processor with the following information:

- **Source** - The indexed address of your input file.
- **Reference** - The indexed address of the file that contains the data with which you compare your input file.
- **Result** - The indexed address of the file where the instruction stores the position (bit) number of each detected mismatch.
- **Control** - The control is the address of TWO continuous control structures (i.e. R6:0 and R6:1). The first control structure is a comparison control, which stores status bits, the length of the source and reference files (in bits), and the next bit position during operation. The second control structure is a result control, which stores the bit position number each time the instruction finds a mismatch between source and reference files.

Use the result control address with mnemonic when you address these parameters:

- **Length** (.LEN) is the decimal number of elements in the result file. Make the length long enough to record the maximum number of expected mismatches.
- **Position** (.POS) is the current position in the result file. Enter a value only if you want the instruction to start at an offset concurrent with a control file offset for one scan.

ATTENTION

Do not use the same address for more than one control structure. Duplication of these addresses could result in unpredictable operation, possibly causing equipment damage and/or injury to personnel.

Using Status Bits

To use the FBC or DDT instruction correctly, examine the control bits in both the comparison and result control elements. You address these bits by mnemonic.

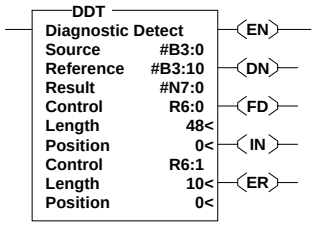
Table 7.7 FBC and DDT Status Bits

Bit:		Function:
Comparison Control Bits	Enable EN (bit 15)	Starts operation on a false-to-true rung transition. If the IN bit is set for one bit-at-a-time operation, the ladder program must toggle the EN bit after the instruction compares each bit.
	Done DN (bit 13)	Is set when the processor reaches the end of the source and reference files.
	Error ER (bit 11)	Is set when the processor detects an error and stops operation of the instruction. For example, an error occurs if the length (LEN) is less than or equal to zero or if the position (POS) is less than zero. The ladder program must reset the ER bit if the instruction detects an error.
	Inhibit IN (bit 09)	Determines the mode of operation. When this bit is reset, the processor detects all mismatches in one scan. When this bit is set, the processor stops the search at each bit and waits for the ladder program to re-enable the instruction before continuing the search.
	Found FD (bit 08)	Is set each time the processor records a mismatch bit number in the result file (one bit-at-a-time operation) or after recording all mismatches (all per scan).
Result Control Bits	Done DN (bit 13)	Is set when the result file fills. The instruction stops and requires another false-to-true rung transition to reset the result DN bit and then continue. If the instruction finds another mismatch, it wraps the new position number around to the beginning of the file, writing over previous position numbers.

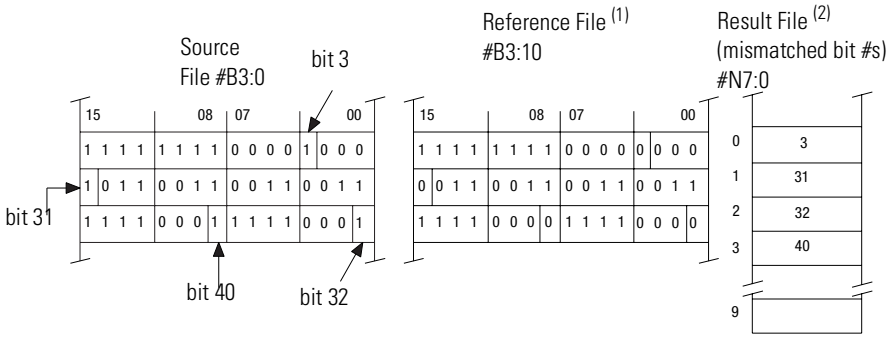
The instruction control bits are reset when the rung's input conditions go false. The instruction control elements reset on a false-to-true rung transition under the following conditions:

1. Compare position equals compare length (clears compare position and result position).

2. Result position equals result length (clears result position).



EXAMPLE The DDT instruction below compares the bits in the source file (B3:0) with the bits in the reference file (B3:10), recording the mismatched bit positions in the result file (N7:0).



The FBC and DDT instructions detect mismatches and record their locations by bit number in a result file.

(1) The DDT instruction changes the status of the corresponding bit in the reference file to match the input file when it detects a mismatch.

(2) The length of the result file is the length that you enter for RESULT CONTROL.

Table 7.8 FBC and DDT Explanation

This Parameter:	Tells the Processor:
Source (B3:0)	Where to find input data for comparison.
Reference (B3:10)	Where to find the reference file.
Results (N7:0)	Where to store mismatched bit numbers.
Compare Control (R6:0)	What control structure controls the comparison.
Length (48)	The number of bits to be compared (2048 max).
Position (0)	To start at the beginning of the file.
Result Control (R6:1)	What control structure controls the result.
Length (10)	The number of words reserved for mismatches (256 max).
Position (0)	To start at the beginning of the file.

Block Transfer Instructions

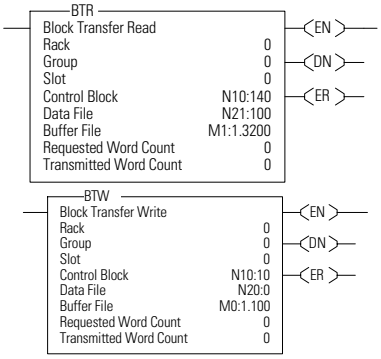
This chapter contains general information about block transfer instructions and explains how they function in your application program. Each of the block transfer instructions includes information on:

- what the instruction symbol looks like
- how to use the instruction

Table 8.1 Block Transfer Instructions

Instruction Mnemonic	Instruction Name	Purpose	Page
BTR	Block Transfer Read	A BTR is used to receive data from a remote device.	8-1
BTW	Block Transfer Write	A BTW is used to send data to a remote device.	8-1

Block Transfer Instructions (BTR and BTW)



Fixed	SLC 5/01	SLC 5/02	SLC 5/03	SLC 5/04	SLC 5/05
			•	•	•

Block transfer instructions are supported by SLC 5/03 (OS302, Series C), SLC 5/04 (OS401, Series C) and SLC 5/05 (OS501, Series C) and higher processors using RSLogix 500 version 4.10 and higher. With block-transfer instructions, you can transfer up to 64 words to or from a remote device over an Allen-Bradley Remote I/O (RIO) link. A Block Transfer Read (BTR) is used to receive data from a remote device. A Block Transfer Write (BTW) is used to send data to a remote device. The RIO Series B scanner (1747-SN) modules and the back-up scanner (1747-BSN) modules perform block transfers via M0 and M1 file buffers.

A false-to-true rung transition initiates a BTW or BTR instruction. The BTW instruction tells the processor to write data stored in the BTW Data File to a device at the specified RIO rack/group/slot address. The BTR instruction tells the processor to read data from a device at the specified RIO rack/group/slot address and store it in the BT Data File. The Data File may be any valid integer, floating point or binary data table file. A total of 32 block transfer buffers are available; you can execute a maximum of 32 different block transfers. Each buffer is made up of 100 consecutive words. The processor runs each block

transfer request in the order it is requested. When the processor changes to Program mode, all pending block transfers are cancelled.

A BTR or BTW instruction writes information into its control structure address (a three-word integer Control Block) when the instruction is entered. The processor uses these values to execute the transfer.

You must enter an M1 file address into BTR Instructions and an M0 file address into BTW Instructions. However, each instruction uses both the M0 and M1 file for that one hundred word buffer (1 through 32). For example, to use the first available buffer (1) for a BTR, enter M1:e.100 into the "Buffer File" field. However, M0:e.100 is also used by this BTR. So, the next BT instruction must use another M-file buffer (2 through 32).

RIO Block Transfer General Functional Overview

The RIO scanner performs block transfers through control/status buffers allocated in the scanner's M0 and M1 files. For BTW's, the data stored in the Data File is copied into the M0 block transfer buffer, the M0 block transfer buffer is then transferred to the RIO device. The corresponding M1 block transfer buffer contains only BTW status information. For BTR's, the M0 block transfer buffer contains only BTR control information. The actual data read from the remote device is received in the scanner's M1 block transfer buffer. This data is then copied into the BTR Data File. A total of 32 block transfer control/status buffers exist in the M0 (output/control) and the M1 (input/status) files.

Entering Parameters for BTR and BTW

The instructions have the following parameters:

- **Data File** - The address in the SLC processor's data file containing the BTW or BTR data. Valid file types are B, N and F.
- **BTR/BTW Buffer File** - Block transfer buffer file address; i.e. M0: e.x00, where "e" is the slot number of the scanner and "x" is the buffer number. The range of the buffer number is from 1 to 32. Each BTR and BTW instruction uses both the M1 and M0 files for a specific buffer number. M0 is used for BTR control and for BTW data. M1 is used for BTW status and BTR data.

TIP

Since buffer number 32 is utilized by the SLC processor for Remote I/O passthru, you should not assign buffer number 32 to a block transfer instruction unless you do not intend to ever use Remote I/O passthru.

- **Control** - The control block is an integer data file address that stores all the block transfer control and status information. The control block is three words in length. Note that these integer file addresses should not be used for any other instructions. You should provide the following information for the control structure:
  - **Rack** - The I/O rack number (0 to 3) of the I/O chassis in which you placed the target I/O module.
  - **Group** - The I/O group number (0 to 7) which specifies the position of the target I/O module in the I/O chassis. When using 1/2-slot addressing, only even group numbers are valid.
  - **Slot** - The slot number (0 or 1) within the group. When using 2-slot addressing, the 0 slot is the low (right) slot and the 1 slot is the high (left) slot within the group. When using 1-slot or 1/2-slot addressing, always select slot 0.
  - **Requested Word Count** - The number of words to transfer. If you set the length to 0, the processor reserves 64 words for block transfer data. The block transfer module transfers the maximum words the adapter can handle. If you set the length from 1 to 64, the processor transfers the number of words specified.

TIP

The three-word control block has the following structure. Before executing a block transfer, the BTR and BTW instructions clear all status bits and initialize word 2 to 0. See Table 8.2, “Control Block Structure,” for more information.

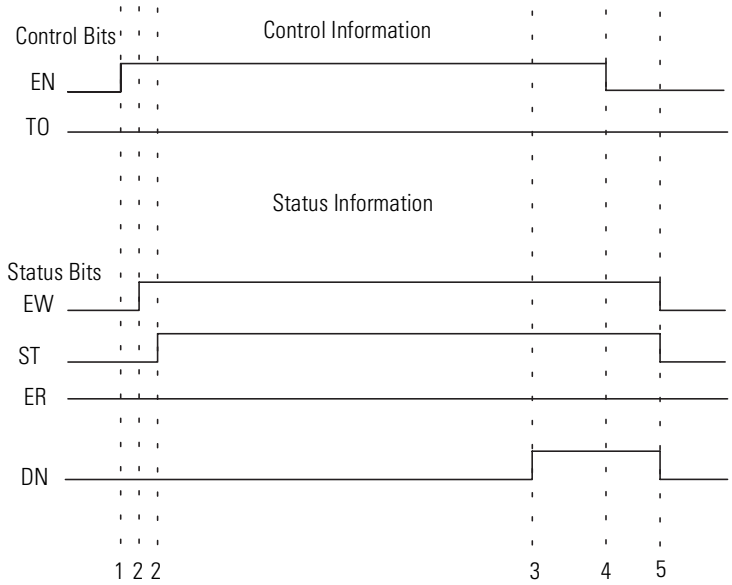
Table 8.2 Control Block Structure

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Word 0	EN	ST	DN	ER		EW		TO	RW	Rack			Group			Slot
Word 1	Requested word count															
Word 2	Transmitted word count/Error code															

Control Status Bits

To use the BTR and BTW instructions correctly, examine the instruction's control and status bits stored in the control structure. These bits are mapped to bits in word 0 of the control block structure.

Figure 8.1 Successful Block Transfer



Successful Block Transfer Read/Write

Figure 8.1 illustrates a successful BT operation.

1. The SLC control program copies new data to the data file (BTW only) and solves the BT rung true, which sets the enable (EN) bit.

2. The scanner detects that the EN bit is set, validates the control block information, puts the BT request on the RIO link successfully, and since no other BTs are pending for the same logical rack, sets the enable waiting (EW) and start (ST) bits.

3. The scanner receives a BT reply (with no errors) from the RIO link device, copies the received data to the data file (BTR only) and sets the done (DN) bit.

4. The SLC control program detects the DN bit, processes the BTR data and solves the BT rung false, which clears the enable (EN) bit.

5. The scanner detects that the SLC control program has completed processing (because the EN bit is clear) and clears the EW, ST and DN bits. At this point, the SLC control program could re-initiate the same BT operation by solving the BT rung true again.

TIP Except for the time-out bit, TO (bit 08), do not modify any controller status bits while the block transfer is in progress.

IMPORTANT The BTR/BTW instruction must be scanned (true or false) in order to update the control and status bits.

TIP In order to conserve scan time, place each block transfer instruction in its own subroutine and only call the subroutine while the block transfer instruction is enabled.

Table 8.3 Control and Status Bit Descriptions

Control/Status Bit	Description
Enable EN (bit 15)	Block Transfer Enabled - (EN = Enabled). The processor sets/resets this bit depending on the rung state (true/false). The processor sends the enable bit to the RIO scanner when the BTR/BTW instruction is scanned. If the BT is not waiting (EW set) and is not started (ST set), and the EN bit sees a false-to-true transition, the RIO scan triggers a BT.
Start ST (bit 14)	Block Transfer Started - (ST = Started). When the instruction is scanned (true or false), the processor reads this bit from the RIO scanner. The scanner sets this bit when the BT starts. The scanner resets this bit when the ladder logic (processor) clears the EN bit indicating the BT is finished.
Done DN (bit 13)	Block Transfer Successful - (DN = Done). When this bit is set, it indicates the successful completion of a block transfer operation. When the instruction is scanned (true or false), the processor reads the DN bit from the RIO scanner. The scanner clears the DN bit when the ladder logic (processor) clears the EN bit.
Error ER (bit 12)	Block Transfer Error - (ER = Error). When this bit is set, it indicates that the process detected a failed block transfer. When the instruction is scanned (true or false), the processor reads the ER bit from the RIO scanner. The scanner clears the ER bit when the ladder logic (processor) clears the EN bit.

Table 8.3 Control and Status Bit Descriptions

Control/Status Bit	Description
Enable-waiting EW (bit 10)	Block Transfer Enabled and waiting for block transfer to start - (EW = Enable Waiting). When the EW bit is set and the ST bit is clear, this indicates that a block transfer operation is pending. When the instruction is scanned (true or false), the processor reads the EW bit from the scanner. The scanner clears the EW bit after the ladder logic (processor) clears the EN bit.
Time Out TO (bit 08)	Block Transfer Time-out (TO = Time-out). You can set this bit to cancel block transfer operation by forcing the BT to time out once the Enabled Waiting (EW) bit sets and before the RIO scanner's internal four-second block transfer timer times out or the block transfer completes. Cancelling a block transfer causes an error (ER) bit to set and an error code of -9 to display in the control structure. Note that the Time-out (TO) bit must be cleared before initiating a new block transfer. The RIO scanner ignores a block transfer request if both TO and EN bits are set at the same time.
Read-Write RW (bit 07)	Block Transfer Type. This bit is controlled by the instruction type. A "0" indicates a write operation (BTW); a "1" indicates a read operation (BTR).

In addition to the control and status bits, the control block contains two other parameters the processor uses to execute the block transfer instructions.

Requested Word Count, Word 1 (RLEN)

This is used to configure BTR/BTW length information (0 to 64). Length is the number of BTR/BTW words read from or written to the RIO device. If RLEN = 0 for a BTW instruction, 64 words are sent. If RLEN = 0 for a BTR instruction, the actual length is determined by the RIO device responding to the block transfer read request.

Transmitted Word Count/Error Code, Word 2 (DLEN)

Transmitted Word Count is the status of the actual number of BTW words sent or the number of BTR words received. The processor uses this number to verify the transfer. This number should match the requested word count (unless the transmitted word count is zero). If these numbers do not match, the processor sets the ER bit (bit 12). If there is an error, the processor gives the error code in Word 2 of the control structure in the form of a negative number. See Table 8.4, "BTR/BTW Error Codes," for a list of error codes. Only one error code is stored at a time (a new error code overwrites the previous error code).

Table 8.4 BTR/BTW Error Codes

Error Code	Description
0	The block transfer completed successfully.
-6	Illegal block transfer length requested.
-7	Block transfer communication error occurred when block transfer request was initiated.

Table 8.4 BTR/BTW Error Codes

Error Code	Description
-8	Error in block transfer protocol.
-9	Block Transfer Time-out - Either the SLC user program cancelled the block transfer or the scanner's block transfer timer timed out. Note that a time-out error occurs if a block transfer is attempted at a location that is not configured for block transfer operation (e.g., requesting a block transfer for a location that is an output module).
-10	No RIO channel configured.
-11	Attempted a block transfer either to a non-configured block transfer device (i.e., an invalid logical rack, group, or slot), or at a complementary device location where there is no corresponding primary image space allocated.
-12	Attempted a block transfer to an inhibited device.

Instruction Operation

1. The scanner processes the BTR/BTW when it detects that the SLC control program rung, which contains the BTR/BTW, goes true.

If the RIO scanner detects any problem at this point (such as invalid block transfer control field, or unconfigured device), the control structure word 2 fills with the error code and the ER bit (bit 12) is set. If no problems occur, the EW bit (bit 10) and ST bit (bit 14) are set in the control block.

TIP

The ST bit is not set if the scanner is already in the process of block transferring data to a location within the same logical RIO rack. The ST bit is set only after any previous pending block transfers to the same logical rack are completed and the block transfer request is scheduled on the RIO link.

The SLC control program can monitor the block transfer by examining bits in word 0 of the control block. They indicate when the scanner has started processing (EW and ST) the block transfer and whether the block transfer operation completed successfully (DN) or failed (ER). The SLC control program can take different actions based on these status bits.

2. When a block transfer completes successfully, the DN bit is set. This indicates that the block transfer control block has been updated with the actual transmitted word count. This is important for BTR instructions, because this indicates the number of valid data words received from the remote device. This data is stored in the BTR data file.

3. If the block transfer fails, the length field and the data file are not updated. The ER bit is set and the error code field indicates the problem.

4. The SLC control program must indicate to the scanner when it is done processing the status word in the control structure (because DN or ER was set) so the corresponding control bits can be reused for another block transfer operation. The SLC control program indicates that it is done processing the block transfer when it solves the BT rung false, which clears the EN bit in the control block.

5. When the RIO scanner detects that the EN bit cleared, it then clears the EW, ST and DN or ER bits, as well as the Transmitted Word Count/Error Code. This ensures that the status bits in the control block are not reflecting the results of the previous block transfer operation.

IMPORTANT To prevent configuration conflicts, it is highly recommended that each M-file buffer (My:e.x00) should be used by only one block transfer instruction.

Programming Examples

Table 8.5 Block Transfer Programming Examples

Figure 8.2, "Directional" on page 8-9
Figure 8.3, "Directional Repeating" on page 8-9
Figure 8.4, "Directional Continuous" on page 8-9
Figure 8.5, "Bi-directional Continuous" on page 8-10
Figure 8.6, "Bi-directional Alternating" on page 8-10
Figure 8.7, "Bi-directional Alternating Repeating" on page 8-10

Figure 8.2 Directional

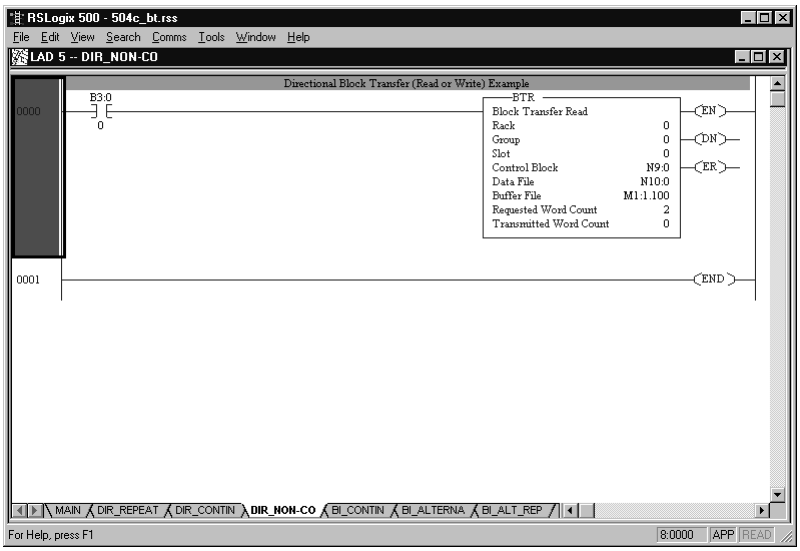


Figure 8.3 Directional Repeating

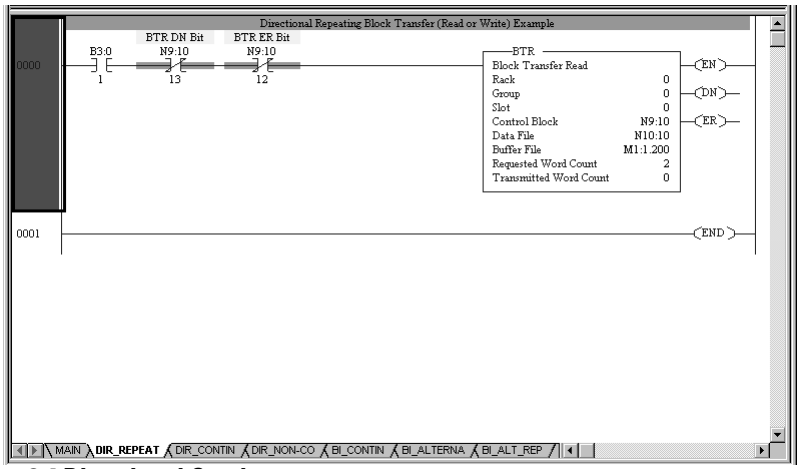


Figure 8.4 Directional Continuous

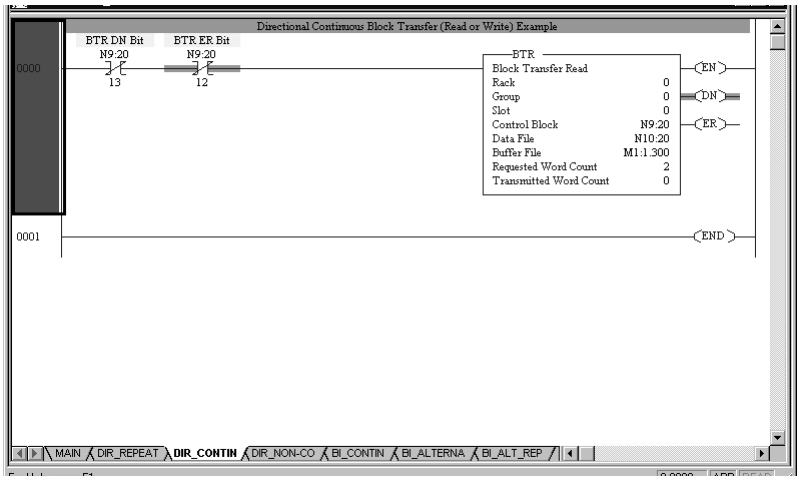


Figure 8.5 Bi-directional Continuous

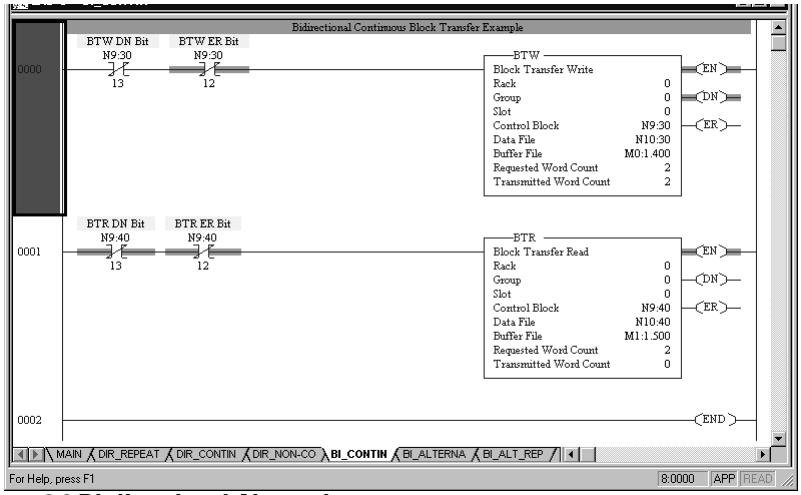


Figure 8.6 Bi-directional Alternating

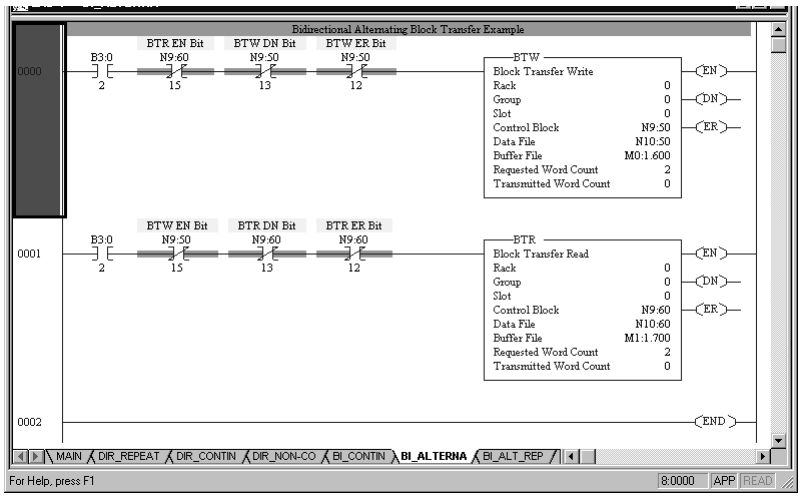
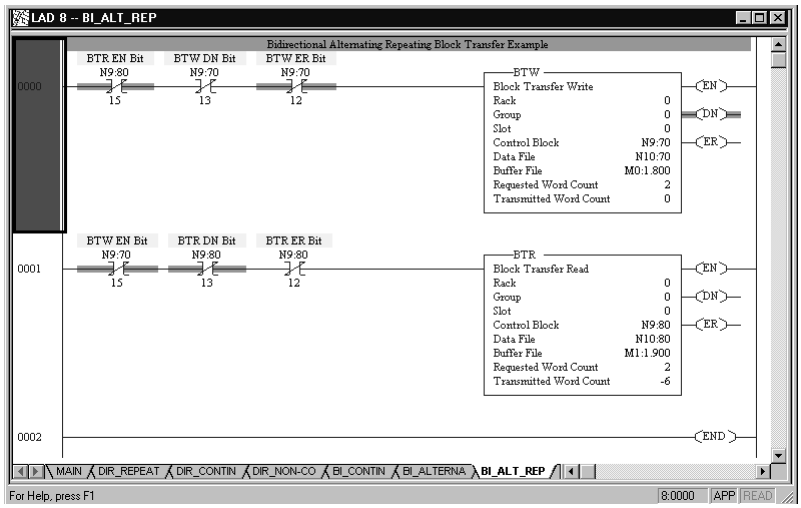


Figure 8.7 Bi-directional Alternating Repeating



Comparison to the PLC-5 BTR and BTW

BTR/BTW in SLC processors are quite similar to the instructions in the PLC-5. However, some differences exist between them, as shown in the table below.

Table 8.6 Block Transfer Comparison

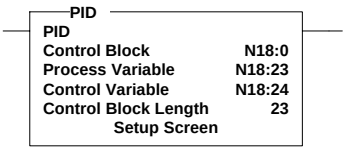
Parameter	SLC	PLC-5
Control Block	3-element integer (N) type	5-element integer (N) type or 1-element block transfer (BT) type.
EN (Enable Bit)	Follows BT rung state.	Gets set when BT rung goes true. Remains set until the BT finishes or fails, and the BT rung goes false.
NR (No Response bit)	None	This bit is in control block word 0 bit 9.
CO (Continuous bit)	None	This bit is in control block word 0 bit 11.
FILE (File Number)	None	This word is control block word 3.
ELEM (Element Number)	None	This word is control block word 4.
Error Codes	7 error codes	11 error codes
BTR/BTW number limitation for one scanner/channel	32	64
BT Status Bits	Can only change when BT rung is scanned.	Can change at any point in the program scan.

IMPORTANT Do not manipulate the I/O image words of the RIO scanner for modules you are block transferring to. These words are used by the RIO scanner and the remote device as block transfer handshake bits. Any manipulation of them by the user program while a block transfer is in progress causes the block transfer to fail.

Proportional Integral Derivative Instruction

This chapter describes the Proportional Integral Derivative (PID) instruction.

Overview



Fixed	SLC 5/01	SLC 5/02	SLC 5/03	SLC 5/04	SLC 5/05
		•	•	•	•

Output Instruction

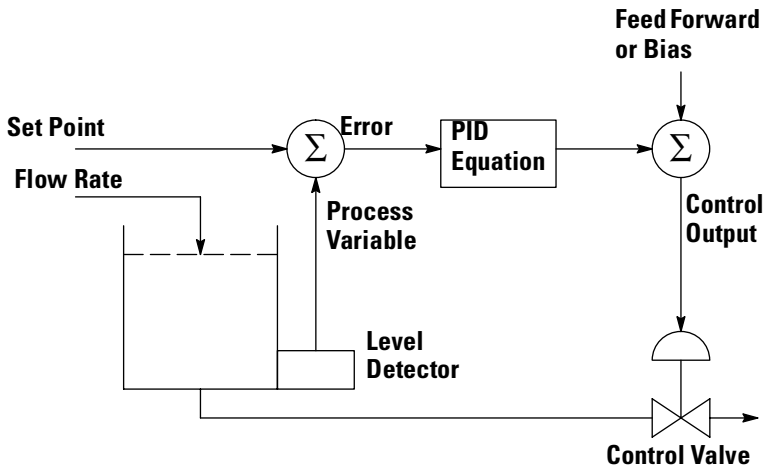
This is an output instruction that controls physical properties such as temperature, pressure, liquid level, or flow rate using process loops.

The PID instruction normally controls a closed loop using inputs from an analog input module and provides an output to an analog output module. For temperature control, you can convert the analog output to a time proportioning on/off output for driving a heater or cooling unit. An example appears on pages 9-26 through 9-27.

The PID instruction can be operated in the timed mode or the STI mode. In the timed mode, the instruction updates its output periodically at a user-selectable rate. In the STI mode, the instruction should be placed in an STI interrupt subroutine. It then updates its output every time the STI subroutine is scanned. The STI time interval and the PID loop update rate must be the same in order for the equation to execute properly.

The PID Concept

PID closed loop control holds a process variable at a desired set point. A flow rate/fluid level example is shown below.



The PID equation controls the process by sending an output signal to the control valve. The greater the error between the setpoint and process variable input, the greater the output signal, and vice versa. An additional value (feed forward/bias) can be added to the control output as an offset. The result of PID calculation (control variable) drives the process variable you are controlling toward the set point.

The PID Equation

The PID instruction uses the following algorithm:

Standard equation with dependent gains:

$$Output = K_C\left[(E) + \frac{1}{T_I}\int (E)dt + T_D \cdot \frac{D(PV)}{df}\right] + \text{Feed Forward/Bias}$$

Standard Gains constants are:

Term	Range (Low to High)	Reference
Controller Gain K_C	0.1 to 25.5 (dimensionless)(1) 0.01 to 327.67 (dimensionless)(2)	Proportional
Reset Term $1/T_I$	25.5 to 0.1 (minutes per repeat)(1) 327.67 to 0.01 (minutes per repeat)(2)	Integral
Rate Term T_D	0.1 to 25.5 (minutes)(1) 0.01 to 327.67 (minutes)(2)	Derivative

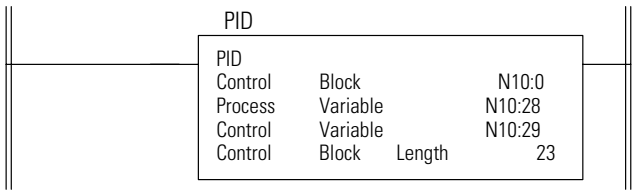
(1) SLC 5/02 processors.

(2) Applies to SLC 5/03 and higher processors PID ranges when bit Reset and Gain Range (RG) bit is set to 1.

The derivative term (rate) provides smoothing by means of a low-pass filter. The cutoff frequency of the filter is 16 times greater than the corner frequency of the derivative term.

The PID Instruction

The figure below shows a PID instruction with typical addresses for these parameters entered:



Place the PID instruction on a rung without any conditional logic. If a PID instruction goes false, the integral term is cleared.

TIP

The PID instruction is an integer - only type of PID algorithm and does not allow you to enter floating point values for any of its parameters. So, if you attempt to move a floating point value to one of the PID parameters using ladder logic, a floating point-to-integer conversion occurs.

During programming, enter the Control Block, Process Variable, and Control Variable addresses after you have placed the PID instruction on a rung.

Entering Parameters

- **Control Block** is a file that stores the data required to operate the instruction. The file length is fixed at 23 words and should be entered as an integer file address. For example, an entry of N10:0 will allocate elements N10:0 through N10:22. The control block layout is shown on page 9-4.

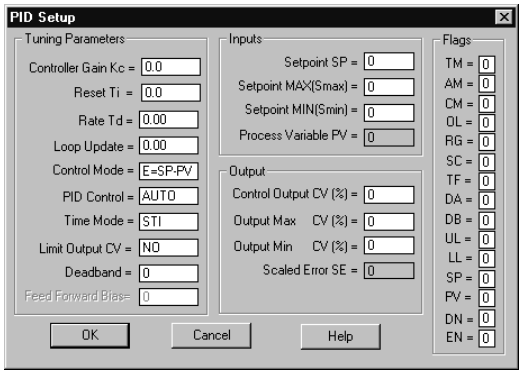
  Do not write to control block addresses with other instructions in your program except as described later in this chapter. If you are re-using a block of data which was previously allocated for some other use, it is good practice to first zero the data.

TIP

We recommend that you use a unique data file to contain your PID control blocks (for example, N10:0). This avoids accidental re-use of the PID control block addresses by other instructions in your program.

- **Process Variable PV** is an element address that stores the process input value. This address can be the location of the analog input word where the value of the input A/D is stored. This value could also be an integer value if you choose to pre-scale your input value to the range 0 to 16383.

- **Control Variable CV** is an element address that stores the output of the PID instruction. The output value ranges from 0 to 16383, with 16383 being the 100% "on" value. This is normally an integer value, so that you can scale the PID output range to the particular analog range your application requires.



PID Control Block Layout

The control block length is fixed at 23 words and should be programmed as an integer file. PID instruction flags (word 0) and other parameters are located as follows:

Table 9.1 Control Block Structure

	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
Word 0	EN		DN	PV	SP	LL	UL	DB	DA(1)	TF	SC	RG(1)	OL(2)	CM(2)	AM(2)	TM(2)
Word 1	PID Sub Error Code (MSbyte)															
Word 2	Setpoint SP															
Word 3	Gain K_c															
Word 4	Reset T_i															
Word 5	Rate T_d															
Word 6	Feed Forward/Bias															
Word 7	Setpoint Max (SMax)															
Word 8	Setpoint Min (SMin)															
Word 9	Deadband															
Word 10	Internal Use Do Not Change															
Word 11	Output Max															
Word 12	Output Min															
Word 13	Loop Update															
Word 14	Scaled Process Variable															
Word 15	Scaled Error SE															

Table 9.1 Control Block Structure

	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
Word 16	Output CV% (0 to 100%)															
Word 17	MSW Integral Sum															
Word 18	LSW Integral Sum															
Word 19	Internal Use Do Not Change															
Word 20	Internal Use Do Not Change															
Word 21	Internal Use Do Not Change															
Word 22	Internal Use Do Not Change															

(1) Applies to the SLC 5/03 and higher processors.

(2) You may alter the state of these values with your ladder program.

ATTENTION

Do not alter the state of any PID control block value unless you fully understand its function and related effect on your process.

Controller Gain (K_c)

Tuning Parameter Descriptions	Address	Data Format	Range	Type	User Program Access
KC - Controller Gain	Word 3	word (INT)	0 to 32,767	control	read/write

Gain K_c (word 3) is the proportional gain, ranging from 0 to 3276.7 (when RG = 0), or 0 to 327.67 (when RG = 1). Set this gain to one-half the value needed to cause the output to oscillate when the reset (T_i) and rate terms (T_d) (below) are set to zero.

TIP

Controller gain is affected by the reset and gain range (RG) bit. For information, see Reset and Gain Enhancement Bit (RG) on page 9-10.

Reset Term (T_i)

Tuning Parameter Descriptions	Address	Data Format	Range	Type	User Program Access
TI - Reset Term - T_i	Word 4	word (INT)	0 to 32,767	control	read/write

Reset Term T_i (word 4) is the Integral gain, ranging from 0 to 3276.7 (when RG = 0), or 327.67 (when RG = 1) minutes per repeat. Set the reset time equal to the natural period measured in the above gain calibration. A K_c value of 1 adds the maximum integral term into the PID equation.

TIP

Reset term is affected by the reset and gain range (RG) bit. For information, see Reset and Gain Enhancement Bit (RG) on page 9-10.

Rate Term (T_d)

Tuning Parameter Descriptions	Address	Data Format	Range	Type	User Program Access
TD - Rate Term - T_d	Word 5	word (INT)	0 to 32,767	control	read/write

Rate T_d (word 5) is the Derivative term. The adjustment range is 0 to 327.67 minutes. Set this value to 1/8 of the integral gain T_i.

TIP

This word is not affected by the reset and gain range (RG) bit. For information, see Reset and Gain Enhancement Bit (RG) on page 9-10.

Feed Forward/Bias

Applications involving transport lags may require that a bias be added to the CV output in anticipation of a disturbance. This bias can be accomplished using the processor by writing a value to the Feed Forward/Bias element, the seventh element (word 6) in the control block file (See page 9-4.) The value you write is added to the output, allowing a feed forward action to take place. You may add a bias by writing a value between -16383 and +16383 to word 6 with your programming terminal or ladder program.

Mode (TM)

Tuning Parameter Descriptions	Address	Data Format	Range	Type	User Program Access
TM - Mode	Word 0, Bit 0	binary	0 or 1	control	read/write

The mode bit specifies when the PID is in timed mode (1) or STI mode (0). This bit can be set or cleared by instructions in your ladder program.

When set for timed mode, the PID executes and updates the CV at the rate specified in the loop update parameter (word 13).

When set for STI mode, the PID executes and updates the CV every time the PID instruction is scanned in the control program. When you select STI, program the PID instruction in the STI interrupt subroutine. The STI routine should have a time interval equal to the setting of the PID "loop update" parameter. Set the STI period in word S:30. For example, if the loop update time contains the value 10 (for 100 ms), then the STI time interval must also equal 100 (for 100 ms).

TIP

When using timed mode, your processor scan time should be at least ten times faster than the loop update time to prevent timing inaccuracies or disturbances.

Loop Update

Tuning Parameter Descriptions	Address	Data Format	Range	Type	User Program Access
Loop Update	Word 13	word (INT)	1 to 1024	control	read/write

The loop update (word 13) is the time interval between PID calculations. The entry is in 0.01 second intervals. Enter a loop update time five to ten times faster than the natural period of the load. The natural period of the load is determined by setting the reset and rate parameters to zero and then increasing the gain until the output begins to oscillate. When in STI mode, this value must equal the STI time interval value loaded in S:30. The valid range is 0.01 to 10.24 seconds. See page 11-13 for help entering STI setpoint.

Deadband

Tuning Parameter Descriptions	Address	Data Format	Range	Type	User Program Access
DB - Deadband	Word 9	word (INT)	0 to 32,767	control	read/write

The deadband extends above and below the setpoint by the value entered. The deadband is entered at the zero crossing of the process variable and the setpoint. This means that the deadband is in effect only after the process variable enters the deadband *and* passes through the setpoint.

The valid range is 0 to the scaled maximum, or 0 to 16,383 when no scaling exists.

Scaled Error

Tuning Parameter Descriptions	Address	Data Format	Range	Type	User Program Access
Scaled Error	Word 15	word (INT)	-32,768 to +32,767	status	read only

Scaled error is the difference between the process variable and the setpoint. The format of the difference (E = SP-PV or E = PV-SP) is determined by the control mode (CM) bit. See Control (CM) on page 9-9.

Auto / Manual (AM)

Tuning Parameter Descriptions	Address	Data Format	Range	Type	User Program Access
AM - Automatic/Manual	Word 0, Bit 1	binary (bit)	0 or 1	control	read/write

The auto/manual bit can be set or cleared by instructions in your ladder program. When off (0), it specifies automatic operation. When on (1), it specifies manual operation. In automatic operation, the instruction controls the control variable (CV). In manual operation, the user/control program controls the CV. During tuning, set this bit to manual.

TIP Output limiting is also applied when in manual.

Control (CM)

Tuning Parameter Descriptions	Address	Data Format	Range	Type	User Program Access
CM - Control Mode	Word 0, Bit 2	binary (bit)	0 or 1	control	read/write

Control mode, or forward-/reverse-acting, toggles the values E=SP-PV and E=PV-SP.

When set (1) - Forward acting (E=PV-SP) causes the control variable to increase when the process variable is greater than the setpoint.

When cleared (0) - Reverse acting (E=SP-PV) causes the control variable to decrease when the process variable is greater than the setpoint.

Deadband (DB)

Tuning Parameter Descriptions	Address	Data Format	Range	Type	User Program Access
DB - PV in Deadband	Word 0, Bit 8	binary (bit)	0 or 1	status	read/write

This bit is set (1) when the process variable is within the zero-crossing deadband range.

Reset and Gain Enhancement Bit (RG)

Tuning Parameter Descriptions	Address	Data Format	Range	Type	User Program Access
RG - Reset and Gain Enhancement	Word 0, Bit 4	binary (bit)	0 or 1	control	read/write

When set (1), the reset and gain range enhancement bit (RG) causes the reset minute/repeat value (TI) and the gain multiplier (KC) to be divided by a factor of 10. That means a reset multiplier of 0.01 and a gain multiplier of 0.01.

When clear (0), this bit allows the reset minutes/repeat value and the gain multiplier value to be evaluated with a reset multiplier of 0.1 and a gain multiplier of 0.1.

Example with the RG bit set: The reset term (TI) of 1 indicates that the integral value of 0.01 minutes/repeat (0.6 seconds/repeat) is applied to the PID integral algorithm. The gain value (KC) of 1 indicates that the error is multiplied by 0.01 and applied to the PID algorithm.

Example with the RG bit clear: The reset term (TI) of 1 indicates that the integral value of 0.1 minutes/repeat (6.0 seconds/repeat) is applied to the PID integral algorithm. The gain value (KC) of 1 indicates that the error is multiplied by 0.1 and applied to the PID algorithm.

TIP The rate multiplier (TD) is not affected by this selection. Valid on SLC 5/03 and higher processors.

Setpoint Scaling (SC)

Tuning Parameter Descriptions	Address	Data Format	Range	Type	User Program Access
SC - Setpoint Scaling	Word 0, Bit 5	binary (bit)	0 or 1	control	read/write

The SC bit is cleared when setpoint scaling values are specified.

Loop Update Time Too Fast (TF)

Tuning Parameter Descriptions	Address	Data Format	Range	Type	User Program Access
TF - Loop Update Too Fast	Word 0, Bit 6	binary (bit)	0 or 1	status	read/write

The TF bit is set by the PID algorithm if the loop update time specified cannot be achieved by the controller due to scan time limitations.

If this bit is set, correct the problem by updating your PID loop at a slower rate or move the PID instruction to an STI interrupt routine. Reset and rate gains will be in error if the instruction operates with this bit set.

Derivative Rate Action Bit (DA)

Tuning Parameter Descriptions	Address	Data Format	Range	Type	User Program Access
DA - Derivative Action Bit	Word 0, Bit 7	binary (bit)	0 or 1	control	read/write

When set (1), the derivative (rate) action (DA) bit causes the derivative (rate) calculation to be evaluated on the error instead of the process variable (PV). When clear (0), this bit allows the derivative (rate) calculation to be evaluated where the derivative is performed on the PV.

Output Alarm Upper Limit (UL)

Tuning Parameter Descriptions	Address	Data Format	Range	Type	User Program Access
UL - Output Alarm Upper Limit	Word 0, Bit 9	binary (bit)	0 or 1	status	read/write

The control variable upper limit alarm bit is set (1) when the calculated CV output exceeds the upper CV limit.

Output Alarm Lower Limit (LL)

Tuning Parameter Descriptions	Address	Data Format	Range	Type	User Program Access
LL - Output Alarm Lower Limit	Word 0, Bit 10	binary (bit)	0 or 1	status	read/write

The control variable lower limit alarm bit is set (1) when the calculated CV output is less than the lower CV limit.

Setpoint Out Of Range (SP)

Tuning Parameter Descriptions	Address	Data Format	Range	Type	User Program Access
SP - Setpoint Out of Range	Word 0, Bit 11	binary (bit)	0 or 1	status	read/write

This bit is set (1) when the setpoint:

- exceeds the maximum scaled value, or
- is less than the minimum scaled value.

PV Out Of Range (PV)

Tuning Parameter Descriptions	Address	Data Format	Range	Type	User Program Access
PV - PV Out of Range	Word 0, Bit 12	binary (bit)	0 or 1	status	read/write

The process variable out of range bit is set (1) when the unscaled process variable (PV):

- exceeds 16,383, or
- is less than zero.

PID Done (DN)

Tuning Parameter Descriptions	Address	Data Format	Range	Type	User Program Access
DN - PID Done	Word 0, Bit 13	binary (bit)	0 or 1	status	read only

The PID done bit is set (1) for one scan when the PID algorithm is computed. It resets automatically.

PID Enable (EN)

Tuning Parameter Descriptions	Address	Data Format	Range	Type	User Program Access
EN - PID Enable	Word 0, Bit 15	binary (bit)	0 or 1	status	read only

The PID enabled bit is set (1) whenever the PID instruction is enabled. It follows the rung state.

Integral Sum

Tuning Parameter Descriptions	Address	Data Format	Range	Type	User Program Access
Integral Sum	Word 17 and 18	long word (32-bit INT)	-2,147,483,648 to 2,147,483,647	status	read/write

This is the result of the integration $\frac{1}{T_I}\int E(dt)$.

Input Parameters

The table below shows the input parameter addresses, data formats, and types of user program access. See the indicated pages for descriptions of each parameter.

Table 9.2 Input Parameters

Input Parameter Descriptions	Address	Range	Type	User Program Access	For More Information
SP - Setpoint	Word 2	0 to 16383[2]	control	read/write	9-14
SPV - Process Variable Scaled	Word 14	0 to 16383	status	read only	9-14
SMAX - Maximum Setpoint [1]	Word 7	0 to 16383[3]	control	read/write	9-15
SMIN - Minimum Setpoint	Word 8	0 to 16383[4]	control	read/write	9-15

(1) The SMAX must be greater than SMIN or the processor will fault with error code.

(2) The range listed in the table is for when scaling is not enabled. With scaling, the range is from minimum scaled (SMIN) to maximum scaled (SMAX).

(3) Maximum SP scaling range: SLC 5/02 is -16382 to +16383; SLC 5/03 and higher is -32767 to +32767.

(4) Minimum SP scaling range: SLC 5/02 is -16383 to +16382; SLC 5/03 and higher is -32768 to +32766.

Setpoint (SP)

Input Parameter Descriptions	Address	Data Format	Range	Type	User Program Access
SP - Setpoint	Word 2	word (INT)	0 to 16383[1]	control	read/write

(1) The range listed in the table is for when scaling is not enabled. With scaling, the range is from minimum scaled (SMIN) to maximum scaled (SMAX).

The SP (Setpoint) is the desired control point of the process variable.

Scaled Process Variable (SPV)

Input Parameter Descriptions	Address	Data Format	Range	Type	User Program Access
SPV - Scaled Process Variable	Word 14	word (INT)	0 to 16383	control	read/write

The SPV (Scaled Process Variable) is the analog input variable. If scaling is enabled, the range is the minimum scaled value (SMIN) to maximum scaled value (SMAX).

If the SPV is configured to be read in engineering units, then this parameter corresponds to the value of the process variable (PV) in engineering units.

Setpoint Maximum Scaled (SMAX)

Input Parameter Descriptions	Address	Data Format	Range	Type	User Program Access
SMAX - Maximum Scaled	Word 7	word (INT)	-32,767 to +32,767(1)	control	read/write

(1) SLC 5/02 valid range is -16382 to +16383.

If the SPV is read in engineering units, then the SMAX (Setpoint Maximum) parameter corresponds to the value of the setpoint in engineering units when the control input is at its maximum value.

Setpoint Minimum Scaled (SMIN)

Input Parameter Descriptions	Address	Data Format	Range	Type	User Program Access
SMIN - Minimum Scaled	Word 8	word (INT)	-32,768 to +32,766(1)	control	read/write

(1) SLC 5/02 valid range is -16383 to +16382.

If the SPV is read in engineering units, then the SMIN (Setpoint Minimum) parameter corresponds to the value of the setpoint in engineering units when the control input is at its minimum value.

TIP

SMIN - SMAX scaling allows you to work in engineering units. The deadband, error, and SPV are also displayed in engineering units. The process variable, PV, must be within the range of 0 to 16383. Use of *SMIN - SMAX* does not minimize PID PV resolution.

Output Parameters

The table below shows the output parameter addresses, data formats, and types of user program access. See the indicated pages for descriptions of each parameter.

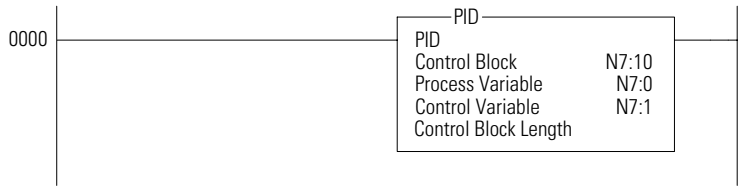
Table 9.3 Output Parameters

Output Parameter Descriptions	Address	Data Format	Range	Type	User Program Access	For More Information
CV - Control Variable	User-defined	word (INT)	0 to 16,383	control	read/write	9-16
CV% - Output CV Percent	Word 16	word (INT)	0 to 100	control	read	9-16
OL - Output Limiting Enable	Word 0, Bit 3	binary	1 = enabled 0 = disabled	control	read/write	9-17
CVH - Output Maximum	Word 11	word (INT)	0 to 100%	control	read/write	9-17
CVL - Output Minimum	Word 12	word (INT)	0 to 100%	control	read/write	9-18

Control Variable (CV)

Output Parameter Descriptions	Address	Data Format	Range	Type	User Program Access
CV - Control Variable	User-defined	word (INT)	0 to 16,383	control	read/write

The CV (Control Variable) is user-defined. See the ladder rung below.



Control Variable Percent (CVP)

Output Parameter Descriptions	Address	Data Format	Range	Type	User Program Access
CVP - Control Variable Percent	User-defined	word (INT)	0 to 16,383	control	read/write

CVP (Control Variable Percent) displays the control variable as a percentage. The range is 0 to 100%. If the AM bit word 0, bit 1 of the PID Control Block is off (automatic mode), this value tracks the control variable (CV) output. Any value written by the programming software is overwritten. If the PID 10:0/AM bit is on (MANUAL mode), this value can be set by the programming software, and the control variable output tracks the control variable percent value.

Output Limiting Enable (OL)

Output Parameter Descriptions	Address	Data Format	Range	Type	User Program Access
OL - Output Limiting Enable	Word 0, Bit 3	binary	1 = enabled 0 = disabled	control	read/write

A value of one enables output limiting to the values defined in Control Variable Maximum (Word 11) and Control Variable Minimum (Word 12).

A value of zero disables OL (Output Limiting).

Output Maximum (CVH)

Output Parameter Descriptions	Address	Data Format	Range	Type	User Program Access
CVH - Output Maximum	Word 11	word (INT)	0 to 100%	control	read/write

When the output limiting bit (OL) word 0, bit 3 of PID Control Block is enabled (1), the CVH (Control Value High) you enter is the maximum output (in percent) that the control variable attains. If the calculated CV exceeds the CVH, the CV is set (overridden) to the CVH value you entered and the upper limit alarm bit (UL) is set.

When the output limiting bit (OL) word 0, bit 3 of PID Control Block is disabled (0), the CVH value you enter determines when the upper limit alarm bit (UL) is set. If CV exceeds the maximum value, the output is not overridden and the upper limit alarm bit (UL) is set.

Output Minimum (CVL)

Output Parameter Descriptions	Address	Data Format	Range	Type	User Program Access
CVL - Output Minimum	Word 12	word (INT)	0 to 100%	control	read/write

When the output limiting bit (OL) word 0, bit 3 of PID Control Block is enabled (1), the CVL (Control Value Low) you enter is the minimum output (in percent) that the Control Variable attains. If the calculated CV is below the minimum value, the CV is set (overridden) to the CVL value you entered and the lower limit alarm bit (LL) is set.

When the output limiting bit (OL) word 0, bit 3 of PID Control Block is disabled (0), the CVL value you enter determines when the lower limit alarm bit (LL) is set. If CV is below the minimum value, the output is not overridden and the lower limit alarm bit (LL) is set.

Runtime Errors

Error code 0036 appears in the status file when a PID instruction runtime error occurs. Code 0036 covers the following PID error conditions, each of which has been assigned a unique single byte code value that appears in the MS byte of the second word of the control block.

Error Code	Description of Error Condition or Conditions		Corrective Action	
11H	*SLC 5/02*	*SLC 5/03 and higher*	*SLC 5/02*	*SLC 5/03 and higher*
	1. Loop update time $\mathbf{D_t} > 255$	1. Loop update time $\mathbf{D_t} > 1024$	1. Change loop update time $\mathbf{D_t}$ to $0 < \mathbf{D_t} \leq 255$	Change loop update time $0 < \mathbf{D_t} \leq 1024$
	2. Loop update time $\mathbf{D_t} = 0$	2. Loop update time $\mathbf{D_t} = 0$		
12H	*SLC 5/02*	*SLC 5/03 and higher*	*SLC 5/02*	*SLC 5/03 and higher*
	1. Proportional gain $\mathbf{K_c} > 255$, or	1. Proportional gain $\mathbf{K_c} < 0$	Change proportional gain $\mathbf{K_c}$ to $0 < \mathbf{K_c} \leq 255$	Change proportional gain $\mathbf{K_c}$ to $\mathbf{K_c} \geq 0$
	2. Proportional gain $\mathbf{K_c} = 0$			
13H	*SLC 5/02*	*SLC 5/03 and higher*	*SLC 5/02*	*SLC 5/03 and higher*
	Integral gain (reset) $\mathbf{T_i} > 255$	Integral gain (reset) $\mathbf{T_i} < 0$	Change integral gain (reset) $\mathbf{T_i}$ to $0 \leq \mathbf{T_i} \leq 255$	Change integral gain (reset) $\mathbf{T_i}$ to $\mathbf{T_i} \geq 0$
14H	*SLC 5/02*	*SLC 5/03 and higher*	*SLC 5/02*	*SLC 5/03 and higher*
	Derivative gain (rate) $\mathbf{T_d} > 255$	Derivative gain (rate) $\mathbf{T_d} < 0$	Change derivative gain (rate) $\mathbf{T_d}$ to $0 \leq \mathbf{T_d} \leq 255$	Change derivative gain (rate) $\mathbf{T_d}$ to $\mathbf{T_d} \geq 0$

Error Code	Description of Error Condition or Conditions		Corrective Action	
21H (SLC 5/02 only)	1. Scaled setpoint max **Smax** > 16383, or 2. Scaled setpoint max **Smax** < -16383		Change scaled setpoint max **Smax** to -16383 ≤ **Smax** ≤ 16383	
22H (SLC 5/02 only)	1. Scaled setpoint min **Smin** > 16383, or 2. Scaled setpoint min **Smin** < -16383		Change scaled setpoint min **Smin** to -16383 ≤ Smin ≤ Smax ≤ 16383	
23H	Scaled setpoint min **Smin** > Scaled setpoint max **Smax**		Change scaled setpoint min **Smin** to -16383 ≤ **Smin** ≤ **Smax** ≤ 16383 (SLC 5/03 and higher -32768 to +32767)	
31H	If you are using setpoint scaling and **Smin** > setpoint **SP** > **Smax**, or If you are not using setpoint scaling and 0 > setpoint **SP** > 16383, then during the initial execution of the PID loop, this error occurs and bit 11 of word 0 of the control block is set. However, during subsequent execution of the PID loop if an invalid loop setpoint is entered, the PID loop continues to execute using the old setpoint, and bit 11 of word 0 of the control block is set.		If you are using setpoint scaling, then change the setpoint **SP** to **Smin** ≤ **SP** ≤ **Smax**, or If you are not using setpoint scaling, then change the setpoint **SP** to 0 ≤ **SP** ≤ 16383.	
41H	Scaling Selected	Scaling Deselected	Scaling Selected	Scaling Deselected
	1. **Deadband** < 0, or	1. **Deadband** < 0, or	Change **deadband** to 0 ≤ **deadband** ≤ **(Smax - Smin)** ≤ 16383	Change **deadband** to 0 ≤ **deadband** ≤ 16383
	2. **Deadband** > **(Smax - Smin)**, or	2. **Deadband** > 16383		
	3. **Deadband** > 16383 (5/02 specific)			
51H	1. **Output high limit** < 0, or 2. **Output high limit** > 100		Change **output high limit** to 0 ≤ **output high limit** ≤ 100	
52H	1. **Output low limit** < 0, or 2. **Output low limit** > 100		Change **output low limit** to 0 ≤ **output low limit** ≤ **output high limit** ≤ 100	
53H	**Output low limit** > **output high limit**		Change **output low limit** to 0 ≤ **output low limit** ≤ **output high limit** ≤ 100	
60H	*SLC 5/02* - PID is being entered for the second time. (PID loop was interrupted by an I/O interrupt, which is then interrupted by the PID STI interrupt.		You have at least three PID loops in your program: One in the main program or subroutine file, one in an I/O interrupt file, and one in the STI subroutine file. You must alter your ladder program and eliminate the potential nesting of PID loops.	

PID and Analog I/O Scaling

For the SLC 500 PID instruction, the numerical scale for both the process variable (PV) and the control variable (CV) is 0 to 16383. To use engineering units, such as PSI or degrees, you must first scale your analog I/O ranges within the above numerical scale. To do this, use the Scale (SCL) instruction and follow the steps described below.

1. Scale your analog input by calculating the slope (or rate) of the analog input range to the PV range (0 to 16383.) For example, an analog input with a range of 4 to 20mA has a decimal range of 3277 to 16384. The decimal range must be scaled across the range of 0 to 16383 for use as PV.

2. Scale the CV to span evenly across your analog output range. For example, an analog output which is scaled at 4 to 20mA has a decimal range of 6242 to 31208. In this case, 0 to 16383 must be scaled across the range of 6242 to 31208.

   Once you have scaled your analog I/O ranges to/from the PID instruction, you can enter the minimum and maximum engineering units that apply to your application. For example, if the 4 to 20mA analog input range represents 0 to 300 PSI, you can enter 0 and 300 as the minimum (Smin) and maximum (Smax) parameters respectively. The Process Variable, Error, Setpoint, and Deadband are displayed in engineering units in the PID Data Monitor screen. Setpoint and Deadband can be entered into the PID instruction using engineering units.

The following equations show the linear relationship between the input value and the resulting scaled value.

Scaled value = *(input value x slope) +offset*

Slope = *(scaled MAX. –scaled MIN.) / (input MAX. –input MIN)*

Offset = *scaled MIN. –(input MIN. x slope)*

Using the SCL Instruction

Use the following values in an SCL instruction to scale common analog input ranges to PID process variables

Parameter	4 to 20mA	0 to 5V	0 to 10V
Rate/10,000	12,499	10,000	5,000
Offset	-4096	0	0

Use the following values in an SCL instruction to scale control variables to common analog outputs.

Parameter	4 to 20mA	0 to 5V	0 to 10V
Rate/10,000	15,239	10,000	19,999
Offset	6242	0	0

Using the SCP Instruction

Use the following values in an SCP instruction to scale your analog inputs to the PV range and scale the CV range to your analog output

.

Parameter	4 to 20mA	0 to 5V	0 to 10V
Input minimum	3277	0	0
Input maximum	16384	16384	32767
Scaled minimum	0	0	0
Scaled maximum	16383	16383	16383

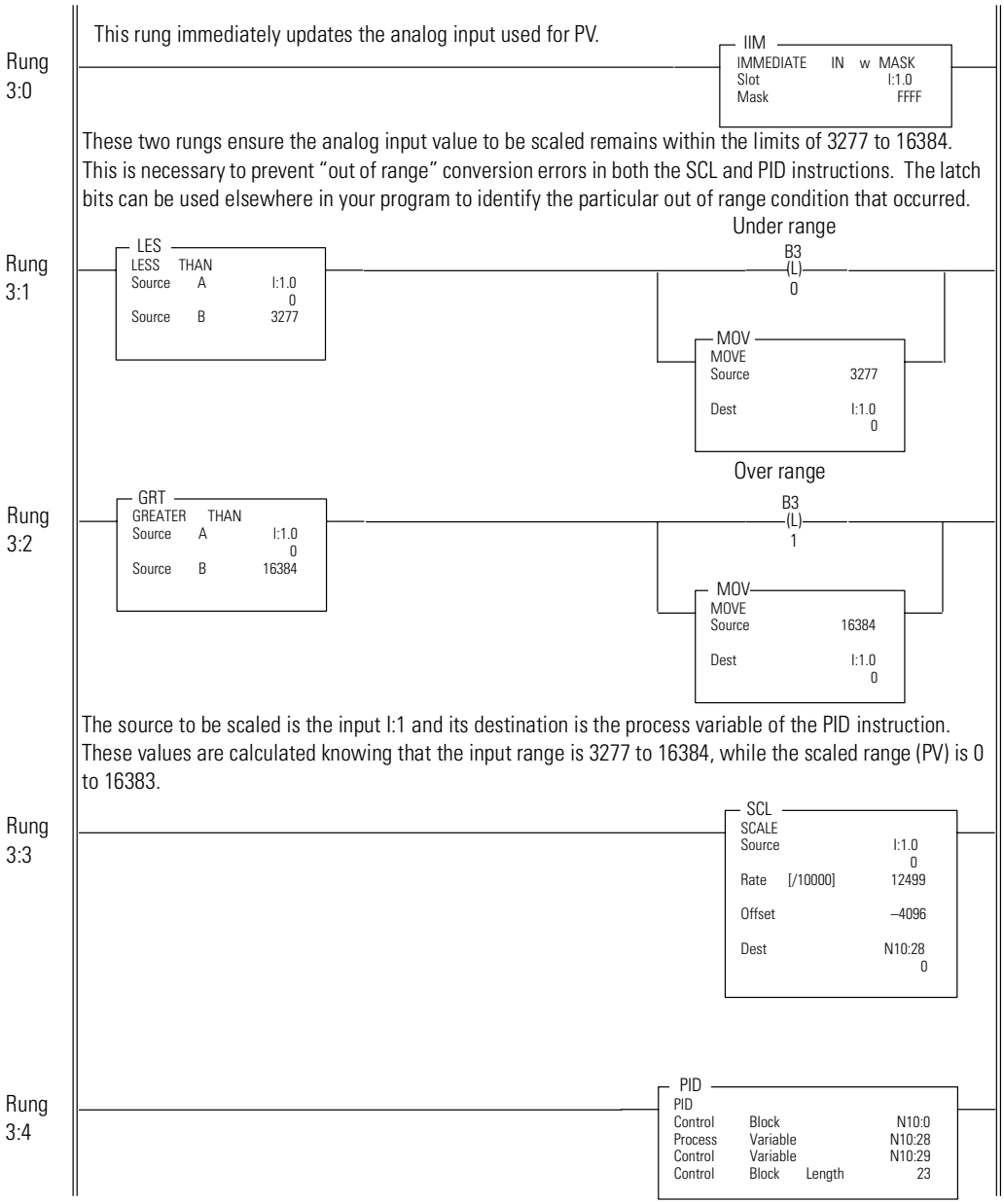
Use the following values in an SCP instruction to scale control variables to common analog outputs.

Parameter	4 to 20mA	0 to 5V	0 to 10V
Input minimum	0	0	0
Input maximum	16383	16383	16383
Scaled minimum	6242	0	0
Scaled maximum	31208	16384	32764

Example

The following ladder diagram shows a typical PID loop that is programmed in the STI mode. This example is provided primarily to show the proper scaling techniques. It shows a 4 to 20mA analog input and a 4 to 20mA analog output. The following parameters are used:

- STI subroutine file (S:31) = 3
- STI Setpoint (S:30) = 10
- STI Enabled bit (S:2/1) = 1



The STI routine should have a time interval equal to the setting of the PID "loop update" parameter.

Application Notes

The following paragraphs discuss:

- Input/Output Ranges
- Scaling to Engineering Units
- Output Alarms
- Output Limiting with Anti-reset Windup
- The Manual Mode
- PID Rungstate
- Time Proportioning Outputs

Input/Output Ranges

The input module measuring the process variable (PV) must have a full scale binary range of 0 to 16383. If this value is less than 0 (bit 15 set), then a value of zero is used for PV and the "Process var out of range" bit is set (bit 12 of word 0 in the control block). If the process variable is >16383 (bit 14 set), then a value of 16383 is used for PV and the "Process var out of range" bit is set.

The Control Variable, calculated by the PID instruction, has the same range of 0 to 16383. The Control Output (word 16 of the control block) has the range of 0 to 100%. You can set lower and upper limits for the instruction's calculated output values (where an upper limit of 100% corresponds to a Control Variable limit of 16383).

Scaling to Engineering Units

Scaling lets you enter the setpoint and zero-crossing deadband values in engineering units, and display the process variable and error values in the same engineering units. Remember, the process variable PV must still be within the range 0-16383. The PV is displayed in engineering units, however.

Select scaling as follows:

1. Enter the maximum and minimum scaling values Smax and Smin in the PID control block. Refer to the control block of the PID instruction on page 9-4. The Smin value corresponds to an analog value of zero for the lowest reading of the process variable, and Smax corresponds to an analog value of 16383 for the highest reading. These values reflect the process limits. Setpoint scaling is selected by entering a non-zero value for one or both parameters. If you enter the same value for both parameters, setpoint scaling is disabled.

For example, if measuring a full scale temperature range of -270°C (PV=0) to +1000°C (PV=16383), enter a value of -270 for Smin and 1000 for Smax. Remember that inputs to the PID instruction must be 0 to 16383. Signal conversions could be as follows:

Parameter	Range
Process limits	–270 to +1000° C
Transmitter output (if used)	+4 to +20 mA
Output of analog input module	0 to 16383
PID instruction, Smin to Smax	–270 to +1000° C

2. Enter the setpoint (word 2) and deadband (word 9) in the same scaled engineering units. Read the scaled process variable and scaled error in these units as well. The control output percentage (word 16) is displayed as a percentage of the 0 to 16383 CV range. The actual value transferred to the CV output is always between 0 and 16383.

When you select scaling, the instruction scales the setpoint, deadband, process variable, and error. You must consider the effect on all these variables when you change scaling.

Output Alarms

You may set an output alarm on the control output (CO) at a selected value above and/or below a selected output percent. When the instruction detects that the output (CO) has exceeded either value, it sets an alarm bit (bit 10 for lower limit, bit 9 for upper limit) in word 0 of the PID control block. Alarm bits are reset by the instruction when the output (CO) comes back inside the limits. The instruction does not prevent the output (CO) from exceeding the alarm values unless you select output limiting.

Select upper and lower output alarms by entering a value for the upper alarm (word 11) and lower alarm (word 12). Alarm values are specified as a percentage of the output. If you do not want alarms, enter zero and 100% respectively for lower and upper alarm values and ignore the alarm bits.

Output Limiting with Anti-Reset Windup

You may set an output limit (percent of output) on the control output. When the instruction detects that the output (CO) has exceeded a limit, it sets an alarm bit (bit 10 for lower limit, bit 9 for upper limit) in word 0 of the PID control block, and prevents the output (CO) from exceeding either limit value. The instruction limits the output (CO) to 0 and 100% if you choose not to limit.

Select upper and lower output limits by setting the limit enable bit (bit 3 of control word 0), and entering an upper limit (word 11) and lower limit (word 12). Limit values are a percentage (0 to 100%) of the control output (CO).

The difference between selecting output alarms and output limits is that you must select output limiting to enable limiting. Limit and alarm values are stored in the same words. Entering these values enables the alarms, but not limiting. Entering these values and setting the limit enable bit enables limiting and alarms.

Anti-reset windup is a feature that prevents the integral term from becoming excessive when the output (CO) reaches a limit. When the sum of the PID and bias terms in the output (CO) reaches the limit, the instruction stops calculating the integral sum until the output (CO) comes back in range. The integral sum is contained in words 17 and 18 of the control block.

The Manual Mode

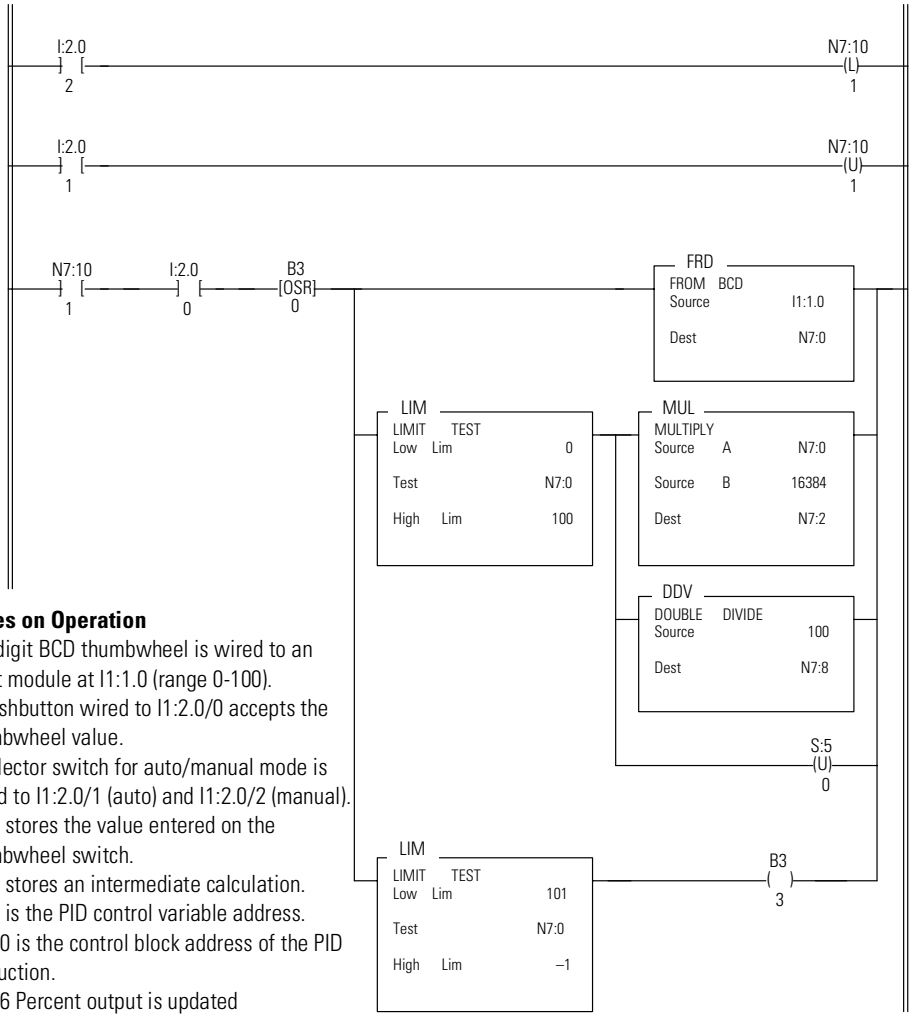
In the manual mode, the PID algorithm does not compute the value of the control variable. Rather, it uses the value as an input to adjust the integral sum (words 17 and 18) so that a bumpless transfer takes place upon re-entering the AUTO mode.

To set the manual output level, design your ladder program to write to the CV address when in the manual mode. Note that this number is in the range of 0 to 16383, not 0 to 100. Writing to the CV percent (word 16) with your ladder program has no effect in the manual mode but adversely effects bumpless transfer.

The example on the next page shows how you can manually control the control variable (CV) output with your ladder program.

PID Rungstate

If the PID rung is false, the integral sum (words 17 and 18) is cleared and CV remains in its last state.

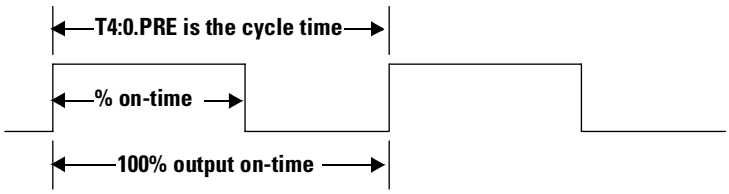


Notes on Operation

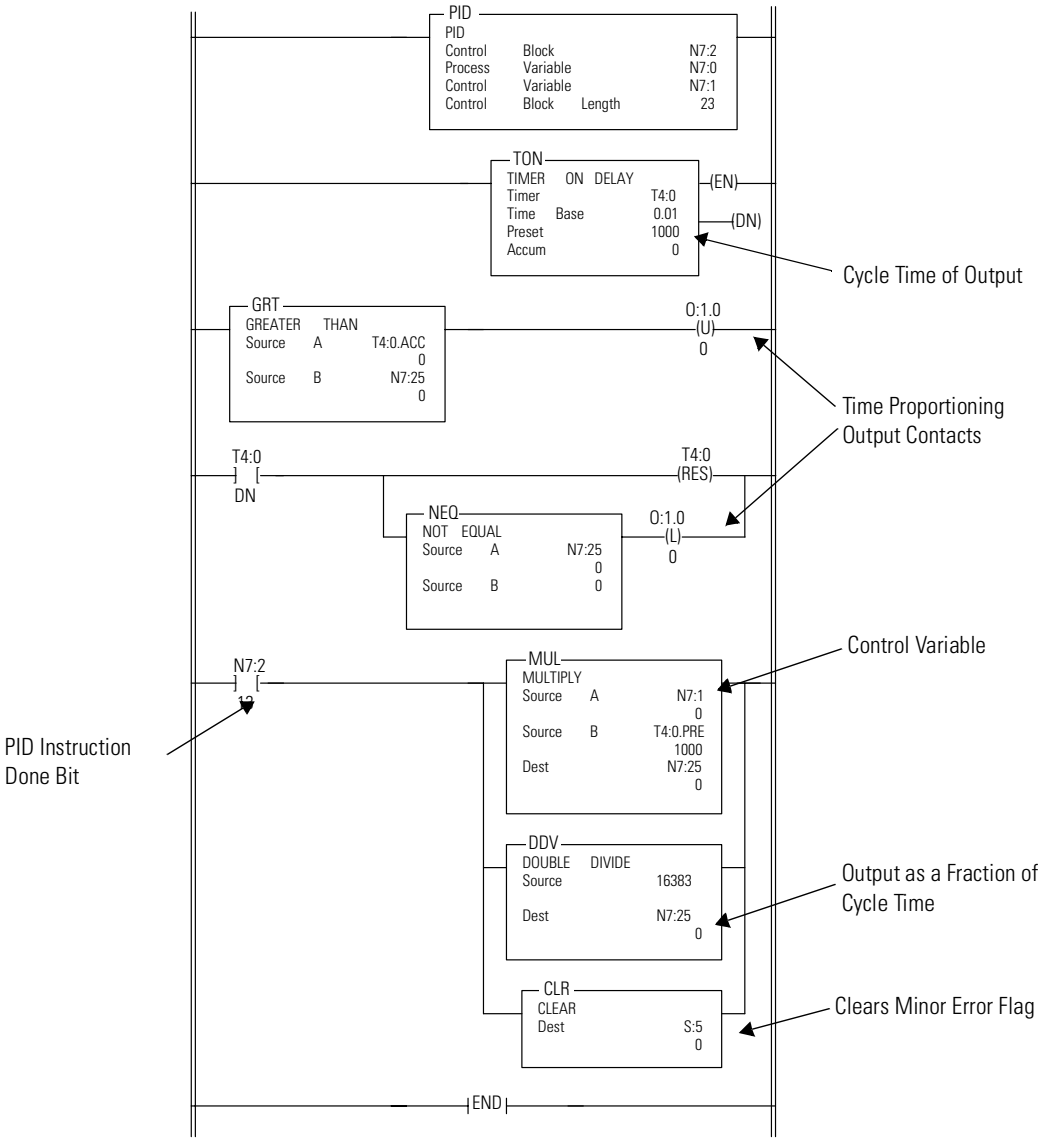
A 3-digit BCD thumbwheel is wired to an input module at I1:1.0 (range 0-100).
A pushbutton wired to I1:2.0/0 accepts the thumbwheel value.
A selector switch for auto/manual mode is wired to I1:2.0/1 (auto) and I1:2.0/2 (manual).
N7:0 stores the value entered on the thumbwheel switch.
N7:2 stores an intermediate calculation.
N7:8 is the PID control variable address.
N7:10 is the control block address of the PID instruction.
N7:26 Percent output is updated automatically by the PID instruction.

Time Proportioning Outputs

For heating or cooling applications, the Control Variable analog output is typically converted to a time-proportioning output. While this cannot be done directly with the processor, you can use the program on the following page to convert the Control Variable to a time proportioning output. In this program, cycle time is the preset of timer T4:0. Cycle time relates to% on-time as follows:



Example - Time proportioning outputs



ASCII Instructions

This chapter contains general information about the ASCII instructions and explains how they function in your application program. Each of the instructions includes information on:

- what the instruction symbol looks like
- how to use the instruction

Table 10.1 ASCII Instruction

Instruction Mnemonic	Instruction Name	Purpose	Page
ABL	Test Buffer for Line	Determine the number of characters in the buffer, up to and including the user configured end of line character(s).	**10-6**
ACB	Number of Characters in Buffer	Determine the total number of characters in the buffer.	**10-7**
ACI	String to Integer	Convert a string to an integer value.	**10-8**
ACL	ASCII Clear Receive and/or Send Buffer	Clear the receive and/or transmit buffers.	**10-9**
ACN	String Concatenate	Link two strings into one.	**10-10**
AEX	String Extract	Extract a portion of a string to create a new string.	**10-10**
AHL	ASCII Handshake Lines	Set or reset modem handshake lines.	**10-11**
AIC	Integer to String	Convert an integer value to a string.	**10-13**
ARD	ASCII Read Characters	Read characters from the input buffer and place them into a string.	**10-13**
ARL	ASCII Read Line	Read one line of characters from the input buffer and place them into a string.	**10-16**
ASC	String Search	Search a string.	**10-17**
ASR	ASCII String Compare	Compare two strings.	**10-18**
AWA	ASCII Write with Append	Write a string with user-configured character(s) appended.	**10-19**
AWT	ASCII Write	Write a string.	**10-21**

ASCII Instruction Overview

ASCII instructions are available in SLC 5/03 OS301 and above processors, and all SLC 5/04 and SLC 5/05 processors. There are two types of ASCII instructions:

- ASCII port control - these include instructions that use or alter the communication channel for receiving or transmitting data. When using these instructions, the system configuration must be set to "User Mode."

  (ABL, ACB, ACL*, AHL*, ARD, ARL, AWA*, AWT*)
  *may be in either user or system mode

  ASCII port control instructions are queued in the order that they are executed and are dependent on one another to execute (except ACL which executes immediately). For example, if you have an ARD (ASCII Read instruction) and then an AWT (ASCII Write instruction), the Done bit or the Error bit of the ARD must be set before the AWT can begin executing (even if the AWT was enabled while the processor was executing the ARD). A second ASCII port control instruction cannot begin executing until the first has completed. However, the processor does not wait for an ASCII port control instruction to complete before continuing to execute your ladder program.

- ASCII string control - these include instructions that manipulate string data. (ACI, ACN, AEX, AIC, ASC, ASR)

  ASCII string control instructions execute immediately. They are never sent to the queue to wait their turn for execution.

Protocol Parameter Overview

Listed below are the ASCII protocol parameters that you set via the Channel 0 configuration screens in your programming software.

Description	Specification
Baud Rate	Toggles between 110, 300, 600, 1.2K, 2.4K, 4.8K, 9.6K, and 19.2K (additional rate of 38.4K for SLC 5/04 and SLC 5/05 only). The default is 19.2K.
Start Bits	The default is 1 and cannot be changed.
Stop Bits	Options include 1, 1.5, and 2 The default is 1.
Parity	Toggles between None, Odd, and Even. The default is None.

Description	Specification
Data Bits	Toggles between 7 and 8. The default is 8.
Termination Characters	Allows you to configure up to 2 ASCII characters. The default is CR.
Append Characters	Allows you to configure up to 2 ASCII characters. The AWA instruction adds the characters to the end of every string to serve as termination characters for the receiving device. The default is CR LF.

Using the ASCII Data File Type

These are 1-word elements. Assign ASCII addresses as follows:

Table 10.2 Addressing Format

Format	Explanation		
Af:e/b	**A**	ASCII file	
	f	File number. A file number between 9 to 255 can be used.	
	:	Element delimiter	
	e	Element number	Ranges from 0 to 255. This is a 1-word element.
	/	Bit delimiter	
	b	Bit number	Bit location within the element. Ranges from 0 to 15.
Examples		A9:2	Element 2, ASCII file 9
		A10:0/7	Bit 7, Element 0, ASCII file 10

Using the String (ST) Data File Type

This file type is valid for SLC 5/03 OS301 and higher, SLC 5/04, and SLC 5/05 processors. These are 42-word elements. You can address string lengths by adding a .LEN to any string address (for example, ST17:1.LEN). Valid string data file numbers are 9 to 255.

String lengths must be between 0 and 82 bytes. In general, lengths that are outside of this range cause the processor to set the ASCII Error bit (S:5/15) and the instruction is not executed.

TIP

You configure append or end-of-line characters via the Channel Configuration screen. The default append characters are carriage return and line feed; the default end-of-line (termination) character is a carriage return.

All instructions except ACL and AHL will error if the port is disabled.

Assign string addresses as follows:

Format	Explanation		
STf:e.s/b	**ST**	String file	
	f	File number. A file number between 9 to 255 can be used.	
	:	Element delimiter	
	e	Element number	Ranges from 0 to 255. These are 42-word elements.
	.	Subelement delimiter	
	s	Subelement number	Ranges from 0 to 41. Word 0 is the length,.LEN. Words 1 to 41 are entered in as .DATA[0] to .DATA[40}
	/	Bit delimiter	
	b	Bit number	Bit location within the element. Ranges from 0 to 15.
Examples		ST9:2	Element 2, string file 9
		ST9:2.LEN	Length, in bytes of element 2, string file 9
		ST9:2.DATA[0]	Word 1 of element 2, string file 9
		ST9:2.DATA[0]/15	Bit 15 of word 1 of element 2, string file 9

Entering Parameters

The control element for ASCII instructions includes eight status bits, an error code byte, and two character words:

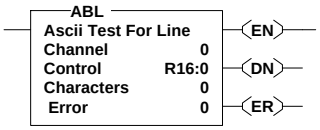
Table 10.3 ASCII Control Element

	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
Word 0	EN	EU	DN	EM	ER	UL	IN	FD	Error Code							
Word 1	Number of Characters For Sending or Receiving (LEN)															
Word 2	Number of Characters Sent or Received (POS)															

EN = Enable Bit
EU = Queue Bit
DN = Asynchronous Done Bit
EM = Synchronous Done Bit
ER = Error Bit
UL = Unload Bit
IN = Running Bit (This is the IN Bit in the control data file [R6:].)
FD = Found Bit

- Found Bit FD (bit 8) indicates that the instruction found the end of characters or termination characters in the buffer (applies to ABL and ACB instructions)
- Running Bit IN (bit 9) indicates that a queued instruction is executing.
- Unload Bit UL (bit 10) ceases instruction operation before (may be queued) or during execution. If this bit is set while an instruction is executing, any data already processed is sent to the destination. Note that the instruction is not removed from the queue; any remaining data is just not processed. You set this bit.
- Error Bit ER (bit 11) indicates that an error occurred while executing the instruction, such as a mode change via channel 1, or the instruction was cancelled using the UL bit or ACL instruction.
- Synchronous Done Bit EM (bit 12) is set concurrently to a program scan to indicate the completion of an ASCII instruction.
- Asynchronous Done Bit DN (bit 13) is set opposite to a program scan when an instruction successfully completes its operation. Note that an instruction can take longer than one program scan to finish executing.
- Queue Bit EU (bit 14) indicates that an ASCII instruction was placed in the ASCII queue. This action is delayed if the queue is already filled. The queue may contain up to 16 instructions.
- Enable Bit EN (bit 15) indicates that an instruction is enabled due to a false-to-true transition. This bit remains set until the instruction has completed executing or errors.

Test Buffer for Line (ABL)



Fixed	SLC 5/01	SLC 5/02	SLC 5/03	SLC 5/04	SLC 5/05
			•	•	•

Output Instruction

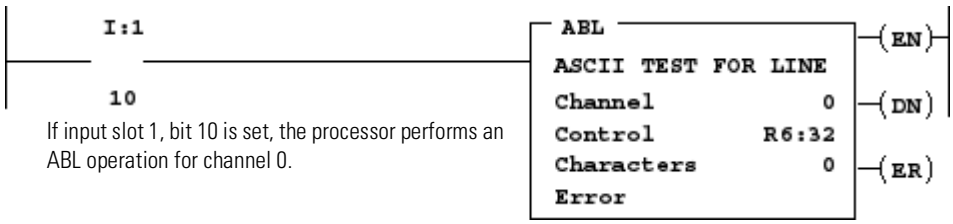
Use the ABL instruction to determine the total number of characters in the input buffer, up to and including the end-of-line characters (termination). This instruction looks for two termination characters that you configure via the ASCII port configuration screen. On a false-to-true transition, the processor reports the number of characters in the POS field of the ASCII control block. The serial port must be configured for User mode.

Entering Parameters

Enter the following parameters when programming this instruction:

- **Channel** is the number of the RS-232 port (Channel 0).
- **Control** is the area that stores the control register required to operate the instruction.
- **Characters** are the number of characters in the buffer that the processor finds (0-1024). This parameter is display only and resides in word 2 of the control block.
- **Error** displays the hexadecimal error code that indicates why the ER bit was set in the control data file (R6:). See page 10-23 error code descriptions.

Example



If input slot 1, bit 10 is set, the processor performs an ABL operation for channel 0.

When the rung goes from false-to-true, the Enable bit (EN) is set. The instruction is put in the ASCII instruction queue, the Queue bit (EU) is set, and program scan continues. The instruction is then executed outside of the program scan. However, if the queue is empty the instruction executes immediately. Upon execution, the Running bit (IN) is set.

The processor determines the number of characters (up to and including the end-of-line/termination characters) and puts this value in the position field. The Done bit (DN) is then set.

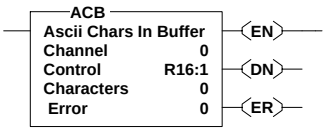
If a zero appears in the POS field, no end-of-line/termination characters were found. The Found bit (FD) is set if the position field was set to a non-zero value.

When the program scans the instruction and finds the Done bit (DN) set, the processor then sets the Synchronous Done bit (EM). The EM bit acts as a secondary done bit corresponding to the program scan.

The Error bit (ER) is set during the execution of the instruction if:

- the instruction is aborted - serial port not in User mode
- the instruction is aborted due to channel mode change
- the Unload bit (UL) is set and the instruction is not executed

Number of Characters In Buffer (ACB)



Fixed	SLC 5/01	SLC 5/02	SLC 5/03	SLC 5/04	SLC 5/05
			•	•	•

Output Instruction

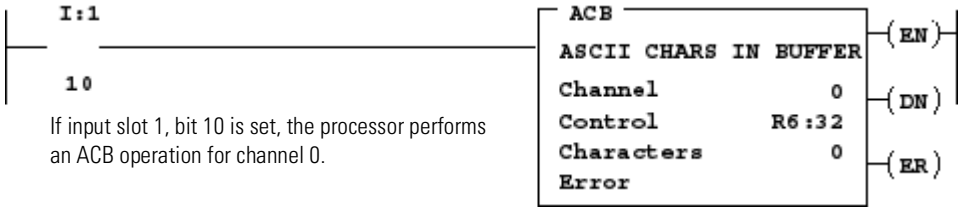
Use the ACB instruction to determine the total characters in the buffer. On a false-to-true transition, the processor determines the total number of characters and records it in the position field of the ASCII control block. The serial port must be in User mode.

Entering Parameters

Enter the following parameters when programming this instruction:

- **Channel** is the number of the RS-232 port (Channel 0).
- **Control** is the area that stores the control register required to operate the instruction.
- **Characters** are the number of characters in the buffer that the processor finds (0-1024). This parameter is display only.
- **Error** displays the hexadecimal error code that indicates why the ER bit was set in the control data file (R6:). See page 10-23 for error descriptions.

Example



If input slot 1, bit 10 is set, the processor performs an ACB operation for channel 0.

When the rung goes from false-to-true, the Enable bit (EN) is set. When the instruction is placed in the ASCII queue, the Queue bit (EU) is set. The Running bit (IN) is set when the instruction is executing. The Done bit (DN) is set upon completion of the instruction.

The processor determines the number of characters in the buffer and puts this value in the position field of the control block. The Done bit

(DN) is then set. If a zero appears in the characters field, no characters were found.

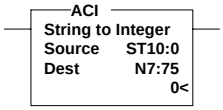
When the program scans the instruction and finds the Done bit (DN) set, the processor then sets the Synchronous bit (EM). The EM bit acts as a secondary done bit corresponding to the program scan.

The Error bit (ER) is set during the execution of the instruction if:

- the instruction is aborted - serial port not in User mode
- the instruction is aborted due to channel mode change
- the Unload bit (UL) is set and the instruction is not executed

String to Integer (ACI)

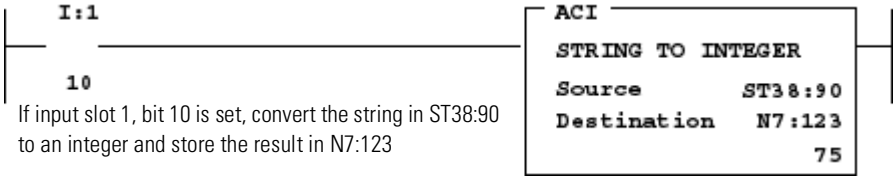
Use the ACI instruction to convert a numeric ASCII string to an integer value between -32,768 and 32,767.



Fixed	SLC 5/01	SLC 5/02	SLC 5/03	SLC 5/04	SLC 5/05
			•	•	•

Output Instruction

Example



If input slot 1, bit 10 is set, convert the string in ST38:90 to an integer and store the result in N7:123

The processor searches the source (file type ST) for the first character between 0 and 9. All numeric characters are extracted until a non-numeric character or the end of the string is reached. Action is taken *only* if numeric characters are found. If the string contains an invalid length (<0 or >82) the ASCII Error bit S:5/15 is set. Commas and signs (+, -) are allowed in the string. However, only the minus sign is displayed in the data table.

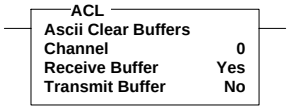
The extracted numeric string is then converted to an integer. The ASCII Error bit S:5/15 is set if the string contains an invalid string length. The value of 32,767 is returned as the result.

This instruction also sets the arithmetic flags (found in word 0, bits 0-3 in the processor status file S:0):

With this Bit:		The Processor:
S:0/0	Carry (C)	is reserved.
S:0/1	Overflow (V)	sets if the integer value is outside of the valid range.
S:0/2	Zero (Z)	sets if the integer value is zero.
S:0/3	Sign (S)	sets if the result is negative.

ASCII Clear Receive and/or Send Buffer (ACL)

Use this instruction to clear an ASCII buffer. ASCII instructions are removed from the queue and then the Error bit (ER) is set. This instruction executes immediately upon the rung transitioning to a true state. The instruction works when the channel is in User Mode or System Mode. In System Mode, only clearing the send buffer will operate and then only if DF1 is selected as the System Mode protocol.



Fixed	SLC 5/01	SLC 5/02	SLC 5/03	SLC 5/04	SLC 5/05
			•	•	•

Output Instruction

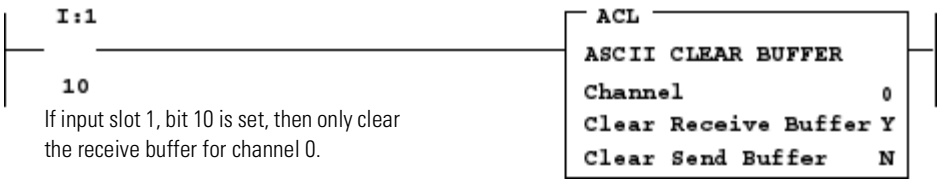
Entering Parameters

Enter the following parameters when programming this instruction:

- **Channel** is the number of the RS-232 port (Channel 0).
- **Clear Receive Buffer** clears the receive buffer and removes the ARD and ARL instructions from the queue. The Error bit (ER) is set in each of these instructions.
- **Clear Send Buffer** clears the send buffer and removes the AWA and AWT instructions from the queue. The Error bit (ER) is set in each of these instructions.

  When Clear Receive Buffer and Clear Send Buffer are both set to Yes, all instructions are removed from the queue.

Example

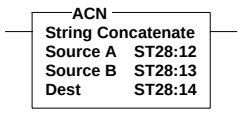


If input slot 1, bit 10 is set, then only clear the receive buffer for channel 0.

When the rung goes true, the selected buffer(s) will be cleared and the ASCII instruction(s) are removed from the ASCII instruction queue.

String Concatenate (ACN)

The ACN instruction combines two strings using ASCII strings as operands. The second string is appended to the first and the result stored in the destination.



Fixed	SLC 5/01	SLC 5/02	SLC 5/03	SLC 5/04	SLC 5/05
			•	•	•

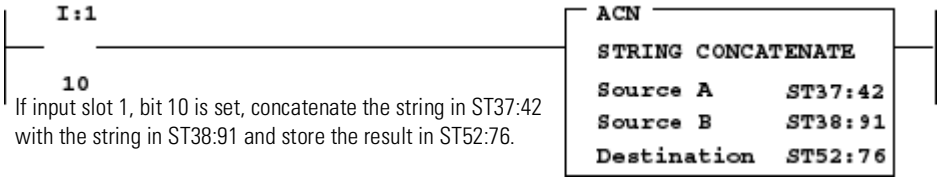
Output Instruction

Entering Parameters

Enter the following parameters when programming this instruction:

- **Source A** is the first string in the concatenation procedure.
- **Source B** is the second string in the concatenation procedure.
- **Destination** is where the result of Source A and B is stored.

Example

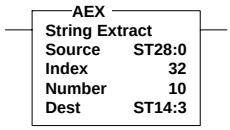


If input slot 1, bit 10 is set, concatenate the string in ST37:42 with the string in ST38:91 and store the result in ST52:76.

Only the first 82 characters (0 - 81) are written to the destination. If the result is > 82 the ASCII Error bit S:5/15 is set.

String Extract (AEX)

Use the AEX instruction to create a new string by taking a portion of an existing string and linking it to a new string.



Fixed	SLC 5/01	SLC 5/02	SLC 5/03	SLC 5/04	SLC 5/05
			•	•	•

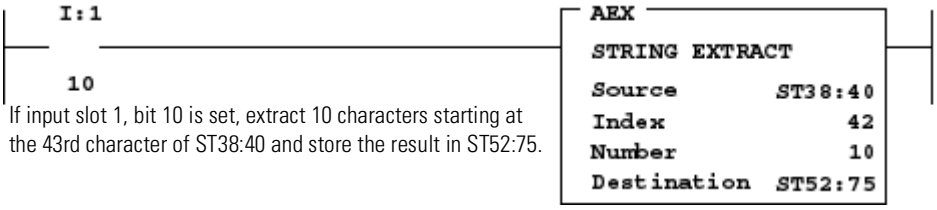
Output Instruction

Entering Parameters

Enter the following parameters when programming this instruction:

- **Source** is the existing string. The source value is not affected by this instruction.
- **Index** is the starting position (from 1 to 82) of the string you want to extract. (An index of 1 indicates the left-most character of the string.)
- **Number** is the number of characters (from 1 to 82) you want to extract, starting at the indexed position. If the index plus the number is greater than the total characters in the source string, the destination string will be the characters from the index to the end of the source string.
- **Destination** is the string element (ST) where you want the extracted string stored.

Example



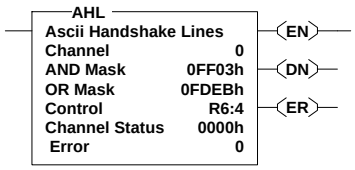
If input slot 1, bit 10 is set, extract 10 characters starting at the 43rd character of ST38:40 and store the result in ST52:75.

The following conditions cause the processor to set the ASCII Error bit (S:5/15):

- invalid source string length or string length of zero
- index or number values outside of range
- index value greater than the length of the source string

The destination string is not changed in any of the above error conditions. However, the destination will be changed if the index value plus the number value are greater than the string length. Note that the ASCII Error bit (S:5/15) is not set.

ASCII Handshake Lines (AHL)



Fixed	SLC 5/01	SLC 5/02	SLC 5/03	SLC 5/04	SLC 5/05
			•	•	•

Output Instruction

Use the AHL instruction to set or reset the RS-232 Data Terminal Ready (DTR) and Request to Send (RTS) handshake control lines for your modem. On a false-to-true transition, the processor uses the two masks to determine whether to set or reset the DTR and RTS lines, or leave them unchanged. This instruction will operate when the port is in either mode or is disabled.

TIP

Make sure the automatic modem control used by the port, in system mode, does not conflict with this instruction.

Entering Parameters

Enter the following parameters when programming this instruction:

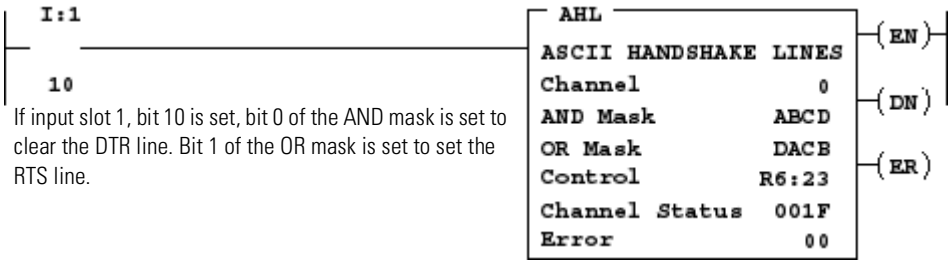
- **Channel** is the number of the RS-232 port (Channel 0).
- **AND** Mask is the type of mask used to reset the DTR and RTS control lines. Bit 0 corresponds to the DTR line and bit 1 corresponds to the RTS control line. A 1 at the mask bit causes the line to be reset; a 0 leaves the line unchanged. Note that mask values do not have a one-to-one correspondence to the modem control lines.
- **OR** Mask is the type of mask used to set the DTR and RTS control lines. Bit 0 corresponds to the DTR line and bit 1 corresponds to the RTS control line. A 1 at the mask bit causes the line to be set; a 0 leaves the line unchanged. Note that mask values do not have a one-to-one correspondence to the modem control lines.
- **Control** is the area that stores the control register required to operate the instruction.
- **Channel Status** displays the current status (0000 to 001F) of the handshake lines for the channel, specified above. This field is display only and resides in word 2 of the control element.
- **Error** displays the hexadecimal error code that indicates why the ER bit was set in the control data file (R6:). See page 10-23 for error code descriptions.

Example

The following shows the channel status as 001F.

Channel Status Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Bit	Reserved											DTR	DCD	DSR	RTS	CTC
Line	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1
Channel Status	0				0				1				F			

Example



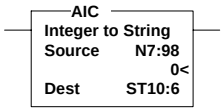
If input slot 1, bit 10 is set, bit 0 of the AND mask is set to clear the DTR line. Bit 1 of the OR mask is set to set the RTS line.

The Error bit (ER) is set during the execution of the instruction if:

- the instruction is aborted due to channel mode change
- the Unload bit (UL) is set and the instruction is not executed

Integer to String (AIC)

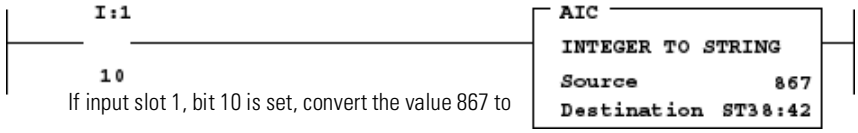
The AIC instruction converts an integer value (-32,768 and 32,767) to an ASCII string. The source can be a constant or an integer address.



Fixed	SLC 5/01	SLC 5/02	SLC 5/03	SLC 5/04	SLC 5/05
			•	•	•

Output Instruction

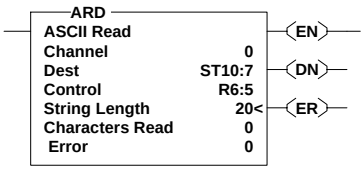
Example



If input slot 1, bit 10 is set, convert the value 867 to

ASCII Read Characters (ARD)

Use the ARD instruction to read characters from the buffer and store them in a string. To repeat the operation, the rung must go from false-to-true.



Fixed	SLC 5/01	SLC 5/02	SLC 5/03	SLC 5/04	SLC 5/05
			•	•	•

Output Instruction

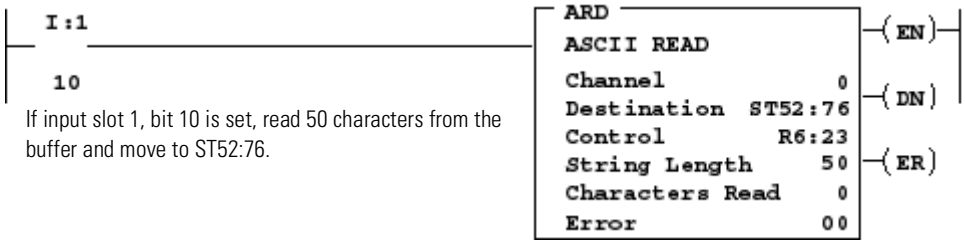
Entering Parameters

Enter the following parameters when programming this instruction:

- **Channel** is the number of the RS-232 port (Channel 0).
- **Destination** is the string element where you want the characters stored.
- **Control** is the address of the control block used to store data for the ARD instruction.

- **String Length** (.LEN) is the number of characters you want to read from the buffer. The maximum is 82 characters. If you specify a length larger than 82, only the first 82 characters will be read. (A 0 defaults to 82.) This is word 1 in the control block.
- **Characters Read** (.POS) are the number of characters that the processor moved from the buffer to the string (0 to 82). This field is updated during the execution of the instruction and is display only. This is word 2 in the control block.
- **Error** displays the hexadecimal error code that indicates why the ER bit was set in the control data file (R6:). See page 10-23 for error code descriptions.

Example



If input slot 1, bit 10 is set, read 50 characters from the buffer and move to ST52:76.

When the rung goes from false-to-true, the Enable bit (EN) is set. When the instruction is placed in the ASCII queue, the Queue bit (EU) is set. The Running bit (IN) is set when the instruction is executing. The DN bit is set upon completion of the instruction.

Once the requested number of characters are in the buffer, the characters are moved to the destination string. The number of characters moved is put in the POS field of the control block. The number in the Characters Read field is continuously updated and the Done bit (DN) is not set until all of the characters are read.

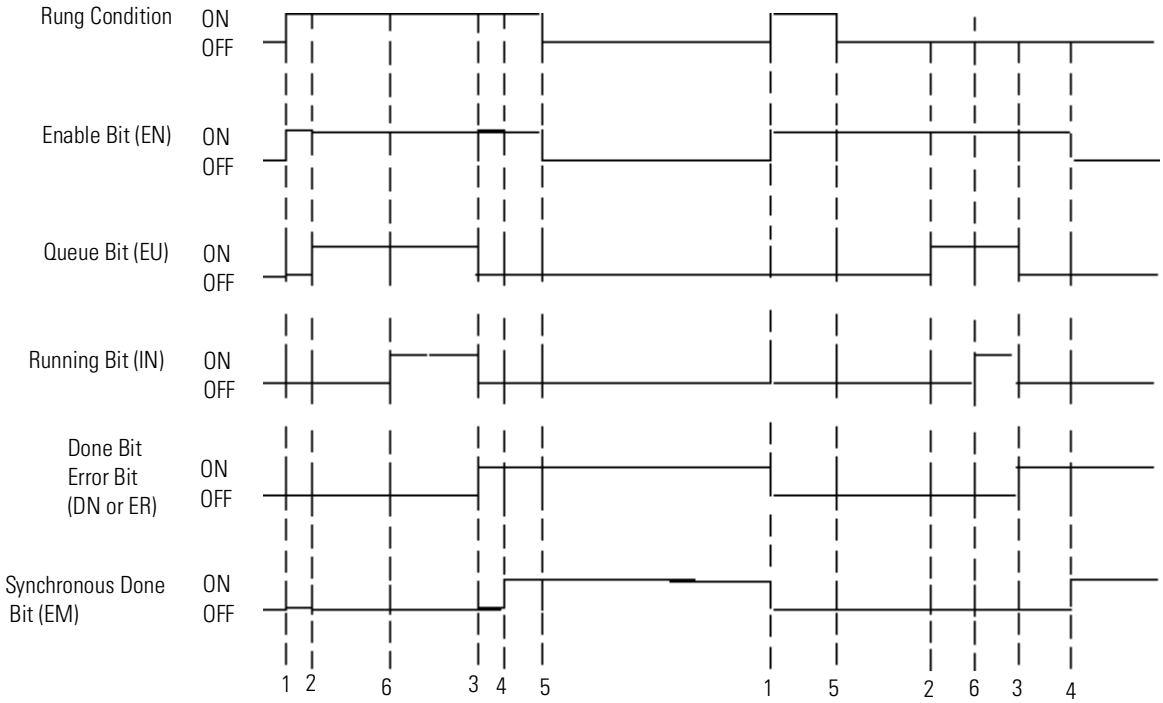
When the program scans the instruction and finds the Done bit (DN) set, the processor then sets the Synchronous Done bit (EM). The EM bit acts as a secondary done bit corresponding to the program scan.

The Error bit (ER) is set during the execution of the instruction if:

- the instruction is aborted - serial port is not in User mode
- the modem is disconnected (control line selection is other than “NO HANDSHAKING”)
- the instruction is aborted due to channel mode change
- the Unload bit (UL) is set. The instruction stops executing, but received characters are sent to the destination.

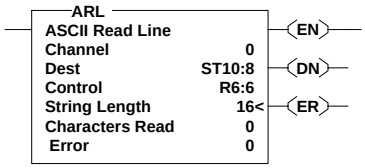
- an ACL to clear the receive buffer is executed, removing the ARD instruction from the ASCII queue

Timing Diagram for a Successful ARD, ARL, AWA, and AWT Instructions



1 - rung goes trues
2 - instruction successfully queued
3 - instruction execution complete
4 - instruction scanned for the first time after execution is complete
5 - rung goes false
6 - either the instruction is not in the queue or its being executed

ASCII Read Line (ARL)



Fixed	SLC 5/01	SLC 5/02	SLC 5/03	SLC 5/04	SLC 5/05
			•	•	•

Output Instruction

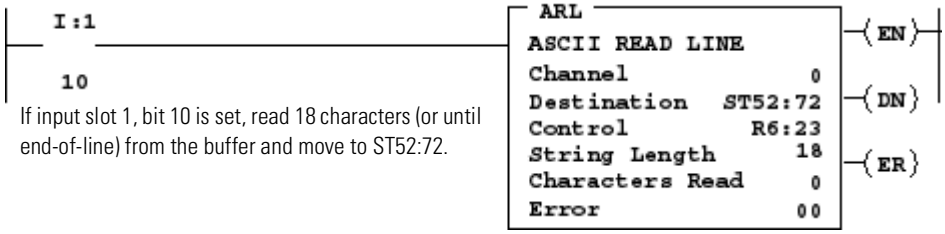
Use the ARL instruction to read characters from the buffer, up to and including the end-of-line (termination) characters, and store them in a string. The end-of-line characters are specified via the ASCII Configuration screen.

Entering Parameters

Enter the following parameters when programming this instruction:

- **Channel** is the number of the RS-232 port (Channel 0).
- **Destination** is the string element where you want the characters stored.
- **Control** is the address of the control block used to store data for the ARL instruction.
- **String Length** (LEN) is the number of characters you want to read from the buffer. The maximum is 82 characters. If you specify a length larger than 82, only the first 82 characters are read and moved to the destination. (A 0 defaults to 82.) This is word 1 in the control block.
- **Characters Read** (POS) are the number of characters that the processor moved from the buffer to the string (0 to 82). This field is display only and resides in word 2 of the control block.
- **Error** displays the hexadecimal error code that indicates why the ER bit was set in the control data file (R6:). See page 10-23 for error code descriptions.

Example



If input slot 1, bit 10 is set, read 18 characters (or until end-of-line) from the buffer and move to ST52:72.

When the rung goes from false-to-true, the control element Enable (EN) bit is set. When the instruction is placed in the ASCII queue, the Queue bit (EU) is set. The Running bit (IN) is set when the instruction is executing. The DN bit is set upon completion of the instruction.

Once either the end of line character(s) are received or, the requested number of characters are in the buffer, all characters (including the end-of-line characters) are moved to the destination string. The number of characters moved is stored in the POS word of the control block. The number in the Characters Read field is continuously

updated and the Done bit (DN) is not set until either the end of line character(s) are received or, all of the characters have been read.

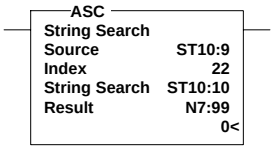
When the program scans the instruction and finds the Done bit (DN) set, the processor then sets the Synchronous bit (EM). The EM bit acts as a secondary done bit corresponding to the program scan.

The Error bit (ER) is set during the execution of the instruction if:

- the instruction is aborted - serial port is not in User mode
- the modem is disconnected (when control line selection is other than “NO HANDSHAKING”)
- the instruction is aborted due to channel mode change
- the Unload bit (UL) is set. The instruction stops executing, but received characters are sent to the destination.
- an ACL to clear the receive buffer is executed, removing the ARL instruction from the ASCII queue

TIP For information on the timing of this instruction, refer to the timing diagram on page 10-15.

String Search (ASC)



Fixed	SLC 5/01	SLC 5/02	SLC 5/03	SLC 5/04	SLC 5/05
			•	•	•

Output Instruction

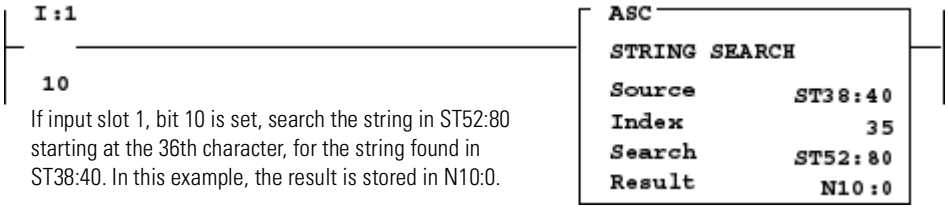
Use the ASC instruction to search an existing string for an occurrence of the source string.

Entering Parameters

Enter the following parameters when programming this instruction:

- **Source** is the string you want to find when examining the search string.
- **Index** is the starting position (from 1 to 82) of the portion of the string you want to find. (An index of 1 indicates the left-most character of the string.)
- **Search** is the string you want to examine.
- **Result** is an integer where the processor stores the position of the search string where the source string begins. If no match is found, result is set equal to zero.

Example



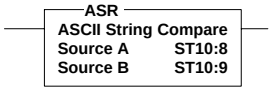
The following conditions cause the processor to set the ASCII Error bit (S:5/15).

- invalid string length or string length of zero
- index value outside of range
- index value greater than the length of the source string

The destination is not changed in any of the above conditions.

ASCII String Compare (ASR)

Use the ASR instruction to compare two ASCII strings. The system looks for a match in length and upper/lower case characters. If two strings are identical, the rung is true; if there are any differences, the rung is false.



Fixed	SLC 5/01	SLC 5/02	SLC 5/03	SLC 5/04	SLC 5/05
			•	•	•

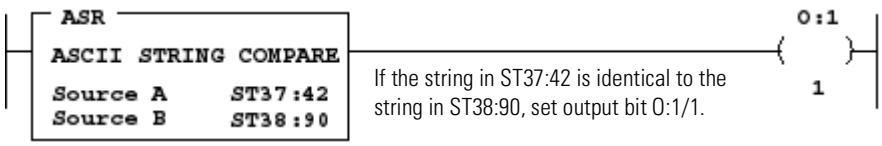
Input Instruction

Entering Parameters

Enter the following parameters when programming this instruction:

- **Source A** is string one for comparison.
- **Source B** is string two for comparison.

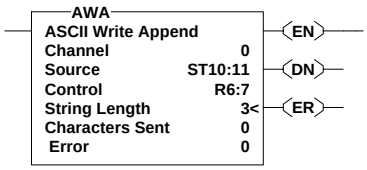
Example



An invalid string length causes the processor to set ASCII Error bit S:5/15, and the rung goes false.

ASCII Write with Append (AWA)

Use the AWA instruction to write characters from a source string to an external device. This instruction adds the one or two appended characters that you configure on the ASCII Configuration screen. The default is a carriage return and line feed appended to the end of the string. When using this instruction you can also perform in-line indirection. See page 10-20 for more information.



Fixed	SLC 5/01	SLC 5/02	SLC 5/03	SLC 5/04	SLC 5/05
			•	•	•

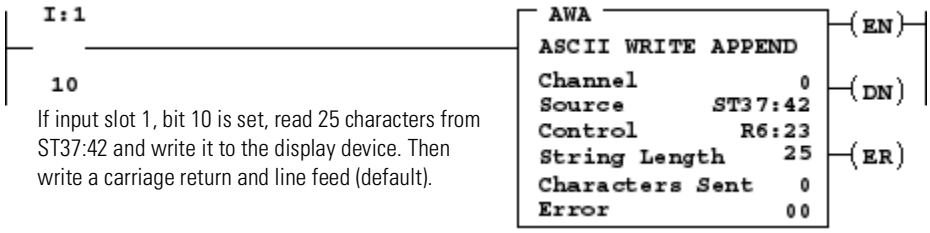
Output Instruction

Entering Parameters

Enter the following parameters when programming this instruction:

- **Channel** is the number of the RS-232 port (Channel 0).
- **Source** is the string element you want to write.
- **Control** is the area that stores the control register required to operate the instruction.
- **String Length** (.LEN) is the number of characters you want to write from the source string (0 to 82). If you enter a 0, the entire string will be written. This is word 1 in the control block.
- **Characters Sent** (.POS) are the number of characters that the processor sent to the display area (0 to 82). This field is continuously updated during the execution of the instruction. This value can be greater than the string length if appended characters or inserted values from in-line indirection are used. If the string length is greater than 82, the string written to the destination is truncated to 82 characters. This is word 2 in the control block.
- **Error** displays the hexadecimal error code that indicates why the ER bit was set in the control data file (R6:). See page 10-23 for error code descriptions.

Example



If input slot 1, bit 10 is set, read 25 characters from ST37:42 and write it to the display device. Then write a carriage return and line feed (default).

When the rung goes from false-to-true, the control element Enable (EN) bit is set. When the instruction is placed in the ASCII queue, the Queue bit (EU) is set. The Running bit (IN) is set when the instruction is executing. The DN bit is set on completion of the instruction.

The system sends 25 characters from the start of string ST37:42 to the display device and then sends user-configured append characters. The

Done bit (DN) is set and a value of 27 is present in .POS word of the ASCII control block.

When the program scans the instruction and finds the Done bit (DN) set, the processor then sets the Synchronous Done bit (EM) to act as a secondary done bit corresponding to the program scan.

The Error bit (ER) is set during execution of the instruction if:

- the modem is disconnected (control line selection is other than "NO HANDSHAKING")
- port is in System Mode and is configured for DH485
- the Unload bit (UL) is set. The instruction stops executing, but received characters are sent to the destination.
- an ACL to clear the send buffer is executed, removing the AWA instruction from the ASCII queue

TIP For information on the timing of this instruction, refer to the timing diagram on page 10-15.

Using In-line Indirection

This allows you to insert integer and floating point values into ASCII strings. The Running bit (IN) must be set before the string value can be used.

The following conditions apply to performing in-line indirection:

- all valid integer (N) and floating point (F) files can be used. Valid ranges include 7, 8, and 9-255.
- file types are not case sensitive and can include either a colon (:) or semicolon (;)
- positive values and leading zeros are not printed. Negative values are printed with a leading minus sign.

Examples

For the following examples:
N7:0 = 250
N7:1 = -37
F8:0 = 2.015000
F8:1 = 0.873000

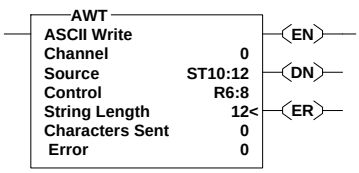
Valid in-line direction:	
Input:	Flow rate is currently [N7:0] GPH and contains [F8:0] PPM contaminants.
Output:	Flow rate is currently 250 GPH and contains 2.015000 PPM contaminants.
Input:	Current position is [N7:1] at a speed of [F8:1] RPM.
Output:	Current position is -37 at a speed of 0.873000 RPM.
Invalid in-line indirection:	
Input:	Current position is [N5:1] at a speed of [F8:1] RPM.
Output:	Current position is [N5:1} at a speed of 0.873000 RPM.

TIP

Truncation occurs in the output string if the indirection causes the output to exceed 80 characters. The appended characters are always applied to the output.

ASCII Write (AWT)

Use the AWT instruction to write characters from a source string to an external device. To repeat the instruction, the rung must go from false-to-true. When using this instruction you can also perform in-line indirection. See page 10-20 for more information.



Fixed	SLC 5/01	SLC 5/02	SLC 5/03	SLC 5/04	SLC 5/05
			•	•	•

Output Instruction

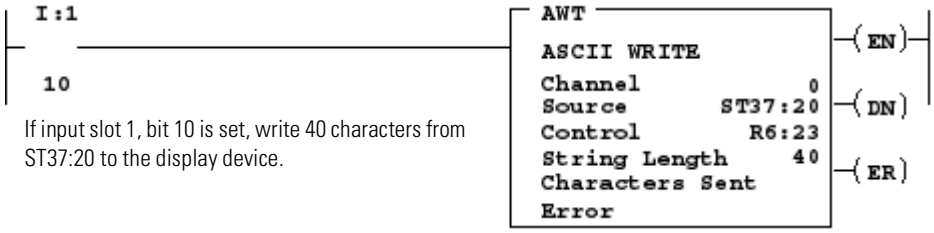
Entering Parameters

Enter the following parameters when programming this instruction:

- **Channel** is the number of the RS-232 port (Channel 0).
- **Source** is the string element you want to write.
- **Control** is the area that stores the control register required to operate the instruction.
- **String Length** (LEN) is the number of characters you want to write from the source string (0 to 82). If you enter a 0, the entire string will be written.

- **Characters Sent** (POS) is the number of characters that the processor sent to the display area (0 to 82). Only after the entire string is sent is this field updated (no running total for each character is stored). This field is display only. This value can be greater than the string length if inserted values from in-line indirection are used. If the string length is greater than 82, the string written to the destination is truncated to 82 characters.
- **Error** displays the hexadecimal error code that indicates why the ER bit was set in the control data file (R6:). See page 10-23 for error code descriptions.

Example



If input slot 1, bit 10 is set, write 40 characters from ST37:20 to the display device.

When the rung goes from false-to-true, the control element Enable (EN) bit is set. When the instruction is placed in the ASCII queue, the Queue bit (EU) is set. The Running bit (IN) is set when the instruction is executing. The DN bit is set on completion of the instruction.

Forty characters from string ST37:40 are sent through channel 0. The Done bit (DN) is set and a value of 40 is present in the POS word of the ASCII control block.

When the program scans the instruction and finds the Done bit (DN) set, the processor then sets the Synchronous Done bit (EM) to act as a secondary done bit corresponding to the program scan.

The Error bit (ER) is set during execution of the instruction if:

- the modem is disconnected (control line selection is other than “NO HANDSHAKING”)
- port is in System Mode and is configured for DH485
- the Unload bit (UL) is set. The instruction stops executing, but received characters are sent to the destination.
- an ACL to clear the send buffer is executed, removing the AWT instruction from the ASCII queue

TIP For information on the timing of this instruction, refer to the timing diagram on page 10-15.

ASCII Instruction Error Codes

The following error codes indicate why the Error bit (ER) is set in the control data file (R6:).

Table 10.4 ASCII Error Codes

Error Code (HEX)	Conditions Resulting in the Setting of the ER Bit	Recommended Action
00	No error. The instruction completed successfully.	None required.
02	Operation cannot be completed because the modem went offline.	Check modem cabling to communication channel. If the channel is configured for modem handshaking, both the DCD (Data-Carrier-Detect) and DSR (Data-Set-Ready) lines to the channel must be active for the modem to be online.
03	Transmission cannot be completed because the Clear-to-Send signal was lost.	Check modem and modem cabling connections.
04	Cannot perform ASCII receives because the communication channel is configured for System Mode.	Reconfigure the communication channel for User Mode.
05	While attempting to perform ASCII transmission, System Mode (DF1) communication was detected.	Verify that the modem is online and communicating with required devices.
07	Cannot perform ASCII send or receive because channel configuration has been shut down via the channel configuration menu.	Reconfigure the channel configuration menu and retry operation.
08	Cannot perform ASCII write due to an ASCII transmission already in progress.	Resend the transmission.
09	ASCII communication requested is not supported by current channel configuration. (Channel 0 is configured for DH-485 while trying to initiate an ASCII transmission or modem handshake control.)	Configure channel 0 for DF1, Full-Duplex.
0A	The Unload bit (UL) was set, stopping instruction execution.	None required.
0B	The requested length for the string is either a negative number or greater than 82. Applies to ARD and ARL instructions.	Enter a valid string length and retry operation.
0C	The length of the source string is either a negative number or greater than 82. Applies to AWA and AWT instructions.	Enter a valid string length and retry operation.

Table 10.4 ASCII Error Codes

Error Code (HEX)	Conditions Resulting in the Setting of the ER Bit	Recommended Action
0D	The requested length (.LEN) in the control block is a negative number or a value greater than 82. Applies to AWA and AWT instructions.	Enter a valid length and retry operation.
0E	The ACL instruction was aborted.	None required.
0F	The channel configuration mode was changed.	None required.

ASCII Conversion Table

The table below lists the decimal, hexadecimal, and ASCII conversions.

Table 10.5 ASCII Conversion Table

Decimal	Hex	ASCII	Enter as:	Displayed as:
0	00	NUL	\00	\00
1	01	SOH	^A or ^a or \01	^A
2	02	STX	^B or ^b or \02	^B
3	03	ETX	^C or ^c or \03	^C
4	04	EOT	^D or ^d or \04	^D
5	05	ENQ	^E or ^e or \05	^E
6	06	ACK	^F or ^f or \06	^F
7	07	BEL	^G or ^g or \07	^G
8	08	BS	^H or ^h or \08	^H
9	09	HT	^I or ^i or \09	^I
10	0A	LF	^J or ^j or \0A or \0a	^J
11	0B	VT	^K or ^k or \0B or \0b	^K
12	0C	FF	^L or ^l or \0C or \0c	^L
13	0D	CR	^M or ^m or \0D or \0d	^M
14	0E	SOH	^N or ^n or \0E or \0e	^N
15	0F	SI	^O or ^o or \0F or \0f	^O
16	10	DLE	^P or ^p or \10	^P
17	11	DC1	^Q or ^q or \11	^Q
18	12	DC2	^R or ^r or \12	^R
19	13	DC3	^S or ^s or \13	^S
20	14	DC4	^T or ^t or \14	^T
21	15	HAK	^U or ^u or \15	^U
22	16	SYN	^V or ^v or \16	^V

Table 10.5 ASCII Conversion Table

Decimal	Hex	ASCII	Enter as:	Displayed as:
23	17	ETB	^W or ^w or \17	^W
24	18	CAN	^X or ^x or \18	^X
25	19	EM	^Y or ^y or \19	^Y
26	1A	SUB	^Z or ^z or \1A or \1a	^Z
27	1B	ESC	\1B or \1b	\1B
28	1C	FS	\1C or \1c	\1C
29	1D	GS	\1D or \1d	\1D
30	1E	RS	\1E or \1e	\1E
31	1F	US	\1F or \1f	\1F
32	20	SP	\20	
33	21	!	! or \21	!
34	22	””””	”””” or \22”	””””
35	23	#	# or \23	#
36	24	$	$ or \24	$
37	25	%	% or \25	%
38	26	&	& or \26	&
39	27	‘	‘ or \27	‘
40	28	(	(or \28	(
41	29	)	) or \29	)
42	2A	*	* or \2A or \2a	*
43	2B	+	+ or \2B or \2b	+
44	2C	,	, or \2C or \2c	,
45	2D	-	- or \2D or \2d	-
46	2E	.	. or \2E or \2e	.
47	2F	/	/ or \2F or \2f	/
48	30	0	0 or \30	0
49	31	1	1 or \31	1
50	32	2	2 or \32	2
51	33	3	3 or \33	3
52	34	4	4 or \34	4
53	35	5	5 or \35	5
54	36	6	6 or \36	6
55	37	7	7 or \37	7
56	38	8	8 or \38	8
57	39	9	9 or \39	9
58	3A	:	: or \3A or \3a	:

Table 10.5 ASCII Conversion Table

Decimal	Hex	ASCII	Enter as:	Displayed as:
23	17	ETB	^W or ^w or \17	^W
24	18	CAN	^X or ^x or \18	^X
25	19	EM	^Y or ^y or \19	^Y
26	1A	SUB	^Z or ^z or \1A or \1a	^Z
27	1B	ESC	\1B or \1b	\1B
28	1C	FS	\1C or \1c	\1C
29	1D	GS	\1D or \1d	\1D
30	1E	RS	\1E or \1e	\1E
31	1F	US	\1F or \1f	\1F
32	20	SP	\20	
33	21	!	! or \21	!
34	22	””””	”””” or \22”	””””
35	23	#	# or \23	#
36	24	$	$ or \24	$
37	25	%	% or \25	%
38	26	&	& or \26	&
39	27	’	’ or \27	’
40	28	(	(or \28	(
41	29	)	) or \29	)
42	2A	*	* or \2A or \2a	*
43	2B	+	+ or \2B or \2b	+
44	2C	,	, or \2C or \2c	,
45	2D	-	- or \2D or \2d	-
46	2E	.	. or \2E or \2e	.
47	2F	/	/ or \2F or \2f	/
48	30	0	0 or \30	0
49	31	1	1 or \31	1
50	32	2	2 or \32	2
51	33	3	3 or \33	3
52	34	4	4 or \34	4
53	35	5	5 or \35	5
54	36	6	6 or \36	6
55	37	7	7 or \37	7
56	38	8	8 or \38	8
57	39	9	9 or \39	9
58	3A	:	: or \3A or \3a	:

Table 10.5 ASCII Conversion Table

Decimal	Hex	ASCII	Enter as:	Displayed as:
59	3B	;	; or \3B or \3b	;
60	3C	<	< or \3C or \3c	<
61	3D	=	= or \3D or \3d	=
62	3E	>	> or \3E or \3e	>
63	3F	?	? or \3F or \3f	?
64	40	@	@ or \40	@
65	41	A	A or \41	A
66	42	B	B or \42	B
67	43	C	C or \43	C
68	44	D	D or \44	D
69	45	E	E or \45	E
70	46	F	F or \46	F
71	47	G	G or \47	G
72	48	H	H or \48	H
73	49	I	I or \49	I
74	4A	J	J or \4A or \4a	J
75	4B	K	K or \4B or \4b	K
76	4C	L	L or \4C or \4c	L
77	4D	M	M or \4D or \4d	M
78	4E	N	N or \4E or \4e	N
79	4F	O	O or \4F or \4f	O
80	50	P	P or \50	P
81	51	Q	Q or \51	Q
82	52	R	R or \52	R
83	53	S	S or \53	S
84	54	T	T or \54	T
85	55	U	U or \55	U
86	56	V	V or \56	V
87	57	W	W or \57	W
88	58	X	X or \58	X
89	59	Y	Y or \59	Y
90	5A	Z	Z or \5A or \5a	Z
91	5B	[	[or \5B or \5b	[
92	5C	\	\ or \5C or \5c	\
93	5D	]	] or \5D or \5d	]
94	5E	^	\^ or \5E or \5e	\^

Table 10.5 ASCII Conversion Table

Decimal	Hex	ASCII	Enter as:	Displayed as:
95	5F	_	_ or \5F or \5f	_
96	60	‘	‘ or \60	‘
97	61	a	a or \61	a
98	62	b	b or \62	b
99	63	c	c or \63	c
100	64	d	d or \64	d
101	65	e	e or \65	e
102	66	f	f or \66	f
103	67	g	g or \67	g
104	68	h	h or \68	h
105	69	i	I or \69	i
106	6A	j	j or \6A or \6a	j
107	6B	k	k or \6B or \6b	k
108	6C	l	l or \6C or \6c	l
109	6D	m	m or \6D or \6d	m
110	6E	n	n or \6E or \6e	n
111	6F	o	o or \6F or \6f	o
112	70	p	p or \70	p
113	71	q	q or \71	q
114	72	r	r or \72	r
115	73	s	s or \73	s
116	74	t	t or \74	t
117	75	u	u or \75	u
118	76	v	v or \76	v
119	77	w	w or \77	w
120	78	x	x or \78	x
121	79	y	y or \79	y
122	7A	z	z or \7A or \7a	z
123	7B	{	{ or \7B or \7b	{
124	7C	|	| or \7C or \7c	|
125	7D	}	} or \7D or \7d	}
126	7E	~	~ or \7E or \7e	~
127	7F	DEL	\7F or \7f	?
128 to 255	80 to FF		\80 to \FF	

Understanding Interrupt Routines

This chapter contains general information about interrupt routines and explains how they function in your logic program. Each interrupt routine includes:

- an overview
- programming procedure
- operational description
- associated bit description

In addition, each interrupt routine contains an application example that shows the interrupt routine in use.

Table 11.1 Interrupt Routine Instructions

Instruction Mnemonic	Instruction Name	Purpose	Page
	User Fault Routine	Provides the option of preventing a processor shutdown.	11-2
STI	Selectable Timed Interrupt	Allows you to interrupt the scan of the main program file automatically, on a periodic basis, to scan a specified subroutine file.	11-8
STD	Selectable Timed Disable	Disables STI's from occurring.	11-17
STE	Selectable Timed Enable	Enables STI's to occur.	11-17
STS	Selectable Timed Start	Sets or changes the file number or setpoint frequency of the STI routine.	11-18
DII	Discrete Input Interrupt	Allows the processor to execute a subroutine when the input bit pattern of a discrete I/O card matches a compare value that you programmed.	11-19
ISR	I/O Interrupt	Allows a specialty I/O module to interrupt the normal processor operating cycle in order to scan a specified subroutine file.	11-29
IID	I/O Interrupt Disable	Disables I/O interrupts from occurring.	11-34
IIE	I/O Interrupt Enable	Enables I/O interrupts to occur.	11-34
RPI	Reset Pending Interrupt	Aborts a pending I/O Interrupt.	11-36
INT	Interrupt Subroutine	Optional instruction to identify interrupt subroutines.	11-36

User Fault Routine Overview

Fixed	SLC 5/01	SLC 5/02	SLC 5/03	SLC 5/04	SLC 5/05
		•	•	•	•

The user fault routine gives you the option of preventing a processor shutdown when a specific user fault occurs. The file is executed when any recoverable or non-recoverable user fault occurs. The file is not executed for non-user faults.

A fault routine is programmed in a program file other than 2. The program file used is specified as the fault routine in word S:29 in the status file. More than one user fault routine can exist. The example on page 11-5 shows how this can be accomplished.

Faults are classified as recoverable and non-recoverable user faults, and non-user faults. A complete list of faults appears in Appendix B.

Non-User Fault	Non-Recoverable User Fault	Recoverable User Fault
The Fault Routine does not execute.	The Fault Routine executes for 1 pass.	The Fault Routine may clear the fault by clearing bit S:1/13.

TIP You may initiate a MSG instruction to another node to identify the fault condition of the processor.

TIP For SLC 5/02 processors: You must save your program with test single step selected in order for S:20 and S:21 to be activated.

For SLC 5/03 and higher processors: If your program contains four message instructions with the Continuous Operation (CO) bit set, the fault routine's message instruction is not executed.

Status File Data Saved

Data in the following words is saved on entry to the user fault subroutine and re-written upon exiting the subroutine.

- S:0 Arithmetic flags
- S:13 and S:14 Math register
- S:24 Index register

Creating a User Fault Subroutine

To use the user fault subroutine:

1. Create a subroutine file: valid range is 3-255.

2. Enter the file number in word S:29 of the status file.
   Project Tree - Controller Folder - Processor Status File - Errors Tab

The occurrence of recoverable or non-recoverable user faults causes the processor to read S:29 and execute the subroutine number contained in S:29. If the fault is recoverable, the routine can be used to correct the problem and clear the fault bit S:1/13. The processor then continues in the REM Run mode.

The routine does not execute for non-user faults.

Words S:20 and S:21 can be examined in your fault routine to pinpoint the file and rung number where the fault occurred. If the fault occurred outside of the ladder scan, this value will contain the rung number where the TND, END, or REF instruction is located. Use words S:20 and S:21 with your power-up protection fault routine to determine the exact point that the previous power down occurred. Refer to Appendix B for more information about the Startup Protection Fault bit, S:1/9.

User Interrupt Routine Application Example

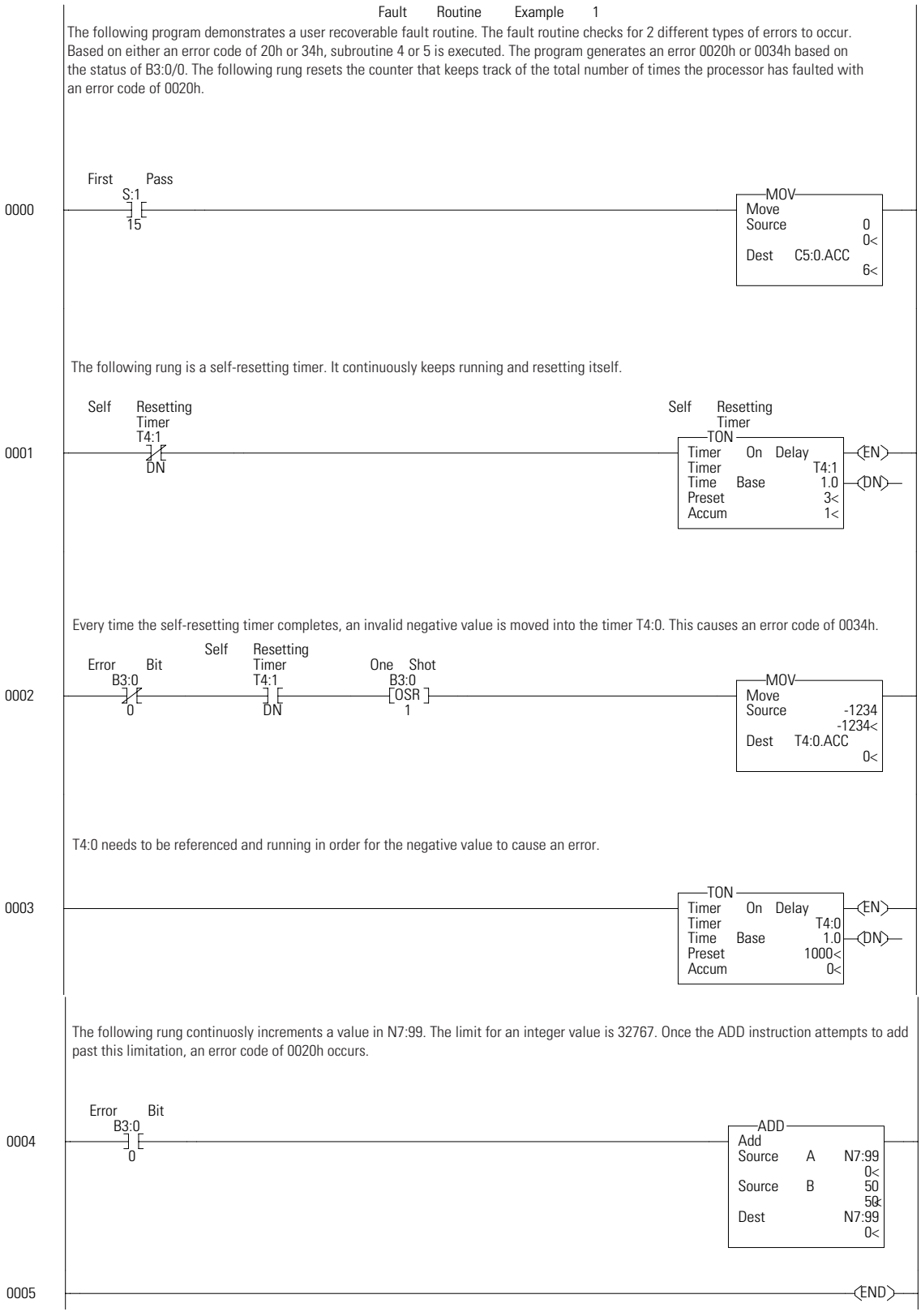
Suppose you have a program in which you want to control major errors 0020h (MINOR ERROR AT END OF SCAN) and 0034h (NEGATIVE VALUE IN TIMER PRE OR ACC) under the following conditions:

- Prevent a processor shutdown if the overflow trap bit S:5/0 is set. Permit a processor shutdown when S:5/0 is set more than five times.
- Prevent a processor shutdown if the accumulator value of timer T4:0 becomes negative. Reset the negative accumulator value to zero. Energize an output to indicate that the accumulator has gone negative one or more times.
- Allow a processor shutdown for all other user faults.

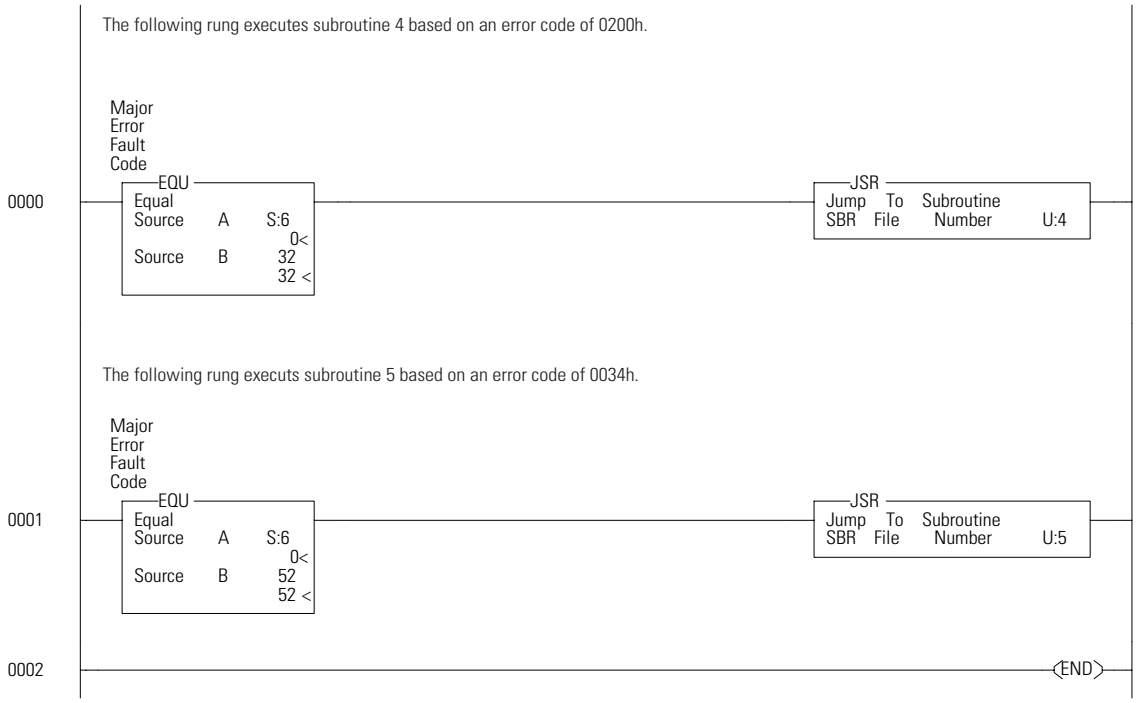
A possible method of accomplishing this is shown in the following examples. The user fault routine is designated as file 3.

When a recoverable or non-recoverable user error occurs, the processor scans subroutine file 3. The processor jumps to file 4 if the error code is 0020 and it jumps to file 5 if the error code is 0034h. For all other recoverable and non-recoverable errors, the processor exits the fault routine and halts operation in the fault mode.

Fault Routine - Subroutine File 3



Subroutine File 4 - Executed for Error 0020h

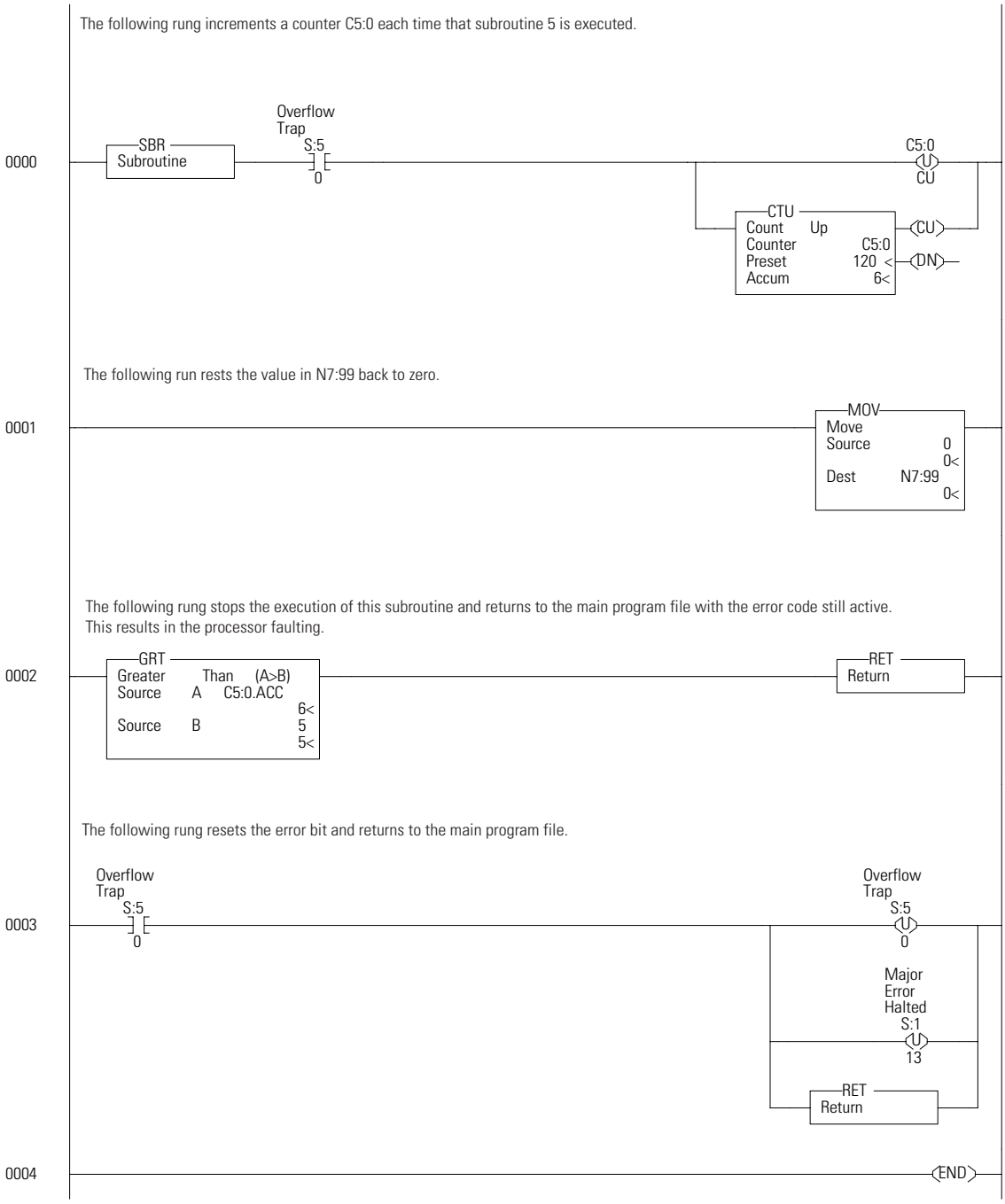


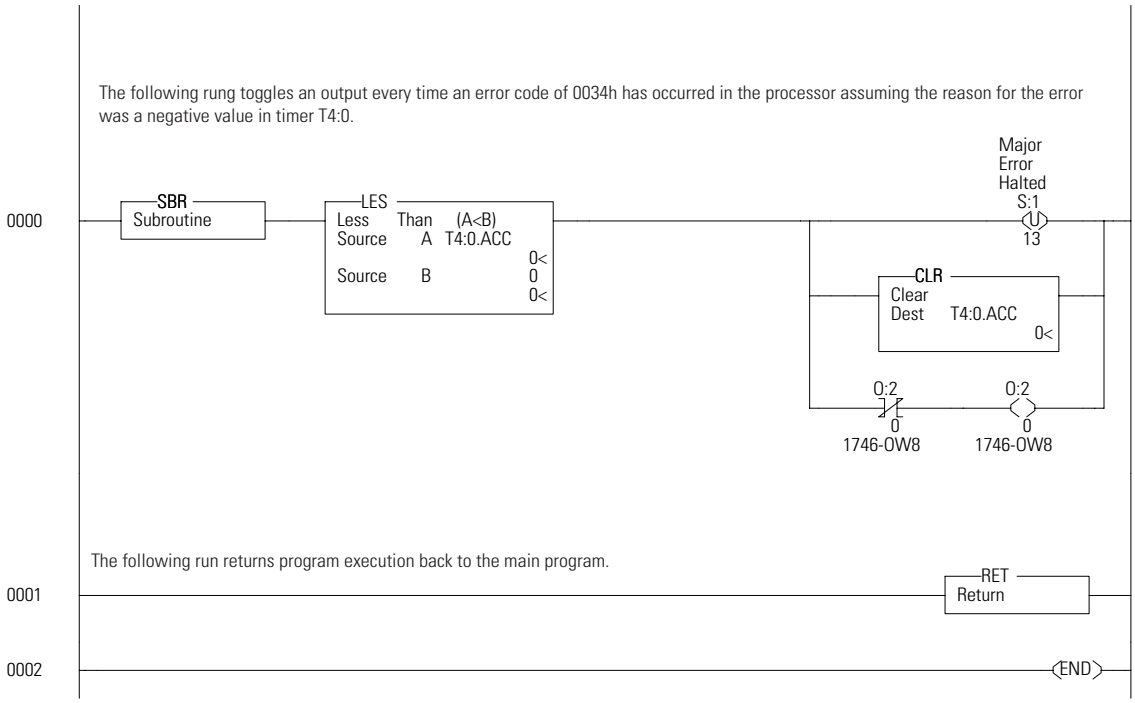
If the overflow trap bit, S:5/0 is set, counter C5:0 increments.

If the count of C5:0 is 5 or less, the overflow trap, S:5/0 is cleared, the major error halted bit S:1/13 is cleared, and the processor remains in the REM Run mode. If the count is greater than 5, the processor sets S:5/0 and S:1/13 and enters the Fault mode.

Subroutine file 5 is executed if the control register error bit S:5/2 is set.

Subroutine File 5 - Executed for Error 0034h





If the accumulator value of timer T4:0 is negative, the major error halted bit, S:1/13 is unlatched, preventing the processor from entering the Fault mode. At the same time, the accumulator value T4:0 ACC is cleared to zero and output O:3.0/3 is energized. Fault code 0034h is displayed in the status file.

If the preset of timer T4:0 is negative, S:1/13 remains set and the processor enters the Fault mode (O:3.0/3 will be reset if previously set). Also, if either the preset or accumulator value of any other timer in the program is negative, S:1/13 is set and the processor enters the Fault mode. If previously set, O:3.0/3 is reset.

Selectable Timed Interrupt Overview

This function allows you to interrupt the scan of the processor automatically, on a periodic basis, to scan a specified subroutine file. Afterward, the processor resumes executing from the point where it was interrupted.

This section describes:

- STI programming procedure
- STI operation and parameters
- STD and STE instructions
- STS instruction

Fixed	SLC 5/01	SLC 5/02	SLC 5/03	SLC 5/04	SLC 5/05
		•	•	•	•

Basic Programming Procedure for the STI Function

To use the STI function in your application file:

1. Create a subroutine file and enter the desired ladder rungs. This is your STI subroutine file. The valid range is 3 to 255.

2. Enter the STI subroutine file number in word S:31 of the status file (Project Tree - Controller Folder - Processor Status File - STI Tab). Refer to page B-41 in this manual for more information. A file number of zero disables the STI function.

3. Enter the setpoint (the time between successive interrupts) in word S:30 of the status file. Refer to page B-41 for more information.
   - For SLC 5/02: The range is 10 to 2550 ms (entered in 10 ms increments). A setpoint of zero disables the STI function. Refer to page B-14 in this manual for more information about the STI Resolution bit S:2/10.
   - For SLC 5/03: The range is 2 to 32,767 ms. A setpoint of zero disables the STI function. Refer to page B-14 in this manual for more information about the STI Resolution bit S:2/10.
   - For SLC 5/04 and higher processors: The range is from 1 to 32,767 ms (entered in 1 ms increments). A setpoint of zero disables the STI function.

TIP

The setpoint value must be a longer time than the execution time of the STI subrouting file plus the maximum interrupt latency, or a minor error bit is set. For all processors, the STI Pending bit and STI Overflow bit will be set. Additionally, for the SLC 5/03 and higher processors, the STI Lost bit may be set.

Operation

After you download your program and enter the REM Run mode, the STI begins operation as follows:

1. The STI timer begins timing.

2. When the STI interval expires, the STI timer is reset, the processor scan is interrupted and the STI subroutine file is scanned.

3. If while executing the STI subroutine, another STI interrupt occurs, the STI Pending bit (S:2/0) is set.

4. If while an STI is pending, the STI timer expires, the STI Lost bit (S:36/9) is set. (For SLC 5/02 processors, the Overflow (S:5/10) bit is set.)

5. When the STI subroutine scan is completed, scanning of the main program file resumes at the point where it left off, unless an STI is pending. In this case, the subroutine is immediately scanned again.

6. The cycle repeats.

For identification of your STI subroutine, include an INT instruction as the first instruction on the first rung of the file.

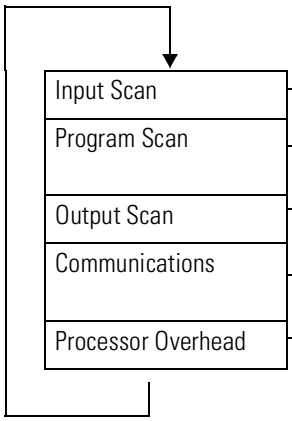
STI Subroutine Content

The STI subroutine contains the rungs of your application logic. You can program any instruction inside the STI subroutine except a TND, REF, or SVC instruction. IIM or IOM instructions are needed in an STI subroutine if your application requires immediate update of input or output points. End the STI subroutine with an RET instruction.

JSR stack depth is limited to 3. You may call other subroutines to a level 3 deep from an STI subroutine.

Interrupt Latency and Interrupt Occurrences

Interrupt latency is the interval between the STI time-out and the start of the interrupt subroutine. STI interrupts can occur at any point in your program, but not necessarily at the same point on successive interrupts. The tables below show the interaction between an interrupt and the processor operating cycle.



	SLC 5/02 STI	SLC 5/03 and Higher STI with Bit S:33/8 set	SLC 5/03 and Higher STI with Bit S:33/9 cleared
Input Scan	Between slot updates	Between word updates	Between slot updates
Program Scan	Between instruction updates	Between word updates	Between rung updates
Output Scan	Between slot updates	Between word updates	Between slot updates
Communications	Between communication packets	Between word packet updates	Between communication packets
Processor Overhead	At start and end	Between word updates	Between word updates

Events in the Processor Operating Cycle

Note that STI execution time adds directly to the overall scan time. During the latency period, the processor is performing operations that cannot be disturbed by the STI interrupt function.

Latency periods are:

- SLC 5/02 processors interrupts are serviced within 2.4 ms maximum.
- SLC 5/03 and higher processors: If an interrupt occurs while the processor is performing a multi-word slot update and your interrupt subroutine accesses that same slot, the multi-word transfer finishes to completion prior to performing the interrupt subroutine slot access. The Interrupt Latency Control bit (S:33/8) functions as follows:
  - When the bit is set (1), interrupts are serviced within the interrupt latency time.
  - When the bit is clear (0), INTs are serviced per rung, slot, and packet execution time.

The default state is cleared (0). To determine the interrupt latency with S:33/8 clear, you must calculate the execution time of each and every rung in your program. Use the longest calculated execution time plus your maximum interrupt latency.

Interrupt Priorities

Interrupt priorities for the processors are:

SLC 5/02 Processor	SLC 5/03 and Higher Processors
1. User Fault Routine	1. User Fault Routine
2. Selectable Timed Interrupt Subroutine	2. Discrete Input Interrupt (DII)
3. Interrupt Subroutine (ISR)	3. Selectable Timed Interrupt Subroutine
	4. Interrupt Subroutine (ISR)

An executing interrupt can only be interrupted by an interrupt having higher priority.

TIP Under certain conditions, though, it is possible for a lower priority task to run during the DII execution.

Status File Data Saved

Data in the following words is saved on entry to the STI subroutine and re-written upon exiting the STI subroutine.

- S:0 Arithmetic flags
- S:13 and S:14 Math register
- S:24 Index register

STI Parameters

The following parameters are associated with the STI function. These parameters have status file addresses that are described here and also in Appendix B of this manual.

- STI file number (Word S:31) - This can be any number from 3 to 255. A value of zero disables the STI function. An invalid number generates fault 0023h.
- Setpoint (Word S:30) - This is the time between the starting point of successive scans of the STI file. It can be any value from 10 to 2550 milliseconds. (For SLC 5/02 see page 11-9.) You enter a value of 1 to 255, which results in a 10 to 2550 ms setpoint. A value of zero disables the STI function. An invalid time generates fault 0024h.

  SLC 5/03 and higher processors: If S:2/10 is set, time is in 1 ms increments. If this bit is clear, time is in 10 ms increments.

  If the STI is initiated while in the REM Run mode by loading the status registers, the interrupt starts timing from the end of the program scan in which the status registers were loaded. If the STI has been previously configured (with a different setpoint), the new setpoint takes effect only after the previously-configured STI has timed out.

- STI Pending Bit (S:2/0) - This bit is set when the STI timer has timed out and the STI routine is waiting to be executed. This bit is reset upon starting the STI routine, execution of a true STS instruction, power-up, or exit from the REM Run or Test mode.

  SLC 5/02 specific: The STI pending bit is not set if the STI timer expires while executing the fault routine.

  SLC 5/03 and higher processors: This bit is set if the STI timer expires while executing the DII subroutine or fault routine.

- STI Enable Bit (S:2/1) - The default value is 1 (set). When a file number between 3 and 255 is present in word S:31 and a setpoint value between 1 and 255 is present in word S:30, a set enable bit allows scanning of the STI file. If the bit is reset by an STD instruction, scanning of the STI file no longer occurs. If the bit is set by an STE or STS instruction, scanning is again allowed. The enable bit only enables/disables the scanning of the STI subroutine. It does not affect the STI timer. The STS instruction affects both the enable bit and the STI timer. The default state is enabled. If this bit is set or reset using the STE, STD, or STS instruction, enable/disable takes effect immediately. If this bit is set in the user program using an instruction other than STE, STD, or STS, it takes effect at the next end of scan.

SLC 5/02 specific: If this bit is set or reset by the user program or communications, it does not take effect until the next end of scan.

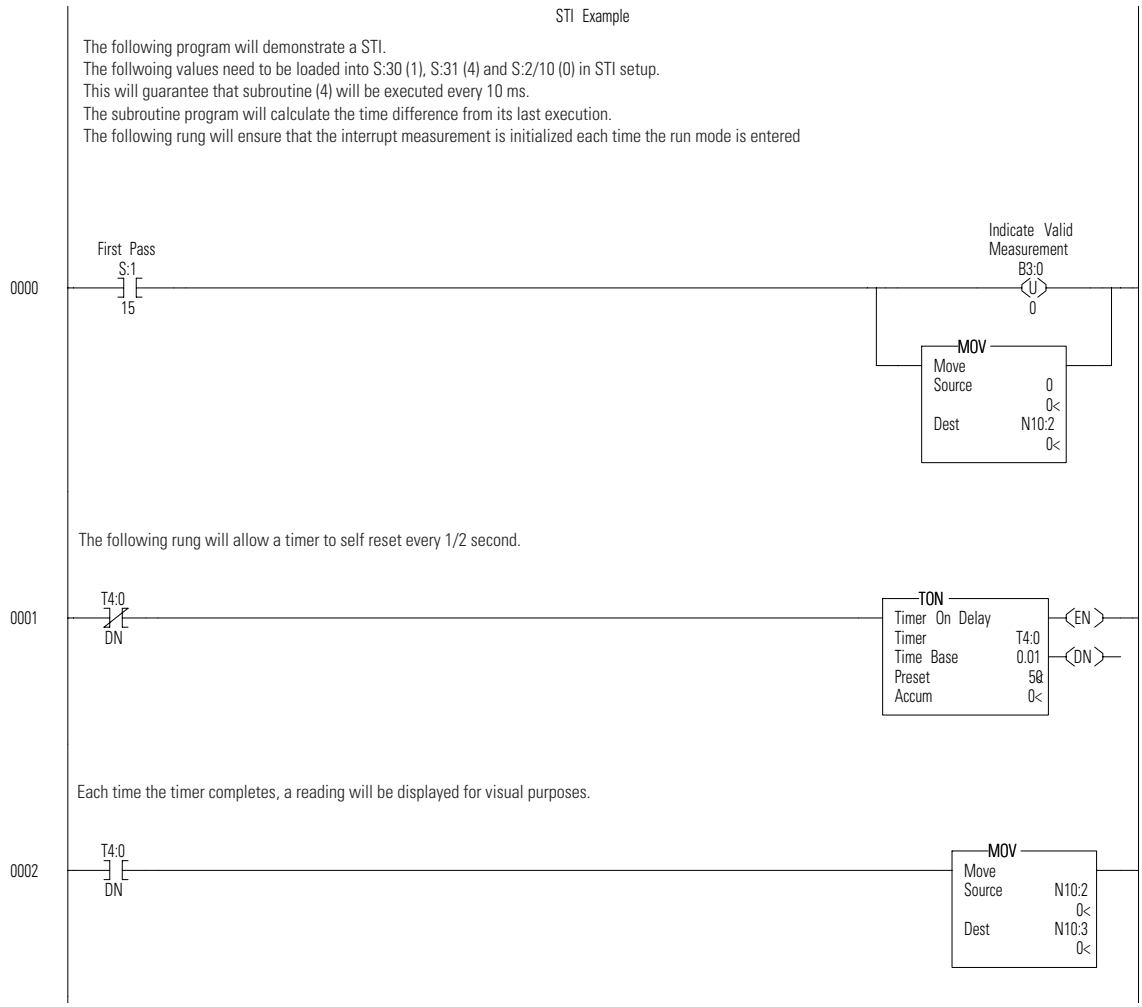
SLC 5/03 and higher processors: If this bit is set or reset by the user program or communications, it takes effect upon the STI timer expiration or next end of scan (whichever occurs first).

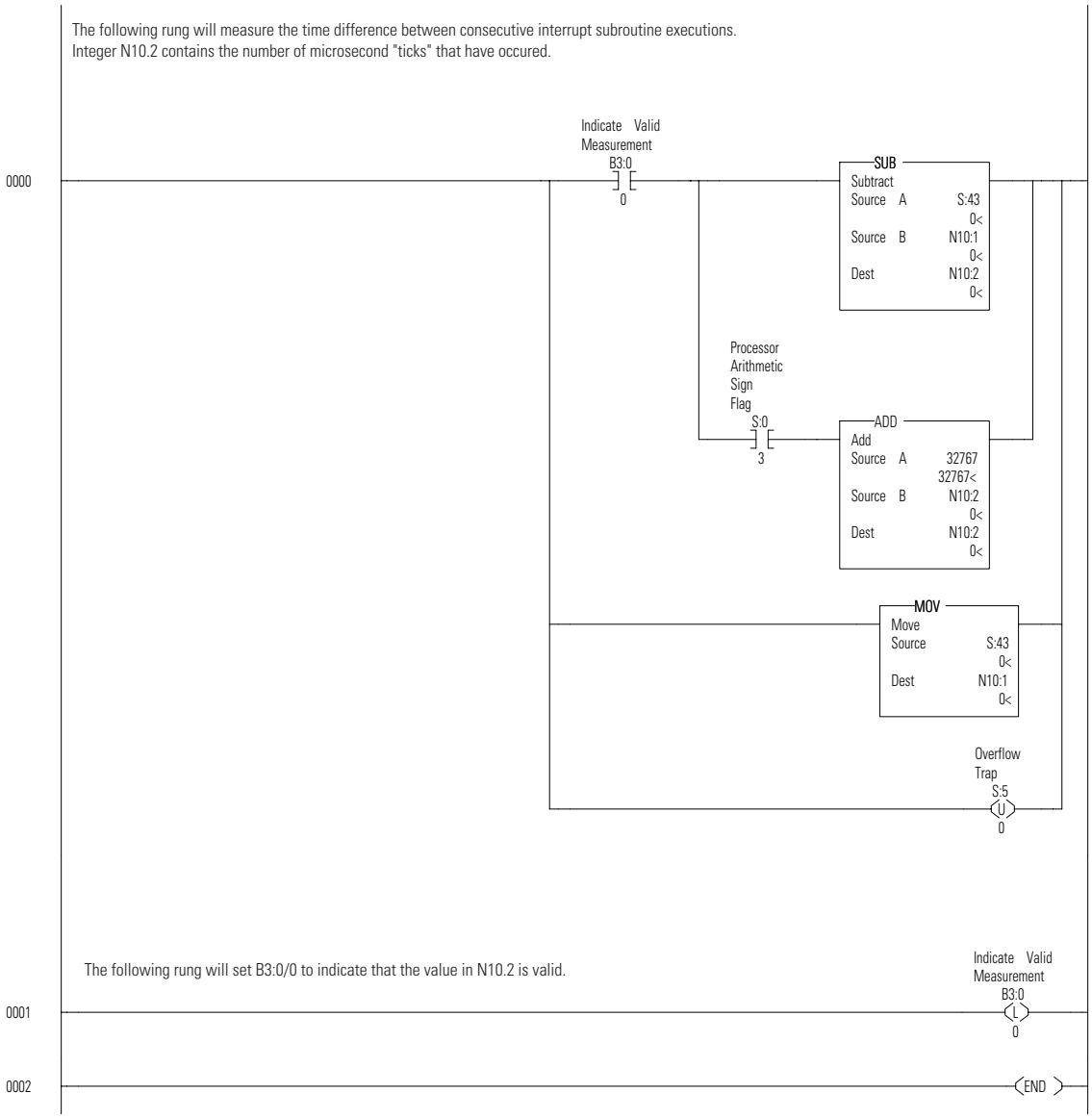
- STI Executing Bit (S:2/2) - This bit is set when the STI file is being scanned and cleared when the scan is completed. The bit is also cleared on power-up and entry into the REM Run mode.
- STI Resolution Selection Bit (S:2/10) - This bit is clear by default. When clear, this bit selects a 10 ms increment for the STI Setpoint (S:30) value. When set, this bit selects a 1 ms increment for the STI Setpoint (S:30) value. To program this feature, use the data monitor function to set/clear this bit, or address this bit with your ladder program.

  This bit is user configurable and takes effect on a REM PROG to REM RUN mode transition.

- Overflow Bit (S:5/10) - This minor error bit is set whenever the STI timer expires while the STI routine is executing or disabled while the pending bit is set. When this occurs, the STI timer continues to operate at the rate present in word S:30. If the overrun bit becomes set, take the corrective action your application dictates, then clear the bit.
- STI Lost Bit (Word S:36/9) - This bit is set anytime an STI interrupt occurs while the STI Pending bit is also set. When set, you are notified that a STI interrupt has been lost. For example, the interrupt is lost because a previous interrupt was already pending and waiting execution. Examine this bit in your user program and take appropriate action if your application cannot tolerate this condition. Then clear this bit with your user program in order to prepare for the next possible occurrence of this error.

Use the following rungs to initialize and measure the amount of time between two consecutive STI subroutine executions. The 10 µs timer is also available in the DII interrupt and I/O interrupt. This application example can also be used for the Event I/O interrupt or the DII interrupt by replacing S:43 with either S:44 or S:45 respectively.


STI Example
The following program will demonstrate a STI.
The follwoing values need to be loaded into S:30 (1), S:31 (4) and S:2/10 (0) in STI setup.
This will guarantee that subroutine (4) will be executed every 10 ms.
The subroutine program will calculate the time difference from its last execution.
The following rung will ensure that the interrupt measurement is initialized each time the run mode is entered
First Pass
S:1
15
0000
Indicate Valid
Measurement
B3:0
U
0
MOV
Move
Source 0
Dest N10:2
The following rung will allow a timer to self reset every 1/2 second.
T4:0
DN
0001
TON
Timer On Delay
Timer T4:0
Time Base 0.01
Preset 50
Accum 0
EN
DN
Each time the timer completes, a reading will be displayed for visual purposes.
T4:0
DN
0002
MOV
Move
Source N10:2
Dest N10:3



TIP The math overflow selection bit (S:2/14) must be set prior to entering RUN mode.

STD and STE Instructions

The STD and STE instructions are used to create zones in which STI ladder execution *cannot* occur. The STI timer continues to operate at the rate present in word S:30.

Fixed	SLC 5/01	SLC 5/02	SLC 5/03	SLC 5/04	SLC 5/05
		•	•	•	•

Selectable Timed Disable - STD

When true, this instruction resets the STI enable bit and prevents the STI subroutine from executing. When the rung goes false, the STI enable bit remains reset until a true STS or STE instruction is executed. The STI timer continues to operate while the enable bit is reset.

Selectable Timed Enable - STE

This instruction, upon a false-true transition of the rung, sets the STI enable bit and allows execution of the STI subroutine. When the rung goes false, the STI enable bit remains set until a true STD instruction is executed. This instruction has no effect on the operation of the STI timer or setpoint. When the enable bit is set, the first execution of the STI subroutine can occur at any fraction of the timing cycle up to a full timing cycle later.

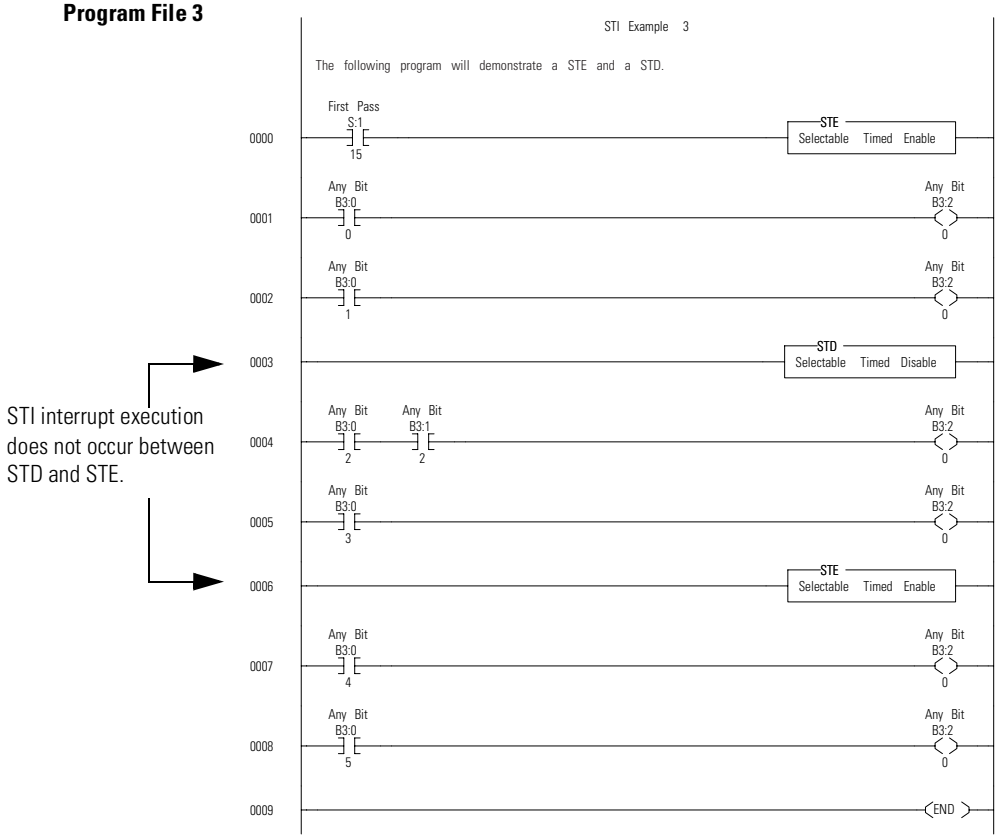
STD/STE Zone Example

In the program that follows, the STI function is in effect. The STD and STE instructions in rungs 6 and 12 are included in the ladder program to avoid having STI subroutine execution at any point in rungs 7 through 11.

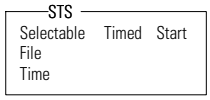
The STD instruction (rung 6) resets the STI enable bit and the STE instruction (rung 12) sets the enable bit again. The STI timer increments and may time out in the STD zone, setting the pending bit S:2/0 and overrun bit S:5/10.

The first pass bit S:1/15 and the STE instruction in rung 0 are included to insure that the STI function is initialized following a power cycle. You should include this rung any time your program contains an STD/STE zone or an STD instruction.

Program File 3



Selectable Timed Start (STS)



Fixed	SLC 5/01	SLC 5/02	SLC 5/03	SLC 5/04	SLC 5/05
		•	•	•	•

Use the STS instruction to condition the start of the STI timer upon entering the REM Run mode - rather than starting automatically. You can also use it to set up or change the file number or setpoint/frequency of the STI routine that is executed when the STI timer expires.

This instruction is not required to configure a basic STI interrupt application.

The STS instruction requires you to enter two parameters, the STI file number and the STI setpoint. Upon a true execution of the rung, this instruction enters the file number and setpoint in the status file (S:31, S:30), overwriting the existing data. At the same time, the STI timer is reset and begins timing; at timeout, the STI subroutine execution occurs. When the rung goes false, the STI function remains enabled at the setpoint and file number you've entered in the STS instruction.

TIP

SLC 5/03 and higher processors: The STS instruction uses the setting of the STI resolution bit S:2/10 to determine the timebase to be used upon STS instruction execution.

Discrete Input Interrupt Overview

Fixed	SLC 5/01	SLC 5/02	SLC 5/03	SLC 5/04	SLC 5/05
			•	•	•

Use the Discrete Input Interrupt (DII) for high-speed processing applications or any application that needs to respond to an event quickly. This instruction allows the processor to execute a ladder subroutine when the input bit pattern of a discrete I/O card matches a compare value that you programmed.

The status file contains six bit values and six word values used to program and monitor the DII function. The DII does not require ladder logic instructions for configuration. You program the DII to examine the input bit pattern of the first bits of any single I/O slot, which contains any discrete input card (such as IG16, IV16, IB8, IB32). When the input bit pattern matches the compare value, the accumulator is incremented. The DII accumulator counts to the preset value and, once the interrupt is generated, it immediately wraps around and begins counting again at zero.

While scanning the DII subroutine, you can reconfigure the DII to look for an entirely different event. This facilitates DII sequencing. The DII can be programmed to compare each input point to either a high (1) or low (0) state. The accumulator is incremented on the input transition that causes the input points to match the compare value.

IIM or IOM instructions are needed in the DII subroutine if your application requires immediate update of input or output points. End the DII subroutine with an RET instruction.

Basic Programming Procedure for the DII Function

To use the DII function with your main program file, do the following:

1. Create a subroutine file (range is from 3 to 255) and enter the desired ladder rungs. This is your DII subroutine file.
   (Project Tree - Controller Folder - Processor Status File - DII Tab)

2. Enter the Input Slot number (word S:47).

3. Enter the Bit Mask (word S:48). Valid Bit Masks range from 0 to 255.

4. Enter the Compare Value (word S:49). Valid Compare Values range from 0 to 255.

5. Enter the Preset Value (word S:50). Valid Preset Values range from 0 to 32,767.

6. Enter the DII subroutine File Number in word S:46 of the status file. (See page B-49.) A zero value disables the DII function.

TIP

PLC users: The main difference between the DII and the PLC 5/40 PII is that the DII requires all stated transitions to occur prior to generating a count, while the PII requires that only one of the stated transitions occur. Also, the PLC term "count" is referred to as "preset" in the DII.

Example

The DII can be programmed to count items on a high-speed conveyer. Each time 100 items pass a photo-switch, the DII subroutine is executed. The DII subroutine then uses Immediate I/O instructions to package the products.

Operation

After you download your program and enter the REM Run mode, the DII begins operation as follows:

Counter Mode

This mode is active when the Preset Value (S:50) contains a value greater than 1.

1. The DII reads the first byte of input data of a selected discrete input card at least once every 100µs.(1) Note that this "polling" of the input data has no effect on processor scan time.

(1) You must add interrupt latency time to the final transition or count that causes the interrupt subroutine to execute.

2. When the input data matches the programmed masked value, the accumulator is incremented by one. The next count occurs when input data transitions to non-matched and then back to matched.

3. When the accumulator reaches or exceeds the preset value, between 1 and 32,767, the interrupt is generated and the accumulator is reset to zero.

4. The DII subroutine is executed.

5. The cycle repeats.

Event Mode

This mode is active when the preset value (S:50) contains a 0 or 1.

1. The DII reads the first byte of input data of a selected discrete input card at least once every 100µs.(1) Note that this "polling" of the input data has no effect on processor scan time.

2. When the input data matches the programmed masked value, the interrupt is generated.

3. The DII subroutine is executed.(2)

4. The cycle repeats.(1)

DII Subroutine Content

For identification of your DII subroutine, use the INT instruction as the first instruction in your first rung.

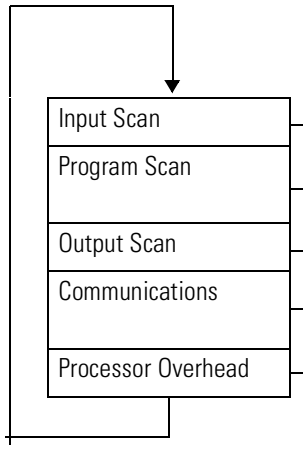
The DII subroutine contains the rungs of your application logic. You can program any instruction inside the DII subroutine except a TND, REF, or SVC instruction. IIM or IOM instructions are needed in a DII subroutine if your application requires immediate update of input or output points. End the DII subroutine with an RET instruction.

JSR stack depth is limited to 3. You may call other subroutines to a level 3 deep from an DII subroutine.

(1) You must add interrupt latency time to the final transition or count that causes the interrupt subroutine to execute.
(2) The DII continues to compare the input data to the programmed masked value while executing the DII subroutine.

Interrupt Latency and Interrupt Occurrences

Interrupt latency is the interval between DII detection and the start of the interrupt subroutine. DII interrupts can occur at any point in your program, but not necessarily at the same point on successive interrupts. Interrupts can occur between instructions in your program, inside the I/O scan (between slots), or between the servicing of communications packets. The table below shows the interaction between an interrupt and the processor operating cycle.



	DII	DII with Bit S:33/8 set	DII with Bit S:33/8 cleared
Input Scan	Between slot updates	Between word updates	Between slot updates
Program Scan	Between instruction updates	Between word updates	Between rung updates
Output Scan	Between slot updates	Between word updates	Between slot updates
Communications	Between communication packets	Between word packet updates	Between communication packets
Processor Overhead	At start and end	Between word updates	Between word updates

Events in the Processor Operating Cycle

If an interrupt occurs while the SLC 5/03 (or higher) processor is performing a multi-word slot update and your interrupt subroutine accesses that same slot, the multi-word transfer completes prior to performing the interrupt subroutine slot access.

Note that DII execution time adds directly to the overall scan time. During the latency period, the processor is performing operations that cannot be disturbed by the DII interrupt function. The Interrupt Latency Control Bit (S:33/8) functions as follows:

- When the bit is set (1) interrupts are serviced within the minimum time possible. The time will vary depending upon which processor and communication protocol you are using.
- The default state is cleared (0). When S:33/8 is clear (0), user interrupts occur between rungs and I/O slot updates. To determine the interrupt latency with S:33/8 clear, you must calculate the execution time of each and every rung of your program, then the add the execution time of the longest rung to the latency time.

Interrupt Priorities

Interrupt priorities for the SLC 5/03 and higher processors are:

1. User fault routine
2. Discrete Input Interrupt (DII)
3. STI Subroutine
4. I/O Interrupt Subroutine

An executing interrupt subroutine can only be interrupted by the fault routine.

TIP Under certain conditions, though, it is possible for a lower priority task to run during the DII execution.

Status File Data Saved

Data in the following words is saved on entry to the DII subroutine and re-written upon exiting the DII subroutine.

- S:0 Arithmetic flags
- S:13 and S:14 Math register
- S:24 Index register

Reconfigurability

You can reconfigure the DII entirely or in part, depending on the particular parameter(s) you choose. You can reconfigure some of the parameters simply by writing the new value over the old value. Other values require you to set the reconfiguration bit in addition to writing the new value. The DII is non-retentive and always reconfigures itself upon entry into the REM Run mode. Refer to "DII Parameters" for details on reconfiguring each parameter.

Example

The DII can be programmed to count items on a high-speed conveyer. Each time 100 items pass a photo-switch, the DII subroutine is executed. The DII subroutine then uses Immediate I/O instructions to package the products.

If you want to vary the number of items that are packaged together, simply change the number in the DII preset parameter using a MOV instruction.

DII Parameters

The following parameters are associated with the DII function. These parameters have status file addresses that are described here and also in Appendix B.

- DII Pending Bit (S:2/11) - When set, this bit indicates that the DII Accumulator (S:52) equals the DII preset (S:50) and the ladder file number specified by the DII File Number (S:46) is waiting to be executed. It is cleared when the DII File Number (S:46) begins executing, or on exit from the REM Run or REM Test mode.
- DII Enable Bit (S:2/12) - To program this feature, use the data monitor function to set/clear this bit, or address this bit with your ladder program. This bit is set in its default condition. If set, it allows execution of the DII subroutine if the DII File Number (S:46) is non-zero. If clear, when the interrupt occurs, the DII subroutine will not execute and the DII Pending bit is be set. The DII function continues to run anytime the DII file (S:46) is non-zero. If the pending bit is set, the enable bit is examined at the next end of scan.
- DII Executing Bit (S:2/13) - When set, this bit indicates that the DII interrupt has occurred and the DII subroutine is currently being executed. This bit is cleared on completion of the DII routine, power-up, or REM Run mode entry.
- DII Overflow Bit (S:5/12) - This bit is set whenever the DII interrupt occurs while still executing the DII subroutine or whenever the DII interrupt occurs while pending or disabled.
- Reconfigure Bit (S:33/10) - When this bit is set (1), it indicates that at the next end of scan (END, TND, or REF), fault routine exit, STI ISR exit, Event ISR exit, or next DII ISR exit the:
  - DII accumulator is cleared,
  - values at status words S:47 to S:50 are applied,
  - the pending bit is cleared, and
  - the DII Reconfigure bit is cleared.
- DII Lost Bit (S:36/8) - This bit is set if a DII interrupt occurs while the DII Pending bit is set.

- File Number (Word S:46) - Enter a program file number (3 to 255) to be used as the discrete input interrupt subroutine. Write a 0 value to disable the function. This value is applied upon detection of a DII Reconfigure bit, each DII ISR exit, and each end of scan (END, TND, or REF).
- Slot Number (Word S:47) - You enter the slot number (1 to 30) to be used as the discrete input interrupt subroutine. A zero value disables the function. This value is applied on detection of the DII Reconfigure bit, or on entry into the REM Run mode.
- Bit Mask (Word S:48) - You enter the bit-mapped value that corresponds to the bits you wish to monitor on the discrete I/O module (0 to 255). Only bits 0 to 7 are used in the DII function. Setting a bit indicates that you wish to include the bit in the comparison of the discrete I/O card's bit pattern to the DII compare value (S:49). This value is applied on detection of the DII Reconfigure bit, each DII ISR exit, and at each end of scan (END, TND, or REF).
- Compare Value (Word S:49) - You enter a bit-mapped value that corresponds to the bit pattern that must occur on the discrete I/O card for a count or interrupt to occur (0 to 255). Only bit 0 to 7 are used in the DII function. The bit must be set (1) or cleared (0) in order to satisfy the compare condition for that bit. An interrupt or count is generated upon the last bit transition of the compare value. This value is applied on detection of DII Reconfigure bit, each DII ISR exit, and at each end of scan (END, TND, or REF).

  To provide protection from inadvertent data monitor alteration of your selection, program an unconditional MOV instruction containing the compare value of the DII to S:49.

- Preset (Word S:50) - When this value is equal to 0 or 1, an interrupt is generated each time the comparison specified in words S:48 and S:49 is satisfied. When this value is between 2 and 32767, a count occurs each time the bit comparison is satisfied. An interrupt is generated when the accumulator value reaches 1 or exceeds the preset value. This value is applied on detection of DII Reconfigure bit, each DII ISR exit, and at each end of scan (END, TND, or REF).

  To provide protection from inadvertent data monitor alteration of your selection, program an unconditional MOV instruction containing the preset value of the DII to S:50.

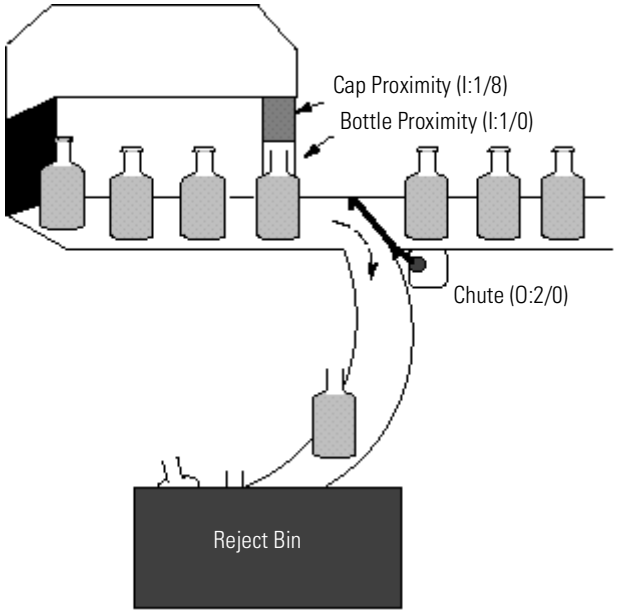
- Return Mask (Word S:51) - The Return Mask is updated immediately preceding entry into the DII subroutine. This value contains the bit map of the last bit transition that caused the interrupt. If more than one bit transitions in the same 100µs DII sample period, it is included in the return mask. This bit is cleared by the processor on exit from the DII subroutine. Use this value to validate the last interrupt transition that caused the input pattern to match the compare value. Or when dynamically reconfiguring (sequencing) the DII, use this value inside of your DII's subroutine to help determine/validate its position of the sequence.
- Accumulator (Word S:52) - The DII accumulator contains the number of counts that have occurred. When a count occurs and the accumulator is greater than or equal to the preset value, a DII interrupt is generated and the accumulator is cleared.

  For applications that measure the rate of incoming DII pulses while using a STI (Selectable Timed Interrupt), SLC 5/03 OS301 and above updates the DII accumulator prior to executing the first rung of the STI subroutine.

Discrete Input Interrupt Application Example

The following example shows how to use the Discrete Input Interrupt to control a high-speed application. In the example, the DII is used to ensure that all bottles exiting a filling and capping machine have their caps installed.

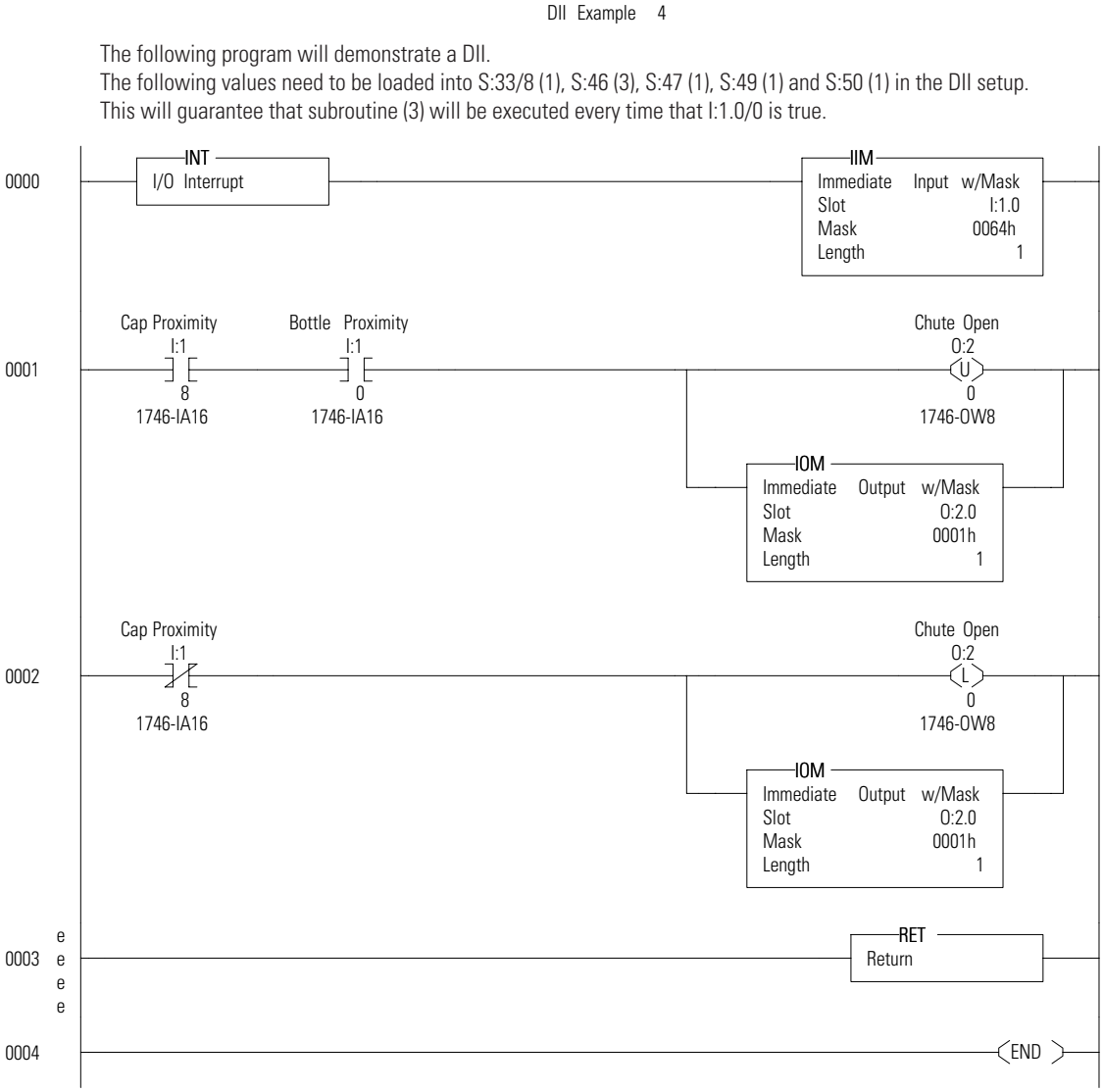
The bottle proximity switch is used as the DII input. When a bottle passes the proximity switch, the processor executes the DII subroutine. In the subroutine the processor reads the state of the cap proximity switch. If the cap is installed, the chute solenoid does not energize; allowing the bottle to continue down the line. If the cap is missing, the chute solenoid energizes, causing the defective bottle to divert down the chute and into the reject bin.



The following parameters are used to program the DII for the above application:

- S:33/8 Interrupt Latency Control Bit = 1
- S:46 File = 3
- S:47 Slot = 1
- S:48 Mask = 00000001
- S:49 Compare = 00000001
- S:50 Preset = 1

Ladder Diagram for the Bottling Application



Refer to Appendix G for another application example using the DII to count pulses from an encoder.

I/O Interrupt Overview

This function allows a specialty I/O module to interrupt the normal processor operating cycle in order to scan a specified subroutine file. Interrupt operation for a specific module is described in the user's manual for the module.

Not all specialty I/O modules are capable of generating I/O interrupts. Refer to the user manual of the specific specialty I/O module to see if it supports this feature. For example, you cannot use a standard discrete I/O module to accomplish an I/O event-driven interrupt.

Fixed	SLC 5/01	SLC 5/02	SLC 5/03	SLC 5/04	SLC 5/05
		•	•	•	•

This section describes:

- I/O operation
- I/O interrupt parameters
- IID and IIE instructions
- RPI instruction
- INT instruction

Basic Programming Procedure for the I/O Interrupt Function

- When you are configuring the specialty I/O module slot with the programming device, make sure you program the "ISR" (interrupt subroutine) program file number (range 3 to 255) that you want the processor to execute when the module generates an interrupt. Specialty I/O modules that create interrupts should be configured in the lowest numbered I/O slots.
- Create the subroutine file that you have specified as the ISR number in the I/O module slot configuration.

Operation

When you restore your program and enter the REM Run mode, the I/O interrupt begins operation as follows:

1. The specialty I/O module determines that it needs servicing and generates an interrupt request to the SLC processor.
2. The processor is interrupted from what it is doing, and the specified interrupt subroutine file (ISR) is scanned.
3. When the ISR scan is completed, the specialty I/O module is notified. This informs the specialty I/O module that it is allowed to generate a new interrupt.
4. The processor resumes normal operation from where it left off.

Interrupt Subroutine (ISR) Content

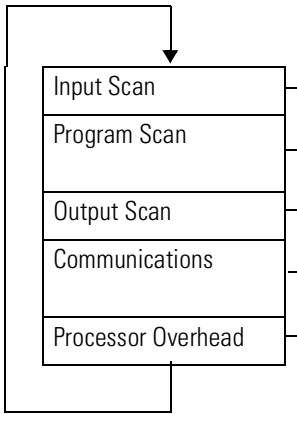
The Interrupt Subroutine (INT) instruction should be the first instruction in your ISR. This identifies the subroutine file as an I/O interrupt subroutine.

The ISR contains the rungs of your application logic. You can program any instruction inside an ISR except a TND, REF, or SVC instruction. IIM or IOM instructions are needed in an ISR if your application requires immediate update of input or output points. Terminate the ISR with an RET (return) instruction.

JSR stack depth is limited to 3. That is, you may call other subroutines to a level 3 deep from an ISR.

Interrupt Latency and Interrupt Occurrences

Interrupt latency is the interval between the I/O module's request for service and the start of the interrupt subroutine. I/O interrupts can occur at any point in your program, but not necessarily at the same point on successive interrupts. Interrupts can only occur between instructions in your program, inside the I/O scan (between slots), or between the servicing of communication packets. The following table shows the interaction between an interrupt and the processor operating cycle.



	SLC 5/02 I/O Interrupts	5/03 and Higher I?O Interrupts with Bit S:33/8 set	5/03 and Higher I/O Interrupts with Bit S:33/8 cleared
Input Scan	Between slot updates	Between word updates	Between slot updates
Program Scan	Between instruction updates	Between word updates	Between rung updates
Output Scan	Between slot updates	Between word updates	Between slot updates
Communications	Between communication packets	Between word packet updates	Between communication packets
Processor Overhead	At start and end	Between word updates	Between word updates

Events in the Processor Operating Cycle

Note that ISR execution time adds directly to the overall scan time. During the latency period, the processor is performing operations that cannot be disturbed by the ISR interrupt function. Latency periods are:

- SLC 5/02 interrupts are serviced within 2.4ms maximum.
- SLC 5/03 and higher processors: If an interrupt occurs while the processor is performing a multi-word slot update and your interrupt subroutine accesses that same slot, the multi-word transfer finishes to completion prior to performing the interrupt subroutine slot access. The Interrupt Latency Control bit (S:33/8) functions as follows:
  - When the bit is set (1) interrupts are serviced within the interrupt latency time.
  - When S:33/8 is clear (0), user interrupts occur between rungs and I/O slot updates.

The default state is cleared (0). To determine the interrupt latency with S:33/8 clear, you must calculate the execution time of each and every rung in your program.

Interrupt Priorities

Interrupt priorities are as follows:

SLC 5/02 Processor	SLC 5/03 and Higher Processors
1. Fault Routine	1. Fault Routine
2. STI Subroutine	2. Discrete Input Interrupt (DII)
3. I/O Interrupt Subroutine (ISR)	3. STI Subroutine
	4. I/O Interrupt Subroutine (ISR)

An executing interrupt can only be interrupted by an interrupt having higher priority. The I/O interrupt cannot interrupt an executing fault routine, an executing DII subroutine, an executing STI subroutine, or another executing I/O interrupt subroutine. If an I/O interrupt occurs while the fault routine, DII, or STI subroutine is executing, the processor waits until the higher priority interrupts are scanned to completion. The I/O interrupt subroutine is then scanned.

TIP

SLC 5/02 specific: It is important to understand that the I/O pending bit associated with the interrupting slot remains clear during the time that the processor is waiting for the fault routine or STI subroutine to finish.

SLC 5/03 and higher processors: The I/O pending bit is always set when the interrupt occurs. You can examine the state of these bits within your higher priority interrupt routines.

If a major fault occurs while executing the I/O interrupt subroutine, execution immediately switches to the fault routine. If the fault was recovered by the fault routine, execution resumes at the point that it left off in the I/O interrupt subroutine. Otherwise, the fault mode is entered.

If a DII interrupt occurs while executing the I/O interrupt subroutine, execution immediately switches to the DII subroutine. When the DII subroutine is scanned to completion, execution resumes at the point that it left off in the I/O interrupt subroutine.

If the STI timer expires while executing the I/O interrupt subroutine, execution immediately switches to the STI subroutine. When the STI subroutine is scanned to completion, execution resumes at the point that it left off in the I/O interrupt subroutine.

If two or more I/O interrupt requests are detected by the processor at the same instant, or while waiting for a higher or equal priority interrupt subroutine to finish, the interrupt subroutine associated with the specialty I/O module in the lowest slot number is scanned first. For example, if slot 2 (ISR 20) and slot 3 (ISR 11) request interrupt service at the same instant, the processor first scans ISR 20 to completion, then ISR 11 to completion.

Status File Data Saved

Data in the following words is saved on entry to the I/O interrupt subroutine and re-written upon exiting the I/O interrupt subroutine.

- S:0 Arithmetic flags
- S:13 and S:14 Math register
- S:24 Index register

I/O Interrupt Parameters

The I/O interrupt parameters below have status file addresses. They are described here and also in Appendix B of this manual.

- ISR Number - Specifies the subroutine file number that will be executed when an I/O interrupt is generated by an I/O module. The ISR Numbers are not part of the status file, but they are part of the I/O configuration for each slot in the SLC system.
- I/O Slot Enables (Words S:11 and S:12) - These words are bit mapped to the 30 I/O slots. Bits S:11/1 through S:12/14 refer to slots 1 through 30. Bits S:11/0 and S:12/15 are reserved.

  The enable bit associated with an interrupting slot must be set when an interrupt occurs. Otherwise a major fault will occur. Changes made to these bits using the Data Monitor function take effect at the next end of scan.

- I/O Interrupt Pending Bits (Words S:25 and S:26) - These words are bit mapped to the 30 I/O slots. Bits S:25/1 through S:26/14 refer to slots 1 through 30. Bits S:25/0 and S:26/15 are reserved. The pending bit associated with an interrupting slot is set when the corresponding I/O slot interrupt enable bit is clear at the time of an interrupt request. It is cleared when the corresponding I/O event interrupt enable bit is set, or when an associated RPI instruction is executed. The pending bit for an executing I/O interrupt subroutine remains clear when the ISR is interrupted by a DII, STI, or fault routine.

  SLC 5/02 specific: Likewise, the pending bit remains clear if interrupt service is requested at the time that a higher or equal priority interrupt is executing (fault routine, STI, or other ISR).

  SLC 5/03 and higher processors: This bit is set if interrupt service is requested at the time a higher or equal priority interrupt is executing (fault routine, DII, STI, or other ISR).

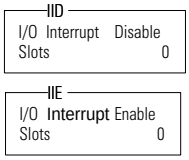
- I/O Interrupt Enables (Words S:27 and S:28) - These words are bit mapped to the 30 I/O slots. Bits S:27/1 through S:28/14 refer to slots 1 through 30. Bits S:27/0 and S:28/15 are reserved. The enable bit associated with an interrupting slot must be set when the interrupt occurs to allow the corresponding ISR to execute. Otherwise the ISR will not execute and the associated I/O slot interrupt pending bit will be set.

  SLC 5/02 specific: Changes made to these bits using the data monitor function or ladder instruction take effect at the next end of scan.

SLC 5/03 and higher processors: Changes made to these bits using the data monitor function or ladder instruction take effect immediately.

- I/O Interrupt Executing (Word S:32) - This word contains the slot number of the specialty I/O module that generated the currently executing ISR. This value is cleared upon completion of the ISR, run mode entry, or upon power up. You can interrogate this word inside of your DII or STI subroutine or fault routine if you wish to know if these higher priority interrupts have interrupted an executing ISR. You may also use this value to discern interrupt slot identity when multiplexing two or more specialty I/O module interrupts to the same ISR.

I/O Interrupt Disable (IID) and I/O Interrupt Enable (IIE)



Fixed	SLC 5/01	SLC 5/02	SLC 5/03	SLC 5/04	SLC 5/05
		•	•	•	•

These instructions are generally used in pairs to prevent I/O interrupts from occurring during time-critical or sequence-critical portions of your main program or subroutine. The I/O Event-Driven Interrupt function is used with specialty I/O modules capable of generating an interrupt.

Use these instructions together to create a zone in your main ladder program file or subroutine file in which I/O interrupts cannot occur. Both instructions take effect immediately upon execution. You must specify a subroutine to be executed upon receipt of such an interrupt.

SLC 5/02 specific: Setting/clearing the I/O interrupt enable bits (S:27 and S:28) with a programming device or standard instruction such as MVM takes effect at the END of the scan only.

SLC 5/03 and higher processors: Setting/clearing the I/O interrupt enable bits (S:27 and S:28) with a programming device or standard instruction such as MVM takes effect immediately.

IID Operation

When true, this instruction clears the I/O interrupt enable bits (S:27/1 through S:28/14) corresponding to the slots parameter of the instruction. Interrupt subroutines of the affected slots are not able to execute when an interrupt request is made. Instead, the corresponding I/O pending bits (S:25/1 through S:26/14) are set. The ISR is not executed until an IIE instruction with the same slot parameter is executed, or until the end of the scan during which you use a programming device to set the corresponding status file bit.

IIE Operation

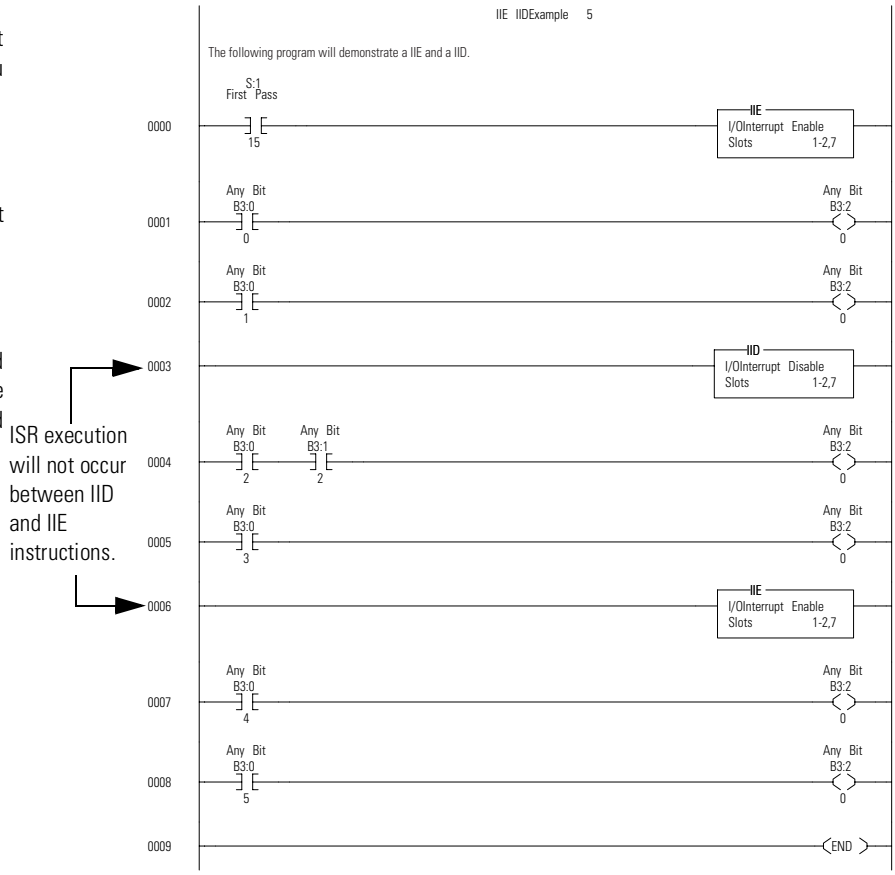
When true, this instruction sets the I/O interrupt enable bits (S:27/1 through S:28/14) corresponding to the slots parameter of the instruction. Interrupt subroutines of the affected slots regain the ability to execute when an interrupt request is made. If an interrupt was pending (S:25/1 through S:26/14) and the pending slot corresponds to the IIE slots parameter, the ISR associated with that slot executes immediately.

IID/IIE Zone Example

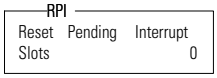
In the program below, slots 1, 2, and 7 are capable of generating I/O interrupts. The IID and IIE instructions in rungs 6 and 12 are included to avoid having I/O interrupt ISRs execute as a result of interrupt requests from slots 1, 2, or 7. This allows rungs 7 through 11 to execute without interruption.

The first pass bit S:1/15 and the IIE instruction in rung 0 are included to insure that the I/O interrupt function is initialized following a power cycle. You should include a rung such as this any time your program contains an IID/IIE zone or an IID instruction.

The IID instruction in rung 6 clears the I/O interrupt enable bits associated with slots 1,2, and 7 (S:27/1, S:27/2, and S:27/7). The IIE instruction in rung 12 sets these same bits. If an I/O interrupt is detected by the processor while the processor is executing rungs 7-11, the interrupt will be marked as pending. (S:25//1, S:25/2, and/or S:25/7 will be set.) All interrupts marked as pending are serviced upon execution of rung 12. The lowest numbered slot is serviced first when multiple pending bits are set.



Reset Pending Interrupt (RPI)



Fixed	SLC 5/01	SLC 5/02	SLC 5/03	SLC 5/04	SLC 5/05
		•	•	•	•

This instruction resets the pending status of the specified slots and informs the corresponding I/O modules that you have aborted their interrupt requests. This instruction is not required to configure a basic I/O interrupt application.

When true, this instruction clears the I/O pending bits (S:25/1 through S:26/14) corresponding to the slots parameter of the instruction. In addition, the processor notifies the specialty I/O modules in those slots that their interrupt request was aborted. Following this notice, the slot may once again request interrupt service. This instruction does *not* affect the I/O slot interrupt enable bits (S:27/1 through S:28/14).

Entering Parameters

Enter the I/O slot numbers (1 to 30) involved. Examples:

Slot #	Slot Indicated
6	indicates slot 6
6,8	indicates slots 6 and 8
6 to 8	indicates slots 6, 7, and 8
1 to 30	indicates all slots

Interrupt Subroutine (INT)

Fixed	SLC 5/01	SLC 5/02	SLC 5/03	SLC 5/04	SLC 5/05
		•	•	•	•

Use the INT instruction in I/O event-driven interrupt subroutines (ISRs) and STIs for identification purposes. Use of this instruction is optional.

This instruction has no control bits and is always evaluated as true. When used, the INT should be programmed as the first instruction of the first rung of the ISR.

SLC Communication Instructions

This chapter contains general information about the SLC communication instructions. Each of the instructions includes information on:

- what the instruction symbol looks like
- how to use the instruction
- an application example and timing diagrams

Table 12.1 Communication Instructions

Instruction Mnemonic	Instruction Name	Purpose	Page
SVC	Service Communications	When conditions preceding the SVC instruction in the rung are true, the SVC instruction interrupts the program scan to execute the service communication portion of the operating cycle.	12-2
MSG	Message Read/Write	This instruction transfers data from one node to another on the communication network. When the instruction is enabled, message transfer is pending. Actual data transfer takes place at the end of scan.	12-3

About the Communication Instructions

Use the SVC instruction to enhance communication performance of your processor. Use the Message (MSG) instruction to send and receive data from other processors and devices.

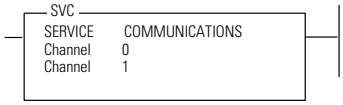
In this chapter you will find a general overview preceding each type of instruction:

- Service Communication instruction for SLC 5/02 and higher processors
- Message instruction for the SLC 5/02 and higher processors

Service Communications (SVC)

Fixed	SLC 5/01	SLC 5/02	SLC 5/03	SLC 5/04	SLC 5/05
	•	•	•	•	•

Output Instruction



Using an SLC 5/02 Processor

The SVC instruction is an output instruction that has no programming parameters. When it is evaluated as true, the program scan is interrupted to execute the service communications part of the operating cycle. The scan then resumes at the instruction following the SVC instruction. Use this instruction to enhance the communication performance of your SLC 5/02 processor.

You are not allowed to place an SVC instruction in an STI interrupt, I/O interrupt, or user fault subroutine.

Using SLC 5/03 and Higher Processors

When using SLC 5/03 and higher processors, the SVC instruction operates as described above. These processors also allow you to select a specific communication channel (0, 1, or both) to be serviced. You are not allowed to place an SVC instruction in a Fault, DII, STI, or I/O Event subroutine.

- SLC 5/03 processor
  - channel 0 is RS-232/DF1 Full-Duplex or Half-Duplex (master or slave), DH-485, or ASCII
  - channel 1 is DH-485
- SLC 5/04 processor
  - channel 0 is RS-232/DF1 Full-Duplex or Half-Duplex (master or slave), DH-485, or ASCII
  - channel 1 is DH+
- SLC 5/05 processor
  - channel 0 is RS-232/DF1 Full-Duplex or Half-Duplex (master or slave), DH-485, or ASCII
  - channel 1 is Ethernet

The following status bits allow you to customize or monitor communications servicing. Refer to Appendix B for more information on monitoring and configuring these communication servicing bits.

Channel 1		Channel 0[1]	
S:2/5	Incoming Command Pending Bit	S:33/0	Incoming Command Pending Bit
S:2/6	Message Reply Pending Bit	S:33/1	Message Reply Pending Bit
S:2/7	Outgoing Message Command Pending Bit	S:33/2	Outgoing Message Command Pending Bit
S:2/15	Communications Servicing Selection Bit	S:33/5	Communications Servicing Selection Bit
S:33/7	Message Servicing Selection Bit	S:33/6	Message Servicing Selection Bit

(1) SLC 5/03 and higher processors only.

Channel Servicing

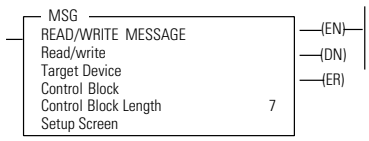
Whether a channel is selected to be serviced by the SVC instruction or not, that channel is still serviced normally at the end of the scan.

TIP

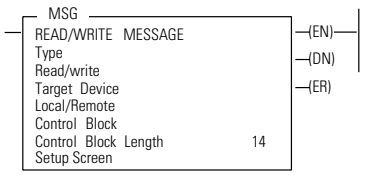
You may program the SVC instruction unconditionally across the rungs. This is the normal programming technique for the SVC instruction.

Message Instruction Overview

This is an output instruction that allows you to read or write data from one processor to another processor via the communication channel(s). The SLC 5/02 processor can service one message instruction at any given time. The SLC 5/03 and higher processors can service up to four message instructions per channel at a time, for a maximum of eight message instructions at any given time.



SLC 5/02



SLC 5/03 and higher

Fixed	SLC 5/01	SLC 5/02	SLC 5/03	SLC 5/04	SLC 5/05
		•	•	•	•

Output Instruction

To invoke the MSG instruction, toggle the MSG instruction rung from false-to-true. Do not toggle the rung again until the MSG instruction has successfully or unsuccessfully completed the previous message, indicated by the processor setting either the DN or ER bit.

Operation

SLC 5/02 - Although only one message instruction can be serviced at a time, the processor may hold several messages "enabled and waiting" (control block status bits EN and EW set). Waiting messages are serviced one at a time in sequential order.

Ladder logic should be included with every SLC 5/02 MSG instruction to time out the message in the event that the MSG starts transmitting successfully (MSG control block ST bit set), but the response is not received back in a reasonable amount of time. See Figure 12.1 and Figure 12.2 on how to use the MSG control block TO bit to accomplish this.

Figure 12.1 SLC 5/02 Messaging Example with MSG Timeout

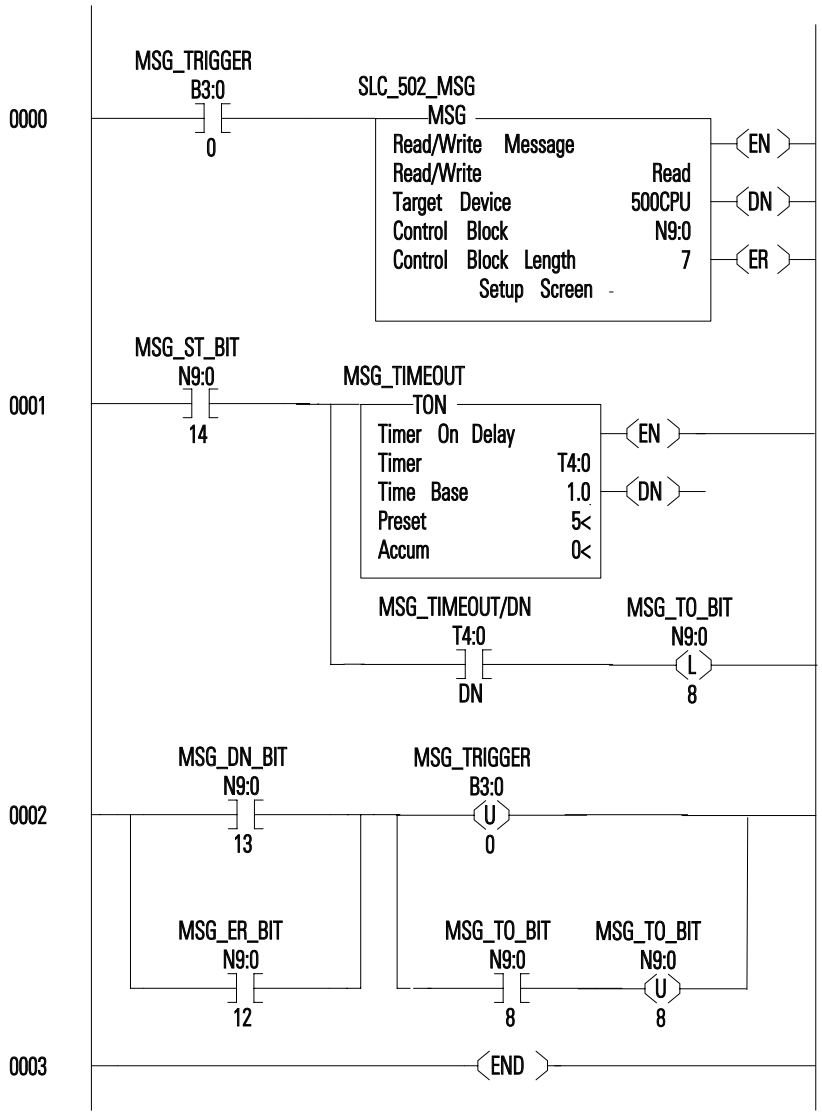
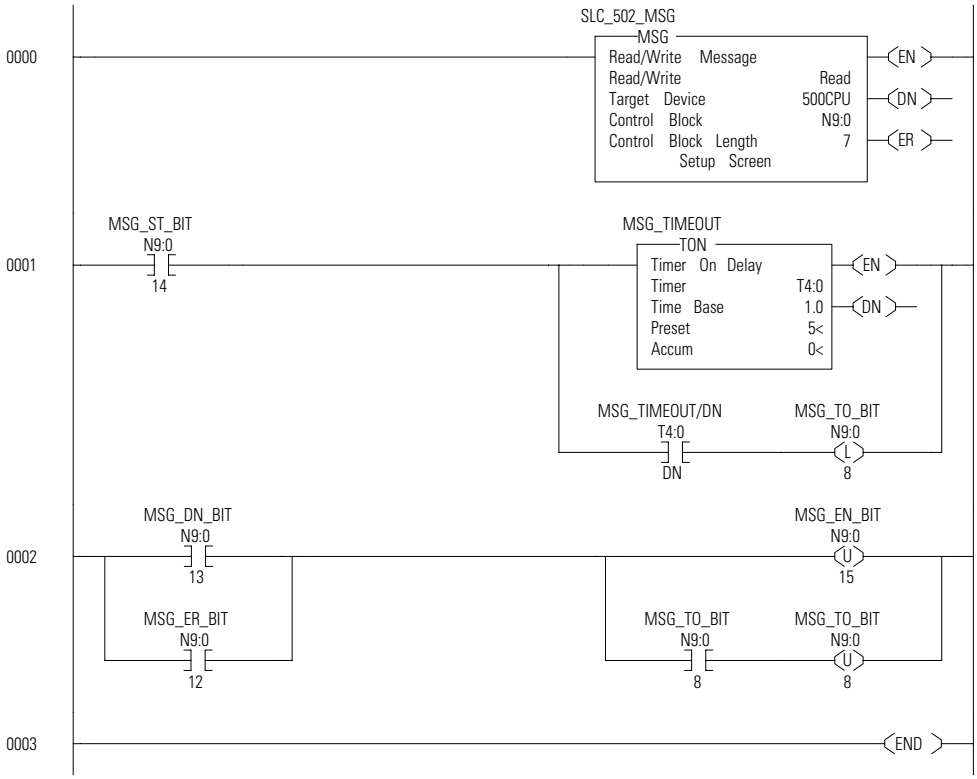


Figure 12.2 SLC 5/02 Repeating Messaging Example with MSG Timeout



SLC 5/03 and higher - If a MSG instruction has entered one of the four "channel dependent" transmission buffers and is waiting to be transmitted, its control block will have status bits EN and EW set. If more than four MSG instructions for that channel are enabled at one time, a "channel dependent" overflow queue is used to store the MSG instruction header blocks (not the data for a MSG write) from the fifth instruction to the fourteenth. These instructions, queued in a FIFO order, will only have control block status bit EN set.

If more than 14 MSG instructions are enabled at one time for any one channel, only control block status bit WQ is set, as there is no room available to currently queue the instruction. This instruction must be re-scanned with true rung conditions until space exists in the overflow queue.

TIP

If you consistently enable more MSG instructions than the buffers and queues can accommodate, the order in which MSG instructions enter the queue is determined by the order in which they are scanned. This means MSG instructions closest to the beginning of the program enter the queue regularly and MSG instructions later in the program may not ever enter the queue.

You can use the timeout control similar to the SLC 5/02 MSG instruction or use the built in timeout control (recommended). If the timeout value is set to 0, the functionality is similar to the SLC 5/02 MSG instruction. It differs in that once the TO bit is set, it will be reset automatically along with the ER bit on the next MSG rung false-to-true transition. We highly recommend setting the internal timeout value to something other than zero.

Figure 12.3 SLC 5/03, SLC 5/04 and SLC 5/05 Messaging Example

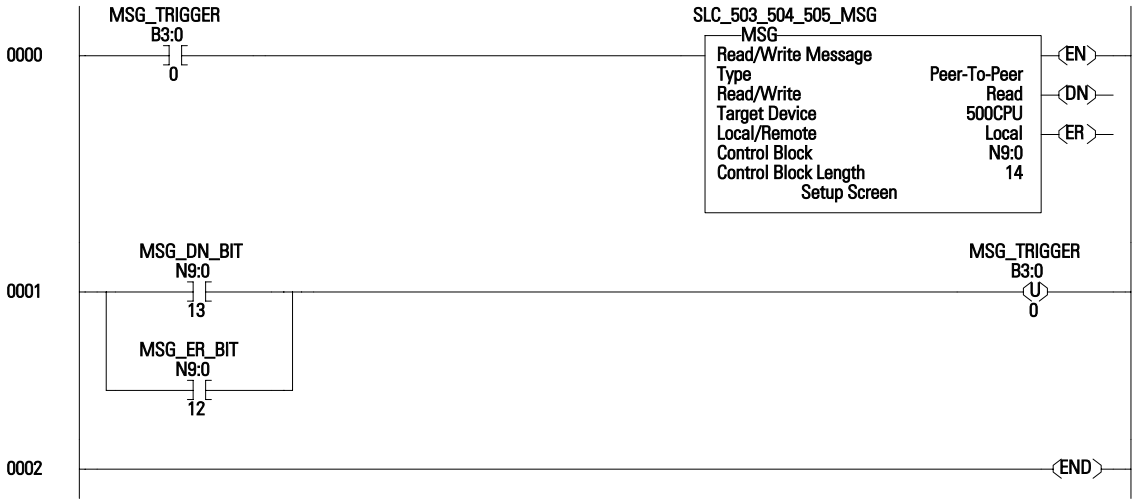
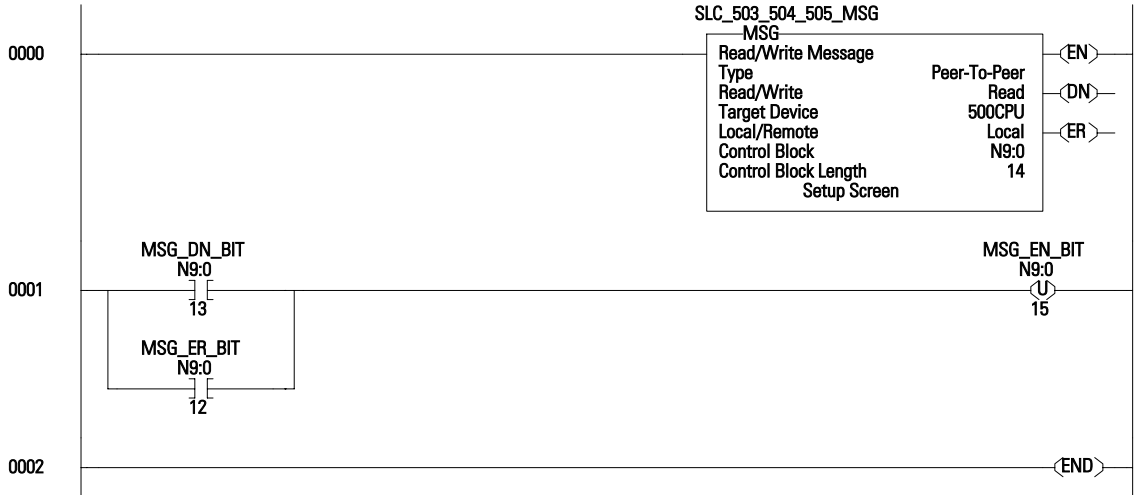


Figure 12.4 SLC 5/03, SLC 5/04 and SLC 5/05 Repeating Messaging Example



For the SLC 5/05 Channel 1 Ethernet, TCP/IP protocol is used to establish Ethernet connections, in order to send the MSG commands. Connections can be initiated by either a client program (INTERCHANGE or RSLinx application) or a processor.

The client program or processor must first establish a connection to the SLC 5/05 to enable the SLC 5/05 to *receive* solicited messages from a client program or another processor (incoming connection). The client program must also establish a connection to the SLC 5/05 to enable the SLC 5/05 to *send* unsolicited messages to a client program (outgoing connection).

In order to send a peer message, the SLC 5/05 must first establish a connection with the destination node at a specified IP address on the Ethernet network. A connection is established when a MSG instruction executes and no previous connection to the same device exists. When a MSG instruction executes, the SLC 5/05 checks to see whether a connection has been established with the destination node. If a connection has not been established, the SLC 5/05 attempts to establish a connection of the peer type. The connections are dedicated as follows:

Number of Connections[1]	Dedicated to:
4	peer messages (outgoing)
4	client messages (incoming)
8[2]	either peer or client messages

(1) Connections established by an INTERCHANGE client, RSLinx client, and peers are all included when counting the number of connections

(2) OS-501, Series C, FRN 5 and higher in the 32k (1747-L552) and 64k (1747-L553) processors support 16 "either peer or client messages", for a total of 24 connections.

IMPORTANT For peer connections, no more than one connection per destination node is established. If multiple MSG instructions use the same destination node, they share the same connection.

Available Configuration Options

The following configuration options are available on all SLC 5/02 and higher processors. Refer to Appendix D for valid parameters when programming the Message instruction.

- Peer-to-Peer Read/Write on a Local network to another SLC 500 processor
- Peer-to-Peer Read/Write on a Local network to a 485CIF device (PLC-2 emulation)

In addition, the following configuration options are available on all SLC 5/03 and higher processors.

- Peer-to-Peer Read/Write on a Local network to a PLC-5 processor
- Peer-to-Peer Read/Write on a Remote network to another SLC 500 processor
- Peer-to-Peer Read/Write on a Remote network to a 485CIF device (PLC-2 emulation)
- Peer-to-Peer Read/Write on a Remote network to a PLC-5 processor

In addition, the following configuration option is available on SLC 5/05 processors.

- Peer-to-peer Read/Write Multihop from Ethernet to ControlLogix processor or through ControlLogix gateway to ControlNet, DH+ or DH-485.

MSG Instruction Parameters

Enter the following parameters when programming this instruction:

- **Read/Write** - Read indicates that the local processor (processor in which the instruction is located) is receiving data; write indicates that it is sending data.
- **Target Device** identifies the type of device which responds to the MSG command. Valid options are:
  - 500CPU, if the target device is another SLC processor
  - 485CIF, if the target device is a PLC-2 emulator device
  - PLC-5, if the target device accepts PLC-5 commands(1)
- **Local** or **Remote** identifies if the message is sent to a device on a local network, or to a remote device on another network through a bridge. Valid options are:
  - Local, if the target device is on the local network
  - Remote, if the target device is on a remote network(1)
- **Control Block** is an integer file address that you select. It is a block of words, containing the status bits, target file address, and other data associated with the message instruction.
- **Control Block Length** is a display-only field that indicates how many integer file words are being used by the control block. See Table 12.B for possible control block lengths.

TIP

The MSG control block length increases from 7 to at least 14 words when changing from an SLC 5/02 to an SLC 5/03, SLC 5/04 or SLC 5/05 processor program. For ease of program portability between SLC processors, dedicate an integer file to each MSG instruction control block.

(1) SLC 5/03 and higher only.

Table 12.B MSG Instruction Control Block Lengths

	485 CIF	500 CPU	PLC 5	PLC 5 with Logical ASCII/Symbolic Addressing
SLC 5/02	7	7	Not Applicable	Not Applicable
SLC 5/03	14	14	14	56[1]
SLC 5/04	14	14	14	56[2]
SLC 5/05 Channel 0	14	14	14	56
SLC 5/05 Channel 1	51	51	51	93

(1) OS302, Series C, FRN 5 or higher with RSLogix 500 version 4.5 or higher.

(2) OS401,Series C, FRN 5 or higher with RSLogix 500 version 4.5 or higher.

MSG Setup Screen Parameters

Parameters for "This Controller":

- Data Table Address:
  - For a Read, this is the starting address which receives the data that is read from the target device.
  - For a Write, this is the starting address of the data which is written to the target device.
- Size in Elements:
  - Specifies the length of the message in elements. The maximum number of elements that are transferred via a MSG instruction is determined by the size of the destination data type.
  - For a Read, the data type in the local processor determines the maximum number of elements.
  - For a Write, the data type in the target device determines the maximum number of elements. The maximum number of elements that are transferred may be further limited based on the processor type. See Table 12.C for the maximum number of elements.

Table 12.C MSG Instruction Maximum Number of Elements

File Types	SLC 500, SLC 5/01, SLC 5/02	SLC 5/03, SLC 5/04, SLC 5/05 Channel 0	SLC 5/05 Channel 1
O, I, B, N	41	103	256
T, C, R	13	34[1]	256[2]
F	Not Applicable	51	256
St	Not Applicable	2	25
A	Not Applicable	103	256

(1) PLC-5 type timer element maximum is 20.

(2) PLC-5 type timer element maximum is 208.

- Channel[1]:

  Specifies the communication channel that is used to transmit the message request. Available channels:

  – SLC 5/03 - (Channel 0, RS-232) or (Channel 1, DH-485)
  – SLC 5/04 - (Channel 0, RS-232) or (Channel 1, DH+)
  – SLC 5/05 - (Channel 0, RS-232) or (Channel 1, Ethernet)

(1) SLC 5/03 and higher only.

Parameters for "Target Device":

- Message Timeout[1]:

  Specifies the length of the message timer in seconds. A timeout of 0 seconds means that there is no timer and a message that has been acknowledged waits indefinitely for a reply. Valid range is 0 to 254.

- Data Table Address[2]:
  - For a Read, this is the starting address where the data is being read from.
  - For a Write, this is the starting address where the data is being written to.
  - To enter in a PLC-5 type logical ASCII address, begin the address name with a $ and enclose the name in double quotes (example: "$N7:0"). To enter in a PLC-5 type symbolic address, enclose the name in double quotes (example: "READ_TAG").

TIP

You may use the PLC-5 type symbolic address to read/write controller tags in Logix controllers. The supported address forms are:

- "tag_name"
- "tag_name[x]"
- "tag_name[x,y]" or "tag_name[x][y]"
- "tag_name[x,y,z]" or "tag_name[x][y][z]"

Use INT data type for integer files or REAL data type for floating point files.

- Data Table Offset[3]:

  Specifies the word offset into an SLC 500 or MicroLogix Common Interface File (CIF) or byte offset into a PLC-5 or Logix PLC-2 compatibility file. Valid range is 0 to 255.

- Local Node Addr (dec)/(oct)[4] [5]:

(1) SLC 5/03 and higher only.

(2) 500 CPU and PLC-5 Target Devices only.

(3) 485 CIF Target Devices only.

(4) Local MSG only.

(5) All processors except SLC 5/05 Channel 1.

Specifies the node number of the target device that is receiving the message. Valid range is 0 to 31 (decimal)/0 to 37 (octal) for DH-485, 0 to 63 (decimal)/0 to 77 (octal) for DH+, or 0 to 254 (decimal)/0 to 376 (octal) for DF1.

- Ethernet (IP) Address [1]:

  Specifies the Ethernet IP address of the target device that is receiving the message. When using OS501, Series C, FRN 5, or higher firmware and RSLogix 500 version 5.20, or higher programming software, you may optionally enter in the device name (as defined on your local network's Domain Name System (DNS) servers), in place of the target device's Ethernet IP address. This device name must be no longer than 41 characters. You must have a PRimary Name Server and/or Secondary Name Server defined in the Channel 1 Ethernet configuration in order to successfully use this device name functionality. If you have a Default Domain Name defined in the Channel 1 Ehternet configuration, then the Default Domain Name is appended to the device name when requesting the corresponding IP address from the DNS server.

- MultiHop[2]:

  Specifies whether or not the message is either routed to a Logix controller or through a ControlLogix gateway. If yes, then the MSG route must be configured in the MultiHop tab of the MSG Setup Screen.

- Local Bridge Addr (dec)/(oct)[3] [4]:

  Specifies the node number of the bridge device on the local network. Valid range is 0 to 31 (decimal)/0 to 37 (octal) for DH-485, 0 to 63 (decimal)/0 to 77 (octal) for DH+, or 0 to 254 (decimal)/0 to 376 (octal) for DF1.

- Remote Bridge Addr (dec)[3]:

  Specifies the node number of the bridge device on the bridging network, when the bridge is configured for "gateway" mode. Otherwise, leave at 0.

- Remote Station Addr (dec)[3]:

(1) SLC 5/05 channel 1 no Multihop only.

(2) SLC 5/05 channel 1 only.

(3) Remote MSG only.

(4) All processors except SLC 5/05 Channel 1.

Specifies the node number of the target device on the remote network. Valid range is 0 to 31 (decimal) for DH-485, 0 to 63 (decimal) for DH+ or 0 to 254 (decimal) for DF1.

- Remote Bridge Link ID[3]:

  Specifies the link ID of the remote network where the target device resides. Valid range is 0 to 255.

MSG Setup Screen Status Bits

The column in the display below lists the various status bits associated with the SLC 500 MSG instruction.

ignore if timed out:	TO
to be retried:	NR
awaiting execution:	EW
continuous run:	CO
error:	ER
message done:	DN
message transmitting:	ST
message enabled:	EN
waiting for queue space:	WQ

- Timeout Bit TO (word 0, bit 08) Set this bit in your application to remove an active message instruction from processor control. You must use your own timeout control routine for the SLC 5/02 MSG instruction. (See Figure 12.1 on page 12-5.) or you may use the internal timeout control for SLC 5/03 and higher processors. For these processors, we recommend using the built in timeout control because it simplifies the user program.

  To utilize the internal timeout control, a value greater than 0 (default values are 5 seconds for SLC 5/03, SLC 5/04 and SLC 5/05 channel 0, and 23 seconds for SLC 5/05 channel 1) must be entered for the MSG instruction time-out parameter. A time-out value of 0 means no time-out value. In other words, if communication is interrupted, the processor will wait forever for a reply. If an acknowledgement is received (as indicated by the ST bit being set), but the reply is not received, the MSG instruction will appear to be locked up, although it is merely waiting for the reply.

  When a value greater than 0 is entered for the MSG time-out parameter and communication is interrupted, the MSG instruction will time-out and error after the time expires, allowing the user program to retry the same message if desired.

  With an SLC 5/02 MSG instruction, the ladder logic must reset the Timeout Bit before triggering the MSG instruction.

TIP When programming timeout control in SLC 5/03 and higher processors, omit the Timeout Bit manual reset rung.

- No Response Bit NR (bit 09) is set if the target processor responds to the MSG instruction that it can't process the message at the current time (for DH-485 and DH+ protocols only). This means that the MSG should be retried. The NR bit is reset when the ER or ST bit is set. Do not set or reset this bit. It is informational only.
- Enabled and Waiting Bit EW (bit 10) is set after the enable bit is set and the message is buffered and waiting to be sent in the buffer. Do not set or reset this bit. It is informational only.
- Continuous Operation CO (bit 11)[(1)] Set this bit if you wish to continually resend the MSG instruction. We recommend that internal timeout control be used for this option and the rung be unconditionally true. Use this bit to turn the mode on and off. A MSG instruction occupies one of the four channel transmission buffers when its CO bit is set. Therefore, a maximum of four MSG instructions per channel may have their CO bit set.

  This mode will continuously operate provided that the rung is continually scanned. If the instruction errors prior to the MSG timeout, it will automatically retry until it is successful. If the instruction errors due to a MSG timeout, the MSG stops triggering. The EN bit must be toggled off and back on to resume operation.

TIP If your program contains four message instructions assigned to the same channel with the Continuous Operation (CO) bit set, no other message instructions can be executed out that same channel, including message instructions which may be in the fault routine.

- Error Bit ER (bit 12) is set when message transmission has failed. The ER bit is reset the next time the associated rung goes from false to true. Do not set or reset this bit. It is informational only.

(1) SLC 5/03 and higher processors only.

- Done Bit DN (bit 13) is set when the message is transmitted successfully. The DN bit is reset the next time the associated rung goes from false to true. Do not set or reset this bit. It is informational only.
- Start Bit ST (bit 14) is set when the processor receives acknowledgment (ACK) from the target device. The ST bit is reset when the DN, ER, or TO bit is set. Do not set or reset this bit. It is informational only.

  For SLC 5/05 Ethernet (channel 1) communications, the ST bit indicates internally that the Ethernet daughterboard has received a command and it is acceptable for a transmission attempt. The command has not yet been transmitted.

- Enable Bit EN (bit 15) is set when rung conditions go true and there is space available in either the MSG buffer or MSG queue. It remains set until message transmission is completed and the rung goes false. You may reset this bit once either the ER or DN bit is set in order to retrigger a MSG instruction with true rung conditions on the next scan. Do not set this bit.
- Waiting for Queue Space Bit WQ (Word 7, bit 0)(1) is set when the queue is full. This bit is cleared when space is available in the active queue. Do not set or reset this bit. It is informational only.

TIP

When the WQ bit is set, or when only the EN bit is set, and you are using a MSG Write instruction, your source data is unbuffered. If you application requires buffered (or “snapshot”) data, wait until the EW bit is set before overwriting your source data.

- EN = 1 and EW = 1 when MSG gets in the buffer
- EN = 1 when MSG gets into queue
- WQ = 1 when queue (which holds 10 MSGs) is full:
  buffer - holds 4 messages with data
  queue - stores pointer (waiting list)

(1) SLC 5/03 and higher processors only.

MSG Instruction Control Block

Limitations for Manipulating the Control Block Bits

Do not manipulate the MSG instruction control block values except as noted below. For example, do not clear the first word of the control block, do not unlatch the time-out control bit (except in an SLC 5/02 MSG instruction), and so on.

The only MSG instruction control bits that may be manipulated by the ladder program without adversely affecting the operation of the instruction are the CO, EN, and TO bits. The enable bit (EN = bit 15) may be unlatched, but only when the done bit (DN = bit 13) or error bit (ER = bit 12) has been set, indicating the successful or unsuccessful completion of the previous message.

In addition, when a MSG is in progress and the ladder program wishes to terminate it for any reason, this may be done by enabling the time-out bit (TO = bit 8). The next time the processor scans the MSG instruction with the TO bit set, it will error the MSG (ER = 1). The MSG instruction may then be re-enabled with a false-to-true transition on the next program scan.

Control Block Layouts

The control block layout is shown below for 500CPU or PLC-5 as the target device:

.

Table 12.D Read or Write, Local or Remote(1) to a 500 CPU or PLC-5(1) (Without Logical ASCII/Symbolic Addressing)

	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	0
Word 0	EN	ST	DN	ER	CO(1)	EW	NR	TO	Error Code							
Word 1	Reserved								Target Node Address (Local)/Remote Station Address (Remote)							
Word 2	Number of Elements															
Word 3	File Number															
Word 4	File Type (S, B, T, C, R, N) (O, I, F, St, A)(1)															
Word 5	Element Number															
Word 6	Not Used															
Word 7(1)	Remote Bridge Address (Remote only)								Reserved (Internal Messaging Bits)							WQ
Word 8(1)	Reserved (Internal Messaging Bits)								Message Timer Preset							
Word 9(1)	Message Timer Scaled Zero															
Word 10(1)	Message Timer Accumulator															
Word 11(1)	Reserved (Internal Messaging Bits)															
Word 12(1)	AO=0	Reserved (Internal Messaging Bits)														
Word 13(1)	Reserved (Internal Messaging Bits)															

(1) SLC 5/03 and higher processors only.

The control block layout is shown below for 485CIF as the target device:

.

Table 12.E Read or Write, Local or Remote(1) to a 485 CIF

	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	0
Word 0	EN	ST	DN	ER	CO(1)	EW	NR	TO	Error Code							
Word 1	Not Used								Target Node Address (Local)/Remote Station Address (Remote)							
Word 2	Number of Words															
Word 3	Offset in Words															
Word 4	Not Used															
Word 5	Not Used															
Word 6	Not Used															
Word 7(1)	Remote Bridge Address (Remote only)								Reserved (Internal Messaging Bits)							WQ
Word 8(1)	Reserved (Internal Messaging Bits)								Message Timer Preset							
Word 9(1)	Message Timer Scaled Zero															
Word 10(1)	Message Timer Accumulator															
Word 11(1)	Reserved (Internal Messaging Bits)															
Word 12(1)	Reserved (Internal Messaging Bits)															
Word 13(1)	Reserved (Internal Messaging Bits)															

(1) SLC 5/03 and higher processors only.

Table 12.F Read or Write, Local or Remote to a PLC-5 (with Logical ASCII/Symbolic Addressing)[1] [2] [3]

	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	0
Word 0	EN	ST	DN	ER	CO	EW	NR	TO	Error Code							
Word 1	Not Used								Target Node Address (Local)/Remote Station Address (Remote)							
Word 2	Number of Elements															
Word 3	Not Used															
Word 4	File Type (based on local source or destination address)															
Word 5	Not Used															
Word 6	Not Used															
Word 7	Remote Bridge Address (Remote only)								Reserved (Internal Messaging Bits)							WQ
Word 8	Reserved (Internal Messaging Bits)								Message Timer Preset							
Word 9	Message Timer Scaled Zero															
Word 10	Message Timer Accumulator															
Word 11	Reserved (Internal Messaging Bits)															
Word 12	AO=1	Reserved (Internal Messaging Bits)														
Word 13	Reserved (Internal Messaging Bits)								Lower byte internal use.							
Word 14	Logical ASCII Address String Length including NULL Termination Character (bytes)															
Word 15	First Byte of Address String								Second Byte of Address String							
Word 16	Third Byte of Address String								...							
...	...								...							
...	...								...							
Word 55	Eighty-First Byte of ASCII Address String								NULL Byte of Longest ASCII Address String							

(1) SLC 5/05 Channel 0.

(2) SLC 5/04 OS401, Series C, FRN 5 or higher with RSLogix 500 version 4.5 or higher.

(3) SLC 5/03 OS302,Series C, FRN 5 or higher with RSLogix 500 version 4.5 or higher.

Table 12.G SLC 5/05 Channel 1 Read or Write, Local or Remote to an SLC 500 CPU or PLC-5 (without Logical ASCII/Symbolic Addressing)

	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
Word 0	EN	ST	DN	ER	CO	EW	NR	TO	Error Codes							
Word 1	Reserved								Remote Station Address (Remote only)							
Word 2	Number of Elements															
Word 3	File Number															
Word 4	File Type (based on local source or destination address)															
Word 5	Element Number															
Word 6	Not Used															
Word 7	Remote Bridge Address (Remote Only)								Reserved (Internal Messaging Bits)							WQ
Word 8	Reserved (Internal Messaging Bits)								Message Timer Preset							
Word 9	Message Timer Scaled Zero															
Word 10	Message Timer Accumulator															
Word 11	Reserved (Internal Messaging Bits)															
Word 12	AO= 0	Reserved (Internal Messaging Bits)														
Word 13	Reserved (Internal Messaging Bits)															
Word 14	First Byte of IP Address String(1)								Second Byte of IP Address String							
Word 15	Third Byte of IP Address String								...							
...	...								...							
...	...								...							
Word 34	Forty-First Byte of IP Address String								NULL Byte of Longest IP Address String							
Word 35(3)	Not Used								Ethernet Message Type = 0(2) or 2(3)							
Word 36(3)	ASA Service								Internal Object Identifier (IOI) Size in Words (1 to 5)							
Words 37 - 41(3)	ASA Internal Object Identifier (IOI)															
Word 42(3)	Not Used								Connection Path Size in Words (1 to 8)							
Words 43 - 50(3)	Connection Path															

(1) The IP Address format is up to 42 ASCII characters including a terminating NULL character. The first byte in the array is the left most character in the string as written. For example: If the IP Address is 423.156.78.012, the first byte is the ASCII character "4". If the MSG destination is an INTERCHANGE client on a host computer, the destination is specified as "client" and stored as a NULL terminated string.

(2) Not Multi-Hop MSG.

(3) Multi-Hop MSG.

Table 12.H SLC 5/05 Channel 1 Read or Write, Local or Remote to a PLC-5 (with Logical ASCII/Symbolic Addressing)

	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
Word 0	EN	ST	DN	ER	CO	EW	NR	TO	Error Code							
Word 1	Reserved								Remote Station Address (Remote only)							
Word 2	Number of Elements															
Word 3	Not Used															
Word 4	File Type (Based on Local Source or Destination Address)															
Word 5	Not Used															
Word 6	Not Used															
Word 7	Remote Bridge Address (Remote only)								Reserved (Internal Messaging Bits)							WQ
Word 8	Reserved (Internal Messaging Bits)								Message Timer Preset							
Word 9	Message Timer Scaled Zero															
Word 10	Message Timer Accumulator															
Word 11	Reserved (Internal Messaging Bits)															
Word 12	AO=1	Reserved (Internal Messaging Bits)														
Word 13	Reserved (Internal Messaging Bits)															
Word 14	Logical ASCII Address String Length including NULL Termination Character (bytes)															
Word 15	First Byte of ASCII Address String								Second Byte of ASCII Address String							
Word 16	Third Byte of ASCII Address String								...							
	...								...							
	...								...							
Word 55	Eighty-First Byte of ASCII Address String								NULL Byte of Longest ASCII Address String							
Word 56	First Byte of IP Address String[1]								Second Byte of IP Address String							
Word 57	Third Byte of IP Address String								...							
	...								...							
	...								...							
Word 76	Forty-First Byte of IP Address String								NULL Byte of Longest IP Address String							
Word 77	Not Used								Ethernet Message Type = 0[2] or 2[3]							
Word 78[3]	ASA Service								Internal Object Identifier (IOI) Size in Words (1 to 5)							
Words 79 - 83[3]	ASA Internal Object Identifier (IOI)															
Word 84[3]	Not Used								Connection Path Size in Words (1 to 8)							
Words 85 - 92[3]	Connection Path															

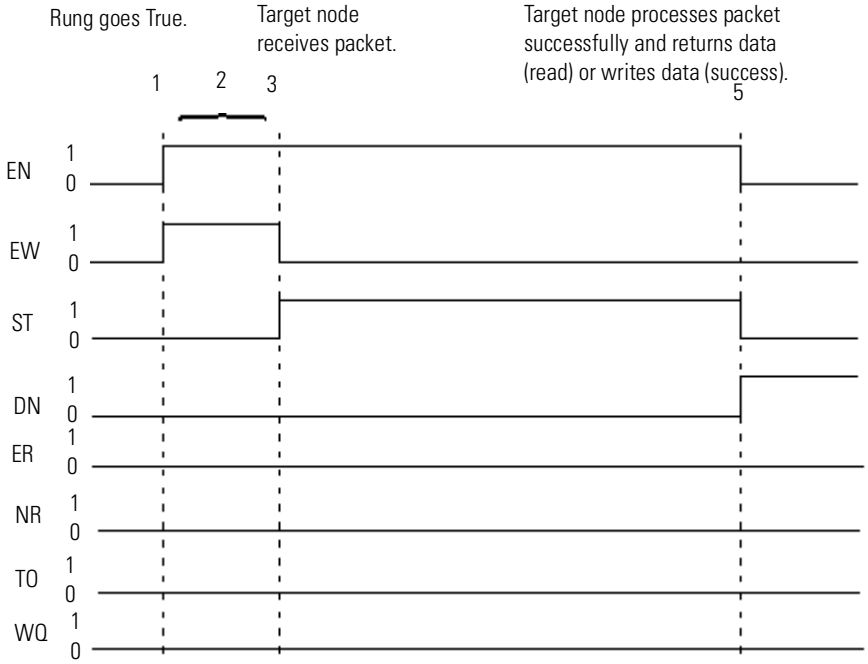
(1) The IP Address format is up to 42 ASCII characters including a terminating NULL character. The first byte in the array is the left most character in the string as written. For example: If the IP Address is 423.156.78.012, the first byte is the ASCII character "4". If the MSG destination is an INTERCHANGE client on a host computer, the destination is specified as "client" and stored as a NULL terminated string.

(2) Not Multi-Hop MSG.

(3) Multi-Hop MSG.

Timing Diagram for SLC 5/03, SLC 5/04, and SLC 5/05 MSG Instruction

The following section describes the timing diagram for a SLC 5/03, SLC 5/04, or SLC 5/05 MSG instruction.



1. When the MSG rung becomes true and the MSG is scanned, if there is room in any of the four active MSG buffers, the EN and EW bits are set. If this were a MSG Write instruction, the source data would be transferred to the MSG buffer at this time. If there is no room in the four MSG buffers, but a position is available in the 10-position MSG Queue, only the EN bit is set. The 10-position MSG Queue works on a first-in-first-out basis that allows the SLC processor to remember the order the MSG instructions were enabled. Note that the program does not have access to the SLC MSG Queue.

If there is no room in any of the four MSG buffers and no room in the 10-position MSG Queue, only the WQ bit is set. Note that when the WQ bit is set, the MSG instruction must be re-scanned with true rung conditions at a later time when there is room in either the four MSG buffers or the 10-position MSG Queue.

Once the EN bit is set, it remains set until the entire MSG process is complete and either the DN, ER, or TO bit is set. The MSG Timeout period begins timing when the EN bit is set. If the timeout period expires before the MSG instruction completes it function, the ER bit is set and an error code (37H) is placed in the MSG block to inform you of the timeout error.

If you choose to set the CO bit, your MSG instruction will "take up" permanent residence in one of the four active MSG buffers. The MSG instruction continues to re-transmit its data each time the DN or ER bit is set. If this were a MSG Write instruction, your source data would be updated each MSG cycle.

2. At the next end of scan or SVC, the SLC processor determines if it should examine the MSG Queue for "something to do." The processor bases its decision on the state of bits S:2/15, S:33/7, S:33/5, S:33/6, network communication requests from other nodes, and if previous MSG instructions are already in progress. If the SLC processor determines that it should not access the queue, the MSG instruction remains as it was. (Either the EN and EW bits remain set, or only the EN bit is set, or only the WQ bit is set until the next end of scan or SVC. If only the WQ bit is set, the MSG instruction must be re-scanned later with true rung conditions.)

If the SLC processor determines that it has "something to do," it unloads the MSG Queue entries into the MSG buffers until all four MSG buffers are full. Each MSG buffer contains a valid network packet. If a packet cannot be successfully built from the MSG Queue, the ER bit is set and a code is placed in the MSG block to inform you of an error. When a MSG instruction is loaded into a MSG buffer, the EN and EW bits are set.

The SLC processor then exits the end of scan or SVC portion of the scan. The processor's background communication function sends the packets to the Target Nodes that you specified in your MSG instruction. Depending on the state of bits S:2/15, S:33/7, S:33/5, and S:33/6 you can have up to eight MSG instructions active at any given time.

3. If the Target Node successfully receives the packet, it sends back an ACK (acknowledge). The ACK causes the processor to clear the EW bit and set the ST bit. The Target Node has not yet examined the packet, to see if it understands your request. Note that the Target Node is not required to respond within any given time frame.

For SLC 5/05 Ethernet communication, there is no ACK/NAK mechanism. The ST bit is set when the Ethernet daughterboard internally indicates it has received the command from the main processor and will send it out. Skip step 4 for SLC 5/05 processors.

TIP

If the Target Node faults or power cycles during this time frame of a MSG transaction, you will never receive a reply. This is why it is recommended to use a MSG Timeout value in your MSG instruction.

Step 4 not shown in the timing diagram.

4. If you do not receive an ACK, step 3 does not occur. Instead, either no response or a NAK (no acknowledge) is received. When this happens, the ST bit remains clear.

No response may be caused by:

- the target node is not there
- the target node does not respond because the packet became too corrupted in transmission to be properly received
- the response was corrupted in transmission back

A NAK may be caused by:

- target node is too busy
- target node received a corrupt packet.

When a NAK occurs, the EW bit is cleared and the NR bit is set for one scan. The next time the MSG instruction is scanned, the ER bit is set and the NR bit is cleared. This indicates that the MSG instruction failed. Note that if the Target Node is too busy, the ER bit is not set. Instead, the MSG instruction re-queues itself for re-transmission.

5. Following the successful receipt of the packet, the Target Node sends a reply packet. The reply packet will contain one of the following responses:

- I have successfully performed your write request.
- I have successfully performed your read request, and here is your data.
- I have not performed your request, you are in error.

At the next end of scan or SVC, following the Target Node's reply, the SLC processor examines the packet from the target device. If the reply contains "I have successfully performed your write request," the DN bit is set and the ST bit is cleared. The MSG instruction function is complete. If the MSG rung is false,

the EN bit is cleared the next time the MSG instruction is scanned.

If the reply contains "I have successfully performed your read request, and here is your data," the data is written to the data table, the DN bit is set and the ST bit is cleared. The MSG instruction function is complete. If the MSG rung is false, the EN bit is cleared the next time the MSG instruction is scanned.

If the reply contains "I have not performed your request, you are in error," the ER bit is set and the ST bit is cleared. The MSG instruction function is complete. If the MSG rung is false, the EN bit is cleared the next time the MSG instruction is scanned.

For SLC 5/03 (OS301 and higher), SLC 5/04, and SLC 5/05 processors, there are four MSG buffers per channel. Each channel has its own 10-position MSG Queue. The SLC processor unloads the two MSG queues into the MSG buffers evenly at end of scan or SVC. This allows both channels equal access to communications. If you program a SVC instruction that is configured to service only one channel, then only that channel will have its MSG Queue unloaded into the MSG buffers (until the next end of scan or SVC when both channels will again be unloaded evenly).

MSG Instruction Error Codes

When the processor detects an error during the transfer of message data, the processor sets the ER bit and enters an error code that you can monitor from your programming software.

Error Code	Description of Error Condition
02H	Target node is busy. The MSG instruction will automatically reload. If other messages are waiting, the message is placed at the bottom of the stack.
03H	Target node cannot respond because message is too large.
04H	Target node cannot respond because it does not understand the command parameters OR the control block may have been inadvertently modified.
05H	Local processor is offline (possible duplicate node situation).
06H	Target node cannot respond because requested function is not available.
07H	Target node does not respond.
08H	Target node cannot respond.
09H	Local modem connection has been lost.
0AH	Buffer unavailable to receive SRD reply.
0BH	Target node does not accept this type of MSG instruction.
0CH	Received a master link reset (one possible source is from the DF1 master).

Error Code	Description of Error Condition
10H	Target node cannot respond because of incorrect command parameters or unsupported command, or the data file specified does not exist.
11H	Local file has constant file protection.
12H	Local channel configuration protocol error exists.
13H	Local MSG configuration error in the Remote MSG parameters.
14H	Local communication driver is incompatible with the MSG instruction.
15H	Local channel configuration parameter error exists.
16H	Target or Local Bridge address is higher than the maximum node address.
17H	Local service is not supported.
18H	Broadcast (Node Address 255) is not supported.
19H	Improperly formatted Logical ASCII Address string. String not properly terminated with a NULL character or the string length does not match the value in the length parameter.
20H	Target Node responded with: Host has a problem and will not communicate.
30H	Target Node responded with: Remote station host is not there, disconnected, or shutdown.
37H	Message timed out in local processor.
38H	Message disabled pending link response.
40H	Target Node responded with: Host could not complete function due to hardware fault.
45h	Parameters of a reply to a MSG command do not match what was expected.
50H	Target node is out of memory.
60H	Target node cannot respond because file is protected.
70H	Target Node responded with: Processor is in Program Mode.
80H	Target Node responded with: Compatibility mode file missing or communication zone problem.
90H	Target Node responded with: Remote station cannot buffer command.
B0H	Target Node responded with: Remote station problem due to download.
C0H	Target Node responded with: Cannot execute command due to active IPBs.
D0H	No IP address configured for the network, -or- Bad command - unsolicited message error, -or- Bad address - unsolicited message error, -or- No privilege - unsolicited message error -or- Multihop messaging cannot route request
D1H	Maximum connections used - no connections available
D2H	Invalid internet address or host name
D3H	No such host / Cannot communicate with the name server
D4H	Connection not completed before user-specified timeout
D5H	Connection timed out by the network
D7H	Connection refused by destination host
D8H	Connection was broken
D9H	Reply not received before user-specified timeout
DAH	No network buffer space available

Error Code	Description of Error Condition
DBH	Multi-hop messaging CIP message format error
DFH	Multi-hop messaging has no IP address configured for network
E1H	Target Node responded with: Illegal Address Format, a field has an illegal value.
E2H	Target Node responded with: Illegal Address format, not enough fields specified.
E3H	Target Node responded with: Illegal Address format, too many fields specified.
E4H	Target Node responded with: Illegal Address, symbol not found.
E5H	Target Node responded with: Illegal Address Format, symbol is 0 or greater than the maximum number of characters support by this device.
E6H	Target Node responded with: Illegal Address, address does not exist, or does not point to something usable by this command.
E7H	Target node cannot respond because length requested is too large.
E8H	Target Node responded with: Cannot complete request, situation changed (file size, for example) during multi-packet operation.
E9H	Target Node responded with: Data or file is too large. Memory unavailable.
EAH	Target Node responded with: Request is too large; transaction size plus word address is too large.
EBH	Target node cannot respond because target node denies access.
ECH	Target node cannot respond because requested function is currently unavailable.
EDH	Target Node responded with: Resource is already available; condition already exists.
EEH	Target Node responded with: Command cannot be executed.
EFH	Target Node responded with: Overflow; histogram overflow.
F0H	Target Node responded with: No access
F1H	Local processor detects illegal target file type.
F2H	Target Node responded with: Invalid parameter; invalid data in search or command block.
F3H	Target Node responded with: Address reference exists to deleted area.
F4H	Target Node responded with: Command execution failure for unknown reason; PLC-3 histogram overflow.
F5H	Target Node responded with: Data conversion error.
F6H	Target Node responded with: The scanner is not able to communicate with a 1771 rack adapter.
F7H	Target Node responded with: The adapter is not able to communicate with a module.
F8H	Target Node responded with: The 1771 module response was not valid - size, checksum, etc.
F9H	Target Node responded with: Duplicated Label.
FAH	Target node cannot respond because another node is file owner (has sole file access).
FBH	Target node cannot respond because another node is program owner (has sole access to all files).
FCH	Target Node responded with: Disk file is write protected or otherwise inaccessible (off-line only).
FDH	Target Node responded with: Disk file is being used by another application; update not performed (off-line only).
FFH	Local communication channel is shut down.

TIP

For 1770-6.5.16 DFI Protocol and Command Set Reference Manual users:

The MSG error code reflects the STS field of the reply to your MSG instruction.
Codes E0 - EF represent EXT STS codes 0 - F.
Codes F0 - FC represent EXT STS codes 10 - 1C.

SLC Communication Channels

Use the information in this chapter to understand how to configure and monitor the SLC 500 communication channels. The following communication drivers are supported:

Table 13.1 Supported Communication Drivers

Communication Drivers		SLC 500 Fixed	SLC 5/01	SLC 5/02	SLC 5/03	SLC 5/04	SLC 5/05	Page
Channel 1	DH-485	x	x	x	x			13-6
	DH+					x		13-11
	Ethernet						x	13-23
Channel 0	DH-485				x	x	x	13-6
	DF1 Full-Duplex				x	x	x	13-39
	DF1 Half-Duplex Master				x	x	x	13-43
	DF1 Half-Duplex Slave				x	x	x	13-54
	ASCII				x	x	x	13-64

Overview

DH-485 - The SLC 500 Fixed, SLC 5/01, SLC 5/02 and SLC 5/03 have a dedicated channel for DH-485. SLC 5/03, SLC 5/04 and SLC 5/05 RS-232 channel 0 can be reconfigured for DH-485. This network is a multi-master, token-passing network protocol capable of supporting up to 32 devices (nodes). This protocol allows:

- monitoring of data and processor status, along with program uploading and downloading of any device on the network from one location
- SLC processors to pass data to each other (peer-to-peer communication)
- operator interface devices on the network to access data from any SLC processor on the network

Data Highway Plus (DH+) - The Data Highway Plus protocol is used by the SLC 5/04 processor. This protocol is similar to DH-485, except that it can support up to 64 devices (nodes) and runs at faster communication (baud) rates.

Ethernet - The Ethernet TCP/IP protocol is used by the SLC 5/05 processor. Standard Ethernet, utilizing the TCP/IP protocol, is used as the backbone network in many office and industrial buildings. Ethernet is a local area network that provides communication between various devices at 10 Mbps or higher. This network provides the same capabilities as DH+ or DH-485 networks, plus:

- SNMP support for Ethernet network management
- optional dynamic configuration of the processor IP address using a BOOTP utility
- SLC 5/05 Ethernet data rate up to 40 times faster than SLC 5/04 DH+ messaging
- ability to message entire SLC 5/05 data files
- an “unlimited” number of nodes on a single network are possible compared to DH-485 (32) and DH+ (64)
- unlimited number of physical connections.
- 16 maximum message connections opened at one time; 4 incoming, 4 outgoing and 8 either incoming or outgoing

DF1 Full-Duplex - DF1 Full-Duplex protocol (also referred to as DF1 point-to-point protocol) allows two devices to communicate with each other at the same time. This protocol allows:

- transmission of information across modems (dial-up, leased line, radio, or direct cable connections)
- communication to occur between Allen-Bradley products and third-party products

DF1 Half-Duplex (Master and Slave) - DF1 Half-Duplex protocol provides a multi-drop single master/multiple slave network capable of supporting up to 255 devices (nodes). This protocol also provides modem support and is ideal for SCADA (Supervisory Control and Data Acquisition) applications because of the multidrop capability.

ASCII - The ASCII protocol provides connection to other ASCII devices, such as bar code readers, weigh scales, serial printers, and other intelligent devices.

DH-485 Communications

The DH-485 network offers:

- interconnection of 32 devices
- multi-master capability
- token passing access control
- the ability to add or remove nodes without disrupting the network
- maximum network length of 1219 m (4,000 feet)

DH-485 Network Protocol

The following section describes the protocol used to control message transfers on the DH-485 network. The protocol supports two classes of devices: initiators and responders. All initiators on the network get a chance to initiate message transfers. To determine which initiator has the right to transmit, a token passing algorithm is used.

DH-485 Token Rotation

A node holding the token can send valid packets onto the data link. The token hold parameter determines the number of transmissions (plus retries) each time the node receives the token.

After a node sends one message packet, it attempts to give the token to its successor by sending a "token pass" packet. If no network activity occurs, the initiator attempts to find a new successor.

The node address range for an initiator is 0-31. The node address range for all responders is 1 to 31. There must be at least one initiator on the network.

TIP

The maximum address that the initiator searches for before wrapping to zero is the value in the configurable parameter "maximum node address." The default value of this parameter is 31 for all initiators and responders.

SLC 500 processors do not allow node address zero to be applied. If you attempt to apply a zero, node address one becomes the processor node address. Node address zero is reserved for a programming device, such as the Hand-Held Terminal (HHT) or personal computer running programming software.

DH-485 Network Initialization

Network initialization begins when a period of inactivity exceeds the time of a "link dead timeout." When the time for the "link dead timeout" is exceeded, usually the initiator with the lowest address claims the token.

Building a network begins when the initiator that claimed the token tries to pass the token to the successor node. If the attempt to pass the token fails, or if the initiator has no established successor (for example, when it powers up), it begins a linear search for a successor starting with the node above it. It will wrap to node 0 upon reaching its maximum node address value.

When the initiator finds another active initiator, it passes the token to that node, which repeats the process until the token is passed all the way around the network to the first node. At this point, the network is in a state of normal operation.

DH-485 Network Considerations

DH-485 Network considerations include the configuration of the network and the parameters that can be set to the specific requirements of the network. The following are major configuration factors that have a significant effect on network performance:

- number of nodes on the network
- addresses of those nodes
- baud rate
- maximum node address selection
- SLC 5/03 and higher - token hold factor
- maximum number of communicating devices

The following sections explain network considerations and describe ways to select parameters for optimum network performance (speed).

Number of Nodes

The number of nodes on the network directly affects the data transfer time between nodes. Unnecessary nodes (such as a second programming terminal that is not being used) slow the data transfer rate. The maximum number of nodes on the network is 32.

Setting Node Addresses

The best network performance occurs when node addresses start at 0 and are assigned in sequential order. SLC 500 processors default to node address 1. The node address is stored in the processor status file (S:15L). Processors cannot be node 0. Also, initiators such as personal computers should be assigned the lowest numbered addresses to minimize the time required to initialize the network.

If some nodes are connected on a temporary basis, do not assign addresses to them. Simply create nodes as needed and delete them when they are no longer required.

Setting Processor Baud Rate

The best network performance occurs at the highest baud rate. All devices must be at the same baud rate. The default DH-485 baud rate for SLC 500 devices is 19.2K baud. The baud rate is stored in the processor status file (S:15H).

Maximum Node Address Setting

The maximum node address parameter should be set as low as possible. This minimizes the amount of time used in soliciting successors when initializing the network. If all nodes are addressed in sequence from 0, and the maximum node address is equal to the address of the highest addressed node, the token rotation will improve by the amount of time required to transmit a solicit successor packet plus the slot timeout value.

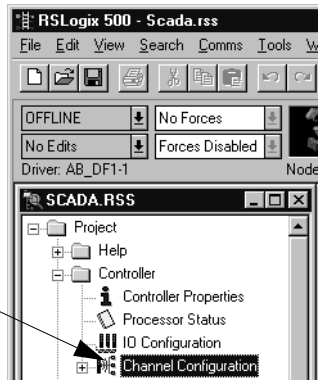
Maximum Number of Communicating Devices

SLC 500 fixed and SLC 5/01 processors can be selected by no more than two initiators at the same time. Using more than two initiators to select the same SLC 500 fixed and SLC 5/01 processors at the same time can cause communication time-outs.

Configuring a Channel for DH-485

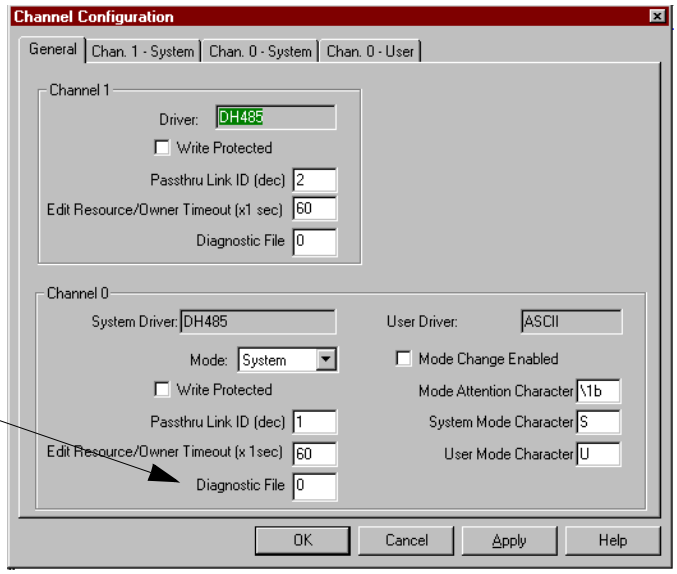
To configure an SLC processor channel for DH-485, do the following using your programming software:

To bring up the Channel Configuration interface, double-click on the Channel Configuration icon.



For an SLC 500 Fixed, SLC 5/01 and SLC 5/02 processor, enter in the Baud Rate and Node Address parameters.

For SLC 5/03, SLC 5/04 and SLC 5/05 processors, define the location of the diagnostic file used for Channel Status here. See Table 13.2 for diagnostic file details.



1. On the Channel 1 or 0 tab, choose DH-485 for your Driver.
2. Configure the communication driver characteristics according to Table 13.2.

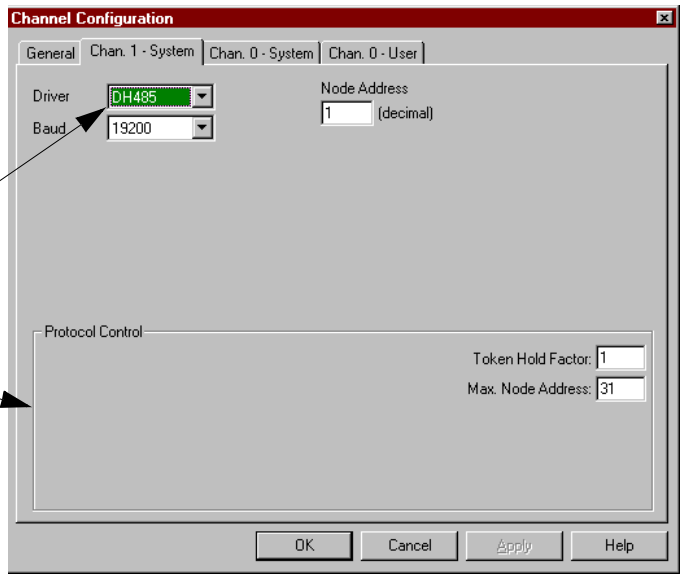


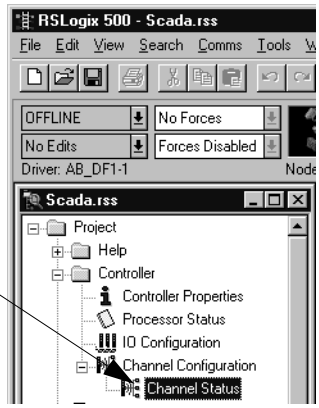
Table 13.2 Define these communication parameters when configuring an SLC 5/03 or higher processor for DH-485 communications.

Tab:	Parameter:	Default	Selections:
General	Diagnostic File	0	Select an unused file (9 to 255) to store channel status information. You must define a diagnostic file in order to be able to view channel 0 or channel 1 status. See Table 13.3 for a file description.
Channel 0 or Channel 1 System	Baud Rate	19.2k	Toggles between the communication rate of 1200, 2400, 9600, and 19200.
	Node Address	1	This is the node address of the processor on the DH-485 network. The valid range is 1 to 31.
	Max Node Address	31	This is the maximum node address of an active processor. The valid range is 1 to 31. The SLC 500 Fixed, SLC 5/01 and SLC 5/02 processors are factory set to 31.
	Token Hold Factor	1	Determines the number of transactions allowed to make each DH-485 token rotation. Increasing this value allows your processor to increase its DH-485 throughput. This also decreases throughput to other processors on the DH-485 link. The valid range is 1 to 4. The SLC 500 Fixed, SLC 5/01 and SLC 5/02 processors are factory set to 1.

DH-485 Channel Status

For SLC 5/03 (OS302, Series C and higher), SLC 5/04 (OS401, Series C and higher) and SLC 5/05, channel status data is stored in the diagnostic file defined on the Channel Configuration screen. Table 13.3 on page 13-8 explains information regarding the diagnostic counter data displayed.

Double-click on the Channel Status Icon Located beneath the Configuration icon to bring up the Channel Status screen.



See Table 13.3 for details concerning the DH-485 Channel Status Screen.

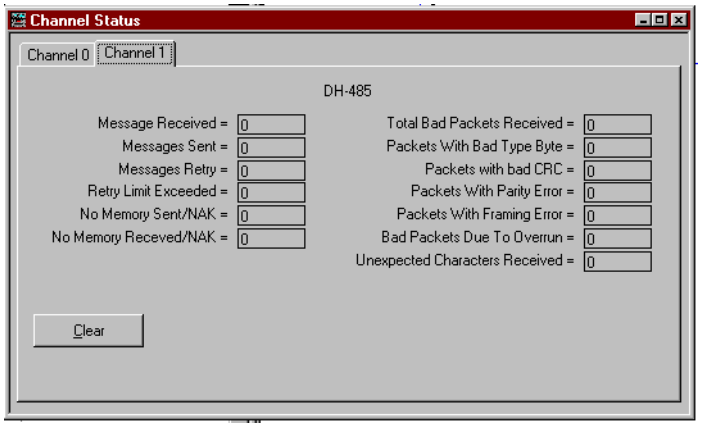


Table 13.3 SLC 5/03 Channel 1 and SLC 5/03 and Higher Channel 0 DH-485 Channel Status

Status field:	Bytes	Displays the:
Messages Received	0,1	Number of error-free messages that the station has received.
Messages Sent	2,3	Number of messages sent by the channel.
Messages Retry	4	Number of messages resent due to errors.
Retry Limit Exceeded	5	Number of times that the processor exceeded its retry limit in trying to send a message.
No Memory Sent/NAK	6	Number of times the processor could not receive a message because it did not have enough memory.
No Memory Received/NAK	7	Number of negative acknowledgements received by the processor.
Total Bad Packets Received	8	Number of incorrect data packets the processor has received.
Packets with bad type byte	9	Number of messages that the processor could not receive because they were of an illegal type that contained a bad control byte.
Packets with bad CRC	10	Number of messages received with a CRC (cyclic redundancy check) transmission integrity error.
Packets with Parity Error	11	Number of messages that could not be processed because of a parity error. Parity is used to detect errors in data bytes. If the parity of a received character is invalid, this counter is incremented.

Table 13.3 SLC 5/03 Channel 1 and SLC 5/03 and Higher Channel 0 DH-485 Channel Status

Status field:	Bytes	Displays the:
Packets with Framing error	12	Number of messages containing misaligned data. Each data byte has a start bit and a stop bit. These bits "frame" the actual data byte. If a character is received where these bits are not correctly placed, the character is invalid and this counter is incremented. These errors normally mean an electrically noisy environment or poorly terminated cabling.
Bad Packets due to Overrun	13	Number of messages that could not be handled because the processor could not move data fast enough before new data arrived. In this case data characters are lost and the transmission is bad.
Unexpected Characters Received	14	Number of characters the processor received with parity or with errors and discarded. If this is not zero, it could indicate that an internal hardware problem has occurred. This problem could be that the communication channel's receiver has been disabled, but the DH-485 driver is still receiving characters. If the controller is operating properly, this counter should remain at zero.

Data Highway Plus Communications

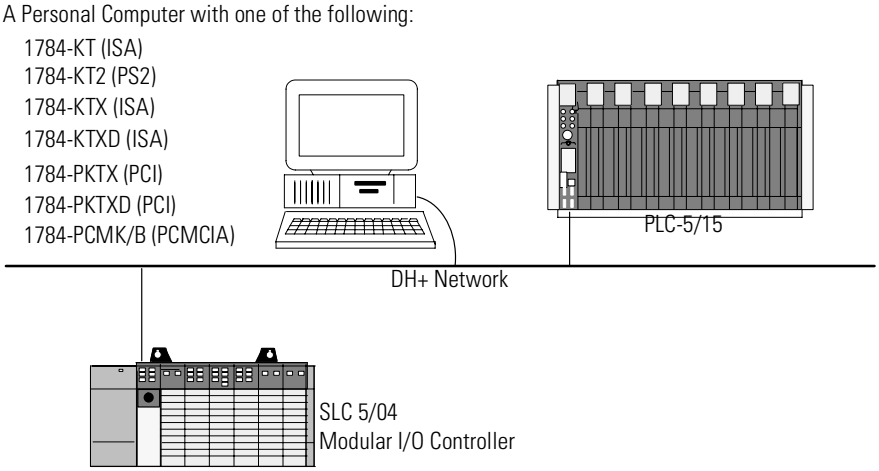
Data Highway Plus implements peer-to-peer communication with a token-passing scheme to rotate link mastership among a maximum of 64 nodes. Since this method does not require polling, it helps provide time-efficient reliable data transport. The DH+features:

- remote programming of PLC-2, PLC-3, PLC-5 and SLC 500 processors on your network
- direct connections to PLC-5 processors and industrial programming terminals
- easy re-configuration and expansion if you want to add more nodes later
- communication rates of 57.6K baud, 115.2K baud, or 230.4K baud

The DH+uses factory set time-outs to restart token-passing communication if the token is lost because of a defective node.

Example

The example below shows the connectivity of an SLC 5/04 processor to a PLC-5 processor using the DH+ protocol. A communication rate of 57.6K baud is used.

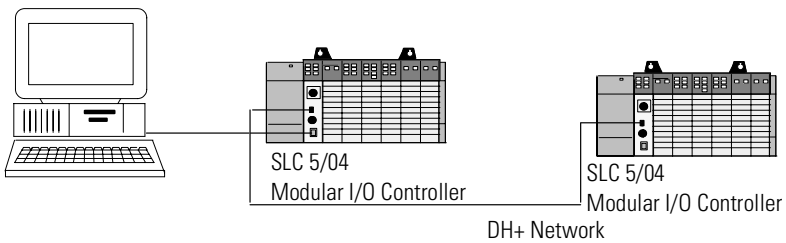


Example

The example below shows a DH+ protocol using two SLC 5/04 controllers using the higher baud rates of 115.2K baud or 230K baud.

TIP

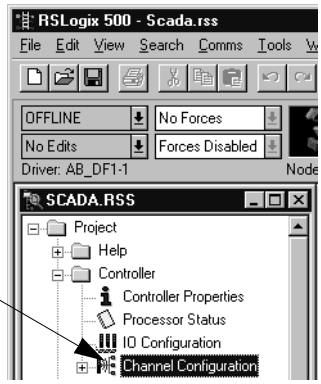
DH+ communication rates 115.2K baud and 230K baud are not available for the programming terminal unless a 1784-KTX(D), 1784-PKTX(D) or 1784-PCMK/B card is used together with RSLinx. In the example below, the programming terminal is connected to the serial port of the SLC 5/04 processor to communicate on the DH+ network at the higher baud rate. This method uses the DF1 to DH+ passthru feature. For more information on passthru see Chapter 14.



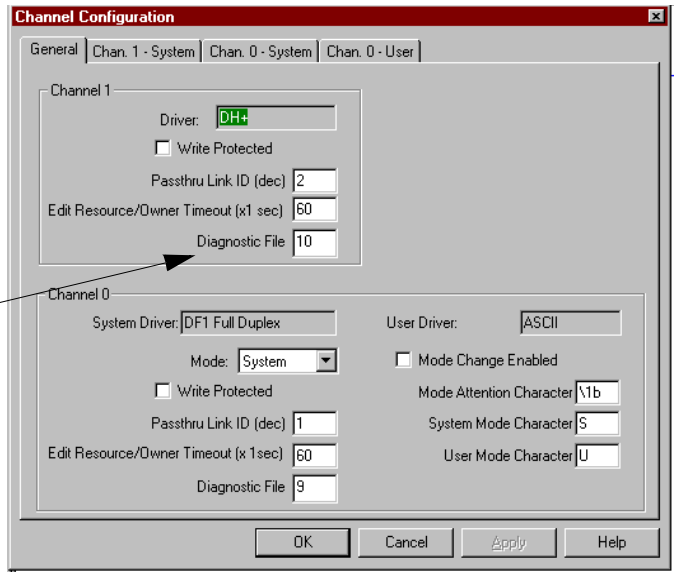
Configuring Channel 1 for DH+

To configure an SLC 5/04 processor channel for DH+, do the following using your programming software:

To bring up the Channel Configuration interface, double-click on the Channel Configuration icon.



Define the location of the diagnostic file used for Channel Status here. See Table 13.5 on page 13-14 for diagnostic file details.



1. On the Channel 1 tab, choose DH+ for your Driver.
2. Configure the communication driver characteristics according to Table 13.4.

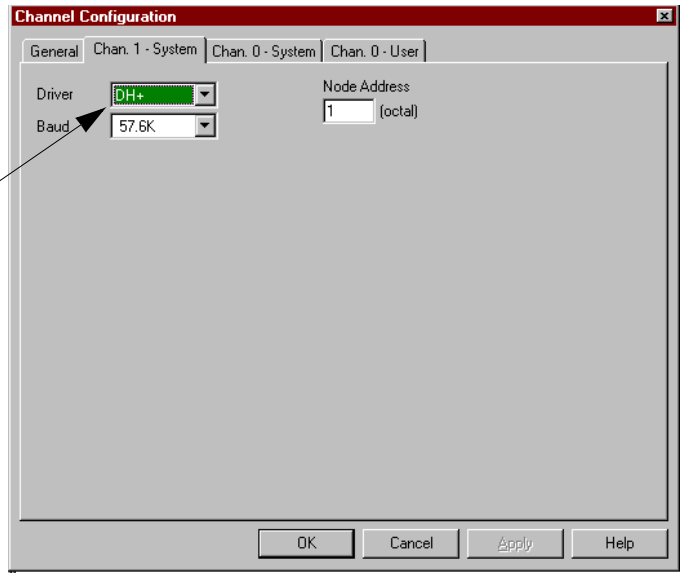


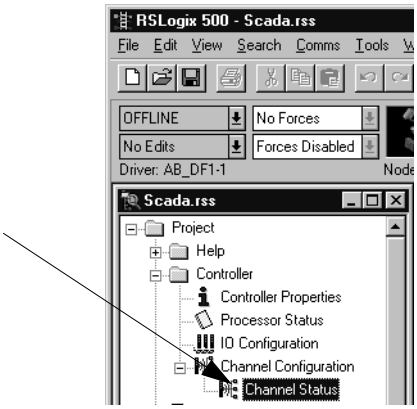
Table 13.4 Define these communication parameters when configuring an SLC 5/04 for DH+ communications.

Tab:	Parameter:	Default	Selections:
General	Diagnostic File	0	Select an unused file (9 to 255) to store channel status information. You must define a diagnostic file in order to be able to view channel 1 status. See Table 13.5 for a file description.
Channel 1 System	Baud Rate	57.6k	Toggles between the communication rates of 57.6k, 115.2k and 230.4k.
	Node Address	1	Valid range is 0 to 77 octal.

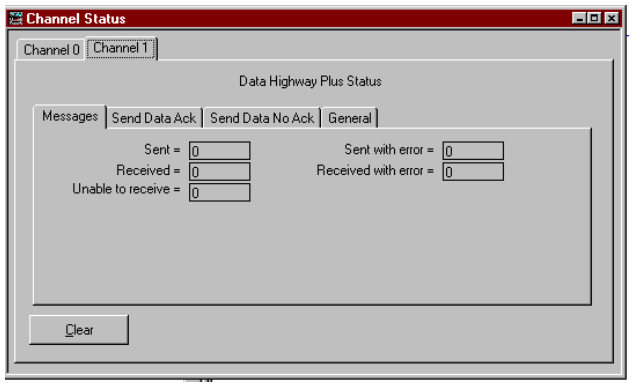
DH+ Channel Status

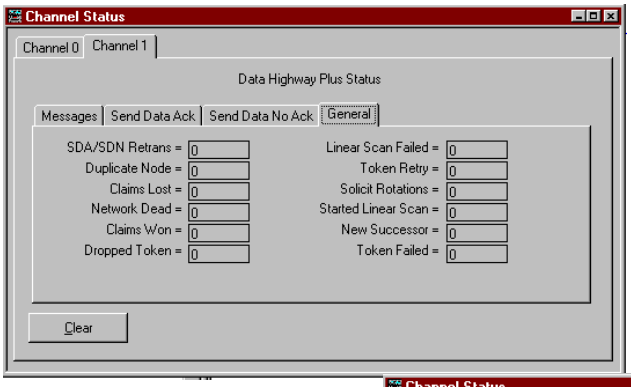
For SLC 5/04 (OS401, Series C and higher), channel status data is stored in the diagnostic file defined on the Channel 1 Configuration screen. See Table 13.5 on page 13-14 for information regarding the diagnostic counter data displayed.

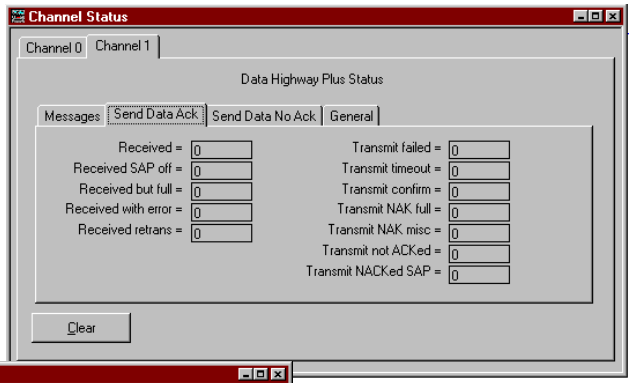
Double-click on the Channel Status Icon Located beneath the Configuration icon to bring up the Channel Status screen.



See Table 13.5 for details concerning the DH+ Channel Status Screens for Messages, General, Data Sent with Acknowledgement, and Data Sent without Acknowledgement.







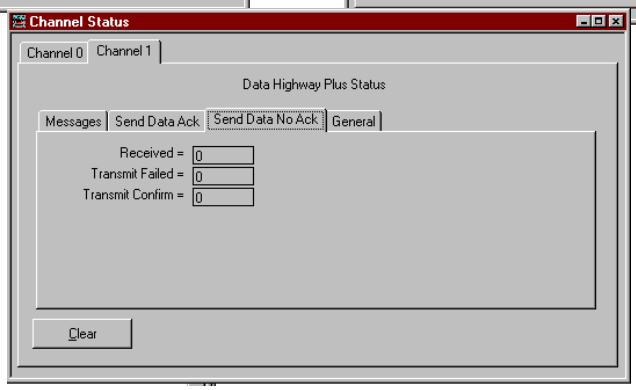


Table 13.5 SLC 5/04 Channel 1 DH+ Channel Status

Status field:		Word	Displays the:
Messages	Received	0	Number of error-free messages the station has received. This number is the sum of the SDA and SDN received counters.
	Sent	1	Total number of messages sent by the station. This number is the sum of the send data acknowledge counters (SDA) and send data no acknowledge (SDN) transmit confirm counters.
	Received with error	2	Number of invalid messages that the station has received. This number is the sum of the SDA received with error and the SDA received SAP off counters.
	Sent with error	3	Number of messages sent that were not acknowledged. This number is the sum of the following: • SDA transmit NAK misc • transmit NAK full • SDA transmit NAKed SAP • SDA/SDN retrans • dropped token
	Unable to receive	4	Total number of times the station NAKed an incoming message due to the lack of an available buffer. This number should be the same as the SDA received but full counter.

Table 13.5 SLC 5/04 Channel 1 DH+ Channel Status

Status field:		Word	Displays the:
General	Network dead	5	Number of times the station detects no traffic on the network. This usually occurs when the station with the token is powered down or is removed from the network. The other stations are waiting for the token to be passed to them. Eventually a network dead situation is declared and a claim token sequence is initiated. (See claims won for more information.)
	Claims won	6	Number of times the station has won the claim token sequence. All the stations initiated a claim token sequence when a network goes down, is just powered up and the stations on the network detect that no one has the token, or when a station with the token is powered down or removed from the network. A claim token sequence is when all the stations on a network attempt to claim the token. When multiple stations attempt to claim the token, the lowest numbered station wins.
	Claims lost	7	Number of times the station did not win the claim token sequence. See claims won for more information.
	New successor	8	Number of times the station found a new successor for the token. A new successor occurs when the station detects that a new station with a station number between its and a the station it was passing the token to was added to the link. The station now must past the token to the newly added station.
	Token retry	9	Number of times the station had to re-transmit a token pass. The station re-transmits a token pass if it detects that the station it passed the token to did not receive the token. Noise can cause this to occur.
	Token failed	10	Number of times station could not pass token to its listed successor. THis usually occurs due to: • the station being removed from the network • noise or cabling problems
	Started linear scan	11	Number of times the station has attempted to pass the token to everyone in its active node table and no one has responded. The station will then start a linear scan where it solicits every station number until a station responds.
	Linear scan failed	12	Number of times the station solicited every station number without getting a response. See started linear scan for more information.
	Duplicate node	13	Number of times the station has detected the same station address as itself on the network. As a result, the station goes offline.
	Dropped token	14	Number of times that the station detected that a duplicate node existed on the link and consequently dropped itself off the link. A station determines that there is a duplicate node when it detects that the response to a message or solicit successor is incorrect. For example, if a response is received from a station which was not communicated with, then the sending station assumes that the response is for a packet sent by another station with the same node number. Once the station drops itself off the link, it waits indefinitely to be solicited back into the network. It will only be solicited back into the network if the duplicate node is removed from the link, because station numbers that already exist on the link are not solicited into the network.
	SDA/SDN retransmissions	24	Total number of SDA or SDN messages that were re-transmitted. Some reasons why the station would retry a message are: • the ACK was lost or corrupted on an SDA message, indicating a possible noise problem • the original message was NACKed.

Table 13.5 SLC 5/04 Channel 1 DH+ Channel Status

Status field:		Word	Displays the:
General	Solicit rotations	30	Number of times a complete solicit successor of all stations not on the link is completed. A solicit successor occurs during a token pass around the link. Here a station that is currently not on the link is solicited to see if it has been added to the link. During each token pass, a different station number is solicited; solicitation occurs sequentially. A station can only join the link when it is solicited into it.
Data Sent with Acknowled-gement (SDA)	Received	15	Number of error-free SDA messages that the station received.
	Received with error	16	Number of invalid SDA messages that the station received. Some causes are: • bad CRC • the message has an invalid source address • the message has an unrecognizable control byte • the transmission was aborted This counter indicates noise; increase the cable's shielding from noise.
	Received retransmissions	17	Number of times the sending station re-transmitted an SDA message, which was ACKed or NAKed. If node sends a message but does not receive an ACK or a NAK response, the node will re-transmit the message. If a node retransmitted a message because the acknowledge response to the first message was lost, the node receiving the message detects the retransmission and sends an acknowledge response. But the receiving node discards the duplicate message. High counts of this counter indicates noise or cable problems; check that the cable is secure and properly shielded from noise.
	Received but full	18	Number of SDA messages that the station could not receive because of lack of memory.
	Received SAP off	19	Number of SDA messages that the station received but could not process because its service access point (SAP) was off. This counter should always be 0.
	Transmit confirm	20	Number of SDA messages successfully sent to and acknowledged by the addressed station.
	Transmit NAK misc.	21	Number of incoming SDA messages that were NAKed due to reasons other than the NAKed full and NAKed inactive counters (e.g., a NAK due to a bad CRC).
	Transmit time-out	22	Number of SDA messages that were sent but not ACKed or NAKed by the receiving station. This counter increments even if the message does get through during a retry and if the receiving station is unable to communicate. This counter indicates a noise or a cabling problem (the receiving station is not seeing the messages).
	Transmit not ACKed	23	Number of SDA messages that were sent but were not ACKed by the receiving station. The following could have occurred: • message could have been NAKed • an invalid ACK was returned • nothing was returned This counter can indicate: • a noise or a cabling problem • the receiving station has been removed from the link • the receiving station cannot communicate

Table 13.5 SLC 5/04 Channel 1 DH+ Channel Status

Status field:		Word	Displays the:
Data Sent with Acknowled-gement (SDA)	Transmit failed	25	Number of SDA messages sent by the station that were determined to be in error. This counter is the sum of the SDA transmit not ACKed and SDA transmit time-out counter.
	Transmit NAK full	26	Number of times the station received NAK to a message because the destination station was full. This indicates that messages are being sent to the receiving station faster than the processor can process them. Most likely, more than one station on the DH+ link is sending messages to the same station. Check to see that you are: • not scheduling unnecessary traffic (e.g., you are sending continuous messages when you only need updates once per second) • implementing report-by-exception so that data is sent only if it is new data.
	Transmit NAKed SAP	27	Number of SDA messages that were successfully sent to but were NAKed by the addressed station because the SAP specified in the message was illegal. This counter should always be 0.
Data Sent without Acknowled-gement (SDN)	Transmit confirm	28	Number of valid SDN messages sent by the station.
	Transmit failed	29	Number of SDN messages sent by the station that were in error. This error should never be seen.
	Received	31	Number of valid SDN messages received.

Global Status Word Overview

When a processor passes the DH+ token to the next node, it can also send a 16-bit word called the Global Status Word (GSW). Every node on the network sees the token pass message, but only the "next" node on the network accepts the token. However, all of the nodes on the network read the Global Status Word sent with each token pass and save it to memory. Each processor on the DH+ network has a table in memory to store Global Status Word(s) it receives from other nodes. In each SLC 5/04 processor's status file, there is a designation for the:

- Global Status Word

  This word is located in memory at S:99. If, S:34/3 is set, data in this memory location is transmitted every time the processor passes the DH+ token. Note that all other DH+ nodes see this data.

- Global Status File

  This file is located in memory at S:100 to S:163, representing one memory location for each of the 64 possible nodes on the DH+ network. As other nodes transmit Global Status information with their token passes, the SLC 5/04 processor collects this information and stores it in the Global Status File. Memory location S:100 corresponds to node #0 (octal), S:101 corresponds to node #1 (octal), and S:163 corresponds to node #77 (octal).

One word of every node's Global Status File is updated each token pass. This can function as a high-speed broadcast message, useful for status passing and synchronization of processors.

If the Global Status Word Transmit Enable bit (S:34/3) and Global Status Word Receive bit (S:34/4) are never set, you can use the Global Status File (S:100 to S:163) for other storage uses. If these bits are reset, this area in the System Status File is never altered by the SLC 5/04 processor, even after a power cycle to the processor.

S:34/3 Global Status Word Transmit Enable Bit

Transmission of the Global Status Word is enabled by setting bit S:34/3 in the status file. If this bit is set (1), the processor transmits the data in S:99 with every DH+ token pass. If this bit is not set (0), the processor passes the token and does not attach the Global Status Word. This bit is dynamically configurable and the default setting is zero. Keep the following guidelines in mind when using the Global Status Word Transmit Enable bit:

- If this bit is not set, the DH+ Token Pass transmitted out Channel 1 will contain no Global Status Word bytes
- If this bit is set, but the SLC 5/04 *is not* in RUN mode, REMote Run, or one of the three test modes, the DH+ Token Pass transmission will contain a 2-byte Global Status Word of 0x0000.
- If this bit is set and the SLC 5/04 *is in* RUN mode, REMote Run, or one of the three tests modes the DH+ Token Pass transmission will contain a 2-byte GSW equal to the value in S:99 (Global Status Word). The word is also placed in the 64-word Global Status File (S:100 to S:163) in the location corresponding to the DH+ node address associated with the SLC 5/04 processor.

  For example, if the SLC 5/04 processor is operating at octal address 22 (18 decimal), the transmitted GSW is written to word S:118.

- The word in the Global Status File corresponding to the SLC 5/04 processor's DH+ address will be set to 0x0000 if any thing is done to inhibit the transmission of the Global Status Word from S:99. This includes:
  - clearing S:34/3, Global Status Word Transmit Enable bit
  - placing the SLC 5/04 into a mode other than Run mode or Test mode
  - disabling Channel 1
  - an error occurring on the DH+ link to cause the Channel 1 LED to flash red or go solid red (This could be caused by a duplicate node address.)

  - having an OS400 user program downloaded to the SLC 5/04 processor
- If S:34/3 is not set from the time the SLC 5/04 is powered up, the word corresponding to its DH+ address in the Global Status File will never be written to during the end-of-scan.

S:34/4 Global Status Word Receive Enable Bit

Receiving the Global Status Words of other processors on the network is enabled by setting bit S:34/4 in the status file. If this bit is set (1), the processor fills in the Global Status File with Global Status Words transmitted by other processors on the network. If this bit is not set (0), the processor ignores any Global Status Word activity on the network. This bit is dynamically configurable and the default setting is zero. Note that transmitting and receiving Global Status Words are independent of each other.

Keep the following guidelines in mind when using the Global Status Word Receive Enable bit:

- If this bit is not set, the Global Status File (S:100 to S:163) is not updated with Global Status Word information being passed on the link.
- An error occurring on the DH+ link to cause the Channel 1 LED to flash red or go solid red disables Global Status Word receptions. (This could be caused by a duplicate node address.)
- Global Status File (S:100-S:163) support is enabled when the following four conditions are met:
  - Channel 1 is configured for DH+ protocol communication
  - the System Status File is at least 164 words in length
  - the Global Status Word Receive Enable bit (S:34/3) is set
  - operation on the DH+ link is working (Channel 1 LED is green)
- The only processor mode that Global Status Word reception will not operate in is while downloading a program.

Note that all 164 words are updated during each end-of-scan. The following table describes possible states of the DH+ node address and the value written to the Global Status Word (S:99).

Table 13.6 DH+ Node Address State

State of the DH+ Node Address	Value written into S:99 by the SLC 5/04 processor
Device is not active on the DH+ link	0x0000
Device is active on the DH+ link, but not sending GSW bytes in its Token Pass	0x0000
Device is active on the DH+ link and is sending 1 byte of GSW data in its Token Pass	High byte is set to 0x00; Low byte is set equal to 1 byte of GSW data
Device is active on the DH+ link and is sending 2 bytes of GSW data in its Token Pass	High byte is set equal to the second byte; Low byte is set equal to the first byte (or High and Low bytes are set equal to each other)
Device is active on the DH+ link and is sending 3 or 4 bytes of GSW data in its Token Pass	High byte is set equal to the second byte; Low byte is set equal to the first byte, and the third and fourth bytes are ignored

- If the Global Status File (S:100-S:163) is working and then Channel 1 is disabled, the entire Global Status File is zeroed out.
- If the Global Status File (S:100-S:163) is working and bit S:34/4 is reset, the entire Global Status File is zeroed out except for the one word corresponding to the Channel 1 DH+ node address.
- If the Global Status File (S:100-S:163) is working and then a DH+ link error occurs, the entire Global Status File is zeroed out. If the SLC 5/04 processor recovers from the error on its own, then the Global Status File updating resumes automatically.
- If the Global Status File (S:100-S:163) is working and then a user program with a System Status File of less than 164 words is downloaded, the SLC 5/04 processor detects this before any further updating of the Global Status File is attempted. In other words, no corruption of the user program results even if all other criteria are still met to support the GSW reception table feature.

TIP

The SLC 5/04 processor maintains a working Global Status Word table regardless if Channel 1 DH+ Active Node Table operation is enabled, (by setting S:34/1).

Ethernet Communications

This section:

- describes SLC 5/05 performance considerations
- describes Ethernet network connections and media
- explains how the SLC 5/05 establishes node connections
- lists Ethernet configuration parameters and procedures
- describes configuration for subnet masks and gateways

The SLC 5/05 supports Ethernet communication via the Ethernet communication channel 1. Ethernet is a local area network that provides communication between various devices at 10 Mbps or higher. The physical communication media options for the SLC 5/05 are:

- built-in
  - 10 Mbps twisted pair (10Base-T)
- with media converters, hubs, or switches
  - fiber optic
  - broadband
  - thick-wire coaxial cable (10Base-5)
  - thin-wire coaxial cable (10Base-2)
  - 100 Mbps twisted pair (100Base-T)
  - 1,000 Mbps twisted pair (1000Base-T)

SLC 5/05 Performance Considerations

Actual performance of an SLC 5/05 processor varies according to:

- size of Ethernet messages
- frequency of Ethernet messages
- network loading
- the implementation of and performance of your processor application program

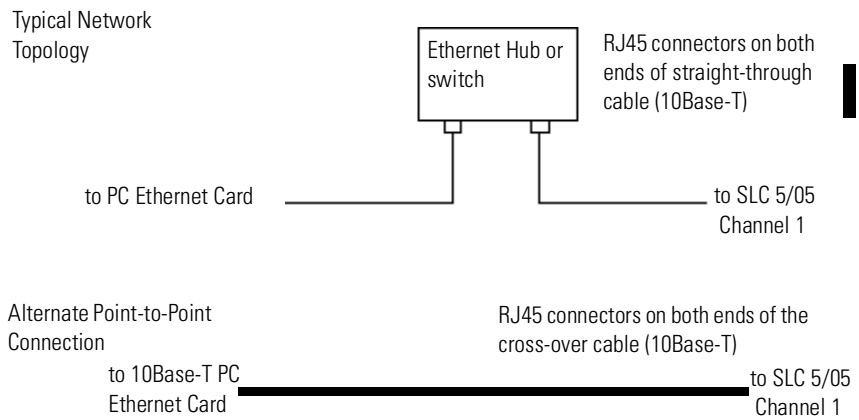
Optimal Performance: PC to SLC 5/05 Processor (2-node Ethernet network)

Operation	Words	MSG per second	ms per MSG	Words per second
Single Typed Read	1	33	30.8	33
Single Typed Reads	20	32	31.1	640
Single Typed Reads	100	32	31.2	3200

SLC 5/05 and PC Connections to the Ethernet Network

TCP/IP is the mechanism used to transport Ethernet messages. The SLC 5/05 processor uses TCP/IP to establish sessions and to send MSG commands. Connections can be initiated by either a client program (INTERCHANGE or RSLinx application) or a processor. Refer to Message Instruction Overview on page 12-3 for information on how connections are established using the MSG instruction

The SLC 5/05 Ethernet connector conforms to ISO/IEC 8802-3 STD 802.3 and utilizes 10Base-T media. Connections are made directly from the SLC 5/05 to an Ethernet hub or switch. The network setup is simple and cost effective. The typical network topology is pictured below, as well as an alternate point-to-point connection using a 10Base-T cross-over cable.



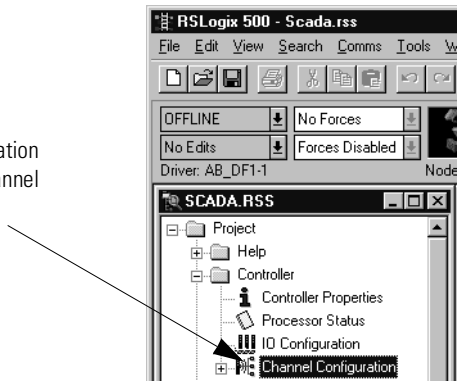
IMPORTANT The SLC 5/05 processor contains a 10Base-T, RJ45 Ethernet connector which connects to standard Ethernet hubs via 8-wire twisted pair straight-through cable. To access other Ethernet mediums, use 10Base-T media converters or Ethernet hubs that can be connected together via fiber, thin-wire, or thick-wire coaxial cables, or any other physical media commercially available with Ethernet hubs or switches.

There are two ways to configure the SLC 5/05 Ethernet channel 1. The configuration can be done via a BOOTP request at processor powerup (see page 13-28), or by manually setting the configuration parameters using RSLogix 500 Programming Software (refer to Configuring Channel 1 for Ethernet).

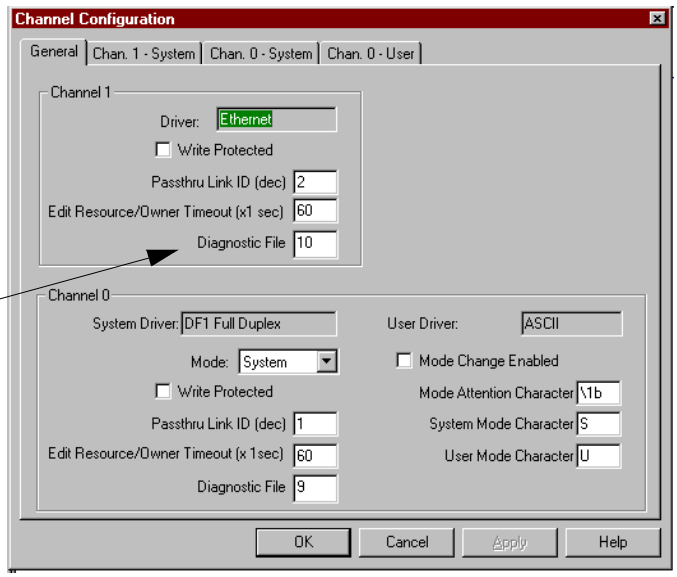
Configuring Channel 1 for Ethernet

To configure and SLC 5/05 processor channel 1 for Ethernet, do the following using your programming software:

To bring up the Channel Configuration interface, double-click on the Channel Configuration icon.



Define the location of the diagnostic file used for Channel Status here. See Table 13.8 on page 13-27 for diagnostic file details.



Configure the communication driver characteristics according to Table 13.7.

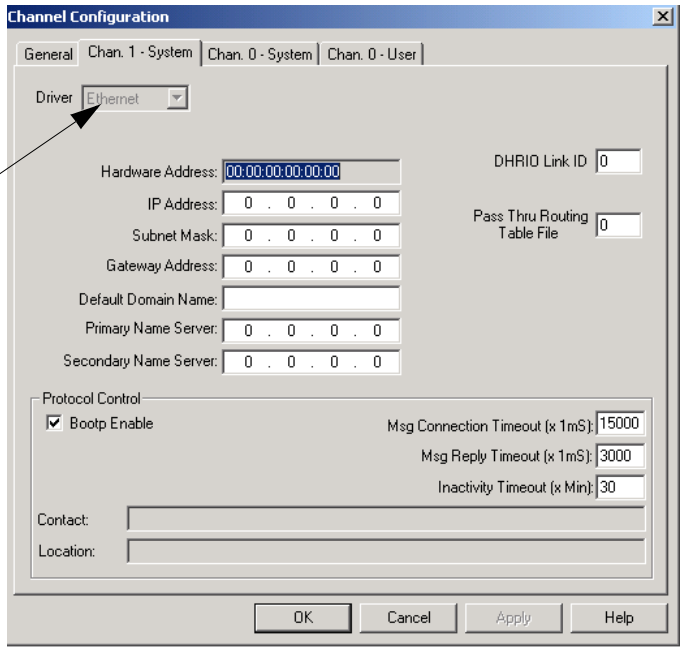


Table 13.7 Define these communication parameters when configuring an SLC 5/05 processor for Ethernet communications.

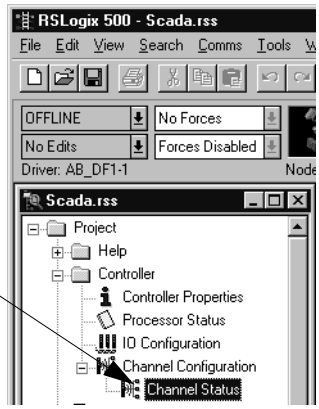
Tab:	Parameter:	Default:	Selections:
General	Diagnostic File	0	Select an unused file to store channel status information. You must define a diagnostic file in order to be able to view channel 1 status. The Diagnostic File Number must be an integer within the limits of 9 to 255. See Table 13.8 for a file description.
Channel 1 System	MSG Connection Timeout	15,000 ms	The amount of time (in ms) allowed for a MSG instruction to establish a connection with the destination node. The MSG Connection Timeout has a range from 250 ms to 65,500 ms.
	MSG Reply Timeout	3,000 ms	The amount of time (in ms) that the SLC 5/05 waits for a reply to a command that it has initiated via a MSG instruction. The MSG Reply Timeout has a range from 250 ms to 65,500 ms.
	Inactivity Timeout	30 minutes	The amount of time (in minutes) that a MSG connection may remain inactive before it is terminated. The Inactivity Timeout has a 1 minute resolution and a range from 1 to 65,500 minutes.
	IP Address	0, 0, 0, 0 (undefined)	The SLC 5/05 internet address (in network byte order). The internet address must be specified to connect to the TCP/IP network.
	Subnet Mask	0, 0, 0, 0	The SLC 5/05 subnet mask (in network byte order). The Subnet Mask is used to interpret IP addresses when the internet is divided into subnets. A Subnet Mask of all zeros indicates that no subnet mask has been configured.
	Gateway Address	0, 0, 0, 0	The address of a gateway (in network byte order) that provides connection to another IP network. A Gateway Address of all zeros indicates that no gateway has been configured.
	Default Domain Name(1)		This is the portion of the Domain Name that is the same for every device on the local Ethernet network. When a target device name is entered in the Ethernet MSG instruction in place of the target device IP address, then the Default Domain Name is appended to the device name when the MSG is triggered and a request is made for the corresponding IP address from the DNS server. The default domain name can have the following formats: “a.b.c”, “a.b” or “a”, where a, b, c must start with a letter, end with a letter or digit, and have as interior characters only letters, digits or hyphens. Maximum length is 47 characters.
	Primary Name Server(1)	0.0.0.0	This is the IP address of the computer acting as the local Ethernet network Primary Domain Name System (DNS) server.
	Secondary Name Server(1)	0.0.0.0	This is the IP address of the computer acting as the local Ethernet network Secondary Domain Name System (DNS) server.
	BOOTP Enable	1 (enabled)	The BOOTP enable switch. When BOOTP is enabled, the SLC 5/05 attempts to learn its network related parameters at powerup via a BOOTP request. There must be a BOOTP server on the network capable of responding to this BOOTP request. When BOOTP is disabled, the SLC 5/05 uses the locally configured network related parameters (IP Address, Subnet Mask, Broadcast Address, etc.).
	Hardware Address	Ethernet hardware address	The SLC 5/05 Ethernet hardware address.
	DHRIO Link ID	0	The link ID assigned to this SLC 5/05 in a 1756-DHRIO module routing table so that DH+ devices can initiate communications to this SLC 5/05.
	Pass Thru Routing Table File	0	Enter an integer file from 9 to 255. This file (routing table) stores up to 128 IP addresses that may be accessed by devices connected to channel 0 using either DF1 Full-Duplex or DH-485 protocols.

(1) Parameter is only functional in OS501, Series C, FRN 5 and higher.

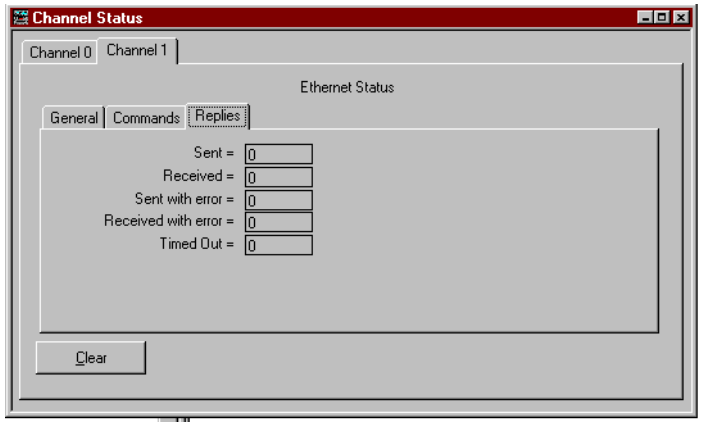
Ethernet Channel Status

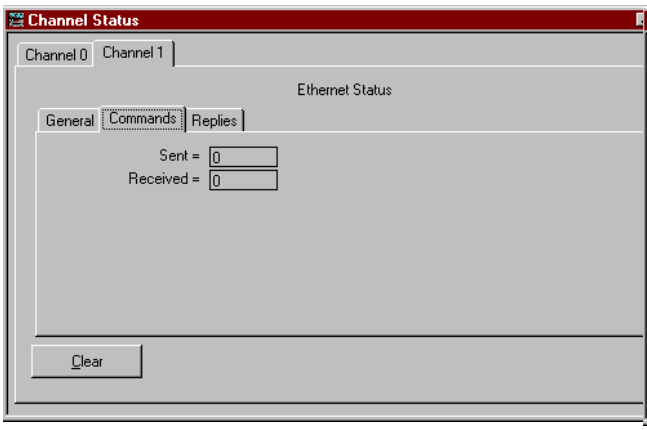
For SLC 5/05 processors, channel status data is stored in the diagnostic file defined on the Channel 0 Configuration screen. See Table 13.8 on page 13-27 for information regarding the diagnostic counter data displayed.

Double-click on the Channel Status Icon located beneath the Configuration icon to bring up the Channel Status screen.



See Table 13.8 for details concerning the Ethernet Channel Status Screen.





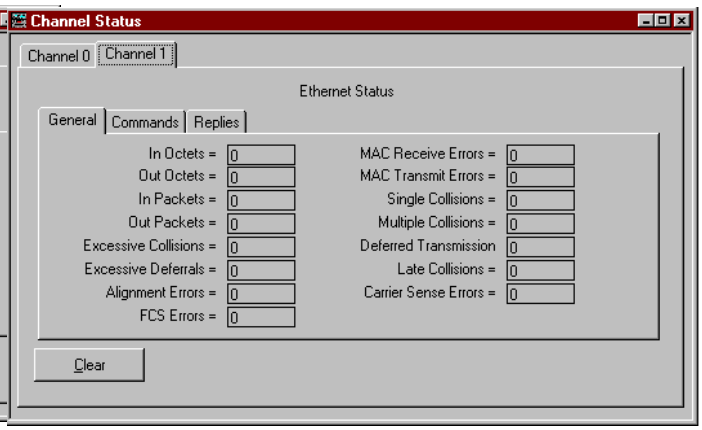


Table 13.8 SLC 5/05 Channel 1 Ethernet Channel Status

Status field:		Words	Displays the number of:
Commands	sent	0,1	Commands sent by the channel.
	received	2,3	Commands received by the channel.
Replies	sent	4,5	Replies sent by the channel.
	sent with error	6.7	Replies containing errors sent by the channel.
	received	8,9	Replies received by the channel.
	received with error	10,11	Replies containing errors received by the channel.
	timed out	12,13	Replies not received within the specified time-out period.
Ethernet	In Octets	14,15	Octets received on the channel.
	Out Octets	16,17	Octets sent on the channel.
	In Packets	18,19	Packets received on the channel, including broadcast packets.
	Out Packets	20,21	Packets sent on the channel, including broadcast packets.
	alignment errors	22,23	Frames received on the channel that are not an integral number of octets in length.
	FCS errors	24,25	Frames received on the channel that do not pass the FCS check.
	carrier sense errors	26,27	Times that the carrier sense condition was lost or never asserted while trying to transmit a frame.
	excessive collisions	28,29	Frames for which a transmission fails due to excessive collisions.
	excessive deferrals	30,31	Frames for which transmission is deferred for an excessive period of time.
	MAC receive errors	32,33	Frames for which reception on an interface fails due to internal MAC sublayer receive error.
	MAC transmit errors	34,35	Frames for which reception on an interface fails due to internal MAC sublayer transmit error.
	single collisions	36,37	Successfully transmitted frames for which transmission was delayed because of collision.
	multiple collisions	38,39	Successfully transmitted frames for which transmission was delayed more than once because of collision.
	deferred transmission	40,41	Frames for which the first transmission attempt is delayed because the medium is busy.
	late collisions	42,43	Times that a collision is detected later than 512 bit–times into the transmission of a packet.
	network storms(1)	44	Times that extremely high levels of Ethernet traffic, or network storms, have been detected.
	Ethernet hardware address(1)	45 through 47	Processor's Ethernet hardware address, which is unique to every processor.
	assigned IP address(1)	48,49	Each of the four bytes holds one of the numbers of the assigned IP address in hex in the dot address format. For example, an IP address of 142.169.121.1 will be displayed in hex as 8EA97901.

Table 13.8 SLC 5/05 Channel 1 Ethernet Channel Status

Status field:		Words	Displays the number of:
Connections	total message connections[1]	50	Total existing Ethernet message connections (16 max).
	incoming message connections[1]	51	Existing incoming Ethernet message connections (12 max).
	outgoing message connections[1]	52	Existing outgoing Ethernet message connections (12 max).
	Maximum Connections Allowed[2]	53	Maximum number of connections allowed.

(1) OS501, Series C, FRN4 and higher.

(2) OS501, Series C, FRN5 and higher.

Configuration Via BOOTP

BOOTP is a standard protocol that TCP/IP nodes use to obtain start-up information. By default, the SLC 5/05 broadcasts BOOTP requests at powerup. The BOOTP Valid parameter remains clear until a BOOTP reply has been received. BOOTP lets you dynamically assign IP Addresses to processors on the Ethernet Link.

To use BOOTP, a BOOTP Server must exist on the local Ethernet subnet. The server is a computer that has BOOTP Server software installed and reads a text file containing network information for individual nodes on the network.

The BOOTP request can be disabled by clearing the BOOTP Enable box on the Channel 1 System Configuration screen. When BOOTP Enable box is cleared (disabled), the SLC 5/05 uses the existing channel configuration data.

IMPORTANT If BOOTP is disabled, or no BOOTP server exists on the network, you must use SLC 500 programming software to enter/change the IP address for each processor.

The host system's BOOTP configuration file must be updated to service requests from SLC 5/05 processors. The following parameters must be configured:

Table 13.9 BOOTP Configuration

Parameter	Description
IP Address	A unique IP Address for the SLC 5/05 processor.
Subnet Mask	Specifies the net and local subnet mask as per the standard on subnetting RFC 950, Internet Standard Subnetting Procedure.
Gateway	Specifies the IP address of a gateway on the same subnet as the SLC 5/05 that provides connections to another IP network.

TIP

If you do not have BOOTP Server capabilities on your network, and you want to dynamically configure Channel 1, you may download a free Allen-Bradley BOOTP server from the Rockwell Automation Knowledge Base website (http://www.ab.com/mem/technotes/kbhome.html) document number 13948.

BOOTP Operation at Power-Up

When BOOTP is enabled, the following events occur at power-up:

- The processor broadcasts a BOOTP-request message containing its hardware address over the local network or subnet.
- The BOOTP server compares the hardware address with the addresses in its look-up table in the BOOTPTAB file.
- The BOOTP server sends a message back to the processor with the IP address and other network information that corresponds to the hardware address it received.

With all hardware and IP addresses in one location, you can easily change IP addresses in the BOOTP configuration file if your network needs change.

Using DOS/Windows BOOTP

The Allen-Bradley BOOTP Server contains DOS-based and Windows-based BOOTP server utilities. Both provide BOOTP services for SLC 5/05 processors. Regardless of the platform you are using, you must:

- install the boot-server utility
- edit the boot-server configuration file
- run the boot-server utility

IMPORTANT Do not use the BOOTP utility disk if you already have INTERCHANGE software installed. Instead, use the boot-server capabilities that came with your INTERCHANGE software.

Install the DOS/Windows BOOTP server

To install the DOS BOOTP server:

1. Put the utility disk that came with your processor in your disk drive.
2. Change directory to the disk drive.
3. Type install, and press [Enter].
4. The software is installed in C:\ABIC\BIN. Put this directory in the path statement of your AUTOEXEC.BAT file.

Edit the DOS/Windows BOOTP Configuration File

The boot-server configuration file, BOOTPTAB, is located in the C:\ABIC\BIN directory. This file contains the information needed to boot SLC 5/05 processors.

You must edit the BOOTPTAB file, which is an ASCII text file, to include the name, IP address, and hardware address for each SLC 5/05 processor you want the server to boot. To edit this file:

1. Open the BOOTPTAB file using a text editor.

   The file contains lines that look like this:

   ```
   #Default string for each type of Ethernet client
   defaults5E: ht=1:vm=rfc1048
   ```

   These are the default parameters for SLC 5/05 processors and must always precede the client lines in the BOOTPTAB file.

   The file also contains a line that looks like this:

```
plc5name: tc=defaults5E:ip=aa.bb.cc.dd:ha=0000BC1Dxxyy
```

IMPORTANT Use this line as the configuration template for SLC 5/05 processors.

2. Make one copy of the SLC 5/05 processor template for every SLC 5/05 processor in your system.

3. Edit each copy of the template as follows:
   1. Replace plc5name with the name of the SLC 5/05 processor. Use only letters and numbers; do not use underscores.
   2. Replace aa.bb.cc.dd with the IP address to be assigned to the processor.
   3. Replace xxyy with the last four digits of the hardware address. Use only valid hexadecimal digits (0-9, A-F); do not use the hyphens that separate the numbers. (You will find the hardware address on a label affixed to the printed circuit board of the SLC 5/05 processor.

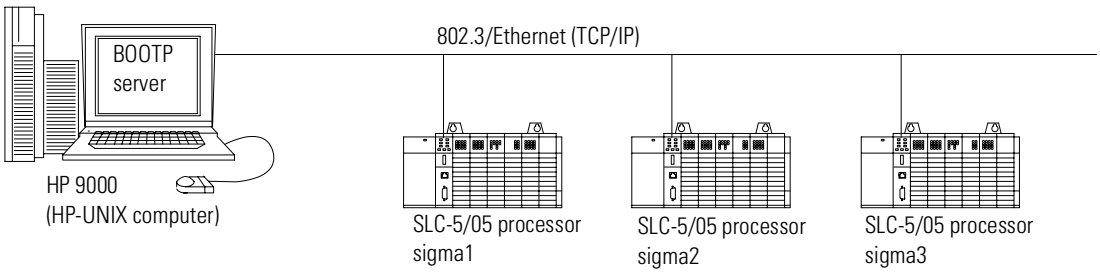
4. Save, close, and make a backup copy of this file.

Example

In this example there are three SLC 5/05 processors and an HP 9000 programming terminal. The names and hardware addresses are device specific:

Table 13.10

Device	Name	IP Address	Hardware Address
SLC 5/05	sigma 1	12.34.56.1	00-00-BC-1D-12-34
SLC 5/05	sigma 2	12.34.56.2	00-00-BC-1D-56-78
SLC 5/05	sigma 3	12.34.56.3	00-00-BC-1D-90-12



Based on this configuration, the BOOTPTAB file looks like:

Table 13.11 BOOTPTAB File

```
#    Legend:    gw    --    gateways
#               ha    --    hardware address
#               ht    --    hardware type(1)
#               ip    --    host IP address
#               sm    --    subnet mask
#               vm    --    BOOTP vendor extensions format(2)
#               tc    --    template host

#Default string for each type of Ethernet client
defaults5E: ht=1:vm=rfc1048

#Entries for SLC 5/05 processors:
sigma1: tc=defaults5E:ip=12.34.56.1:ha=0000BC1D1234
sigma2: tc=defaults5E:ip=12.34.56.2:ha=0000BC1D5678
sigma3: tc=defaults5E:ip=12.34.56.3:ha=0000BC1D9012
```

(1) 1 = 10 MB Ethernet

(2) Use rfc 1048

Run the Boot Server Utility

You can run either the DOS-based utility or the Windows-based BOOTP utility, but not both.

If you have BOOTP enabled and the message BOOTP response not received appears, check the cabling connections and the BOOTP server system.

If you're using this platform	then invoke this executable	from the
DOS-based	DTLBOOTD.EXE	DOS command line (specify optional parameters if necessary)
Windows	DTLBOOTW.EXE	Windows Program Manager

Both utilities are located in the C:\ABIC\BIN directory and use the information contained in the BOOTPTAB file.

Be sure to place the BOOTPTAB file in the directory from which you are running the BOOTP utility. If this file is not found in that directory, the utility will try to find the file in the directory specified by the environment variable ABIC_CONFIG.

Running the DOS-Based Utility

To run the boot-server utility, DTLBOOTD.EXE, follow these steps:

1. At the DOS prompt, type:

   dtlbootd [**-D**] [**-T** <*timeout*>] [**-B** <*numboots*>] [**-F** <*numfiles*>] [*configfile*] [*logfile*]

Parameter	Description
-D	provide additional information for debug purposes.
-T <timeout>	exit after <timeout> seconds of inactivity.
-B <numboots>	exit after answering <numboots> number of boot requests.
-F <numfiles>	exit after answering <numfiles> number of file requests.
configfile	name of the boot server configuration file to use. The default configuration file is %ABIC_CONFIG%\BOOTPTAB.
logfile	name of the log file to use. The default log file is %ABIC_CONFIG%\DTLBOOTD.LOG.

   Once you invoke the utility, it runs until the specified exit parameter is satisfied. Exit any time by pressing [Esc]**.**

2. Apply power to all chassis containing SLC 5/05 processors.

   At power-up, each SLC 5/05 processor broadcasts a BOOTP request if BOOTP was enabled at the channel 1 configuration screen. The Ethernet boot server compares the hardware address with those listed in BOOTPTAB and responds by sending the corresponding IP address and other configuration data to the client via a BOOTP reply.

Running the Windows-Based Utility

To run the boot-server utility, DTLBOOTW.EXE, follow these steps:

1. Start Microsoft Windows®, if it is not already running.
2. Open the Program Manager window, if it is not already open.
3. Choose File on the menu bar and select Run from the menu.
4. In the dialog box, type C:\ABIC\BIN\DTLBOOTW; then, choose OK or press [Enter].

Once you invoke the utility, it will run until you terminate it by closing the DTLBOOTW.EXE window and exiting from Windows.

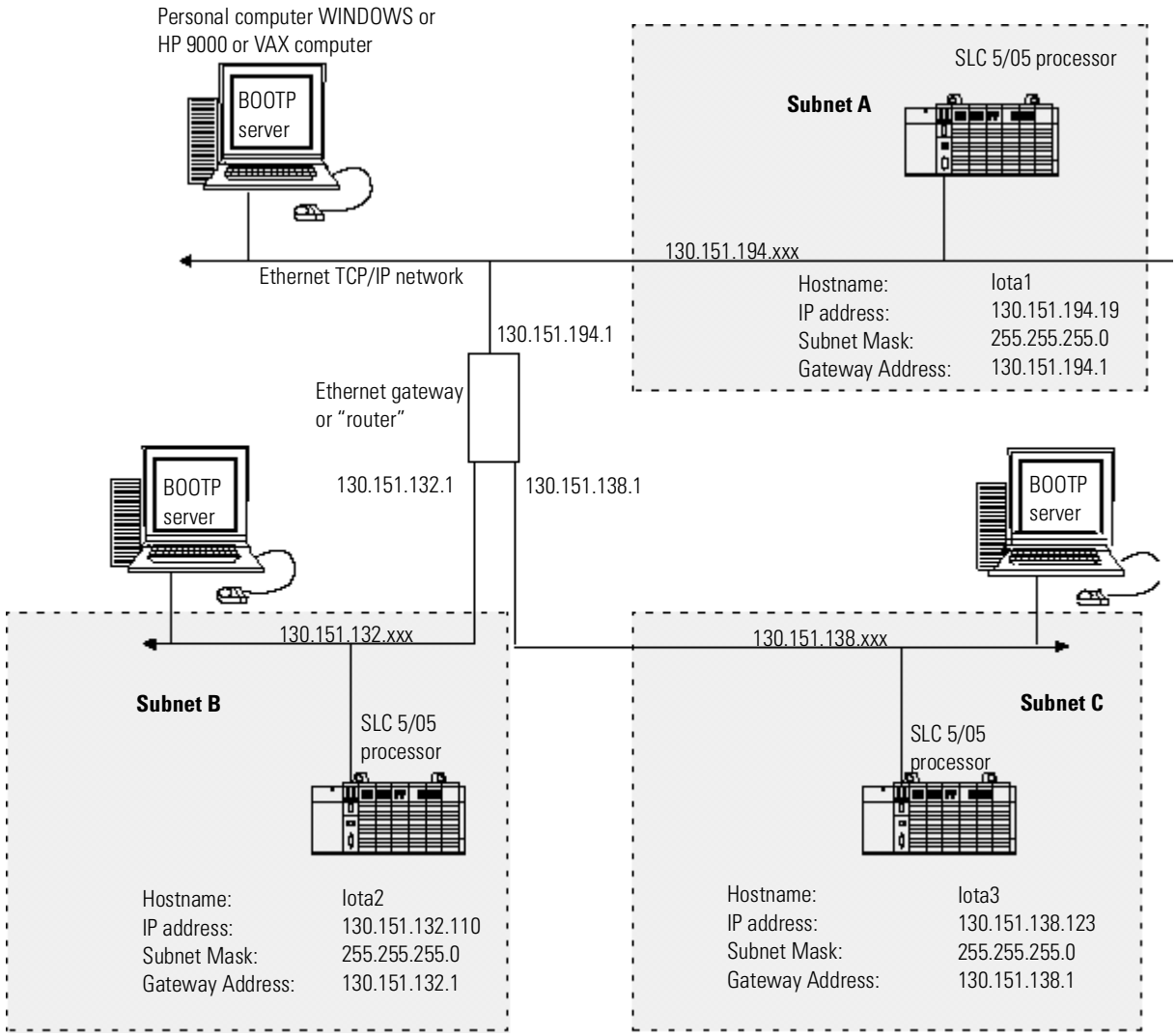
5. Apply power to all chassis containing and SLC 5/05 processors.

At power-up, each SLC 5/05 processor broadcasts a BOOTP request. The Ethernet boot server compares the hardware address with those listed in BOOTPTAB and responds by sending the corresponding IP address and other configuration data to the client via a BOOTP reply.

Using BOOTP to Configure Channel 1 for Processors on Subnets

Configure the BOOTPTAB file according to the subnet mask and gateway address for each SLC 5/05 processor on the link. See the example below and the corresponding BOOTPTAB file on the next page.

IMPORTANT Because BOOTP requests are seen only on the local subnet, each subnet needs its own BOOTP server and BOOTPTAB file.



The BOOTPTAB files that correspond to the example looks like:

Table 13.12 Iota1 Configuration

#	Legend:	gw	--	gateways
#		ha	--	hardware address
#		ht	--	hardware type
#		ip	--	host IP address
#		sm	--	subnet mask
#		vm	--	BOOTP vendor extensions format
#		tc	--	template host

#Default string for each type of Ethernet client
defaults5E: ht=1:vm=rfc1048**:sm=255.255.255.0**

#Entries for SLC 5/05 processors:
iota1:\

tc=defaults5E:\
gw=130.151.194.1:\
ha=0000BC1D1234:/
ip=130.151.194.19

Table 13.13 Iota 2 Configuration

#	Legend:	gw	--	gateways
#		ha	--	hardware address
#		ht	--	hardware type
#		ip	--	host IP address
#		sm	--	subnet mask
#		vm	--	BOOTP vendor extensions format
#		tc	--	template host

#Default string for each type of Ethernet client
defaults5E: ht=1:vm=rfc1048**:sm=255.255.255.0**

#Entries for SLC 5/05 processors:
iota2:\

tc=defaults5E:\
gw=130.151.132.1:\
ha=0000BC1D5678:/
ip=130.151.132.110

Table 13.14 Iota 3 Configuration

#	Legend:	gw	--	gateways
#		ha	--	hardware address
#		ht	--	hardware type
#		ip	--	host IP address
#		sm	--	subnet mask
#		vm	--	BOOTP vendor extensions format
#		tc	--	template host
#Default string for each type of Ethernet client defaults5E: ht=1:vm=rfc1048:**sm=255.255.255.0**				
#Entries for SLC 5/05 processors: iota3:\				
	tc=defaults5E:\ gw=130.151.138.1:\ ha=0000BC1D9012:/ ip=130.151.138.123			

DF1 Communications

The SLC 5/03, SLC 5/04, and SLC 5/05 processors support DF1 Full-Duplex protocol and DF1 Half-Duplex master/slave protocol via channel 0. Refer to *DF1 Protocol and Command Set Reference Manual*, publication 1770-6.5.16, for more information on these communication protocols.

For more information about using the SLC 500 processors in SCADA applications, see the:

- *SCADA System Selection Guide*, publication AG-2.1
- *SCADA System Application Guide*, publication AG-6.5.8

DF1 Full-Duplex Protocol

DF1 Full-Duplex protocol (also referred to as DF1 point-to-point protocol) is provided for applications where RS-232 point-to-point communication is required. This type of protocol supports simultaneous transmissions between two devices in both directions. You can use channel 0 as a programming port, or as a peer-to-peer port using the MSG instruction.

In full-duplex mode, the SLC 5/03 (or higher) processor can send and receive messages. When the processor receives messages, it acts as an end device - a device that stops the transmission of data packets. The processor ignores the destination and source addresses received in the data packets. However, the processor exchanges these addresses in

the reply that it transmits in response to any command data packet that it has received.

If you use a modem with DF1 channel 0 in the full-duplex mode, it must be capable of operating in full-duplex mode. Typically, a dial-up modem is used for communication over telephone lines.

DF1 Half-Duplex Master/Slave Protocol

DF1 Half-Duplex Master/Slave protocol provides a multi-drop single master/multiple slave network. In contrast to DF1 full-duplex, communication takes place in one direction at a time. You can use channel 0 as a programming port, or as a peer-to-peer port using the MSG instruction.

The master device initiates all communication by "polling" each slave device. The slave device may only transmit data packets when it is polled by the master. It is the master's responsibility to poll each slave on a regular and sequential basis to collect data. During a polling sequence, the master polls a slave repeatedly until the slave indicates that it has no more data packets to transmit. The master then transmits the data packets for that slave.

Several Allen-Bradley products support half-duplex master protocol. They include the enhanced PLC-5 processors, SLC 5/03 (OS301 and higher), SLC 5/04, and SLC 5/05 processors. RSLinx (V2.0 and higher) software also supports half-duplex master protocol.

Typically, the master keeps two separate tables - one for online slaves and one for offline slaves. The online slaves are polled on a regular basis. The offline slaves are polled occasionally to see if they have come back online.

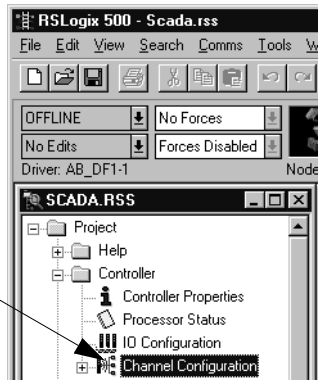
A master device supports routing of data packets from one slave to another.

DF1 half-duplex supports up to 255 slave devices (address 0 to 254) with address 255 reserved for master broadcasts. Either half-duplex or full-duplex modem types can be used for DF1 half-duplex network. The SLC 5/03, SLC 5/04, and SLC 5/05 support broadcast reception, but cannot initiate a broadcast command.

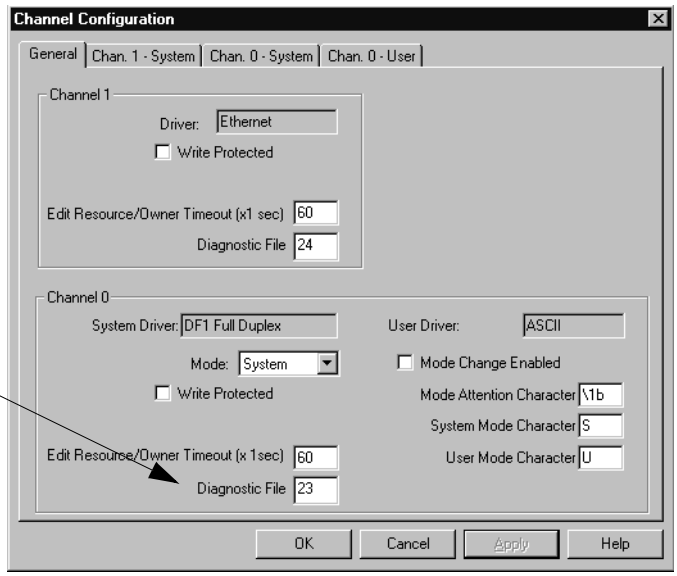
Configuring Channel 0 for DF1 Full-Duplex

To configure an SLC 5/03, SLC 5/04 or SLC 5/05 processor channel 0 for DF1 full-duplex, do the following using your programming software:

To bring up the Channel Configuration interface, double-click on the Channel Configuration icon.



Define the location of the diagnostic file used for Channel Status here. See Table 13.16 on page 13-42 for diagnostic file details.



1. On the Channel 0 tab, choose DF1 Full-Duplex for your Driver.
2. Configure the communication driver characteristics according to Table 13.15.

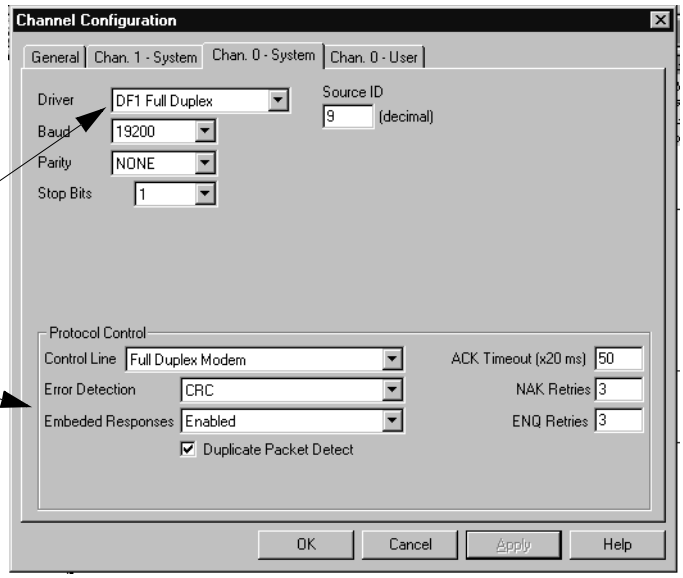


Table 13.15 Define these communication parameters when configuring a SLC 5/03, 5/04, or 5/05 processor for DF1 full-duplex communication.

Tab:	Parameter:	Default:	Selections:
General	Diagnostic File	0	SLC 5/03 (OS 302, Series C or higher), SLC 5/04 (OS 401, Series C or higher) and SLC 5/05 only. Select an unused file (9 to 255) to store channel status information. You must define a diagnostic file in order to be able to view channel 0 status. See Table 13.16 on page 13-42 for a file description.
Chan. 0 System	Baud Rate	19,200	Select a communication rate that all devices in your system support. Configure all devices in the system for the same communication rate.
	Parity	None	Parity provides additional message packet error detection. To implement even parity checking, choose Even. To implement no parity checking, choose None.
	Stop Bits	1	Match the number of stop bits to the devices with which you are communicating.
	Source ID	9	This is the address, in decimal, that is used as the source address in any message initiated by this processor. When DF1 pass-thru is enabled (S:34/5 is set), configure the source ID to equal the channel 1 DH+ address in an SLC 5/04, or zero in an SLC 5/05 processor.
	Control Line	No Handshaking	This parameter defines the mode in which the driver operates. Choose a method appropriate for your system's configuration: • If you are not using a modem, choose NO HANDSHAKING. • If you are using full-duplex modems, choose FULL-DUPLEX MODEM. See page 13-60 for descriptions of the control line operation settings

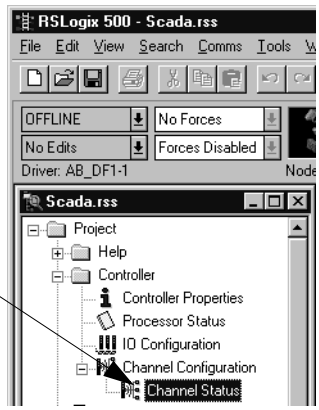
Table 13.15 Define these communication parameters when configuring a SLC 5/03, 5/04, or 5/05 processor for DF1 full-duplex communication.

Tab:	Parameter:	Default:	Selections:
Chan. 0 System	Error Detection	CRC	With this selection, you choose the how the processor checks the accuracy of each DF1 packet transmission. BCC: This algorithm provides a medium level of data security. It cannot detect: • transposition of bytes during transmission of a packet • the insertion or deletion of data values of zero within a packet CRC: This algorithm provides a higher level of data security. Select an error detection method that all devices in your configuration can use. When possible, choose CRC.
	Embedded Responses	Enabled	To use embedded responses, choose Enabled. If you want the processor to use embedded responses only when it detects embedded responses from another device, choose Auto-detect. If you are communicating with another Allen-Bradley device, choose Enabled. Embedded responses increase network traffic efficiency.
	Duplicate Packet Detect	Enabled	Duplicate Detect lets the SLC detect if it has received a message that is a duplicate of its most recent message from the master station. If you choose duplicate detect, the processor will acknowledge (ACK) the message but will not act on it since it has already performed the message's task when it received the command from the first message. If you want to detect duplicate packets and discard them, check this parameter. If you want to accept duplicate packets and execute them, leave this parameter unchecked.
	ACK Timeout	50	The amount of time in 20 millisecond increments that you want the processor to wait for an acknowledgment to the message it has sent before sending an enquiry (ENQ) for the reply.
	NAK Retries	3	The number of times the processor will resend a message packet because the processor received a NAK response to the previous message packet transmission.
	ENQ Retries	3	The number of enquiries (ENQs) that you want the processor to send after an ACK timeout occurs.

DF1 Full-Duplex Channel Status

Channel Status data is stored in the diagnostic file defined on the Channel 0 Configuration screen. See Table 13.16 for information regarding the diagnostic counter data displayed.

Double-click on the Channel Status Icon Located beneath the Configuration icon to bring up the Channel Status screen.



See Table 13.16 for details concerning the DF1 Full-Duplex Channel Status Screen.

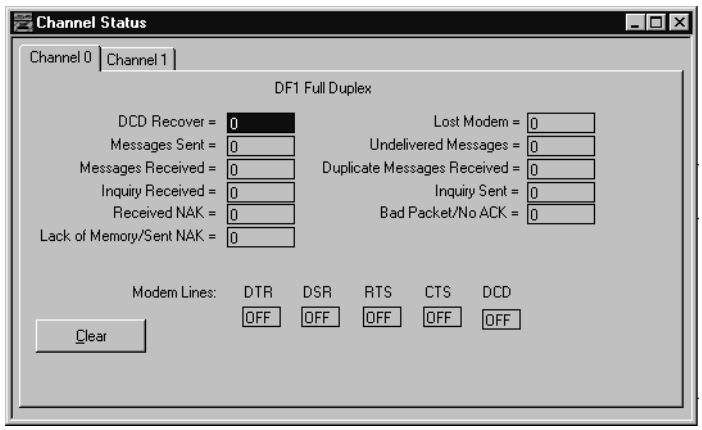


Table 13.16 SLC 5/03 and Higher Channel 0 DF1 Full-Duplex Channel Status

Status Field	Diagnostic File Location	Definition
DCD Recover	word 11	The number of times the processor detects the DCD handshaking line has gone low to high
Messages Sent	word 1	The total number of DF1 messages sent by the processor (including message retries)
Messages Received	word 2	The number of messages received with no errors
Inquiry Received	word 6	The number of ENQs received by the processor
Received NAK	word 5	The number of NAKs received by the processor
Lack of Memory/Sent NAK	word 8	The number of times the processor could not receive a message because it did not have available memory
Lost Modem	word 12	The number of times the lost modem bit has gone low to high

Table 13.16 SLC 5/03 and Higher Channel 0 DF1 Full-Duplex Channel Status

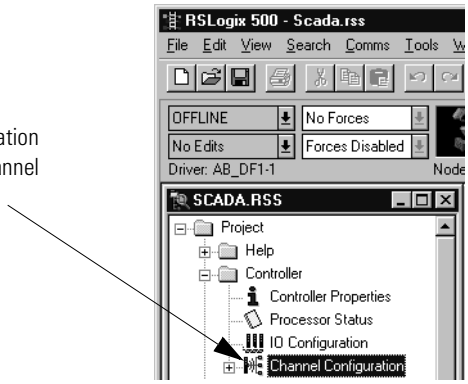
Status Field	Diagnostic File Location	Definition
Undelivered Messages	word 3	The number of messages that were sent by the processor but not acknowledged by the destination device
Duplicate Messages Received	word 9	The number of times the processor received a message packet identical to the previous message packet
Inquiry Sent	word 4	The number of ENQs sent by the processor
Bad Packet/No ACK	word 7	The number of incorrect data packets received by the processor for which a NAK was returned
DTR (Data Terminal Ready)	word 0;bit 4	The status of the DTR handshaking line (asserted by the processor)
DSR (Data Set Ready)	word 0;bit 2	The status of the DSR handshaking line (received by the processor)
RTS (Request to Send)	word 0;bit 1	The status of the RTS handshaking line (asserted by the processor)
CTS (Clear to Send)	word 0;bit 0	The status of the CTS handshaking line (received by the processor)
DCD (Data Carrier Detect)	word 0;bit 3	The status of the DCD handshaking line (received by the processor)

Configuring Channel 0 for Standard-Mode DF1 Half-Duplex Master

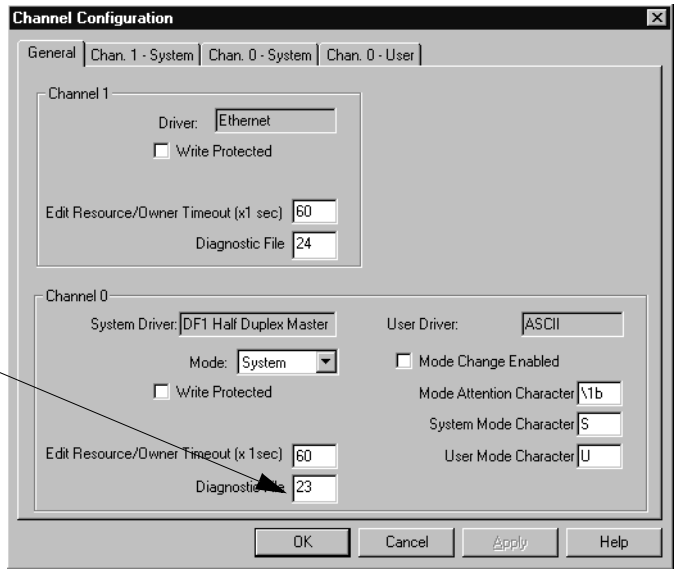
Choose DF1 half-duplex master in standard mode if you want to query slave stations for information based upon user-configured polling ranges. This mode is used most often in point-to-multipoint configurations.

To configure the processor for a master station using standard communication, place the processor into program mode and do the following using your programming software:

To bring up the Channel Configuration interface, double-click on the Channel Configuration icon.



Define the location of the diagnostic file used for Channel Status here. For Channel Status details, see page 13-49.



1. On the Channel 0 tab, choose DF1 Half-Duplex for your Driver.
2. Choose a Standard Polling Mode.
3. Configure the rest of the communication driver characteristics according to Table 13.17.

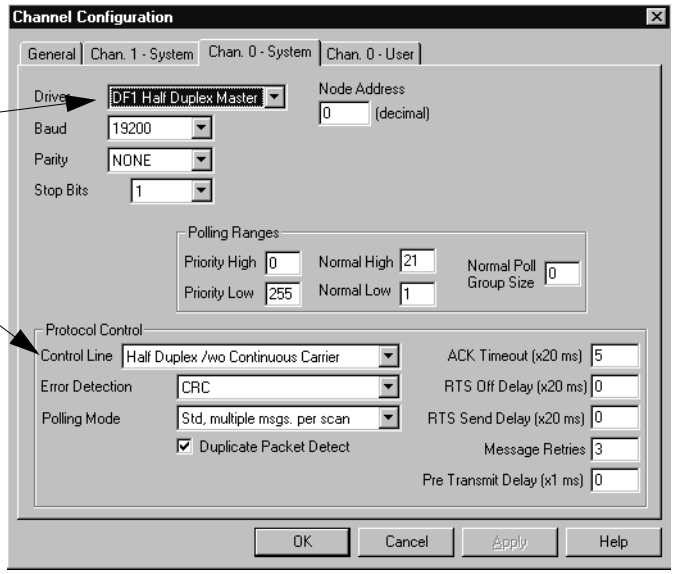


Table 13.17 Define these parameters when configuring a SLC 5/03, 5/04, or 5/05 processor as a master station using standard-communication mode to talk to slave stations.

Tab:	Parameter:	Default:	Selections:
General	Diagnostic File	0	SLC 5/03 (OS 302, Series C or higher), SLC 5/04 (OS 401, Series C or higher) and SLC 5/05 only. Select an unused file (9 to 255) to store channel status information. You must define a diagnostic file in order to be able to view channel 0 status. See Table 13.19 on page 13-49 for a file description.
Channel 0 System	Baud Rate	1200	Select a communication rate that all devices in your system support. Configure all devices in the system for the same communication rate.
	Parity	None	Parity provides additional message packet error detection. To implement even parity checking, choose Even. To implement no parity checking, choose None.
	Stop Bits	1	Match the number of stop bits to the device with which you are communicating.
	Node Address	0	A node address identifies the processor on the DF1 half-duplex link. Each station on a link must have a unique address. Choose an address between 0_{10} and 254_{10}. Node address255_{10} is the broadcast address, and cannot be selected as a station's individual address
	Control Line	No Handshaking	This parameter defines the mode in which the driver operates. Choose a method appropriate for your system's configuration: • If you are not using a modem, choose NO HANDSHAKING. • If the master modem is full duplex, choose FULL-DUPLEX MODEM. • If all the modems in the system are half-duplex, choose HALF-DUPLEX WITHOUT CONTINUOUS CARRIER.
	Error Detection	CRC	With this selection, you choose the how the processor checks the accuracy of each DF1 packet transmission. BCC: This algorithm provides a medium level of data security. It cannot detect: • transposition of bytes during transmission of a packet • the insertion or deletion of data values of zero within a packet CRC: This algorithm provides a higher level of data security. Select an error detection method that all devices in your configuration can use. When possible, choose CRC.
	Polling Mode	Message Based	If you want to receive: • only one message from a slave station per its turn, choose STANDARD (SINGLE MESSAGE TRANSFER PER NODE SCAN). Choose this method only if it is critical to keep the poll list scan time to a minimum. • as many messages from a slave station as it has, choose STANDARD (MULTIPLE MESSAGE TRANSFER PER NODE SCAN).

Table 13.17 Define these parameters when configuring a SLC 5/03, 5/04, or 5/05 processor as a master station using standard-communication mode to talk to slave stations.

Tab:	Parameter:	Default:	Selections:
Channel 0 System	Duplicate Packet Detect	Enabled	Duplicate Detect lets the SLC detect if it has received a message that is a duplicate of its most recent message from another station. If you choose duplicate detect, the processor will acknowledge (ACK) the message but will not act on it since it has already performed the message's task when it received the command from the first message. If you want to detect duplicate packets and discard them, check this parameter. If you want to accept duplicate packets and execute them, leave this parameter unchecked.
	ACK Timeout	50	The amount of time in 20 millisecond increments that you want the processor to wait for an acknowledgment to the message it has sent before the processor retries the message or the message errors out. This timeout value is also used for the poll response timeout. See page 13-47 for recommendations to minimize this value.
	RTS Off Delay	0	Defines the amount of time in 20 millisecond increments that elapses between the end of the message transmission and the de-assertion of the RTS signal. This time delay is a buffer to make sure that the modem has transmitted the message but should normally be left at zero.
	RTS Send Delay	0	Defines the amount of time in 20 millisecond increments that elapses between the assertion of the RTS signal and the beginning of the message transmission. This time allows the modem to prepare to transmit the message. The Clear-to-Send (CTS) signal must be high for transmission to occur.
	Pre-Transmit Delay	0	Defines the amount of time in 1 millisecond increments that elapses between when the processor has a message to send and when it asserts the RTS signal.
	Message Retries	3	Defines the number of times a master station retries either: • a message before it declares the message undeliverable • or a poll packet to an active station before the master station declares that station to be inactive.
	Priority Polling Range – High	0	Select the last slave station address to priority poll.
	Priority Polling Range – Low	255	Select the first slave station address to priority poll. Entering 255 disables priority polling.
	Normal Polling Range – High	0	Select the last slave station address to normal poll.
	Normal Polling Range – Low	255	Select the first slave station address to normal poll. Entering 255 disables normal polling.
	Normal Poll Group Size	0	Enter the quantity of active stations located in the normal poll range that you want polled during a scan through the normal poll range before returning to the priority poll range. If no stations are configured in the Priority Polling Range, leave this parameter at 0.

Minimum DF1 Half-Duplex Master Channel 0 ACK Timeout

The governing timeout parameter to configure for a DF1 Half-Duplex Master is the channel 0 ACK Timeout. The ACK Timeout is the amount of time you want the processor to wait for an acknowledgment of its message transmissions. Set in 20 millisecond intervals, the value is the amount of time the master will wait for:

- an ACK to be returned by a slave when the master has just sent it a message, or
- a poll response or message to be returned by a slave when the master has just sent it a poll packet.

The timeout must be long enough that after the master has transmitted the last character of the poll packet, there is enough time for a slave to transmit (and the master receive) a maximum sized packet before the time expires.

To calculate the minimum ACK timeout, you must know:

- the modem baud rate
- maximum sized data packet (the maximum number of data words that a slave write command or read reply packet might contain)
- the RTS/CTS or “turnaround” delay of the slave modem
- the configured RTS Send Delay in the slave
- the program scan time of the slave

Determining Minimum Master ACK Timeout

To determine the minimum ACK Timeout, you must first calculate the transmission time by multiplying the maximum sized data packet for your processor by the modem rate in ms/byte. For an example we will assume an SLC 5/03 processor (103 data words or 224 bytes total packet size including overhead) and a 9600 bps modem, which transmits at approximately 1 ms/byte. Therefore, the message transmission time is 224 ms. For approximate modem transmission rates, see the following table.

Table 13.18 Approximate modem transmission rates

modem bps	approx. ms/byte
4800	2 ms/byte
9600	1 ms/byte
19200	.5 ms/byte

Next, you need to determine the average slave program scan time. In RSLogix 500, double click on the Processor Status icon and then locate Average on the Scan Times tab. For this example, lets assume an average slave program scan time of 20 ms. Remember, program scan time will vary by application.

Finally, you must determine the larger of two values, either the configured slave RTS Send Delay or the turnaround time of the slave modem. The RTS Send Delay time can be found by double-clicking on the slave's Channel Configuration icon and looking at the Chan. 0 System tab of the Channel Configuration screen. Note that the RTS Send Delay time is in intervals of 20 ms, so with a value of 3 in the box, the RTS Send Delay time would be 20 ms multiplied by 3. Using this value (60 ms) for our example, and assuming that the turnaround time of the modem is 50 ms (which will vary by modem) you would choose to use the RTS Send Delay time of 60 ms for your calculation.

Having determined the maximum message transmission time (224 ms), the average slave program scan time (20 ms) and the largest of either RTS Send Delay (60 ms) or the modem turnaround time, the minimum ACK timeout is simply the sum of these values.

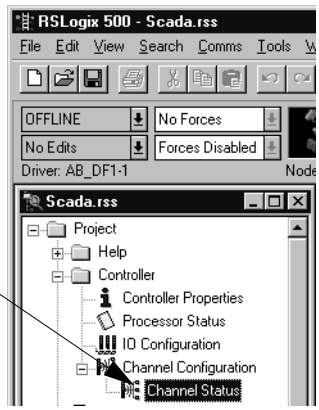
Use only the largest of these two values

Parameter	Example Values (in ms)
Max message transmission time	224
Average program scan time	20
RTS Send Delay	60
modem turnaround time	50
calculated ACK Timeout	304
round up to nearest 20 ms	320

DF1 Half-Duplex Master Channel Status

Channel Status data is stored in the diagnostic file defined on the Channel 0 Configuration screen. See Table 13.19 on page 13-49 for information regarding the diagnostic counter data displayed.

Double-click on the Channel Status Icon Located beneath the Configuration icon to bring up the Channel Status screen.



See Table 13.19 for details concerning the DF1 Half-Duplex Master Channel Status Screen.

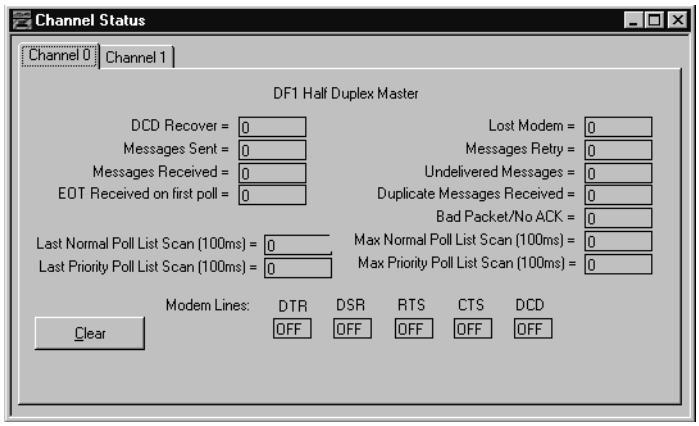


Table 13.19 SLC 5/03 and Higher Channel 0 DF1 Half-Duplex Master Channel Status

Status Field	Diagnostic File Location	Definition
DCD Recover	word 11	The number of times the processor detects the DCD handshaking line has gone low to high
Messages Sent	word 1	The total number of DF1 messages sent by the processor (including message retries)
Messages Received	word 2	The number of messages received with no errors
EOT Received on First Poll	word 8	Not implemented
Last Normal Poll List Scan	word 5	Time in 100 ms increments of last scan through Normal Poll List
Last Priority Poll List Scan	word 10	Time in 100 ms increments of last scan through Priority Poll List
Lost Modem	word 12	The number of times the lost modem bit has gone low to high

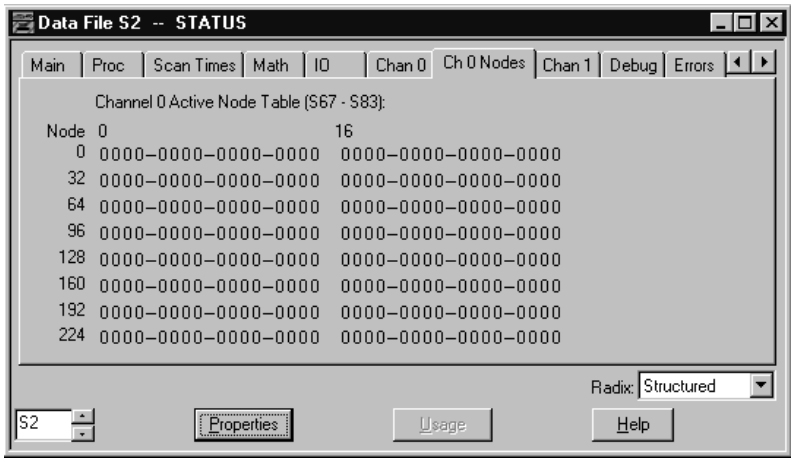
Table 13.19 SLC 5/03 and Higher Channel 0 DF1 Half-Duplex Master Channel Status

Status Field	Diagnostic File Location	Definition
Message Retry	word 4	The number of message retries sent by the processor
Undelivered Messages	word 3	The number of messages that were sent by the processor but not acknowledged by the destination device
Duplicate Messages Received	word 9	The number of times the processor received a message packet identical to the previous message packet
Bad Packet/No ACK	word 7	The number of incorrect data packets received by the processor for which no ACK was returned
Max Normal Poll List Scan	word 6	Maximum time in 100 ms increments to scan the Normal Poll List
Max Priority Poll List Scan	word 13	Maximum time in 100 ms increments to scan the Priority Poll List
DTR (Data Terminal Ready)	word 0;bit 4	The status of the DTR handshaking line (asserted by the processor)
DSR (Data Set Ready)	word 0;bit 2	The status of the DSR handshaking line (received by the processor)
RTS (Request to Send)	word 0;bit 1	The status of the RTS handshaking line (asserted by the processor)
CTS (Clear to Send)	word 0;bit 0	The status of the CTS handshaking line (received by the processor)
DCD (Data Carrier Detect)	word 0;bit 3	The status of the DCD handshaking line (received by the processor)

Monitor Active Stations

To see what stations are active, view the channel 0 active node table in the SLC 5/03, SLC 5/04, or SLC 5/05 processor status file (S:67/0-S:82/15). Each bit in the file represents a station on the link. The stations are numbered in order as a continuous bitstream file starting with the first bit in word S:67 (See Figure 13.1 below).

Figure 13.1 Example Active Node Table



At powerup or after reconfiguration, the master station assumes that all slave stations are inactive. A station is shown active only after it responds to a poll packet.

Configuring Channel 0 for Message-based Mode DF1 Half-Duplex Master

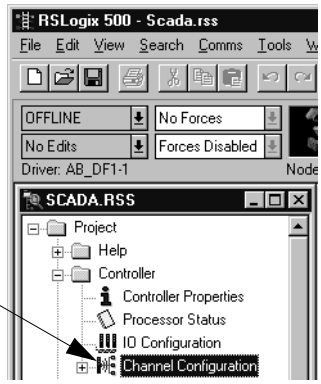
Choose DF1 half-duplex master in message-based communication mode if you want to use MSG instructions in user programming to communicate with one station at a time. If your application uses satellite transmission or public switched telephone network transmission, consider choosing message-based. Communication to a slave station can be initiated on an as-needed basis.

Message-based communication should also be used in redundant SLC master station systems implemented with the 1746-BSN backup communication module.

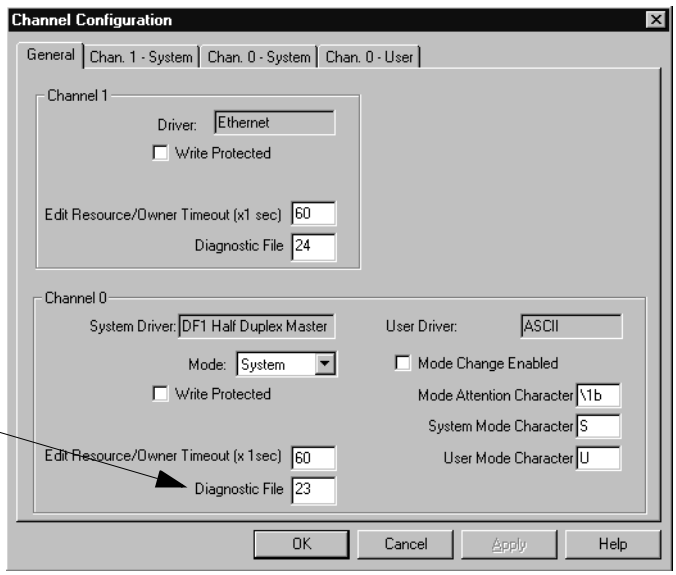
With message-based mode, you do not have an active node file that you can use to monitor station status. Also, you cannot implement slave station-to-slave station messaging or slave programming.

To configure the processor for a master station using message-based communication, place the processor in program mode and do the following using your programming software:

To bring up the Channel Configuration interface, double-click on the Channel Configuration icon.



Define the location of the diagnostic file used for Channel Status here. For Channel Status details, see page 13-49.



1. On the Channel 0 tab, choose DF1 Half-Duplex Master for your Driver.
2. Choose a Message-based Polling Mode.
3. Configure the communication driver characteristics according to Table 13.20.

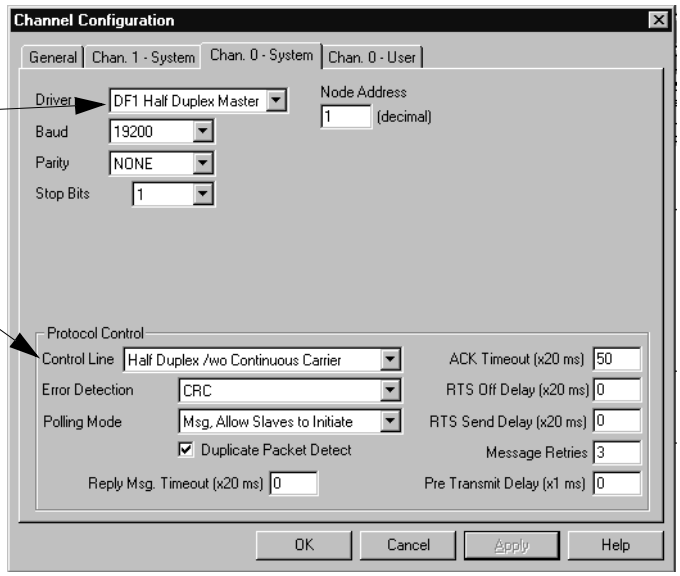


Table 13.20 Define these parameters when configuring a SLC 5/03, 5/04, or 5/05 processor as a master station using message-based communication mode to talk to slave stations.

Tab:	Parameter:	Default:	Selections:
General	Diagnostic File	0	SLC 5/03 (OS 302, Series C or higher), SLC 5/04 (OS 401, Series C or higher) and SLC 5/05 only. Select an unused file (9-255) to store channel status information. You must define a diagnostic file in order to be able to view channel 0 status. See Table 13.19 on page 13-49 for a file description.
Channel 0 System	Baud Rate	1200	Select a communication rate that all devices in your system support. Configure all devices in the system for the same communication rate.
	Parity	None	Parity provides additional message packet error detection. To implement even parity checking, choose Even. To implement no parity checking, choose None.
	Stop Bits	1	Match the number of stop bits to the devices with which you are communicating.
	Node Address	0	A node address identifies the processor on the DF1 half-duplex link. Each station on a link must have a unique address. Choose an address between 0_{10} and 254_{10}. Node address255_{10} is the broadcast address, and cannot be selected as a station's individual address
	Control Line	No Handshaking	This parameter defines the mode in which the driver operates. Choose a method appropriate for your system's configuration: • If you are not using a modem, choose NO HANDSHAKING. • If the master modem is full duplex, choose FULL-DUPLEX. • If all the modems in the system are half-duplex, choose HALF-DUPLEX WITHOUT CONTINUOUS CARRIER. See page 13-60 for descriptions of control line operation settings.
	Error Detection	CRC	With this selection, you choose the how the processor checks the accuracy of each DF1 packet transmission. BCC: This algorithm provides a medium level of data security. It cannot detect: • transposition of bytes during transmission of a packet • the insertion or deletion of data values of zero within a packet CRC: This algorithm provides a higher level of data security. Select an error detection method that all devices in your configuration can use. When possible, choose CRC.

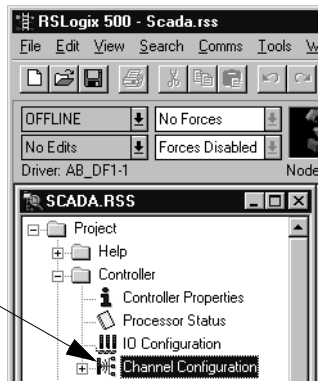
Table 13.20 Define these parameters when configuring a SLC 5/03, 5/04, or 5/05 processor as a master station using message-based communication mode to talk to slave stations.

Tab:	Parameter:	Default:	Selections:
Channel 0 System	Polling Mode	Message Based	If you want to: • accept unsolicited messages from slave stations, choose MESSAGE BASED (ALLOW SLAVES TO INITIATE MESSAGES) Slave station-initiated messages are acknowledged and processed after all master station-initiated (solicited) messages. **Note:** Slave stations can only send messages when they are polled. If the message-based master station never sends a slave station a message, the master station will never send the slave station a poll. Therefore, to regularly obtain a slave station-initiated message from a slave station, you should choose to use standard communication mode instead. • ignore unsolicited messages from slave stations, choose MESSAGE BASED (DO NOT ALLOW SLAVES TO INITIATE MESSAGES) Slave station-initiated messages are acknowledged and discarded. The master station acknowledges the slave station-initiated message so that the slave station removes the message from its transmit queue, which allows the next packet slated for transmission into the transmit queue.
	Duplicate Packet Detect	Enabled	Duplicate Detect lets the SLC detect if it has received a message that is a duplicate of its most recent message from another station. If you choose duplicate detect, the processor will acknowledge (ACK) the message but will not act on it since it has already performed the message's task when it received the command from the first message. If you want to detect duplicate packets and discard them, check this parameter. If you want to accept duplicate packets and execute them, leave this parameter unchecked.
	Reply Message Wait Time	1	Define the amount of time in 20 millisecond increments that the master station will wait after receiving an ACK (to a master-initiated message) before polling the slave station for a reply. Choose a time that is, at minimum, equal to the longest time that a slave station needs to format a reply packet. This would typically be the maximum scan time of the slave station.
	ACK Timeout	50	The amount of time in 20 millisecond increments that you want the processor to wait for an acknowledgment to the message it has sent before the processor retries the message or the message errors out. This timeout value is also used for the poll response timeout. See page 13-47 for recommendations to minimize this value.
	RTS Off Delay	0	Defines the amount of time in 20 millisecond increments that elapses between the end of the message transmission and the de-assertion of the RTS signal. This time delay is a buffer to make sure that the modem has transmitted the message but should normally be left at zero. See page 13-62 for further guidelines for setting this parameter.
	RTS Send Delay	0	Defines the amount of time in 20 millisecond increments that elapses between the assertion of the RTS signal and the beginning of the message transmission. This time allows the modem to prepare to transmit the message. The Clear-to-Send (CTS) signal must be high for transmission to occur.
	Pre-Transmit Delay	0	Defines the amount of time in 1 millisecond increments that elapses between when the processor has a message to send and when it asserts the RTS signal.
	Message Retries	3	Defines the number of times a master station retries a message before it declares the message undeliverable.

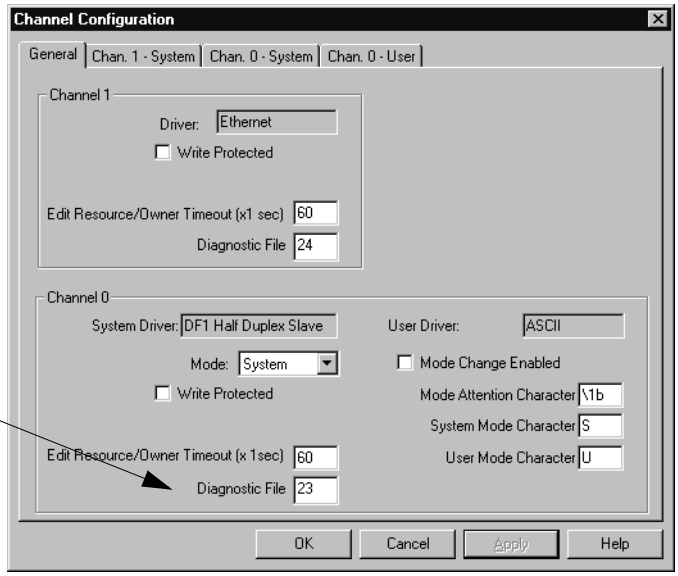
Configuring Channel 0 for DF1 Half-Duplex Slave

To configure an SLC 5/03, SLC 5/04 or SLC 5/05 processor channel 0 for DF1 half-duplex slave, do the following using your programming software:

To bring up the Channel Configuration interface, double-click on the Channel Configuration icon.



Define the location of the diagnostic file used for Channel Status here. For Channel Status details, see page 13-57.



1. On the Channel 0 tab, choose DF1 Half-Duplex Slave for your Driver.
2. Configure the communication driver characteristics according to Table 13.21.

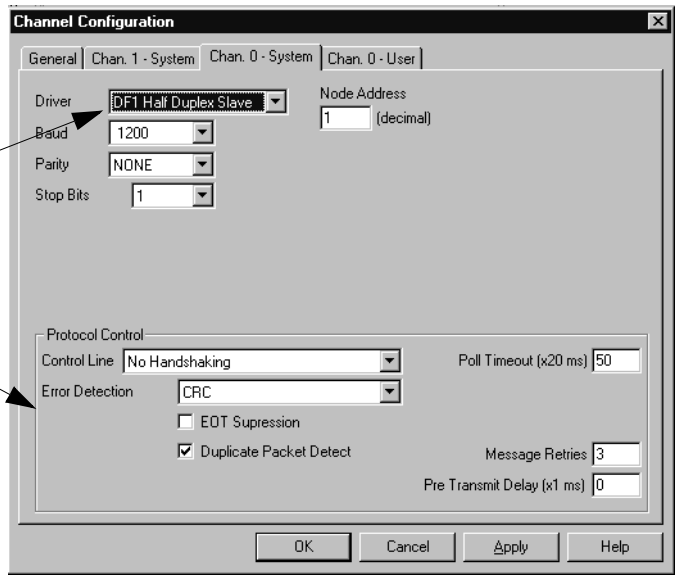


Table 13.21 Define these parameters when configuring a SLC 5/03, 5/04, or 5/05 processor as a slave station.

Tab:	Parameter:	Default:	Selections:
General	Diagnostic File	0	SLC 5/03 (OS 302, Series C or higher), SLC 5/04 (OS 401, Series C or higher) and SLC 5/05 only. Select an unused file (9 to 255) to store channel status information. You must define a diagnostic file in order to be able to view channel 0 status. See Table 13.22 on page 13-58 for a file description.
Chan. 0 System	Baud Rate	1200	Select a communication rate that all devices in your system support. Configure all devices in the system for the same communication rate.
	Parity	None	Parity provides additional message packet error detection. To implement even parity checking, choose Even. To implement no parity checking, choose None.
	Stop Bits	1	Match the number of stop bits to the device with which you are communicating.
	Node Address	0	A node address identifies the processor on the DF1 half-duplex link. Each station on a link must have a unique node address. Choose an address between 0_{10} and 254_{10}. Node address 255_{10} is the broadcast address, which you cannot select as a station's individual address.
	Control Line	No Handshaking	This parameter defines the mode in which the driver operates. Choose a method appropriate for your system's configuration: • If you are not using a modem, choose NO HANDSHAKING. • If the master modem is full duplex and the slave modem is half-duplex, choose HALF-DUPLEX WITH CONTINUOUS CARRIER. • If all the modems in the system are half-duplex, choose HALF-DUPLEX WITHOUT CONTINUOUS CARRIER. See page 13-60 for descriptions of the control line operation settings.

Table 13.21 Define these parameters when configuring a SLC 5/03, 5/04, or 5/05 processor as a slave station.

Tab:	Parameter:	Default:	Selections:
Chan. 0 System	Error Detection	CRC	With this selection, you choose the how the processor checks the accuracy of each DF1 packet transmission. BCC: This algorithm provides a medium level of data security. It cannot detect: • transposition of bytes during transmission of a packet • the insertion or deletion of data values of zero within a packet CRC: This algorithm provides a higher level of data security. Select an error detection method that all devices in your configuration can use. When possible, choose CRC.
	Duplicate Packet Detect	Enabled	Duplicate Detect lets the SLC detect if it has received a message that is a duplicate of its most recent message from the master station. If you choose duplicate detect, the processor will acknowledge (ACK) the message but will not act on it since it has already performed the message's task when it received the command from the first message. If you want to detect duplicate packets and discard them, check this parameter. If you want to accept duplicate packets and execute them, leave this parameter unchecked.
	Poll Timeout	3000	The timer keeps track of how often the station is polled. If the station has a message to send, it starts a timer. If the poll timeout expires before the message timeout, which you specify in the MSG control block, the MSG error bit is set and the message is removed from the transmit queue. If the message timeout, which you specify in the MSG control block, expires before the poll timeout expires, the MSG error bit and MSG timeout bit are set. The poll timeout can be disabled by entering a zero. See page page 13-57 for recommendations to minimize this value
	RTS Off Delay	0	Defines the amount of time in 20 millisecond increments that elapses between the end of the message transmission and the de-assertion of the RTS signal. This time delay is a buffer to make sure that the modem has transmitted the message, but should normally be left at zero. See page 13-62 for further guidelines for setting this parameter.
	RTS Send Delay		Defines the amount of time in 20 millisecond increments that elapses between the assertion of the RTS signal and the beginning of the message transmission. This time allows the modem to prepare to transmit the message. The Clear-to-Send (CTS) signal must be high for transmission to occur.See page 13-62 for further guidelines for setting this parameter.
	Message Retries		Defines the number of times a slave station resends its message to the master station before the slave station declares the message undeliverable.
	Pre-Transmit Delay		Defines the amount of time in 1 millisecond increments that elapses between when the processor has a message to send and when it asserts the RTS signal.
	EOT Suppression		If you want to minimize traffic on the network, you can choose to have the slave station not send EOT packets to the master station. When EOT packets are suppressed, the master station automatically assumes a slave station has no data to give if the slave station does not send a message packet as a response to a poll. A disadvantage of suppressing EOTs is that the master station cannot distinguish between an active station that has no data to transmit and an inactive station. A possible application for suppressing EOTs is the following: conserving power with a radio modem because the radio transmitter does not have to power-up to transmit a DLE EOT packet ("no data to give" packet). To suppress EOTs, check this parameter. To have the processor send EOTs, leave this parameter unchecked.

Configuring Channel 0 Poll Timeout

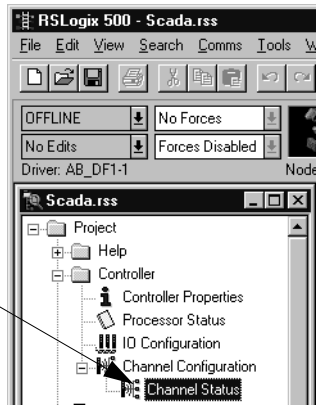
The Channel 0 Poll Timeout is only used when the DF1 half-duplex slave is initiating MSG instructions in ladder logic. This implies that the Master is most likely configured for Standard Polling Mode. The minimum Poll Timeout value is dependent on the maximum Master poll scan rate. Since the Master's polling and the Slave's triggering of a MSG instruction are asynchronous events, it is possible that in the instant just after the slave was polled, the MSG instruction gets triggered. This means the MSG instruction will remain queued-up for transmission until the Master has polled every other slave first. Therefore, the minimum Slave channel 0 Poll Timeout value is equal to the maximum Master poll scan rate rounded up to the next 20 ms increment.

Minimum Channel 0 Poll Timeout = (maximum Master scan poll rate)

DF1 Half-Duplex Slave Channel Status

Channel Status data is stored in the diagnostic file defined on the Channel 0 Configuration screen. See Table 13.22 for information regarding the diagnostic counter data displayed.

Double-click on the Channel Status Icon Located beneath the Configuration icon to bring up the Channel Status screen.



See Table 13.22 for details concerning the DF1 Half-Duplex Slave Channel Status Screen.

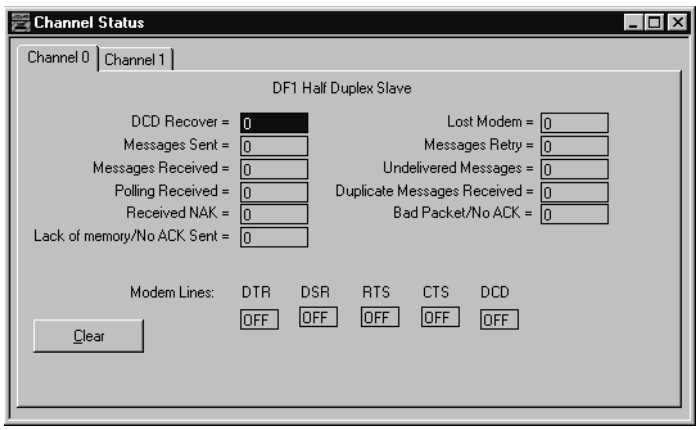


Table 13.22 SLC 5/03 and Higher Channel 0 DF1 Half-Duplex Slave Channel Status

Status Field	Diagnostic File Location	Definition
DCD Recover	word 11	The number of times the processor detects the DCD handshaking line has gone low to high
Messages Sent	word 1	The total number of DF1 messages sent by the processor (including message retries)
Messages Received	word 2	The number of messages received with no errors
Polling Received	word 6	The number of master poll packets received by the processor
Received NAK	word 5	The number of NAKs received by the processor
Lack of Memory/No ACK Sent	word 8	The number of times the processor could not receive a message because it did not have available memory
Lost Modem	word 12	The number of times the lost modem bit has gone low to high
Messages Retry	word 4	The number of message retries sent by the processor
Undelivered Messages	word 3	The number of messages that were sent by the processor but not acknowledged by the destination device
Duplicate Messages Received	word 9	The number of times the processor received a message packet identical to the previous message packet
Bad Packet/No ACK	word 7	The number of incorrect data packets received by the processor for which no ACK was returned
DTR (Data Terminal Ready)	word 0;bit 4	The status of the DTR handshaking line (asserted by the processor)
DSR (Data Set Ready)	word 0;bit 2	The status of the DSR handshaking line (received by the processor)
RTS (Request to Send)	word 0;bit 1	The status of the RTS handshaking line (asserted by the processor)
CTS (Clear to Send)	word 0;bit 0	The status of the CTS handshaking line (received by the processor)
DCD (Carrier Detect)	word 0;bit 3	The status of the DCD handshaking line (received by the processor)

Using Modems that Support DF1 Communication Protocols

The types of modems that you can use with SLC processors include dial-up phone modems, leased-line modems, radio modems and line drivers. For point-to-point full-duplex modem connections, use DF1 full-duplex protocol. For point-to-multipoint modem connections, use DF1 half-duplex master and slave protocols. In this case, one (and only one) of the other devices must be configured for DF1 half-duplex master protocol. *Do not attempt to use DH-485 protocol through modems under any circumstance.*

Dial-Up Phone Modems

Dial-up phone line modems support point-to-point full-duplex communications. Normally an SLC processor, on the initiating or receiving end of the dial-up connection, will be configured for DF1 full-duplex protocol with the control line parameter set for "Full-Duplex Modem". See page 13-60 for details on the operation of

the RS-232 modem control signals when “Full-Duplex Modem” is selected.

When an SLC processor is the initiator of the dial-up connection, use one of the ASCII write instructions to send out the “AT” dial-up string (for example: ATDT 555-1212). The status file modem lost bit (S:5/14) provides the feedback that the connection has been successfully made. To hang up the connection, use the ASCII AHL instruction to temporarily lower the DTR signal.

Leased-Line Modems

Leased-line modems are used with dedicated phone lines that are typically leased from the local phone company. The dedicated lines may be in a point-to-point topology supporting full-duplex communications between two modems or in a point-to-multipoint topology supporting half-duplex communications between three or more modems. In the point-to-point topology, configure the SLC processor for DF1 full-duplex protocol with the control line parameter set to “Full-Duplex Modem”. In the point-to-multipoint topology, configure the SLC processors for DF1 half-duplex master or slave protocol with the control line parameter set to “Half-Duplex Modem without Continuous Carrier”. See page 13-61 for details on the operation of the RS-232 modem control signals when “Half-Duplex Modem without Continuous Carrier” is selected.

Radio Modems

Radio modems may be implemented in a point-to-point topology supporting either half-duplex or full-duplex communications, or in a point-to-multipoint topology supporting half-duplex communications between three or more modems. In the point-to-point topology using full-duplex radio modems, configure the SLC processors for DF1 full-duplex protocol. In the point-to-point topology using half-duplex radio modems, or point-to-multipoint topology using half-duplex radio modems, configure the SLC processors for DF1 half-duplex master or slave protocol. If these radio modems require RTS/CTS handshaking, configure the control line parameter to “Half-Duplex Modem without Continuous Carrier”. See page 13-61 for details on the operation of the RS-232 modem control signals when “Half-Duplex Modem without Continuous Carrier” is selected.

Line Drivers

Line drivers, also called short-haul "modems", do not actually modulate the serial data, but rather condition the electrical signals to operate reliably over long transmission distances (up to several miles). Allen-Bradley's AIC+ Advanced Interface Converter is a half-duplex line driver that converts an RS-232 electrical signal into an RS-485 electrical signal, increasing the signal transmission distance from 50 to 4000 feet. In a point-to-point line driver topology, configure the SLC processor for DF1 full-duplex protocol. In a point-to-multipoint line driver topology, configure the SLC processors for DF1 half-duplex slave protocol. If these line drivers require RTS/CTS handshaking, configure the control line parameter to "Half-Duplex Modem without Continuous Carrier".

Modem Control Line Operation in SLC 5/03, SLC 5/04 and SLC 5/05 Processors

The following explains the operation of the SLC 5/03, SLC 5/04, and SLC 5/05 processors when you configure the RS232 channel for the following applications.

DF1 Full-Duplex

When you configure the SLC 5/03, SLC 5/04, and SLC 5/05 processors for full-duplex DF1, the following control line operation takes effect:

No Handshaking Selected - DTR is always active and RTS is always inactive. Receptions and transmissions take place regardless of the states of DSR, CTS, or DCD inputs. This selection should only be made when the SLC 5/03, SLC 5/04 and SLC 5/05 processors are directly connected to another DTE device.

Full-Duplex Modem Selected - DTR and RTS are always active except at the following times. If DSR goes inactive, both DTR and RTS are dropped for 1 to 2 seconds then reactivated. The modem lost bit (S:5/14) is turned on immediately. While DSR is inactive, the state of DCD is ignored. Neither receptions nor transmissions are performed.

If DCD goes inactive while DSR is active, then receptions are not allowed. If DCD remains inactive for 9 to 10 seconds, then DTR is set inactive until DSR goes inactive. At this point, the modem lost bit is also set. If DSR does not go inactive, then DTR is raised again in 5 to 6 seconds.

Transmission requires all three inputs (CTS, DCD, and DSR) to be active. Whenever DSR and DCD are both active, the modem lost bit is reset.

DF1 Half-Duplex Slave

When you configure the SLC 5/03, SLC 5/04, and SLC 5/05 processors for DF1 half-duplex slave, the following control line operation takes effect:

No Handshaking Selected - DTR is always active and RTS is always inactive. Receptions and transmissions take place regardless of the states of DSR, CTS, or DCD inputs. This selection should only be made when the processor is directly connected to another DTE device.

Half-Duplex Modem with Continuous Carrier Selected - DTR is always active and RTS is only activated during transmissions (and any programmed delays before or after transmissions). The handling of DCD and DSR are exactly the same as with Full-Duplex Modem. Transmissions require CTS and DSR to be active.

Half-Duplex Modem without Continuous Carrier Selected - This is exactly the same as Half-Duplex Modem with Continuous Carrier except monitoring of DCD is not performed. DCD is still required for receptions but is not required for transmissions. Transmissions still require CTS and DSR. The modem lost bit will only be set when DSR is inactive.

DF1 Half-Duplex Master

When you configure the SLC 5/03, SLC 5/04, and SLC 5/05 processors for DF1 half-duplex master, the following control line operation takes effect:

No Handshaking Selected - DTR is always active and RTS is always inactive. Receptions and transmissions take place regardless of the states of DSR, CTS, or DCD inputs. This selection should only be made when the processor is directly connected to another DTE device.

Full-Duplex Modem Selected - DTR and RTS are always active except at the following times. If DSR goes inactive, both DTR and RTS are dropped for 1 to 2 seconds then reactivated. The modem lost bit (S:5/14) is turned on immediately. While DSR is inactive, the state of DCD is ignored. Neither receptions nor transmissions are performed.

If DCD goes inactive while DSR is active, then receptions are not allowed. If DCD remains inactive for 9 to 10 seconds, then DTR is set inactive until DSR goes inactive. At this point, the modem lost bit is

also set. If DSR does not go inactive, then DTR is raised again in 5 to 6 seconds.

Transmission requires all three inputs (CTS, DCD, and DSR) to be active. Whenever DSR and DCD are both active, the modem lost bit is reset.

Half-Duplex Modem without Continuous Carrier Selected - DTR is always active and RTS is only active during transmissions (and any programmed delays before and after transmissions). The processor does not monitor DCD.

If DSR goes inactive, RTS is dropped. The modem lost bit (S:5/14) is turned on immediately. While DSR is inactive, neither receptions nor transmissions are performed.

Transmission requires two inputs, CTS and DSR, to be active. Whenever DSR is active, the modem lost bit is reset.

RTS Send Delay and RTS Off Delay Parameters

Through your programming software, the parameters RTS Send Delay and RTS Off Delay give you the flexibility of selecting modem control during transmissions. These parameters only apply when you select half-duplex modem with or without continuous carrier.

For use with half-duplex modems that require extra time to “key up” their transmitter even after they have activated CTS, the RTS Send Delay specifies in 20 millisecond increments the amount of delay time after activating RTS to wait before checking to see if CTS has been activated by the modem. If CTS is not yet active, RTS remains active and as long as CTS is activated within one second, the transmission occurs. After one second, if CTS is still not activated, then RTS is set inactive and the transmission is aborted.

For modems that do not supply a CTS signal at all, tie RTS to CTS and use the shortest delay possible without losing reliable operation.

TIP

If an RTS Send Delay of 0 is selected, then transmission starts as soon as CTS is activated. If CTS does not go active within 1 second after RTS is raised, RTS is set inactive and the transmission is aborted.

Certain modems will drop their carrier link when RTS is lost even though the transmission has not been finished yet. The RTS Off Delay parameter specifies in 20 millisecond increments the delay between when the last serial character is sent to the modem and when RTS is deactivated. This gives the modem extra time to transmit the last character of a packet.

ASCII Communications

The SLC 5/03 (OS301 and higher), SLC 5/04, and SLC 5/05 processors support user-defined ASCII protocol by configuring RS-232 (channel 0) for the User mode. In the User mode, all received data is placed in a buffer. To access the data, use the ASCII instructions in your ladder program. See Chapter 10 for more information on ASCII instructions. You can also send ASCII string data to most attached devices that accept ASCII protocol.

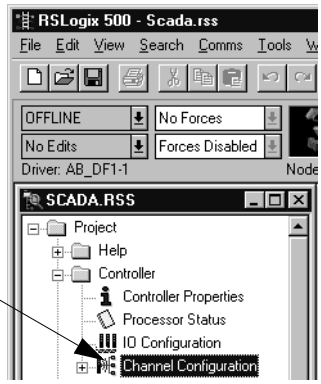
TIP

Only ASCII instructions can be used when User mode is configured. If you use a Message (MSG) instruction that references channel 0, an error occurs.

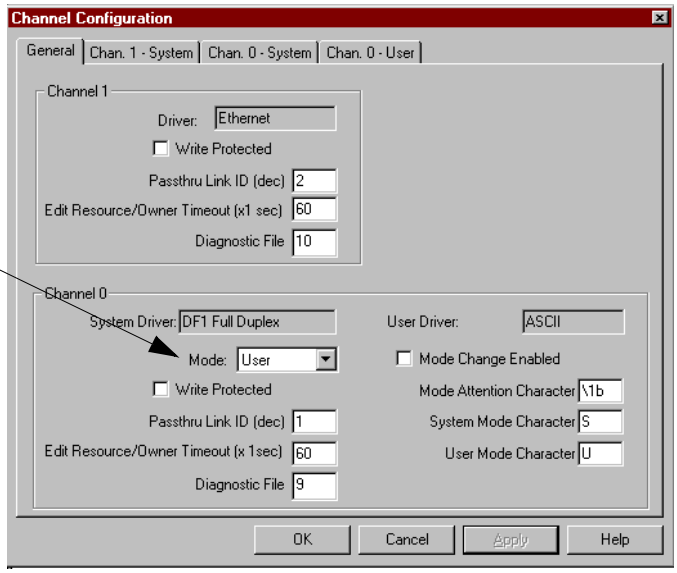
Configuring Channel 0 for ASCII Communications

To configure an SLC 5/03, SLC 5/04 or SLC 5/05 processor channel 0 for DF1 full-duplex, do the following using your programming software:

To bring up the Channel Configuration interface, double-click on the Channel Configuration icon.



Select User Mode



1. On the Channel 0 user tab, choose ASCII for your Driver.
2. Configure the communication driver characteristics according to Table 13.23.

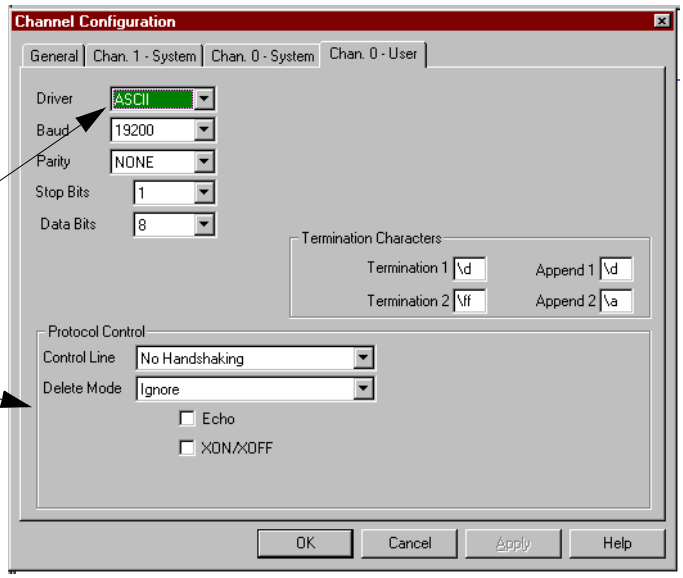


Table 13.23 Define these communication parameters when configuring an SLC 5/03, SLC 5/04, or SLC 5/05 processor for ASCII communication.

Tab:	Parameter:	Default:	Selections:
Channel 0 User	Baud Rate	19200	Toggles between the communication rate of 110, 300, 600, 1.2K, 2.4K, 4.8K, 9.6K and 19.2K (additional rate of 38.4K for SLC 5/04 and SLC 5/05 only).
	Parity	None	Toggles between None, Odd, and Even.
	Stop Bits	1	Toggles between 1, 1.5, and 2.
	Data Bits	8	Toggles between 7 and 8.
	Delete Mode	Ignore	Toggles between Ignore, CRT, and printer. This parameter is dependent on the Echo parameter being Enabled.
	Echo	Disabled	Toggles between Disabled and Enabled.
	RTS Off Delay	0	Allows you to select the RTS off delay value in increments of 20 ms. Valid range is 0 to 65535 (in 20 ms increment). Only valid for Half-Duplex Control Line settings.
	RTS Send Delay	0	Allows you to select the RTS send delay value in increments of 20 ms. Valid range is 0 to 65535 (in 20 ms increment). Only valid for Half-Duplex Control Line settings.
	Control Line	No Handshaking	Toggles between No Handshaking, Half-Duplex With Continuous Carrier, Half-Duplex Without Continuous Carrier, and Full-Duplex Modem.
	XON/XOFF	Disabled	Toggles between Disabled and Enabled.
	Termination 1 Termination 2	\d \ff	Specify \ff for no termination character.
	Append 1 Append 2	\d \a	Specify \ff for no append character.

SLC 5/05 Embedded Web Server Capability

SLC 5/05 processors with OS501, Series C, FRN 5 (or higher) include an embedded web server which allows viewing of module information, TCP/IP configuration and diagnostic information via Ethernet using a standard web browser.

In order to view the web server main menu from a standard web browser, type in http://www.xxx.yyy.zzz for the web address, where www.xxx.yyy.zzz is the IP address of the SLC 5/05 processor.

The following topics appear on the main menu:

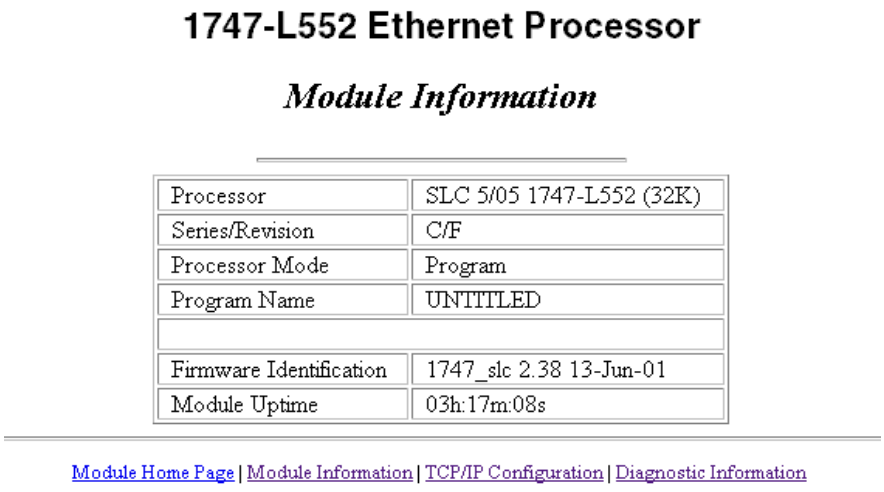
- Module Information
- TCP/IP Configuration
- Diagnostic Information

Figure 13.2 1747 SLC 5/05 Main Page

Module Information

The module information page displays a table showing information about the processor. The specific information displayed includes the processor model, series/revision of the processor, mode of the processor and the name of the program in the processor. Also displayed is the revision/build of the Ethernet firmware and the module uptime (time since power was last applied).

Figure 13.3 1747 SLC 5/05 Module Information Page



1747-L552 Ethernet Processor

Module Information

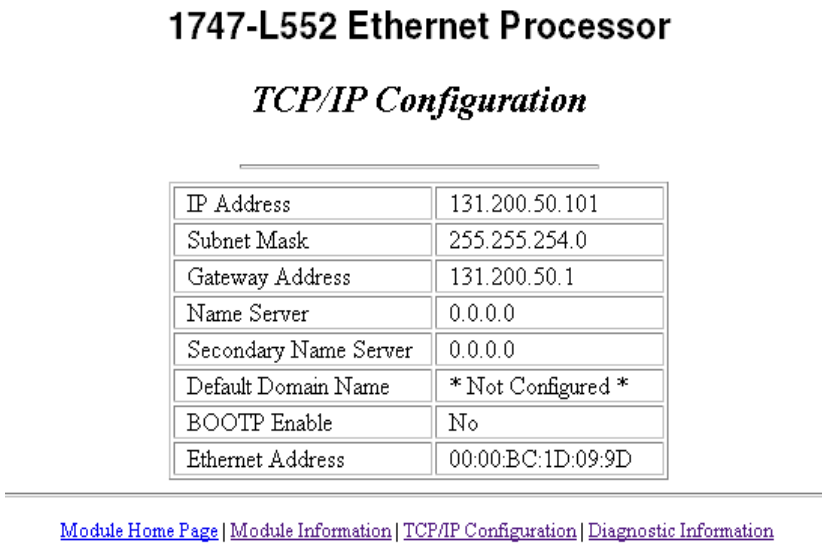
Processor	SLC 5/05 1747-L552 (32K)
Series/Revision	C/F
Processor Mode	Program
Program Name	UNTITLED
Firmware Identification	1747_slc 2.38 13-Jun-01
Module Uptime	03h:17m:08s

Module Home Page | Module Information | TCP/IP Configuration | Diagnostic Information

TCP/IP Configuration

This page will display a table showing information about the current TCP/IP configuration parameters. Included are the IP address of the module, the subnet mask, gateway address, the Ethernet hardware address and whether BOOTP is enabled. Also included for future use are the name server, secondary name server and the default domain name parameters.

Figure 13.4 1747 SLC 5/05 TCP/IP Configuration Page



1747-L552 Ethernet Processor

TCP/IP Configuration

IP Address	131.200.50.101
Subnet Mask	255.255.254.0
Gateway Address	131.200.50.1
Name Server	0.0.0.0
Secondary Name Server	0.0.0.0
Default Domain Name	* Not Configured *
BOOTP Enable	No
Ethernet Address	00:00:BC:1D:09:9D

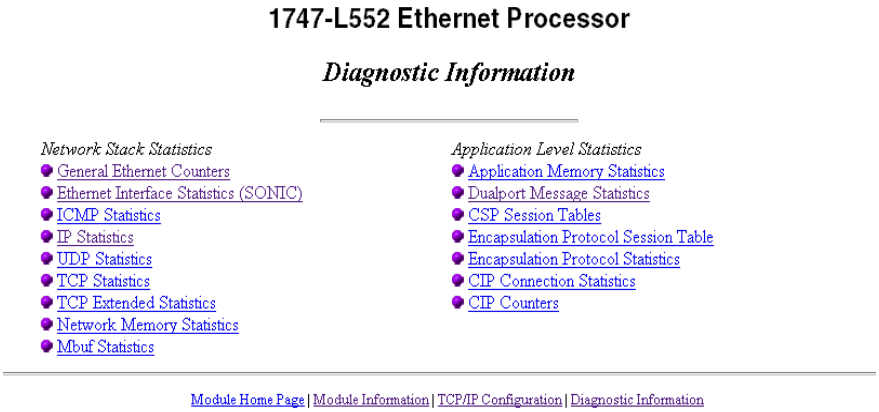
Module Home Page | Module Information | TCP/IP Configuration | Diagnostic Information

Diagnostic Information

This section gives you access to the various diagnostic information screens that are available. This is divided into two sections, the *Network Stack Statistics* and *Application Level Statistics*. The *Network Stack Statistics* detail information about the TCP.IP stack while the *Application Level Statistics are* related to the Allen-Bradley Client Server Protocol (CSP) and Control and Information Protocol (CIP) diagnostics.

The individual diagnostic screens will automatically refresh using a time which is configurable by the user and defaults to 15 seconds.

Figure 13.5 1747 SLC 5/05 Diagnostic Information Main Page



Network Stack Statistics

The General Ethernet Statistics page displays general messaging statistics which are also available through programming software monitoring of the diagnostics file of the processor. Generally information from this page is sufficient in isolating most network problems. The remainder of the Network Stack Statistic pages contain more detailed internal protocol counters that can be used by Rockwell Automation Technical Support personnel to troubleshoot more advanced network problems.

Figure 13.6 1747 SLC 5/05 General Ethernet Counters

1747-L552 Ethernet Processor

General Ethernet Counters

Commands Sent	0	Replies Sent	0
Commands Received	0	Replies Received	0
		Replies Sent With Error	0
		Replies Received With Error	0
		Replies Timed Out	0
In Octets	75706126	Out Octets	118297
In Packets	205656	Out Packets	806
Alignment Errors	0	FCS Errors	0
Carrier Sense Errors	0	Excessive Collisions	0
Excessive Deferrals	0	MAC Receive Errors	0
MAC Transmit Errors	0	Single Collisions	0
Multiple Collisions	0	Deferred Transmissions	0
Late Collisions	0	Packet Storms	10

Clear Counters | Refresh counters every 15 seconds. | Change | Default | Disable

Module Home Page | Module Information | TCP/IP Configuration | Diagnostic Information

- **Commands Sent** - Total number of PCCC commands sent by the interface
- **Replies Sent** - Total number of PCCC replies sent by the interface
- **Commands Received** - Total number of PCCC commands received by the interface
- **Replies Received** - Total number of PCCC replies received by the interface
- **Replies Sent With Error** - Total number of PCCC replies with error status sent by the interface
- **Replies Received With Error** - Total number of PCCC replies with error status received by the interface
- **Replies Timed Out** - Total number of PCCC replies that were not received within the time period specified on the Ethernet Configuration screen
- **In Octets** - Total number of octets received by the interface
- **Out Octets** - Total number of octets sent by the interface
- **In Packets** - Total number of packets received by the interface, including broadcast packets
- **Out Packets** - Total number of packets sent by the interface, including broadcast packets
- **Alignment Errors** - Count of frames received that are not an integral number of octets in length
- **FCS Errors** - Count of frames that do not pass the FCS check
- **Carrier Sense Errors** - Number of times that the carrier sense condition was lost or never asserted when attempting to transmit a frame

- **Excessive Collisions** - Count of frames for which transmission fails due to excessive collisions
- **Excessive Deferrals** - Count of frames for which transmission is deferred for an excessive period of time
- **MAC Receive Errors** - Count of frames for which reception on an interface fails due to an internal MAC sublayer receive error
- **MAC Transmit Errors** - Count of frames for which transmission fails due to an internal MAC sublayer transmit error
- **Single Collisions** - Count of successfully transmitted frames for which transmission is inhibited by exactly collision
- **Multiple Collisions** - Count of successfully transmitted frames for which transmission is inhibited by more than one collision
- **Deferred Transmissions** - Count of frames for which the first transmission attempt is delayed because the medium is busy
- **Late Collisions** - Number of times that a collision is detected later than 512 bit-times into the transmission of a packet
- **Packet Storms** - Number of times the SONIC driver has entered storm or throttle back operation due to excessive traffic

Application Level Statistics

The Application Level Statistics give detailed information on the Client Server Protocol (CSP) and Control and Information Protocol (CIP) counters. This information includes memory usage, inbound and outbound connection information and packet processing. Generally information from the first four pages would be of interest to the user.

- **Application Memory Statistics** - Which gives information on the number of connections available and the number currently in use for inbound and outbound connections.
- **Dualport Memory Statistics** - Which lists the number of Command/Reply packets being processed between the SLC-5/05 motherboard and the Ethernet daughtercard.
- **CSP Session Table** - Which shows inbound/outbound connection information for the Client Server Protocol (CSP) connections.
- **Encapsulation Protocol Session Table** - Which shows inbound/outbound connection information for the Control and Information Protocol (CIP) connections.

The remainder of the Application Level Statistic pages contain more detailed internal protocol counters that can be used by Rockwell Automation Technical Support personnel for advanced CIP trouble shooting.

Messaging Examples

The purpose of this chapter is to illustrate some of the more common but elaborate messaging examples using the SLC500 processors.

The first section will document examples of SLC processors implementing processor passthru features.

The later section will document examples of remote MSG's through a complex system bridging together multiple networks.

Not all examples will appear with a full detailed step by step procedure necessary.

Step by Step procedures are available for all of the examples in KnowledgeBase Documents at the following website:

http://www.ab.com/support/

Select SLC 500 (1747)

from the menu tree and search for the associated Knowledge Base Document Number referenced after the associated example.

It may be necessary to register and obtain a username and password if accessing the website for the first time or if your web browser does not have cookies enabled.

Local versus Remote type Message

Local Message

A Local MSG is used to transmit data from one processor to another on the same network. If two processors are connected together a Local type message is used to transfer information from one to the other.

Remote Message

A Remote MSG is used to exchange information to a device that is not connected on the local network. A device (another processor or an

actual bridging device) on the local network will act as a bridge or gateway to the destination network.

Remote Terminology

Remote Bridge Address

Remote Bridge Address is the remote node address of the bridge device used to connect two networks together.

TIP

SLC Fixed Processors, SLC 5/01 and SLC 5/02 are all non-remote MSG capable processors. When issuing a remote message and the target device is non-remote bridge, a remote bridge address is needed.

Remote Station Address

Remote Station Address is the final destination address of a remote MSG instruction.

Remote Bridge Link ID

Remote Bridge Link ID is a user assigned address that differentiates same type networks from each other when multiple networks are connected together. Passthru Link ID's are required when initiating a Remote MSG.

All processors on a particular network need to have all the Passthru Link ID's set to the same address. Valid Link ID addresses are from 1 to 65,534.

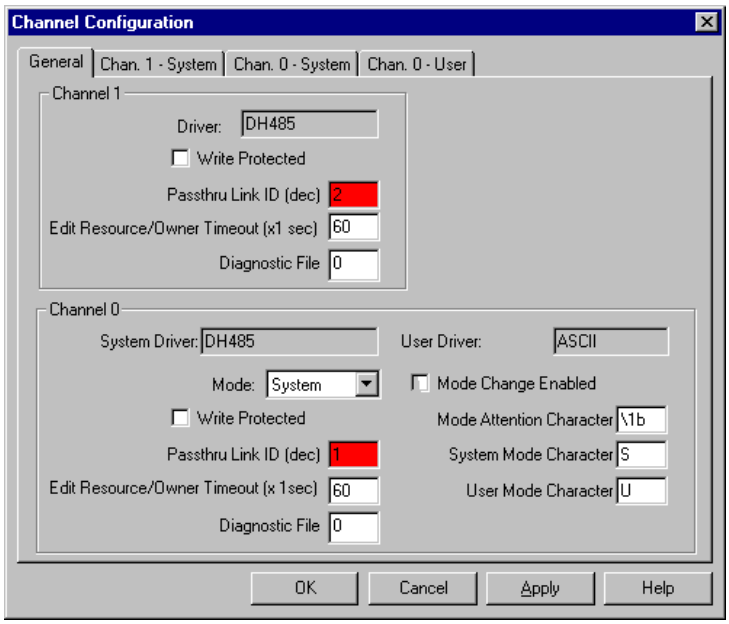
TIP

Make sure that all processors on the same network share the same Link ID.

TIP Link ID's are modified in each processors Channel Configuration properties.

The default Passthru Link ID for the SLC 5/03, 5/04 and 5/05 processors CH0 port is 1.

The default Passthru Link ID for the SLC 5/03, 5/04 and 5/05 processors CH1 port is 2.



Using the Passthru Features

There are several types of passthru available in the SLC 5/03, SLC 5/04 and SLC 5/05 processors. Their operation and associated bits are described below.

DH+ to DH-485 Passthru – (All SLC 5/04 processors)

This type allows the SLC 5/04 to act as a bridge between a DH+ network and a DH-485 network. When bit S:34/0 is reset, communication packets coming into channel 0 (configured for DH-485) that are not intended for the SLC 5/04 processor are resent out channel 1 onto the DH+ network. Also, communication packets coming into channel 1 (DH+) that are not intended for the SLC 5/04

processor are re-sent out channel 0 onto the DH-485 network. This activity has some effect on the scan time of the SLC 5/04 processor's ladder program, but the effects are not dramatic because only one passthru packet is re-routed per scan.

DF1 to DH+ Passthru – (SLC 5/04 OS401 and above processors)

This type allows you to connect a computer to the SLC 5/04 processor's serial port (channel 0 configured for DF1 Full-Duplex) and access any node on the DH+ network, regardless of the baud rate of the DH+ network. You can also connect a modem to the serial port and dial into any node on the DH+ network. Passthru is enabled when bit S:34/5 is set.

DF1 and DH485 (RS232 port CH0) to Ethernet Channel-to-Channel Passthru (SLC 5/05 Processors OS501 FRN3 and above processors)

This type allows a SLC 5/05 processor to act as a bridge, allowing communication data packets to be passed between the RS232 serial port (Channel 0) and the Ethernet port (Channel 1). This RS232-to-Ethernet bridge operates only when the RS232 serial port is configured for DF1 full-duplex communication or DH485 communication. A maximum of 128 Ethernet devices may be accessed using the passthru feature.

To enable passthru of data packets between the RS232 port and the Ethernet port, the SLC 5/05 processor uses a routing table to cross-reference the one-byte addressing used by DF1 and DH485 protocols with the four-byte IP address needed to support Ethernet communication. The routing table is stored in a user-selectable integer file and uses two word elements of the integer file to store one IP address. The routing table file number must be between 9 and 255. The routing table must be at least two words in length.

Remote I/O Passthru

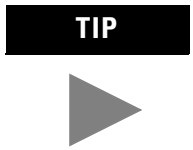
(SLC 5/03 OS302, SLC 5/04 OS401, and SLC 5/05 processors)

This type allows the SLC processor system to act as a bridge between its channel 0 and/or channel 1 network(s) and the remote I/O network supported by the 1747-SN and 1747-BSN Remote I/O modules. This allows personal computers on DH+, DH-485, Ethernet, or DF1 networks to upload or download applications to devices such as PanelView 550s, PanelView 900s, and DataLiners on the remote I/O network.

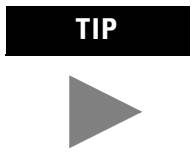
DeviceNet Passthru

(SLC 5/03 OS302, SLC 5/04 OS401, and SLC 5/05 processors)

This type allows the SLC processor system to act as a bridge between its channel 0 and/or channel 1 network(s) and a DeviceNet I/O network supported by the 1747-SDN DeviceNet Scanner I/O module. This allows personal computers on DH+, DH-485, Ethernet, or DF1 networks to upload or download applications to devices such as PanelView 550s, PanelView 900s, and DataLiners on the DeviceNet I/O network.



M0 and M1 length in SDN module advanced configuration must be set for 361 (not the default of 256) to be able to have passthru work.



Passthru is recommended only for the configuration of one parameter at a time - not for a complete network startup. Passthru is slow, slower than RIO passthru. This method is not a replacement for a 1770-KFD, 1784-PCD, or 1784-PCID

Considerations when Passthru is Enabled

Keep the following information in mind when you are using Passthru.

Going Online with an SLC 5/04 Processor using DF1 Full-Duplex

If you want to go on-line using DF1 full-duplex, make sure the destination address under the Full-Duplex Online Configuration Screen is set to the DH+ node address channel 1 of the target SLC 5/04 processor. If the destination address is not set and the SLC 5/04 processor has the DF1 to DH+ passthru feature enabled, the command packets from the programming software may go to a different SLC 5/04 processor than the intended SLC 5/04 processor.

Sending a Message using DF1 Full-Duplex to an SLC 5/04 Processor with DF1 to DH+ Passthru Enabled

If the receiving SLC 5/04 processor has passthru enabled, make sure the target node parameter is set to the channel 1 DH+ address of the SLC 5/04 processor.

Sending a Message using DF1 Full-Duplex from an SLC 5/04 Processor with DF1 to DH+ Passthru Enabled

If you use an SLC 5/04 processor with DF1 to DH+ passthru enabled to send messages out of channel 0 (configured for DF1 full-duplex), you must make sure that the SLC 5/04 processor's DH+ node address appears as the DF1 source address under the Channel 0 System Mode Configuration Screen. If the address is not set correctly, responses coming back to the SLC 5/04 processor may be sent to other nodes on the DH+ network instead.

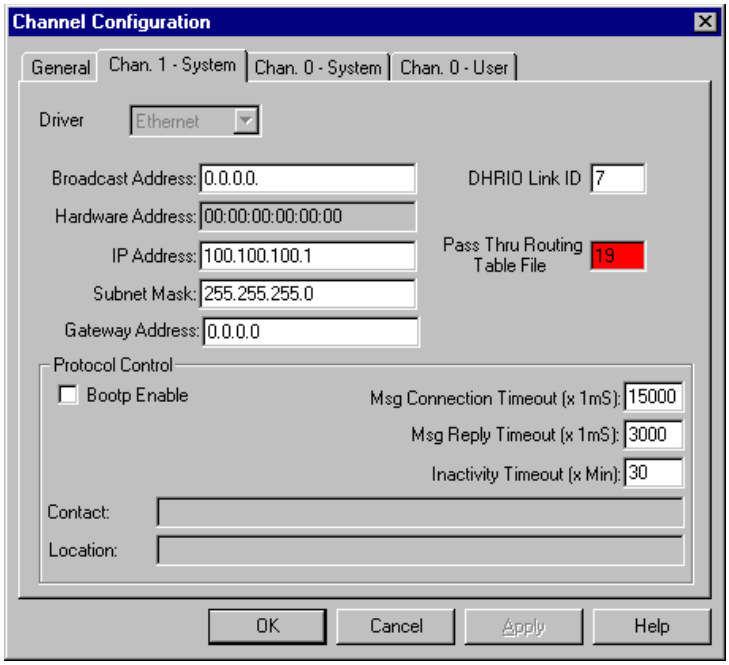
Communicating from an SLC 5/04 Processor using PLC-2 addressing

If you use an SLC 5/04 processor with DF1 to DH+ passthru enabled and are trying to send messages out of channel 0 using the MESSAGE instructions, do not use the 485 CIF message type. Use either the 500CPU or PLC5 message types. If you try to use the 485 CIF message type, the SLC 5/04 processor sending the message will not receive replies from the node it is attempting to communicate with.

Creating and Filling out the Passthru Routing Table File (SLC 5/05 Processor only)

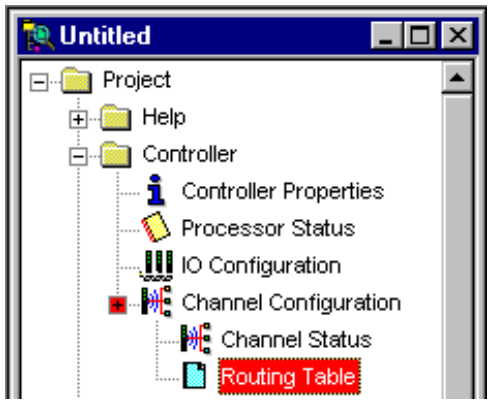
To enable passthru of data packets between the RS232 port (CH0 either set for DF1 Full Duplex or DH485) and the Ethernet port, the SLC 5/05 processor uses a routing table to cross-reference the one-byte addressing used by DF1 and DH485 protocols with the four-byte IP address needed to support Ethernet communication. The routing table is stored in a user-selectable integer Data File and uses two word elements of the integer file to store one IP address.

The Routing Table data file is defined in the Chan. 1 – System menu under Channel Configurations.

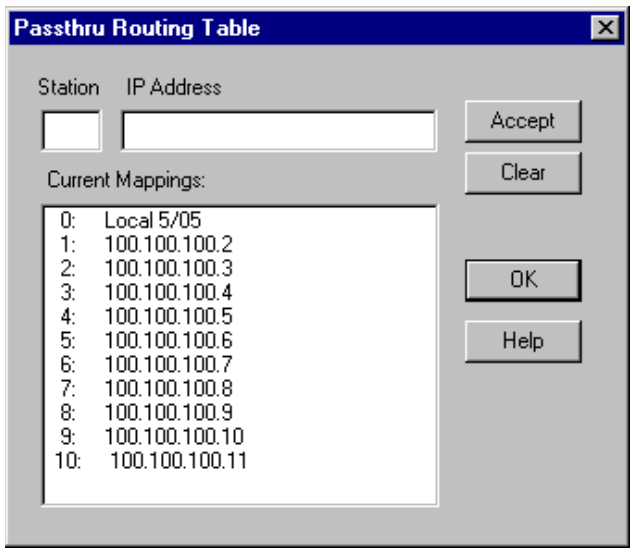


The routing table file number must be between 9 and 255 (a valid data table address). A value of zero will disable the routing table. The routing table must be at least two words in length.

The passthru routing table is located under the channel configuration selection in RSLogix 500 Programming Software. If a Passthru Routing Table File number was entered in the General Tab in the Channel Configuration dialog box, click on the + in front of "Channel Configuration" to reveal the routing table selection.



Double-click on "Routing Table" to view and modify the passthru routing table.



IMPORTANT All Ethernet devices that information needs to be passed to, must have their IP Address listed in the Passthru Routing Table.

Status File Bits - Two status file bits control whether or not the passthru function is enabled.

- S:34/0 Dynamic Configuration Disable Bit

  When this bit is set, DH-485 passthru is disabled. When it is reset, the processor allows packets to be passed from one channel to the other. The default is reset.

- S:34/5 Dynamic Configuration Enable Bit

  When this bit is set, DF1 passthru is enabled. The default is reset.

For further details on the specifics of these bits, please refer to Appendix B, page B-46 through page B-47.

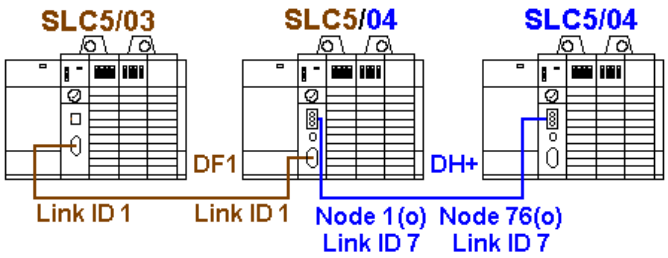
MSG Error Code - When the processor detects an error during the passage of message data, it generates the error code 20H “Host has a problem and cannot communicate”. The cause of the problem could be any of the following:

- The routing table integer file number is out of range (9 to 255).
- The routing table file does not exist in the user program directory or is less than 2 word elements in length.
- The IP Address entry in the routing table does not exist.

SLC 5/04 Passthru Examples

Passthru Example: DF1 to DH+

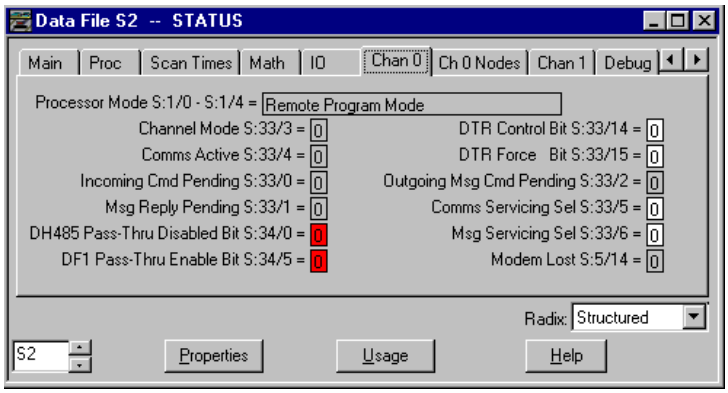
The following illustrates a SLC5/03 sending a local message via DF1 (CH0) to a SLC5/04 processor. The SLC5/04 processor that receives the initial message will send the message out DH+ to the SLC5/04 processor whose address matches the Local Bridge Address on the DH+ network as long as DF1 Passthru is enabled.



1747-CP3 cable is used to connect the SLC5/03 CH0 port to the SLC5/04 CH0 port.

Belden 9463 “Blue Hose” cable is used to connect the SLC5/04 CH1(DH+) ports together.

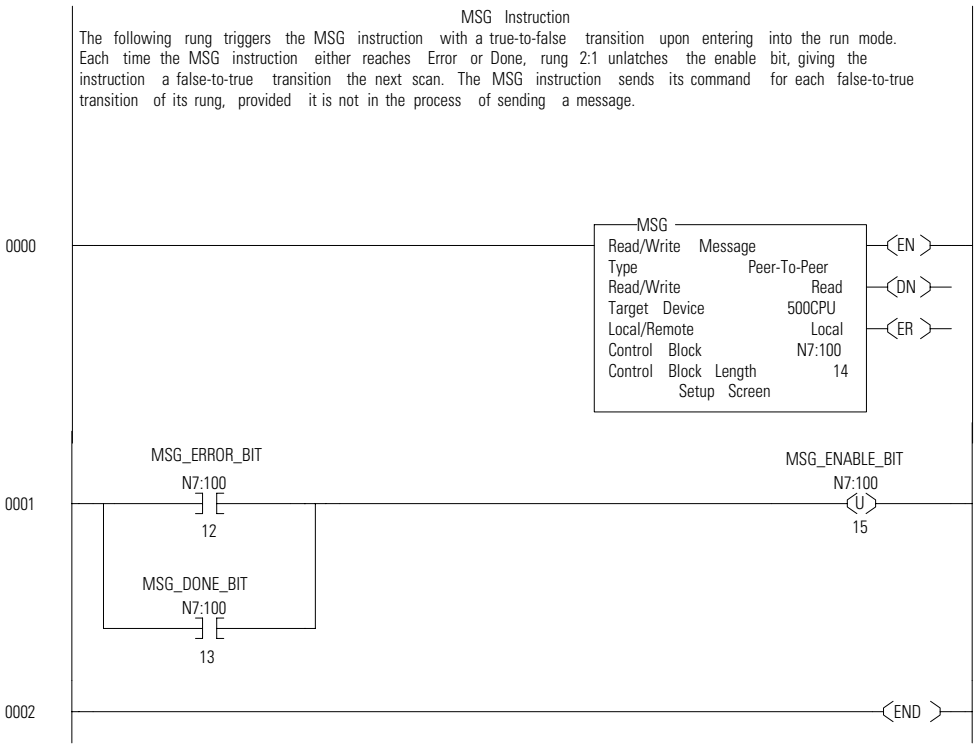
Status File Bit (S:34/5) must be set in the passthru processor in order to enable the DF1-to-DH+ Passthru.



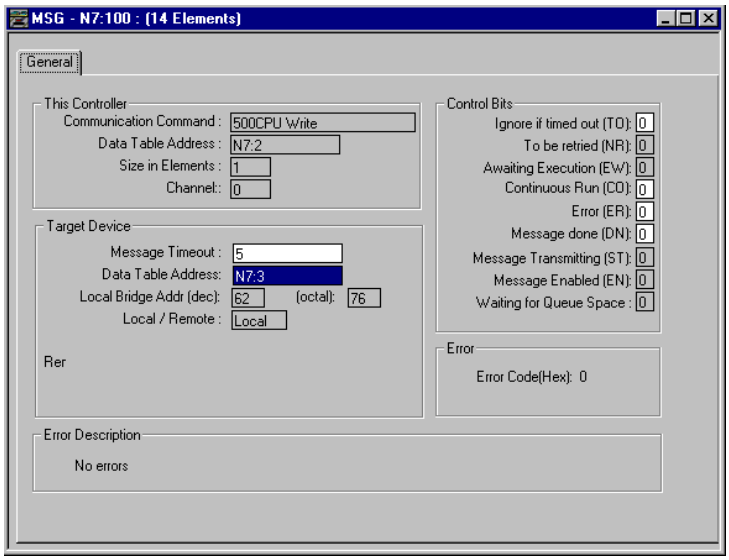
TIP

No programming logic is necessary in the passthru processor.

The following is the ladder logic necessary for the SLC5/03 processor.



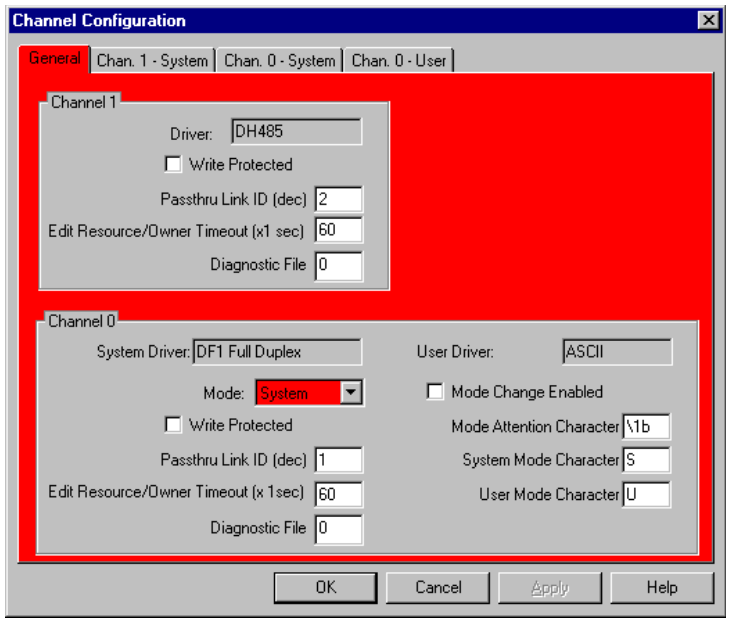
The following is the MSG Setup for the SLC5/03 processor.



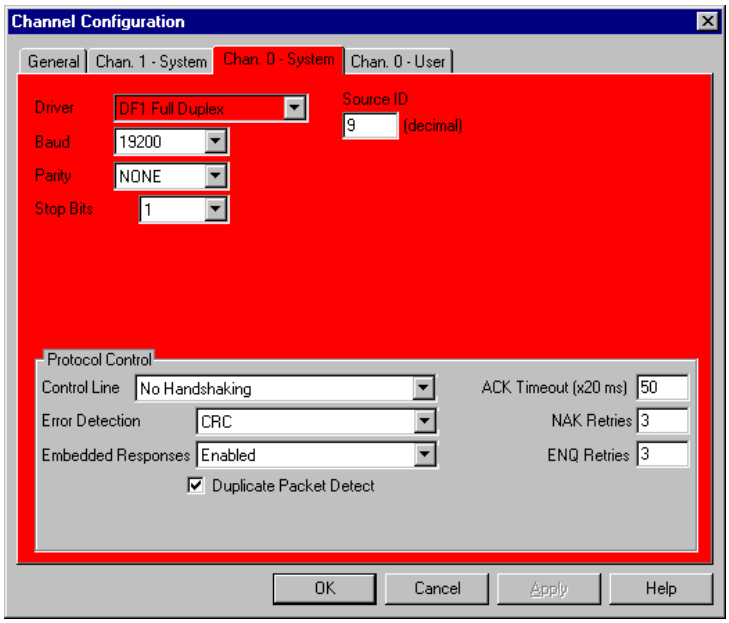
Local Bridge Address is the node number of the destination DH+ address.

The type of MSG instruction is Local.

The following is the Channel configurations MSG Setup for the SLC5/03 processor.



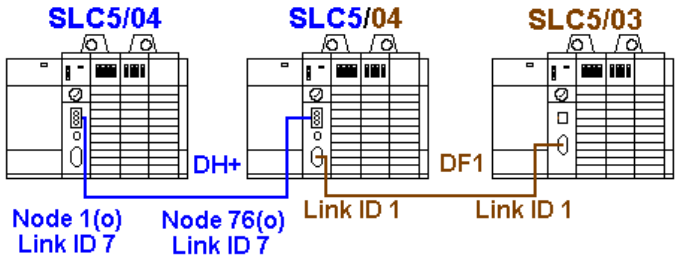
Chan 0 Mode is set for System.

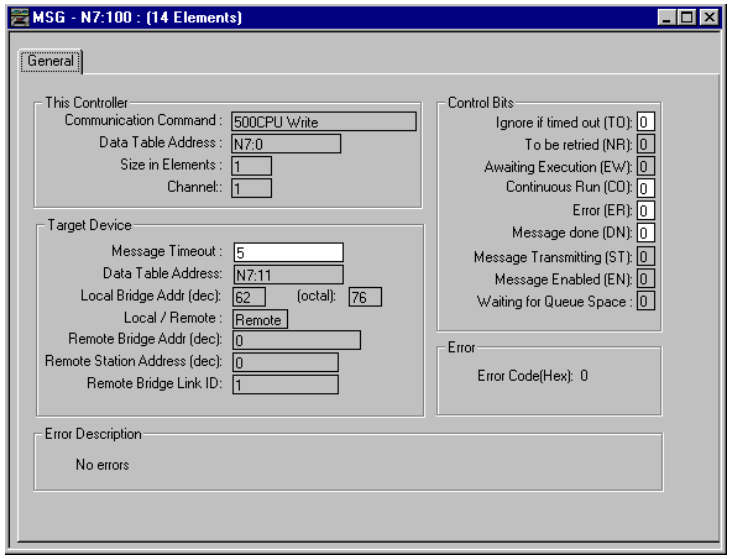


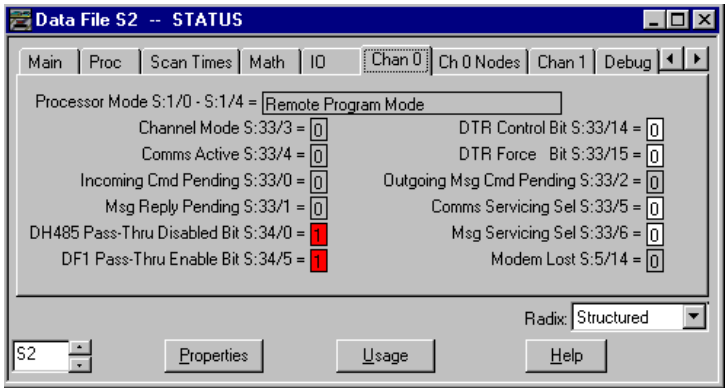
Chan. 0 - System driver is set for DF1 Full Duplex.

Also refer to KB DOC #: 14001 for the step by step procedure for this example.

Passthru Examples DH+ to DF1

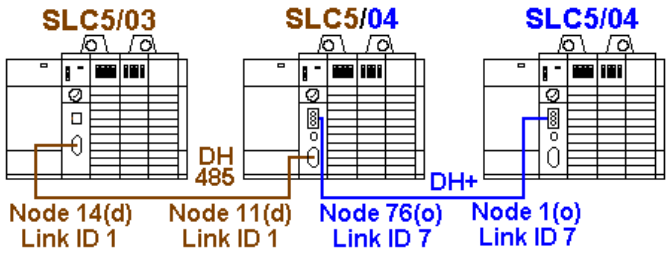


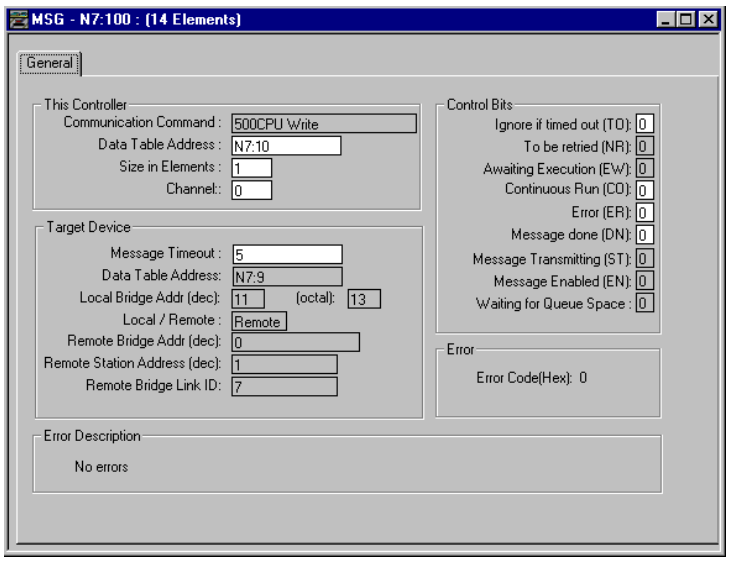


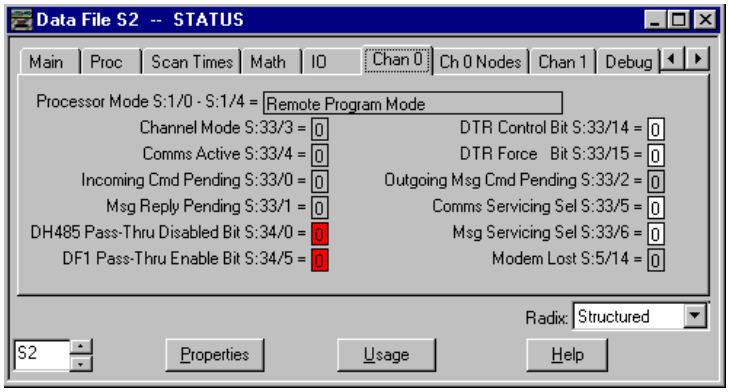


For the Step by Step procedure for this example please refer to KB DOC #: 14002

Passthru Examples DH485 to DH+

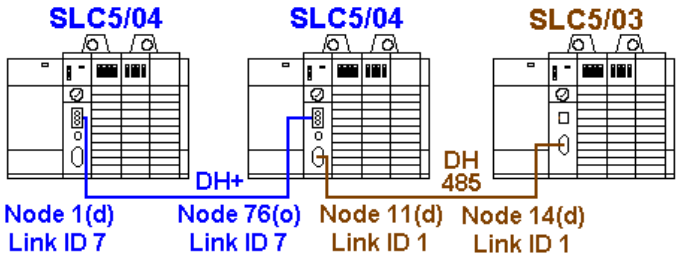


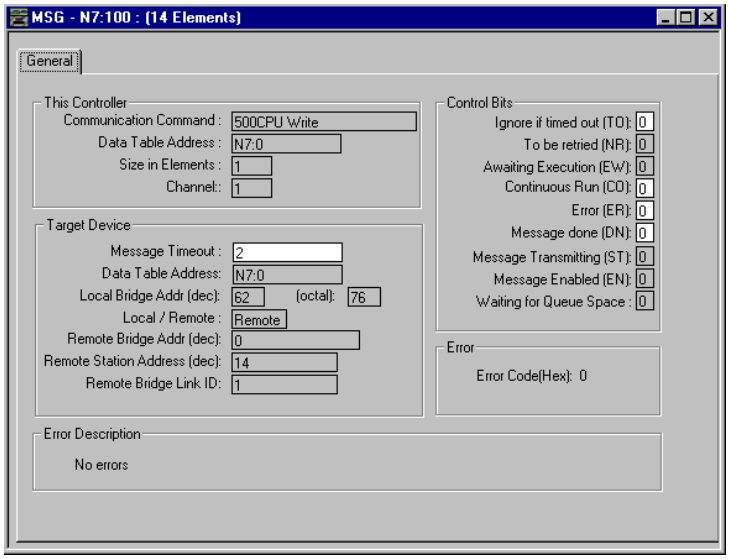


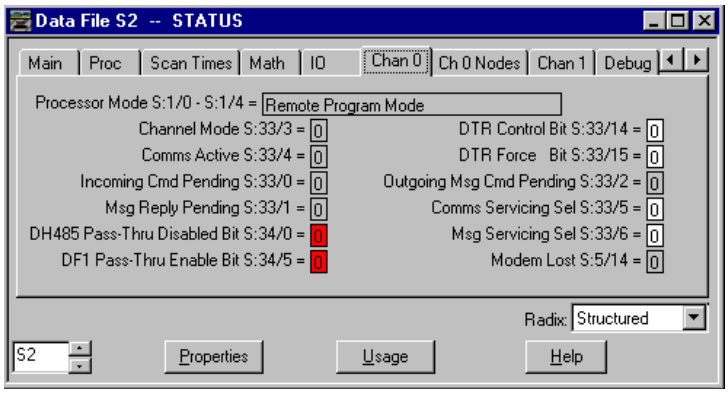


For the Step by Step procedure for this example please refer to KB DOC #: 14003

Passthru Examples DH+ to DH485







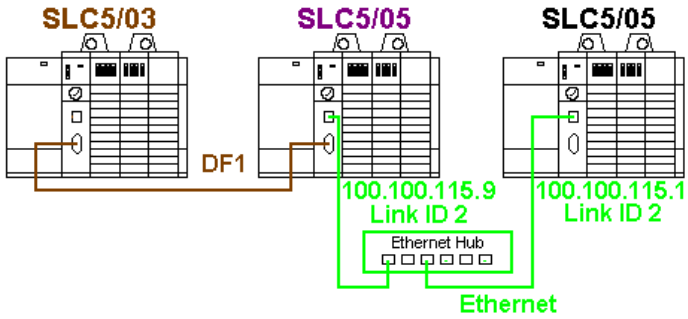
For the Step by Step procedure for this example please refer to KB DOC #: 14004

SLC 5/05 Passthru Examples

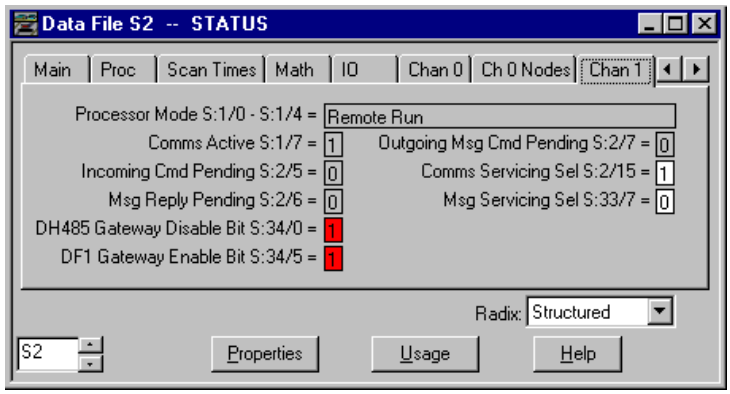
Passthru Examples DF1 to Ethernet

The IP Addresses used in the following illustration are for example purposes only. Contact your system administrator for IP addresses unique to your network.

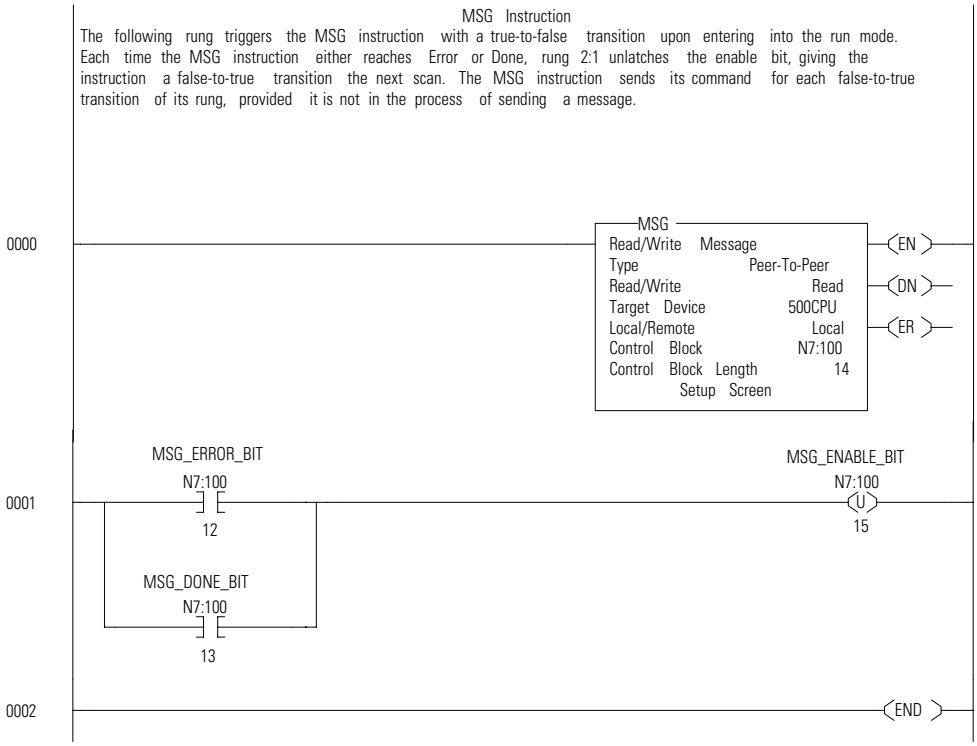
In the following diagram, a SLC 5/03 will send a local message via DF1 to the SLC 5/05 (IP Address 100.100.115.9). The SLC 5/05 acts as a bridge, sending the message out via Ethernet to the SLC 5/05 (IP Address 100.100.115.1), whose address is stored in the routing table.



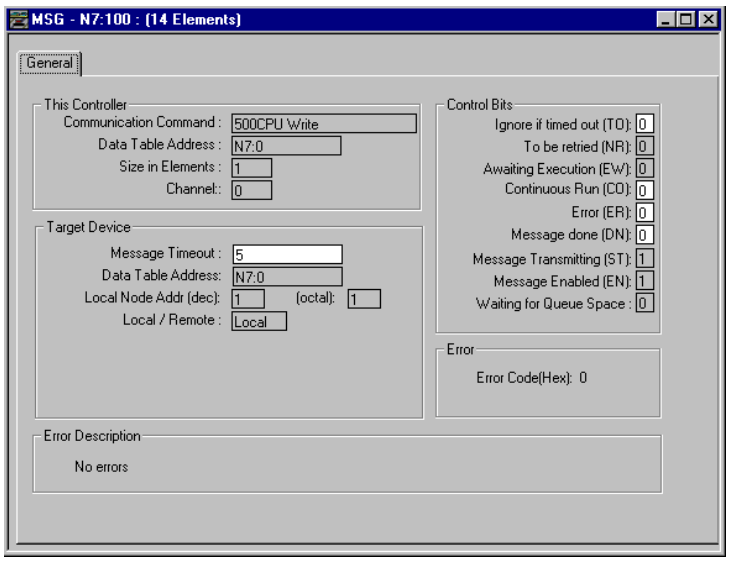
IMPORTANT In the SLC 5/05 (IP Address 100.100.115.9) bridge, Status File Bit S:34/5 must be set to 1 to enable DF1-to-Ethernet passthru.



The following is the logic necessary for the SLC5/03 processor.



The following is the MSG Setup for the SLC5/03 processor.



- Channel is set to zero for DF1 full-duplex protocol.
- Target Node is the station address in the SLC 5/05 (IP Address 100.100.115.9) routing table where the IP address for SLC 5/05 (IP Address 100.100.115.1) is stored.
- Channel 0 Driver is set to DF1 Full Duplex.
- Source ID is the address of the sender of the message. It can be any number from 0 to 254.

SLC 5/05 (IP Address 100.100.115.9) Bridge

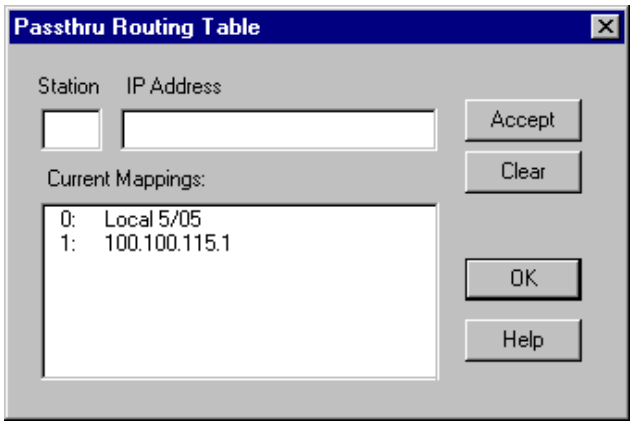
Ladder logic is not required for the SLC 5/05 which acts as the bridge from DF1-to-Ethernet. However, you must set up a passthru routing table when configuring the bridge. The channel configuration is shown below, followed by the routing table.

- Passthru Link IDs are used by other processors to send remote MSG packets through the SLC 5/05 (IP Address 100.100.115.9) when channel-to-channel passthru is used. Passthru Link IDs must be properly specified in the remote MSG instructions to enable channel-to-channel passthru. The default Passthru Link ID for Channel 0 is one. The default Passthru Link ID for Channel 1 is two.
- The Passthru Routing Table File is the integer file used by the processor to store routing table IP addresses and link them to unique node addresses.

IMPORTANT Channel 0 Source ID must be set to 0 when SLC 5/05 (IP Address 100.100.115.9) is used as the bridge between DF1 full-duplex and Ethernet.

Passthru Routing Table

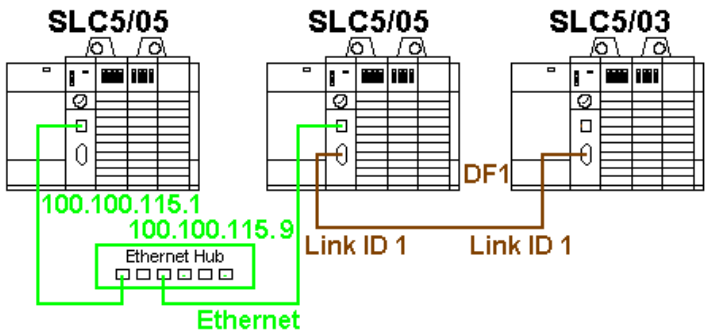
The passthru routing table is located under the channel configuration selection in RSLogix 500 Programming Software. If a Passthru Routing Table File number was entered in the General Tab in the Channel Configuration dialog box, click on the + in front of “Channel Configuration” to reveal the routing table selection.

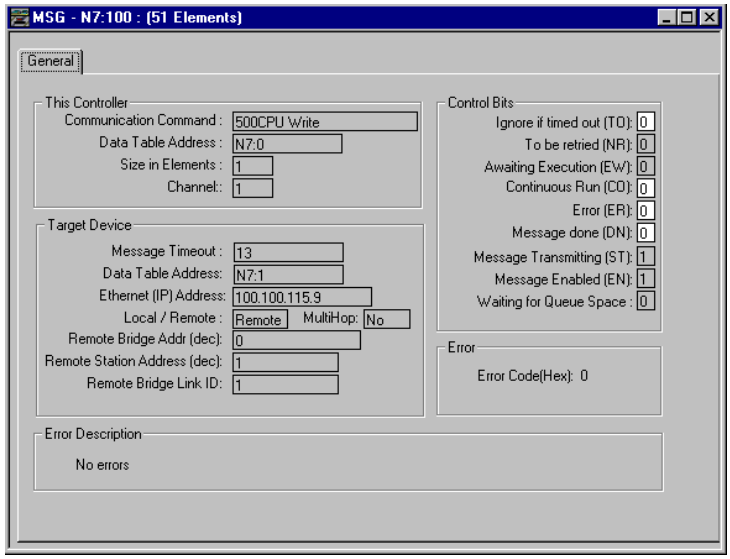


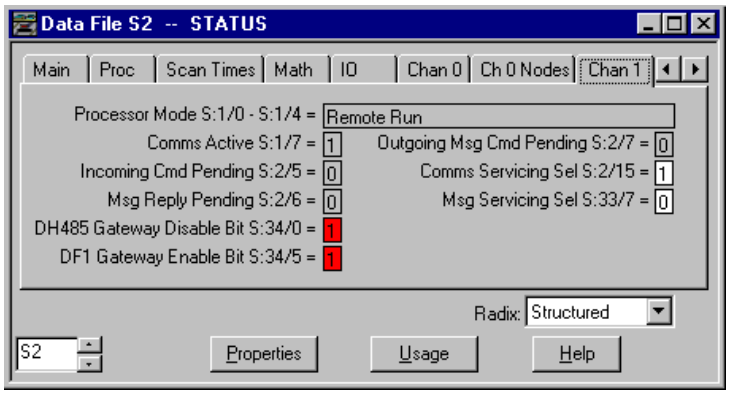
IMPORTANT The routing table must contain the IP address of SLC 5/05 (IP Address 100.100.115.1) at station target node one, as shown in the routing table above. Target node one was identified as the target node in the SLC 5/03 Message Setup dialog box.

For the Step by Step procedure for this example please refer to KB DOC #: 14005

Passthru Examples Ethernet to DF1

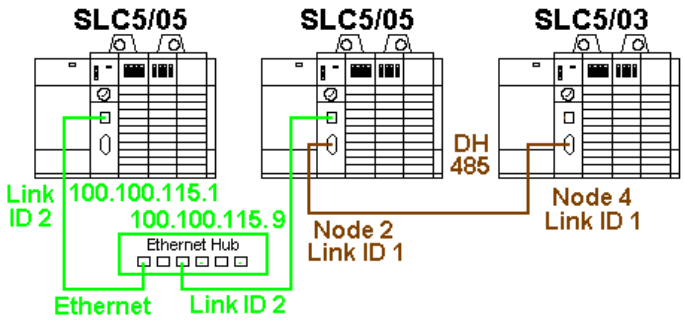


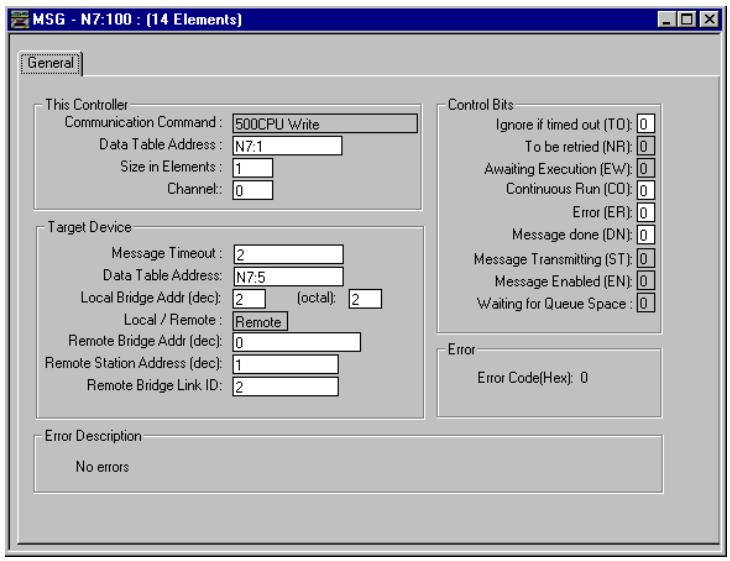


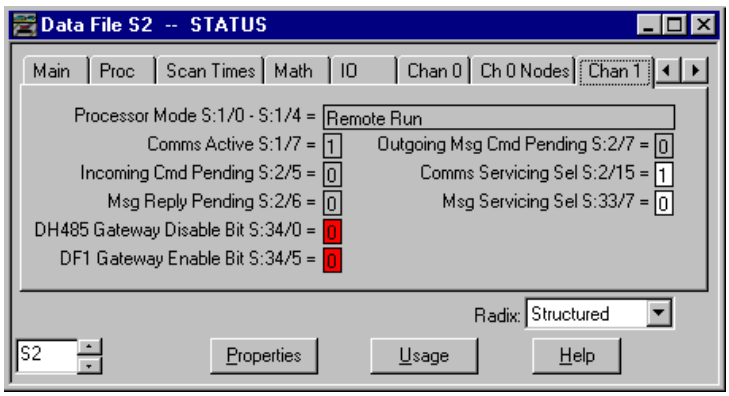


For the Step by Step procedure for this example please refer to KB DOC #: 14006

Passthru Examples DH485 to Ethernet

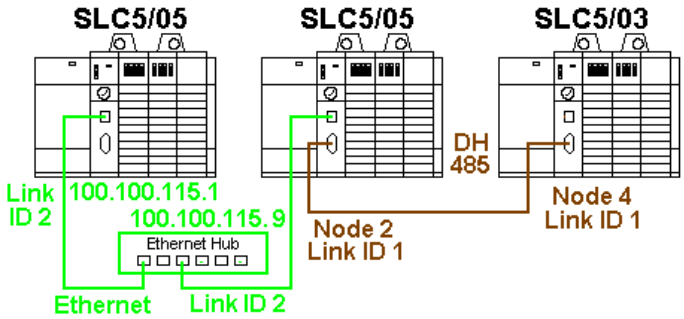


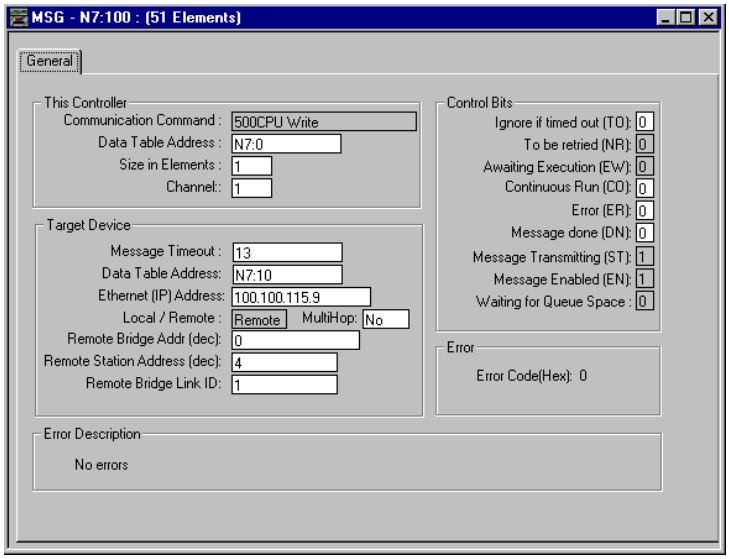


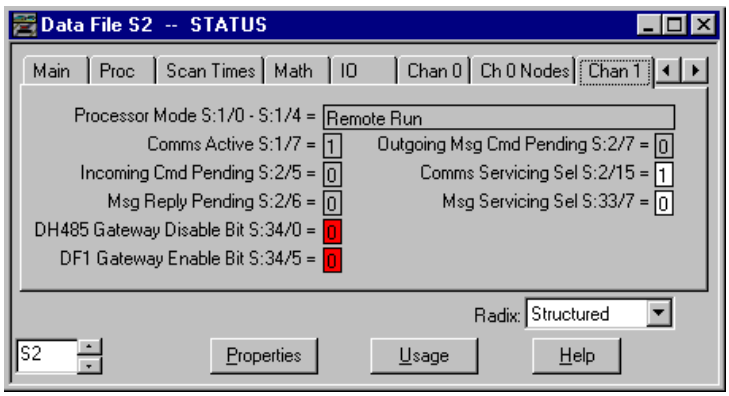


For the Step by Step procedure for this example please refer to KB DOC #: 14007

Passthru Examples Ethernet to DH485





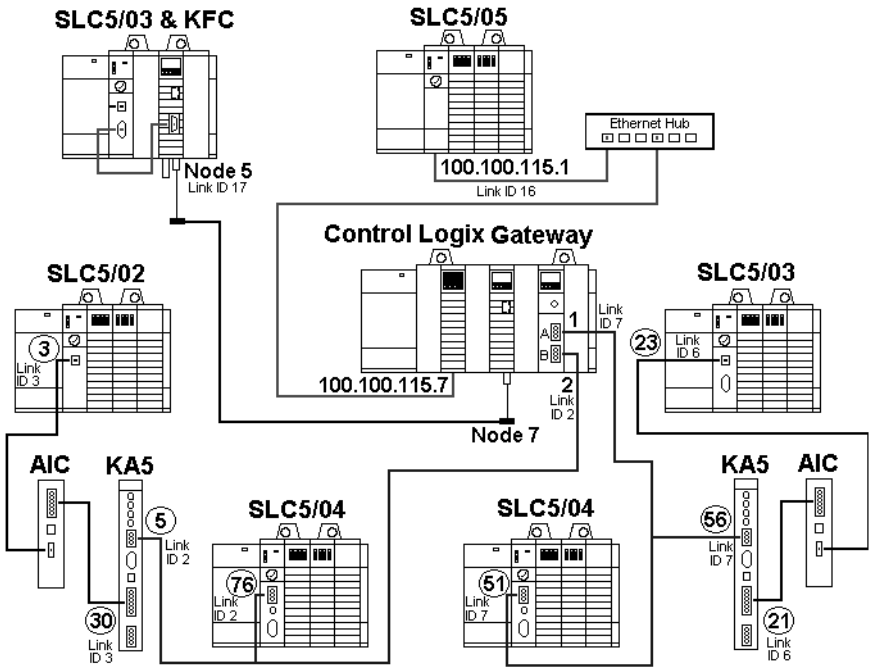


For the Step by Step procedure for this example please refer to KB DOC #: 14008

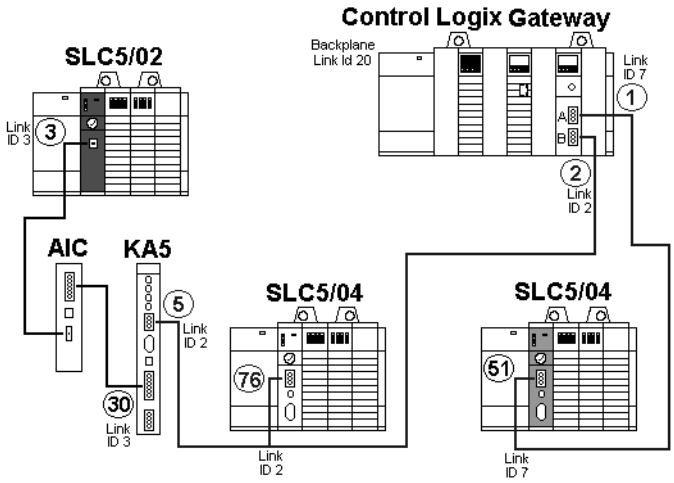
Remote Examples

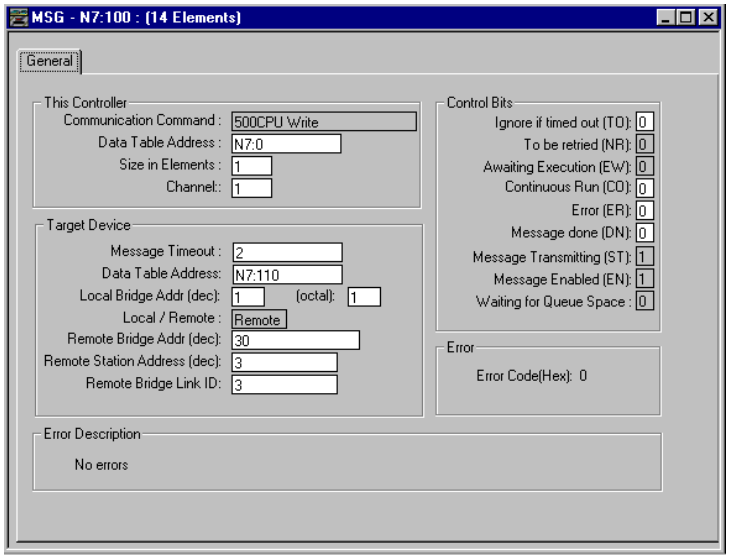
All of the following remote examples were constructed for the following network.

Remote Examples Network Overview



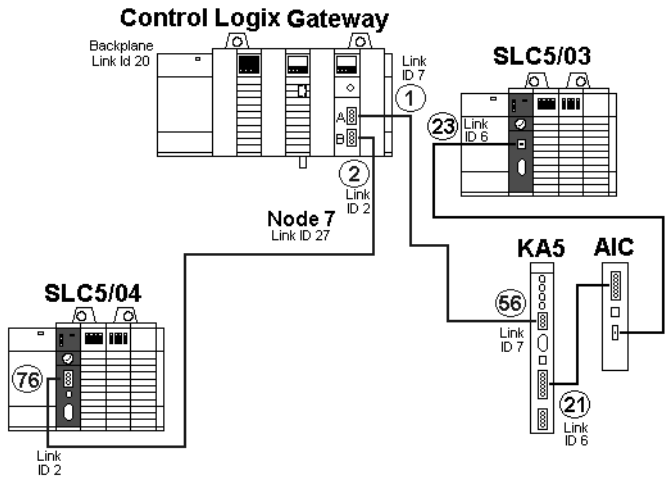
Network Message Example #1: SLC5/04 to SLC5/02 via DHRIO and KA5

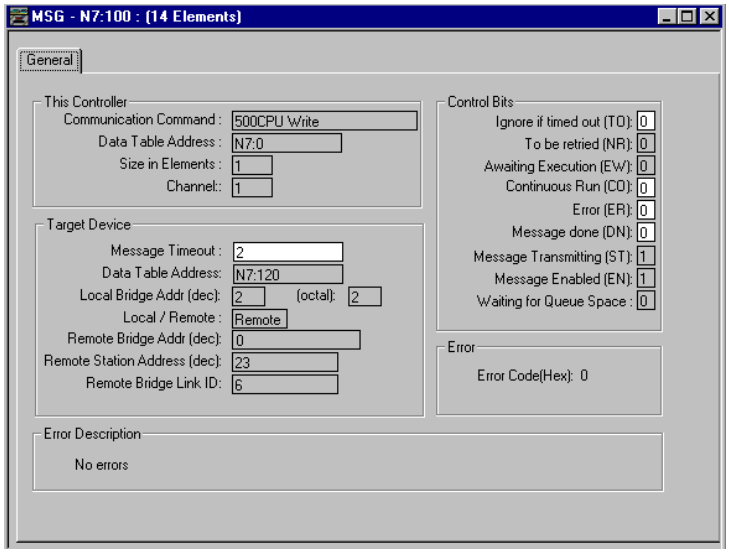




For the Step by Step procedure for this example please refer to KB DOC #: 14009

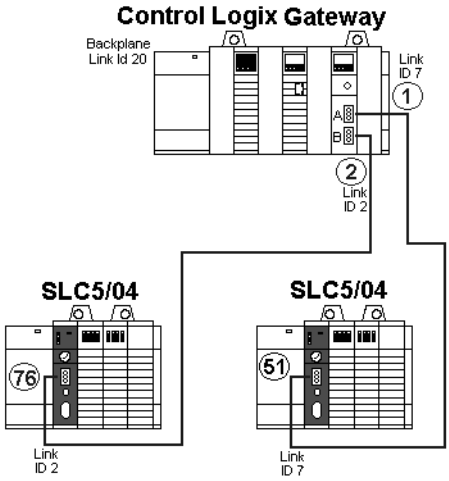
Network Message Example #2: SLC 5/04 to SLC5/03 via DHRIO and KA5

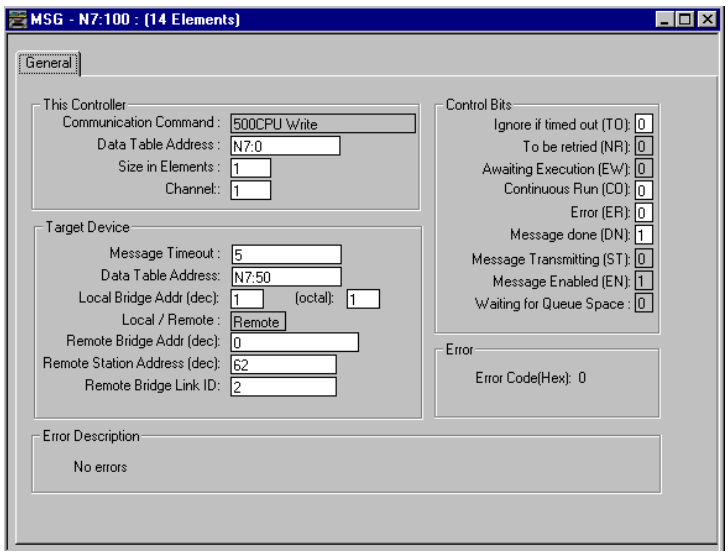




For the Step by Step procedure for this example please refer to KB DOC #: 14010

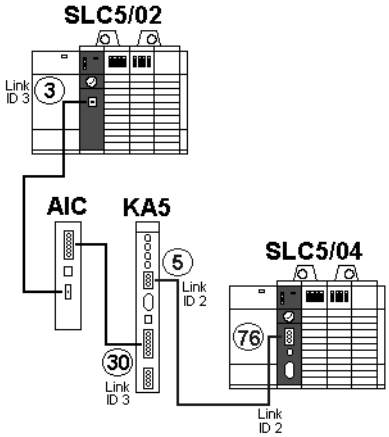
Network Message Example #3: SLC5/04 to SLC5/04 via KA5

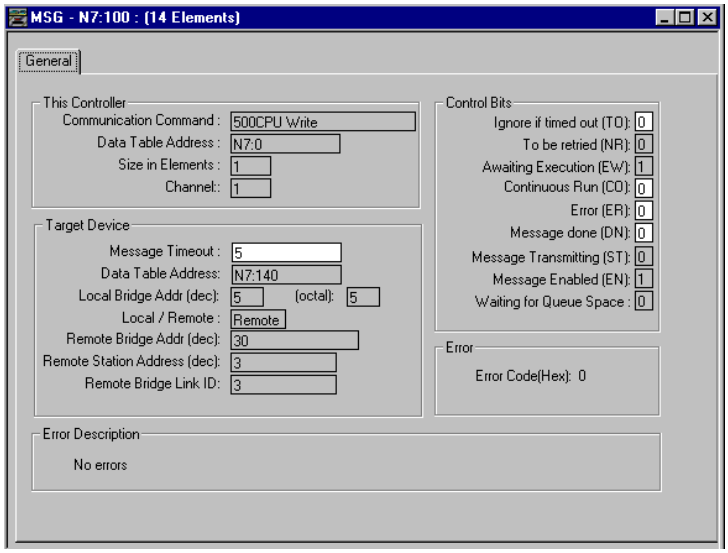




For the Step by Step procedure for this example please refer to KB DOC #: 14011

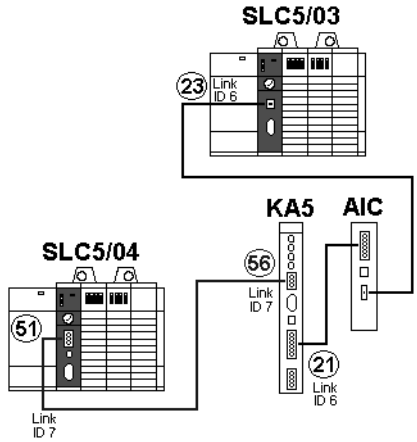
Network Message Example #4: SLC 5/04 to SLC5/02 via KA5

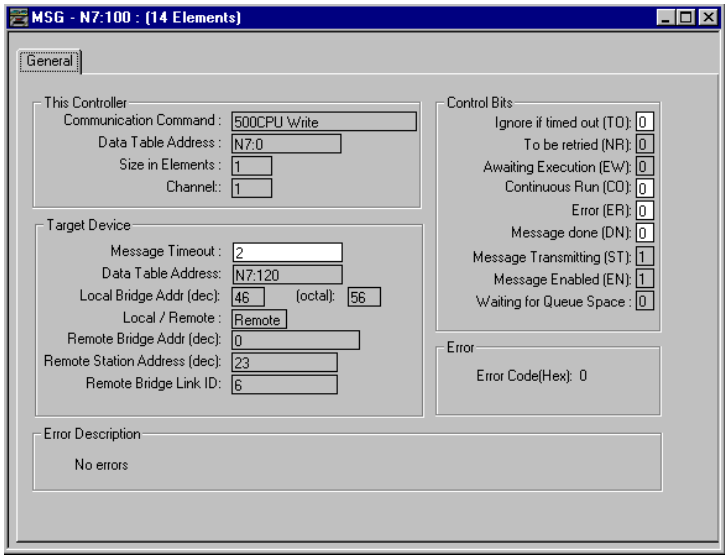




For the Step by Step procedure for this example please refer to KB DOC #: 14012

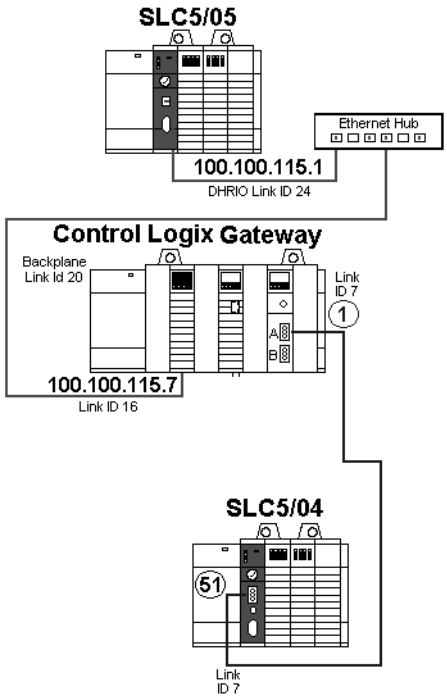
Network Message Example #5: SLC5/04 to SLC5/03 via KA5

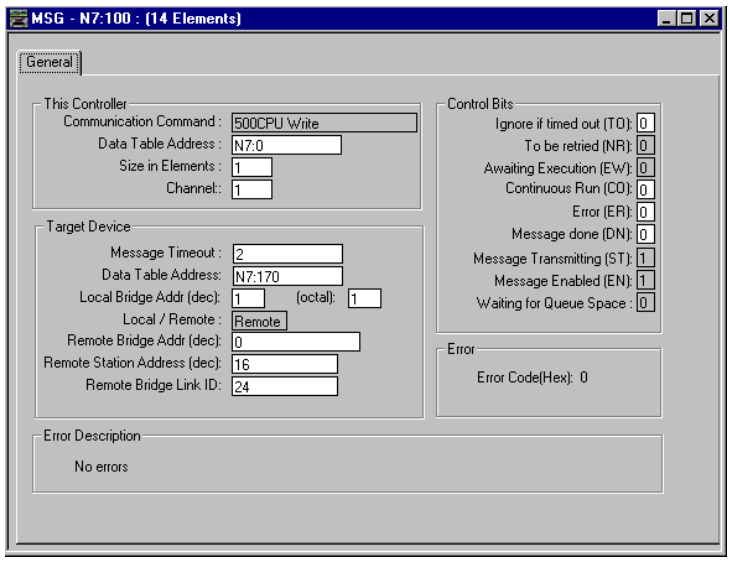




For the Step by Step procedure for this example please refer to KB DOC #: 14013

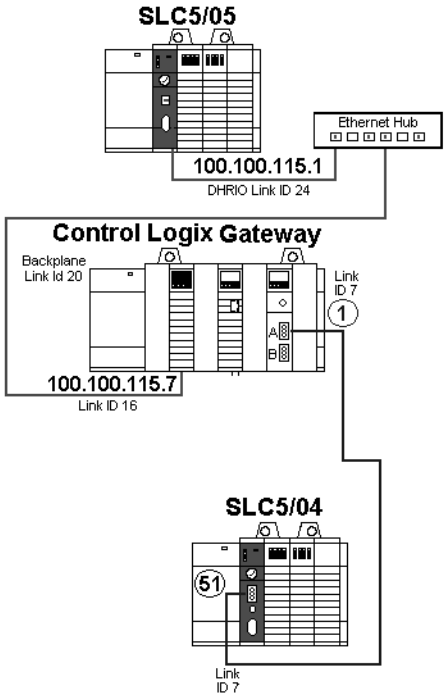
Network Message Example #6: SLC5/04 to SLC5/05 via DHRIO and ENET

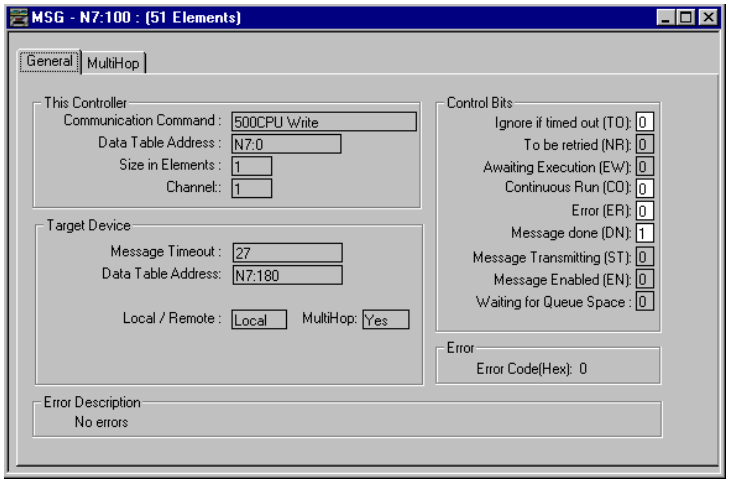


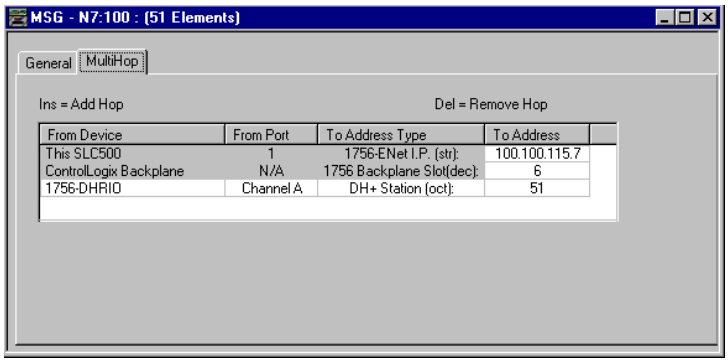


For the Step by Step procedure for this example please refer to KB DOC #: 14014

Network Message Example #7: 5/04 to 5/05 via ENET and DHRIO

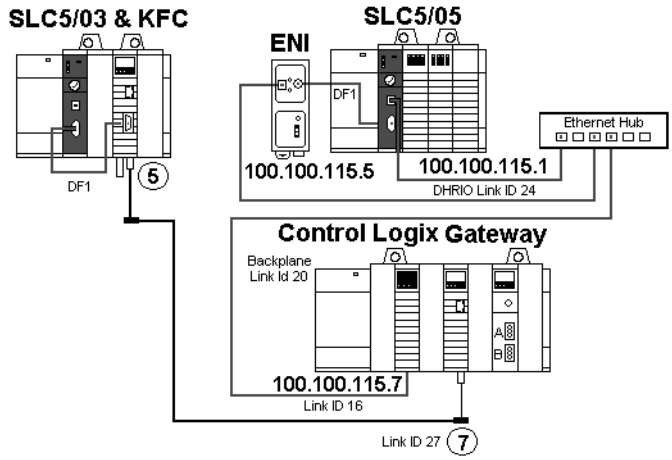


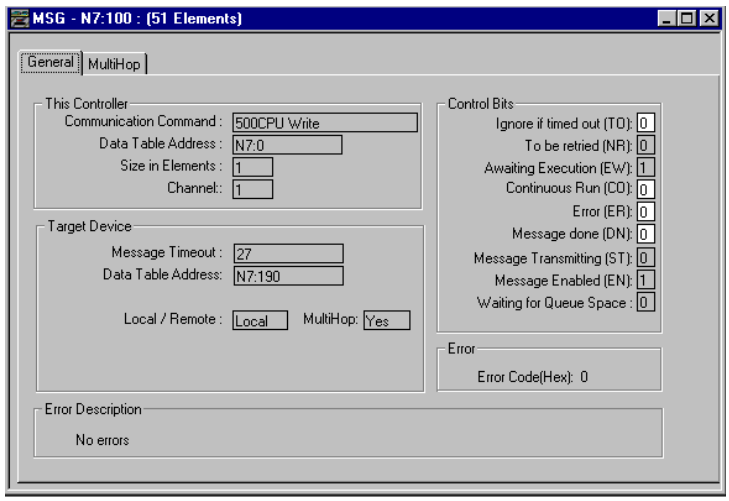


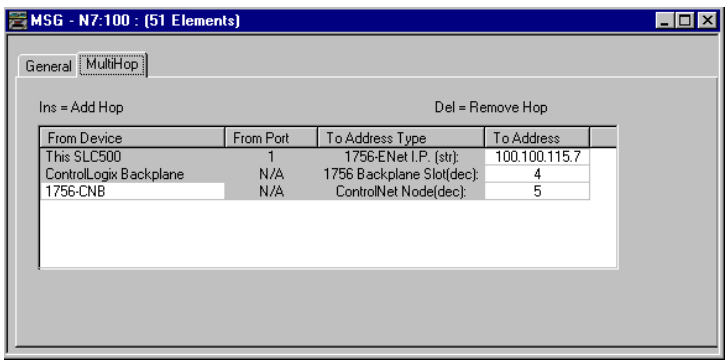


For the Step by Step procedure for this example please refer to KB DOC #: 14015

Network Message Example #8: SLC5/05 to SLC5/03 via ENET, CNB and KFC

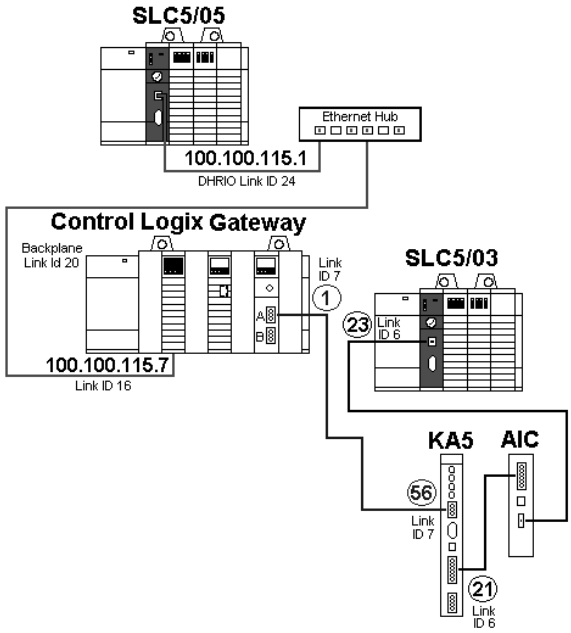


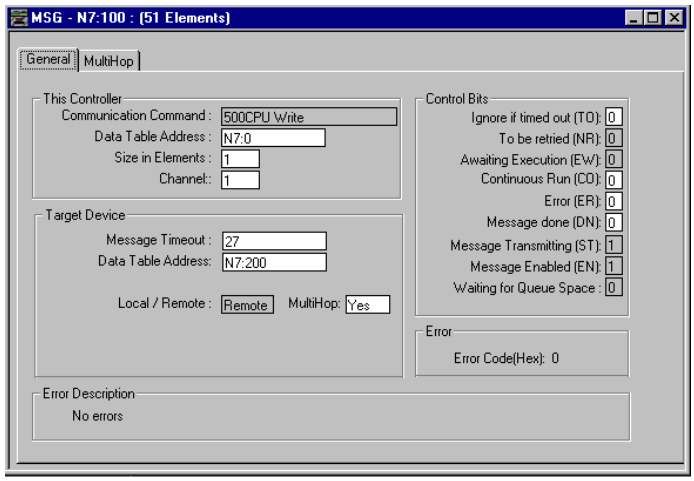


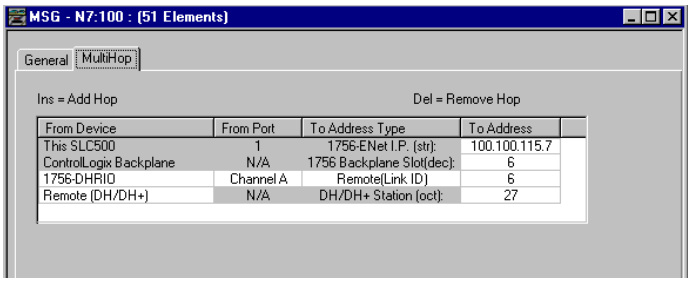


For the Step by Step procedure for this example please refer to KB DOC #: 14016

Network Message Example #9: SLC5/05 to SLC5/03 via ENET, DHRIO and KA5

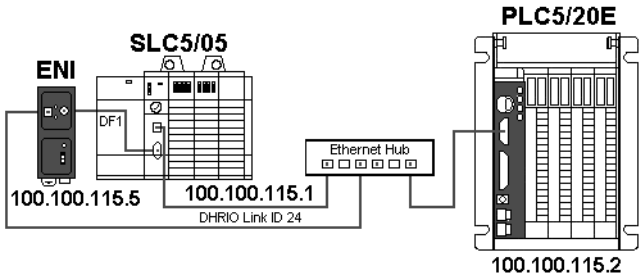


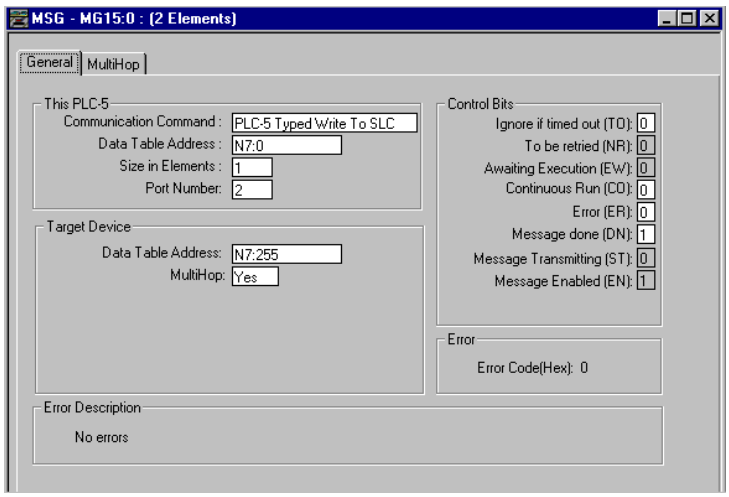


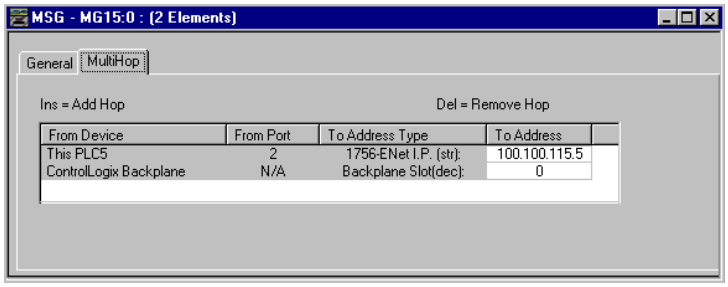


For the Step by Step procedure for this example please refer to KB DOC #: 14017

Network Message Example #10: PLC5/20E to SLC500 CH0 via ENI







For the Step by Step procedure for this example please refer to KB DOC #: 14018.

Troubleshooting Faults

This chapter lists the major error fault codes, indicates the probable causes of faults, and recommends corrective action. This chapter also explains the operating system download faults for the SLC 5/03 (and higher) processors.

Automatically Clearing Faults

The following section describes the different ways to automatically clear a fault using your programming software.

SLC Processors

- Set the Fault Override at Powerup Bit S:1/8 in the status file to clear the fault when power is cycled, assuming the user program is not corrupt.
- Designate a User Fault Routine Number in S:29 and program that ladder subroutine file to attempt to correct the fault and clear S:1/13.
- Set the Startup Protection Fault Bit, S:1/9 which executes the user fault routine prior to the execution of the first scan of the ladder program when a fault condition exists.
- Set one of the autoload bits S:1/10, S:1/11, or S:1/12 in the status file of the program in an EEPROM to automatically transfer a new non-faulted program from the memory module to RAM when power is cycled.

Refer to Appendix B in this manual for more information on status bits S:1/8 to 13 and S:5/0 to 7.

TIP You can declare your own application-specific major fault by writing your own unique value to S:6 and then setting bit S:1/13.

Manually Clearing Faults

The following section describes the different ways to manually clear a fault when using an SLC processor.

- Manually clear the major fault bit S:1/13, and the minor and major error bits S:5/0-7 in the status file, using a programming device or a Data Table Access Module. Place the processor in the REM Program mode. Correct the condition causing the fault, then return the processor to either REM Run or any of the REM Test modes.
- *SLC 5/03 and higher processors*: Toggle the keyswitch from RUN to PROGram and then back to RUN.

ATTENTION

SLC 5/03 and higher processors: Clearing these bits with the keyswitch in the RUN position causes the processor to immediately enter the Run mode.

ATTENTION

If you are online with an SLC 5/03 (or higher) processor with the keyswitch position in RUN and you press the clear major fault function key, you are warned that the processor will enter the Run mode once you clear the fault.

Using the Fault Routine

When designating a subroutine file, the occurrence of recoverable or non-recoverable user faults causes the designated subroutine to be executed for one scan. If the fault is recoverable, the subroutine can be used to correct the problem and clear the fault bit S:1/13. The processor then continues in the RUN mode. If the fault is non-recoverable, the subroutine can send a message via a Message instruction to another node with error code information and/or can do an orderly shutdown of the process.

The subroutine does not execute for non-user faults.

SLC Processor Faults

The processor faults are divided into the following types:

- powerup errors
- going-to-run errors
- run errors
- user program instruction errors

Powerup Errors

Table 15.1 Powerup Errors

Error Code (Hex)	Description	Probable Cause	Recommended Action
0001	NVRAM error.	• Either noise, • lightning, • improper grounding, • lack of surge suppression on outputs with inductive loads, or • poor power source. • Loss of battery or capacitor backup.	Correct the problem, reload the program, and run. You can use the autoload feature with a memory module to automatically reload the program and enter the Run mode.
0002	Unexpected hardware watchdog timeout.	• Either noise, • lightning, • improper grounding, • lack of surge suppression on outputs with inductive loads, or • poor power source.	Correct the problem, reload the program, and run. You can use the autoload feature with a memory module to automatically reload the program and enter the Run mode.
0003	Memory module memory error. This error can also occur when going to the REM Run mode.	Memory module is corrupted.	Re-program the memory module. If the error persists, replace the memory module.

Table 15.1 Powerup Errors

Error Code (Hex)	Description	Probable Cause	Recommended Action
0007	Failure during memory module transfer.	Memory module is corrupted.	Re-program the memory module. If the error persists, replace the memory module.
0008	Internal software error.	An unexpected software error occurred due to: • Either noise, • lightning, • improper grounding, • lack of surge suppression on output with inductive loads, or • poor power source.	Correct the problem, reload the program, and run. You can use the autoload feature with a memory module to automatically reload the program and enter the Run mode. If the problem re-occurs, contact your RSI representative.
0009	Internal hardware error.	An unexpected hardware error occurred due to: • Either noise, • lightning • improper grounding, • lack of surge suppression on output with inductive loads, or • poor power source.	Correct the problem, reload the program, and run. You can use the autoload feature with a memory module to automatically reload the program and enter the Run mode. If the problem re-occurs, contact your A-B representative.

Going-to-Run Errors

Table 15.2 Going-to-Run Errors

Error Code (Hex)	Description	Probable Cause	Recommended Action
0010	The processor does not meet the required revision level.	The revision level of the processor is not compatible with the revision level for which the program was developed.	Consult your local A-B representative to purchase an upgrade kit for your processor.
0011	The executable program file number 2 is absent.	Incompatible or corrupt program is present.	Reload the program or reprogram with RSI programming software.
0012	The ladder program has a memory error.	• Either noise, • lightning, • improper grounding, • lack of surge suppression on outputs with inductive loads, or • poor power source.	Correct the problem, reload the program, and run. If the error persists, be sure to use current RSI programming software to develop and load the program.

Table 15.2 Going-to-Run Errors

Error Code (Hex)	Description	Probable Cause	Recommended Action
0013	• The required memory module is absent, or • S:1/10 or S:1/11 is not set as required by the program.	• Either one of the status bits is set in the program but the required memory module is absent, or • status bit S:1/10 or S:1/11 is not set in the program stored in the memory module, but it is set in the program in the processor memory.	• Either install a memory module in the processor, or • upload the program from the processor to the memory module.
0014	Internal file error.	• Either noise, • lightning, • improper grounding, • lack of surge suppression on outputs with inductive loads, or • poor power source.	Correct the problem, reload the program, and run. If the error persists, be sure to use current RSI programming software to develop and load the program.
0015	Configuration file error.	• Either noise, • lightning, • improper grounding, • lack of surge suppression on outputs with inductive loads, or • poor power source.	Correct the problem, reload the program, and run. If the error persists, be sure to use current RSI programming software to develop and load the program.
0016	Startup protection after power loss. Error condition exists at powerup when bit S:1/9 is set and powerdown occurred while running.	Status bit S:1/9 has been set by the user program. Refer to Appendix B for details on the operation of status bit S:1/9.	• Either reset bit S:1/9 if this is consistent with the application requirements, and change the mode back to run, or • clear S:1/13, the major fault bit, before the end of the first program scan is reached.
0017	NVRAM/memory module user program mismatch.	Bit S:2/9 is set and the memory module user program does not match the NVRAM user program.	Transfer the memory module program to NVRAM then change to Run mode.
0018	Incompatible user program. Operating system type mismatch. This error can also occur during powerup.	The user program is too advanced to be executed in the current operating system.	Contact your local Allen-Bradley representative to purchase an upgrade kit for your processor.
0019	A duplicate label number was detected.	A duplicate or missing label instruction was found in a subroutine.	• Either remove the duplicate label, or • add a label.

Run Errors

Table 15.3 Run Errors

Error Code (Hex)	Description	Probable Cause	Recommended Action
001F	A program integrity problem occurred during an online editing session.	Either noise, communication loss, or a power cycle occurred during an online edit session.	Reload the program and re-enter your changes.
0004	Memory error occurred while in the Run mode.	• Either noise, • lightning, • improper grounding, • lack of surge suppression on outputs with inductive loads, or • poor power source.	Correct the problem, reload the program, and run. You can use the autoload feature with a memory module to automatically reload the program and enter the Run mode.
0020	A minor error bit is set at the end of the scan. Refer to S:5 minor error bits (lower byte only).	• Either a math or FRD instruction overflow has occurred, • sequencer or shift register instruction error was detected, • a major error was detected while executing a user fault routine, or • M0-M1 file addresses were referenced in the user program for a disabled slot.	Correct the programming problem, reload the program and enter the run mode. See also minor error bits S:5 in Appendix B.
0021	A remote power failure of an expansion I/O chassis has occurred. Note: A modular system that encounters an over-voltage or over-current condition in any of its power supplies can produce any of the I/O error codes listed on pages page 15-11 through page 15-13 (instead of code 002). The over-voltage or over-current condition is indicated by the power supply LED being off.	Fixed in FRN 1 to 4 SLC 5/01 processors: Power was removed or the power dipped below specification for an expansion chassis. SLC 5/02 processors and FRN 5 SLC 5/01 processors: This error code is present only while power is not applied to an expansion chassis. This is the only self-clearing error code. When power is re-applied to the expansion chassis, the fault will be cleared.	Fixed and FRN 1 to 4 SLC 5/01 processors: Cycle power on the local chassis. SLC 5/02 processors and FRN 5 SLC 5/01 processors: Re-apply power to the expansion chassis.

ATTENTION

Fixed and FRN 1 through 4 SLC 5/01 processors: If the remote power failure occurred while the processor was in the REM Run mode, error 0021 will cause the major error halted bit (S:1/13) to be cleared at the next powerup of the local chassis.
SLC 5/02 processor and FRN 5 SLC 5/01 processors: Power to the local chassis does not need to be cycled to resume the REM Run mode. Once the remote chassis is re-powered, the CPU will restart the system.

Table 15.3 Run Errors

Error Code (Hex)	Description	Probable Cause	Recommended Action
0022	The user watchdog scan time has been exceeded.	• Either Watchdog time is set too low for the user program, or • user program caught in a loop.	• Either increase the watchdog timeout in the status file (S:3H), or • correct the user program problem.
0023	Invalid or non-existent STI interrupt file.	• Either an STI interrupt file number was assigned in the status file, but the subroutine file was not created, or • the STI interrupt file number assigned was 0, 1, or 2.	• Either disable the STI interrupt setpoint (S:30) and file number (S:31) in the status file, or • create an STI interrupt subroutine file for the file number assigned in the status file (S:31). The file number must not be 0, 1, or 2.
0024	Invalid STI interrupt interval (greater than 2550 ms or negative).	The STI setpoint is out of range (greater than 2550 ms or negative).	• Either disable the STI interrupt setpoint (S:30) and file number (S:31) in the status file, or • create an STI interrupt routine for the file number referenced in the status file (S:31). The file number must not be 0, 1, or 2.
0025	Excessive stack depth/JSR calls for the STI routine.	A JSR instruction is calling for a file number assigned to an STI routine.	Correct the user program to meet the requirements and restrictions for the JSR instruction, then reload the program and run.
0026	Excessive stack depth/JSR calls for an I/O interrupt routine.	A JSR instruction is calling for a file number assigned to an I/O interrupt routine.	Correct the user program to meet the requirements and restrictions for the JSR instruction, then reload the program and run.
0027	Excessive stack depth/JSR calls for the user fault routine.	A JSR instruction is calling for a file number assigned to the user fault routine.	Correct the user program to meet the requirements and restrictions for the JSR instruction, then reload the program and run.
0028	Invalid or non-existent “startup protection” fault routine file value.	• Either a fault routine file number was created in the status file, but the fault routine file was not physically created, or • the file number created was 0, 1, or 2.	• Either disable the fault routine file number (S:29) in the status file, or • create a fault routine for the file number referenced in the status file (S:29). The file number must not be 0, 1, or 2.
0029	Indexed address reference is outside of the entire data file space (range of B3:0 through the last file).	The program is referencing through indexed addressing an element beyond the allowed range. The range is from B3:0 to the last element of the last data file created by the user.	Correct and reload the user program. This problem cannot be corrected by writing to the index register word (S:24).

Table 15.3 Run Errors

Error Code (Hex)	Description	Probable Cause	Recommended Action
ATTENTION 	The SLC processor uses an index value of zero for the faulted instruction following error recovery.		
002A	Indexed address reference is beyond the specific referenced data file.	The program is referencing through indexed addressing an element beyond a file boundary.	Correct the user program, allocate more data space using the memory map, or re-save the program allowing crossing of file boundaries. Reload the user program. This problem cannot be corrected by writing to the index register word (S:24).
002B	An invalid file number for an indirect address exists.	The file number exists, but it is not the correct file type or the file number does not exist.	Check the file type or create the file number.
002C	The referenced indirect address element is outside data file limits.	The indirectly referenced element does not exist, but the file type is correct and it exists.	Create the indirectly referenced element.
002D	An invalid referenced indirect address subelement exists.	Either a subelement is referenced incorrectly or an indirect reference has been made to an M-file.	Correct the references and try again.
002E	Invalid DII Input slot.	The referenced slot is empty or a non-discrete I/O card is present.	Change the input slot to a discrete I/O card.
002F	Invalid or non-existent DII interrupt file.	• Either an DII interrupt file number was assigned in the status file, but the subroutine file was not created, or • the DII interrupt file number assigned was 0, 1, or 2.	Either disable the DII function by writing a zero to this location, or change the value to a valid ladder file (3-255).

User Program Instruction Errors

Table 15.4 User Program Instruction Errors

Error Code (Hex)	Description	Probable Cause	Recommended Action
0030	An attempt was made to jump to one too many nested subroutine files. This code can also mean that a program has potential recursive routines.	• Either more than the maximum of 4 (8 if you are using a 5/02 or 5/03 processor) levels of nested subroutines are called for in the user program, or • nested subroutine(s) are calling for subroutine(s) of a previous level.	Correct the user program to meet the requirements and restrictions for the JSR instruction, then reload the program and run.
0031	An unsupported instruction reference was detected.	The type or series level of the processor does not support an instruction residing in the user program, or you have programmed a constant as the first operand of a compare instruction.	• Either replace the processor with one that supports the user program, or • modify the user program so that all instructions are supported by the processor, then reload the program and run.
0032	A sequencer instruction length/position parameter points past the end of a data file.	The program is referencing an element beyond a file boundary set up by the sequencer instruction.	Correct the user program or allocate more data file space using the memory map, then reload and run.
0033	The length parameter of an LFU, LFL, FFU, FFL, BSL, or BSR instruction points past the end of a data file.	The program is referencing an element beyond a file boundary set up by the instruction.	Correct the user program or allocate more data file space using the memory map, then reload and run.
0034	A negative value for a timer accumulator or preset value was detected. Fixed processors with 24 VDC input only: A negative or zero HSC preset was detected in a HSC instruction.	The accumulated or preset value of a timer in the user program was detected as being negative.	If the user program is moving values to the accumulated or preset word of a timer, make certain these values cannot be negative. Correct the user program, reload, and run.
0034 (related to fixed 5/01 HSC instruction)	A negative or zero HSC preset was detected in an HSC instruction.	The preset value for the HSC instruction is out of the valid range. Valid range is 1 to 32767.	If the user program is moving values to the preset word of the HSC instruction, make certain the values are within the valid range. Correct the user program, reload, and run.
0035	TND, SVC, or REF instruction is called within an interrupting or user fault routine.	A TND, SVC, or REF instruction is being used in an interrupt or user-fault routine. This is illegal.	Correct the user program, reload, and run.
0036	An invalid value is being used for a PID instruction parameter.	An invalid value was loaded into a PID instruction by the user program or by the user via the data monitor function for this instruction.	Code 0036 is discussed on page 9-18.
0038	A RET instruction was detected in a non-subroutine file.	A RET instruction resides in the main program.	Correct the user program, reload, and run.

Table 15.4 User Program Instruction Errors

Error Code (Hex)	Description	Probable Cause	Recommended Action
xx39	An invalid string length was detected in a string file. (xx = data file number)	The first word of string data contains a negative, zero, or value greater than 82.	Check the first word of the string data elements for invalid values and correct the user data.
003A	An attempt to write to a protected data file occurred.	An attempt was made to write to an indirect address located in a file that has constant data file protection.	Remove the protection and retry the function.
003B	Motherboard and Daughter Card firmware do not match.	Motherboard and Daughter Card were not flash upgraded as a pair.	Flash upgrade both Motherboard and Daughter Card to latest version.
003C	STI Watchdog timer time-out	STI setpoint set too low. The processor was not able to service the STI interrupt before its watchdog timer timeout. See page B-29.	Increase the value of the STI setpoint (S:30).
005F	Invalid Rack ID	Invalid rack setup or problem with rack.	For a multi-rack system, make sure no more than three racks are configured. For a single-rack system, replace the rack, the rack may be bad.

I/O Errors

ERROR CODES: The characters xx in the following codes represent the slot number, in hexadecimal. If the exact slot cannot be determined, the characters xx become 03 for fixed controllers and 1F for modular controllers. Refer to the table below.

Table 15.5 Slot Numbers

Slot	xx (1)	Slot	xx	Slot	xx	Slot	xx
0	00	8	08	16	10	24	18
1	01	9	09	17	11	25	19
2	02	10	0A	18	12	26	1A
**3	03	11	0B	19	13	27	1B
4	04	12	0C	20	14	28	1C
5	05	13	0D	21	15	29	1D
6	06	14	0E	22	16	30	1E
7	07	15	0F	23	17	*	1F

(1) Slot numbers (xx) in hexadecimal.

RECOVERABLE I/O FAULTS (SLC 5/02 and higher processors only): Many I/O faults are recoverable. To recover, you must disable the specified slot, xx, in the user fault routine. If you do not disable slot xx, the processor will fault at the end of the scan.

TIP An I/O card that is severly damaged may cause the processor to indicate that an error exists in slot 1.

Table 15.6 I/O Errors

Error Code (Hex)	Description	Probable Cause	Recommended Action
xx50	A chassis data error is detected.	• Either noise, • lightning, • improper grounding, • lack of surge suppression on outputs with inductive loads, or • poor power source.	Correct the problem, clear the fault, and re-enter Run mode.
xx51	A "stuck" runtime error is detected on an I/O module.	If this is a discrete I/O module, this is a noise problem. If this is a specialty I/O module, refer to the applicable user manual for the probable cause.	Cycle power to the system. If this does not correct the problem, replace the module.
xx52	A module required for the user program is detected as missing or removed.	An I/O module configured for a particular slot is missing or has been removed.	• Either disable the slot in the status file (S:11 and S:12), or • insert the required module in the slot.
xx53	When going-to-run, a user program declares a slot as unused, and that slot is detected as having an I/O module inserted. This code can also mean that an I/O module has reset itself.	• Either the I/O slot is not configured for a module, but a module is present, or • the I/O module has reset itself.	• Either disable the slot in the status file (S:11 and S:12), clear the fault and run, • remove the module, clear the fault and run, or • modify the I/O configuration to include the module, then reload the program and run. • If you suspect that the module has reset itself, clear the major fault and run.
	SLC 5/03 specific - An attempt was made to enter the run or test mode with an empty chassis.	A chassis is void of all I/O modules.	Disable all slots in the empty chassis (see S:11 and S:12).
xx54	A module required for the user program is detected as being the wrong type.	An I/O module in a particular slot is a different type than was configured for that slot by the user.	• Either replace the module with the correct module, clear the fault, and run, or • change the I/O configuration for the slot, reload the program, and run.

Table 15.6 I/O Errors

Error Code (Hex)	Description	Probable Cause	Recommended Action
xx55	A discrete I/O module required for the user program is detected as having the wrong I/O count. This code can also mean that a specialty card driver is incorrect.	• If this is a discrete I/O module, the I/O count is different from that selected in the I/O configuration. • If this is a specialty I/O module, the card driver is incorrect.	• If this is a discrete I/O module, replace it with a module having the I/O count selected in the I/O configuration. Then, clear the fault and run, or • change the I/O configuration to match the existing module, then reload the program and run. • If this is a specialty I/O module, refer to the user manual for that module.
xx56	The chassis configuration specified in the user program is detected as being incorrect.	The chassis configuration specified by the user does not match the hardware.	Correct the chassis configuration, reload the program and run.
xx57	A specialty I/O module has not responded to a Lock Shared Memory command within the required time limit.	The specialty I/O module is not responding to the processor in the time allowed.	Cycle chassis power. If this does not correct the problem, refer to the user manual for the specialty I/O module. You may have to replace the module.
xx58	A specialty I/O module has generated a generic fault. The card fault bit is set (1) in the module's status byte.	Refer to the user manual of the specialty I/O module.	Cycle chassis power. If this does not correct the problem, refer to the user manual for the specialty I/O module. You may have to replace the module.
xx59	A specialty I/O module has not responded to a command as being completed within the required time limit.	A specialty I/O module did not complete a command from the processor in the time allowed.	Refer to the user manual for the specialty I/O module. You may have to replace the module.
xx5A	Hardware interrupt problem.	If this is a discrete I/O module, this is a noise problem. If this is a specialty I/O module, refer to the user manual for the module.	Cycle chassis power. Check for a noise problem and be sure proper grounding practices are used. If this is a specialty I/O module, refer to the user manual for the module. You may have to replace the module.
xx5B	G file configuration error - user program G file size exceeds the capacity of the module.	G file is incorrect for the module in this slot.	Refer to the user manual for the specialty I/O module. Reconfigure the G file as directed in the manual, then reload and run.
xx5C	M0-M1 file configuration error - user program M0-M1 file size exceeds capacity of the module.	M0-M1 files are incorrect for the module in this slot.	Refer to the user manual for the specialty I/O module. Reconfigure the M0-M1 files as directed in the manual, then reload and run.
xx5D	Interrupt service requested is not supported by the processor.	The specialty I/O module has requested service and the processor does not support it.	Refer to the user manual for the specialty I/O module to determine which processors support use of the module. Change processor to one that supports the module.
xx5E	Processor I/O driver (software) error.	Corrupt processor I/O driver software.	Reload program using RSI approved APS software.

Table 15.6 I/O Errors

Error Code (Hex)	Description	Probable Cause	Recommended Action
xx60 through xx6F	Identifies an I/O module specific recoverable major error.	-	-
xx70 through xx7F	Identifies an I/O module specific non-recoverable major error.	-	-
xx80 through xx8F	Identifies a specialty I/O module specific non-recoverable major error.	-	-
xx90	Interrupt problem on a disabled slot.	A specialty I/O module requested service while a slot was disabled.	Refer to the user manual for the specialty I/O module. You may have to replace the module.
xx91	A disabled slot has faulted.	A specialty I/O module in a disabled slot has faulted.	Cycle chassis power. If this does not correct the problem, refer to the user manual for the specialty I/O module. You may have to replace the module.
xx92	Invalid or non-existent module interrupt subroutine (ISR) file.	The I/O configuration/ISR file information for a specialty I/O module is incorrect.	Correct the I/O configuration/ISR file information for the specialty I/O module. Refer to the user manual for the module for the correct ISR file information. Then reload the program and run.
xx93	Unsupported I/O module specific major error.	The processor does not recognize the error code from a specialty I/O module.	Refer to the user manual for the specialty I/O module.
xx94	A module has been detected as being inserted under power in the run or test mode. This can also mean that an I/O module has reset itself.	The module was inserted in the chassis under power, or the module has reset itself.	No module should ever be inserted in a chassis under power. If this occurs and the module is not damaged, • Either remove the module, clear the fault and run, or • add the module to the I/O configuration, reference the module in the user program where required, reload the program, and run.
0x00A0 0x00A1 0x00A2	A major fault unique to the SLC 5/04 or SLC 5/05. The error code indicates communication channel hardware fault has occurred.	Ethernet communication fault. or DH+ communication fault.	The fault may be cleared via a write to the System Status File, but Ethernet/DH+/RS-232 communications will be disabled until a power cycle is performed. For the SLC 5/05 only, word 15 of the System Status File provides a specific fault code for the Ethernet daughterboard when user fault code 0x00A1 is generated.

Troubleshooting SLC 5/03 and Higher Processors

Between the time you apply power to the processor, and it has a chance to establish communication with a connected programming device, the only form of communication between you and the processor is through the LED display.

Powerup LED Display

When power is applied, all the LEDs flash on momentarily and then off. This is part of the normal power-up sequence. Following the self test by the processor, all of the LEDs flash on again momentarily. If a user program is in a running state, the RUN LED is illuminated. If a fault exists within the processor, the FLT LED is illuminated.

Identifying Processor Errors while Downloading an Operating System

The download process takes up to 90 seconds. During this time, watch the LED display for status information. While the download is in progress, the RUN and FLT LEDs remain off. The RS232, DH485 or DH+, Ethernet, FORCE, and BATT LEDs illuminate in a pre-defined sequence. If the download is successful, the above LEDs are illuminated.

If during the download process of an operating system type memory module or during the normal power-up self test process an error occurs, the FLT LED is illuminated and the four LEDs flash on and off at a rate of 2 seconds.

The following table describes the possible LED combinations that are displayed every other time the LEDs flash on.

Table 15.7 LED Combinations

ON LED Display	Description
FAULT, FORCE, DH485, DH+, or Ethernet	Fatal hardware error exists.
FAULT, FORCE, RS232, DH485 or DH+	A hardware watchdog timeout exists.
FAULT, BATT	NVRAM error exists.
FAULT, BATT, RS232	The contents of the operating system memory module are corrupt.
FAULT, BATT, DH485 or DH+	The downloadable operating system is not compatible with the hardware.
FAULT, BATT, RS232, DH485, DH+, or Ethernet	An attempt was made to download the operating system onto write-protected memory.

Table 15.7 LED Combinations

ON LED Display	Description
FAULT, BATT, FORCE	Flash EEPROM failure.
FAULT, BATT, FORCE, RS232	Failure during transmission of downloadable operating system.
FAULT, BATT, FORCE, DH485, DH+, or Ethernet	The operating system is missing or has been corrupted.

SLC 5/03 (OS30x), SLC 5/04 (OS40x) and SLC 5/05 (OS50x) Firmware History

OS300, Series A, FRN 1 released: June 1993

Original Release

OS300, Series A, FRN 2 released: July 1993

Enhancements

None

OS300, Series A, FRN 3 released: March 1994

Enhancements

1. On-Line Editing

   Several changes were made to the On-Line Editing sub-system to decrease the impact to scan time.

2. Instruction Performance

   The IOM, JMP, JSR and OSR instructions have been modified to enhance performance.

3. DII accumulator update during an STI

   The STI instruction has been modified to include copying of the DII accumulator during an STI.

OS300, Series A, FRN 4 released: May 1994

Enhancements

None

OS301, Series A, FRN 5 released: August 1994

Enhancements

1. ASCII Instructions

The ASCII instructions ABL, ACB, ACI, ACL, ACN, AEX, AHL, AIC, ARD, ARL, ASC, ASR, AWA, and AWT are supported in this release. The STRING and ASCII data types are also supported.

2. Floating Point Instructions

The instructions EQU, NEQ, LES, LEQ, GRT, GEQ, ADD, SUB, MUL, DIV, NEG, CLR, SQR, MOV, FLL, COP, LIM support floating point data in this release.

3. 10usec User Interrupt Timer

The lower 16 bits of the processor's 20-bit 10usec internal free running clock is copied to either S:43 (STI), S:44(IOI), or S:45(DII) prior to executing the user interrupt ladder file. This allows a user to determine the amount of time that has elapsed between consecutive interrupt subroutines.

4. PLC-5 Read/Write Capability

PLC-5 typed read and write commands can now be initiated and received.

5. Average Scan Time Calculation

The average scan time calculation has been changed to more accurately calculate the average scan time.

6. DTR Control

The controlling of the DTR signal has been changed so that it is turned back ON only if the forcing control bit is enabled when the processor goes from a RUN to non-RUN mode.

7. Dual Message Buffering

The dual message buffering capabilities has been increased from 4 buffers for both channels to 4 buffers per channel.

8. Remote Messaging

Remote messaging is now allowed beyond the maximum node address.

OS400, Series A, FRN 1 released: August 1994

Original Release

OS301, Series A, FRN 6
OS400, Series A, FRN 2
released: November 1994

Enhancements

None

OS301, Series A, FRN 7
OS400, Series A, FRN 3
released: March 1995

Enhancements

1. Selection of number of data bits and stop bits with generic ASCII communications added

   The Generic ASCII protocol has been expanded to allow 7 or 8 data bits and 1, 1.5, or 2 stop bits. Generic ASCII communications is selected when Channel 0 is placed into User Mode.

2. Poll Time-out with DF1 Half-Duplex Slave Communication

   The Poll Time-out feature associated with DF1 Half-Duplex Slave communications on Channel 0 has been changed so that reply data packets queued to be transmitted when a Poll Time-out occurs will no longer be purged from the queue. The only event which will now purge the reply packets is reception of a NAK from the DF1 Master. This ensures that no matter how much time elapses between when a DF1 Master sends a command packet to the 5/03 and when the master polls that same 5/03, the reply to that command will be returned by that 5/03. Command data packets generated by MSG Instructions and which are queued and waiting for transmission will still be purged with their associated MSG Instruction being errored with the type 0005 code.

3. Read of “Initialized” Data Files during download

   The 5/03 processor uses hardware to CRC data files. As a data byte is written, a CRC is generated in the next byte. When the data is read back from the processor, the CRC is automatically checked. If it fails, then a hardware error occurs and the processor resets. Therefore, by the way the CRC works, data cannot be read from the processor until it has been written. In one application, the user was continuously polling all of the processors for information by reading various data files. Since this polling could happen anytime, they were often colliding with a download procedure and causing processors to reset. A change was made in the firmware to not make use of the automatic CRC checking of data files during a download, thereby preventing this from happening.

OS301, Series A, FRN 8
OS400, Series A, FRN 4
released: April 1995

Enhancements

None

OS302, Series A, FRN 9
OS401, Series A, FRN 5
released: December 1995

Enhancements

1. Indirect Addressing

   Allows for simplified programming.

2. Trigonometric and Exponential Math Functions

   Includes SIN, COS, TAN, ASN, ACS, ATN, LN, LOG, ABS, DEG, RAD, and XPY.

3. Compute (CPT) Instruction

   Allows for complex math computations.

4. Swap (SWP) Instruction

   Allows for the exchange of the high and low bytes of a 16-bit word, providing easier manipulation of ASCII data within the SLC processors.

5. Scale with Parameters (SCP) Instruction

   Simplifies scaling of analog parameters.

6. DF1 Half-Duplex Master Protocol

   Allows the processor to support SCADA master RTU applications.

7. Multi-Point List

   Allows for monitoring of any 32 bits from one screen.

8. Global Status Flags on DH+ (OS401 only)

   Provides for high-speed broadcast to all processors.

9. DF1 to DH+ Passthru (OS401 only)

Allows user to connect to the SLC 5/04 processor's serial port with a computer and then access any node on the DH+ network, regardless of the baud rate of the DH+ network.

10. Remote I/O (RIO) Passthru via a 1747-SN Scanner Module (OS401 only)

Allows an SLC 5/04 processor to act as a bridge between DH+ and RIO. Remote I/O passthru also supports uploads/downloads of applications to RIO devices.

11. Program Memory of 12K, 28K, or 60K words and 4K of additional data words (OS401 only)

Offers a variety of modular processors that fit a variety of memory requirements.

Catalog # Memory
Previous: 1747-L542 20k words + 4k
New: 1747-L541
1747-L542
1747-L543
12k words + 4k
28k words + 4k
60k words + 4k

OS401, Series A, FRN 6 released: May 1996

Enhancements

None

OS302, Series B, FRN 10 OS401, Series B, FRN 7 released: July 1997

Enhancements

1. Day of the Week

System Status File word S:53 = 0 for Sunday, 1 for Monday, up to 6 for Saturday. It will contain a random value until a legal date is entered into the System Status File. A download of a user program with a valid date or manual entry of a valid date will work. Once a value is entered, it will be non-volatile.

Special Note

This is the first firmware release for the 8k SLC 5/03.

OS500, Series A, FRN 1 released: October 1997

Original Release

OS302, Series B, FRN 11 OS401, Series B, FRN 8 OS500, Series A, FRN 2 released: November 1997

Enhancements

1. 19200 DF1 FD Default Baud Rate (OS302 and OS401 only)

   The default baud rate of channel 0 has been modified from 1200 to 19200.

OS302, Series B, FRN 12 released: November 1998 OS401, Series B, FRN 9 released: July, 1999 OS501, Series A, FRN 3 released: July 1998

Enhancements (OS501 only)

1. Channel 0 (DF1 FD & DH485) to Channel 1 (Ethernet) Passthru
2. Channel 1 (Ethernet) Remote Messaging

OS302, Series B, FRN 12 released: November 1998 OS401, Series B, FRN 9 released: July 1999 OS501, Series A, FRN 4 released: February 1999

Enhancements

None

OS302, Series B, FRN 14 OS401, Series B, FRN 9 released: July 1999 OS501, Series A, FRN 4 released: February, 1999

Enhancements

1. Daughtercard Fault Signal (OS302 and OS401 only)

   When the daughtercard sends a fault signal to the motherboard, the motherboard will stop the communication of the processor and set error code 0xA1. Communications will be restored after power is cycled to the processor.

2. Improve Interrupt Performance

In the previous release, it took 60 µsec when saving interrupt information at the end of scan. This has been improved to 40 µsec.

3. Improve Accuracy of Last Scan Time

4. Autoload Memory Module Program

   In the previous release, when the bit S2:1/10 (Load Memory Module on Memory Error bit) is set, if the processor has a hardware error of 0x01 at power up, the program in a memory module can be automatically transferred from memory to the processor. In the current release, when the bit S2:1/10 (Load Memory Module on Memory Error bit) is set, if the processor has a hardware error of 0x01 to 0x0F at power up, the program in a memory module can be automatically transferred from memory to the processor.

OS501, Series A, FRN 5 released: April 1999

Enhancements

1. Multi-hop Messaging

   Support was added for sending messages to and receiving messages from ControlLogix Ethernet cards.

OS302, Series C, FRN 3 OS401, Series C, FRN 3 OS501, Series C, FRN 3 released: September 2000

Enhancements

1. Added Block Transfer Instructions (BTR and BTW)

   With block-transfer instructions, you can transfer up to 64 words to or from a remote device over an Allen-Bradley RIO link. A Block Transfer Read (BTR) is used when a remote device transfers data to the SLC. A Block Transfer Write (BTW) is used when an SLC processor writes data to a remote device. The RIO scanners (1747-SN and 1747-BSN) perform block transfer through M0 and M1 files buffers.

   A BTR or BTW instruction writes information into its control structure address (a three-word control file) when the instruction is entered. The processor uses these values to execute the transfer.

2. Read High Speed Clock Instruction (RHC)

The SLC processor maintains a 10μS long integer free running clock/counter. This 20-bit value increments every 10uS. It is accessed using the RHC instruction. When the RHC is evaluated with a false rungstate, during prescan, or inside of a false MCR zone, no operation is performed. When the RHC is evaluated with a true rungstate, the instruction moves the current value of the 10uS free running clock into the destination address. If it is an integer address, it only moves the least 16 bits into the address. If it is a float address, it converts the long integer value into a float and moves it to the relative address. After the free running clock reaches 0xfffff value (10.4857 sec), it will wrap around to 0 and continues incrementing. The RESET signal or Power Cycle will set this free running clock to 0.

3. Compute Time Difference Instruction (TDF)

The compute Time Difference Instruction (TDF) is used to calculate the elapsed time between any 2 timestamps captured using the RHC instruction. This allows the user program to time any event using a 10uS timebase.

When the TDF is evaluated with a false rungstate, during pre-scan, or inside a false MCR zone, no operation is performed. When the TDF is evaluated with a true rungstate, the instruction calculates the number of 10uS “ticks” that have elapsed from the Start value to the Stop value and places the result into the Destination. The TDF instruction with float address will accurately compute the time difference between any 2 timestamps captured within 10.48575 seconds of each other (1048575 10uS ticks). The TDF with float address will calculate an invalid result if more than 10.48575 seconds have elapsed between the start and stop timestamps. Meanwhile, the TDF instruction with integer address will compute the positive time difference between the START and END timestamps. The TDF with integer address will calculate an invalid result if more than 327.67ms have elapsed between the start and stop timestamps.

4. Encode 1 of 16 to 4 Instruction (ENC)

The ENC instruction provides the ability to give the first set bit position in an integer value.

When the rung is true, this output instruction searches the source from the lowest to the highest bit, and looks for the first set bit. The corresponding bit position is written to the destination integer.

5. Ramp Instruction (RMP)

The Ramp (RMP) instruction provides the ability to create linear, acceleration, deceleration, and "S" curve ramp output data wave forms. The instruction provides a means to ramp analog outputs when using them to control devices such as valves.

When the Ramp function is triggered, parameters are validated to be in range. If the parameters are valid, the ramp function places the Beginning Output Value in the Destination register. The format of the control block will be defined as part of the development process. It is permissible for the control block to take up User Ladder Program space as well as use additional user memory for storing runtime ramp information that is not user accessible.

6. File Bit Comparison Instruction (FBC) and Diagnostic Detect Instruction (DDT)

The FBC and DDT diagnostic instructions are output instructions that you can use to monitor machine or process operations to detect malfunctions. If you want to detect malfunction by comparing bits in a file of real-time inputs with a reference bit file that represents correct operation, use FBC instruction. If you want to change the reference file to match the input file, use DDT.

Both the FBC and DDT instructions compare bits in a file of real-time machine or process values (input file) with bits in a reference file, detect deviations, and record mismatched bit numbers. They record the position of each mismatch found and place this information in the result file. If no mismatches are found, the .DN bit is set but the result file remains unchanged.

The difference between the FBC and DDT instructions is that each time the DDT instruction finds a mismatch, the processor changes the reference bit to match the source bit. The FBC instruction does not change the reference bit. Use the DDT instruction to update your reference file to reflect changing machine or process conditions.

7. Messaging Interrupt Message Reply

Change 'Message Reply' disable/enable interrupt scheme to enhance system performance during STI execution.

8. RIO Passthru Function for BSN

The passthru function is now enabled for both the 1747-SN and the 1747-BSN.

9. Error Code Trapping 'Operating System'

The error code trapping is used to get the latest 8 error structures for hard faults. Error structures are retentive after power cycle, assuming the battery is connected and charged. Error code trapping only works for hard faults (0x00—0x0F).

10. Updated Operating System Flash programming Algorithm to include 5V 'JEDEC'

This change is required to allow memory module hardware upgrades to 5V flash technology.

11. Message Error Code (OS501 only)

After a Unix Server has defined the unsolicited "Client" IP address in SLC 5/05, the Server is removed from the network. The SLC 5/05 "Client" messages continue to be initiated since the user program re-triggers them on a message error. The "Client" messages normally error out with a 0x10 error code (invalid command parameters), probably since the Server's IP address has been removed from the "Client" IP table. However, the messages stop re-triggering after several seconds indicating that the message was "done" with an error code 0x10.

The mother board firmware will add one feature to give more error information for the MSG instruction. Two more bits (word 12 bit 0 and word 12 bit 1) in control block are used to give the error information. When daughtercard returns error code, .ER is set, bit 1 (word 12 bit 0) in control block is set, error code, which is a non-zero value, will be put into the control block. When daughtercard returns no error code, and motherboard got an error code in the PCCC command reply, .ER is set, bit 2 (word 12 bit 1) is set, error code, which is a non-zero value, will be put into the control block. When daughtercard has no error return, and motherboard get the right PCCC reply without error, .DN is set, bit 1 and bit 2 are reset error code in control block will be 0. Bits.ER, bit 1 and bit 2 will be reset once the MSG is enabled.

12. Daughtercard Firmware Revision/Series Check (OS501 only)

A new feature will be added in the current release to check the daughter card firmware revision/series. This feature will be implemented in the power up. When the daughter card series is not same to a specific number, or daughter card revision is not same to a specific number, a non-user fault (0x3B) will occur. If series is 9999, no fault occurs.

13. Expanded Channel 1 Diagnostic File (OS501 only)

The processor can support channel 1 diagnostic file. This diagnostic file can be any user defined integer file, which file number is in the range of 9 to 255. The existing diagnostic information structure size in firmware is 44 words. However, the new daughtercard supports 50 words information. The 45th word contains the number of network storms since the last power cycle. The 46th word to 48th word contains the Ethernet hardware address. The 49th word and 50th word contains IP address. In the new firmware release, the diagnostic information structure in firmware will be expanded to 50 words to get all the channel 1 information from daughtercard to match daughtercard requirement.

14. Update NETBSD TCP/IP Stack (OS501 only)

The Berkley NETBSD TCP/IP stack was recently ported over to the 1756-ENET module to replace the Berkley Software Distribution (BSD) that has been in use since the development of the PLC-5 Ethernet over eight years ago. This stack was also ported over to the legacy Ethernet products to let us take advantage of any bug fixes we did not pickup over the years, enhanced UDP message support and the ability to do super-netting.

15. Addition of Connection Count Added to Diagnostics (OS501 only)

The diagnostic sub-segment in the dualport was modified to contain a count of the total ethernet connections, the inbound connections and the outbound connections. This data is transferred to the ethernet channel diagnostic file (if of sufficient length) during the motherboard housekeeping. The counts are available at word 50 (total), 51 (inbound) and 52 (outbound) in the diagnostic file for the ethernet channel.

16. Addition of IBD (I Be Dead) Data Added to Diagnostics (OS501 only)

The diagnostic sub-segment in the dualport was modified to contain IBD data upon crash of the daughtercard. If the crash occurs after communication with the motherboard is established, this data is transferred to the ethernet channel diagnostic file (if of sufficient length) during the motherboard housekeeping. The data contains register, stack, address, traceback and error information about the crash. The data starts at word offset 5 of the diagnostic file. This will allow the IBD data to be maintained in the diagnostic file over subsequent power-cycles, allowing for support analysis.

17. Memory Module Bootp problem (OS501 only)

Currently, there is a bug for the memory module. If S2:1/10 bit (Load Memory Module on Memory Error Bit) is set, when processor has hardware fault, program in memory module will be loaded into the processor, the IP address in memory module will be applied with bootp is disabled in the memory module. However, current processor applies the IP address set by bootp under above situation.

OS302, Series C, FRN 4
OS401, Series C, FRN 4
OS501, Series C, FRN 4
released: February 2001

Enhancements

None

OS302, Series C, FRN 5
OS401, Series C, FRN 5
OS501, Series C, FRN 5
released: October 2001

Enhancements

1. Additional Ethernet connections for 32k and 64k processors (OS501 only)

   The total number of available Ethernet connections has increased by eight from 16 to 24 in the 32k (L552) and 64k (L553) SLC 5/05 processors. Four connections are reserved for incoming (client) messages, four connections are reserved for outgoing (peer) messages and 16 connections can be used for either incoming or outgoing messages. Therefore, the maximum number of connections in either direction is 20.

   For the 16k (L551) SLC 5/05 processor, the total number of available Ethernet connections remains at 16, and the maximum number of connections in either direction remains at 12.

2. Embedded web server capability for module information, TCP/IP configuration and diagnostic information (OS501 only)

   The SLC 5/05 processor now includes an embedded web server which allows viewing of module information, TCP/IP configuration and diagnostic information via Ethernet using a standard web browser. In order to view the web server main menu, type in “http://www.xxx.yyy.zzz” for the web address in the web browser, where www.xxx.yyy.zzz is the IP address of the SLC 5/05 processor.

3. Domain Name System (DNS) support in the Ethernet message instruction (OS501 only)

The SLC 5/05 Ethernet MSG instruction now includes DNS support, which allows entering in a device name of up to 41 characters inplace of an IP addresss. Before making a connection to that device, the SLC 5/05 will query a Domain Name Server on the Ethernet network for the IP address of the device with that name. In order to successfully use the DNS capability, a “Primary Name Server” must be defined in thte Channel 1 Ethernet Configuration. A “Secondary Name Server” and “Default Domain Name” may also be configured. If a “Default Domain Name” is configured, then it gets appended to every device name that is entered into an Ethernet MSG instruction.

TIP

RSLogix 500 version 5.20 or higher must be used to take advantage of this DNS capability.

SLC Status File

This appendix lists the:

- SLC processor status file overview
- status file detailed word/bit descriptions

This appendix discusses the status file functions of the Fixed, SLC 5/01, SLC 5/02, SLC 5/03, SLC 5/04 and SLC 5/05 processors. The processors function similarly, but the higher numbered processors utilize more features. The tables in this appendix indicate which functions are supported by each processor.

The appendix starts with an overview listing of the status file. A more detailed description of each status word follows. Use the overview list to find the page number of the detailed description.

Status File Overview

The status file lets you monitor how your operating system works and lets you direct how you want it to work. This is done by using the status file to set up interrupts, load memory module programs, and monitor both hardware and software faults.

TIP

Do not write to status file data unless the word or bit is listed as dynamic/static configuration in the descriptions that follow. If you intend writing to status file data, it is imperative that you first understand the function fully.

The status file S: contains the following words:

Word	Function	Applies To	Page
S:0	Arithmetic and Scan Status Flags	**all processors**	B-5
S:1L	Processor Mode Status/Control		B-6
S:1H	Processor Mode Status/Control		B-6
S:2	Processor Alternate Mode Status/Control		B-12
S:3L	Current Scan Time		B-18
S:3H	Watchdog Scan Time		B-19
S:4	Free Running Clock		B-20
S:5	Minor Error Bits		B-20
S:6	Major Error Fault Code		B-24
S:7, S:8	Suspend Code/Suspend File		B-32
S:9	DH-485 Active Nodes (Fixed, SLC 5/01, SLC 5/02) Channel 1 Active Nodes (SLC 5/03) Unused (SLC 5/04) Ethernet Daughterboard Firmware Series (SLC 5/05)		B-32
S:10	DH-485 Active Nodes (Fixed, SLC 5/01, SLC 5/02) Channel 1 Active Nodes (SLC 5/03) Unused (SLC 5/04) Ethernet Daughterboard Firmware Revision (SLC 5/05)		B-32
S:11, S;12	I/O Slot Enables		B-33
S:13, S:14	Math Register		B-34
S:15L	DH-485 Node Address (Fixed, SLC 5/01, SLC 5/02) Channel 1 DH-485 Node Address (SLC 5/03) Channel 1 DH+ Node Address (SLC 5/04) Ethernet Daughterboard Fault Code (SLC 5/05)		B-35
S:15H	DH-485 Baud Rate (Fixed, SLC 5/01, SLC 5/02) Channel 1 DH-485 Baud Rate (SLC 5/03) Channel 1 DH+ Baud Rate (SLC 5/04) Ethernet Daughterboard Fault Code (SLC 5/05)		B-35

Word	Function	Applies To	Page
S:53H and S:54	Reserved	**SLC 5/03 and higher**	B-52
S:55	Last DII ISR Scan Time		B-51
S:56	Maximum DII ISR Scan Time		B-51
S:57	Operating System Catalog Number		B-51
S:58	Operating System Series		B-51
S:59	Operating System FRN		B-51
S:60	Processor Catalog Number		B-51
S:61	Processor Series		B-51
S:62	Processor Revision		B-51
S:63	User Program Type		B-51
S:64	User Program Functionality Index		B-51
S:65	User RAM Size		B-51
S:66	Flash EEPROM Size		B-52
S:67 to S:82	Channel 0 Active Node Table		B-52
S:83 to S:86	Channel 1 Active Node Table	**SLC 5/04**	B-52
S:87 to S:98	Reserved		B-52
S:99	Global Status Word		B-52
S:100 to S:163	Global Status File		B-52

Status File Details

Conventions Used in the Displays

The following tables describe the status file functions, beginning at address S:0 and ending at address S:163. A bullet (•) indicates that the function applies to the specified processor.

The following classifications are used:

- **Status** - Use these words, bytes, or bits to monitor processor options or processor status information. The information is seldom written to the user program or programming device (unless you want to reset or clear a function such as a minor error bit).
- **Dynamic Configuration** - Use these words, bytes, or bits to select processor options while in the RUN mode.
- **Static Configuration** - Use these words, bytes, or bits to select processor options prior to entering the RUN mode. Note that some options must be selected while in the offline program mode, prior to restoring the user program.

Address	Classification	Description	Fixed 5/01	5/02	5/03	5/04	5/05
S:0		**Arithmetic and Scan Status Bits** The arithmetic flags are assessed by the processor following the execution of any math, logical, or move instruction. The state of these bits remains in effect until the next math, logical, or move instruction in the program is executed.	•	•	•	•	•
S:0/0	**Status**	**Carry Bit** This bit is set by the processor if a mathematical carry or borrow is generated. Otherwise the bit remains cleared. This bit is assessed as if a function of unsigned math.	•	•	•	•	•
		When a STI, I/O Slot, or Fault Routine interrupts normal execution of your program, the original value of S:0/0 is restored when execution resumes.		•	•	•	•
		When a DII interrupts normal execution of your program, the original value of S:0/0 is restored when execution resumes.			•	•	•
S:0/1	**Status**	**Overflow Bit** This bit is set by the processor when the result of a mathematical operation does not fit in its destination. Otherwise the bit remains cleared. Whenever this bit is set, the overflow trap bit S:5/0 is also set. Refer to S:5/0.	•	•	•	•	•
		When a STI, I/O Slot, or Fault Routine interrupts normal execution of your program, the original value of S:0/1 is restored when execution resumes.		•	•	•	•
		When a DII interrupts normal execution of your program, the original value of S:0/1 is restored when execution resumes.			•	•	•
S:0/2	**Status**	**Zero Bit** This bit is set by the processor when the result of a math, logical, or move instruction is zero. Otherwise the bit remains cleared.	•	•	•	•	•
		When a STI, I/O Slot, or Fault Routine interrupts normal execution of your program, the original value of S:0/2 is restored when execution resumes.		•	•	•	•
		When a DII interrupts normal execution of your program, the original value of S:0/2 is restored when execution resumes.			•	•	•
S:0/3	**Status**	**Sign Bit** This bit is set by the processor when the result of a math, logical, or move instruction is negative. Otherwise the bit remains cleared.	•	•	•	•	•
		When a STI, I/O Slot, or Fault Routine interrupts normal execution of your program, the original value of S:0/3 is restored when execution resumes.		•	•	•	•
		When a DII interrupts normal execution of your program, the original value of S:0/3 is restored when execution resumes.			•	•	•
S:0/4 to S:0/15	**NA**	**Reserved**	•	•	•	•	•

Address	Classification	Description	Fixed 5/01	5/02	5/03	5/04	5/05
S:1/0 to S:1/4	**Status**	**Processor Mode Status/Control** Bits 0-4 function as follows:	•	•	•	•	•
		0 0000 = (0) Remote Download in progress.					
		0 0001 = (1) Remote Program mode (the fault mode exists when bit S:1/13 is set along with mode 0 0001)					
		0 0011 = (3) Suspend Idle (operation halted by SUS instruction execution) fault mode exists when bit S:1/13 is set along with mode 0 0011.					
		0 0110 = (6) Remote Run mode					
		0 0111 = (7) Remote Test continuous mode					
		0 1000 = (8) Remote Test single scan mode					
		0 1001 = (9) Remote Test single step (step until)					
	TIP	All modes in the fixed, SLC 5/01, and SLC 5/02 processors are considered as remote because they do not have a keyswitch.					
		1 0000 = (16) Download in progress (keyswitch=PROGram)			•	•	•
		1 0001 = (17) PROGram mode - the fault mode exists when bit S:1/13 is set along with mode 1 0001.					
		1 1011 = (27) Suspend Idle - the fault mode exists when bit S:1/13 is set along with mode 1 1011 (keyswitch=RUN)					
		1 1110 = (30) RUN - the fault mode exists when bit S:1/13 is set along with mode 1 1110 (keyswitch=RUN).					
		All other values for bits 0-4 are reserved.					
S:1/5	**Status**	**Forces Enabled Bit** This bit is set by the processor if you have enabled forces in a ladder program. Otherwise, the bit remains cleared. The processor Forced I/O LED is on continuously when forces are enabled.	•	•	•	•	•
S:1/6	**Status**	**Forces Installed Bit** This bit is set by the processor if you have installed forces in a ladder program. The forces may or may not be enabled. Otherwise the bit remains cleared. The processor Forced I/O LED flashes when forces are installed, but not enabled.	•	•	•	•	•
S:1/7	**Status**	**Communications Active Bit (Channel 1)** This bit is set by the processor when at least one other node is present on the network attached to channel 1. Otherwise, the bit remains cleared. When the node is active, it is a recognized participant in a DH-485 or DH+ token-passing network. For Ethernet communications, this bit is only an indication that the Ethernet daughter board is functioning properly, not necessarily that there are any other active Ethernet nodes, or that channel 1 is connected to an Ethernet network.	•	•	•	•	•

Address	Classification	Description	Fixed 5/01	5/02	5/03	5/04	5/05
S:1/8	**Dynamic Config**	**Fault Override at Powerup Bit** When set, this bit causes the processor to clear the Major Error Halted bit S:1/13 and Minor error bits S:5/0 to S:5/7 on power up; if the processor had previously been in the REM Run mode and had faulted. The processor then attempts to enter the REM Run mode. When this bit remains cleared (default value), the processor remains in a major fault state at power up. To program this feature, set this bit using the Data Monitor function.	•	•	•	•	•
S:1/9	**Dynamic Config**	**Startup Protection Fault Bit** When this bit is set and power is cycled while the processor is in the REM Run mode, the processor executes your fault routine prior to the execution of the first scan of your program. You then have the option of clearing the Major Error Halted bit S:1/13 to resume operation in the REM Run mode. If your fault routine does not reset bit S:1/13, the fault mode results. To program this feature, use the Data Monitor function, then program your fault routine logic accordingly. When executing the startup protection fault routine, S:6 (major error fault code) will contain the value 0016H.		•	•	•	•
S:1/10	**Static Config** (See table on page B-53 for all setting combinations.)	**Load Memory Module on Memory Error Bit** You can use this bit to transfer a memory module program to the processor in the event that a processor memory error is detected at power-up. A memory error means the processor cannot run the program in the RAM because the program has been corrupted, as detected by a parity or checksum error. This type of error is caused by battery or capacitor drain, noise, or a power problem. You must set S:1/10 in the status file of the program in the memory module. When a memory module is installed that has bit S:1/10 set, a processor memory error detected at power-up causes the memory module program to be transferred to the processor, and the REM Run mode to be entered. When S:1/10 is cleared in the memory module, the processor remains in a major fault condition if a memory error is detected on power-up, regardless if a memory module exists. When S:1/10 is set in the status file of the user program in RAM memory, the memory module must be installed at all times to enter the REM Run or REM Test modes. To program this feature, set this bit using the Data Monitor function. Then store the program in the memory module.	•	•	•	•	•

<table>
<tr><th>Address</th><th>Classification</th><th>Description</th><th>Fixed 5/01</th><th>5/02</th><th>5/03</th><th>5/04</th><th>5/05</th></tr>
<tr><td rowspan="6">S:1/11</td><td rowspan="6">Static Config
(See table on page B-53 for all setting combinations.)</td><td>Load Memory Always Bit
When this bit is set, you can overwrite a processor program with a memory module program by cycling processor power. A programming device is not required. The processor mode after powerup is as follows for SLC 5/02 and higher processors:
<table><tr><th>Mode before Powerdown</th><th>Mode after Powerup</th></tr><tr><td>REM Test/Program
REM Run
Fault after REM
Test/Program
Fault after REM Run
REM Idle
REM Download</td><td>REM Program
REM Run
REM Program
REM Run
REM Program
REM Program</td></tr></table></td><td></td><td>•</td><td>•</td><td>•</td><td>•</td></tr>
<tr><td><table><tr><th>Mode before Powerdown</th><th>Mode after Powerup (same keyswitch position)</th></tr><tr><td>Run
Program
Idle
Fault after Run
Fault after Program</td><td>RUN
PROGram
RUN
RUN
PROGram</td></tr></table></td><td></td><td></td><td>•</td><td>•</td><td>•</td></tr>
<tr><td>TIP
  </td><td>All modes in the fixed, SLC 5/01, and SLC 5/02 processors are considered to be remote because they do not have a keyswitch.</td><td rowspan="4"></td><td rowspan="4">•</td><td rowspan="4">•</td><td rowspan="4">•</td><td rowspan="4">•</td></tr>
<tr><td></td><td>The memory module you install in the processor must have status file bit S:1/11 set. Loading takes place if the master password and/or password in the processor and memory module match. Loading can also take place if the processor has neither a password nor master password.
When S:1/11 is also set in the status file of the user program in RAM, the memory module must be installed at all times to enter the REM Run or REM Test modes.</td></tr>
<tr><td>ATTENTION
  </td><td>The overwriting process, including data tables, is repeated each time you cycle power.</td></tr>
<tr><td></td><td>To program this feature, set this bit using the Data Monitor function. Then store the program in the memory module.</td></tr>
<tr><td></td><td></td><td>You may choose not to overwrite data files on a per file basis.</td><td></td><td></td><td>•</td><td>•</td><td>•</td></tr>
</table>

<table>
<tr><th>Address</th><th>Classification</th><th>Description</th><th>Fixed 5/01</th><th>5/02</th><th>5/03</th><th>5/04</th><th>5/05</th></tr>
<tr><td rowspan="3">S:1/12</td><td rowspan="3">Static Config
(See table on page B-53 for all setting combinations.)</td><td>Load Memory Module and Run Bit
With this bit, you can overwrite a processor program with a memory module program by cycling processor power. A programming device is not required. The processor will attempt to enter the REM Run mode, regardless of what mode was in effect before cycling power:</td><td></td><td>•</td><td>•</td><td>•</td><td>•</td></tr>
<tr><td><table><tr><th>Mode before Powerdown</th><th>Mode after Powerup</th></tr><tr><td>REM Test/Rem Program
REM Run/Rem Fault
REM Idle/Rem Download</td><td>REM Run
REM Run
REM Run</td></tr></table></td><td></td><td>•</td><td>•</td><td>•</td><td>•</td></tr>
<tr><td><table><tr><th>Mode before Powerdown</th><th>Mode after Powerup
(same keyswitch position)</th></tr><tr><td>Run
Idle
Program/Download
Fault after Run
Fault after Program</td><td>RUN
Run
PROGram
RUN
PROGram</td></tr></table></td><td></td><td></td><td>•</td><td>•</td><td>•</td></tr>
</table>

Address	Classification	Description	Fixed 5/01	5/02	5/03	5/04	5/05
S:1/12 continued	**TIP** 	All modes in the fixed, SLC 5/01, and SLC 5/02 processors are considered to be remote because they do not have a keyswitch.		•	•	•	•
		The memory module you install in the processor must have a status file bit S:1/12 set. Loading takes place if the master password and/or password in the processor and memory module match. Loading can also take place if the processor has neither a password not master password. When S:1/12 is set in the status file of the user program in RAM, it does not require the presence of the memory module to enter the REM Run or REM Test mode. **Application example:** Set both S:1/11 and S:1/12 to autoload and run every power cycle, and require the presence of the memory module to enter the REM Run or REM Test modes.					
	ATTENTION 	If you leave the memory module installed, the overwriting process, including data tables, is repeated each time you cycle power. The mode is changed to REM Run each and every power cycle.					
		To program this feature, use the Data Monitor function. Then store the program in the memory module. This feature is particularly useful when you are troubleshooting hardware failures with "spares" (replacement modules). Use this feature to facilitate application logic upgrades in the field without a programming device.					
		You may choose not to overwrite data files on a per file basis.			•	•	•

Address	Classification	Description	Fixed 5/01	5/02	5/03	5/04	5/05
S:1/13	**Dynamic Config**	**Major Error Halted Bit** This bit is set by the processor any time a major error is encountered. The processor enters a fault condition. Word S:6, Fault Code will contain a code which can be used to diagnose the fault condition. Any time bit S:1/13 is set, the processor: • either places all outputs in a safe state and energizes the fault LED, or	•	•	•	•	•
		• enters the user fault routine with outputs active, allowing the fault routine ladder logic to attempt recovery from the fault condition. If your fault routine determines that recovery is required, clear S:1/13 using ladder logic prior to exiting the fault routine. If the fault routine ladder logic does not understand the fault code, or if the routine determines that it is not desirable to continue operation, exit the fault routine with bit S:1/13 set. The outputs will be placed in a safe state and the fault LED will be energized.		•	•	•	•
		When you clear bit S:1/13 using a programming device, the processor mode changes from fault to either Remote Program, or Remote Idle Suspend depending on the previous mode of the processor. You can move a value to S:6, then set S:1/13 in your ladder program to generate an application specific Major Error.	•	•	•	•	•
	TIP 	Once a major fault state exists, you must correct the condition causing the fault, and you must also clear this bit in order for the processor to accept a mode change attempt (into REM Program, REM Run, or REM Test). Also, clear S:6 to avoid the confusion of having an error code but no fault condition.					
	TIP 	Do not re-use error codes that are defined in the SLC error code list in chapter as application specific error codes. Instead, create your own unique codes. This prevents you from confusing application errors with system errors. We recommend using error codes FF00 to FF0F to indicate application specific major errors.					
		When you clear bit S:1/13 using a programming device, the processor mode changes from fault to either Program, Run, or Idle Suspend depending on the previous mode of the processor. You can move a value to S:6, then set S:1/13 in your ladder program to generate an application specific major error.			•	•	•
	ATTENTION 	If you clear this bit with the keyswitch in RUN, the processor immediately enters the RUN mode.					
		You can clear faults S:1/13 and S:6 by cycling the keyswitch to PROGram and then to RUN					

Address	Classification	Description	Fixed 5/01	5/02	5/03	5/04	5/05
S:1/14	**Status**	**Access Denied Bit (OEM Lock)** You can allow or deny future access to a processor file. Set this bit to deny access. This indicates that a programming device must have a matching copy of the processor file in its memory in order to monitor the ladder program. A programming device that does not have a matching copy of the processor file is denied access. To program this feature, select "Future Access Disallow" when saving your program. To provide protection from inadvertent data monitor alteration of your selection, program an unconditional OTL instruction at address S:1/14, to deny future access. Program an unconditional OTU instruction at address S:1/14 to allow future access. When this bit is cleared, it indicates that any compatible programming device can access the ladder program (provided that password conditions are satisfied).	•	•	•	•	•
		When access is denied, the programming device (APS or HHT) may not access the ladder program. Functions such as change mode, clear memory, restore program, and transfer memory module are allowed regardless of this selection. A device such as the DTAM is not affected by this function.					
S:1/15	**Status**	**First Pass Bit** Use this bit to initialize your program as the application requires. When this bit is set by the processor, it indicates that the first scan of the user program is in progress (following power up in the RUN mode or entry into a REM Run or REM Test mode). The processor clears this bit following the first scan. When this bit is cleared, it indicates that the program is not in the first scan of a REM Test or REM Run mode.	•	•	•	•	•
		This bit is set during execution of the startup protection fault routine. Refer to S:1/9 for more information.		•	•	•	•
S:2/0	**Status**	**STI (Selectable Timed Interrupt) Pending Bit** When set, this bit indicates that the STI timer has timed out and the STI routine is waiting to be executed. This bit is cleared upon starting of the STI routine, power up, exit of the REM Run mode, or execution of a true STS instruction.		•	•	•	•
		The STI pending bit will not be set if the STI timer expires while executing the fault routine.		•			
		This bit is set if the STI timer expires while executing the DII subroutine or fault routine.			•	•	•

Address	Classification	Description	Fixed 5/01	5/02	5/03	5/04	5/05
S:2/1	**Static Config**	**STI (Selectable Timed Interrupt) Enabled Bit** This bit is set in its default condition, or when set by the STE or STS instruction. If set, it allows execution of the STI if the STI file (S:31) and STI setpoint (S:30) are non-zero. If clear, when an interrupt occurs, the STI subroutine does not execute and the STI Pending bit is set. The STI Timer continues to run when disabled. The STD instruction clears this bit.		•	•	•	•
	Dynamic Config	Use the Data Monitor function to set and clear this bit, or address this bit with your ladder logic program. This bit is set in its default condition, or when set by the STE or STS instruction. If set, it allows execution of the STI if the STI file (word 31) and STI rate (word 30) are non-zero. If clear, the STI subroutine does not execute and the STI pending bit is set. The STI timer continues to run. The STD instruction clears this bit.			•	•	•
S:2/2	**Status**	**STI (Selectable Timed Interrupt) Executing Bit** When set, this bit indicates that the STI timer has timed out and the STI subroutine is currently being executed. This bit is cleared upon completion of the STI routine, powerup, or REM Run mode entry. **Application example:** You can examine this bit in your fault routine to determine if your STI was executing when the fault occurred.		•	•	•	•
S:2/3	**Static Config**	**Index Addressing File Range Bit** When clear, the index register can only index within the same data file of the specified base address. When set, the index register can index anywhere from data file B3:0 to the end of the last declared data file. This bit is selected at the time you save your program.		•	•	•	•
		The SLC 5/03 and higher processors allow you to index from 0:0 to the last data file.			•	•	•
	TIP	Change this bit while in the offline mode only. Save the program after changing the bit.					
S:2/4	**Static Config**	**Saved with Single Step Test Enabled Bit** When clear, the Single Step Test mode function is not available. Clear also indicates that debug registers S:16 through S:21 are inoperative. When set, the program can operate in the Single Step Test mode. See descriptions of S:16 through S:21. When set, your program requires 0.375 instruction words (3 bytes) per rung of additional memory. This bit is selected at the time you save your program.		•			
	TIP	This bit is not applicable to the SLC 5/03 and higher processors since its functionality is always available and requires no special compile time selection.			•	•	•

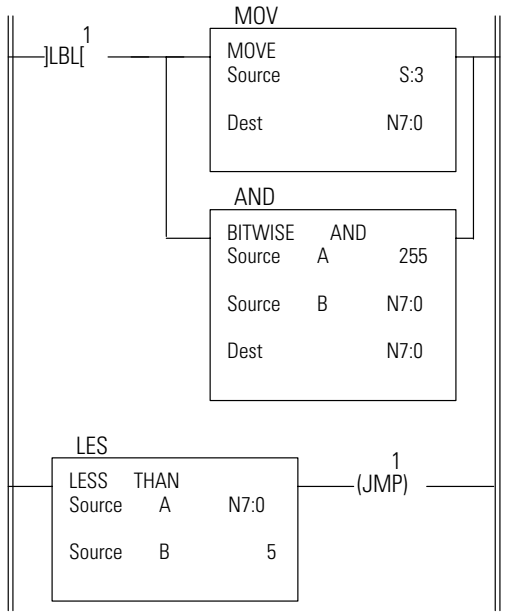
Address	Classification	Description	Fixed 5/01	5/02	5/03	5/04	5/05
S:2/5	**Status**	**Incoming Command Pending Bit (Channel 1)** This bit is set when the processor determines that another node on the network has requested information or supplied a command to it. This bit can be set at any time. This bit is cleared when the processor services the request (or command). Use this bit as a condition of an SVC instruction to enhance the communication capability of your processor.		•	•	•	•
S:2/6	**Status**	**Message Reply Pending Bit (Channel 1)** This bit is set when another node on the network has supplied the information you requested in the MSG instruction of your processor. This bit is cleared when the processor stores the information and updates your MSG instruction. Use this bit as a condition of an SVC instruction to enhance the communication capability of your processor.		•	•	•	•
S:2/7	**Status**	**Outgoing Message Command Pending Bit (Channel 1)** This bit is set when one or more messages in your program are enabled and waiting, but no message is being transmitted at the time. As soon as transmission of a message begins, the bit is cleared. After transmission, the bit is set again if there are further messages waiting. It remains cleared if there are no further messages waiting. Use this bit as a condition of an SVC instruction to enhance the communication capability of your processor.		•	•	•	•
S:2/8	**Dynamic Config**	**CIF (Common Interface File) Addressing Mode** This bit controls the mode used to address elements in the CIF file (data file 9) when processing a communication request. Word address mode - in effect when the bit is clear (0): This is the default setting, compatible with other SLC 500 devices on the DH-485 network. Byte address mode - in effect when the bit is set (1): This mode is used when the processor is receiving a message from a device on the network, possibly through a bridge or gateway. This setting is compatible with Allen-Bradley PLC inter-processor communication.		•	•	•	•
S:2/9	**Static Config**	**Memory Module Program Compare** When this bit is set inside a valid program that is contained in a memory module, no modification of the NVRAM user program files is allowed. This includes online editing, program downloading, and clear memory commands. Use this feature to prevent a programming device from altering the NVRAM program from the program contained in the Memory Module. If a memory module is installed with this bit set, and a different NVRAM user program is contained in NVRAM, the processor will not enter the Run mode. You must transfer the memory module program to NVRAM in order to enter the Run mode.			•	•	•
S:2/10	**Static Config**	**STI Resolution Selection (1 ms or 10 ms) Bit** This bit is cleared by default. When clear, this bit uses a 10 ms timebase for the STI Setpoint (S:30) value. For example, the value 4 uses a 40 ms STI setpoint. When set, this bit uses a 1 ms timebase for the STI Setpoint (S:30). For example, the value 4 uses a 4 ms STI setpoint. To program this feature, use the Data Monitor function to set, clear, or address this bit with your ladder program.			•	•	•

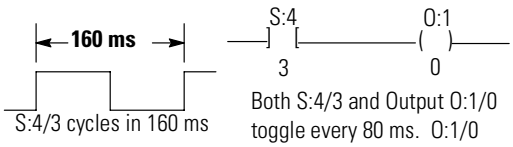
Address	Classification	Description	Fixed 5/01	5/02	5/03	5/04	5/05
S:2/11	**Status**	**Discrete Input Interrupt Pending Bit** When set, this bit indicates that the DII accumulator (S:52) equals the DII preset (S:50) and the ladder file number specified by the DII file number (S:46) is waiting to be executed. It is cleared when the DII file number (S:46) begins executing, or on exit of the REM Run or REM Test mode.			•	•	•
S:2/12	**Dynamic Config**	**Discrete Input Interrupt Enabled Bit** To program this feature, use the Data Monitor function to set, clear, or address this bit with your ladder program. This bit is set in its default condition. If set, it allows execution of the DII Subroutine if the DII file (S:46) is non-zero. If clear, when the interrupt occurs, the DII subroutine does not execute and the DII Pending bit is set. The DII function continues to run anytime the DII file (S:46) is non-zero. If the pending bit is set, the enable bit is examined at the next end of scan.			•	•	•
S:2/13	**Status**	**Discrete Input Interrupt Executing Bit** When set, this bit indicates that the DII interrupt has occurred and the DII subroutine is currently being executed. This bit is cleared on completion of the DII routine, power up, or REM Run mode entry. **Application example:** You can examine this bit in your fault routine to determine if your DII was executing when the fault occurred.			•	•	•

Address	Classification	Description	Fixed 5/01	5/02	5/03	5/04	5/05
S:2/14	**Dynamic Config**	**Math Overflow Selection Bit** Set this bit when you intend to use 32-bit addition and subtraction. When S:2/14 is set, and the result of an ADD, SUB, MUL, or DIV instruction cannot be represented in the destination address (underflow or overflow), • the overflow bit S:0/1 is set, • the overflow trap bit S:5/0 is set, and • the destination address contains the unsigned truncated least significant 16 bits of the result The default condition of S:2/14 is reset (0). When S:2/14 is reset, and the result of an ADD, SUB, MUL, or DIV instruction cannot be represented in the destination address (underflow or overflow), • the overflow bit S:0/1 is set, • the overflow trap bit S:5/0 is set, and • the destination address contains 32767 if the result is positive or - 32768 if the result is negative.		•	•	•	•
	TIP 	The status of bit S:2/14 has no effect on the DDV instruction. Also, it has no effect on the math register content when using MUL and DIV instructions.					
		To program this feature, use the Data Monitor function to set or clear this bit. To provide protection from inadvertent data monitor alteration of your selection, program an unconditional OTL instruction at address S:2/14 to ensure the new math overflow operation. Program an unconditional OTU instruction at address S:2/14 to ensure the original math overflow operation. See page 4-7 in this manual for an application example of 32-bit signed math.					

Address	Classification	Description	Fixed 5/01	5/02	5/03	5/04	5/05
S:2/15	**Dynamic Config**	**Communications Servicing Selection Bit (Ethernet Channel 1 for SLC 5/05) (DH+Channel 1 for SLC 5/04) (DH485 Channel 1 for SLC 5/03)** When set, only one communication request/command can be serviced per END, TND, REF, or SVC. When clear, all serviceable incoming or outgoing communication requests/commands can be serviced per END, TND, REF, or SVC. When clear, communication throughput increases. However, your scan time will increase if several communication requests/commands are received in the same scan. One communication request/command consists of either an incoming command, a message reply, or an outgoing message command. See S:2/5, S:2/6, and S:2/7 and S:33/7. To program this feature, use the Data Monitor function to set or clear this bit. To provide protection from inadvertent data monitor alteration of your selection, program an unconditional OTL instruction at address S:2/15 to ensure one request/command operation, or program an unconditional OTU instruction at address S:2/15 to ensure multiple request/command operation. Alternately, your program may change the state of this bit using ladder logic if your application requires dynamic selection of this function. **Application example:** Suppose you have a system consisting of an SLC 5/03 processor, programming software, and a DTAM. The program scan time for your user program is extremely long. Because of this, the programming device or DTAM takes an unusually long time to update its screen. Improve this update time by clearing S:2/15. In a case such as this, the additional time spent by the processor to service all communication at the end of the scan is insignificant compared to the time it takes to complete one scan. You could increase communication throughput even further by using an SVC instruction. See page 12-2 in this manual for more information.		•	•	•	•

Address	Classification	Description	Fixed 5/01	5/02	5/03	5/04	5/05
S:3L	**Status**	**Current/Last 10 ms Scan Time** The value of this byte tells you how much time elapses in a program cycle. A program cycle includes: • scanning the ladder program, • housekeeping, • scanning the I/O, and • servicing of the communication port. The byte value is zeroed by the processor each scan, immediately preceding the execution of rung 0 of program file 2 (main program file) or on return from the REF instruction. The byte is incremented every 10 ms thereafter, and indicates, in 10 ms increments, the amount of time elapsed in each program cycle. If this value ever equals the value in S:3H Watchdog, a user watchdog major error will be declared (code 0022). The resolution of the scan time value is +0 to −10 ms. Example: The value 9 indicates that 80-90 ms has elapsed since the start of the program cycle.	•	•			
	TIP 	When SVC or REF instructions are contained in your program, this value will appear to be erratic when you monitor it with a programming device. This is because the SVC or REF instructions allow this value to be read in mid-scan, while it is still incrementing.		•	•	•	•

Address	Classification	Description	Fixed 5/01	5/02	5/03	5/04	5/05
S:3L continued		**Application example:** Your application requires that each and every program scan execute in the same length of time. You measure the maximum and minimum scan times and find them to be 40 ms and 20 ms. You can make every scan equal to precisely 50 ms by programming the following rungs as the last rungs of your program.  This example assumes that your I/O scan and communication servicing takes less than 10 ms. If it exceeds 10 ms, the resolution of +0 to -1 tick (10 ms) must be added to the scan time.					
S:3H	**Dynamic Config**	**Watchdog Scan Time Byte** This byte value contains the number of 10 ms ticks allowed to occur during a program cycle. The default value is 10 (100 ms), but you can increase this to 250 (2.5 seconds) or decrease it to 2, as your application requires. If the program scan S:3L value equals the watchdog value, a watchdog major error will be declared (code 0022). This value is applied each END, TND, or REF.	•	•	•	•	•

Address	Classification	Description	Fixed 5/01	5/02	5/03	5/04	5/05
S:4	**Status**	**Free Running Clock** Only the first 8 bits (byte value) of this word are assessed by the processor. This value is zeroed at powerup in the REM Run mode. You can use any individual bit of this byte in your user program as a 50% duty cycle clock bit. Clock rates for S:4/0 to S:4/7 are: 20, 40, 80, 160, 320, 640, 1280, and 2560 ms. The application using the bit must be evaluated at a rate more than two times faster than the clock rate of the bit. This is illustrated in the following example for SLC 5/02 and higher processors.	•				
	Dynamic Config	All 16 bits of this word are assessed by the processor. The value of this word is zeroed upon power up in the REM Run mode or entry into the REM Run or REM Test mode. It is incremented every 10 ms thereafter.		•	•	•	•
	TIP 	You can write any value to S:4. It will begin incrementing from this value.					
		You can use any individual bit of this word in your user program as a 50% duty cycle clock bit. Clock rates for S:4/0 to S:4/15 are: 20, 40, 80, 160, 320, 640,1280, 2560, 5120, 10240, 20480, 40960, 81920, 163840, 327680, and 655360 ms The application using the bit must be evaluated at a rate more than two times faster than the clock rate of the bit. In the following example, bit S:4/3 toggles every 80 ms, producing a 160 ms clock rate. To maintain accuracy of this bit in your application, the instruction using bit S:4/3 (O:1/0 in this case) must be evaluated at least once every 79.999 ms.  Both S:4/3 and Output O:1/0 toggle every 80 ms. O:1/0 must be evaluated at least					
S:5		**Minor Error Bits** The bits of this word are set by the processor to indicate that a minor error has occurred in your ladder program. Minor errors, bits 0 to 7, revert to major error 0020H if any bit is detected as being set at the end of the scan. HHT users: If the processor faults for error code 0020H, you must clear minor error bits S:5/0-7 along with S:1/13 to attempt error recovery.	•	•	•	•	•

Address	Classification	Description	Fixed 5/01	5/02	5/03	5/04	5/05
S:5/0	**Dynamic Config**	**Overflow Trap Bit** When this bit is set by the processor, it indicates that a mathematical overflow has occurred in the ladder program. See S:0/1 for more information. If this bit is ever set upon execution of the END, TND, or REF instruction, major error (0020) will be declared. To avoid this type of major error from occurring, examine the state of this bit following a math instruction (ADD, SUB, MUL, DIV, DDV, NEG, SCL, TOD, or FRD), take appropriate action, and then clear bit S:5/0 using an OTU instruction with S:5/0 or a CLR instruction with S:5.	•	•	•	•	•
S:5/1	**NA**	**Reserved**	•	•	•	•	•
S:5/2	**Dynamic Config**	**Control Register Error Bit** The LFU, LFL, FFU, FFL, BSL, BSR, SQO, SQC, and SQL instructions are capable of generating this error. When bit S:5/2 is set, it indicates that the error bit of the control instruction has been set. If this bit is ever set upon execution of the END, TND, or REF instruction, major error (0020) will be declared. To avoid this type of major error from occurring, examine the state of this bit following a control register instruction, take appropriate action, and then clear bit S:5/2 using an OTU instruction with S:5/2 or a CLR instruction with S:5.	•	•	•	•	•
S:5/3	**Dynamic Config**	**Major Error Detected while Executing User Fault Routine Bit** When set, the major error code (S:6) represents the major error that occurred while processing the fault routine due to another major error. If this bit is ever set upon execution of the END, TND, or REF instruction, major error (0020) will be declared. To avoid this type of major error from occurring, examine the state of this bit inside your fault routine, take appropriate action, and then clear bit S:5/3 using an OTU instruction with S:5/3 or a CLR instruction with S:5. **Application example:** Suppose you are executing your fault routine for fault code 0016H Startup Protection. At rung 3 inside this fault routine, a TON containing a negative preset is executed. When rung 4 is executed, fault code 0016H is overwritten to indicate code 0034H, and S:5/3 is set. If your fault routine did not determine that S:5/3 was set, major error 0020H would be declared at the end of the first scan. To avoid this problem, examine S:5/3, followed by S:6, prior to returning from your fault routine. If S:5/3 is set, take appropriate action to remedy the fault, then clear S:5/3.		•	•	•	•

Address	Classification	Description	Fixed 5/01	5/02	5/03	5/04	5/05
S:5/4	**Dynamic Config**	**M0-M1 Referenced on Disabled Slot Bit** This bit is set whenever any instruction references an M0 or M1 module file element for a slot that is disabled (via its I/O slot enable bit). When set, the bit indicates that an instruction could not execute properly due to the unavailability of the addressed M0 or M1 data. If this bit is ever set upon execution of the END, TND, or REF instruction, major error (0020) is declared. To avoid this type of major error from occurring, examine the state of this bit following a M0-M1 referenced instruction, take appropriate action, and then clear bit S:5/4 using an OTU instruction with S:5/4 or a CLR instruction with S:5.		•	•	•	•
S:5/5 to S:5/7	**NA**	**Reserved** Reserved for minor errors that revert to major errors at the end of the scan.	•	•	•	•	•
S:5/8	**Status**	**Memory Module Boot Bit** When this bit is set by the processor, it indicates that a memory module program has been transferred to the processor. This bit is not cleared by the processor. Your program can examine the state of this bit on entry into the REM Run mode to determine if the memory module content has been transferred. Bit S:1/15 will be set to indicate REM Run mode entry. This information is useful when you have an application that contains retentive data and a memory module that has only bit S:1/10 set (Load Memory Module on Memory error). Use this bit to indicate that retentive data has been lost. This bit is also helpful when using bits S:1/11 (Load Memory Module Always) or S:1/12 (Load Memory Module Always and Run) to distinguish a power up REM Run mode entry from a REM Program (or REM Test) mode to REM Run mode entry.	•	•	•	•	•
S:5/9	**Status**	**Memory Module Password Mismatch Bit** This bit is set on REM Run mode entry, whenever loading from the memory module is specified (word 1, bits 11 or 12) and the processor user program is password protected, and the memory module program does not match that password. Use this bit to inform your application program that an autoloading memory module is installed but did not load due to a password mismatch.	•	•	•	•	•
S:5/10	**Status**	**STI (Selectable Timed Interrupt) Overflow Bit** This bit is set whenever the STI timer expires while the STI routine is either executing or disabled and the pending bit is already set.		•	•	•	•
S:5/11	**Status**	**Battery Low Bit** This bit is set whenever the Battery Low LED is on. The bit is cleared when the Battery Low LED is off.		•	•	•	•
S:5/12	**Status**	**Discrete Input Interrupt Overflow Bit** This bit is set whenever the DII interrupt occurs while still executing the DII subroutine or whenever the DII interrupt occurs while pending or disabled.			•	•	•

Address	Classification	Description	Fixed 5/01	5/02	5/03	5/04	5/05
S:5/13	**Dynamic Config**	**Unsuccessful Operating System Load Was Attempted** This bit is set whenever an operating system memory module load is attempted and is unsuccessful. Unsuccessful loads can occur when either the protection jumper is in the protect position or is missing, or if the operating system memory module is incompatible with the SLC 5/03, SLC 5/04, or SLC 5/05 processors' hardware platform. Examine the state of this bit with your user program to diagnose this condition.			•	•	•
S:5/14	**Status**	**Channel 0 Modem Lost** This bit indicates the status of the modem connected to Channel 0 (RS232 serial port). The state of the bit is determined by: • the protocol Channel 0 is configured for • the Control Line selected • the states of DCD (Data Carrier Detect) and DSR (Data Set Ready) If the bit is set, then the modem is not properly connected to Channel 0 or it is in a state where unreliable communication exchanges may take place via Channel 0. The following conditions apply: • If Channel 0 is disabled or configured for DH485, the bit is always cleared. • If Channel 0 is configured for one of the DF1 protocols in System Mode or Generic ASCII in User Mode, then the Control Line selection determines how DCD and DSR affect the modem status: - If Control Line = NO HANDSHAKING: The bit is always set. - If Control Line = FULL-DUPLEX or HALF-DUPLEX WITHOUT CONTINUOUS CARRIER: The bit is set if DSR goes inactive and cleared when DSR goes active. (DCD has no affect on modem status in this case.) - If Control Line = HALF-DUPLEX WITH CONTINUOUS CARRIER: The bit is set if either DSR goes inactive or DCD remains inactive for more than 10 seconds. This bit is cleared when both DSR and DCD go active.			•	•	•
S:5/15	**Status**	**ASCII String Manipulation Error** This bit applies to SLC 5/03 (OS301 and higher), SLC 5/04, and SLC 5/05 processors. This bit is set to 1 when an attempt is made to process a string using an ASCII instruction that exceeds 82 characters in length.			•	•	•

<table>
<tr><th>Address</th><th>Classification</th><th>Description</th><th>Fixed 5/01</th><th>5/02</th><th>5/03</th><th>5/04</th><th>5/05</th></tr>
<tr><td rowspan="4">S:6</td><td rowspan="4">Status</td><td>Major Error Fault Code
A hexadecimal code is entered in this word by the processor when a major error is declared. Refer to S:1/13. The code defines the type of fault, as indicated on the following pages. This word is not cleared by the processor.
Error codes are presented, stored, and displayed in a hexadecimal format. Refer to Appendix F for more information on the hexadecimal numbering system.</td><td>•</td><td>•</td><td>•</td><td>•</td><td>•</td></tr>
<tr><td>If you enter a fault code as a parameter in an instruction in your ladder program, you must convert the code to decimal. For example, if you program an EQU instruction to go true when the error 0016 occurs, enter S:6 as source A and 22, the decimal equivalent of 0016H, as source B:
EQU
EQUAL
Source A S:6
Source B 22</td><td></td><td>•</td><td>•</td><td>•</td><td>•</td></tr>
<tr><td>Application example: You can declare your own application specific major fault by writing a unique value to S:6 and then setting bit S:1/13.
SLC 5/02 processor users: Interrogate the value of S:6 in your fault routine to determine the type of fault that occurred. If your program was saved with the test single step enabled, you can also interrogate S:20 and S:21 to pinpoint the exact rung that was executing when the fault occurred.
Fault Classifications: Faults are classified as Non-User, Non-Recoverable, and Recoverable.
<table>
<tr><th>Non-User Fault</th><th>Non-Recoverable User Fault</th><th>Recoverable User Fault</th></tr>
<tr><td>The fault routine does not execute.</td><td>The fault routine executes for 1 pass. (You may initiate a MSG instruction to another node to identify the fault condition of the processor.)</td><td>The fault routine may clear the fault by clearing bit S:1/13.</td></tr>
</table></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>Error code descriptions and classifications are listed on pages B-25 through B-31. Categories are:
• powerup errors
• going-to-run errors
• runtime errors
• user program instruction errors
• I/O errors
See Chapter 15 of this manual for fault cause and recovery information.</td><td>•</td><td>•</td><td>•</td><td>•</td><td>•</td></tr>
</table>

Address	Error Code (Hex)	Errors	Fault Classification			Processor				
			Non-User	User		Fixed 5/01	5/02	5/03	5/04	5/05
				Non-Recov	Recov					
S:6 continued	**0001**	NVRAM error.	**X**			•	•	•	•	•
	0002	Unexpected hardware watchdog timeout.	**X**			•	•	•	•	•
	0003	Memory module memory error. This error can also occur while going into the REM Run mode.	**X**				•	•	•	•
	0004	Memory error occurred while in the Run mode.	**X**				•	•	•	•
	0005	Reserved			**X**			•	•	•
	0006	Reserved			**X**			•	•	•
	0007	Failure during memory module transfer.	**X**					•	•	•
	0008	Internal software error.	**X**					•	•	•
	0009	Internal hardware error.	**X**					•	•	•
	0010	The Processor does not meet the required revision level.	**X**			•	•	•	•	•
	0011	The executable program file number 2 is absent.	**X**			•	•	•	•	•
	0012	The ladder program has a memory error.	**X**			•	•	•	•	•
	0013	• The required memory module is absent or • S:1/10 or S:1/11 is not set as required by the program.			**X**	•	•	•	•	•
	0014	Internal file error.	**X**			•	•	•	•	•
	0015	Configuration file error.	**X**			•	•	•	•	•
	0016	Startup protection after power loss. Error condition exists at powerup when bit S:1/9 is set and powerdown occurred while running.			**X**		•	•	•	•
	0017	NVRAM/memory module user program mismatch.		**X**				•	•	•
	0018	Incompatible user program - Operating system type mismatch. This error can also occur during powerup.	**X**					•	•	•
	0019	Missing or duplicate label was detected.		**X**				•	•	•
	001F	A program integrity problem occurred during an online editing session.	**X**					•	•	•

Address	Error Code (Hex)	Errors	Fault Classification			Processor				
			Non-User	User		Fixed 5/01	5/02	5/03	5/04	5/05
				Non-Recov	Recov					
S:6 continued	**0020**	A minor error bit is set at the end of the scan. Refer to S:5 minor error bits.			**X**	•	•	•	•	•
	0021	Remote power failure of an expansion I/O chassis occurred.	**X**			•	•	•	•	•
		TIP  A modular system that encounters an over-voltage or over-current condition in any of its power supplies can produce any of the I/O error codes listed on pages -29 and -30 (instead of code 0021). The over-voltage or over-current condition is indicated by the power supply LED being off.								
		ATTENTION  Fixed and FRN 1 to 4 SLC 5/01 processors - If the remote power failure occurred while the processor was in the REM Run mode, error 0021 will cause the major error halted bit (S:1/13) to be cleared at the next powerup of the local chassis. SLC 5/02 processors and FRN 5 SLC 5/01 processors - Power to the local chassis does not need to be cycled to resume the REM Run mode. Once the remote chassis is re-powered, the CPU will restart the system.								
	0022	The user watchdog scan time has been exceeded.		**X**		•	•	•	•	•
	0023	Invalid or non-existent STI interrupt file.		**X**			•	•	•	•
	0024	Invalid STI interrupt interval (greater than 2559 ms or negative).		**X**			•	•	•	•
	0025	Excessive stack depth/JSR calls for STI routine.		**X**			•	•	•	•
	0026	Excessive stack depth/JSR calls for I/O interrupt routine.		**X**			•	•	•	•
	0027	Excessive stack depth/JSR calls for user fault routine.		**X**			•	•	•	•
	0028	Invalid or non-existent "startup protection" fault routine file value.	**X**				•	•	•	•
	0029	Indexed address reference outside of entire data file space (range of B3:0 through the last file).			**X**		•			
		ATTENTION  The SLC 5/02 processor uses an index value of zero for the faulted instruction following error recovery.								
	002A	Indexed address reference is beyond specific referenced data file.		**X**			•	•	•	•

Address	Error Code (Hex)	Errors	Fault Classification			Processor				
			Non-User	User		Fixed 5/01	5/02	5/03	5/04	5/05
				Non-Recov	Recov					
S:6 continued	**002B**	The file number exists, but it is not the correct file type or the file number does not exist.			**X**			•	•	•
	002C	The indirectly referenced element does not exist, but the file type is correct and it exists. For example, T4:[N7:0] N7:0=10, but T4 only goes to T4:9.			**X**			•	•	•
	002D	Either a subelement is referenced incorrectly or an indirect reference has been made to an M-file.			**X**			•	•	•
	002E	Invalid DII Input slot.			**X**			•	•	•
	002F	Invalid or non-existent DII interrupt file.		**X**				•	•	•

I/O Errors

ERROR CODES: The characters xx in the following codes represent the slot number, in hexadecimal. If the exact slot cannot be determined, the characters xx become 1F.

RECOVERABLE I/O FAULTS (SLC 5/02, SLC 5/03, SLC 5/04, and SLC 5/05 processors only): Many I/O faults are recoverable. To recover, you must disable the specified slot, xx, in the user fault routine. If you do not disable slot xx, the processor will fault at the end of the scan.

TIP

An I/O card that is severely damaged may cause the processor to indicate that an error exists in slot 1 even though the damaged card is installed in a slot other than 1.

Slot	xx (2)	Slot	xx	Slot	xx	Slot	xx
0	00	8	08	16	10	24	18
1	01	9	09	17	11	25	19
2	02	10	0A	18	12	26	1A
3(1)	03	11	0B	19	13	27	1B
4	04	12	0C	20	14	28	1C
5	05	13	0D	21	15	29	1D
6	06	14	0E	22	16	30	1E
7	07	15	0F	23	17	(3)	1F

(1) This value indicates that the slot was not found (500 fixed controller).

(2) Slot Numbers (xx) in hexadecimal

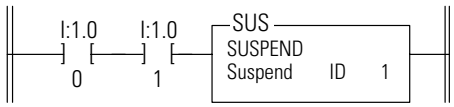
(3) This value indicates that the slot was not found (SLC 5/01, SLC 5/02, SLC 5/03, SLC 5/04, and SLC 5/05 processors).

Address	Error Code (Hex)	User Program Instruction Errors	Fault Classification			Processor				
			Non-User	User		Fixed 5/01	5/02	5/03	5/04	5/05
				Non-Recov	Recov					
S:6 continued	**0030**	Attempt was made to jump to one too many nested subroutine files. This code can also mean that a program has potentially recursive routines.		**X**		•	•	•	•	•
	0031	An unsupported instruction reference was detected.		**X**		•	•	•	•	•
	0032	A sequencer length/position parameter points past the end of a data file.			**X**	•	•	•	•	•
	0033	The length of LFU, LFL, FFU, FFL, BSL, or BSR instruction points past the end of a data file.			**X**	•	•	•	•	•
	0034	A negative value for a timer accumulator or preset value was detected.			**X**	•	•	•	•	•
		Fixed processors with 24 VDC inputs only: A negative or zero HSC preset was detected in a HSC instruction.			**X**	•				
	0035	TND, SVC, or REF instruction is called within an interrupting or user fault routine.		**X**			•	•	•	•
	0036	An invalid value is being used for a PID instruction parameter.			**X**		•	•	•	•

Address	Error Code (Hex)	User Program Instruction Errors	Fault Classification			Processor				
			Non-User	User		Fixed 5/01	5/02	5/03	5/04	5/05
				Non-Recov	Recov					
S:6 continued	**0038**	A RET instruction was detected in a non-subroutine file.	**X**			•	•	•	•	•
	xx39	Invalid string length was detected in a string file. (xx = data file number)			**X**			•	•	•
	003A	An attempted write to a constant data file.						•	•	•
	003B	Motherboard and Daughter Card firmware do not match.								•
	003C	STI watchdog time-out.						•	•	•
	005F	Invalid Rack ID						•	•	•
	xx50	A chassis data error is detected. (xx = slot number)			**X**	•	•	•	•	•
	xx51	A "stuck" runtime error is detected on an I/O module. (xx = 31)		**X**		•	•	•	•	•
	xx52	A module required for the user program is detected as missing or removed. (xx = slot number)			**X**	•	•	•	•	•
	xx53	When going-to-run, a user program declares a slot as unused, and that slot is detected as having an I/O module inserted. This can also mean that an I/O module has reset itself. (xx = slot number)			**X**	•	•	•	•	•
		An attempt to enter the run or test mode was made with an empty chassis.			**X**			•	•	•
	xx54	A module required for the user program is detected as being the wrong type. (xx = slot number)			**X**	•	•	•	•	•
	xx55	A discrete I/O module required for the user program is detected as having the wrong I/O count. This code can also mean that a specialty card driver is incorrect. (xx = slot number)			**X**	•	•	•	•	•
	xx56	The chassis configuration specified in the user program is detected as being incorrect.	**X**			•	•	•	•	•

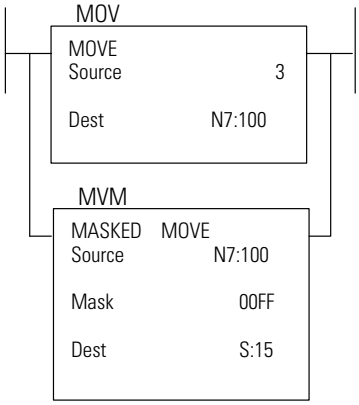
Address	Error Code (Hex)	User Program Instruction Errors	Fault Classification			Processor				
			Non-User	User		Fixed 5/01	5/02	5/03	5/04	5/05
				Non-Recov	Recov					
S:6 continued	**xx57**	A specialty I/O module has not responded to a lock shared memory command within the required time limit. (xx = slot number)			**X**	•	•	•	•	•
	xx58	A specialty I/O module has generated a generic fault. The card fault bit is set (1) in the module's status byte. (xx = slot number)		**X**		•	•	•	•	•
	xx59	A specialty I/O module has not responded to a command as being completed within the required time limit. (xx = slot number)			**X**	•	•	•	•	•
	xx5A	Hardware interrupt problem. (xx = slot number)		**X**			•	•	•	•
	xx5B	G file configuration error - user program G file size exceeds capacity of the module. (xx = slot number)			**X**		•	•	•	•
	xx5C	M0-M1 file configuration error - user program M0-M1 file size exceeds capacity of the module. (xx = slot number)			**X**		•	•	•	•
	xx5D	Interrupt service requested is not supported by the processor. (xx = slot number)			**X**		•	•	•	•
	xx5E	Processor I/O driver (software) error. (xx = slot number)			**X**		•	•	•	•
	xx60 to xx6F	Identifies an I/O module specific recoverable major error. Refer to the user manual supplied with the specialty module. (xx = slot number)			**X**		•	•	•	•
	xx70 to xx7F	Identifies an I/O module specific non-recoverable major error. Refer to the user manual supplied with the specialty module. (xx = slot number)		**X**			•	•	•	•
	xx80 to xx8F	Identifies a specialty I/O module specific major error. Refer to the user manual supplied with the specialty module. (xx = slot number)	**X**					•	•	•

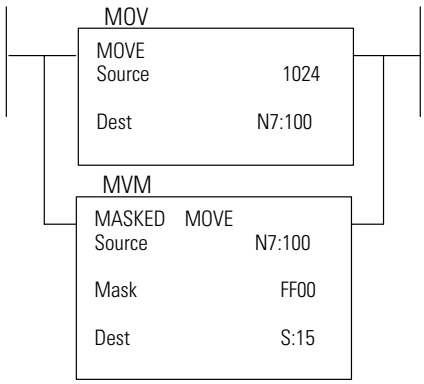
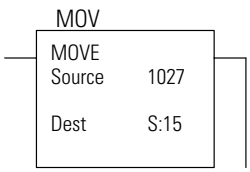
Address	Error Code (Hex)	User Program Instruction Errors	Fault Classification			Processor				
			Non-User	User		Fixed 5/01	5/02	5/03	5/04	5/05
				Non-Recov	Recov					
S:6 continued	**xx90**	Interrupt problem on disabled slot.		**X**			•	•	•	•
	xx91	A disabled slot has faulted.		**X**			•	•	•	•
	xx92	An invalid or non-existent module interrupt subroutine (ISR) file.		**X**			•	•	•	•
	xx93	Unsupported I/O module specific major error.		**X**			•	•	•	•
	xx94	In the REM Run or REM Test mode, a module has been detected as being inserted under power. This can also mean that an I/O module has reset itself. (xx = slot number)		**X**			•	•	•	•
	0x00A0 0x00A1 0x00A2	Indicates a communications channel hardware fault has occurred. With the SLC 5/05, only the Ethernet channel (channel 1) may generate this fault. With the SLC 5/04 only the DH+ channel (channel 1) may generate this fault. Ethernet, DH+ and RS-232 communications will be disabled until a power cycle is performed. For the SLC 5/05, system status file word 15 (S:15) provides a specific fault code for the Ethernet Daughterboard when user fault code 0x00A1 is generated.	**X**						•	•

Address	Classification	Description	Fixed 5/01	5/02	5/03	5/04	5/05
S:7 and S:8	**Status**	**Suspend Code/Suspend File** When a non-zero value appears in S:7, it indicates that the SUS instruction identified by this value has been evaluated as true, and the Suspend Idle mode is in effect. This pinpoints the conditions in the application that caused the Suspend Idle mode. This value is not cleared by the processor. Word S:8 contains the program file number in which a true SUS instruction is located. This value is not cleared by the processor. Use the SUS instruction with startup troubleshooting, or as runtime diagnostics for detection of system errors. **Application example:** You believe that limit switches connected to I:1/0 and I:1/1 cannot be energized at the same time, yet your application program acts as if they can be. To determine if you have a limit switch problem or a ladder logic problem, add the following rung to your program:  If your program enters the SUS Idle mode for code 1 when you run the program, you have a limit switch control problem; if the SUS Idle mode for code 1 does not occur, you have a ladder logic problem.	•	•	•	•	•
S:9	**Status**	**Ethernet Daughter Board Firmware Series (Channel 1 - SLC 5/05 processors)** A value of 1 corresponds to Series 1, for example.					•
S:10	**Status**	**Ethernet Daughter Board Firmware Revision (Channel 1 - SLC 5/05 processors)** A value of 12 corresponds to Revision 12, for example.					•
S:9 and S:10	**Status**	**Active Nodes (Channel 1-SLC 5/03 processors)** These two words are bit mapped to represent the 32 possible nodes on a DH-485 link. S:9/0 through S:10/15 represent node addresses 0-31. These bits are set by the processor when a node exists on the DH-485 link that your processor is connected to. The bits are cleared when a node is not present on the link.	•	•	•		

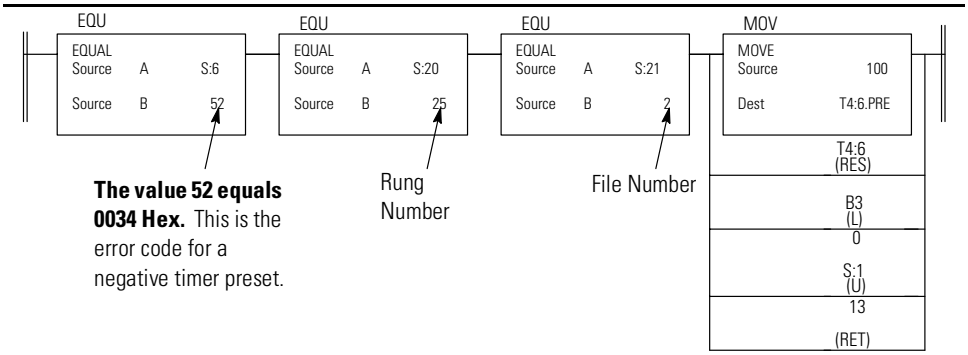
Address	Classification	Description	Fixed 5/01	5/02	5/03	5/04	5/05
S:11 and S:12	**Dynamic Config**	**I/O Slot Enables** These two words are bit mapped to represent the 30 possible I/O slots in an SLC 500 system. S:11/0 represents I/O slot 0 for fixed I/O systems. (Slot 0 is used for the CPU in modular systems.) S:11/1 through S:12/14 represent I/O slots 1-30. S:12/15 is unused. When a bit is set (default condition), it allows the I/O module contained in the reference slot to be updated in the I/O scan of the processor operating cycle. When you clear a bit, it causes the I/O module in the referenced slot to be ignored. That is, an I/O slot enable value of 0 causes the input image data of an input module to freeze at its last value. Also, the outputs of an output module will freeze at their last values, regardless of values contained in the output image. Outputs remain frozen until: • either power is removed, • the REM Run mode is exited, or • a major fault occurs. At that time the outputs are zeroed, until the slot is again enabled (set). Disabled slots do not have to match the user program configuration.	•	•	•	•	•
	ATTENTION 	Make certain that you have thoroughly examined the effects of disabling (clearing) a slot enable bit before doing so in your application.					
	TIP 	The SLC 5/02 and higher processors inform each specialty I/O module that has been disabled/enabled. Some I/O modules may perform other actions when disabled or re-enabled. Refer to the user information supplied with the specialty I/O module for possible differences from the above descriptions.		•	•	•	•
	ATTENTION 	The DII instruction ignores the slot enable/disable status. Do not run the DII on a faulted slot. If you apply the DII on a disabled slot, the interrupt will occur. However, the input image will not reflect the present state of the card. This bit is applied upon detection of a DII Reconfigure bit, each DII ISR exit, and at each end of scan (END, TND, or REF).			•	•	•

Address	Classification	Description	Fixed 5/01	5/02	5/03	5/04	5/05
S:13 and S:14	**Status and Dynamic Config**	**Math Register** Use this double register to produce 32-bit signed divide and multiply operations, precision divide or double divide operations, and 5-digit BCD conversions. These two words are used in conjunction with the MUL, DIV, DDV, FRD, and TOD math instructions. The math register value is assessed upon execution of the instruction and remains valid until the next MUL, DIV, DDV, FRD, or TOD instruction is executed in the user program. An explanation of how the math register operates is included with the instruction definitions. If you store 32-bit signed data values, you must manage this data type without the aid of an assigned 32-bit data type. For example, combine B10:0 and B10:1 to create a 32-bit signed data value. We recommend that you keep all 32-bit signed data in a unique data file and that you start all 32-bit values on an even or odd word boundary for ease of application and viewing. Also, we recommend that you design, document, and view the contents of 32-bit signed data in either the hexadecimal or binary radix. See Chapter 4 for more information on how the math register is effected by each instruction.	•	•	•	•	•
		When an STI, I/O Slot, or Fault Routine interrupts normal execution of your program, the original value of the math register is restored when execution resumes. Note that S:13 and S:14 are not used when the source or destination is defined as floating point data.		•	•	•	•
		When a DII interrupts normal execution of your program, the original value of the math register is restored when execution resumes.			•	•	•

Address	Classification	Description	Fixed 5/01	5/02	5/03	5/04	5/05
S:15L	**Static Config**	**Node Address** This byte value contains the node address of your processor on the DH-485 or DH+ link. Each device on the DH-485 link must have a unique address between the decimal values 0 and 31. Each device on the DH+ link must have a unique address between the decimal values 0 and 63. To change a processor node address, write a value between 1 and 31 for DH-485 and between 0 and 63 for DH+ using either the Data Monitor or node function of your programmer, then cycle power to the processor. The default node address of a processor is 1. The default DH-485 node address of HHT programmer is 0. To provide runtime protection from inadvertent data monitor alteration of your selection, program this value using an unconditional MVM instruction. Use the MOV instruction in place of MVM if you also wish to protect the baud rate. The following example show runtime protection of node address 3  	•	•	•	•	

Address	Classification	Description	Fixed 5/01	5/02	5/03	5/04	5/05
S:15H continued	**Static Config**	When a configure channel command is received for channel 1, the node address is overwritten with the value contained in your channel configuration.			•	•	
		Baud Rate This byte value contains a code used to select the baud rate of the processor on the DH-485 or DH+ link. SLC 5/01 and fixed processors provide a baud rate of 19.2K or 9.6K only. SLC 5/02 and SLC 5/03 processors provide a baud rate of 19.2K, 9.6K, 2.4K, or 1.2K. SLC 5/04 processors provide a baud rate of 57.6K, 115.2K, and 230.4K. To change the baud rate from the default values of 19.2K or 57.6K, use either the Data Monitor or baud function of your programmer. The processor uses code 1 for 1.2K, code 2 for 2.4K, code 3 for 9.6K, code 4 for 19.2 K, code 11 for 57.6K, code 12 for 115.2K, and code 13 for 230.4K baud. Example showing runtime protection of baud rate 19.2K (code 4):  S:15H equal to 4 = 1024 decimal = 0400 hex = 0000 0100 0000 0000 binary	•	•	•	•	
	Static Config	Example showing runtime protection for both baud rate 19200 (code 4) and node address 3:  S:15H equal to 4 and S:15L equal to 3 = 1027 decimal = 0403 hex = 0000 0100 0000 0011 binary					
		When a configure channel command is received for channel 1, the baud rate is overwritten with the value contained in your channel configuration.			•	•	
	Static Config	**Ethernet Daughter Board Fault Code** This word value contains the fault code when User Fault Code 0x00A0 or 0xYYA0 occurs.					•

Address	Classification	Description	Fixed 5/01	5/02	5/03	5/04	5/05
S:16 and S:17	**Status**	**Test Single Step - Start Step On - Rung/File** These registers indicate the executable rung (word S:16) and file (word S:17) number that the processor executes next when operating in the Test Single Step mode. To enable this feature, you must select the Test Single Step option at the time you save your program. These values are updated upon completion of every rung. Refer to word S:2/4 for more information. Your programming device interrogates this value when providing "start step on file x, rung y" status line information. There is no known use for this feature when addressed by your ladder program.		•	•	•	•
		This feature is built into the SLC 5/03 and higher processors. Selection is not required.			•	•	•
S:18 and S:19	**Status and Dynamic Config**	**Test Single Step - Breakpoint - Rung/File** These registers indicate the executable rung (word S:18) and file (word S:19) number that the processor should stop in front of when executing in the Test Single Step mode. To enable this feature, you must select the Test Single Step option at the time you save your program. If both the rung and file number are 0, the processor steps to the next rung only; otherwise the processor continues until it finds a rung/file equaling the S:18/S:19 value. The processor stops, then clears S:18 and S:19 when it finds a match, while remaining in the Test Single Step mode. The processor operates indefinitely if it cannot find the end rung/file that you have entered. It operates until it finds a match, receives a mode change, or powers down. See S:2/4. Your programming device interrogates this value when providing "end step before file x, rung y" status line information. Your programming device also writes this value when prompting you for "set end rung." There is no known use for this feature when addressed by your ladder program.		•	•	•	•
		This feature is built into the SLC 5/03 and higher processors. Selection is not required.			•	•	•

Address	Classification	Description	Fixed 5/01	5/02	5/03	5/04	5/05
S:20 and S:21	**Status**	**Test - Fault/Powerdown - Rung/File** These registers indicate the executable rung (word S:20) and file (word S:21) number that the processor last executed before a major error or powerdown occurred. To enable this feature, you must select the Test Single Step option at the time you save your program. You can use these registers to pinpoint the execution point of the processor at the last powerdown or fault routine entry. This function is also active in the REM Run mode. See S:2/4. **Application example:** Suppose your program contains several TON instructions. TON T4:6 in file 2, rung 25 occasionally obtains a negative preset. Recovery from the negative preset fault is possible by placing the preset at 100 and resetting the timer. Place the following rung in your fault routine to accomplish this. Bit B3/0 is latched as evidence that an application recovery has been initiated.		•	•	•	•
		This feature is built into the SLC 5/03 and higher processors. Selection is not required.			•	•	•
							

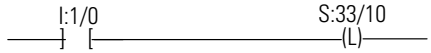
Address	Classification	Description	Fixed 5/01	5/02	5/03	5/04	5/05
S:22	**Status**	**Maximum Observed Scan Time** This word indicates the maximum observed interval between consecutive scans. Consecutive scans are defined as intervals between file 2/rung 0 and the END, TND, or the REF instruction. This value indicates, in 10 ms increments, the time elapsed in the longest program cycle of the processor. The processor compares each last scan value to the value contained in S:22. If the processor determines that the last scan value is larger than the value stored at S:22, the last scan value is written to S:22. Resolution of the maximum observed scan time value is -0 to -10 ms. For example, the value 9 indicates that 80-90 ms was observed as the longest program cycle. Interrogate this value using the Data Monitor function if you need to determine or verify the longest scan time of your program.		•	•	•	•
		The Scan Time Selection Bit (S:33/13) determines the timebase used for average and maximum Scan Times. When clear, operation is as described above. When set, the timebase is expressed in 1 ms increments (instead of 10 ms increments). When S:33/13 is set, resolution of the maximum observed scan time value is -0 to -1 ms. For example, the value 9 indicates that 8 to 9 ms was observed as the largest program cycle.			•	•	•
S:23	**Status**	**Average Scan Time** This word indicates a weighted running average time. The value indicates, in 10 ms increments, the time elapsed in the average program cycle of the processor. For every Scan t: $$\mathbf{Avg} = \frac{\mathbf{(Avg * 7) + Scan_t}}{\mathbf{8}}$$ Resolution of the average scan time value is -0 to -10 ms. For example, the value 2 indicates that 10 to 20 ms was calculated as the average program cycle.		•	•	•	•
		The Scan Time Selection bit S:33/13 determines the timebase used for average Scan time. When clear, operation is as described above. When set, the timebase is expressed in 1 ms increments (instead of 10 ms increments). When S:33/13 is set, resolution of the average scan time value is +0 to -1 ms. For example: the value 2 indicates that 1 to 2 ms was calculated as the average program cycle.			•	•	•
S:24	**Dynamic Config**	**Index Register** This word indicates the element offset used in indexed addressing. When an STI, I/O Slot, or Fault Routine interrupts normal execution of your program, the original value of this register is restored when execution resumes.		•	•	•	•
		When a DII interrupts normal execution of your program, the original value of this register is restored when execution resumes.			•	•	•

Address	Classification	Description	Fixed 5/01	5/02	5/03	5/04	5/05
S:25 and S:26	**Status**	**I/O Interrupt Pending** These two words are bit-mapped to the 30 I/O slots. Bits S:25/1 through S:26/14 refer to slots 1 through 30. Bits S:25/0 and S:26/15 are reserved. The pending bit associated with an interrupting slot is set when the corresponding I/O Slot Interrupt Enable bit is clear at the time of an interrupt request. It is cleared when the corresponding I/O Event Interrupt Enable bit is set, or when an associated RPI instruction is executed. The pending bit for an executing I/O interrupt subroutine remains clear when the ISR is interrupted by an STI or fault routine. Likewise, the pending bit remains clear if interrupt service is requested at the time that a higher or equal priority interrupt is executing (fault routine, STI, or other ISR). I/O interrupts are discussed in Chapter 11 of this manual.		•	•	•	•
		The pending bit associated with an interrupting slot is set when the corresponding I/O Slot Interrupt Enable bit is clear at the time of an interrupt request. It is cleared when the corresponding I/O Event Interrupt Enable bit is set, or when an associated RPI instruction is executed. The pending bit is always set when interrupt service is requested and the processor is executing an interrupt of equal or higher priority. Interrupt priority does not affect the setting of these bits. For example, while executing an STI subroutine, slot 6 requests an I/O Event Interrupt. The STI executes to completion; however, slot 6 pending bit (S:25/6) becomes set within execution of the STI. Examine the state of these bits within your interrupt subroutines if your application requires this information.			•	•	•
S:27 and S:28	**Status**	**I/O Interrupt Enabled** These two words are bit-mapped to the 30 I/O slots. Bits S:27/1 through S:28/14 refer to slots 1 through 30. Bits S:27/0 and S:28/15 are reserved. The default value of each bit is 1 (set). The enable bit associated with an interrupting slot must be set when the interrupt occurs to allow the corresponding ISR to execute. Otherwise, the ISR does not execute and the associated I/O slot interrupt pending bit becomes set. Changes made to these bits using the Data Monitor function or ladder instructions other than IID or IIE take affect at the next end of scan. I/O interrupts are discussed in Chapter 11 of this manual.		•	•	•	•
	Dynamic Config	These bits may be set/reset by the user program, comms., or with the IIE or IID instruction. Changes made to these bits using a programming terminal's Data Monitor function or any ladder instruction take effect immediately.			•	•	•

Address	Classification	Description	Fixed 5/01	5/02	5/03	5/04	5/05
S:29	**Dynamic Config**	**User Fault Routine File Number** You enter a program file number (3 to 255) to be used in all recoverable and non-recoverable major errors. Program the ladder logic of your fault routine in the file you have specified. Write a zero value to disable the fault routine. To provide protection from inadvertent Data Monitor alteration of your selection, program an unconditional MOV instruction containing the program file number of your fault routine to S:29, or program a CLR instruction at S:29 to prevent fault routine operation. The fault routine is discussed in Chapter 11 of this manual.		•	•	•	•
S:30	**Dynamic Config**	**Selectable Timed Interrupt - Setpoint** You enter the timebase, in tens of milliseconds, to be used in the selectable timed interrupt. Your STI routine executes per the value you enter. Write a zero value to disable the STI. To provide protection from inadvertent Data Monitor alteration of your selection, program an unconditional MOV instruction containing the setpoint value of your STI to S:30, or program a CLR instruction at S:30 to prevent STI operation. If the STI is initiated while in the REM Run mode by loading the status registers, the interrupt starts timing from the end of the program scan in which the status registers were loaded. If the STI has been previously configured (with a different setpoint), the new setpoint takes effect only after the previously-configured STI has timed out. Selectable timed interrupts are discussed on page 11-8 of this manual.		•	•	•	•
		The STI Setpoint timebase can be either 10 ms or 1 ms depending on the value of the STI Setpoint Selection bit S:2/10. When clear, operation is as described above. When set, the timebase is expressed in 1 ms increments. The STE enables and STD disables the STI instruction.			•	•	•
S:31	**Dynamic Config**	**Selectable Timed Interrupt - File Number** You enter a program file number (3 to 255) to be used as the selectable timed interrupt subroutine. Write a zero value to disable the STI. To provide protection from inadvertent Data Monitor alteration of your selection, program an unconditional MOV instruction containing the file number value of your STI to S:31, or program a CLR instruction at S:31 to prevent STI operation. Selectable timed interrupts are discussed on page 11-8 of this manual.		•	•	•	•

Address	Classification	Description	Fixed 5/01	5/02	5/03	5/04	5/05
S:32	**Status**	**I/O Interrupt Executing** This word indicates the slot number of the specialty I/O module that generated the currently executing ISR. This value is cleared upon completion of the ISR, REM Run mode entry, or upon power-up. You can interrogate this word inside of your STI subroutine or fault routine if you wish to know if these higher priority interrupts have interrupted an executing ISR. You may also use this value to discern interrupt slot identity when multiplexing two or more specialty I/O module interrupts to the same ISR. I/O interrupts are discussed on page 11-29 of this manual.		•	•	•	•
		You can interrogate this word inside your DII subroutine if you wish to know if these higher priority interrupts have interrupted an executing ISR. You may also use this value to discern interrupt slot identity when multiplexing two or more specialty I/O module interrupts to the same ISR.			•	•	•
S:33/0	**Status**	**Incoming Command Pending (Channel 0)** This bit becomes set when the processor determines that another node on the channel 0 network has requested information or supplied a command to it. This bit can be set at any time. This bit is cleared when the processor services the request (or command). Use this bit as a condition of an SVC instruction to enhance the communication capability of your processor.			•	•	•
S:33/1	**Status**	**Message Reply Pending (Channel 0)** This bit becomes set when another node on the channel 0 network has supplied the information that you requested in the MSG instruction of your processor. This bit is cleared when the processor stores the information and updates your MSG instruction. Use this bit as a condition of an SVC instruction to enhance the communication capability of your processor.			•	•	•
S:33/2	**Status**	**Outgoing Message Command Pending (Channel 0)** This bit is set when one or more channel 0 messages in your program are enabled and waiting, but no message is being transmitted at the time. As soon as transmission of a message begins, the bit is cleared. After transmission, the bit is set again if there are further messages waiting, or it remains cleared if there are no further messages waiting.			•	•	•
S:33/3	**Status**	**Selection Status (Channel 0)** When set, this bit indicates that the channel 0 communication port is in the System mode (DF1 mode). When reset, this bit indicates that channel 0 is in the User mode (ASCII mode). Use your programming devices channel configuration utility to change this selection.			•	•	•
S:33/4	**Status**	**Communications Active (Channel 0)** *DH-485 protocol only*: This bit is set by the processor when at least one other node is active on channel 0 DH-485 network. Otherwise the bit remains cleared.			•	•	•

Address	Classification	Description	Fixed 5/01	5/02	5/03	5/04	5/05
S:33/5	**Dynamic Config**	**Communications Servicing Selection (Channel 0)** When set, only one channel 0 communication request/ command will be serviced per END, TND, REF, or SVC instruction. When clear, all serviceable incoming or outgoing communication requests/commands will be serviced per END, TND, REF, or SVC instruction. One communication request/command consists of either a channel 0 Incoming Command, channel 0 Message Reply, or channel 0 Outgoing Message Command. Refer to Words S:33/0, S:33/1, S:33/2, and S:33/6 for more information. **Note:** When clear, your communication throughput will increase. Your scan time will also increase if several communication commands/requests are received in the same scan. To program this feature, use the Data Monitor function to set and clear this bit. To provide protection from inadvertent data monitor alteration of your selection, program an unconditional OTL instruction at address S:33/5 to ensure one request/command operation, or an unconditional OTU instruction at address S:33/5 to ensure multiple request/command operation. Alternately, your program may change the state of this bit using ladder logic if your application requires dynamic selection of this function.			•	•	•
S:33/6	**Dynamic Config**	**Message Servicing Selection (Channel 0)** This bit is only valid when the channel 0 Comms Servicing Selection (S:33/5) is clear (which selects service all commands). When S:33/6 is clear and S:33/5 is clear, all outgoing channel 0 MSG instructions will be serviced per END, TND, SVC, or REF instruction. Otherwise, only one outgoing channel 0 MSG command or reply will be serviced per END, TND, SVC, or REF instruction.			•	•	•
S:33/7	**Dynamic Config**	**Message Servicing Selection (Channel 1)** This bit is only valid when the channel 1 Comms Servicing Selection bit (S:2/15) is clear (which selects service all commands). When S:33/7 is clear and S:2/15 is clear, all outgoing channel 1 MSG instructions are serviced per END, TND, SVC, or REF instruction. Otherwise, only one outgoing channel 1 MSG command or reply is serviced per END, TND, SVC, or REF instruction.			•	•	•

Address	Classification	Description	Fixed 5/01	5/02	5/03	5/04	5/05
S:33/8	**Static Config**	**Interrupt Latency Control Bit** When set, interrupt latency occurs for user interrupts (DII, STI, and I/O Event). This means that when an interrupt occurs, you are guaranteed to be at rung 0 of your interrupt subroutine within the stated interrupt latency period (assuming no interrupt of equal or higher priority is executing). You must select this at the time you save your program. When clear, user interrupts may only interrupt the processor at predefined points of execution in the user program cycle. Interrupt latency is then defined as the longest period of time that can occur between any two predefined points. When S:33/8 is clear, you must analyze each user program. The bit is clear by default. The following points are the only points in which user interrupt subroutines are allowed to execute when S:33/8 is clear: • at the start of each rung • following the servicing of communication • between slots when updating the input or output image, or any specialty I/O card			•	•	•
S:33/9	**Status**	**Scan Toggle Bit** This bit is cleared upon entry into the RUN mode. This bit changes state each and every execution of an END, TND, or REF instruction. Use this bit in your user program for applications such as multiplexing subroutine execution.			•	•	•
S:33/10	**Dynamic Config**	**Discrete Input Interrupt Reconfiguration Bit** Set this bit with your user program or programming terminal to cause the DII function to reconfigure itself at the next interrupt occurrence or end of each scan (END, TND, or REF). This bit is applied upon a DII ISR, fault routine, STI ISR, or Event ISR exit. The following occurs when the DII is reconfigured: 1. The DII Accumulator is cleared (S:52). 2. DII parameters located in words S:46 through S:50 are applied. 3. The DII reconfigure bit is cleared by the processor. For example, use the following ladder structure to cause a DII reconfiguration from your main ladder file each time input 0 is cycled on.  Use the following ladder structure to cause a DII reconfiguration from an event based subroutine. The subroutine is only executed once, each time the DII reconfiguration is possible. 			•	•	•

Address	Classification	Description	Fixed 5/01	5/02	5/03	5/04	5/05
S:33/11 and S:33/12	**Status**	**Online Edit Status** These two bits represent the four possible Online Edit states: **Bit 12 / Bit 11 / Online Edit Status** 0 / 0 / No online edits exist 0 / 1 / Online edits are disabled 1 / 1 / Testing online edits 1 / 0 / not used Examine the state of these bits with your user program to count the number of online edit sessions, flag an alarm, or place your application in a special state designed for online edit sessions.			•	•	•
S:33/13	**Static Config**	**Scan Time Timebase Selection** This bit determines the timebase used to average the Scan time (S:23) and the maximum Scan Time (S:22). When clear, the value contained in the average and maximum scan times represent the number of 10 ms increments that have occurred. When set, the value contained in the average and maximum scan times represent the number of 1 ms increments that have occurred. This value is clear by default (10 ms timebase).			•	•	•
S:33/14	**Dynamic Config**	**DTR Control Bit (Channel 0)** This bit is used to enable DTR dialing. When clear, the channel 0 DTR signal (pin 4) is directly controlled by the standard communication driver. When set, you can perform DTR dialing by writing to S:33/15, DTR Force Bit. Bit S:33/14 is examined and applied at each end of scan (END, TND, or REF). When in Program, Suspend, or Fault mode, DTR is enabled and remains enabled until an auto-disconnect sequence is detected by the communication driver. An auto-disconnect occurs if the communication driver detects that channel 0 CD signal (pin 1) has been absent for more than 10 seconds or if the channel 0 DSR signal (pin 6) has been disabled. Refer to S:5/14 Channel 0 Modem Lost bit for more information. During an auto-disconnect, the standard communication driver keeps the DTR disabled until either the channel 0 DSR signal is enabled, or 5 seconds elapse. **Note:** When channel 0 is configured for DH485, S:33/14 must be clear for proper operation.			•	•	•

Address	Classification	Description	Fixed 5/01	5/02	5/03	5/04	5/05
S:33/15	**Dynamic Config**	**DTR Force Bit (Channel 0)** This bit is used to force the DTR pin high or low. When S:33/14 is set, the channel 0 DTR signal (pin 4) is applied at each end of scan (END, TND, or REF) using the state of S:33/15. When S:33/14 is clear, this bit has no effect on DTR. When S:33/15 is set, DTR is forced high. When clear (default), DTR is forced low. When in the REM Test or REM Run mode, this bit is only applied at end of scan (END, TND, or REF). When in Program, Suspend, or Fault mode (or on power up), DTR is set unless the communication driver is performing an auto-disconnect.			•	•	•
S:34/0	**Dynamic Config**	**DH+ to DH-485 Passthru Disabled Bit** **Ethernet to DH-485 Passthru Disabled Bit** This bit provides the capability to pass received packets between channels. When set, the processor does not support passthru. When reset, the processor allows packets to be passed from one channel to the other. Channel 0 (RS-232) must be configured for DH485 protocol. Only packets that contain the Internet network layer remote MSG packets and whose Destination Link ID equals that specified for the opposite channel will be passed. The default is reset. The default Link ID for channel 0 is one. The default Link ID for channel 1 is two.				•	•
S:34/1	**Static Config**	**DH+ Active Node Table Enable Bit** This bit enables processing of the DH+ active node table. When set, the DH+ active node table is processed. When clear, the DH+ active node table is not processed. The default is clear. This bit is evaluated upon each entry into the REM Run mode. Note that the processor updates individual status words S:83 to S:86.				•	
S:34/2	**Dynamic Config**	**Floating Point Math Flag Disable Bit** This bit disables the processing of math flags when using floating point math (F8:). The math flags effected are Overflow (S:0/1), Zero (S:0/2), Sign (S:0/3), and the Minor Error Overflow Trap bit (S:5/0). When the bit is clear, the math flags are processed. When the bit is set, the math flags are cleared except for the Minor Error Overflow Trap bit which remains in its last state. The Carry Flag (S:0/0) is reserved for internal use during all floating point operations. The default is clear. Instructions effected by floating point include ADD, SUB, MUL, DIV, NEG, SQR, and MOV. Setting this bit reduces the execution times for the above instructions. This bit is evaluated when each instruction is executed.			•	•	•
S:34/3	**Dynamic Config**	**Global Status Word Transmit Enable Bit** When this bit is set, the Global Status Word at S:99 is transmitted with every DH+ token pass. When clear, the token is passed without the Global Status Word.				•	

Address	Classification	Description	Fixed 5/01	5/02	5/03	5/04	5/05
S:34/4	**Dynamic Config**	**Global Status Word Receive Enable Bit** When this bit is set, the processor collects the Global Status Word being transmitted by other devices on the DH+ network and stores them in the Global Status File (S:100-S:163). When clear, the processor ignores the Global Status information from other devices on the network.				•	
S:34/5	**Dynamic Config**	**DF1 to DH+ Passthru Enabled Bit** When this bit is set, passthru operation is enabled between Channel 0 and Channel 1. Channel 0 must be configured for DF1 full-duplex protocol. Only DH+ packets that contain the internet network layer and whose Destination Link ID equals that configured for channel 0 will be passed from channel 1 to channel 0. Only DF1 packets whose destination address (DST) is a valid DH+ address (0-63), and does not equal the DH+ address of this SLC 5/04 processor, will be passed from channel 0 to channel 1.				•	
		DF1 to Ethernet Passthru Enabled Bit When this bit is set, passthru is enabled. Channel 0 must be configured for DF1 full-duplex protocol. Only Ethernet packets that contain the internet network layer remote MSG packets are passed from channel 1 to channel 0. Only DF1 packets whose destination address (DST) is a valid number (1-128) corresponding to a valid IP address in the routing table are passed from channel 0 to channel 1. DF1 packets with a destination address equal to 0 are processed locally. The default is reset.					•
S:35	**Status**	**Last 1 ms Scan Time** The value of this word tells you how much time elapsed in a program cycle. A program cycle includes the ladder program, housekeeping, I/O scan, and servicing of the communication port. This word value is only updated by the processor once each scan, immediately preceding the execution of rung 0, file 2 (or upon return of a REF instruction).			•	•	•
S:36/0 to S:36/7	**NA**	**Reserved**			•	•	•
S:36/8	**Status**	**DII Lost** This bit is set anytime a DII interrupt occurs while the DII Pending bit (S:2/11) is also set. When set, you are notified that a DII interrupt has been lost. For example, the interrupt is lost because a previous interrupt was already pending and waiting execution. Examine this bit in your user program and take appropriate action if your application cannot tolerate this condition. Then clear this bit with your user program to prepare for the next possible occurrence of this error.			•	•	•

Address	Classification	Description	Fixed 5/01	5/02	5/03	5/04	5/05
S:36/9	**Status**	**STI Lost** This bit is set anytime an STI interrupt occurs while the STI Pending bit (S:2/0) is also set. When set, you are notified that a STI interrupt has been lost. For example, the interrupt is lost because a previous interrupt was already pending and waiting execution. Examine this bit in your user program and take appropriate action if your application cannot tolerate this condition. Then clear this bit with your user program to prepare for the next possible occurrence of this error.			•	•	•
S:36/10	**Status**	**Memory Module Data File Overwrite Protection** Use this bit to determine the validity of retentive data following a memory module transfer. This bit is always set when a memory module to processor transfer occurs with Data File Overwrite Protection selected and protected files are overwritten. Protected files are overwritten anytime a memory module program does not match the processor program at the time of the transfer. This bit is not cleared by the processor.			•	•	•
S:36/11 to S:36/15	**NA**	**Reserved for additional minor errors.**			•	•	•
S:37	**Dynamic Config**	**Clock/Calendar Year** This value contains the year value of the clock/calendar. Valid range is 0-65535. To disable the clock/calendar, write zeros to all clock/calendar words (S:37 to S:42).			•	•	•
S:38	**Dynamic Config**	**Clock/Calendar Month** This value contains the month value of the clock/ calendar. Valid range is 1-12. To disable the clock/calendar, write zeros to all clock or calendar words (S:37 to S:42). January equals the value of 1.			•	•	•
S:39	**Dynamic Config**	**Clock/Calendar Day** This value contains the day value of the clock/calendar. Valid range is 1-31. To disable the clock/calendar, write zeros to all clock or calendar words (S:37 to S:42). The first day of the month equals the value of 1. See status word S:53 for Day-of-Week.			•	•	•
S:40	**Dynamic Config**	**Clock/Calendar Hours** This value contains the hour value of the clock/calendar. Valid range is 0-23. To disable the clock/calendar, write zeros to all clock or calendar words (S:37 to S:42). 0000 hundred hours equals the value of 0.			•	•	•
S:41	**Dynamic Config**	**Clock/Calendar Minutes** This value contains the minute value of the clock/calendar. Valid range is 0-59. To disable the clock/calendar, write zeros to all clock or calendar words (S:37 to S:42).			•	•	•
S:42	**Dynamic Config**	**Clock/Calendar Seconds** This value contains the seconds value of the clock/calendar. Valid range is 0-59. To disable the clock/calendar, write zeros to all clock or calendar words (S:37 to S:42).			•	•	•

Address	Classification	Description	Fixed 5/01	5/02	5/03	5/04	5/05
S:43 **S:44** **S:45**	**Status**	**Selectable Timed Interrupt - 10 µs Timer** **I/O Event Interrupt - 10 µs Timer** **Discrete Input Interrupt - 10 µs Timer** This 16-bit value is "free running" and is used to measure the amount of time that expires between consecutive interrupt subroutine executions (in increments of 10 µs). This value is updated upon each entry into the interrupt subroutine. The 10 µs timer dictates that the maximum amount of time that can expire between any two interrupts and still result in a valid time measurement is 0.32767 seconds. (16-bit signed _ 10 µs = 32767 _.00001 = 0.32767 seconds) The 10 µs timer is common to the STI interrupt, the Event I/O interrupt, and the DII interrupt.			•	•	•
S:46	**Dynamic Config**	**Discrete Input Interrupt - File Number** You enter a program file number (3-255) to be used as the discrete input interrupt subroutine. Write a zero value to disable the function. This value is applied upon detection of a DII Reconfigure bit, each DII ISR exit, and each end of scan (END, TND, or REF). To provide protection from inadvertent data monitor alteration of your selection, program an unconditional MOV instruction containing the file number value of your DII to S:46 or program a CLR instruction at S:46 to prevent DII operation.			•	•	•
S:47	**Dynamic Config**	**Discrete Input Interrupt - Slot Number** You enter the slot number (1-30) that contains the Discrete I/O module to be used as the discrete input interrupt slot. The processor will fault if the slot is empty or contains a non-discrete I/O module. For example, an analog module causes a processor fault to occur. This bit is applied upon detection of the DII Reconfigure bit. This value is only applied upon execution of the DII reconfiguration function (setting bit S:33/10 or upon REM Run mode entry with the DII Enable bit S:2/12 set). To provide protection from inadvertent data monitor alteration of your selection, program an unconditional MOV instruction containing the slot number value of your DII to S:47.			•	•	•
S:48	**Dynamic Config**	**Discrete Input Interrupt - Bit Mask** You enter a bit mapped value that corresponds to the bits that you wish to monitor on the discrete I/O module. Only bits 0 to 7 are used in the DII function. Setting a bit indicates that you wish to include the bit in the comparison of the discrete I/O module's bit transition to the DII Compare Value (S:49). Clearing a bit indicates that the transition state of that particular bit is a "don't care" bit. This value is applied upon detection of a DII Reconfigure bit, each DII ISR exit, and at each end of scan (END, TND, or REF). To provide protection from inadvertent data monitor alteration of your selection, program an unconditional MOV instruction containing the bit mask value of your DII to S:48.			•	•	•

Address	Classification	Description	Fixed 5/01	5/02	5/03	5/04	5/05
S:49	**Dynamic Config**	**Discrete Input Interrupt - Compare Value** You enter a bit mapped value that corresponds to the bit transitions that must occur in the discrete I/O card for a count or interrupt to occur. Only bits 0 to 7 are used in the DII function. Setting a bit indicates that the bit must transition from a 0 to a 1 to satisfy the compare condition for that bit. Clearing a bit indicates that the bit must transition from a 1 to a 0 in order to satisfy the compare condition for that bit. An interrupt or count will be generated upon the last bit transition of the compare value. This value is applied upon detection of a DII Reconfigure bit, each DII ISR exit, and each end of scan (END, TND, or REF). To provide protection from inadvertent data monitor alteration of your selection, program an unconditional MOV instruction containing the compare value of your DII to S:49.			•	•	•
S:50	**Dynamic Config**	**Discrete Input Interrupt - Preset** When this value is equal to 0 or 1, an interrupt is generated each time the comparison specified in words S:48 and S:49 is satisfied. When this value is between 2 and 32767, a count will occur each time the bit comparison is satisfied. An interrupt will be generated when the accumulator value reaches 1 or exceeds the preset value. This value is applied on detection of DII Reconfigure bit, each DII ISR exit, and at each end of scan (END, TND, or REF). To provide protection from inadvertent data monitor alteration of your selection, program an unconditional MOV instruction containing the preset value of your DII to S:50.			•	•	•
S:51	**Status**	**Discrete Input Interrupt - Return Mask** The return mask is updated immediately preceding entry into the DII subroutine. This value contains the bit map of the bit transitions that caused the interrupt. The bit is set if it was included in the list of bit transitions that caused the interrupt, (specified to transition in the S:48 and S:49 comparisons). The bit is cleared if it was masked. This value is cleared by the processor upon exit of the DII subroutine. Use this value to validate the interrupt transitions. Or when dynamically reconfiguring (sequencing) the DII, you can use this value inside your DII's subroutine to help determine or validate its position in the sequence.			•	•	•
S:52	**Status**	**Discrete Input Interrupt - Accumulator** The DII accumulator contains the number of counts that have occurred (see S:50.) When a count occurs, and the accumulator is greater than or equal to the preset value, a DII interrupt is generated.			•	•	•
S:53L	**Dynamic Config**	**Day-of-Week** This value contains the day-of-week value of the clock/calendar. Valid range is 0-6 (Sunday=0). To disable the clock/calendar, write zeros to all clock and calendar words (S:37 to S:42).			•	•	•
S:53H	**NA**	**Reserved**			•	•	•
S:54	**Status**	**Last Major Error** Mirrors the error code value stored in S:6.			•	•	•

Address	Classification	Description	Fixed 5/01	5/02	5/03	5/04	5/05
S:55	**Status**	**Last Discrete Input Interrupt Scan Time** This value indicates, in 1 ms increments, the amount of time elapsed by the most recent DII subroutine. The resolution of this value is +0 to -1 ms.			•	•	•
S:56	**Status**	**Maximum Observed Discrete Input Scan Time** This value indicates, in 1 ms increments, the maximum amount of time elapsed by any single DII subroutine execution. The processor compares each last DII scan value (S:55) to the maximum DII scan value contained in S:56. If the processor determines that the last DII scan value is larger than the value stored at S:56, the last scan value (S:55) is written to S:56, thus becoming the new maximum DII scan time. The resolution of this value is +0 to -1 ms. Interrogate this value using a programming device Data Monitor function if you need to determine or verify the longest scan time of your program.			•	•	•
S:57	**Status**	**Operating System Catalog Number** Indicates the operating system catalog number. For example, the value of 300 indicates operating system -OS300, the value of 301 indicates -OS301.			•	•	•
S:58	**Status**	**Operating System Series** Indicates the operating system series. For example, the value of 0 indicates series A and the value of 1 indicates series B.			•	•	•
S:59	**Status**	**Operating System FRN** Indicates the operating system firmware release number. For example, the value of 1 indicates FRN1 and the value of 2 indicates FRN2.			•	•	•
S:60	**Status**	**Processor Catalog Number** Indicates the catalog number of the processor. For example, the value of 532 indicates -L532 and the value of 534 indicates -L534.			•	•	•
S:61	**Status**	**Processor Series** Indicates the processor series. For example, the value of 0 indicates series A and the value of 1 indicates series B.					
S:62	**Status**	**Processor Revision** Indicates the processor revision. For example, the value of 1 indicates REV1 and the value of 2 indicates REV2.			•	•	•
S:63	**Status**	**User Program Type** Indicates the programming device that created the user program.			•	•	•
S:64	**Status**	**User Program Functionality Index** Indicates the level of functionality contained in a given program type.			•	•	•
S:65	**Status**	**User RAM Size** Applies to SLC 5/03 (OS302), SLC 5/04 (OS401), and SLC 5/05 processors. Indicates the size of NVRAM in instruction words. For example, the value 64 equals 64K instruction words of NVRAM.			•	•	•

Address	Classification	Description	Fixed 5/01	5/02	5/03	5/04	5/05
S:66	**Status**	**Flash EEPROM Size** Indicates the size of operating system memory in 16 bit K words. For example, the value of 128 equals 128K words of memory.			•	•	•
S:67 to S:82	**Status**	**Channel 0 Active Node Table** Only used when channel 0 is configured for DF1 half-duplex master or DH-485. These 16 words are bit-mapped to represent the 255 possible nodes in a DF1 half-duplex network or 32 possible nodes in a DH-485 network. Bit S:67/0 represents node 0, bit S:82/14 represents node 254, and bit S:82/15 is not used since address 255 is reserved for broadcasts. In DF1 half-duplex configuration, these bits are set by the processor, which is configured for either of the two "standard" polling modes, when a node is polled (because it appears in either the normal or priority polling ranges) and it responds to the master poll. A node's bit is cleared if no response is received after a master poll. In DH-485 configuration, these bits are set by the processor when a node exists on the DH-485 link that your processor is connected to. These bits are cleared when a node is not present on the link.			•	•	•
S:83 to S:86	**Status**	**Channel 1 Active Node Table** These 4 words are bit mapped to represent the 64 possible nodes on a DH+ link. S:83/0 through S:86/15 represent node addresses 0-63 (0-77 octal). These bits are set by the processor when a node exists on the DH+ link that your processor is connected to. These bits are cleared when a node is not present on the link. Note that S:34/1 must be set for the above words to work.				•	
S:87 to S:98	**NA**	**Reserved**				•	
S:99	**Dynamic Config**	**Global Status Word** Data placed in this memory location is transmitted as the processor's Global Status Word and is sent to all other devices on the DH+ network every time the processor passes the DH+ token.				•	
S:100 to S:163	**Static Config**	**Global Status File** When a processor passes the DH+ token to the next node, it also sends a 16-bit word called the Global Status Word (S:99 and above). All of the nodes on the network read the Global Status Word sent by each processor and saves the word to memory. Each processor has a table (Global Status File) in memory where global status words from other processors are stored. This table is completely updated every token rotation. (Example: The word from node "x" is placed at S:100 + x.) You can use the Global Status File as a high-speed broadcast message for status passing and synchronization of processors.				•	

The following table lists all combination settings for S:1/10, S:1/11 and S:1/12.

S:1/10	S:1/11	S:1/12	Memory Module Present?	SLC Memory Error?	Memory Module Transfer?	Mode Before/After Powerdown	Major Fault Before/After Powerdown
0	0	0	don't care	N/A	no	don't care/same	don't care/same
1	0	0	no	N/A	no	(REM) PROG/(REM) PROG	dont' care/same
						REM RUN/faulted REM PROG	no fault/faulted
						RUN/faulted RUN	
			yes	no	no	don't care/same	don't care/same
			yes	yes	yes	faulted PROG/PROG	faulted/no fault
						faulted REM PROG/REM RUN	
						faulted RUN/RUN	
0	1	0	no	N/A	no	(REM) PROG/(REM) PROG	don't care/same
						REM RUN/faulted REM PROG	no fault/faulted
						RUN/faulted RUN	
			yes	N/A	yes	don't care/same	don't care/no fault
0	0	1	no	N/A	no	(REM) PROG/(REM) PROG	don't care/same
						(REM) RUN/(REM) RUN	
			yes	N/A	yes	PROG/PROG	don't care/no fault
						REM PROG/REM RUN	
						(REM) RUN/(REM) RUN	
0	1	1	no	N/A	no	(REM) PROG/(REM) PROG	don't care/same
						REM RUN/faulted REM PROG	no fault/faulted
						RUN/faulted RUN	
			yes	N/A	yes	PROG/PROG	don't care/no fault
						REM PROG/REM RUN	
						(REM) RUN/(REM) RUN	

Memory Usage

This appendix provides:

- instruction words for the Fixed, SLC 5/01, SLC 5/02, SLC 5/03, SLC 5/04, and SLC 5/05 processor
- examples on how to estimate the total memory usage of your system

Memory Usage Overview

SLC 500 controllers have the following user memory capacities:

Type of Processor	User Memory Capacity
Fixed and SLC 5/01	1,024 instruction words
SLC 5/02	4,096 instruction words
SLC 5/03 8k	4,096 words
SLC 5/03, SLC 5/04, SLC 5/05 16k	12,288 words(1)
SLC 5/04, SLC 5/05 32k SLC 5/05, SLC 5/05 64k	28,672 words(1) 61,440 words(1)

(1) When your ladder program is larger than 12k words, you must split your program into two files. A main (file 2) and at least one subroutine firl (3 to 255) is required.

The following definitions apply when figuring your memory usage:

- fixed, SLC 5/01, and SLC 5/02: 1 instruction word = 4 data words = 8 bytes
- SLC 5/03, SLC 5/04, and SLC 5/05: 1 instruction word = 1 data word

Fixed and SLC 5/01 Processors

The number of words used by an instruction is indicated in the following table. Since the program is compiled by the programmer, it is only possible to establish *estimates* for the instruction words used by individual instructions. The calculated memory usage will normally be greater than the actual memory usage, due to compiler optimization.

Table C.1 SLC 500 Fixed and SLC 5/01 List of Instructions

Mnemonic	Memory Usage (user words)	Name	Instruction Type	Page
ADD	1.5	Add	Math	4-5
AND	1.5	And	Data Handling	5-20
BSL	2.00	Bit Shift Left	Application Specific	7-4
BSR	2.00	Bit Shift Right	Application Specific	7-4
CLR	1.00	Clear	Math	4-12
COP	1.50	File Copy	Data Handling	5-12
CTD	1.00	Count Down	Basic	2-14
CTU	1.00	Count Up	Basic	2-13
DCD	1.50	Decode 4 to 1 of 16	Data Handling	5-10
DDV	1.00	Double Divide	Math	4-11
DIV	1.50	Divide	Math	4-9
EQU	1.50	Equal	Comparison	3-2
FLL	1.50	Fill File	Data Handling	5-12
FRD	1.00	Convert from BCD	Data Handling	5-5
GEQ	1.50	Greater Than or Equal	Comparison	3-4
GRT	1.50	Greater Than	Comparison	3-3
HSC	1.00	High-Speed Counter	High-Speed Counter	2-15
IIM	1.50	Immediate Input with Mask	Program Flow Control	6-8
IOM	1.50	Immediate Output with Mask	Program Flow Control	6-9
JMP	1.00	Jump to Label	Program Flow Control	6-2
JSR	1.00	Jump to Subroutine	Program Flow Control	6-3
LBL	0.50	Label	Program Flow Control	6-2
LEQ	1.50	Less Than or Equal	Comparison	3-3
LES	1.50	Less Than	Comparison	3-3
MCR	0.50	Master Control Reset	Program Flow Control	6-6
MEQ	1.50	Masked Compare for Equal	Comparison	3-4
MOV	1.50	Move	Data Handling	5-17
MUL	1.50	Multiply	Math	4-8

Table C.1 SLC 500 Fixed and SLC 5/01 List of Instructions

Mnemonic	Memory Usage (user words)	Name	Instruction Type	Page
MVM	1.50	Masked Move	Data Handling	5-18
NEG	1.50	Negate	Data Handling	5-24
NEQ	1.50	Not Equal	Comparison	3-2
NOT	1.00	Not	Data Handling	5-23
OR	1.50	Or	Data Handling	5-21
OSR	1.00	One-Shot Rising	Basic	2-5
OTE	0.75	Output Energize	Basic	2-4
OTL	0.75	Output Latch	Basic	2-4
OTU	0.75	Output Unlatch	Basic	2-4
RES	1.00	Reset	Basic	2-20
RET	0.50	Return from Subroutine	Program Flow Control	6-3
RTO	1.00	Retentive Timer	Basic	2-11
SBR	0.50	Subroutine	Program Flow Control	6-3
SQC	2.00	Sequencer Compare	Application Specific	7-6
SQO	2.00	Sequencer Output	Application Specific	7-6
SUB	1.50	Subtract	Math	4-5
SUS	1.50	Suspend	Program Flow Control	6-8
TND	0.50	Temporary End	Program Flow Control	6-7
TOD	1.00	Convert to BCD	Data Handling	5-2
TOF	1.00	Timer Off-Delay	Basic	2-10
TON	1.00	Timer On-Delay	Basic	2-9
XIC	1.00	Examine If Closed	Basic	2-3
XIO	1.00	Examine If Open	Basic	2-3
XOR	1.50	Exclusive Or	Data Handling	5-22

Estimating Total Memory Usage of Your System Using a Fixed or SLC 5/01 Processor

_____ **1.** Calculate the total instruction words used by the instructions in your program and enter the result. Refer to the table on page C-2.

_____ **2.** Multiply the total number of rungs by .375 and enter the result.

_____ **3.** Multiply the total number of data words (excluding the status field and I/O data words) by .25 and enter the result.

_____ **4.** Add 1 word for each data table file and enter the result

_____ **5.** Multiply the highest numbered program file used by 2 and enter the result.

_____ **6.** Multiply the total number of I/O data words by .75 and enter the result.

_____ **7.** Multiply the total number of I/O slots, used or unused, by .75 and enter the result.

_____ **8.** To account for processor overhead, enter 65 if you are using a fixed controller, enter 67 if you are using a 1747-L511 or 1747-L514.

Total: _____ **9.** Total steps 1 through 8. This is the estimated total memory usage of your application system. Remember, this is an estimate, actual compiled programs may differ by ±12%.

10. If you wish to determine the estimated amount of memory remaining in the processor you have selected, do the following:
If you are using a fixed controller or 1747-L511, subtract the total from 1024. If you are using a 1747-L514, subtract the total from 4096.
The result of this calculation will be the estimated total memory remaining in your selected processor.

TIP The calculated memory usage may vary from the actual compiled program by ±12%.

Fixed Controller Memory Usage Example

L20B Fixed I/O Controller				
42	XIC and XIO	42 x 1.00	=	42.00
10	OTE instructions	10 x 0.75	=	7.50
10	TON instructions	10 x 1.00	=	10.00
1	CTU instruction	1 x 1.00	=	1.00
1	RES instruction	1 x 1.00	=	1.00
Instruction Usage				61.50
21	rungs	21 x .375	=	7.87
37	data words	37 x .250	=	9.25
User Program Total				**78.62**
2	I/O data words	2 x 0.75	=	1.50
1	slot	1 x 0.75	=	0.75
Overhead				65.00
I/O Configuration Total				**67.25**
Estimated total memory usage:				145.87
				(round to 146)

1024 - 146 = 878 instruction words remaining in the processor

SLC 5/01 Processor Memory Usage Example

1747-L514 processor, 30-slot configuration, (15) 1746-IA16, (10) 1746-OA8, (1) 1747-DCM full configuration, (1) 1746-NI4, (1) 1746-NIO4I				
50	XIC and XIO	50 x 1.00	=	50.00
15	OTE instructions	15 x 0.75	=	11.25
5	TON instructions	5 x 1.00	=	5.00
3	GRT instructions	3 x 1.50	=	4.50
1	SCL instruction	1 x 1.75	=	1.75
1	TOD instruction	1 x 1.00	=	1.00
3	MOV instructions	3 x 1.50	=	4.50
10	CTU instructions	10 x 1.00	=	10.00
10	RES instructions	10 x 1.00	=	10.00
Instruction Usage				98.00
30	rungs	30 x .375	=	11.25
100	data words	100 x .250	=	25.00
10	is highest data table file number	10 x 1.00	=	10.00
4	is highest program file number	4 x 2.00	=	8.00
User Program Total				**163.50**
49	I/O data words	49 x 0.75	=	36.75
30	slot	30 x 0.75	=	22.50
Overhead				67.00
I/O Configuration Total				**126.25**
Estimated total memory usage:				289.75
				(round to 290)

4096 - 290 = 3806 instruction words remaining in the processor

SLC 5/02 Processor

The number of instruction words used by an instruction is indicated in the following table. Since the program is compiled by the programmer, it is only possible to establish *estimates* for the instruction words used by individual instructions. The calculated memory usage will normally be greater than the actual memory usage, due to compiler optimization.

Table C.2 SLC 5/02 List of Instructions

Mnemonic	Memory Usage (user words)	Name	Instruction Type	Page
ADD	1.5	Add	Math	4-5
AND	1.5	And	Data Handling	5-20
BSL	2.00	Bit Shift Left	Application Specific	7-4
BSR	2.00	Bit Shift Right	Application Specific	7-4
CLR	1.00	Clear	Math	4-12
COP	1.50	File Copy	Data Handling	5-12
CTD	1.00	Count Down	Basic	2-14
CTU	1.00	Count Up	Basic	2-13
DCD	1.50	Decode 4 to 1 of 16	Data Handling	5-10
DDV	1.00	Double Divide	Math	4-11
DIV	1.50	Divide	Math	4-9
EQU[1]	1.50	Equal	Comparison	3-2
FFL	1.50	FIFO Load	Data Handling	5-26
FFU	1.50	FIFO Unload	Data Handling	5-26
FLL	1.50	Fill File	Data Handling	5-12
FRD	1.00	Convert from BCD	Data Handling	5-5
GEQ[1]	1.50	Greater Than or Equal	Comparison	3-4
GRT[1]	1.50	Greater Than	Comparison	3-3
IID	1.25	I/O Interrupt Disable	Interrupt	11-34
IIE	1.25	I/O Interrupt Enable	Interrupt	11-34
IIM	1.50	Immediate Input with Mask	Program Flow Control	6-8
INT	0.50	Interrupt Subroutine	Interrupt	11-36
IOM	1.50	Immediate Output with Mask	Program Flow Control	6-9
JMP	1.00	Jump to Label	Program Flow Control	6-2
JSR	1.00	Jump to Subroutine	Program Flow Control	6-3
LBL	0.50	Label	Program Flow Control	6-2
LEQ[1]	1.50	Less Than or Equal	Comparison	3-3
LES[1]	1.50	Less Than	Comparison	3-3

Table C.2 SLC 5/02 List of Instructions

Mnemonic	Memory Usage (user words)	Name	Instruction Type	Page
LIM	1.50	Limit Test	Comparison	3-4
LFL	1.50	LIFO Load	Data Handling	5-28
LFU	1.50	LIFO Unload	Data Handling	5-28
MCR	0.50	Master Control Reset	Program Flow Control	6-6
MEQ(1)	1.50	Masked Comparison for Equal	Comparison	3-4
MOV	1.50	Move	Data Handling	5-17
MSG	34.75	Message	Communication	12-3
MUL	1.50	Multiply	Math	4-8
MVM	1.50	Masked Move	Data Handling	5-18
NEG	1.50	Negate	Data Handling	5-24
NEQ(1)	1.50	Not Equal	Comparison	3-2
NOT	1.00	Not	Data Handling	5-23
OR	1.50	Or	Data Handling	5-21
OSR	1.00	One-Shot Rising	Basic	2-5
OTE	0.75	Output Energize	Basic	2-4
OTL	0.75	Output Latch	Basic	2-4
OTU	0.75	Output Unlatch	Basic	2-4
PID	23.25	Proportional Derivative	PID	9-2
REF	0.50	Refresh	Program Flow Control	6-10
RES	1.00	Reset	Basic	2-20
RET	0.50	Return from Subroutine	Program Flow Control	6-3
RPI	1.25	Reset Pending Interrupt	Interrupt	11-36
RTO	1.00	Retentive Timer	Basic	2-11
SBR	0.50	Subroutine	Program Flow Control	6-3
SCL	1.75	Scale Data	Math	4-15
SQC	2.00	Sequencer Compare	Application Specific	7-6
SQL	2.00	Sequencer Load	Application Specific	7-12
SQO	2.00	Sequencer Output	Application Specific	7-6
SQR	1.25	Square Root	Math	4-12
STD	0.50	Selectable Timer Interrupt Disable	Interrupt	11-17
STE	0.50	Selectable Timer Interrupt Enable	Interrupt	11-17
STS	1.25	Selectable Timer Interrupt Start	Interrupt	11-18
SUB	1.50	Subtract	Math	4-6
SUS	1.50	Suspend	Program Flow Control	6-8
SVC		Service Comms	Communication	12-2

Table C.2 SLC 5/02 List of Instructions

Mnemonic	Memory Usage (user words)	Name	Instruction Type	Page
TND	0.50	Temporary End	Program Flow Control	6-7
TOD	1.00	Convert to BCD	Data Handling	5-2
TOF	1.00	Timer Off-Delay	Basic	2-10
TON	1.00	Timer On-Delay	Basic	2-9
XIC(1)	1.00	Examine If Closed	Basic	2-3
XIO(1)	1.00	Examine If Open	Basic	2-3
XOR	1.50	Exclusive Or	Data Handling	5-22

(1) These instructions take zero execution time if they are preceded by conditions that guarantee the state of the rung. Rung logic is solved left to right. Branches are solved top to bottom.

Estimating Total Memory Usage of Your System Using a SLC 5/02 Processor

_____	1.	Calculate the total instruction words used by the instructions in your program and enter the result. Refer to the table on page -7.
_____	**2.**	Multiply the total number of rungs by .375 and enter the result.
_____	**3.**	If you are using a 1747-L524 and have enabled the Single Step Test mode, multiply the total number of rung by .375 and enter the result.
_____	**4.**	Multiply the total number of data words (excluding the status file and I/O data words by .25 and enter the result.
_____	**5.**	Add 1 word for each data table file used and enter the result.
_____	**6.**	Multiply the highest numbered program file used by 2 and enter the result.
_____	**7.**	Multiply the total number of I/O data words by .75 and enter the result.
_____	**8.**	Multiply the total number of I/O slots, used or unused, by .75 and enter the result.
_____	**9.**	To account for processor overhead, enter 204
Total: _____	**10.**	Total steps 1 through 9. This is the estimated total memory usage of your application system. Remember, this is an estimate, actual compiled programs may differ by ±12%.
	11.	If you wish to determine the estimated amount of memory remaining in the processor you have selected, do the following: If you are using a 1747-L524, subtract the total from 4096. The result of this calculation will be the estimated total memory remaining in your selected processor.

TIP The calculated memory usage may vary from the actual compiled program by ±12%.

SLC 5/02 Memory Usage Example

1747-L524 series C processor, 30-slot configuration, (15) 1746-IA16, (10) 1746-OA8, (1) 1747-DCM full configuration, (1) 1746-NI4, (1) 1746-NIO4I				
50	XIC and XIO	50 x 1.00	=	50.00
15	OTE instructions	15 x 0.75	=	11.25
5	TON instructions	5 x 1.00	=	5.00
3	GRT instructions	3 x 1.50	=	4.50
1	SCL instruction	1 x 1.75	=	1.75
1	TOD instruction	1 x 1.00	=	1.00
3	MOV instructions	3 x 1.50	=	4.50
10	CTU instructions	10 x 1.00	=	10.00
10	RES instructions	10 x 1.00	=	10.00
Instruction Usage				98.00
30	rungs	30 x .375	=	11.25
100	data words	100 x .250	=	25.00
10	is highest data table file number	10 x 1.00	=	10.00
4	is highest program file number	4 x 2.00	=	8.00
User Program Total				**163.50**
49	I/O data words	49 x 0.75	=	36.75
30	slot	30 x 0.75	=	22.50
Overhead				204.00
I/O Configuration Total				**263.25**
Estimated total memory usage:				426.75
				(round to 427)

4096 - 290 = 3806 instruction words remaining in the processor

User Word Comparison Between SLC 5/03 (and higher) Processors and the SLC 5/02 Processor

The SLC 5/03 (and higher) processors and the SLC 5/02 processor accumulate user words differently during the creation of a user program. The SLC 5/02 processor is generally more efficient in its word usage than the SLC 5/03 (and higher) processors. However, the SLC 5/02 processor word usage is difficult to estimate since it is tied to the architecture of the microprocessor.

The SLC 5/03 (and higher) processors accumulates words in a way that is easier to understand and estimate than the SLC 5/02 processor. The SLC 5/03 (and higher) processors accumulate words similar to a PLC-5.

See page C-1 for user memory capacities for the various SLC processors. It is important to realize that this does not mean that a 16k, SLC 5/03 processor can hold a user program that is three times larger than an equivalent SLC 5/02 program. Use the information below to determine the SLC 5/03 user program size based on existing SLC 5/02 programs.

Instruction Words

Some instructions use the same amount of memory, while other instructions do not use the same amount of memory. For example, a CTU instruction always uses 1 word. However, an ADD instruction in an SLC 5/02 processor uses 1.5 words; in a SLC 5/03 (or higher) processor an ADD instruction uses 3 words. Also note additional differences below:

Condition	SLC 5/02 Words	SLC 5/03 Words	SLC 5/04 and SLC 5/05 Words
Each rung	0.375	1	1
Each additional Program File	1	5	5
Each additional Data File	1	5	5
Each I/O Slot	0.75	3	3
Overhead	216	236	250

Exact program content determines the program size difference. An SLC 5/03 program consumes 20% to 150% more instruction words than its SLC 5/02 equivalent.

Data Words

Files 0 and 1

In the SLC 5/02 processor, each I/O data word consumes 0.75 words of memory. In the SLC 5/03 processor, each I/O data word consumes 3 words of data.

File 2

The status file word usage is contained in the overhead values for both the SLC 5/02 and SLC 5/03 processors.

File 3 to 255

In the SLC 5/02 processor, 4 data words consume the same amount of memory as 1 instruction word. This is why the SLC 5/02 processor is said to offer 4K of Instruction words or 16K of Data words. This dynamic amount of Data word storage is due to the architecture of the SLC 5/02's microprocessor.

SLC 5/03, SLC 5/04 and SLC 5/05 Processor

The following table shows memory usage times for the SLC 5/03, SLC 5/04, and SLC 5/05 processors. Instructions that support floating point are included within this table. When using a SLC 5/03 processor, it is important to remember that 1 instruction word equals 1 data word.

Table C.3 SLC 5/03, SLC 5/04 and SLC 5/05 List of Instructions

Mnemonic		Applies to SLC				Memory Usage (user words)	Name	Instruction Type	Page
FP = floating point		OS300	OS301, OS400	OS302, OS401, OS501	OS302, OS401, OS501 Series C				
ABL			•	•	•	2.00	Test Buffer for Line	ASCII	10-6
ABS		•	•	•	•	2.00	Absolute	Math	4-24
ABS	FP			•	•	2.00	Absolute	Math	4-24
ACB			•	•	•	2.00	Number of Characters in Buffer	ASCII	10-7
ACI			•	•	•	2.00	String to Integer	ASCII	10-8
ACL			•	•	•	2.00	ASCII Clear Receive and/or Send Buffer	ASCII	10-9

Table C.3 SLC 5/03, SLC 5/04 and SLC 5/05 List of Instructions

Mnemonic FP = floating point		Applies to SLC				Memory Usage (user words)	Name	Instruction Type	Page
		OS300	OS301, OS400	OS302, OS401, OS501	OS302, OS401, OS501 Series C				
ACN			•	•	•	3.00	String Concatenate	ASCII	10-10
ACS				•	•	2.00	Arc Cosine	Math	4-29
ADD		•	•	•	•	3.00, 4.00	Add	Math	4-5
ADD	FP			•	•	4.00	Add	Math	4-5
AEX			•	•	•	4.00	String Extract	ASCII	10-10
AHL			•	•	•	4.00	ASCII Handshake Lines	ASCII	10-11
AIC			•	•	•	2.00	Integer to String	ASCII	10-13
AND		•	•	•	•	3.00	And	Data Handling	5-20
ARD			•	•	•	3.00	ASCII Read Characters	ASCII	10-13
ARL			•	•	•	3.00	ASCII Read Line	ASCII	10-16
ASC			•	•	•	4.00	String Search	ASCII	10-17
ASN	FP			•	•	2.00	Arc Sine	Math	4-28
ASR			•	•	•	3.00	ASCII String Compare	ASCII	10-18
ATN	FP			•	•	2.00	Arc Tangent	Math	4-29
AWA			•	•	•	3.00	ASCII Write with Append	ASCII	10-19
AWT			•	•	•	3.00	ASCII Write	ASCII	10-21
BSL		•	•	•	•	3.00	Bit Shift Left	Application Specific	7-4
BSR		•	•	•	•	3.00	Bit Shift Right	Application Specific	7-4
BTR					•	5.00	Block Transfer Read	Block Transfer	8-1
BTW					•	5.00	Block Transfer Write	Block Transfer	8-1
CLR		•	•	•	•	3.00, 1.00	Clear	Math	4-12
CLR	FP			•	•	1.00	Clear	Math	4-12
COP		•	•	•	•	3.00	File Copy	Data Handling	5-12
COS	FP			•	•	2.00	Cosine	Math	4-30
CPT				•	•	(1)	Compute	Math	4-25
CTD		•	•	•	•	1.00	Count Down	Basic	2-14
CTU		•	•	•	•	1.00	Count Up	Basic	2-13
DCD		•	•	•	•	2.00	Decode 4 to 1 of 16	Data Handling	5-10
DDV			•	•	•	2.00	Double Divide	Math	4-11
DDT					•	6.00	Diagnostic Detect	Application Specific	7-18

Table C.3 SLC 5/03, SLC 5/04 and SLC 5/05 List of Instructions

Mnemonic FP = floating point		Applies to SLC				Memory Usage (user words)	Name	Instruction Type	Page
		OS300	OS301, OS400	OS302, OS401, OS501	OS302, OS401, OS501 Series C				
DEG	FP			•	•	2.00	Degree	Data Handling	5-8
DIV		•	•	•	•	3.00, 4.00	Divide	Math	4-9
DIV	FP			•	•	4.00	Divide	Math	4-9
ENC					•	2.00	Encode 1 of 16 to 4	Data Handling	5-11
EQU		•	•	•	•	3.00	Equal	Comparison	3-2
EQU	FP		•	•	•	3.00	Equal	Comparison	3-2
FBC					•	6.00	File Bit Comparison	Application Specific	7-18
FFL		•	•	•	•	3.00	FIFO Load	Data Handling	5-26
FFU		•	•	•	•	4.00	FIFO Unload	Data Handling	5-26
FLL		•	•	•	•	3.00	Fill File	Data Handling	5-12
FRD		•	•	•	•	2.00	Convert from BCD	Data Handling	5-5
GEQ		•	•	•	•	3.00	Greater Than or Equal	Comparison	3-4
GEQ	FP		•	•	•	3.00	Greater Than or Equal	Comparison	3-4
GRT		•	•	•	•	3.00	Greater Than	Comparison	3-3
GRT	FP		•	•	•	3.00	Greater Than	Comparison	3-3
IID		•	•	•	•	2.00	I/O Interrupt Disable	Interrupt	11-34
IIE		•	•	•	•	2.00	I/O Interrupt Enable	Interrupt	11-34
IIM		•	•	•	•	6.00	Immediate Input with Mask	Program Flow Control	6-8
INT		•	•	•	•	1.00	Interrupt Subroutine	Interrupt	11-36
IOM		•	•	•	•	6.00	Immediate Output with Mask	Program Flow Control	6-9
JMP		•	•	•	•	1.00	Jump to Label	Program Flow Control	6-2
JSR		•	•	•	•	1.00	Jump to Subroutine	Program Flow Control	6-3
LBL		•	•	•	•	2.00	Label	Program Flow Control	6-2
LEQ		•	•	•	•	3.00	Less Than or Equal	Comparison	3-3
LEQ	FP		•	•	•	3.00	Less Than or Equal	Comparison	3-3
LES		•	•	•	•	3.00	Less Than	Comparison	3-3
LES	FP		•	•	•	3.00	Less Than	Comparison	3-3
LFL		•	•	•	•	3.00	LIFO Load	Data Handling	5-28
LFU		•	•	•	•	3.00	LIFO Unload	Data Handling	5-28
LIM		•	•	•	•	1.00	Limit Test	Comparison	3-4

Table C.3 SLC 5/03, SLC 5/04 and SLC 5/05 List of Instructions

Mnemonic FP = floating point		Applies to SLC				Memory Usage (user words)	Name	Instruction Type	Page
		OS300	OS301, OS400	OS302, OS401, OS501	OS302, OS401, OS501 Series C				
LIM	FP		•	•	•	1.00	Limit Test	Comparison	3-4
LN	FP			•	•	2.00	Natural Log	Math	4-30
LOG	FP			•	•	2.00	Log to the Base 10	Math	4-31
MCR		•	•	•	•	1.00	Master Control Reset	Program Flow Control	6-6
MEQ		•	•	•	•	4.00	Masked Comparison for Equal	Comparison	3-4
MOV		•	•	•	•	2.00	Move	Data Handling	5-17
MOV	FP		•	•	•	2.00	Move	Data Handling	5-17
MSG		•	•	•	•	20.00	Message	Communication	12-3
MUL		•	•	•	•	3.00	Multiply	Math	4-8
MUL	FP			•	•	3.00	Multiply	Math	4-8
MVM		•	•	•	•	3.00, 4.00	Masked Move	Data Handling	5-18
NEG		•	•	•	•	3.00	Negate	Data Handling	5-24
NEG	FP			•	•	3.00	Negate	Data Handling	5-24
NEQ		•	•	•	•	3.00	Not Equal	Comparison	3-2
NEQ	FP		•	•	•	3.00	Not Equal	Comparison	3-2
NOT		•	•	•	•	3.00	Not	Data Handling	5-23
OR		•	•	•	•	3.00	Or	Data Handling	5-21
OTE		•	•	•	•	1.00	Output Energize	Basic	2-4
OTL		•	•	•	•	1.00	Output Latch	Basic	2-4
OTU		•	•	•	•	1.00	Output Unlatch	Basic	2-4
PID		•	•	•	•	26.00	Proportional Integral Derivative	PID	9-1
RAD	FP			•	•	2.00	Radian	Data Handling	5-9
REF		•	•	•	•	1.00	I/O Refresh	Program Flow Control	6-10
RES		•	•	•	•	1.00	Reset	Basic	2-20
RET		•	•	•	•	1.00	Return from Subroutine	Program Flow Control	6-3
RHC					•	2.00	Read High Speed Clock	Application Specific	7-17
RMP					•	2.00	Ramp	Math	4-20
RPI		•	•	•	•	2.00	Reset Pending Interrupt	Interrupt	11-36
RTO		•	•	•	•	1.00	Retentive Timer	Basic	2-11
SBR		•	•	•	•	1.00	Subroutine	Program Flow Control	6-3

Table C.3 SLC 5/03, SLC 5/04 and SLC 5/05 List of Instructions

Mnemonic FP = floating point		Applies to SLC				Memory Usage (user words)	Name	Instruction Type	Page
		OS300	OS301, OS400	OS302, OS401, OS501	OS302, OS401, OS501 Series C				
SCL	FP			•	•	4.00	Scale Data	Math	4-15
SCP				•	•	6.00	Scale with Parameters	Math	4-13
SCP	FP			•	•	6.00	Scale with Parameters	Math	4-13
SIN	FP			•	•	2.00	Sine	Math	4-31
SQC		•	•	•	•	5.00	Sequencer Compare	Application Specific	7-6
SQL		•	•	•	•	4.00	Sequencer Load	Application Specific	7-12
SQO		•	•	•	•	5.00	Sequencer Output	Application Specific	7-6
SQR		•	•	•	•	2.00, 3.00	Square Root	Math	4-12
SQR	FP			•	•	3.00	Square Root	Math	4-12
STD		•	•	•	•	1.00	Selectable Timer Interrupt Disable	Interrupt	11-17
STE		•	•	•	•	1.00	Selectable Timer Interrupt Enable	Interrupt	11-17
STS		•	•	•	•	3.00	Selectable Timer Interrupt Start	Interrupt	11-18
SUB		•	•	•	•	3.00	Subtract	Math	4-5
SUB	FP			•	•	4.00	Subtract	Math	4-5
SUS		•	•	•	•	2.00	Suspend	Program Flow Control	6-8
SVC		•	•	•	•	1.00	Service Communication	Communication	12-2
SWP				•	•	2.00	SWAP	Math	4-27
TAN	FP			•	•	2.00	Tangent	Math	4-32
TDF					•	3.00	Compute Time Difference	Application Specific	7-17
TND		•	•	•	•	1.00	Temporary End	Program Flow Control	6-7
TOD		•	•	•	•	2.00	Convert to BCD	Data Handling	5-2
TOF		•	•	•	•	1.00	Timer Off-Delay	Basic	2-10
TON		•	•	•	•	1.00	Timer On-Delay	Basic	2-9
XIC		•	•	•	•	1.00	Examine If Closed	Basic	2-3
XIO		•	•	•	•	1.00	Examine If Open	Basic	2-3
XOR		•	•	•	•	3.00	Exclusive Or	Data Handling	5-22
XPY	FP			•	•	3.00	X to the Power of Y	Math	4-32

(1) To calculate the memory usage, do the following: Take 2 plus the number of instruction words for each operation performed plus the number of operations performed in the compute. For example, 2 + ADD + SUB + 2 =10.

Estimating Total Memory Usage of Your System Using an SLC 5/03, SLC 5/04 or SLC 5/05 Processor

_____	**1.**	Add the total number of data file words used (excluding the status file and I/O data words) and enter the result.
_____	**2.**	Multiply the total number of I/O data words by 3 and enter the result.
_____	**3.**	Multiply the total number of I/O slots, used or unused, by 3 and enter the result.
_____	**4.**	To account for processor overhead, enter 236.
_____	**5.**	Multiply the highest numbered program file used by 5 and enter the result.
_____	**6.**	Multiply the highest numbered program fiel used by 5 and enter the result.
Subtotal: _____	**7.**	Add steps 1 through 6. Enter this as the subtotal (additional 4K word usage).
4096 - (step 7 value)	**8.**	Subtract the value in step 7 from 4096; if the result is positive, enter 12,288 in step 14. If the result is negative, subtract the absolute value from 12,288 and enter the result in step 14. (This decreases the value.)
_____		
_____	**9.**	Calculate the total number of words used by the instructions in your program and enter the result. Refer to the table on page -7.
_____	**10.**	Add the total number of rungs (1 word per rung) and enter the result.
_____	**11.**	Add 1 word for each indexed address reference and enter the result.
_____	**12.**	Add 2 words per rung for each rung that contains an indexed address reference and enter the result
Subtotal: _____	**13.**	Add steps 9 through 12 and enter the result.
_____	**14.**	Enter the result from step 8. This is the available memory.
_____	**15.**	Enter the result from step 13. This is the total number of words used.
Total: _____	**16.**	Subtract step 15 from step 14. This number is the amount of memory available to your system.

SLC 5/03, SLC 5/04 or SLC 5/05 Memory Usage Example

1747-L532 processor, 30-slot configuration, (15) 1746-IA16, (10) 1746-OA8, (1) 1747-DCM full configuration, (1) 1746-NI4, (1) 1746-NIO4I			
100	data words	100 x 1.00	= 100.00
49	I/O data words	49 x 3.00	= 147.00
30	slot	30 x 3.00	= 90.00
Overhead			236.00
10	is the highest data table file number	10 x 5.00	= 50.00
4	is the highest program file number	4 x 5.00	= 20.00
Subtotal			**643.00**
Account for additional 4K data space			
	4096 - 643 = 3453 (result is positive; therefore **12,288** words are available)		
50	XIC and XIO	50 x 1.00	= 50.00
15	OTE instructions	15 x 1.00	= 15.00
5	TON instructions	5 x 1.00	= 5.00
3	GRT instructions	3 x 3.00	= 9.00
1	SCL instruction	1 x 4.00	= 4.00
t	TOD instruction	1 x 2.00	= 2.00
3	MOV instructions	3 x 2.00	= 6.00
10	CTU instructions	10 x 1.00	= 10.00
10	RES instructions	10 x 1.00	= 10.00
Instruction USage			**111.00**
30	rungs	30 x 1.00	= **30.00**
0	indexed address		= 0.00
0	indexed address reference		0.00
Subtotal			**141.00**
Available memory			12,288.00
Words used			- 141.00
Estimated total memory available:			**12,147.00**

Programming Instruction References

This appendix lists all of the available programming instructions along with their parameters, valid addressing modes, and file types.

Valid Addressing Modes and File Types

The following addressing modes are available:

Addressing Mode	Example
Direct	N7:0
Indexed Direct	#N7:0
Indirect	N7:[N10:3]
Indexed Indirect	#N7:[N10:3]

The following file types are available:

Abbreviation	File Type
O	Output
I	Input
S	Status
B	Binary
T	Timer
C	Counter
R	Control
N	Integer
F	Float(1)(2)
A	ASCII(1)(2)
ST	String(1)(2)
M	M0/M1(3)
Immediate	indicates that a constant is a valid file type

(1) Supported only by SLC 5/03 (OS301 and higher), SLC 5/04, and SLC 5/05 processors.

(2) Not supported by Fixed SLC, SLC 5/01, and SLC 5/02 processors.

(3) Not supported by Fixed SLC and SLC 5/01 processors.

Understanding the Different Addressing Modes

The following descriptions will help you understand how to structure a specific type of address.

Direct Addressing

The data stored in the specified address is used in the instruction. For example:

N7:0
ST20:5
T4:8.ACC

Indexed Addressing

You may specify an address as being indexed by placing the "#" character in front of the address. When an address of this form is encountered in the program, the processor takes the element number of the address and adds to it the value contained in the Index Register S:24, then uses the result as the actual address. For example:

#N7:10 where S:24 = 15
The actual address used by the instruction is N7:25.

Indirect Addressing

You may specify an address as being indirect by replacing the file number, element number, or sub-element number with a [Xf:e.s] symbol. The word address inside of the bracket is queried for a value. The queried value then becomes the file, element, or sub-element portion of the indirect address. For example:

B3:[N10:2] states that the element address of Bit file 3 is contained in address N10:2. Therefore, if N10:2 contains the value 5, B3:[N10:2] indirectly refers to address B3:5. Other examples include:

N7:[N7:0]
N7:[T4:0.ACC]
N[N7:0]:[N7:1]
C5:[N7:0]

Indexed Indirect Addressing

You may specify a combination of indirect and indexed addressing. The processor first resolves the indirect portion of the address and

then adds the offset from the Index Register S:24 to come up with the final address. For example:

#N7:[N10:3] where N10:3 = 20 and S:24 = 15
The actual address used by the instruction is N7:35.

Instruction	Description	Instruction Parameter	Valid Addressing Mode(s)	Valid File Types	Immediate Values
ABL[1]	ASCII Test Buffer for Line	channel			0
		control	direct	R	none
		characters			0-1024
ABS[2]	Absolute Value	source	direct, indexed direct indirect, indexed indirect	O, I, S, B, T, C, R, N, F, A, ST, M	-32,768 to 32,767 f-min to f-max
		destination	direct, indexed direct indirect, indexed indirect	O, I, S, B, T, C, R, N, F, A, ST, M	none
ACB[1]	ASCII Number of Characters in Buffer	channel			0
		control	direct	R	none
		characters			0 to 1024
ACI[1]	ASCII String to Integer	source	direct, indirect	ST	none
		destination	direct, indexed direct indirect, indexed indirect	O, I, S, B, T, C, R, N, A, ST, M	none
ACL[1]	ASCII Clear Buffer	channel			0
		transmit buffer			0=no or 1=yes
		receive buffer			0=no or 1=yes
ACN[1]	ASCII String Concatenate	source A	direct, indirect	ST	none
		source B	direct, indirect	ST	none
		destination	direct	ST	none
ACS[2]	Arc Cosine	source	direct, indexed direct indirect, indexed indirect	O, I, S, B, T, C, R, N, F, A, ST M	-32,768 to 32,767 f-min to f-max
		destination	direct, indexed direct indirect, indexed indirect	O, I, S, B, T, C, R, N, F, A, ST, M	none
ADD	Add	source A	direct, indexed direct indirect, indexed indirect	O, I, S, B, T, C, R, N, F, A, ST, M	-32,768 to 32,767 f-min to f-max
		source B	direct, indexed direct indirect, indexed indirect	O, I, S, B, T, C, R, N, F, A, ST, M	-32,768 to 32,767 f-min to f-max
		destination	direct, indexed direct indirect, indexed indirect	O, I, S, B, T, C, R, N, F, A, ST, M	none

Instruction	Description	Instruction Parameter	Valid Addressing Mode(s)	Valid File Types	Immediate Values
AEX[1]	ASCII String Extract	source	direct, indirect	ST	none
		index	direct, indexed direct indirect, indexed indirect	O, I, S, B, T, C, R, N, A, ST, M	1 to 82
		number	direct, indexed direct indirect, indexed indirect	O, I, S, B, T, C, R, N, A, ST, M	1 to 82
		destination	direct	ST	none
AHL[1]	ASCII Set/Reset Handshake Lines	channel			0
		AND mask	direct, indexed direct indirect, indexed indirect	O, I, S, B, T, C, R, N, A, ST, M	0 to FFFF
		OR mask	direct, indexed direct indirect, indexed indirect	O, I, S, B, T, C, R, N, A, ST, M	0 to FFFF
		control	direct	R	none
		channel status			0 to 001F
AIC[1]	ASCII Integer to String	source	direct, indexed direct indirect, indexed indirect	O, I, S, B, T, C, R, N, A, ST, M	-32,768 to 32,767
		destination	direct	ST	none
AND	Logical AND	source A	direct, indexed direct indirect, indexed indirect	O, I, S, B, T, C, R, N, A, ST, M	-32,768 to 32,767
		source B	direct, indexed direct indirect, indexed indirect	O, I, S, B, T, C, R, N, A, ST, M	-32,768 to 32,767
		destination	direct, indexed direct indirect, indexed indirect	O, I, S, B, T, C, R, N, A, ST, M	none
ARD[1]	ASCII Read Characters	channel			0
		destination	direct	ST	none
		control	direct	R	none
		string length			0 to 82
		characters read			0 to 82
ARL[1]	ASCII Read Line	channel			0
		destination	direct	ST	none
		control	direct	R	none
		string length			0 to 82
		characters read			0 to 82
ASC[1]	ASCII String Search	source	direct, indirect	ST	none
		index	direct, indexed direct indirect, indexed indirect	O, I, S, B, T, C, R, N, A, ST, M	1 to 82
		search	direct, indirect	ST	none
		result	direct, indexed direct indirect, indexed indirect	O, I, S, B, T, C, R, N, A, ST, M	none

Instruction	Description	Instruction Parameter	Valid Addressing Mode(s)	Valid File Types	Immediate Values
ASN[2]	Arc Sine	source	direct, indexed direct indirect, indexed indirect	O, I, S, B, T, C, R, N, F, A, ST, M	-32,768 to 32,767 f-min to f-max
		destination	direct, indexed direct indirect, indexed indirect	O, I, S, B, T, C, R, N, F, A, ST, M	none
ASR[1]	ASCII String Compare	source A	direct, indirect	ST	none
		source B	direct, indirect	ST	none
ATN[2]	Arc Tangent	source	direct, indexed direct indirect, indexed indirect	O, I, S, B, T, C, R, N, F, A, ST, M	-32,768 to 32,767 f-min to f-max
		destination	direct, indexed direct indirect, indexed indirect	O, I, S, B, T, C, R, N, F, A, ST, M	none
AWA[1]	ASCII Write with Append	channel			0
		source	direct	ST	none
		control	direct	R	none
		string length			0 to 82
		characters sent			0 to 82
AWT[1]	ASCII Write	channel			0
		source	direct	ST	none
		control	direct	R	none
		string length			0 to 82
		characters sent			0 to 82
BSL	Bit Shift Left	file	indexed direct indexed indirect	O, I, S, B, N, A, ST	none
		control	direct	R	none
		bit address	direct, indirect	O, I, S, B, T, C, R, N, A, ST, M	none
		length			0 to 2048
BSR	Bit Shift Right	file	indexed direct indexed indirect	O, I, S, B, N, A, ST	none
		control	direct	R	none
		bit address	direct, indirect	O, I, S, B, T, C, R, N, A, ST, M	none
		length			0 to 2048
BTR	Block Transfer Read	control	direct	N	
		source "Data"	direct	I, O, S, B, N, A, F	
		buffer	direct M1 file address	M	
BTW	Block Transfer Write	control	direct	N	
		source "Data"	direct	I, O, S, B, N, A, F	
		buffer	direct M0 file address	M	

Instruction	Description	Instruction Parameter	Valid Addressing Mode(s)	Valid File Types	Immediate Values
CLR	Clear	destination	direct, indexed direct indirect, indexed indirect	O, I, S, B, T, C, R, N, F, A, ST, M	none
COP	Copy File	source	indexed direct indexed indirect	O, I, S, B, T, C, R, N, F, A, ST, M	none
		destination	indexed direct indexed indirect	O, I, S, B, T, C, R, N, F, A, ST, M	none
		length			1 to 128
COS[2]	Cosine	source	direct, indexed direct indirect, indexed indirect	O, I, S, B, T, C, R, N, F, A, ST, M	-32,768 to 32,767 f-min to f-max
		destination	direct, indexed direct indirect, indexed indirect	O, I, S, B, T, C, R, N, F, A, ST, M	none
CPT[2]	Compute	destination	direct, indexed direct indirect, indexed indirect	O, I, S, B, T, C, R, N, F, A, ST, M	none
		expression			<expression>
CTD	Count Down	counter	direct	C	none
		preset			-32,768 to 32,767
		accum			-32,768 to 32,767
CTU	Count Up	counter	direct	C	none
		preset			-32,768 to 32,767
		accum			-32,768 to 32,767
DCD	Decode 4 to 1 of 16	source	direct, indexed direct indirect, indexed indirect	O, I, S, B, T, C, R, N, A, ST, M	none
		destination	direct, indexed direct indirect, indexed indirect	O, I, S, B, T, C, R, N, A, ST, M	none
DDV	Double Divide	source	direct, indexed direct indirect, indexed indirect	O, I, S, B, T, C, R, N, A, ST, M	-32,768 to 32,767
		destination	direct, indexed direct indirect, indexed indirect	O, I, S, B, T, C, R, N, A, ST, M	none
DDT	Diagnostic Detect	control	direct	R	
		source “file”	index direct	I, O, S, B, N, A	
		reference “file”	index direct	I, O, S, B, N, A	
		result “file”	index direct	I, O, S, B, N, A	
DEG[2]	Radians to Degrees	source	direct, indexed direct indirect, indexed indirect	O, I, S, B, T, C, R, N, F, A, ST, M	-32,768 to 32,767 f-min to f-max
		destination	direct, indexed direct indirect, indexed indirect	O, I, S, B, T, C, R, N, F, A, ST, M	none

Instruction	Description	Instruction Parameter	Valid Addressing Mode(s)	Valid File Types	Immediate Values
DIV	Divide	source A	direct, indexed direct indirect, indexed indirect	O, I, S, B, T, C, R, N, F, A, ST, M	-32,768 to 32,767 f-min to f-max
		source B	direct, indexed direct indirect, indexed indirect	O, I, S, B, T, C, R, N, F, A, ST, M	-32,768 to 32,767 f-min to f-max
		destination	direct, indexed direct indirect, indexed indirect	O, I, S, B, T, C, R, N, F, A, ST, M	none
ENC	Encode 1 of 16 to 4	source	direct, indirect, index, indirect index	I, O, S, B, N, A, T	
		destination	direct, indirect, index, indirect index	I, O, S, B, N, A, C, R	
EQU	Equal	source A	direct, indexed direct indirect, indexed indirect	O, I, S, B, T, C, R, N, F, A, ST, M	none
		source B	direct, indexed direct indirect, indexed indirect	O, I, S, B, T, C, R, N, F, A, ST, M	-32,768 to 32,767 f-min to f-max
FBC	File Bit Comparison	control	direct	R	
		source "file"	index direct	I, O, S, B, N, A	
		reference "file"	index direct	I, O, S, B, N, A	
		result "file"	index direct	I, O, S, B, N, A	
FFL[3]	FIFO Load	source	direct, indexed direct indirect, indexed indirect	O, I, S, B, T, C, R, N, A, ST, M[4]	-32,768 to 32,767
		FIFO array	indexed direct indexed indirect	O, I, S, B, N, A	none
		FIFO control	direct	R	none
		length			1 to 128
		position			0 to 127
FFU[3]	FIFO Unload	FIFO array	indexed direct indexed indirect	O, I, S, B, N, A	none
		destination	direct, indexed direct indirect, indexed indirect	O, I, S, B, T, C, R, N, A, ST, M[4]	none
		FIFO control	direct	R	none
		length			1 to 128
		position			0 to 127
FLL	Fill File	source	direct, indirect	O, I, S, B, T, C, R, N, F, A, ST, M	-32,768 to 32,767 f-min to f-max
		destination	indexed direct indexed indirect	O, I, S, B, T, C, R, N, F, A, ST, M	none
		length			1 to 128

Instruction	Description	Instruction Parameter	Valid Addressing Mode(s)	Valid File Types	Immediate Values
FRD	From BCD to Binary	source (SLC 5/01)	direct	O, I, S, B, T, C, R, N, A, ST, M	none
		source (SLC 5/02, 5/03, 5/04, 5/05)	direct, indexed direct indirect, indexed indirect	O, I, S, B, T, C, R, N	none
		destination	direct, indexed direct indirect, indexed indirect	O, I, S, B, T, C, R, N, A, ST, M	none
GEQ	Greater Than or Equal	source A	direct, indexed direct indirect, indexed indirect	O, I, S, B, T, C, R, N, F, A, ST, M	none
		source B	direct, indexed direct indirect, indexed indirect	O, I, S, B, T, C, R, N, F, A, ST, M	-32,768 to 32,767 f-min to f-max
GRT	Greater Than	source A	direct, indexed direct indirect, indexed indirect	O, I, S, B, T, C, R, N, F, A, ST, M	none
		source B	direct, indexed direct indirect, indexed indirect	O, I, S, B, T, C, R, N, F, A, ST, M	-32,768 to 32,767 f-min to f-max
HSC[5]	High-Speed Counter (SLC 5/01)	counter			none
		preset			1 to 32,767
		counter	direct	C	none
		preset			-32,768 to 32,767
		accum			-32,768 to 32,767
		source	direct	B and N	none
		length			always 5
IID[6]	I/O Interrupt Disable	slots			double hex word (list of slots)
IIE[6]	I/O Interrupt Enable	slots			double hex word (list of slots)
IIM	Immediate Input with Mask	slot	direct	I	none
		mask	direct, indexed direct indirect, indexed indirect	O, I, S, B, T, C, R, N, A, ST, M	-32,768 to 32,767
		length (SLC 5/03, 5/04, and 5/05)			1 to 32
INT[3]	I/O Interrupt				none
IOM	Immediate Output with Mask	slot	direct	O	none
		mask	direct, indexed direct indirect, indexed indirect	O, I, S, B, T, C, R, N, A, ST, M	-32,768 to 32,767
		length (SLC 5/03, 5/04, and 5/05)			1 to 32
JMP	Jump	label number			0 to 999
JSR	Jump to Subroutine	subroutine file number			3 to 255

Instruction	Description	Instruction Parameter	Valid Addressing Mode(s)	Valid File Types	Immediate Values
LBL	Label Declaration	label number			0 to 999
LEQ	Less Than or Equal To	source A	direct, indexed direct indirect, indexed indirect	O, I, S, B, T, C, R, N, F, A, ST, M	none
		source B	direct, indexed direct indirect, indexed indirect	O, I, S, B, T, C, R, N, F, A, ST, M	-32,768 to 32,767 f-min to f-max
LES	Less Than	source A	direct, indexed direct indirect, indexed indirect	O, I, S, B, T, C, R, N, F, A, ST, M	none
		source B	direct, indexed direct indirect, indexed indirect	O, I, S, B, T, C, R, N, F, A, ST, M	-32,768 to 32,767 f-min to f-max
LFL[3]	LIFO Load	source	direct, indexed direct indirect, indexed indirect	O, I, S, B, T, C, R, N, A, ST, M[4]	-32,768 to 32,767
		LIFO array	indexed direct indexed indirect	O, I, S, B, N, A	none
		LIFO control	direct	R	none
		length			1 to 128
		position			0 to 127
LFU[3]	LIFO Unload	LIFO array	indexed direct indexed indirect	O, I, S, B, N, A	none
		destination	direct, indexed direct indirect, indexed indirect	O, I, S, B, T, C, R, N, A, ST, M[4]	none
		LIFO control	direct	R	none
		length			1 to 128
		position			0 to 127
LIM[3]	Limit Test (circ)	low limit	direct, indexed direct indirect, indexed indirect	O, I, S, B, T, C, R, N, F, A, ST, M	-32,768 to 32,767 f-min to f-max
		test	direct, indexed direct indirect, indexed indirect	O, I, S, B, T, C, R, N, F, A, ST, M	-32,768 to 32,767 f-min to f-max
		high limit	direct, indexed direct indirect, indexed indirect	O, I, S, B, T, C, R, N, F, A, ST, M	-32,768 to 32,767 f-min to f-max
LN[2]	Natural Log	source	direct, indexed direct indirect, indexed indirect	O, I, S, B, T, C, R, N, F, A, ST, M	-32,768 to 32,767 f-min to f-max
		destination	direct, indexed direct indirect, indexed indirect	O, I, S, B, T, C, R, N, F, A, ST, M	none
LOG[2]	Log to the Base 10	source	direct, indexed direct indirect, indexed indirect	O, I, S, B, T, C, R, N, F, A, ST, M	-32,768 to 32,767 f-min to f-max
		destination	direct, indexed direct indirect, indexed indirect	O, I, S, B, T, C, R, N, F, A, ST, M	none
MCR	Master Control Relay				none

Instruction	Description	Instruction Parameter	Valid Addressing Mode(s)	Valid File Types	Immediate Values
MEQ	Mask Compare Equal To	source	direct, indexed direct indirect, indexed indirect	O, I, S, B, T, C, R, N, A, ST, M	none
		source mask	direct, indexed direct indirect, indexed indirect	O, I, S, B, T, C, R, N, A, ST, M	-32,768 to 32,767
		compare	direct, indexed direct indirect, indexed indirect	O, I, S, B, T, C, R, N, A, ST, M	-32,768 to 32,767
MOV	Move	source	direct, indexed direct indirect, indexed indirect	O, I, S, B, T, C, R, N, F, A, ST, M	-32,768 to 32,767 f-min to f-max
		destination	direct, indexed direct indirect, indexed indirect	O, I, S, B, T, C, R, N, F, A, ST, M	none
MSG (5/02 only)	Message	read/write			0=read,1=write
		target device			2=500CPU, 4=485CIF
		control block	direct	N	none
		control block length			7
		local address	direct	O, I, S, B, T, C, R, N, A	none
		target node			0 to 31
		target address	direct	O, I, S, B, T, C, R, N, A	0 to 255
		message length		T, C, R	1 to 13
				I, O, S, B, N	1 to 41

Instruction	Description	Instruction Parameter	Valid Addressing Mode(s)	Valid File Types	Immediate Values
MSG (5/03, 5/04, and 5/05[7])	Message	type			64=peer-to-peer
		read/write			0=read, 1=write
		target device			2=500CPU, 4=485CIF, 8=PLC5
		local/remote			16=local, 32=remote
		control block	direct	N	none
		control block length			14[8]
		channel number			5/03 and 5/04: 0 or 1 5/05: 0 only
		target node			0 to 31, 0 to 254 if 485CIF
		remote bridge link ID			0 to 254, 0 when local
		remote bridge node address			0 to 254, 0 when local
		local bridge node address			0 to 254, 0xFFFF when local
		local file address	direct	O, I, S, B, T, C, R, N, F, A, ST, M[9]	none
		target file address	direct	O, I, S, B, T, C, R, N, F, A, ST, M[9]	0 to 255
		message length		O, I, B, N, A[9]	1 to 103
				S[9]	5/03 and 5/05: 1 to 83 5/04: 1 to 164
				F[9]	1 to 51
				T	1 to 34 (if PLC5: 1 to 20)
				C, R	1 to 34
				ST[9]	2 (if PLC5: 1)
		message timeout			0 to 255 (SLC 5/05-23 seconds, read only)

Instruction	Description	Instruction Parameter	Valid Addressing Mode(s)	Valid File Types	Immediate Values
MSG (5/05 Ethernet)	Message	type			64=peer-to-peer
		read/write			0=read, 1=write
		target device			2=500CPU, 4=485CIF, 8=PLC5
		local			16=local
		control block	direct	N	none
		control block length			51 (93 if logical ASCII addressing is used)
		channel number			1 (Ethernet)
		target node, remote bridge link ID, local and remote bridge node address	not applicable		
		IP address (ww.xx.yy.zz)			any legal IP address
		local file address	direct	O, I, S, B, T, C, R, N, F, A, ST, M	none
		target file address	direct	O, I, S, B, T, C, R, N, F, A, ST, M	0 to 255
		message timeout			0 to 255

(1) Supported only by SLC 5/03 (OS301 and higher), SLC 5/04, and SLC 5/05 processors.

(2) Supported only by SLC 5/03 (OS302), and SLC 5/04 (OS401), and SLC 5/05 processors.

(3) Not supported by SLC 5/01 processors and Fixed controllers.

(4) Indexed addressing is not allowed when using T, C, R, or M addresses.

(5) Supported only by L20, L30, and L40 Fixed SLC processors with DC inputs.

(6) Supported only by SLC 5/02, SLC 5/03, SLC 5/04, and SLC 5/05 processors.

(7) SLC 5/05 Channel 0 (RS-232 serial port) only.

(8) For SLC 5/05, control block length = 55 if logical ASCII addressing is used.

(9) File types F, A, and ST only apply to SLC 5/03 (OS301 or later), SLC 5/04, and SLC 5/05 processors.

TIP

Message lengths for SLC 5/05 processors are shown in the next table.

Message lengths are based on Ethernet buffer size of 2108 bytes (includes command header and system addressing in addition to actual file data). The local file is the destination file for reads and the source files for writes.

In the following table, byte values are the maximum byte length argument values the compiler should generate to be passed to the MSG instruction and then written to word 11 of the MSG control block by the processor at run time.

File Type	485CIF and 500CPU Read/Write		PLC5 Read		PLC5 Write	
	Elements	Bytes	Elements	Bytes	Elements	Bytes
O,I	256	512	256	512	256	512
S	83	166	83	166	83	166
B	256	512	256	512	256	512
T	256	1536	256 (208 when target file type is Timer)	1536 (1248 when target file type is Timer)	208 (201 when using logical ASCII addressing)	1248 (1206 when using logical ASCII addressing)
C	256	1536	256	1536	256	1536
R	256	1536	256	1536	256	1536
N	256	512	256	512	256	512
F	256	1024	256	1024	256	1024
A	256	512 - Treat an ASCII file element here as a whole 16-bit word.	256	512 - Treat an ASCII file element here as 1 byte like PLCs do; compensate by doubling the number of elements that can be read.	256	512 - Treat an ASCII file element here as 1 byte like PLCs do; compensate by doubling the number of elements that can be read.
ST	25 (24 for 500CPU write)	2100 (2016 for 500CPU write)	1		1	
M	256	512	256	512	256	512

Instruction	Description	Parameter	Valid Addressing Mode(s)	Valid File Types	Immediate Values
MUL	Multiply	source A	direct, indexed direct indirect, indexed indirect	O, I, S, B, T, C, R, N, F, A, ST, M	-32,768 to 32,767 f-min to f-max
		source B	direct, indexed direct indirect, indexed indirect	O, I, S, B, T, C, R, N, F, A, ST, M	-32,768 to 32,767 f-min to f-max
		destination	direct, indexed direct indirect, indexed indirect	O, I, S, B, T, C, R, N, F, A, ST, M	none
MVM	Masked Move	source	direct, indexed direct indirect, indexed indirect	O, I, S, B, T, C, R, N, A, ST, M	none
		source mask	direct, indexed direct indirect, indexed indirect	O, I, S, B, T, C, R, N, A, ST, M	-32,768 to 32,767
		destination	direct, indexed direct indirect, indexed indirect	O, I, S, B, T, C, R, N, A, ST, M	none
NEG	Negate	source	direct, indexed direct indirect, indexed indirect	O, I, S, B, T, C, R, N, F, A, ST, M	none
		destination	direct, indexed direct indirect, indexed indirect	O, I, S, B, T, C, R, N, F, A, ST, M	none
NEQ	Not Equal To	source A	direct, indexed direct indirect, indexed indirect	O, I, S, B, T, C, R, N, F, A, ST, M	none
		source B	direct, indexed direct indirect, indexed indirect	O, I, S, B, T, C, R, N, F, A, ST, M	-32,768 to 32,767 f-min to f-max
NOT	Logical NOT	source	direct, indexed direct indirect, indexed indirect	O, I, S, B, T, C, R, N, A, ST, M	none
		destination	direct, indexed direct indirect, indexed indirect	O, I, S, B, T, C, R, N, A, ST, M	none
OR	Logical OR	source A	direct, indexed direct indirect, indexed indirect	O, I, S, B, T, C, R, N, A, ST, M	-32,768 to 32,767
		source B	direct, indexed direct indirect, indexed indirect	O, I, S, B, T, C, R, N, A, ST, M	-32,768 to 32,767
		destination	direct, indexed direct indirect, indexed indirect	O, I, S, B, T, C, R, N, A, ST, M	none
OSR	One-Shot Rising	bit address	direct, indirect	O, I, S, B, T, C, R, N, A, ST	none
OTE	Output Energize	bit address	direct, indirect	O, I, S, B, T, C, R, N, A, ST, M	none
OTL	Output Latch	bit address	direct, indirect	O, I, S, B, T, C, R, N, A, ST, M	none
OTU	Output Unlatch	bit address	direct, indirect	O, I, S, B, T, C, R, N, A, ST, M	none

Instruction	Description	Parameter	Valid Addressing Mode(s)	Valid File Types	Immediate Values
PID[1]	PID	control block	direct	N	none
		process variable	direct, indirect	O, I, B, T, C, R, N, A	none
		control variable	direct, indirect	O, I, B, T, C, R, N, A	none
		control block length			23 always
RAD[2]	Degrees to Radians	source	direct, indexed direct indirect, indexed indirect	O, I, S, B, T, C, R, N, F, A, ST, M	-32,768 to 32,767 f-min to f-max
		destination	direct, indexed direct indirect, indexed indirect	O, I, S, B, T, C, R, N, F, A, ST, M	none
REF[1]	I/O Refresh	channel 0			0=no, 1=yes
		channel 1			0=no, 1=yes
RES	Timer/Counter Reset	structure	direct	T, C, R	none
RET	Return				none
RHC	Read High Speed Clock	destination	direct	N, F	
RMP	Ramp	control	direct	N	
		destination	direct	O, I, S, B, N	
RPI[1]	Reset Pending Interrupt	slots			double hex word (list of slots)
RTO	Retentive Timer On	timer	direct	T	none
		time base (SLC 5/01)			0.01 only
		time base (SLC 5/02, 5/03, 5/04, 5/05)			0.01 or 1.00
		preset			0 to 32,767
		accum			0 to 32,767
SBR	Subroutine				none
SCL[3]	Scale	source	direct, indexed direct indirect, indexed indirect	O, I, S, B, T, C, R, N, A, ST, M	none
		rate	direct, indexed direct indirect, indexed indirect	O, I, S, B, T, C, R, N, A, ST, M	-32,768 to 32,767
		offset	direct, indexed direct indirect, indexed indirect	O, I, S, B, T, C, R, N, A, ST, M	-32,768 to 32,767
		destination	direct, indexed direct indirect, indexed indirect	O, I, S, B, T, C, R, N, A, ST, M	none

Instruction	Description	Parameter	Valid Addressing Mode(s)	Valid File Types	Immediate Values
SCP[2]	Scale with Parameters	input	direct, indexed direct indirect, indexed indirect	O, I, S, B, T, C, R, N, F, A, ST, M	none
		input min.	direct, indexed direct indirect, indexed indirect	O, I, S, B, T, C, R, N, F, A, ST, M	-32,768 to 32,767 f-min to f-max
		input max.	direct, indexed direct indirect, indexed indirect	O, I, S, B, T, C, R, N, F, A, ST, M	-32,768 to 32,767 f-min to f-max
		scaled min.	direct, indexed direct indirect, indexed indirect	O, I, S, B, T, C, R, N, F, A, ST, M	-32,768 to 32,767 f-min to f-max
		scaled max.	direct, indexed direct indirect, indexed indirect	O, I, S, B, T, C, R, N, F, A, ST, M	-32,768 to 32,767 f-min to f-max
		scaled output	direct, indexed direct indirect, indexed indirect	O, I, S, B, T, C, R, N, F, A, ST, M	none
SIN[2]	Sine	source	direct, indexed direct indirect, indexed indirect	O, I, S, B, T, C, R, N, F, A, ST, M	-32,768 to 32,767 f-min to f-max
		destination	direct, indexed direct indirect, indexed indirect	O, I, S, B, T, C, R, N, F, A, ST, M	none
SQC	Sequencer Compare	file	indexed direct indexed indirect	O, I, S, B, N, A, ST	none
		mask	direct, indexed direct[4] indirect, indexed indirect	O, I, S, B, T, C, R, N, A, ST, M	-32,768 to 32,767
		source	direct, indexed direct[4] indirect, indexed indirect	O, I, S, B, T, C, R, N, A, ST, M	none
		control	direct	R	none
		length			1 to 255
		position			0 to 255
SQL[3]	Sequencer Load	file	indexed direct indexed indirect	O, I, S, B, N, A, ST	none
		source	direct, indexed direct[4] indirect, indexed indirect	O, I, S, B, T, C, R, N, A, ST, M	-32,768 to 32,767
		control	direct	R	none
		length			1 to 255
		position			0 to 255
SQO	Sequencer Output	file	indexed direct indexed indirect	O, I, S, B, N, A, ST	none
		mask	direct, indexed direct[4] indirect, indexed indirect	O, I, S, B, T, C, R, N, A, ST, M	-32,768 to 32,767
		destination	direct, indexed direct[4] indirect, indexed indirect	O, I, S, B, T, C, R, N, A, ST, M	none
		control	direct	R	none
		length			1 to 255
		position			0 to 255

Instruction	Description	Parameter	Valid Addressing Mode(s)	Valid File Types	Immediate Values
SQR[3]	Square Root	source	direct, indexed direct indirect, indexed indirect	O, I, S, B, T, C, R, N, F, A, ST, M	-32,768 to 32,767 f-min to f-max
		destination	direct, indexed direct indirect, indexed indirect	O, I, S, B, T, C, R, N, F, A, ST, M	none
STD[3]	Selectable Timed Interrupt Disable				none
STE[3]	Selectable Timed Interrupt Enable				none
STS[3]	Selectable Timed Interrupt Start	file	direct, indexed direct indirect, indexed indirect	O, I, S, B, T, C, R, N, A, ST, M	0, 3 to 255
		time	direct, indexed direct indirect, indexed indirect	O, I, S, B, T, C, R, N, A, ST, M	0 to 255 (SLC 5/02), 0 to 32,767 (SLC 5/03, 5/04, 5/05)
SUB	Subtract	source A	direct, indexed direct indirect, indexed indirect	O, I, S, B, T, C, R, N, F, A, ST, M	-32,768 to 32,76 f-min to f-max
		source B	direct, indexed direct indirect, indexed indirect	O, I, S, B, T, C, R, N, F, A, ST, M	-32,768 to 32,767 f-min to f-max
		destination	direct, indexed direct indirect, indexed indirect	O, I, S, B, T, C, R, N, F, A, ST, M	none
SUS	Suspend	suspend ID			-32,768 to 32,767
SVC[1]	Service Communications	channel 0 (SLC 5/03, 5/04, 5/05)			0=no, 1=yes
		channel 1 (SLC 5/03, 5/04, 5/05)			0=no, 1=yes
SWP[2]	Swap	source	indexed direct indexed indirect	B, N, A, ST	none
		length			1 to 128: bit, 1 to 128: integer, 1 to 41: string, 1 to 128: ASCII
TAN[2]	Tangent	source	direct, indexed direct indirect, indexed indirect	O, I, S, B, T, C, R, N, F, A, ST, M	-32,768 to 32,767 f-min to f-max
		destination	direct, indexed direct indirect, indexed indirect	O, I, S, B, T, C, R, N, F, A, ST, M	none
TDF	Compute Time Difference	start	direct	N, F	
		stop	direct	N, F	
		destination	direct	N, F	
TND	Temporary End				none

Instruction	Description	Parameter	Valid Addressing Mode(s)	Valid File Types	Immediate Values
TOD	Convert to BCD	source (SLC 5/01)	direct	O, I, S, B, T, C, R, N	none
		source (SLC 5/02, 5/03, 5/04, 5/05)	direct, indexed direct indirect, indexed indirect	O, I, S, B, T, C, R, N, A, ST, M	
		destination	direct	O, I. S. B. T, C, R, N, A, ST, M	none
TOF	Timer Off Delay	timer	direct	T	none
		time base (SLC 5/01)			0.01 only
		time base (SLC 5/02, 5/03, 5/04, 5/05)			0.01 or 1.00
		preset			0 to 32,767
		accum			0 to 32,767
TON	Timer on Delay time base (SLC 5/01) time base (SLC 5/02, 5/03, 5/04, 5/05) preset accum	timer	direct	T	none
					0.01 only
					0.01 or 1.00
					0 to 32,767
					0 to 32,767
XIC	Examine On (Examine if Closed Contact)	source bit	direct, indirect	O, I, S, B, T, C, R, N, A, ST, M	none
XIO	Examine Off (Examine if Open Contact)	source bit	direct, indirect	O, I, S, B, T, C, R, N, A, ST, M	none
XOR	Logical Exclusive OR	address A	direct, indexed direct indirect, indexed indirect	O, I, S, B, T, C, R, N, A, ST, M	-32,768 to 32,767
		address B	direct, indexed direct indirect, indexed indirect	O, I, S, B, T, C, R, N, A, ST, M	-32,768 to 32,767
		destination	direct, indexed direct indirect, indexed indirect	O, I, S, B, T, C, R, N, A, ST, M	none
XPY(2)	X to the Power of Y	source A	direct, indexed direct indirect, indexed indirect	O, I, S, B, T, C, R, N, F, A, ST, M	-32,768 to 32,767 f-min to f-max
		source B	direct, indexed direct indirect, indexed indirect	O, I, S, B, T, C, R, N, F, A, ST, M	-32,768-32,767 f-min-f-max
		destination	direct, indexed direct indirect, indexed indirect	O, I, S, B, T, C, R, N, F, A, ST, M	none

(1) Supported by SLC 5/02, SLC 5/03, SLC 5/04, and SLC 5/05 procesors.

(2) Supported by SLC 5/03 (OS302), SLC 5/04 (OS401), and SLC 5/05 processors.

(3) Not supported by SLC 5/01 processors and Flxed controllers.

(4) Indexed addressing is not allowed when using T, C, R, or M addresses.

Data File Organization and Addressing

This chapter discusses the following topics:

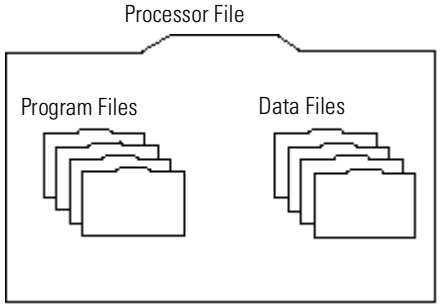
- data file organization and addressing
- specifying indexed addressing
- specifying indirect addressing
  (SLC 5/03 OS302, SLC 5/04 OS401, and SLC 5/05 processors)
- specifying indirect indexed addressing
  (SLC 5/03 OS302, SLC 5/04 OS401, and SLC 5/05 processors)
- addressing file instructions (using the file indicator #)
- numeric constants
- M0-M1 files, G files
  (SLC 5/02 and higher processors with specialty I/O modules)

Understanding File Organization

The processor provides control through the use of a program you create, called a processor file. This file contains other files that break your program down into more manageable parts.

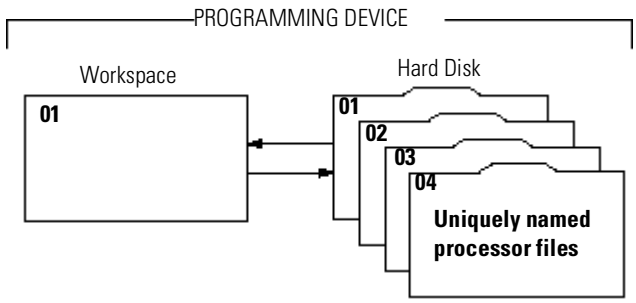
Processor File Overview

Most of the operations you perform with the programming device involve the processor file and the two components created with it: program files and data files.



The programming device stores processor files on *hard disk (or floppy disk)*. Monitoring and editing of processor files is done in the *workspace* of the computer. After you select a file from disk and edit it, you then *save* the file hard to disk, replacing the original disk

version with the edited version. The hard disk is the recommended location for a processor file.



Processor files are created in the offline mode using the programming device. These files are then restored (downloaded), to the processor for online operation.

Program Files

Program files contain controller information, the main ladder program, interrupt subroutines, and any subroutine programs. These files are:

- **System Program** (file 0) - This file contains various system related information and user-programmed information such as processor type, I/O configuration, processor file name, and password.
- **Reserved** (file 1) - This file is reserved.
- **Main Ladder Program** (file 2) - This file contains user-programmed instructions defining how the controller is to operate.
- **Subroutine Ladder Program** (file 3 to 255) - These files are user-created and accessed according to subroutine instructions residing in the main ladder program file.

Data Files

Data files contain the status information associated with external I/O and all other instructions you use in your main and subroutine ladder program files. In addition, these files store information concerning processor operation. You can also use the files to store “recipes” and look-up tables if needed.

These files are organized by the type of data they contain. The data file types are:

- **Output** (file 0) - This file stores the state of the output terminals for the controller.

- **Input** (file 1) - This file stores the status of the input terminals for the controller.
- **Status** (file 2) - This file stores controller operation information. This file is useful for troubleshooting controller and program operation.
- **Bit** (file 3) - This file is used for internal relay logic storage.
- **Timer** (file 4) - This file stores the timer accumulator and preset values and status bits.
- **Counter** (file 5) - This file stores the counter accumulator and preset values and the status bits.
- **Control** (file 6) - This file stores the length, pointer position, and status bits for specific instructions such as shift registers and sequencers.
- **Integer** (file 7) - This file is used to store numeric values or bit information.
- **Floating Point** (file 8) - This file stores single precision non-extended 32-bit numbers. Applies to SLC 5/03 (OS301 and higher), SLC 5/04, and SLC 5/05 processors.
- **String** (user-defined file) - Applies to SLC 5/03 (OS301 and higher), SLC 5/04, and SLC 5/05 processors.
- **ASCII** (user-defined file) - Applies to SLC 5/03 (OS301 and higher), SLC 5/04, and SLC 5/05 processors.

Addressing Data Files

For the purposes of addressing, each data file type is identified by a letter (identifier) and a file number.

File numbers 0 through 7 are the default files that fixed, SLC 5/01, SLC 5/02, and SLC 5/03 OS300 processors create for you. File number 8 applies only to SLC 5/03 (OS301 and higher), SLC 5/04, and SLC 5/05 processors. If you need additional storage, you can create files by specifying the appropriate identifier and a file number from 9 to 255. Refer to the tables below:

Table 5.A Data file types, identifiers, and numbers (data files in processor memory)

File Type	Identifier	File Number
Output	O	0
Input	I	1
Status	S	2
Bit	B	3
Timer	T	4
Counter	C	5
Control	R	6
Integer	N	7
Float	F	8

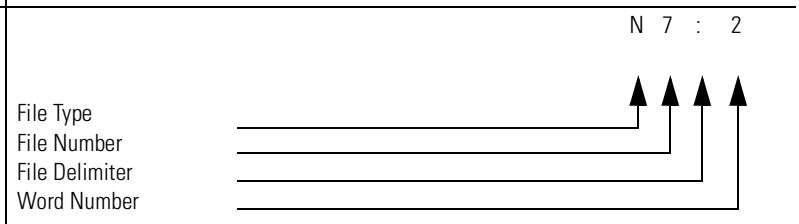
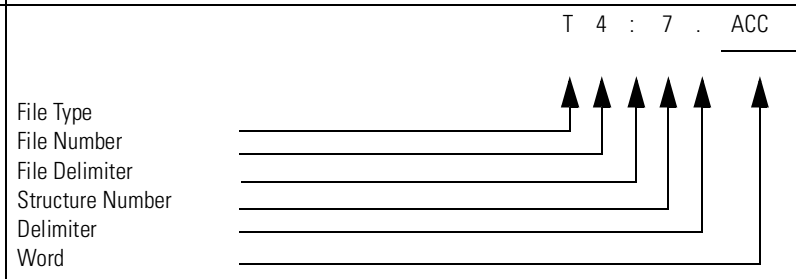
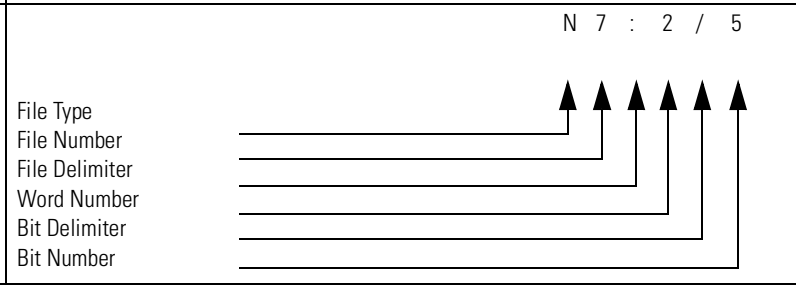
User-Defined Files		
File Type	**Identifier**	**File Number**
Bit	B	9 to 255
Timer	T	
Counter	C	
Control	R	
Integer	N	
Float	F	
String	St	
ASCII	A	

TIP

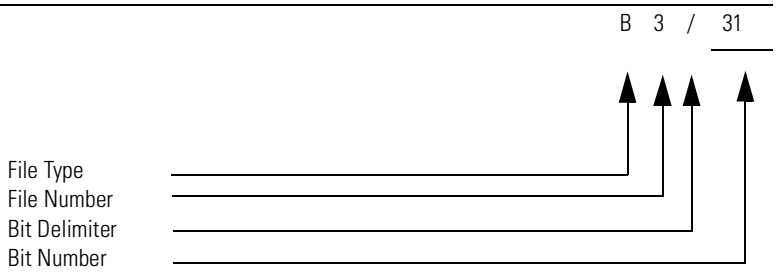
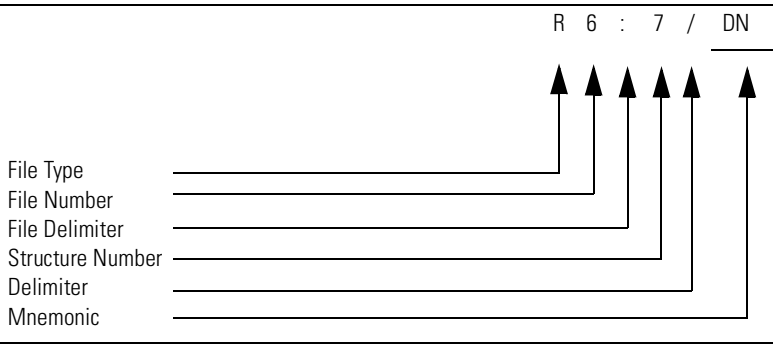
Floating point, string, and ASCII file types are only available when using SLC 5/03 (OS301 and higher), SLC 5/04, and SLC 5/05 processors.

Specifying Logical Addresses

You assign logical addresses to instructions from the highest level (element) to the lowest level (bit). Addressing examples are shown in the table below.

To specify the address of a:	Use these parameters:[1]
Word within an integer file	
Word within a structure file (e.g., a timer file)	
Bit within an integer file	

(1) Some programming devices support short addressing. This allows you to eliminate the file number and file delimiter from addresses. Consult your programming device's user manual for information on addressing capabilities. (For example: N7:2 = N2; T4:12.ACC = T12.ACC; B3:2/12 = B2/12)

To specify the address of a:	Use these parameters:[1]
Bit within a bit file	
Bit within an integer file	

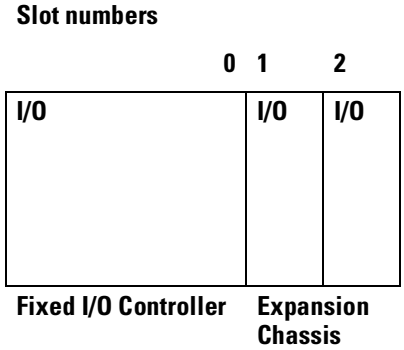
(1) Some programming devices support short addressing. This allows you to eliminate the file number and file delimiter from addresses. Consult your programming device's user manual for information on addressing capabilities. (For example: N7:2 = N2; T4:12.ACC = T12.ACC; B3:2/12 = B2/12)

You can also address at the bit level using mnemonics for timer, counter, or control data types. The available mnemonics depend on the type of data.

I/O Addressing for a Fixed I/O Controller

In the following figure, a fixed I/O controller has 24 inputs and 16 outputs. An expansion chassis has been added. Slot 1 of the chassis contains a module having 6 inputs and 6 outputs. Slot 2 contains a module having 8 outputs.

The following tables show how these outputs and inputs are arranged in data files 0 and 1. For these files, the element size is always 1 word.



Slot	Inputs	Outputs
0	24	16
1	6	6
2	None	8

Table 5.B Data File 0 - Output Image

	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00	
Slot 0 Outputs (0 to 15)												x					O:0
Slot 1 Outputs (0 to 5)	INVALID																O:1
Slot 2 Outputs (0 to 7)									x								O:2

Table 5.C Data File 1 - Input Image

	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00	
Slot 0 Inputs (0 to 15)	x											x					I:0
Slot 0 Inputs (16 to 23)	INVALID								x								I:0.1
Slot 1 Inputs (0 to 5)	INVALID											x					I:1

The table on the following page explains the addressing format for outputs and inputs. Note that the format specifies e as the slot number and s as the word number. When you are dealing with file instructions, refer to the element as e.s (slot and word), taken together.

Assign I/O addresses to fixed I/O controllers as shown in the following table:

Format	Explanation		
O:e.s/b **I:e.s/b**	**O**	Output	
	I	Input	
	:	Element delimiter	
		Slot number (decimal)	Fixed I/O controller: 0
	e		Left slot of expansion chassis: 1 Right slot of expansion chassis: 2
	.	Word delimiter. Required only if a word number is necessary as noted below.	
	s	Word number	Required if the number of inputs or outputs exceeds 16 for the slot. Range: 0 to 255 (range accommodates multi-word "specialty cards")
	/	Bit delimiter	
	b	Terminal number	Inputs: 0 to 15 (or 0 to 23, slot 0) Outputs: 0 to 15

Examples (applicable to the controller shown on page F-10):

O:0/4	Controller output 4 (slot 0)
O:2/7	Output 7, slot 2 of the expansion chassis
I:1/4	Input 4, slot 1 of the expansion chassis
I:0/15	Controller input 15 (slot 0)
I:0.1/7	Controller input 23 (bit 07, word 1 of slot 0)

Alternate way of addressing I/O terminals 16 and higher: As indicated above, address I:0.1/7 applies to input terminal 23 of slot 0. You can also address this terminal as I:0/23.

Word addresses:

O:1	Output word 0, slot 1
I:0	Input word 0, slot 0
I:0.1	Input word 1, slot 0

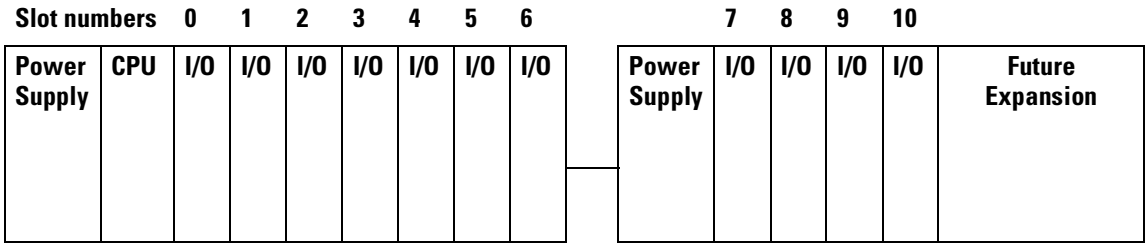
Default Values: Your programming device will display an address more formally. For example, when you assign the address I:1/4, the programming device will show it as I:1.0/4 (input file, slot 1, word 0, terminal 4).

I/O Addressing for a Modular Controller

With modular controllers, slot number 0 is reserved for the processor module (CPU). Slot 0 is invalid as an I/O slot.

The figure below shows a modular controller configuration consisting of a 7-slot chassis interconnected with a 10-slot chassis. Slot 0 contains the CPU. Slots 1 through 10 contain I/O modules. The remaining slots are saved for future I/O expansion.

The figure indicates the number of inputs and outputs in each slot and also shows how these inputs and outputs are arranged in the data files. For these files, the element size is always 1 word.



Modular controller using a 7-slot chassis interconnected with a 10-slot chassis.

Slot	Inputs	Outputs
1	6	6
2	32	None
3	None	16
4	8	8
5	None	32
6	16	None
7	16	None
8	8	None
9	None	16
10	None	16

Data File 0 - Output Image

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Slot 1 outputs (0 to 5)	INVALID																**O:0**
Slot 3 outputs (0 to 15)	x																**O:3**
Slot 4 outputs (0 to 7)	INVALID																**O:4**
Slot 5, word 0 outputs (0 to 15)																x	**O:5**
Slot 5, word 1 outputs (0 to 15)																	**O:5.1**
Slot 9 outputs (0 to 15)																	**O:9**
Slot 10 outputs (0 to 15)					x												**O:10**

Data File 0 - Input Image

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Slot 1 inputs (0 to 5)	INVALID																**I:1**
Slot 2, word 0 inputs (0 to 15)	x																**I:2**
Slot 2, word 1 inputs (0 to 15)													x				**I:2.1**
Slot 4 inputs (0 to 7)	INVALID															x	**I:4**
Slot 6 inputs (0 to 15)																	**I:6**
Slot 7 inputs (0 to 15)							x										**I:7**
Slot 8 outputs (0 to 7)	INVALID																**I:8**

x = See the examples on the next page.

Specifying Indexed Addresses

The indexed address symbol is the # character. Place the # character immediately before the file-type identifier in a logical address. You can use more than one indexed address in your ladder program.

Enter the offset value in word 24 of the status file (S:24). All indexed instructions use the same word S:24 to store the offset value. The processor starts operation at the base address plus the offset. You can manipulate the offset value in your ladder logic before each indexed address operation.

When you specify indexed addresses, follow these guidelines:

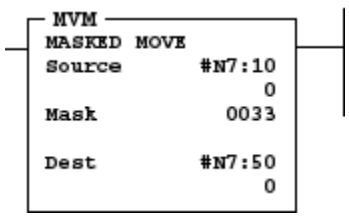
- Make sure the index value (positive or negative) does not cause the indexed address to exceed the file type boundary.
- When an instruction uses more than two indexed addresses, the processor uses the same index value for each indexed address.
- Set the index word to the offset value you want immediately before enabling an instruction that uses an indexed address.

ATTENTION

Instructions with a # sign in an address manipulate the offset value stored at S:24. Make sure you monitor or load the offset value you want prior to using an indexed address. Otherwise unpredictable machine operation could occur with possible damage to equipment and/or injury to personnel.

Example of Indexed Addressing

The following Masked Move (MVM) example uses an indexed address in the source and destination addresses. If the offset value is 10 (stored in S:24), the processor manipulates the data stored at the base address plus the offset.



In this example, the processor uses the following addresses:

Value:	Base Address:	Offset Value in S:24	Offset Address:
Source	N7:10	10	N7:20
Destination	N7:50	10	N7:60

SLC 5/03 (OS301 and higher), SLC 5/04, and SLC 5/05 processors: If the indexed address is a floating point (F8:) data file, then the index offset value in S:24 is the offset in elements. If the indexed address is a string (ST) data file, then the index offset value in S:24 is the offset in sub-elements. This limits string element boundaries from being crossed.

Note that file instructions (SQO, COP, LFL for example) overwrite S:24 when they execute. For this reason, you must insure that the index register is loaded with the intended value prior to the execution of an indexed instruction that follows a file instruction.

Creating Data for Indexed Addresses

Data tables are not expanded automatically to accommodate indexed addresses. You must create this data with the memory map function. In the example on the previous page, data words N7:3 through N7:12 and N11:6 through N11:15 must be allocated. Failure to do so will result in an unintended overwrite condition or a major fault.

Crossing File Boundaries

An offset value may extend operation to an address outside the data file boundary. You can either allow or disallow crossing file boundaries. If you choose to disallow crossing file boundaries, a runtime error occurs if you use an offset value which would result in crossing a file boundary.

SLC 5/02 processors: You are allowed to select crossing file boundaries only if no indexed addresses exist in the O: (output), I: (input), or S: (status) files. This selection is made at the time you save your program. The file order from start to finish is:

- O0:, I1:, S2:, B3:, T4:, C5:, R6:, N7:, x9:, x10: . . .
- x9: and x10: . . . are application-specific files where x can be of types B, T, C, R, N.

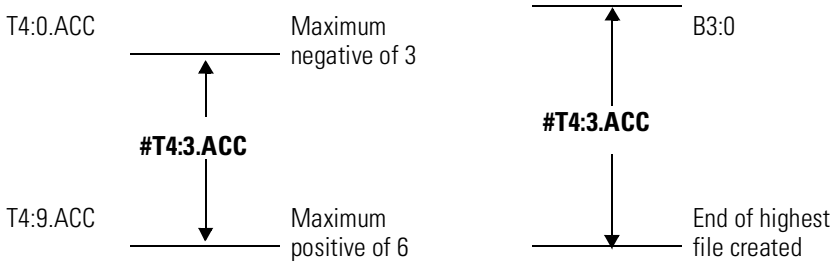
SLC 5/03 (OS301 and higher) SLC 5/04, and SLC 5/05 processors: When an indexed string data file is specified, indexed addressing is not allowed to cross a string element boundary. A run-time error will

occur if you use an offset value that results in crossing a string element boundary.

TIP If a file is constant protected, indexing across file boundaries is not allowed.

Example

The figure below indicates the maximum offset for word address #T4:3.ACC when allowing and disallowing crossing file boundaries.



Crossing file boundaries disallowed

Crossing file boundaries allowed

Crossing file boundaries disallowed: In the example above, the highest numbered element in the timer data file is T4:9. This means that #T4:3.ACC can have a maximum negative offset of -3 and a maximum positive offset of 6.

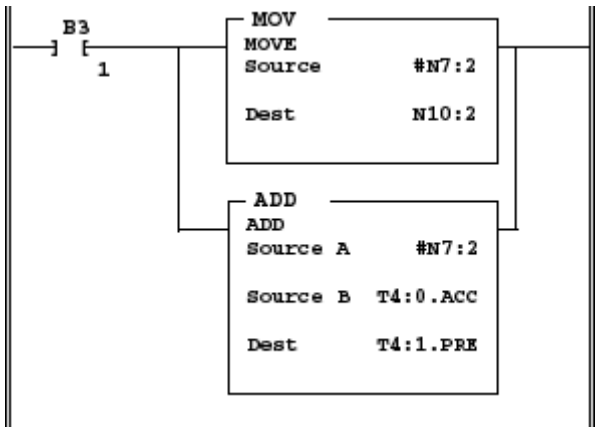
Crossing file boundaries allowed: The maximum negative offset extends to the beginning of data file 3. The maximum positive offset extends to the end of the highest numbered file created.

Monitoring Indexed Addresses

The offset address value is not displayed when you monitor an indexed address. For example, the value at N7:2 appears when you monitor indexed address #N7:2.

Example

If your application requires you to monitor indexed data, we recommend that you use a MOV instruction to store the value.



N10:2 will contain the data value that was added to T4:0.ACC.

File Instructions

The # symbol is also required for addresses in file instructions. The indexed addresses used in these file instructions also make use of word S:24 to store an offset value upon file instruction completion. Refer to the next page for a list of file instructions that use the # symbol for addressing.

ATTENTION

File instructions manipulate the offset value stored in word S:24. Make sure that you load the correct offset value in S:24 prior to using an indexed address that follows a file instruction. Otherwise, unpredictable operation could occur, resulting in possible personal injury and/or damage to equipment.

Effects of Program Interrupts on Index Register S:24

When normal program operation is interrupted by the user error handler, an STI, or an I/O interrupt, the content of index register S:24 is saved; then, when normal program operation is resumed, the content of index register S:24 is restored. This means that if you alter the value in S:24 in these interrupt subroutines, the system will overwrite your alteration with the original value contained on subroutine entry.

Specifying an Indirect Address

Indirect addressing allows you to write less complex ladder logic programs and saves you memory space. You have the option of using word-level and bit-level indirect addresses when using an SLC 5/03 (OS302), SLC 5/04 (OS401), and SLC 5/05 processors. Indirect bit addresses are based on the form of the indirect address and the type of bit instruction.

Use indirect addressing for applications such as indexing sequential batch files in a multiple batch operation. For example, at completion of each operation, let a counter accumulated value call out the next batch file, such as:
N10, N11, N12,...N[C5:0.ACC].

When you specify indirect addresses, follow these guidelines:

- You can indirectly address:
  - file number
  - word number (element + subelement)
  - bit number (in a binary file)
- The substitute address must be any address specified to the word level.
- Enter the substitute address in brackets [ ].

Examples

Valid Address	Variable	Explanation
N7:[C5:7.ACC]	Word number	The word number is the accumulated value of counter 7 in file 5.
B3/[I:0.17]	Bit number	The bit number is stored in input word 17.
N[N7:0]:[N9:1]	File and word number	The file number is stored in integer address N7:0 and the word number in integer address N9:1.
St10:[N7:0].1	Element number	The element number is stored in N7:0.
I:[N7:0].1/1	Slot number	The slot number is stored in N7:0.

Creating Data for Indirect Addresses

Data tables are not expanded automatically to accommodate indirect addresses. You must create this data with your programming software.

Crossing File Boundaries

Crossing file boundaries is not allowed. A runtime error occurs if you use an offset value which would result in crossing a file boundary.

Monitoring Indirect Addresses

An asterisk is displayed at all times when monitoring an indirect address.

Addressing File Instructions - Using the File Indicator (#)

File instructions employ user-created files. These files are addressed with the # sign. They store an offset value in word S:24, just as with indexed addressing discussed in the last section.

COP Copy File
FLL Fill File
BSL Bit Shift Left
BSR Bit Shift Right
FFL (FIFO Load)*
FFU (FIFO Unload)*
LFL (LIFO Load)*
LFU (LIFO Unload)*
SQO Sequencer Output
SQC Sequencer Compare
SQL Sequencer Load*

* Available in the SLC 5/02 and higher processors.

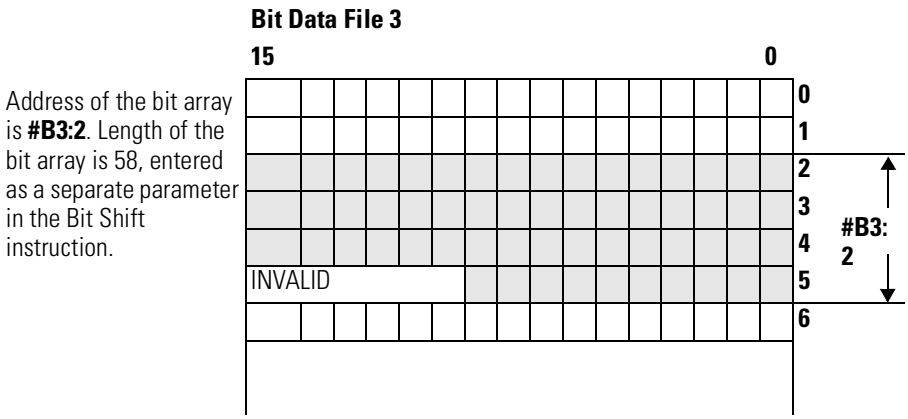
ATTENTION

If you are using file instructions and also indexed addressing, make sure that you monitor and/or load the correct offset value prior to using an indexed address. Otherwise, unpredictable operation could occur, resulting in possible personal injury and/or damage to equipment.

The following paragraphs explain user-created files as they apply to Bit Shift instructions, Sequencer instructions, Copy File, and Fill File instructions.

Bit Shift Instructions

The following figure shows a user-defined file within bit data file 3. For this particular user-defined file, you would enter the following parameters when programming the instruction:

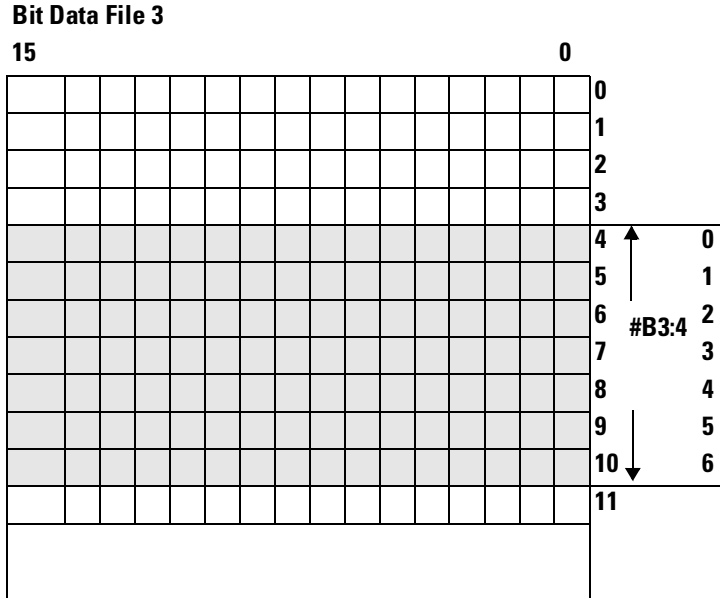


- **#B3:2** The address of the bit array. This defines the starting bit as bit 0 in element 2, data file 3.
- **58** This is the length of the bit array, 58 bits. Note that the bits "left over" in element 5 are unusable.

You can program as many bit arrays as you like in a bit file. Be careful that they do not overlap.

Sequencer Instructions

The following figure shows a user-defined file within bit data file 3. For this particular user-defined file, you would enter the following parameters when programming the instruction:



Address of the user-defined file is #B3:4.
Length of the file is 6 elements beyond the starting address (elements labeled 0-6 in the diagram).

- **#B3:4** The address of the file. This defines the starting element as element 4, bit file 3.
- **6** The specified length of the file, 6 elements *beyond* the starting address (totals 7 elements).

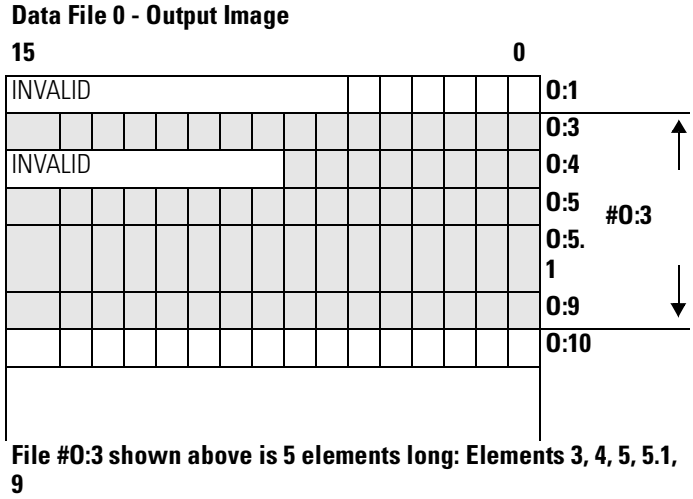
You can use user-defined integer files or bit files with sequencer instructions, depending on the application. You can program as many files as you like within another file. However, be careful that the files do not overlap.

Copy File and Fill File Instructions

These instructions manipulate user-defined files. The files are used as source or destination parameters in Copy File or Fill File instructions. Files can be Output, Input, Status, Bit, Timer, Counter, Control, or Integer files. Two examples are shown in the following figure. Note that the file length is the specified number of elements of the destination file; this differs from the file length specification for sequencer instructions.

The following figure shows a user-defined file within Data File 0-Output Image. We used this particular data file configuration in regard to I/O addressing on page B-12. Here, we are defining a file 5 elements long.

Note that for the output file (and the input file as well), an element is always one word, referenced as the slot and word taken together. For example, element **O:3.0** refers to output file, slot 3, word 0. This defaults to **O:3**, where word 0 is implied.



File #O:3 shown above is 5 elements long: Elements 3, 4, 5, 5.1, 9

Numeric Constants

You can enter numeric constants directly into many of the instructions you program. The range of values for most instructions is -32,768 through +32,767. These values can be displayed or entered in several radixes. The radixes that can be *displayed are:*

- Integer
- Binary
- ASCII
- Hexadecimal

When *entering* values into an instruction or data table element, you can specify the radix of your entry using the appropriate suffix. The radixes that can be used to enter data into an instruction or data table element are:

- Integer (D)
- Binary (B)
- Hexadecimal (H)
- Octal (O)

Numeric constants are used in place of data file elements. They cannot be manipulated by the user program. You must enter the offline program editor to change the value of a constant.

M0 and M1 Data Files - Specialty I/O Modules

M0 and M1 files are data files that reside in specialty I/O modules only. There is no image for these files in the processor memory. The application of these files depends on the function of the particular specialty I/O module. For some modules, the M0 file is regarded as a module output file and the M1 file is regarded as a module input file. In any case, both M0 and M1 files are considered read/write files by the SLC 5/02 and higher processors.

M0 and M1 files can be addressed in your ladder program and they can also be acted upon by the specialty I/O module-independent of the processor scan. It is important that you keep the following in mind in creating and applying your ladder logic:

TIP

During the processor scan, M0 and M1 data can be changed by the processor according to ladder diagram instructions addressing the M0 and M1 files. During the same scan, the specialty I/O module can change M0 and M1 data, independent of the rung logic applied during the scan.

Addressing M0-M1 Files

The addressing format for M0 and M1 files is below:

Mf:e.s/b

Where
M = module
f = file type (0 or 1)
e = slot (1 to 30)
s = word (0 to max. supplied by module)
b = bit (0 to 15)

Restrictions on Using M0 and M1 Data File Addresses

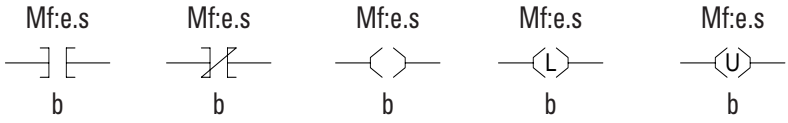
M0 and M1 data file addresses can be used in all instructions except the OSR instruction and the instruction parameters noted below:

Instruction	Parameter (uses file indicator #)
BSL, BSR	File (bit array)
SQO, SQC, SQL	File (sequencer file)
LFL, LFU	LIFO (stack)
FFL, FFU	FIFO (stack)

Monitoring Bit Addresses

SLC 5/02 and Higher Processors with M0 and M1 Monitoring Disabled

When you monitor a ladder program in the run or test mode, the following bit instructions, addressed to an M0 or M1 file, are indicated as false regardless of their actual true/false logical state.



f = file (0 or 1)

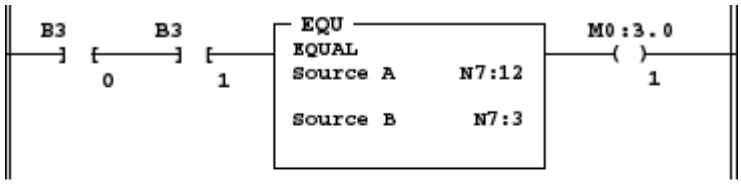
When you are monitoring the ladder program in the run or test mode, the programming terminal does not show these instructions as being true when the processor evaluates them as true.

SLC 5/03 and Higher Processors

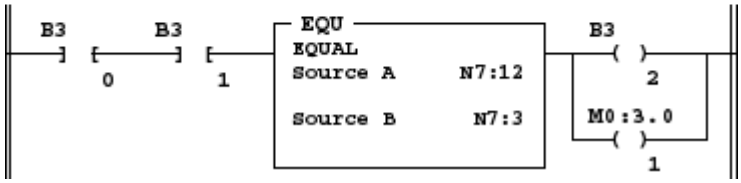
The SLC 5/03 and higher processors allow you to monitor the actual state of each addressed M0/M1 address (or data table). The highlighting appears normal when compared to the other processor data file. The SLC 5/03's performance is degraded to the degree of M0/M1 referenced screen data. For example, if your screen has only one M0/M1 element, degradation is minimal. If your screen has 69 M0/M1 elements, degradation is significant.

If you need to show the state of the M0 or M1 addressed bit, you can transfer the state to an internal processor bit. This is illustrated in the following figure, where an internal processor bit is used to indicate the true/false state of a rung.

This rung will not show its true rungstate because the EQU instruction is always shown as true and the M0 instruction is always shown as false.



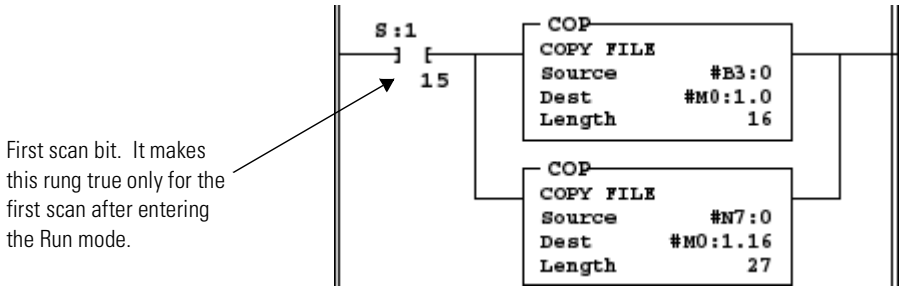
OTE instruction B3/2 has been added to the rung. This instruction shows the true or false state of the rung.



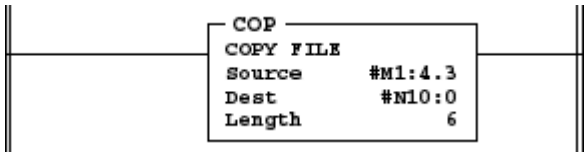
Transferring Data Between Processor Files and M0 or M1 Files

As pointed out earlier, the processor does not contain an image of the M0 or M1 file. As a result, you must edit and monitor M0 and M1 file data via instructions in your ladder program. For example, you can copy a block of data from a processor data file to an M0 or M1 data file or vice versa using the COP instruction in your ladder program.

The COP instructions below copy data from a processor bit file and integer file to an M0 file. Suppose the data is configuration information affecting the operation of the specialty I/O module.



The COP instruction that follows copies data form an M1 data file to an integer file. This technique is used to monitor the contents of an M0 or M1 data file indirectly, in a processor data file.

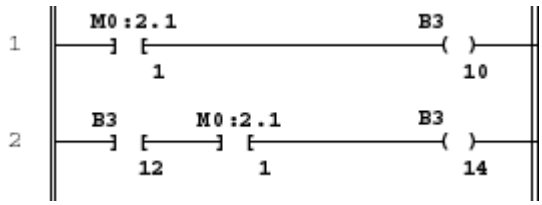


Access Time

During the program scan, the processor must access the specialty I/O card to read/write M0 or M1 data. This access time must be added to the execution time of each instruction referencing M0 or M1 data.

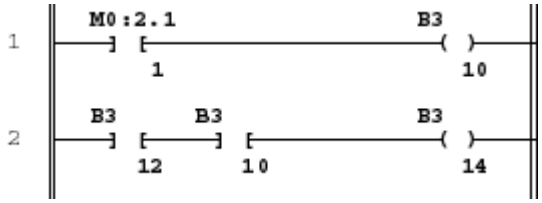
Minimizing the Scan Time

You can keep the processor scan time to a minimum by economizing on the use of instructions addressing the M0 or M1 files. For example, XIC instruction M0:2.1/1 is used in rungs 1 and 2 of the figure below, adding approximately 2 ms to the scan time if you are using an SLC 5/02, Series B processor.



XIC instructions in rungs 1 and 2 are addressed to the M0 data file. Each of these instructions adds approximately 1 ms to the scan time (SLC 5/02, Series B processor).

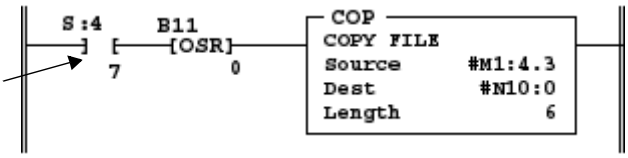
In the equivalent rungs of the following figure, XIC instruction M0:2.1/1 is used only in rung 1, reducing the scan time by approximately 1 ms.



These rungs provide equivalent operation to those of figure A by substituting XIC instruction B3/10 for XIC instruction M0:2.1/1 in rung 2. Scan time is reduced by approximately 1 ms (SLC 5/02, Series B processor).

The following figure illustrates another economizing technique. The COP instruction addresses an M1 file, adding approximately 4.29 ms to the scan time if you are using an SLC 5/02, Series B processor. Scan time economy is realized by making this rung true only periodically, as determined by clock bit S:4/8. (Clock bits are discussed in appendix in this manual.) A rung such as this might be used when you want to monitor the contents of the M1 file, but monitoring need not be on a continuous basis.

S:4/7 causes the #M1:4.3 file to update the #N10:0 file every 2.56 seconds.



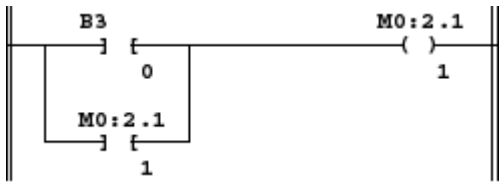
Capturing M0-M1 File Data

The first two ladder diagrams in the last section illustrate a technique allowing you to capture and use M0 or M1 data as it exists at a particular time. In the first figure, bit M0:2.1/1 could change state between rungs 1 and 2. This could interfere with the logic applied in rung 2. The second figure avoids the problem. If rung 1 is true, bit B3/10 captures this information and places it in rung 2.

In the second example of the last section, a COP instruction is used to monitor the contents of an M1 file. When the instruction goes true, the 6 words of data in file #M1:4.3 is captured as it exists at that time and placed in file #N10.0.

Specialty I/O Modules with Retentive Memory

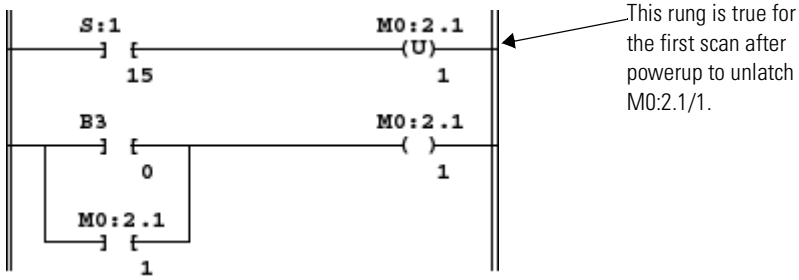
Certain specialty I/O modules retain the status of M0-M1 data after power is removed. See your specialty I/O module user's manual. This means that an OTE instruction having an M0 or M1 address remains on if it is on when power is removed. A "hold-in" rung as shown below will not function as it would if the OTE instruction were non-retentive on power loss. If the rung is true at the time power is removed, the OTE instruction latches instead of dropping out; when power is again applied, the rung is evaluated as true instead of false.



ATTENTION

When used with a specialty I/O module having retentive outputs, this rung can cause unexpected start-up on powerup.

You can achieve non-retentive operation by unlatching the retentive output with the first pass bit at powerup:



G Data Files - Specialty I/O Modules

Some specialty I/O modules use G (confiGuration) files (indicated in the specific specialty I/O module user's manual). These files can be thought of as the software equivalent of DIP switches.

The content of G files is accessed and edited offline under the I/O Configuration function. You cannot access G files under the Monitor File function. Data you enter into the G file is passed on to the specialty I/O module when you download the processor file and enter the REM Run or any one of the REM Test modes.

Editing G File Data

Data in the G file must be edited according to your application and the requirements of the specialty I/O module. You edit the data offline under the I/O configuration function only. With the decimal and hex/bcd formats, you edit data at the word level:

- G1:1 = 234 (decimal format)
  G1:1 = 00EA (hex/bcd format)

TIP

Word 0 of the G file is configured automatically by the processor according to the particular specialty I/O module. Word 0 cannot be edited.

Number Systems

This appendix:

- covers binary and hexadecimal numbers
- explains the use of a hex mask to filter data in certain programming instructions

Binary Numbers

The processor memory stores 16-bit binary numbers. As indicated in the following figure, each position in the number has a decimal value, beginning at the right with 2^0 and ending at the left with 2^{15}.

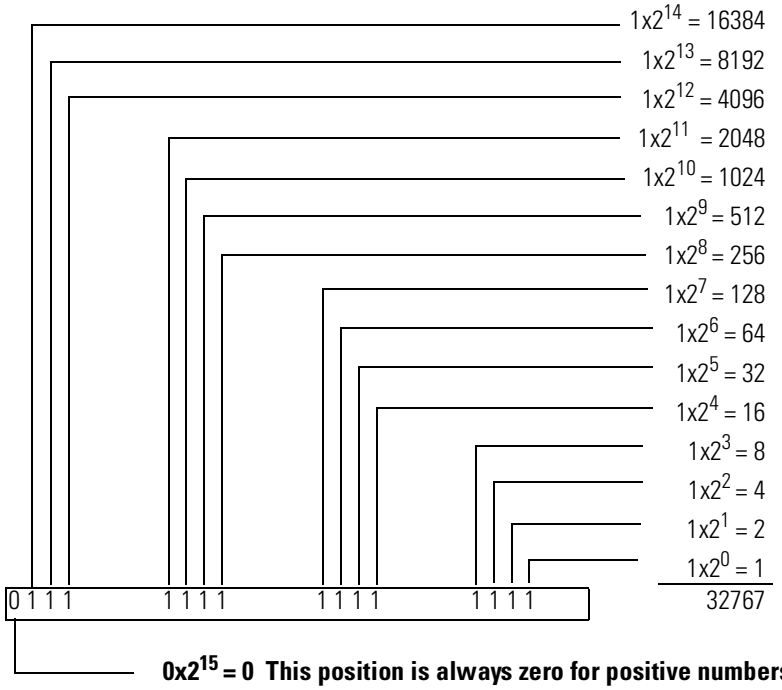
Each position can be 0 or 1 in the processor memory. A 0 indicates a value of 0; a 1 indicates the decimal value of the position. The equivalent decimal value of the binary number is the sum of the position values.

Positive Decimal Values

The far left position will always be 0 for positive values. As indicated in the figure, this limits the maximum positive decimal value to 32767. All positions are 1 except the far left position.

Other examples:

0000 1001 0000 1110	$= 2^{11}+2^8+2^3+2^2+2^1$
	= 2048+256+8+4+2 = 2318
0010 0011 0010 1000	$= 2^{13}+2^9+2^8+2^5+2^3$
	= 8192+512+256+32+8
	= 9000



Negative Decimal Values

The 2s complement notation is used. The far left position is always 1 for negative values. The equivalent decimal value of the binary number is obtained by subtracting the value of the far left position, 32768, from the sum of the values of the other positions. In the following figure, the value is 32767 - 32768 = -1. All positions are 1.

Another example:

1111 1000 0010 0011 =

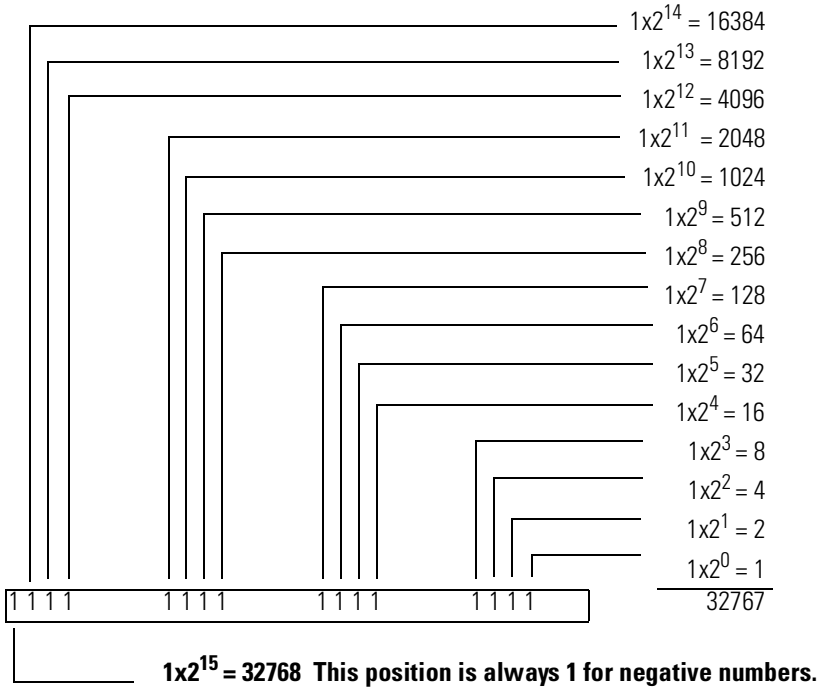
$(2^{14}+2^{13}+2^{12}+2^{11}+2^{5}+2^{1}+2^{0})$ - 215 =

(16384+8192+4096+2048+32+2+1) - 32768 =

30755 - 32768 = -2013.

An often easier way to calculate a value is to locate the last 1 in the string of 1s beginning at the left, and subtract its value from the total value of positions to the right of that position. For example,

1111 1111 0001 1010 = (24+23+21) - 28 = (16+8+2) - 256 = -230.



Hexadecimal Numbers

Hexadecimal numbers use single characters with equivalent decimal values ranging from 0 to 15:

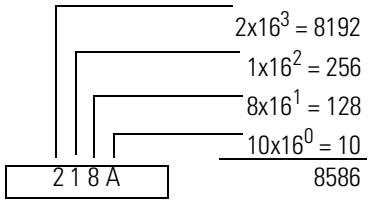
HEX	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
Decimal	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

The position values of hexadecimal numbers are powers of 16, beginning with 16^0 at the right:

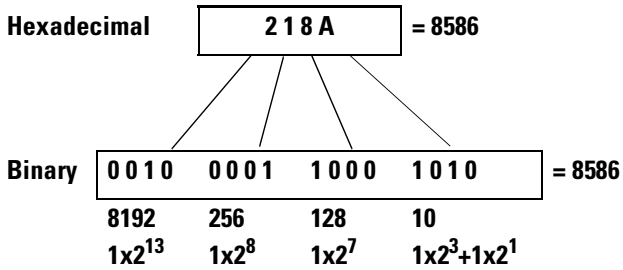
16^3	16^2	16^1	16^0

Example

Hexadecimal number 218A has a decimal equivalent value of 8586:

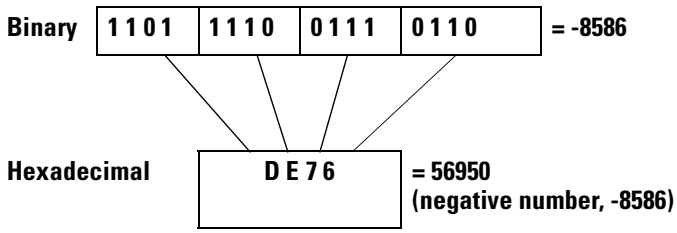


Hexadecimal and binary numbers have the following equivalence:



Example

Decimal number -8586 in equivalent binary and hexadecimal forms:

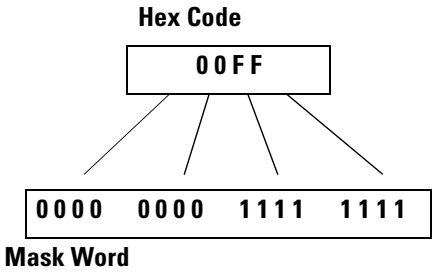


Hexadecimal number DE76 = $13x16^3+14x16^2+7x16^1+6x16^0$ = 56950. We know this is a negative number because it exceeds the maximum positive value of 32767. To calculate its value, subtract 16^4 (the next higher power of 16) from 56950: 56950 - 65536 = -8586.

Hex Mask

This is a 4-character code, entered as a parameter in SQO, SQC, and other instructions to exclude selected bits of a word from being operated on by the instruction. The hexadecimal values are used in their binary equivalent form, as indicated in the figure below. The figure also shows an example of a hexadecimal code and the corresponding mask word.

Hex Value	Binary Value
0	0000
1	0001
2	0010
3	0011
4	0100
5	0101
6	0110
7	0111
8	1000
9	1001
A	1010
B	1011
C	1100
D	1101
E	1110
F	1111



Bits of the mask word that are set (1) will pass data from a source to a destination. Reset bits (0) will not. In the example below, data in bits 0-7 of the source word is passed to the destination word. Data in bits 8-15 of the source word is not passed to the destination word.

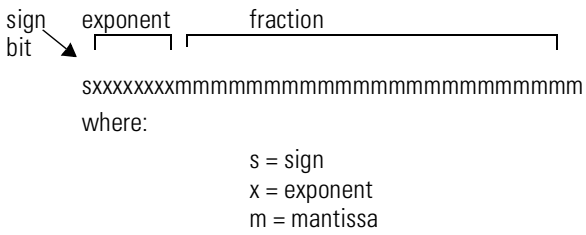
Source Word	**1110 1001 1100 1010**
Mask Word	**0000 0000 1111 1111**
Destination Word (all bits 0 initially)	**0000 0000 1100 1010**

Binary Floating-Point Arithmetic

The SLC 5/03, OS301 and higher, SLC 5/04, and SLC 5/05 processors support the use of floating-point. Use floating-point when you want to manipulate numbers outside of the range –32768 to +32767 or for a resolution finer than one unit. For example, 2.075. Floating-point arithmetic does not support unnormalized, Not a Number (NaN), and infinity. The valid range for a floating-point number is ±3.40282 x 10^{38} to ±1.17550 x 10^{-38}

The following example shows the representation of a floating-point number using the IEEE 754 standard for Single precision floating-point.

The following is the spatial representation of the 32 bits in the register.



When converting to floating-point arithmetic, the following must occur:

1. The sign bit must be set. If the number is positive, then the sign bit is 0 or Off. If the number is negative, then the sign bit is 1 or On.

2. The exponent must be normalized. Do this by always adding +127 to the exponent.

3. The mantissa must be normalized. For example, the binary value of 1010.01 equals 1.01001

4. The fraction must be extracted from the mantissa. For example, the fractional part of 1.01001 is .01001.

The 32-bit floating-point representation of 10.25 decimal equals:

0 10000010 01001000000000000000000

Application Example Programs

This appendix is designed to illustrate various instructions described previously in this manual. Application example programs include:

- paper drilling machine using most of the instructions
- time driven sequencer using TON and SQO instructions
- event driven sequencer using SQC and SQO instructions
- on/off circuit using basic, program flow, and application specific instructions
- interfacing with enhanced bar code decoders over DH-485

Because of the variety of uses for this information, the user of and those responsible for applying this information must satisfy themselves as to the acceptability of each application and use of the program. In no event will Allen-Bradley Company be responsible or liable for indirect or consequential damages resulting from the use of application of this information.

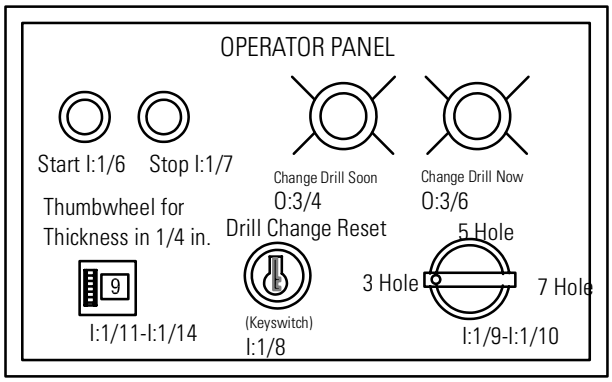
The illustrations, chart, and examples shown in this appendix are intended solely to illustrate the principles of the controller and some of the methods used to apply them. Particularly because of the many requirements associated with any particular installation, Allen-Bradley Company cannot assume responsibility or liability for actual use based upon the illustrative uses and applications.

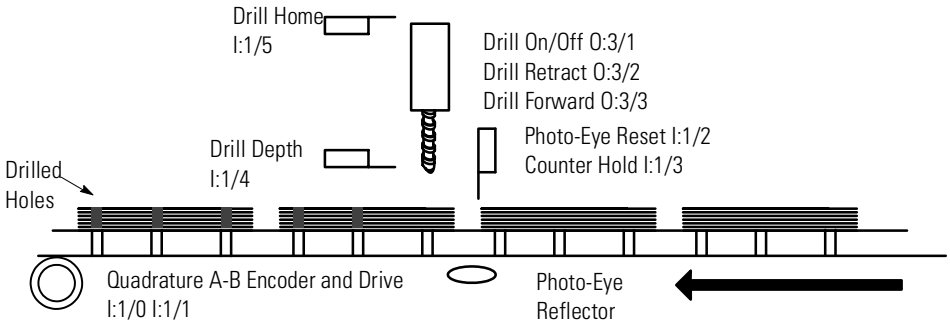
Paper Drilling Machine Application Example

For a detailed explanation of:

- XIC, XIO, OTE, RES, OTU, OTL, and OSR instructions, see Chapter 2.
- EQU and GEQ instructions, see Chapter 3.
- CLR, ADD, and SUB instructions, see Chapter 4.
- MOV and FRD instructions, see Chapter 5.
- JSR and RET instructions, see Chapter 6.
- INT instructions, see Chapter 11
- SQO instructions, see Chapter 7.

This machine can drill 3 different hole patterns into bound manuals. The program tracks drill wear and signals the operator that the bit needs replacement. The machine shuts down if the signal is ignored by the operator.

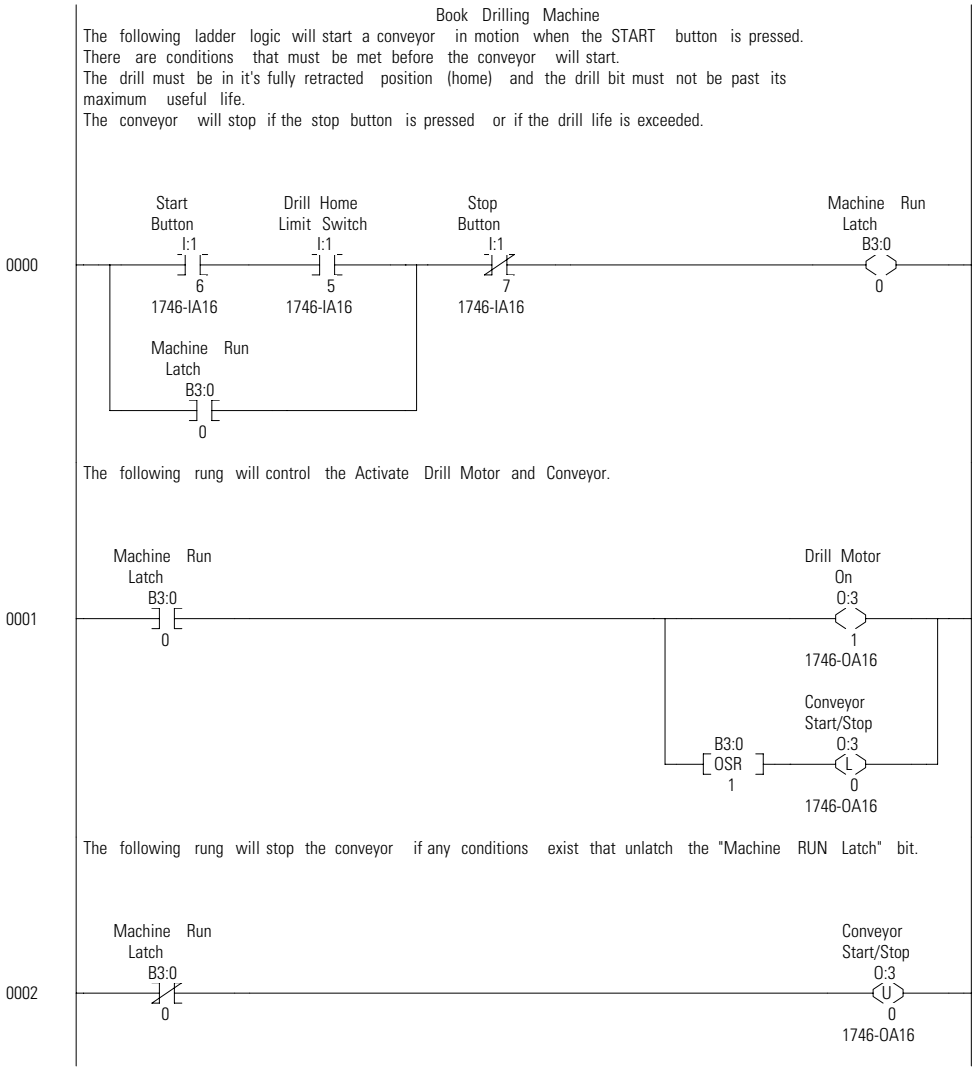


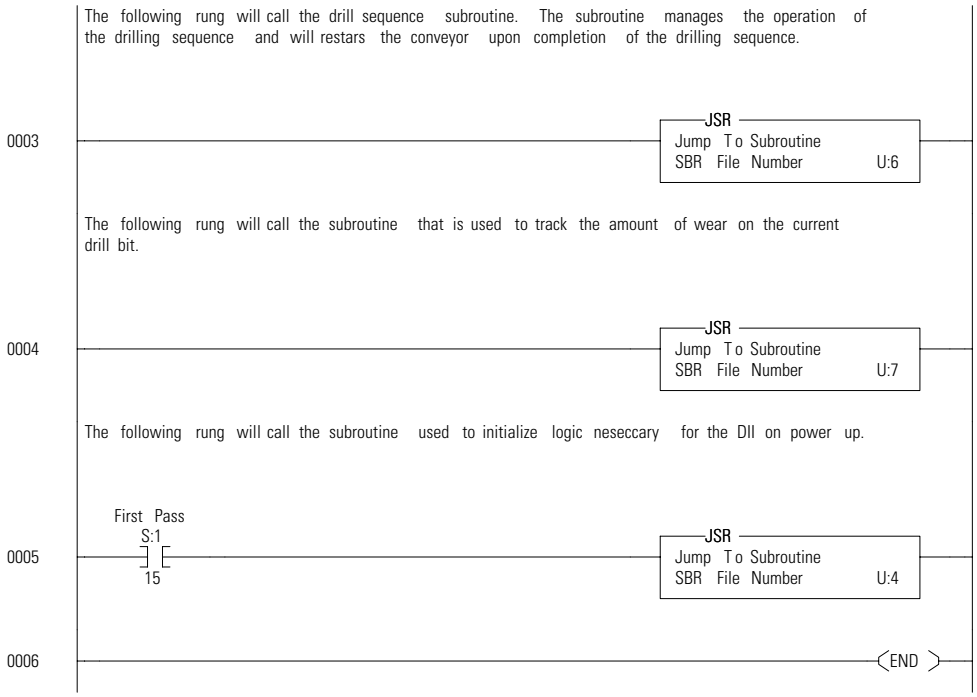


Conveyor Enable wired in series to the Drive O:3/5
Conveyor Drive Start/Stop wired in series to the Drive O:3/0

Paper Drilling Machine Operation Overview

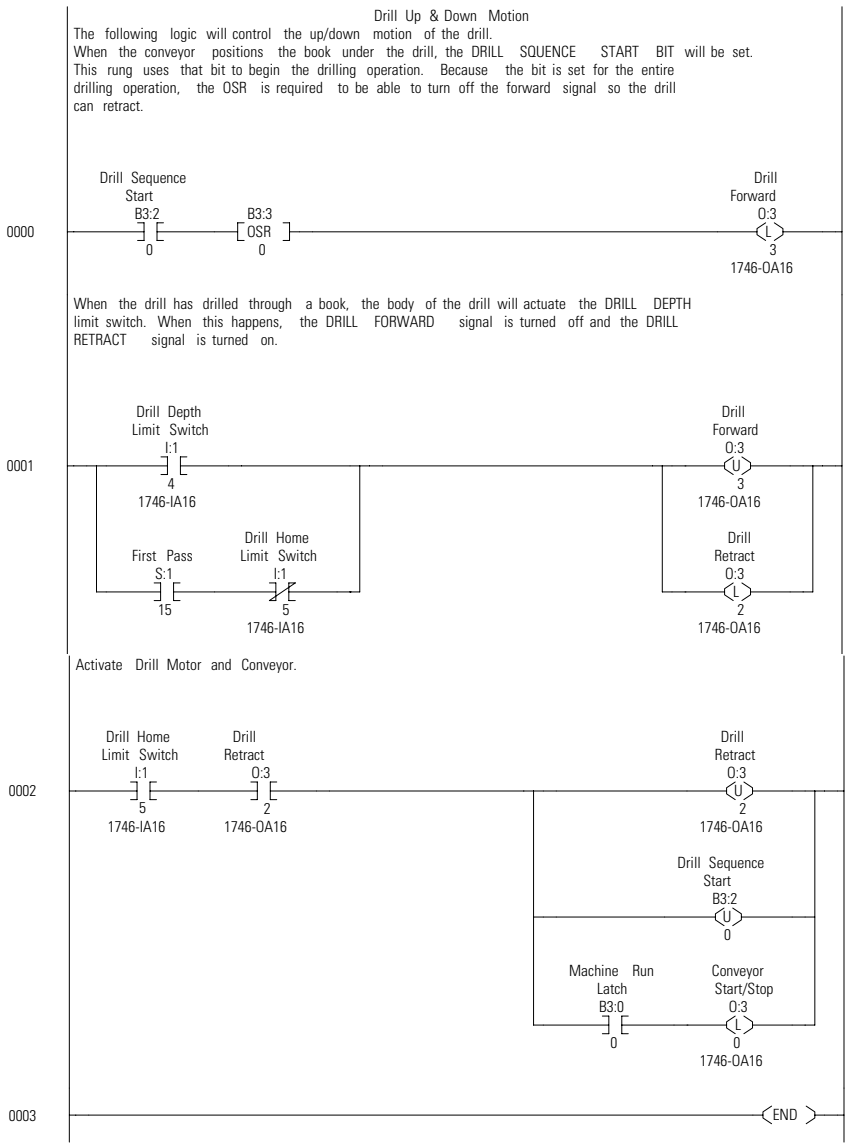
Undrilled books are placed onto a conveyor taking them to a single spindle drill. Each book moves down the conveyor until it reaches the first drilling position. The conveyor stops moving and the drill lowers and drills the first hole. The drill then retracts and the conveyor moves the same book to the second drilling position. The drilling process is repeated until there are the desired holes per book.




The following rung will call the drill sequence subroutine. The subroutine manages the operation of the drilling sequence and will restars the conveyor upon completion of the drilling sequence.
0003
JSR
Jump To Subroutine
SBR File Number U:6
The following rung will call the subroutine that is used to track the amount of wear on the current drill bit.
0004
JSR
Jump To Subroutine
SBR File Number U:7
The following rung will call the subroutine used to initialize logic neseccary for the DII on power up.
First Pass
S:1
15
0005
JSR
Jump To Subroutine
SBR File Number U:4
0006
END

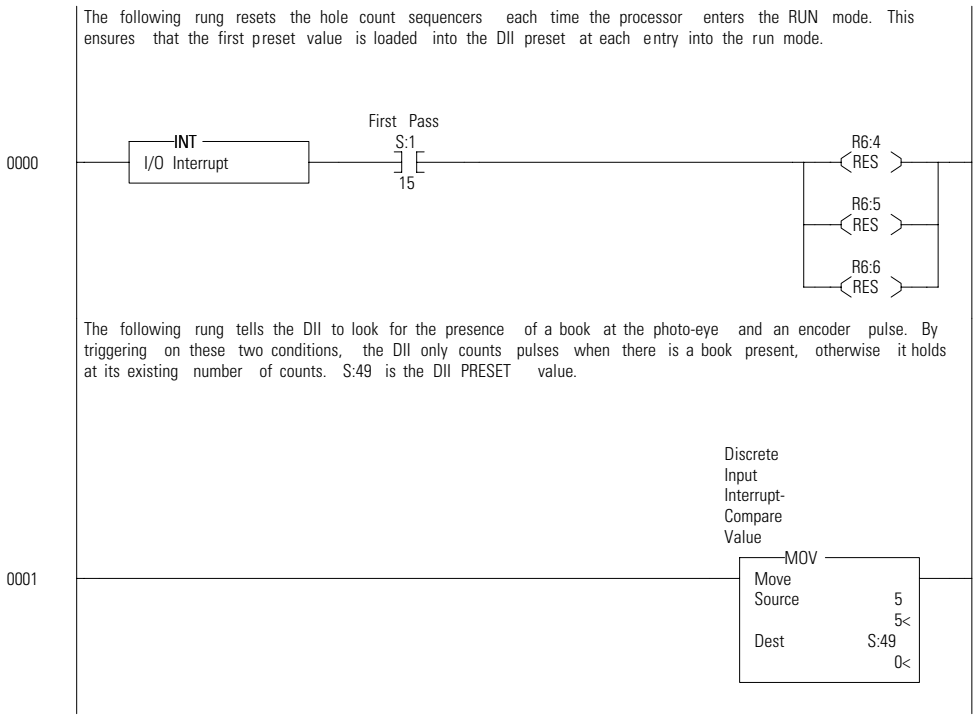
Drill Mechanism Operation

When the operator presses the start button, the drill motor turns on. After the book is in the first drilling position, the conveyor subroutine sets a drill sequence start bit, and the drill moves toward the book. When the drill has drilled through the book, the drill body hits a limit switch and causes the drill to retract up out of the book. When the drill body is fully retracted, the drill body hits another limit switch indicating that it is in the home position. Hitting the second limit switch unlatches the drill sequence start bit and causes the conveyor to move the book to the next drilling position.

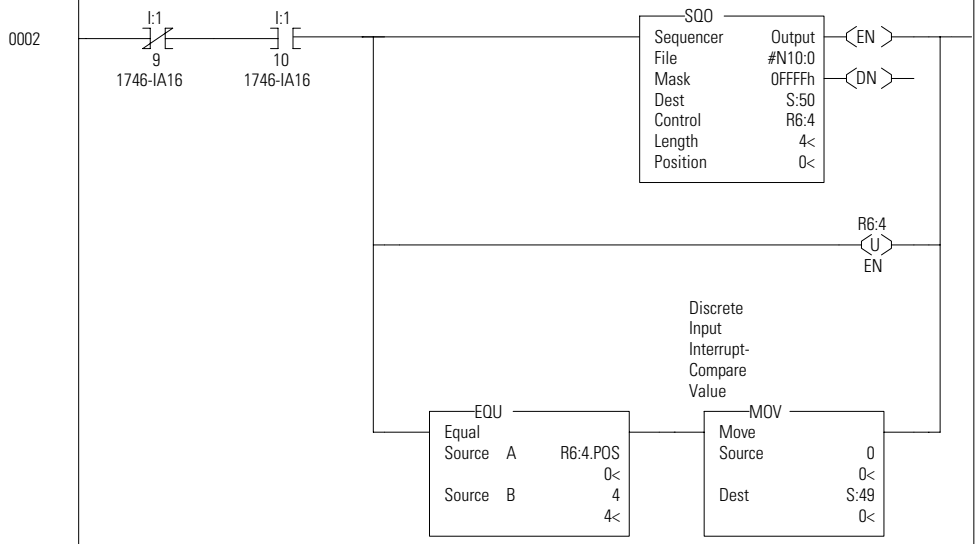


Conveyor Operation

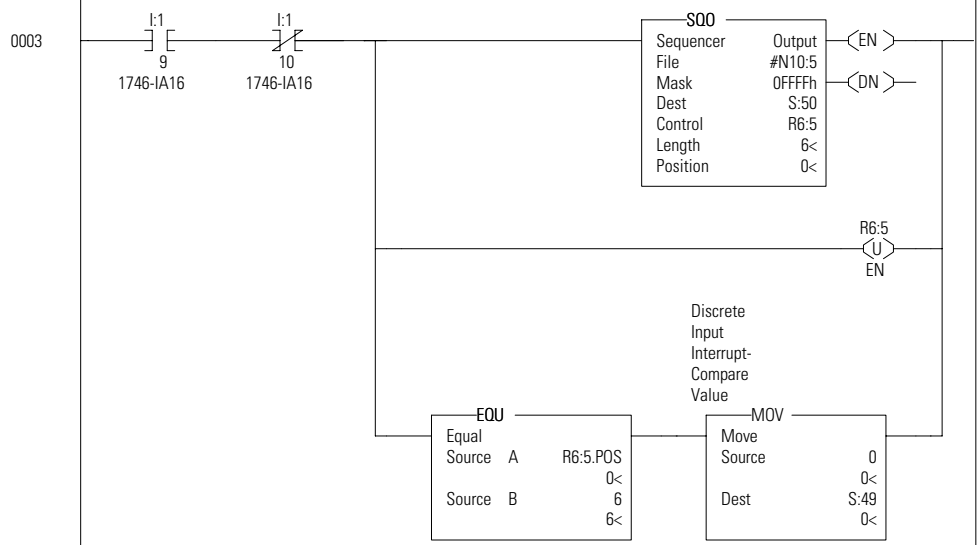
When the start button is pressed, the conveyor moves the books forward. As the first book moves close to the drill, the book trips a photo-eye sensor. This tells the machine where the leading edge of the book is. Based on the position of the selector switch, the conveyor moves the book until it reaches the first drilling position. The drill sequence start bit is set and the first hole is drilled. The drill sequence start bit is now unlatched and the conveyor moves the same book to the second drilling position. The drilling process is then repeated until there are the desired holes per book. The machine then looks for another book to break the photo-eye beam and the process is repeated. The operator can change the number of drilled holes by changing the selector switch.



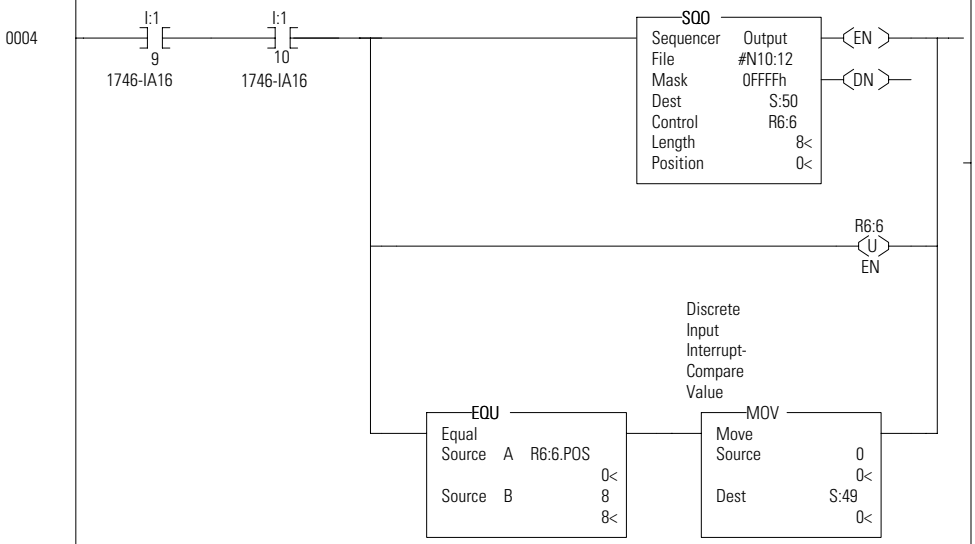
The following rung will keep track of the hole number that is being drilled and loads the next correct DII preset based on the hole count. This rung is only active when the "HOLE SELECTOR" switch is in the "3-HOLE" position. The sequencer uses step 0 as a null step upon reset. It uses the last step as a "go forever" in anticipation of the "end of manual". Moving a 0 into S:49 tells the DII to trigger an interrupt when the trailing edge of the current book is detected.



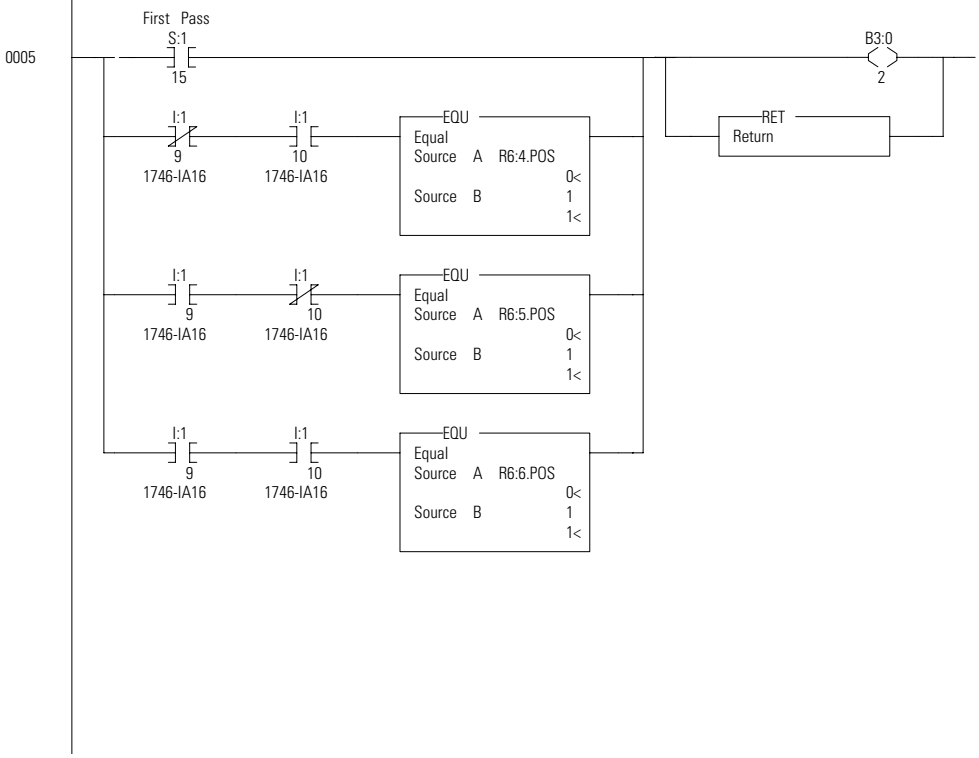
The following rung is identical to the previous rung except that it is only active when the "HOLE SELECTOR" switch is in the "5-HOLE" position.

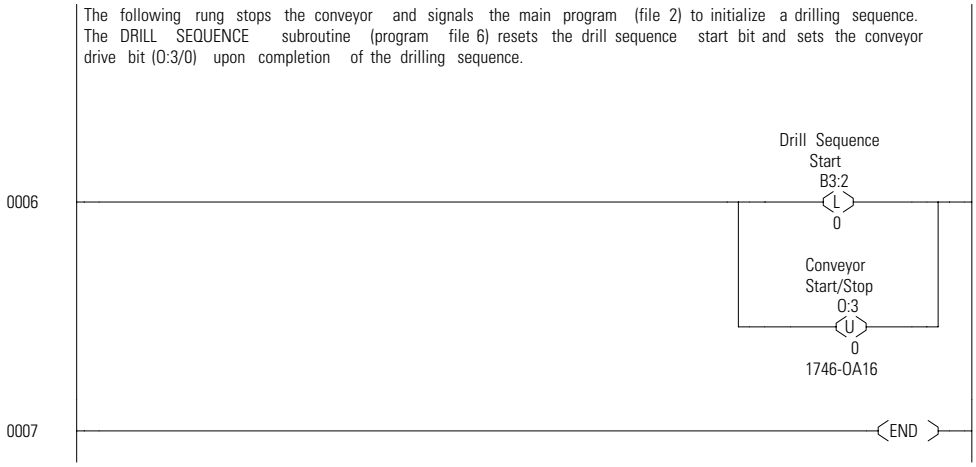


The following rung is identical to the previous two rungs except that it is only active when the "HOLE SELECTOR" switch is in the "7-HOLE" position.



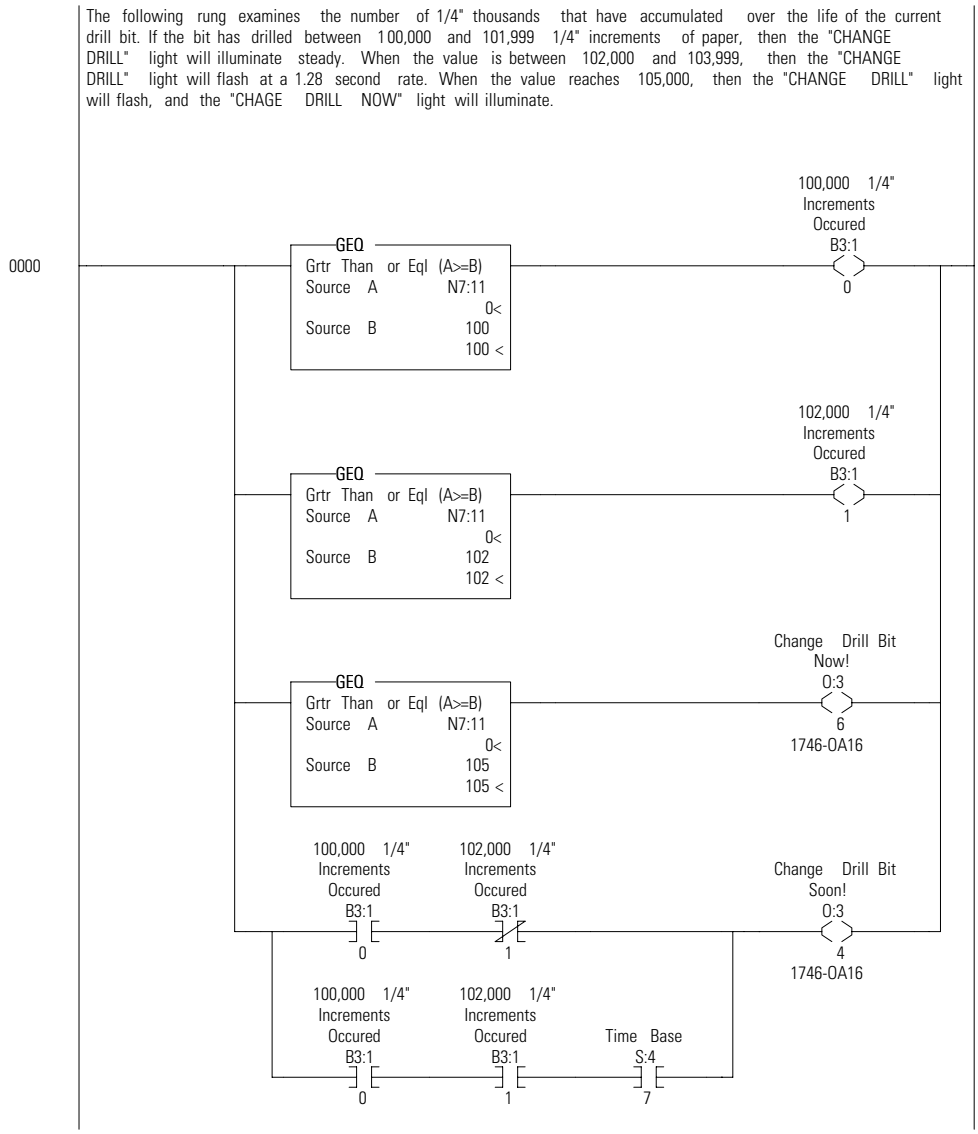
If the processor is in this subroutine either for initialization or due to sensing the trailing edge of a manual, just return and skip the logic that stops the conveyor and starts the drill sequence.

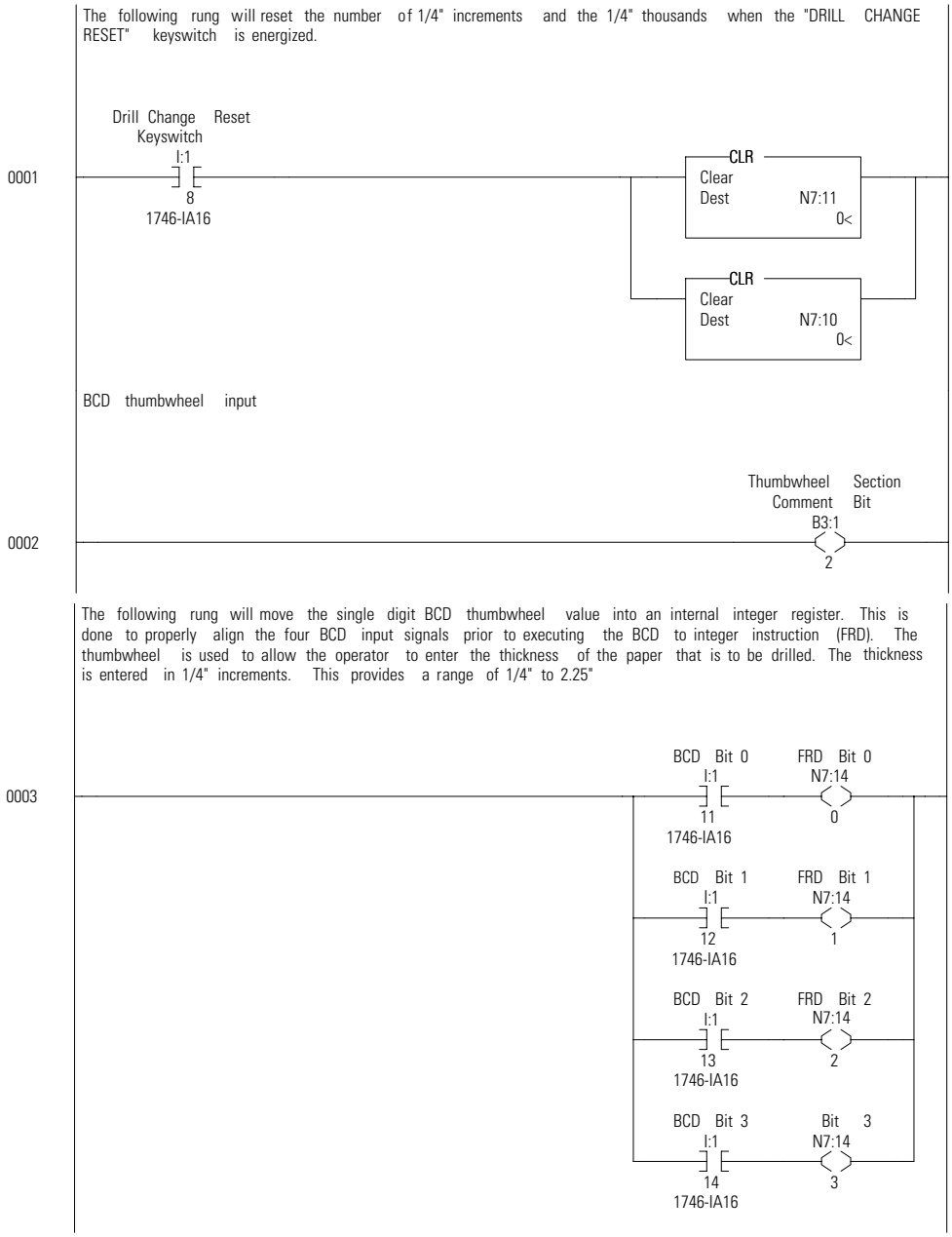



The following rung stops the conveyor and signals the main program (file 2) to initialize a drilling sequence. The DRILL SEQUENCE subroutine (program file 6) resets the drill sequence start bit and sets the conveyor drive bit (O:3/0) upon completion of the drilling sequence.
Drill Sequence Start
B3:2
L
0
Conveyor Start/Stop
O:3
U
0
1746-OA16
0006
0007
END

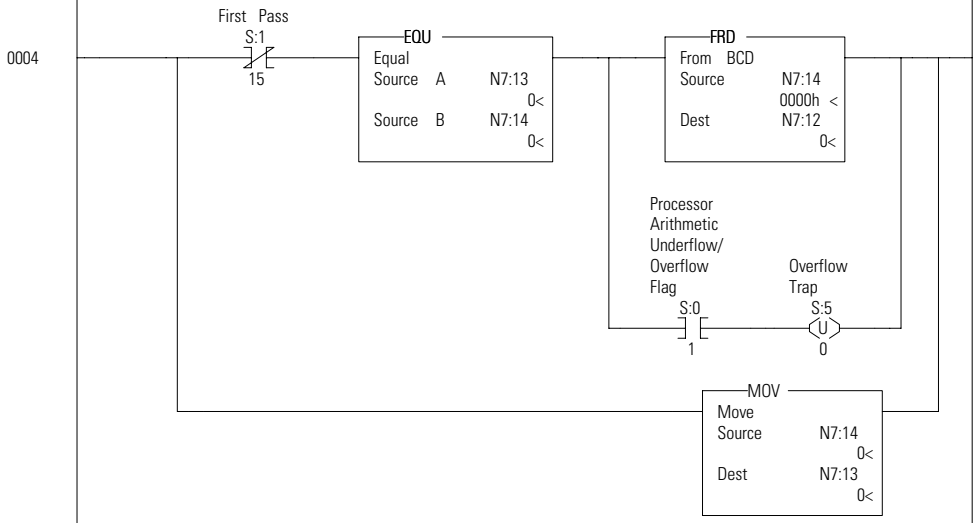
Drill Calculation and Warning

The program tracks the number of holes drilled and the number of inches of material that have been drilled through using a thumbwheel. The thumbwheel is set to the thickness of the book per 1/4 inch. (If the book is 1 1/2 inches thick, the operator would set the thumbwheel to 6.) When 25,000 inches have been drilled, the Change Drill Soon pilot light turns on. When 25,500 inches have been drilled, the Change Drill Soon pilot light flashes. When 26,000 inches have been drilled, the Change Drill Now pilot light turns on and the machine turns off. The operator changes drill bits and then resets the internal drill wear counter by turning the Drill Change Reset keyswitch.

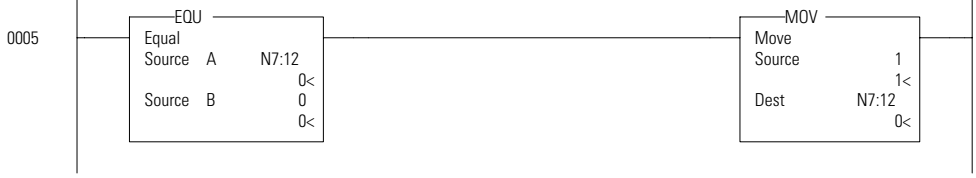


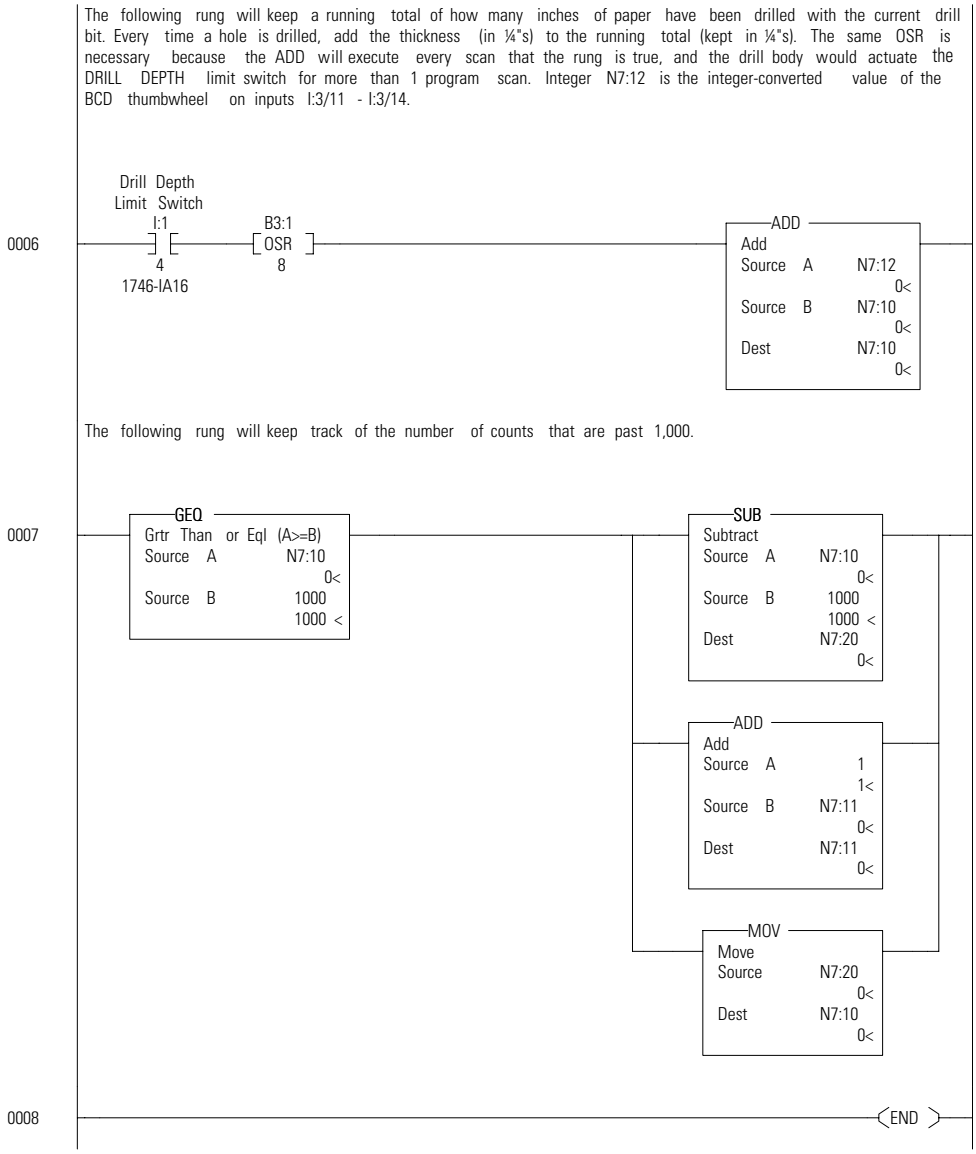

The following rung will reset the number of 1/4" increments and the 1/4" thousands when the "DRILL CHANGE RESET" keyswitch is energized.
Drill Change Reset
Keyswitch
I:1
8
1746-IA16
0001
CLR
Clear
Dest N7:11
0<
CLR
Clear
Dest N7:10
0<
BCD thumbwheel input
Thumbwheel
Comment
Section
Bit
B3:1
2
0002
The following rung will move the single digit BCD thumbwheel value into an internal integer register. This is done to properly align the four BCD input signals prior to executing the BCD to integer instruction (FRD). The thumbwheel is used to allow the operator to enter the thickness of the paper that is to be drilled. The thickness is entered in 1/4" increments. This provides a range of 1/4" to 2.25"
0003
BCD Bit 0
I:1
11
1746-IA16
FRD Bit 0
N7:14
0
BCD Bit 1
I:1
12
1746-IA16
FRD Bit 1
N7:14
1
BCD Bit 2
I:1
13
1746-IA16
FRD Bit 2
N7:14
2
BCD Bit 3
I:1
14
1746-IA16
Bit 3
N7:14
3

The following rung will convert the BCD thumbwheel value from BCD to integer. This is done because the processor operates upon integer values. This rung also "debounces" the thumbwheel to ensure that conversion only occurs on valid BCD values. Note that invalid BCD values can occur while the operator is changing the BCD thumbwheel. This is due to input filter propagation delay differences between the 4 input circuits that provide the BCD input value.



The following rung will ensure that the operator cannot select a paper thickness of 0. If this were allowed, the drill bit life calculation could be defeated resulting in poor quality holes due to a dull drill bit. Therefore the minimum paper thickness that will be used to calculate drill bit wear is ¼".




The following rung will keep a running total of how many inches of paper have been drilled with the current drill bit. Every time a hole is drilled, add the thickness (in ¼"s) to the running total (kept in ¼"s). The same OSR is necessary because the ADD will execute every scan that the rung is true, and the drill body would actuate the DRILL DEPTH limit switch for more than 1 program scan. Integer N7:12 is the integer-converted value of the BCD thumbwheel on inputs I:3/11 - I:3/14.
Drill Depth Limit Switch
I:1
4
1746-IA16
B3:1
OSR
8
0006
ADD
Add
Source A N7:12
0<
Source B N7:10
0<
Dest N7:10
0<
The following rung will keep track of the number of counts that are past 1,000.
0007
GEQ
Grtr Than or Eql (A>=B)
Source A N7:10
0<
Source B 1000
1000 <
SUB
Subtract
Source A N7:10
0<
Source B 1000
1000 <
Dest N7:20
0<
ADD
Add
Source A 1
1<
Source B N7:11
0<
Dest N7:11
0<
MOV
Move
Source N7:20
0<
Dest N7:10
0<
0008
END

Time Driven Sequencer Application Example

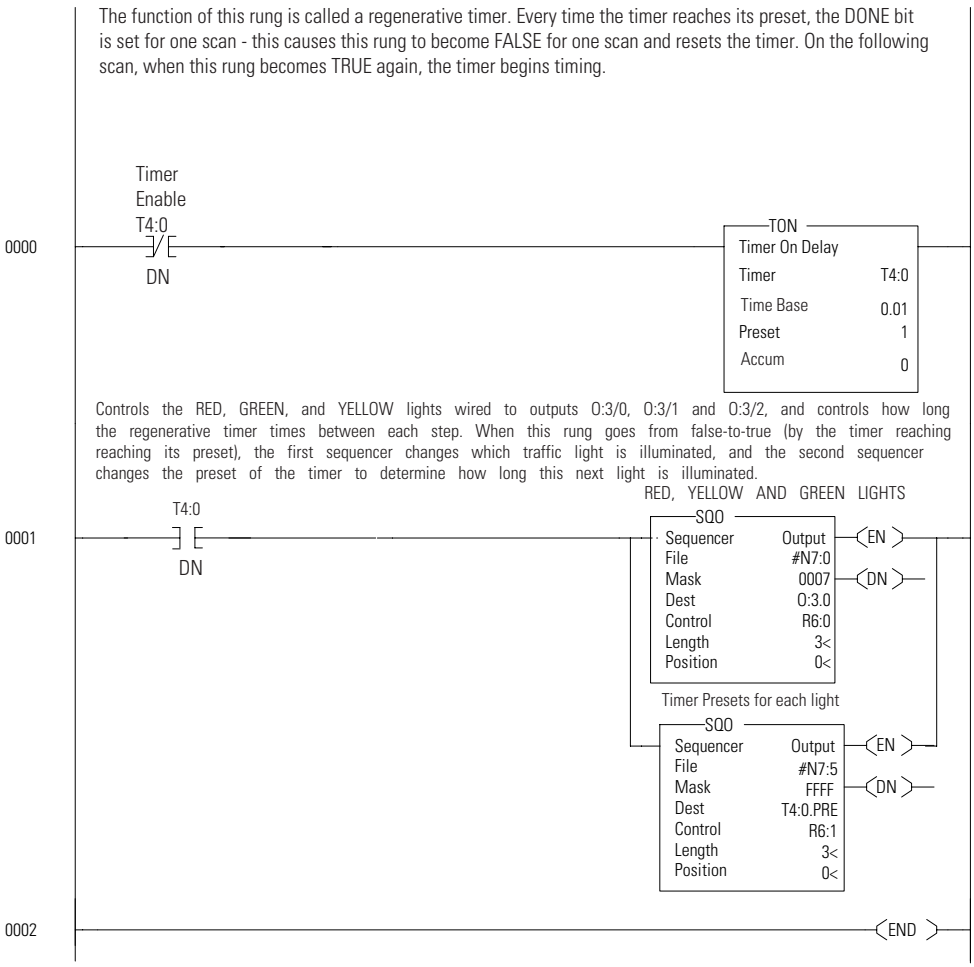
The following application example illustrates the use of the TON and SQO instructions in a traffic signal at an intersection. The timing requirements are:

- Red light - 30 seconds
- Yellow light - 15 seconds
- Green light - 60 seconds

The timer, when it reaches its preset, steps the sequencer that in turn controls which traffic signal is illuminated. For a detailed explanation of:

- XIC, XIO, and TON instructions, see Chapter 2.
- SQO and SQC instructions, see Chapter 7.

Time Driven Sequencer Ladder Program

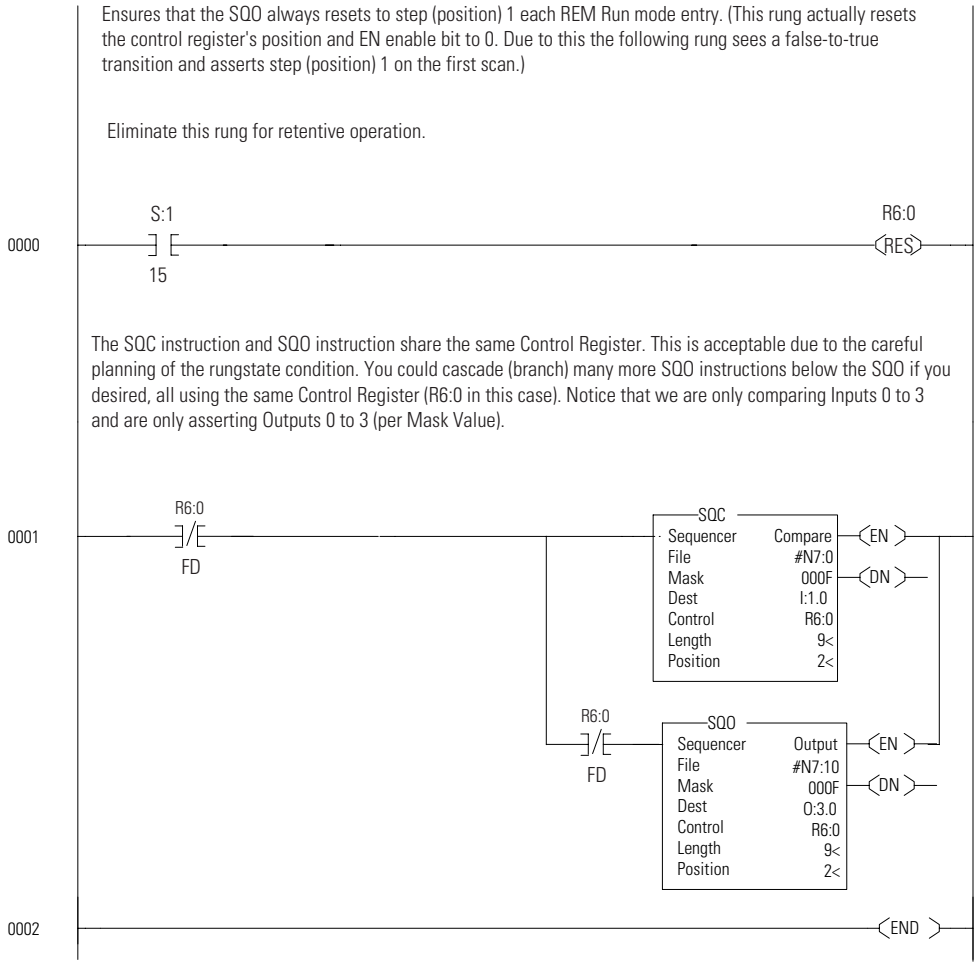


Event Driven Sequencer Application Example

The following application example illustrates how the FD (found) bit on an SQC instruction can be used to advance as SQO to the next step (position). This application program is used when a specific order of events is required to occur repeatedly. By using this combination, you can eliminate using the XIO, XIC, and other instructions. For a detailed explanation of:

- XIC, XIO, and RES instructions, see Chapter 2.
- SQO and SQC instructions, see Chapter 7.

Event Driven Sequencer Ladder Program

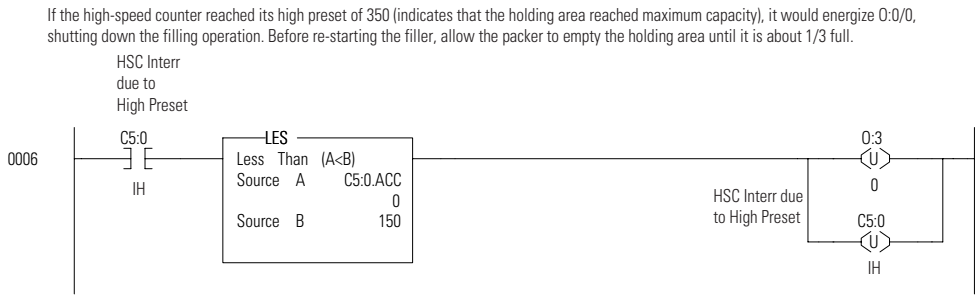


The following table displays the FILE DATA for both sequencers. The SQC compare data starts at N7:9. While the SQO output data starts at N7:10 and ends at N7:19. Please note that the step 0 of the SQO is never active. The reset rung combined with the rung logic of the sequencers guarantees that the sequencers always start at step 1. Both

sequencers also “roll over” to step 1. “Roll Over” to step 1 is integral to all sequencer instructions.

Table G.1 SQC Compare Data

Addresses	Data (Radix = Decimal)									
N7:0	0	1	2	3	4	5	6	7	8	9
N7:10	0	0	1	2	3	4	5	6	7	8

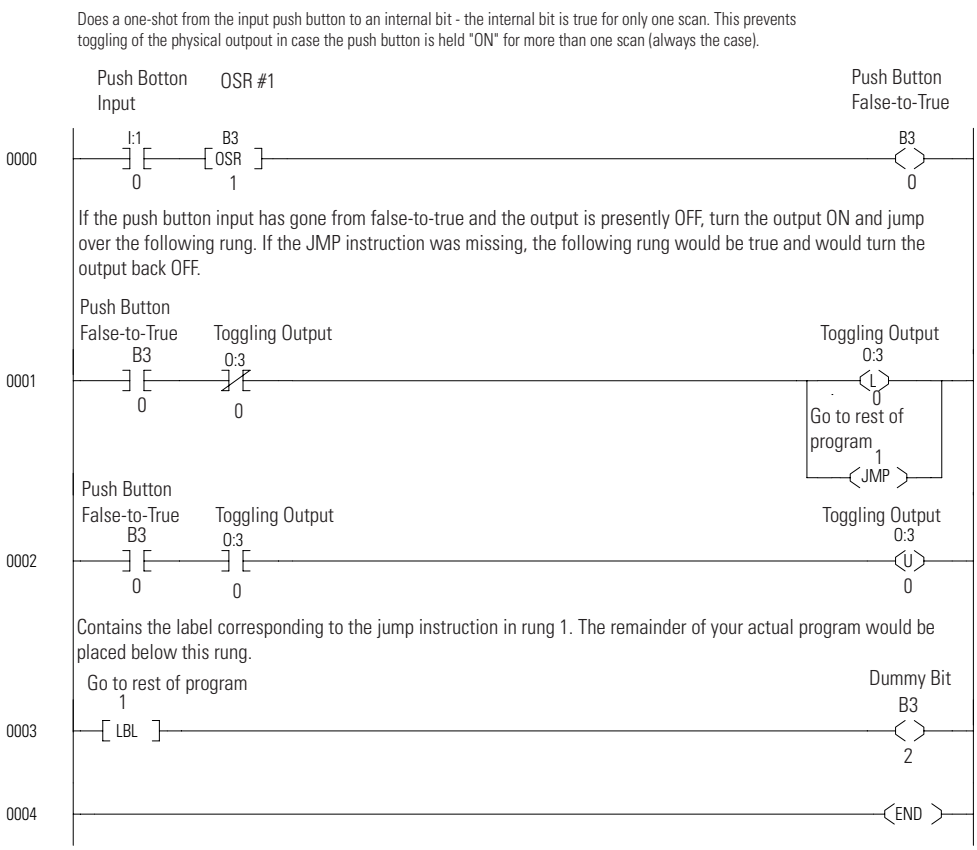


On/Off Circuit Application Example

The following application example illustrates how to use an input to toggle an output either on or off. For a detailed explanation of:

- XIC, XIO, OTE, OTU, OTL, and OSR instructions, see chapter 1.
- JMP and LBL instructions, see chapter 5.

On/Off Circuit Ladder Program



Interfacing with Enhanced Bar Code Decoders Over DH-485 Network Using the MSG Instruction

The purpose of this section is to illustrate how to interface Allen-Bradley Enhanced Bar Code Decoders to SLC 5/03 and higher processors via the DH-485 network. Enhanced Bar Code Decoders act only as slave devices on this network. This means that these decoders cannot initiate the transfer of data to a host device, such as the SLC 5/03 (or higher) processor on DH-485. The SLC processor must initiate commands to a decoder and "poll" that decoder for the reply to those commands.

Processor and Decoder Operation

The Enhanced Bar Code Decoder (catalog number 2755-DS/DD, Series B), when used as a node on a DH-485 network can act as a slave only. This means that the decoder may not initiate communications to any other node on the network. Therefore, in

order for a device to get bar code data from an Enhanced Bar Code Decoder on a DH-485 network, that device must send a “read” command and then “poll” the decoder for the reply with data.

The only devices capable of polling a slave device on DH-485 are the SLC 5/03 and higher processors. For the SLC 5/03 processors (1747-OS302, FRN10 or later), polling can be done via channels 0 and 1. For the SLC 5/04 processors (1747-OS401, FRN7 or later), channel 0 supports this capability.

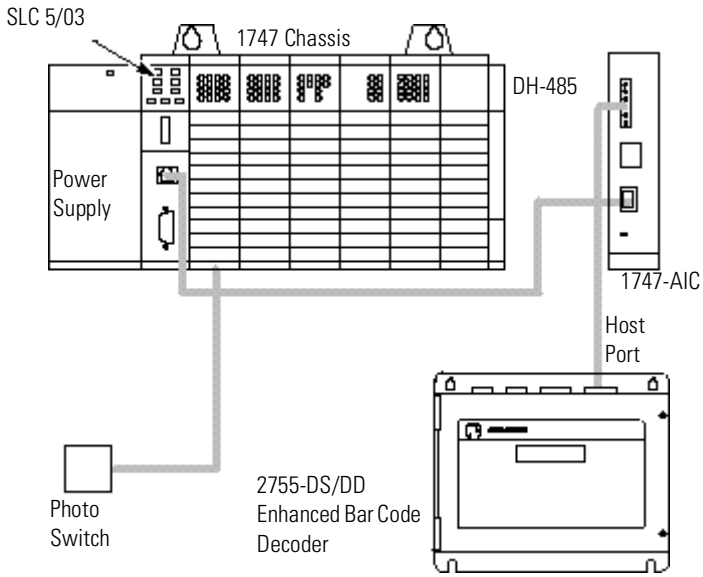
There are many ways to “trigger” bar code decoders to read a bar code label when a label is present.

- a package detect switch wired to both an SLC input module and the bar code decoder
- a package detect switch wired only to an SLC input and an SLC output then used to “trigger” the decoder
- via a software “trigger” command from the SLC processor

For this example, the software “trigger” is used. However, the basic principal is the same for all “trigger” modes.

System Set Up

In this example, a photo switch is located such that when it detects a product is in position for the bar code scanner to read a bar code label on the product, a discrete input to the SLC 5/03 processor is energized.



The 5/03 ladder program then initiates a "MSG Write" to the decoder to "trigger" the decoder to start scanning for a valid bar code label. When the decoder is scanning for a valid bar code label, it operates as shown below:

Result of Scan	Bar Code Decoder Response	Processor Response
Good Read	turns on its "Good Read" onboard output wired to the SLC processor	When one of these two inputs to the SLC are turned on, the SLC will initiate a "MSG Read" to the decoder to get the label data or no-read message data.
No-Read	turns on its "No-Read" onboard output wired to the SLC processor	

In this case, the good read output is turned on as soon as a valid read occurs, and the no-read output is turned on after the decoder has attempted to read a label for a specified amount of time and could not.

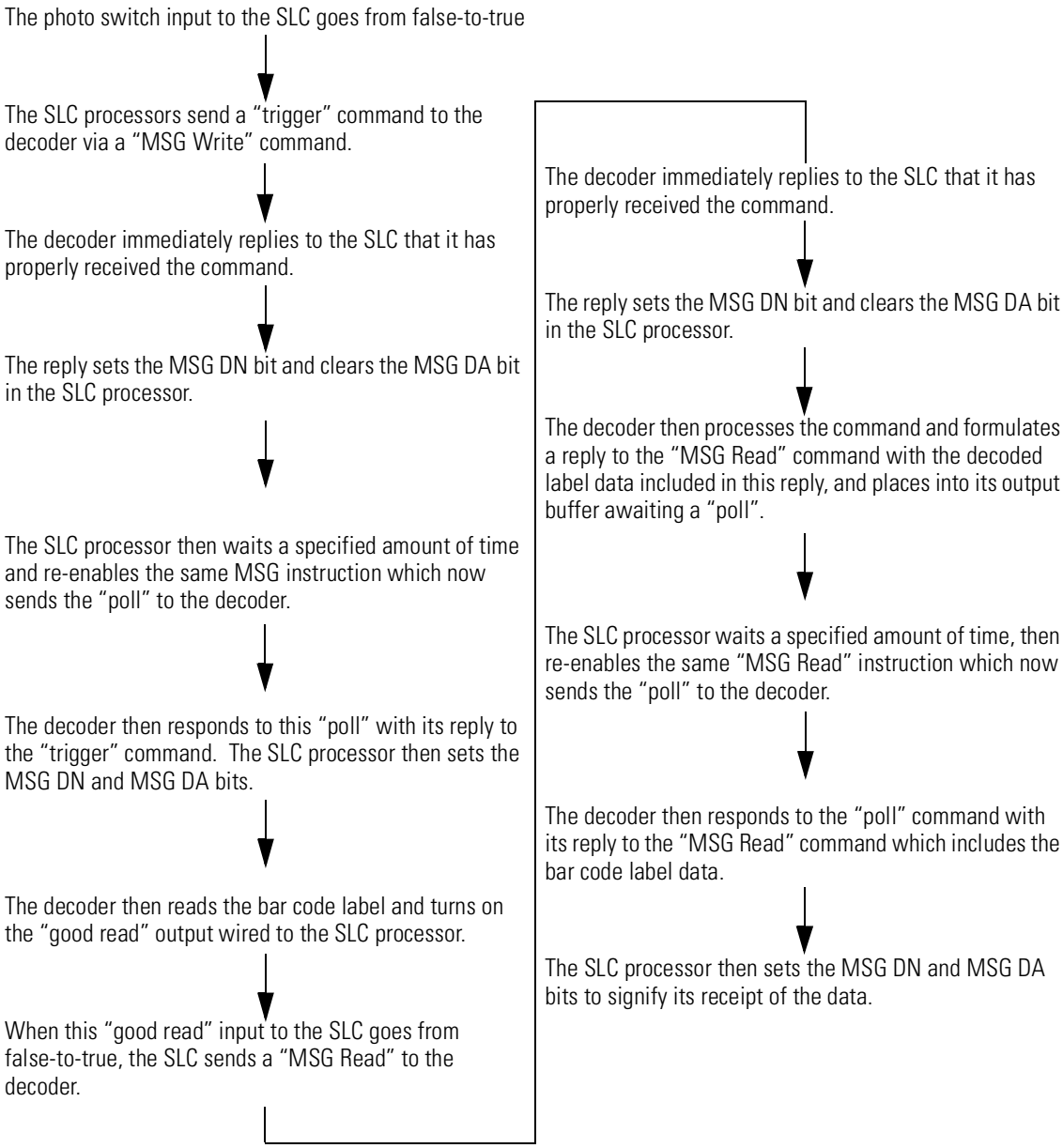
The amount of time the decoder attempts to read a label is variable and is called the "No-Read Timer". For this example it is assumed that the product is moving by the scanner and if the label is not read in 2 seconds, it is not read at all. Therefore, the "No-Read Timer" parameter in the bar code decoder is set to 2 seconds. Refer to the *DS/DD Series Enhanced Bar Code Decoders (Bulletin 2755) User's Manual*, publication 2755-833, for details concerning the configuration of your Allen-Bradley Enhanced Bar Code Decoder.

Operating Sequence

With the bar code decoder configured as previously described, the following series of event take place when a product with a good bar code label breaks the photo switch and this input to the SLC goes from false-to-true. The SLC 5/03 ladder program logic to make it happen is also included. Please note that, as previously stated, a bar code/SLC system may be configured in a variety of ways.

Messages sent by the SLC processor to the Enhanced Bar Code Decoder must be programmed as shown by the example ladder program on page H-29. If this logic is not followed, the communication between the two DH-485 devices could become out of sequence, resulting in no data transfers between the decode and the SLC processor. To correct such a problem, cycle power to the decoder.

Sequence of Events



TIP

These events are described in more detail by the comments listed within the example ladder program page G-21.

Optimizing MSG Time-Out

If the time delay between sending a command to an Enhanced Bar Code Decoder and "polling" for the reply is not long enough, the MSG instruction will time-out (MSG TO bit = 1) each time it is enabled from that point forward. To re-synchronize the SLC processor and the decoder, you need to cycle power on the decoder to clear its buffer.

There are ways of clearing the buffers in the decoder, such as sending a "Clear Buffers" command or a "Reset" command to the decoder. However, the best way to handle this issue is to never let it happen. Optimizing the time delay between sending the initial command and "polling" for the reply is the best way to accomplish this. The delay must be long enough so the decoder has enough time to formulate a reply to the command and short enough to not impact the throughput of the application.

Example MSG Instruction Configuration

The example SLC 5/03 and SLC 5/04 ladder program demonstrates how to send commands to an Enhanced Bar Code Decoder, and then after a time delay, "pol" for a reply. The internal set up screen parameters for the two MSG instructions in the example ladder program are shown below, along with the necessary Enhanced Bar Code Decoder configuration parameters.

Table G.2 Message Configuration

	MSG #1	**MSG #2**
Type	peer-to-peer	peer-to-peer
Read/write	write	read
Target device	485CIF	485CIF
Local/remote	local	local

Table G.2 Message Configuration

	MSG #1	**MSG #2**
Control block	N7:0	N7:20
Channel	1	1
Target node	2	2
Our source file address	N7:15	N7:40
Target CIF offset(1)	0	0
Message length in elements	1	10
Message time-out (seconds)	5	5

(1) The Target CIF Offset when working with Enhanced Bar Code Decoders as slaves on DH-485 must contain a value greater than 255. However, 255 is the largest value SLC programming software allows you to enter into this parameter in a MSG instruction. Therefore, use an unconditioned rung with a MOV instruction to move the proper value into the Target CIF Offset field. The example ladder program in this section demonstrates this. Note that 1586 decimal in a "MSG Write" is the value which results in a properly configured Enhanced Bar Code Decoder to initiate the "trigger" function. A value of 256 in a "MSG Read" requests a specified number of words of data from the bar code decoder. In this example, we are reading 10 words or 20 characters (bytes).

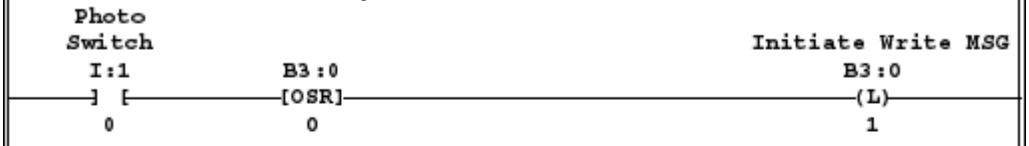
Example Scanner and Decoder Configuration

Table G.3 Scanner and Decoder Configuration

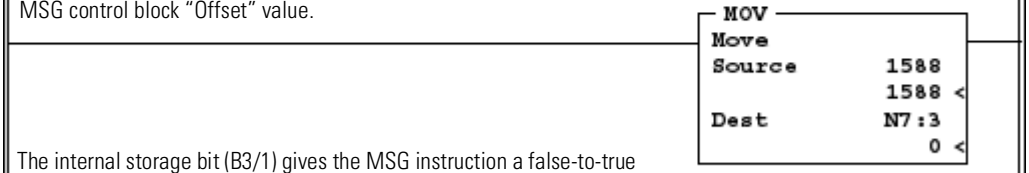
Scanner Configuration Parameters **Scanner Control Page**		**2755-DS/DD Series B Enhanced Bar Code Decoder Configuration Parameters** **Host Configurations Page**	
Discrete I/O:	Read Package 25 ms No-Read Package 25 ms	Baud Rate:	19200
Laser Light:	Triggered	Bits/Char:	8 Data 1 Stop
Decode Mode:	Host	Parity:	Even
No-Read Time:	2000 ms	Host Protocol:	DH485 PCCC-1
Inter Scan Time:	none	Device Address:	2
Capture Count:	2	ACK Char:	none
Symbols/Scan:	1	NAK Char:	none
Symbols/Package:	1	Large Buffer:	No
Match Complete:	1	Send Host Message: Package	Immediately after Valid
		Transmission Check:	none

Example Ladder Program

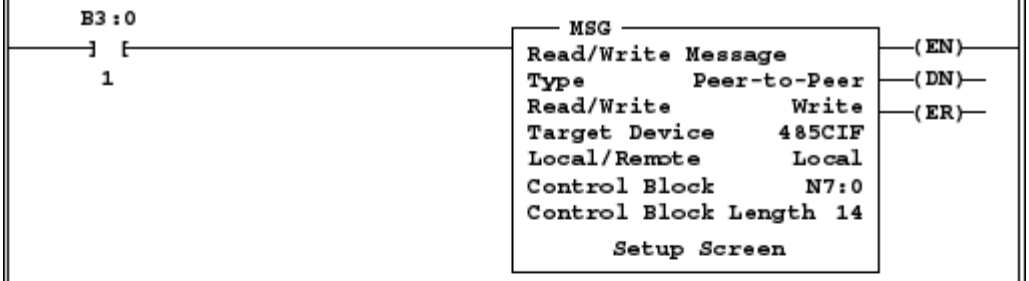
This rung detects the Photo Switch input going from false-to-true, and latches internal storage bit B3/1.



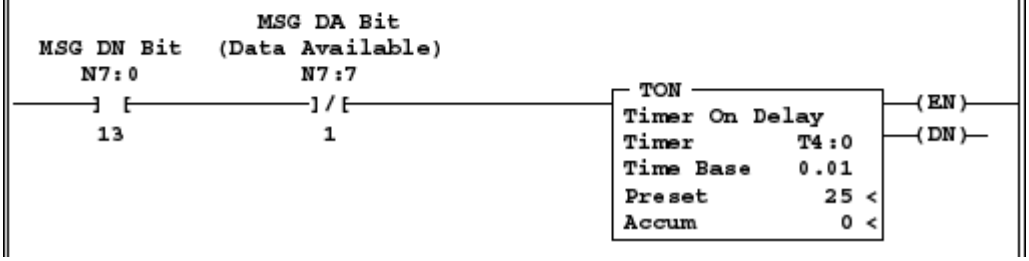
This rung moves the decimal value for the bar code decoder "trigger" command into the MSG instructions "Offset" parameter. The programming software does not allow values greater than 255 decimal to be entered into a MSG control block "Offset" value.



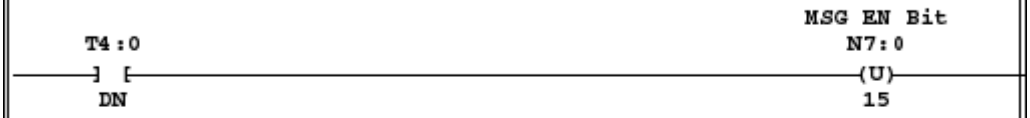
The internal storage bit (B3/1) gives the MSG instruction a false-to-true transition to send the initial command. B3/1 remains latched until both the DN and DA bits are set for the MSG instruction.



The initial reply from an Enhanced Bar Code Decoder will result in MSG DN = 1 and MSG DA = 0. This simply indicates that the decoder has received the command, but has not yet formulated a reply. The maximum time delay needed between sending the initial command and sending a poll to get the reply is 250 ms. In most cases this delay could be much less (30 ms to 100 ms), depending on the number of features the decoder is configured for.

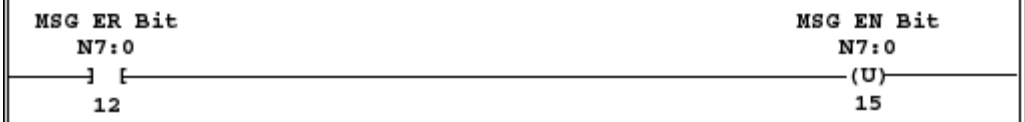


The internal storage bit, B3/1, holds the MSG instruction true until DN and DA are both set, indicating completion of the command sent and reply received sequence. When DN is set and DA is reset, unlatching the MSG EN bit effectively toggles the MSG instruction the same as if the MSG rung were toggled, i.e. rung conditions made false, then true. The MSG instruction is toggled one time after DN and NOT DA plus some time delay, to send a final poll to the decoder to get the MSG reply. When the reply is received, the SLC processor sets DN and DA.

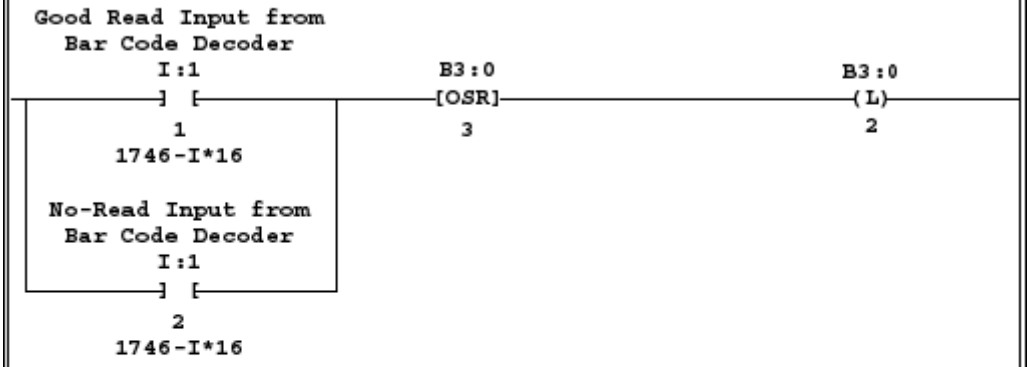


When the SLC processor sets both DN and DA for a MSG instruction, the MSG sequence to an Enhanced Bar Code Decoder is complete. In this case, the decoder has received the "trigger" command and has performed this command. Therefore, unlatch B3/1 at this time to be ready for the next request for "trigger".

If an error occurs with the MSG instruction, the ER bit is set. If this occurs, the user can either try to resend the same message again by unlatching EN, or at this point, could sound an alarm or route the product down a rework loop or some other similar action. If the latter choice is used, you must also unlatch B3/1 at this time to be ready for the next request for "trigger".

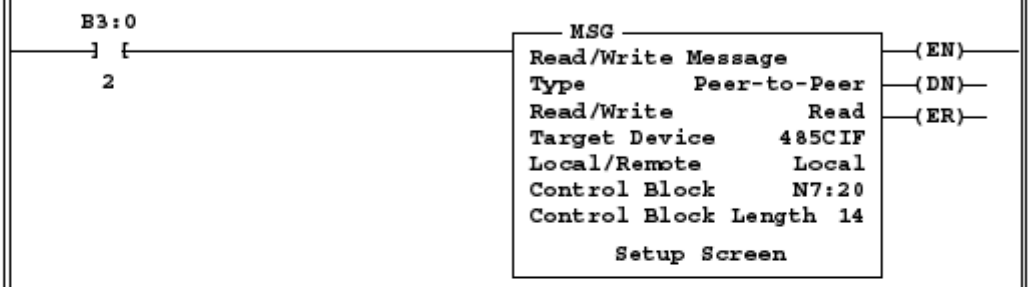


This rung detects false-to-true transitions of either a good read or a no-read input from the bar code decoder and latches internal storage bit B3/2. B3/2 then, in the next rung, initializes the MSG read command to the decoder. This is done if either a good read or a no-read occurs, because the no-read message configured in the decoder is data as much as actual bar code label data. Therefore, the ladder program must distinguish between data that means no-read as well as actual bar code label data.

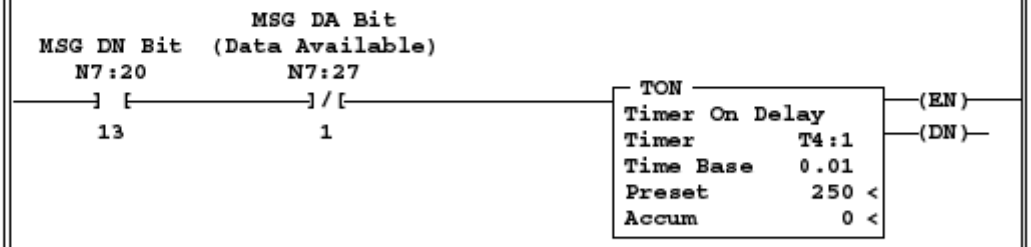


This rung moves the decimal value for the bar code decoder "Read" command into the MSG instruction's "Offset" parameter. The programming software does not allow values greater than 255 decimal to be entered into a MSG control block "Offset" value.

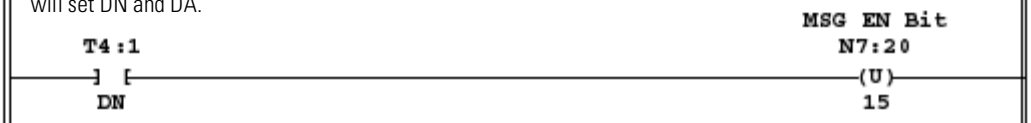
The internal storage bit, B3/2, gives the MSG instruction a false-to-true transition to send the initial command.



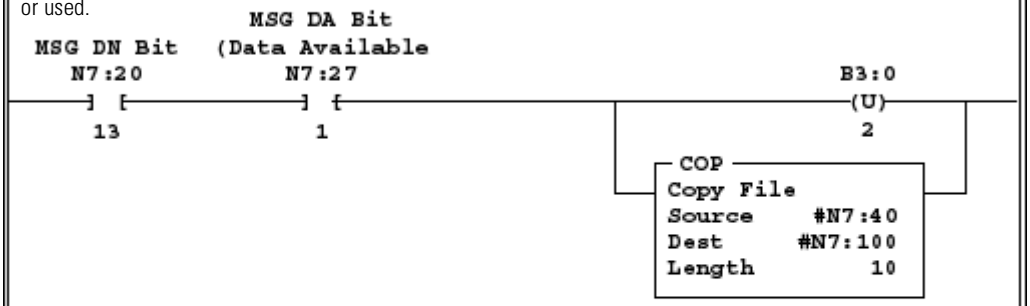
The initial reply from an Enhanced Bar Code Decoder will result in MSG DN = 1 and MSG DA = 0. This indicates that the decoder has received the command, but has not yet formulated a reply. The maximum time delay needed between sending the initial command and sending a poll to get the reply is 250 ms. In most cases this delay could be much less (30 ms to 100 ms), depending on the number of features the decoder is configured for.



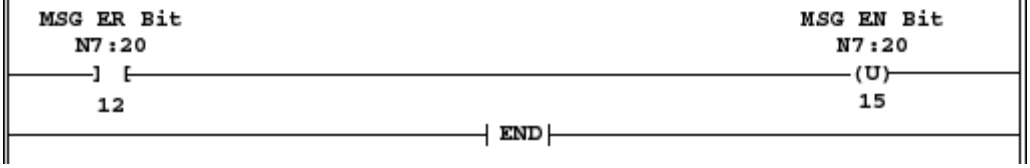
The internal storage bit, B3/2, holds the MSG instruction true until DN and DA are both set, indicating completion of the command sent and reply received sequence. When DN is set and DA is reset, unlatching the MSG EN bit effectively toggles the MSG instruction the same as if the MSG rung were toggled, i.e. rung conditions made false, then true. The MSG instruction is toggled one time after DN and NOT DA plus some time delay, to send a final poll to the decoder to get the MSG reply. When the reply is received, the SLC processor will set DN and DA.



When the SLC processor sets both DN and DA for a MSG instruction, the MSG sequence to an Enhanced BAr Code Decoder is complete. In this case, the decoder has received the "Read" command and has formulated a reply to this command. Therefore, unlatch B3/2 at this time to be ready for the next "REad" request. In addition, when DN and DA are both set, this indicates that the data received with the read reply (except "no-read" data) is valid and may be buffered or used.



If an error occurs with the MSG instruction, the ER bit is set. If this occurs, the user can either try to resend the same message again by unlatching EN, or at this point, could sound an alarm or route the product down a rework loop or some other similar action. If the latter choice is used, you must also unlatch B3/2 at this time to be ready for the next "Read" request.



Numerics

A

B

C

D

E

F

G

H

I

J

L

M

N

O

P

R

S

T

U

W

X

Z

SLC 500 Alphabetical List of Instructions

Reach us now at www.rockwellautomation.com

Wherever you need us, Rockwell Automation brings together leading brands in industrial automation including Allen-Bradley controls, Reliance Electric power transmission products, Dodge mechanical power transmission components, and Rockwell Software. Rockwell Automation's unique, flexible approach to helping customers achieve a competitive advantage is supported by thousands of authorized partners, distributors and system integrators around the world.

Americas Headquarters, 1201 South Second Street, Milwaukee, WI 53204, USA, Tel: (1) 414 382-2000, Fax: (1) 414 382-4444
European Headquarters SA/NV, avenue Herrmann Debroux, 46, 1160 Brussels, Belgium, Tel: (32) 2 663 06 00, Fax: (32) 2 663 06 40
Asia Pacific Headquarters, 27/F Citicorp Centre, 18 Whitfield Road, Causeway Bay, Hong Kong, Tel: (852) 2887 4788, Fax: (852) 2508 1846

Publication 1747-RM001C-EN-P - September 2001

Supersedes Publication 1747-6.15 - January 1998, 1747-DU001A-EN-P - July 2000, 1747-RN001A-EN-P - August 2000
1747-6.15-DU3 - April 1999, 1747-6.15-RN1 - May 1998