Series Six Programmable Controller CCM Communications User's Manual

Series Six™ Programmable Controller CCM Communications

User's Manual

GE Fanuc Automation

September 1988 GEK-25364A

WARNINGS, CAUTIONS, AND NOTES AS USED IN THIS PUBLICATION

WARNING

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

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

CAUTION

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

NOTE

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

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

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

PREFACE

The purpose of the CCM Communications User's Manual is to provide the information needed to implement a serial communications link between a Series Six™ Programmable Logic Controller (PLC), and a host computer, color-graphics terminal, peripheral device, or another Series Six PLC.

This manual is a general update and second edition of what was formerly called the Series Six PLC Data Communication Manual. It includes all information previously found in GEK-90505 *Series Six PLC Supplement to Data Communication Manual*, Chapters 7 (CCM3) and Chapter 8 (CCM3-RTU Protocol).

Chapter 1. Introduction to Series Six PLC Communication is an introduction to data communications with emphasis on those areas pertaining to the Series Six PLC.

Chapter 2. Communications Control Module explains the installation and operation of the Communications Control Module (CCM2 and CCM3). This chapter includes sections on: system configuration and protocol, cable wiring, CCM communications, CCM programming, Operator Interface Unit (OIU) use with the CCM, and an introduction to RTU protocol.

Chapter 3. Input/Output Communications Control Module describes the Input/Output Communications Control Module (I/O CCM) used to link the Series Six PLC and a host computer, programmable terminals, and other intelligent devices.

Chapter 4. Serial Interface Protocols for the CCM defines the CCM serial interface protocol. Discusses CCM peer-to-peer and master-slave protocols and includes detailed flow charts for both.

Chapter 5. RTU Communications Protocol describes in detail the protocol used when configured in Remote Terminal Unit (RTU) mode.

Chapter 6. Communication Applications contains basic Series Six PLC application programs for using the CCM Status Byte, using the CCM Diagnostic Status Words, and setting up a multidrop polling routine.

Appendix A. Host Computer Interface Software is a brief discussion of host computer communication interface software for use with Series Six PLCs equipped with a CCM. It includes sections on features and ordering of the software as well as basic software operation.

Appendix B. Expanded Functions provides programming information for Series Six Communication Control Module (CCM) Expanded Memory mapping, single bit write and programmable timeout and retry.

Appendix C. Glossary of Terms contains a concise, alphabetized listing of conventional communications terms and (where applicable) their asssociated acronyms.

PREFACE

RELATED PUBLICATIONS

- GEK-25361 *Series Six™ PLC Installation and Maintenance Manual*, describes earlier models of Series Six PLCs.

- GFK-0013 *GEnet™ Factory LAN Series Six Programmable Control Network Interface Users Manual*, describes the installation, operation and programming of the GEnet Network Interface. Describes MAP, Datagram and Global Data communication services.

- GEK-96608 *GEnet™ Factory LAN System Users Manual*, contains information on connecting various devices, which use the CCM protocol, to GEnet.

- GEK-25367 *Series Six Data Sheet Manual*, contains the specifications, description and wiring of various communications modules.

- GEK-84866 *Series Six PLC Operator Interface Unit (OIU) Data Sheet*, contains specifications, description and wiring of OIU module.

- GFK-0238 *Series Six PLC Communications Control Module Type 2 and Type 3 Data Sheet*, contains module specifications, description, and wiring for current (CCM2, CCM3) modules combined in one data sheet. The current CCMs support expanded memory addressing but without tape function.

- GEK-90824 *Series Six PLC Input/Output Communications Control Module (I/O CCM) Data Sheet*, contains specifications, description and wiring for the I/O CCM.

- GEK-83539 *Series Six PLC Communications Control Module 1 (CCM1) Data Sheet*, contains specifications, description and wiring of early versions of the CCM1 module. This module has been superseded by the CCM2 and CCM3 modules and is not a production module.
  This document is listed for reference only.

- GEK-83542 *Series Six PLC Communications Control Module 2 (CCM2) Data Sheet*, contains specifications, description and wiring of earlier CCM2 modules having tape functionality. This module has been superseded by enhanced versions of the CCM2 and is not a production module.
  This document is listed for reference only.

- GEK-90763 *Series Six PLC Communications Control Module 3 (CCM3) Data Sheet*, contains specifications, description and wiring of earlier CCM3 modules having tape functionality. This module has been superseded by enhanced versions of the CCM3 and is not a production module.
  This document is listed for reference only.

MODULE COMPATABILITY

You should be aware of the software/hardware compatability among the various types and versions of Communication Control Modules. Also, the module functionality and installation may vary with the module type and revision.

The following table represents the chronological order of module development. All new CCM2/3 orders will be filled with the latest module version IC600CB536 or IC600CB536. These modules are backward compatible, except for the STR-LINK function.

Read this information before installing and attempting to use your Series Six™ Communications Control Module (CCM2, CCM3 or I/O CCM).

MODULE FEATURES AND COMPATABILITY

Module Type	Module Catalog Order Number	Module Installation			Features		
		CPU Rack	Series 6 Plus Rack	Series 6 Plus II	STR-LINK	CCM Mode	RTU Mode
		(Note 1)	(Note 2)	(Note 3)			
Multiple PROM Module (Note 4)							
CCM2	IC600CB516K	Y	Y	N	Y	Y	N
CCM3	IC600CB517K	Y	Y	N	Y	Y	Y
Upgraded 7-PROM Module							
CCM2	44A729763-G01	Y	Y	N	N	Y	N
CCM3	44A729764-G01	Y	Y	N	N	Y	Y
Single PROM Module (Note 5)							
CCM2	IC600CB516L	Y	Y	Y	Y	Y	N
CCM3	IC600CB517L	Y	Y	Y	Y	Y	Y
Enhanced Functions (Note 5)							
CCM2	IC600CB536K	Y	Y	Y	N	Y	N
CCM3	IC600CB537K	Y	Y	Y	N	Y	Y
Upgraded Single PROM Module							
CCM2	IC600CB536L	Y	Y	Y	N	Y	N
CCM3	IC600CB537L	Y	Y	Y	N	Y	Y
I/O CCM	IC600BF948	Y	Y	Y	N	Y	Y

Y = YES (module or feature supported)
N = NO (NON-compatability, or feature NOT supported)

Note 1: CPU Rack — Models 60, 600, and 6000
Note 2: Series Six Plus rack — IC600CP60x, IC600CP63x
Note 3: Series Six Plus II rack — IC600CP61x, IC600CP62x

Note 4: CCM2 IC600CB516K, or earlier modules + upgrade kit = 44A729763-G01
CCM3 IC600CB517K, or earlier modules + upgrade kit = 44A729764-G01

Note 5: CCM2 IC600CB516L + upgrade kit = IC600CB536L
CCM2 IC600CB536K + upgrade kit = IC600CB536L

CCM3 IC600CB517L + upgrade kit = IC600CB537L
CCM3 IC600CB537K + upgrade kit = IC600CB537L

Upgrade Kit: The upgrade kit provides firmware for both the single-PROM and multiple-PROM CCM modules. Part numbers for ordering the upgrade kit are:

CCM2	44A286360-G02
CCM3	44A286385-G03
I/O CCM	44A286359-G01

The single-PROM upgrade will provide full functionality of the IC600CB536L, and IC600CB537L CCM modules .

Due to a hardware limitation, multiple-PROM boards that use the firmware upgrade may not be compatible with some Series Six Plus CPUs. In compatible CPUs they will have full functionality of the latest CCM modules.

MODULE TYPE Communications Control Module (CCM2/CCM3)

Hardware identification for current CCM2 and CCM3 modules is identical -- as follows:

Hardware Id. CCMA3, 44A717545-G02 R02 (R02 or later)

MODULE TYPE Input/Output Communication Control Module (I/O CCM)

Hardware version IC600BF948 (or later) is a 2-PROM module which supports two serial protocols: Communications Control Module (CCM) protocol, and Remote Terminal Unit (RTU) protocol. The I/O CCM module is identified as follows:

Hardware Id. BAMA, 44A717588-G01 R02 (R02 or later)

Software release (203 hex, 515 decimal) is comprised of firmware only. This firmware is part of the I/O CCM module and is identified on the PROM package as follows:

Firmware Id. Version 203 (or later)

PROM Loc. (U33), Label 174-041D (or later) and,
PROM Loc. (U24), Label 174-040C (or later)

If more information is required, contact your local GE Fanuc Automation sales office, or authorized distributor.

CONTENTS

CONTENTS

CONTENTS

CONTENTS

CONTENTS

CONTENTS

CONTENTS

CONTENTS

CONTENTS Page

FIGURES

FIGURES

TABLES

TABLES

CHAPTER 1
INTRODUCTION TO SERIES SIX DATA COMMUNICATIONS

INTRODUCTION TO DATA COMMUNICATIONS

Data communications is generally defined as the electronically encoded transmission of information from one point to another. This chapter will expand on this definition by describing the essential components of data communications emphasizing those areas pertaining to Series Six™ Programmable Logic Controllers (PLCs).

The reader should have some familiarity with the binary and hexadecimal numbering systems and a basic understanding of programmable controllers. The information in this chapter is intended as background information only. Specific information on Series Six PLC Communications Control Modules (CCMs) and related topics can be found in later chapters.

Figure 1.1 shows the main components necessary for serial communications between a host computer or Series Six PLC and another Series Six PLC.

84pc0001

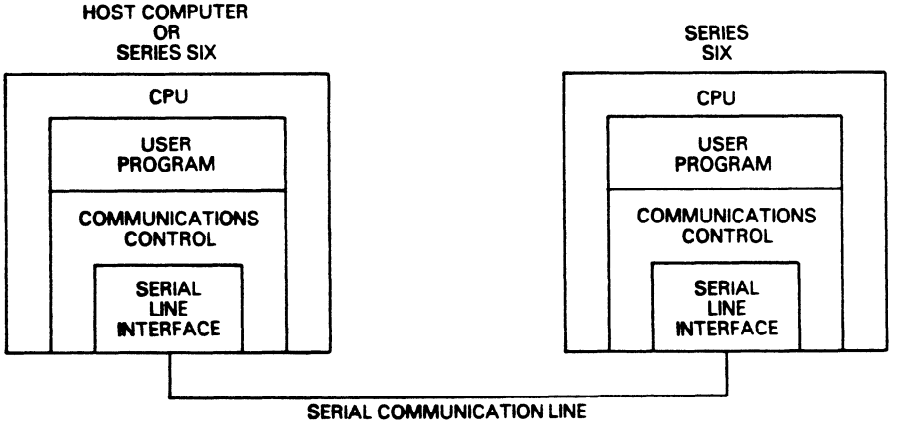


Figure 1.1 COMPONENTS OF SERIES SIX SERIAL COMMUNICATIONS

COMMUNICATIONS NETWORK (SYSTEM) CONFIGURATIONS

The term network (system) configuration refers to the way in which computers, terminals, and communication equipment are interconnected. In Series Six PLC data communications the following system configurations are possible:

- Point-to-point
- Multidrop
- GEnet™ Local Area Network (LAN)

POINT-TO-POINT

This is the simplest type of system configuration; in it only two devices can be connected to the same communication line. Figure 1.2 is a block diagram of the point-to-point configuration.

84pc0007

Figure 1.2 POINT-TO-POINT SYSTEM CONFIGURATION

MULTIDROP

The multidrop configuration is a party-line structure in which several devices share the same communication line. This line may be direct if RS-422 or RS-232D is used, or indirect with modems if RS-232D is used. One device is a master and the rest are slaves; only the master can initiate communication with other elements in the system. Figure 1.3 is a block diagram of the multidrop configuration.

84pc0009

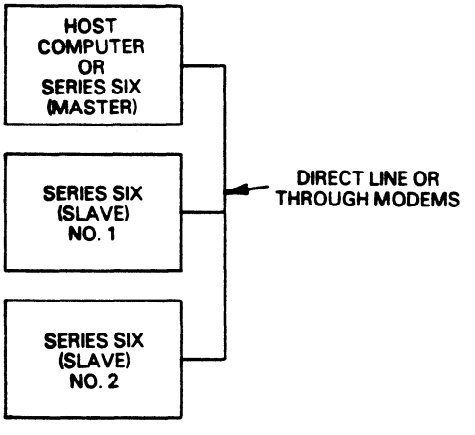


Figure 1.3 MULTIDROP SYSTEM CONFIGURATION

Multidrop or Point-to-Point Terminating Resistor

The Communications Control Module (CCM) is supplied with a 150 Ohm terminating resistor in each RS-422 receiver circuit. If the module is at either end of an RS-422 multidrop or point-to-point link, these resistors should be in the circuit.

If the module is an intermediate drop in the multidrop link, the appropriate resistors should be removed from the circuit by placing their jumpers in the storage position. (Refer to Chapter 2 for detailed information concerning placement of these resistors.)

GEnet™ LOCAL AREA NETWORK (LAN)

For applications requiring much broader communications capabilities than the CCM can provide; the GEnet Factory LAN is available. The GEnet Factory LAN is a 10 Mbps broadband (5 Mbps for carrierband) token passing bus which provides high speed communications between various types of processors such as Programmable Logic Controllers (PLCs), Computer Numerical Controllers (CNCs), other high-level factory-management control systems.

a42530

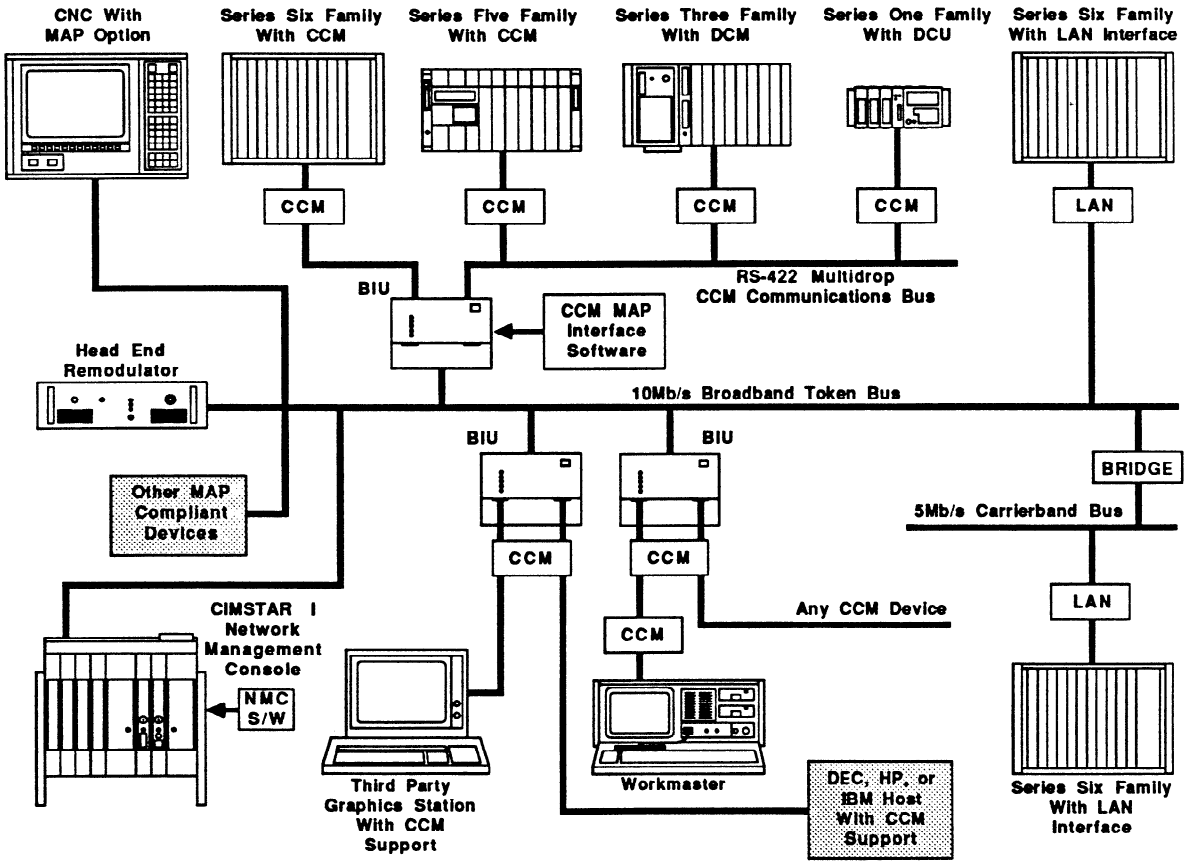


Figure 1.4 GEnet SYSTEM CONFIGURATION

The GEnet Factory LAN architecture is based on accepted industry standards set forth in the Manufacturing Automation Protocol (MAP) specification. MAP services are based on the Open System Interconnection (OSI) Reference Model developed by the International Standards Organization (ISO).

Devices which use the CCM protocol can interface to the GEnet LAN through the GEnet Bus Interface Unit (BIU). The BIU is tailored by loading device specific software to provide the required interface to the various automation product.

The Series Six Plus PLC can be connected directly to the GEnet Factory LAN via the Series Six LAN interface module. For more information refer to the GFK-0013, *GEnet Factory LAN Series Six PLC Network Interface User's Manual.*

The Data Communications Unit (DCU) is used to interface the Series One, Series One Junior, and Series One Plus PLCs to the network via the BIU. Likewise, the Data Communications Module (DCM) is used interface the Series Three PLC to the network via the BIU. For detailed information refer to the GEK-90477, *Series One/Series Three Programmable Controllers Data Communications Manual.*

For further information on connecting various devices, which use the CCM protocol, to the GEnet Factory LAN, refer to the *GEnet Factory LAN System User's Manual*, GEK-96608.

COMMUNICATION MODES (CCM, RTU)

Specific modes of communication are supported by each of the Communication Control Modules. The CCM mode of operation supports peer, master, and slave communications. The Remote Terminal Unit (RTU) mode of operation is a master/slave protocol. It is used to link the PLC with a process controller, computer, or other intelligent device which uses the RTU protocol. Only the master can initiate a communications request when RTU mode is used. The CCM module can be configured only as an slave for RTU mode.

A summary of the Communication Control Modules (CCMs) with associated modes of communication is listed below.

Module	Communication Modes
CCM2	CCM
CCM3	CCM, RTU slave
I/O CCM	CCM, RTU slave

INITIATING THE COMMUNICATION

Transfer of data between a Series Six PLC and another device is initiated by a serial communications request. The device which initiates the request is designated the source; the device which receives the request, the target. This request resides in the user program and contains the following information:

- Identification of the target device which is to receive the communications request.
- Direction of data transfer - the requestor may choose to send or receive data.
- Address to which data is being transferred - in either source or target device.
- Address from which data is being transferred - in either source or target device.
- Amount of data being transferred.

COMMUNICATIONS CONTROL

After the communications request is initiated by the user program of the source device, the request information described above is transferred to communications control. Communications control puts this information into the proper format for transmission via the serial line interface. Serial transmission performs the following functions:

- Encoding and decoding of required information according to a standard information code.
- Assembly and disassembly of the communications request information and data text for transmission according to a set of rules or protocols.
- Method of checking for errors which may occur during transmission.

SERIAL COMMUNICATIONS

The operations on data explained thus far have occurred within the host computer or Series Six and therefore have been in parallel, that is, in terms of 8-bit bytes or 16-bit words. This is because within a computer or Series Six it is easier and faster to transfer and manipulate data in parallel. When transferring information externally, however, the cost of parallel transmission becomes prohibitive for distances more than a few feet. Therefore, serial transmission is normally used between devices.

Once the communications request is initiated and the data is properly formatted according to the protocols mentioned before, the serial line interface transmits it over the communications line.

Figure 1.1 shows a data transfer using the CCM protocol. The host may establish communications with a target Series Six PLC by initiating the communications request which begins with an enquiry sequence. To maintain the communications, the request target (the remote Series Six) must acknowledge the enquiry within the appropriate time.

After establishing communications, the source sends a header (containing information necessary to transfer a block of data) to the target device. When the target receives this information, data can either be transferred from source to target or from target to source.

As characters are received by either device, the sequence discussed earlier for transmitting characters is performed in reverse order. The incoming characters must first be converted from serial to parallel, then the receiver must extract the characters from the protocol to act upon them in the appropriate manner. Ultimately, information is passed from one device's memory to another device's memory via user programs.

In the preceding text, key words or phrases about data communications have been underlined. An explanation of these key words and phrases are given in the remaining sections of this chapter.

INFORMATION CODES

An information code is a standard by which numbers, letters, symbols, and control characters can be formed for serial transmission. In Series Six PLC communications, characters in headers (discussed in the section, Protocols) as well as control characters are encoded. Other characters such as those occurring in data, are uncoded binary data. There are a number of different coding schemes used today, but the most common and the type used in Series Six PLC communications is the American Standard Code for Information Interchange or ASCII code.

As shown in the illustration below, the CCM uses an 8-bit character code plus an optional parity bit to transfer serial data.

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

Table 1.1 shows examples of the binary and hexadecimal forms, including parity bit, of several ASCII characters. The parity bit is explained in the section, Parity Checking. Table 1.2 contains a complete list of the ASCII character set represented in hexadecimal and decimal.

Table 1.1 ASCII INFORMATION CODE FORMAT

PARITY BIT	BINARY FORM OF CHARACTER	HEXADECIMAL FORM OF CHARACTER	ASCII CHARACTER
(odd) 0	0 0 0 0 0 0 1 0	0 2	STX (control char.) Start Of Text
(odd) 1	0 0 1 0 1 0 1 1	2 B	+
(even) 1	0 0 0 1 0 1 0 1	1 5	NAK (control char.) Negative Ack.
(even) 0	0 0 1 1 1 0 0 1	3 9	9

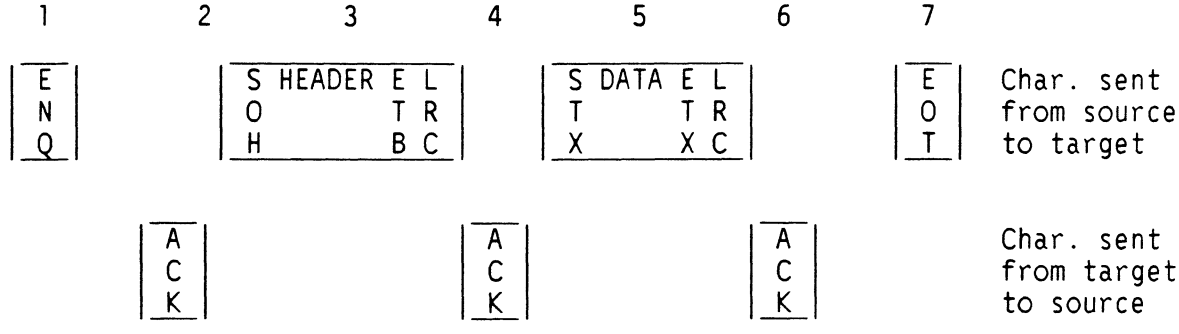
Table 1.2 ASCII CODE LIST

Character	Decimal	Hexadecimal	Character	Decimal	Hexadecimal	Character	Decimal	Hexadecimal
NUL	0	00	0	48	30	[	91	5B
SOH	1	01	1	49	31	\	92	5C
STX	2	02	2	50	32	]	93	5D
ETX	3	03	3	51	33	^	94	5E
EOT	4	04	4	52	34	-	95	5F
ENQ	5	05	5	53	35	'	96	60
ACK	6	06	6	54	36	a	97	61
BEL	7	07	7	55	37	b	98	62
BS	8	08	8	56	38	c	99	63
HT	9	09	9	57	39	d	100	64
LF	10	0A	:	58	3A	e	101	65
VT	11	0B	;	59	3B	f	102	66
FF	12	0C	<	60	3C	g	103	67
CR	13	0D	=	61	3D	h	104	68
SO	14	0E	>	62	3E	i	105	69
SI	15	0F	?	63	3F	j	106	6A
DLE	16	10	@	64	40	k	107	6B
DC1	17	11	A	65	41	l	108	6C
DC2	18	12	B	66	42	m	109	6D
DC3	19	13	C	67	43	n	110	6E
DC4	20	14	D	68	44	o	111	6F
NAK	21	15	E	69	45	p	112	70
SYN	22	16	F	70	46	q	113	71
ETB	23	17	G	71	47	r	114	72
CAN	24	18	H	72	48	s	115	73
EM	25	19	I	73	49	t	116	74
SUB	26	1A	J	74	4A	u	117	75
ESC	27	1B	K	75	4B	v	118	76
FS	28	1C	L	76	4C	w	119	77
GS	29	1D	M	77	4D	x	120	78
RS	30	1E	N	78	4E	y	121	79
US	31	1F	O	79	4F	z	122	7A
SP	32	20	P	80	50	{	123	7B
!	33	21	Q	81	51	/	124	7C
"	34	22	R	82	52	}	125	7D
#	35	23	S	83	53	~	126	7E
$	36	24	T	84	54	DEL	127	7F
%	37	25	U	85	55			
&	38	26	V	86	56			
'	39	27	W	87	57			
(	40	28	X	88	58			
)	41	29	Y	89	59			
*	42	2A	Z	90	5A			
+	43	2B						
,	44	2C						
-	45	2D						
.	46	2E						
/	47	2F						

PROTOCOLS

A protocol is a set of rules which ensures the orderly transmission of data. In Series Six PLC serial communications, it is the set of rules by which a communications link is established and maintained between the device initiating the request (the source) and the device receiving the request (the target). The example below illustrates Series Six CCM peer-to-peer protocol. For a complete explanation of the CCM protocol, refer to Chapter 4, CCM Serial Interface Protocol, and Chapter 5 for the RTU Protocol.

When a Series Six initiates a request, the following sequence must occur for the data transfer to take place.



1. ENQ is an ASCII control character meaning ENQuire which seeks to determine whether or not the target is ready.

2. ACK is an ASCII control character meaning Acknowledge. (Device is ready to communicate)

3. The header block includes the following ASCII coded information:

   - SOH - ASCII control character meaning Start of Header.
   - ID of target device.
   - Direction of data transfer.
   - Type of data being transferred.
   - Target memory address for data being transferred.
   - Amount of data being transferred.
   - ID of source device.
   - ETB - ASCII control character meaning End of Transmission Block.
   - LRC - Longitudinal Redundancy Checking.

4. ACK - Acknowledge, header information is valid.

5. The data block includes the following information:

   - STX - ASCII control character meaning Start of Text.
   - Uncoded binary data.
   - ETX - ASCII control character meaning End of Text.
   - LRC - Longitudinal Redundancy Checking.

6. ACK - Acknowledge, data information is valid.

7. EOT - ASCII control character meaning End of Transmission.

TRANSMISSION ERRORS AND DETECTION

In order to minimize effects of transmission errors due to noise, some means of error checking or detection must be employed.

NOISE ERRORS

The Series Six PLC Communication Control Modules (CCMs) employ two types of noise error checking:

- Parity checking.
- Longitudinal redundancy checking (block check character).

Parity Checking

Parity checking can be generally specified as even, odd, or none. The parity bit, derived by the sender and monitored by the receiver, is dependent on the number of 1s occurring in the binary character. If parity is defined as odd, the total number of 1s in the binary character (in addition to the parity bit) must be odd.

In the example shown below, the ASCII coded A contains two 1s, therefore, the parity bit must be 1 for odd parity. The parity bit would be 0 in this case if parity were defined as even. In the case of no parity the parity bit is not transmitted. For CCM mode, the optional parity bit may be odd or none, and for RTU mode parity may be odd, even or none.

If parity checking is employed, and one of the bits is transmitted incorrectly, the parity bit will reflect the error. Actually, the parity bit will reflect the error any time there is an odd number of bits transmitted incorrectly.

ASCII character, A, received correctly.

Parity Bit	Received Data Byte								ASCII Character
(odd)	8	7	6	5	4	3	2	1	
1	0	1	0	0	0	0	0	1	A

ASCII character, A, received with error in the first bit.

Parity Bit	Received Data Byte								ASCII Character
(odd)	8	7	6	5	4	3	2	1	
1	0	1	0	0	0	0	0	0	A

The parity bit (representing odd parity) which is monitored by the receiver detects the error in transmission because the received character with parity has an even number of 1s instead of an odd number.

If, on the other hand, an even number of bits in a character is transmitted incorrectly, the parity bit will not reflect the error.

Parity Bit	Received Data Byte								ASCII Character
(odd)	8	7	6	5	4	3	2	1	
1	0	1	0	0	0	0	1	0	A

ASCII character, A, received with errors in the first two bits.

The parity bit does not reflect the error because the received character with parity shows an odd number of 1s as it is supposed to.

To further detect transmission errors, longitudinal redundancy checking is used.

Longitudinal Redundancy Checking

Longitudinal Redundancy Checking (LRC) is a method of detecting errors in an entire block. Series Six CCM protocol uses this method to derive the LRC. The sending device inserts the LRC at the end of the header block and each block of data text. The receiving device generates its own block check character based on the incoming data and compares it to the transmitted LRC to detect errors.

At the transmitter the LRC is generated by the exclusive ORing (XOR) of each header or data text byte to be transmitted; at the receiver, of each header or data text byte received. For CCM protocol, the header block SOH and ETB bytes, and the data block STX an ETB/ETX bytes, are not calculated in the LRC. The example below shows how the LRC is derived for a data block containing three bytes of data.

Parity Bit	Data Byte								
(odd)	8	7	6	5	4	3	2	1	
0	0	0	0	0	0	0	0	1	- 1st data char. transmitted
0	0	0	0	0	0	0	1	0	- 2nd data char. transmitted
	0	0	0	0	0	0	1	1	- XOR result of 1st and 2nd data characters
1	0	0	0	0	0	1	0	1	- 3rd data char. transmitted
	0	0	0	0	0	1	1	0	- LRC, XOR of previous XOR and 3rd data char.

TRANSMISSION TIMING ERRORS

Timing problems between transmitter and receiver can produce other kinds of errors such as overrun, framing, and time-out errors. All of these types of errors are detected by the CCM and reflected by a change in the module Light Emitting Diode (LED) display.

Overrun

If timing problems between the transmitter and receiver cause characters to be sent faster than the receiver can handle them, then this produces a situation known as overrun. In this case the previous character is overwritten and an error is indicated.

Framing Errors

In asynchronous transmission (see section, Asynchronous Transmission) this type of error occurs when the receiver mistakes a logic 0 data bit or a noise burst for a start bit. The error is detected because the receiver knows which bit after the start bit must be a logic 1 stop bit. In the case where the start bit is really a data bit, and the expected stop bit is not the stop bit but a start or data bit the framing error will be reported.

Time-out Errors

Time-outs are used to ensure that a good link exists between devices during a communication. When a source device initiates a communication, the target must respond within a certain amount of time or a time-out will occur causing the communication to be aborted. In a Series Six PLC communication, there are a number of instances during a serial communication in which a time-out can occur. For a detailed explanation of these instances refer to the section (Serial Link Time-out) in Chapter 4, CCM Serial Interface Protocol.

SERIAL TRANSMISSION

Asynchronous serial transmission is used in Series Six PLC Communication Control Modules. Although there is no synchronizing clock used, the transmitting and receiving equipment must be operating at the same bit rate or errors mentioned in the previous section will occur.

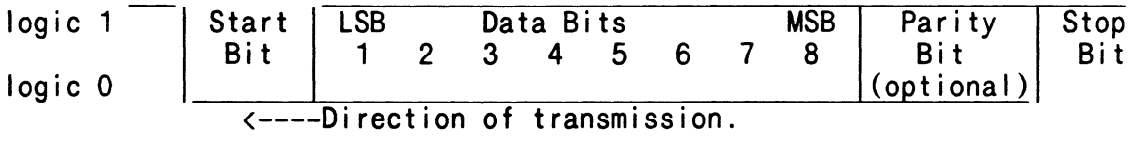
ASYNCHRONOUS TRANSMISSION

The general format for asynchronous communications includes a start bit, seven or eight data bits, an optional parity bit, and a stop bit.

Table 1.3 SERIAL DATA FORMAT

Serial Data Format										
BIT 0	BIT 1	BIT 2	BIT 3	BIT 4	BIT 5	BIT 6	BIT 7	BIT 8	BIT 9	BIT 10
START	LSB <---------- ACTIVE DATA BITS ----------> MSB								(optional) PARITY	STOP
0	<---------- 1 or 0 ---------->									1

When the receiver detects the leading edge of the start bit, which is always logic 0, a timer is triggered to allow sampling to occur in the middle of each bit. After the last data bit (or the parity bit) has been received, the logic state of the line must be a 1 for at least one bit-time before receiving the next character. If no more characters are to be sent, the line will be maintained in the 1 state.



SYNCHRONOUS TRANSMISSION

Synchronous transmission requires the use of a clock to synchronize the transmitter and receiver. Modems for synchronous operation do not use start and stop bits but encode a clock with the data signaling. In addition, a synchronous transmission is preceeded by a unique synchronizing character to assure proper character alignment. The receiver then accepts data until a terminating character is received.

SERIAL COMMUNICATIONS LINE

The serial communications line is the physical medium over which the communications request and data travel. The line may be a direct connection between devices or a connection through modems for long distance communications. The characteristics of the communications line depend on the requirements of the user and the electrical interface standard to which the line is constructed.

MODEMS

The word modem is a acronym of MOdulator/DEModulator. A modem is a device that converts data from digital to analog for transmitting and from analog to digital for receiving over telephone communications lines.

84pc0003

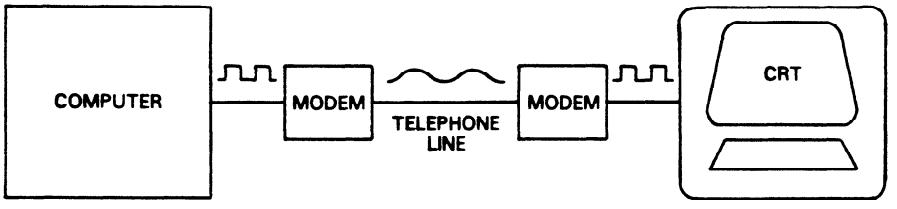


Figure 1.5 MODEMS USED IN THE COMMUNICATIONS LINE

Modems are generally classified as to the type of telephone line facility that can be connected, half- or full-duplex, synchronous or asynchronous, modulation technique for the analog signal, and the maximum data rate in bits per second. Modems were originally designed for and most frequently used with the RS-232D interface.

Communications Modes

There are three modes of communication.

- Simplex - mode in which information can be sent over a communications line in one direction only.
- Half-duplex - mode in which information can be sent in both directions over a communications line, but only one direction at a time.
- Full-duplex - mode in which information can be sent over a communications line in both directions at the same time.

INTERFACE STANDARDS

An interface standard is a set of rules which defines the signal characteristics, cable and connection characteristics, connector pin assignments, and control sequences for a physical link between devices. Series Six PLC serial communications protocols are based on the interface standards explained below.

RS-232D

This standard was developed for interconnecting Data Terminal Equipment (DTE), such as a printer, CRT, or computer, to Data Communications Equipment (DCE), such as a modem, for transmission over a telephone line or network. It can, however, be used over short distances without a modem. Electrically, RS-232D can be described as an unbalanced or single ended voltage interface. This means that all the interchange signals share a common electrical ground. The basic characteristics of RS-232D are:

- Maximum cable length: 50 feet (15 meters)
- Maximum data rate: 20 KBps *
- Logic assignments referenced to signal ground:

  Space or logic 0: +3v to +25v
  Mark or logic 1: -3v to -25v

- Uses 25-pin D-type connector
- Includes 21 interchange circuits including data transmit and receive, data control, and timing. The most commonly used circuits are:

TABLE 1.4 STANDARD (RS-232D) COMMUNICATION INTERFACE SIGNALS

PIN #	FUNCTION	ABBREV.	TYPE	DIRECTION
1	Protective Ground	PROT GND	-	-
2	Transmit data	Txd	Data	From DTE
3	Receive data	Rxd	Data	To DTE
4	Request to send	RTS	Control	From DTE
5	Clear to send	CTS	Control	To DTE
6	Data Set Ready	DSR	Control	To DTE
7	Signal Ground	GND	-	-
8	Recvd Line Sig Det (Carrier Det)	RLSD	Control	To DTE
20	Data Terminal Rdy	DTR	Control	From DTE

* Bps is the number of bits per second transmitted over a communication line.

The RS-232D interface can be used for direct connections not exceeding 50 feet (15 meters). The following illustration shows the lines required for both devices to transmit and to receive.

84pc0005

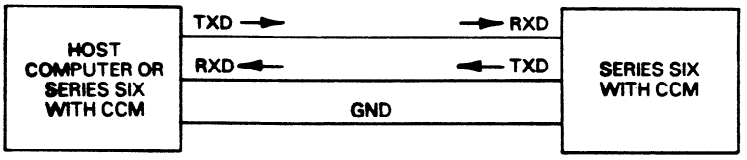


Figure 1.6 RS-232D DIRECT CONNECTION WITHOUT FLOW CONTROL

In this case there is no data flow control; that is, both devices can transmit at any time and there is no check of the communications line before transmission.

When modems are used, without data flow control, again both devices can transmit at any time and there is no check of the transmission line or that the carrier is present.

84pc0006

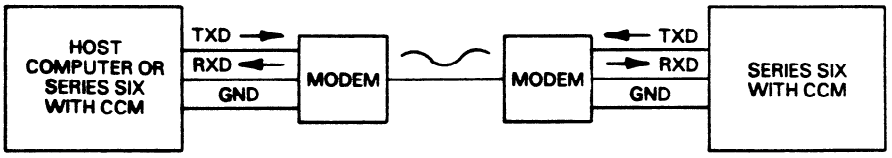


Figure 1.7 RS-232D MODEM CONNECTION WITHOUT FLOW CONTROL

When flow control is desired, the RTS and CTS control circuits can be used to permit the following:

- RTS: The transmitting device can signal the transmitting modem that data is Requested To be Sent.
- CTS: The transmitting modem can signal back to the transmitting device that it is Clear To Send the data.

Refer to Chapter 2 for information on interconnecting the Series Six CCMs via modems. For a complete explanation of control signal usage with modems as well as the electrical and mechanical characteristics of the interface, see Electrical Interface Standard (EIA) RS-232D and the user's manual of the modem to be used in the communications configuration.

RS-449, RS-422, and RS-423

RS-449, RS-422, and RS-423 comprise a "family of standards" reflecting advances in integrated circuit technology. The new standards permit greater distance between equipment and a higher maximum data rate, therefore they are often used for direct connection. RS-422 and RS-423 are standards which define electrical interface characteristics. RS-449 is a standard, used in conjunction with RS-422 and RS-423, which defines the connector pin assignments, cable and connector characteristics, and control signal sequences. RS-423 is an unbalanced voltage interface similar to RS-232D. RS-422 is a balanced or differential voltage interface in which the signal lines are isolated from ground unlike the unbalanced circuit. One of the interface options which can be used in Series Six serial communications is based on the RS-422 and RS-449 standard. The basic characteristics of RS-422 and RS-449 (referenced as RS-422 in this manual) are:

- Maximum cable length: 4000 feet (1200 meters).
- Maximum data rate: 100 KBps at 4000 feet and 10 MBps at 40 feet (12 meters).
- Logic assignments; differential inputs not referenced to ground:

  Space or logic 0: Circuit A is +200 mv to + 6 v with respect to circuit B.
  Mark or logic 1: Circuit A is -200 mv to - 6 v with respect to circuit B.

- 37 pin or 9 pin D-type connector.
- 30 interchange circuits.

The RS-422 signal nomenclature used in this manual can be cross referenced to the RS-422 EIA standard as follows:

Table 1.5 RS-422 SIGNAL CROSS-REFERENCE TO THE EIA STANDARD

FUNCTION	RS-422 STANDARD SIGNAL NAME
Send Data + (TXD+)	B
Send Common - (TXD+)	A
Receive Data + (RXD+)	B'
Receive Common - (RXD-)	A'
Signal Ground	GND

During a mark condition (logic 1), B will be positive with respect to A.
During a space condition (logic 0), B will be negative with respect to A.

For a complete explanation of the electrical and mechanical characteristics of these interfaces, see EIA Standards RS-449, RS-422, and RS-423, and refer to Chapter 2.

Current Loop

There is no true standard for this type of interface. It is normally used when the local environment contains excessive electrical noise from machinery. There are many types of current loop interfaces based on different voltage levels. It is not a modem interface like the RS-232D standard, and generally contains just the transmit and receive data signals. Since there is no proper standard for current loop, the characteristics below are approximations only.

- Maximum cable length: 4000-5000 feet (1200-1500 meters).
- Maximum data rate: 1200 Bps at 4000-5000 feet and 9600 Bps at 500-1000 feet (150-300 meters).
- Logic assignments:

  Polar working: Mark or logic 1 - Current flow in one direction
  Space or logic 0 - Current flow in opposite direction.

  Neutral working: Mark or logic 1 - Presence of current
  Space or logic 0 - Absence of current.

Current loop is only supported on the I/O CCM module.

CHAPTER 2
COMMUNICATIONS CONTROL MODULES (CCM2/CCM3)

INTRODUCTION TO THE CCMs

Communication Control Modules (CCM2 and CCM3) are Series Six™ PLC modules containing--two communications ports, two switches, and four indicator lights--for connection, control, and status of the module. Physically, the CCM2 and CCM3 (CCM) modules are the same. Unless otherwise indicated, CCM applies to both CCM2 and CCM3.

The primary difference between the CCM2 and CCM3 modules is that the CCM3 module supports 2-modes of operation: CCM protocol and Remote Terminal Unit (RTU) protocol. The CCM2 module supports only the CCM protocol. Options for data rate, protocol, turn-around delay, and parity can be selected for both the CCM2 and CCM3 by hardware, using DIP switches, and by software, using configuration registers.

The main purpose of the CCM is to provide a serial interface between the Series Six PLC and any intelligent device which can support communications based on the CCM or RTU protocol and CCM electrical interface requirements. Examples of intelligent devices which can be interfaced to the CCM are:

- DCU in Series One PLC family of controls
- DCM in Series Three PLC family of controls
- CCM2, CCM3, I/O CCM or OptiBASIC OIT
- Host computer or microprocessor based device
- Color-graphics terminal
- GEnet™ Factory LAN (Local Area Network) BIU (Bus Interface Unit)

In addition, the CCM provides an interface to the following:

- Handheld Operator Interface Unit (OIU) which can monitor and modify the CPU registers and I/O points
- Dumb terminal or printer
- Workmaster® or IBM® PC computer
- VuMaster™ color graphics system
- Host device emulating an RTU master

The CCM is capable of initiating data transfers to and from any Series Six PLC memory type including register tables, input and output tables, override tables, scratchpad, and user logic. During these data transfers, the status of the communications link is continuously displayed by the DATA OK light.

If a Series Six PLC with CCM is connected to a host computer or other device that is not a Series Six, the user must write or buy the software necessary to communicate with the CCM module. The details needed to write the communications software to interface a host with the CCM are given in Chapter 4, CCM Serial Interface Protocols. Also, information on communications software packages currently available can be found in Appendix A, Host Computer Interface Software.

The Series Six Plus PLC with expanded microcode increases the number of user addressable I/O points. Expanded microcode allows addressing of channeled I/O points with the Series Six instruction set. The I/O points can be accessed by the CCM2 module in CCM mode, and the CCM3 module in either the CCM or RTU mode. The expanded microcode also allows addressing of the Auxiliary I/O Override table. CCM mode supports this addressing, but RTU mode does not support this feature.

Expanded user memory reference allows addressing up to 64K of the user logic memory. The expanded user logic memory is supported by both the CCM and RTU protocol. Refer to Appendix B, for information concerning Expanded Functions.

GEnet is a Local Area Network (LAN) that provides expanded communication capabilities between various types of processors such as Programmable Logic Controllers (PLCs), Computer Numerical Controllers (CNCs), other high-level factory-management control systems. Interface units compatible with the CCM protocol may access the network using the GEnet Bus Interface Unit (BIU).

MODES OF OPERATION

Two-modes of communication are supported by the Communication Control Modules: CCM protocol for both the CCM2/CCM3 modules, and RTU protocol for the CCM3 module only.

CCM MODE

When the CCM3 is in CCM mode, operation is identical to the CCM2 except that the following protocol options of the CCM2 do not exist on the CCM3.

- RS-422 with clock on port J1
- Test 1 on port J2

These options are not available for the CCM3 because the hardware DIP switch settings and the bit pattern used for the software configuration registers are reserved to select the RTU mode for ports J1 and J2.

RTU MODE

In Remote Terminal Unit (RTU) mode the CCM3 is a slave device designed to link with a host computer or other intelligent device capable of emulating RTU master protocol. When using this mode, the CCM3 is capable of accessing the following Series Six PLC memory types: register tables, input and output tables, override tables, scratchpad, and user logic.

In addition, several Serial Communications REQuests which do not use the CCM protocol (e.g., the Write and Read Character String commands) can be initiated by application programming when using RTU Protocol.

CCM INTERFACE

Both CCM2 and CCM3 provide RS-232D and RS-422 electrical interface capability. RS-232D can be used for direct connections at a maximum distance of 50 feet (15 meters); RS-422, for direct connections up to 4000 feet (1200 meters). The CCM can be connected directly to short haul or telephone line modems via RS-232D if longer transmission distances are required than are capable using RS-422.

Short Haul Modem

This type of modem is used when direct connections over wires can be made in the range of about 5000 to 50,000 feet (1500 to 15,000 meters). It is capable of transmitting up to 9600 Bps and operates in the full-duplex mode.

Telephone Line Modem

This type of long line modem is used over conventional telephone lines or microwaves for virtually unlimited distances at rates of 300 or 1200 Bps in either full or half duplex. The following long line modem types are compatible with the CCM.

- Bell 103
- Bell 212

Concurrent Use of CCM3 in RTU Mode and CCM Mode

One CCM3 communication port can be configured in CCM mode at the same time that the other port is configured in RTU mode. Restrictions regarding the use of the 2 modes concurrently are given in a later section of this chapter, Simultaneous Port Operations.

SYSTEM CONFIGURATION AND PROTOCOL

A system configuration refers to the way in which multiple Series Six PLCs or other elements are combined to form a communications network. The CCM protocol supports three types of system configurations and the RTU protocol supports two types of system configurations as follows:

CCM Protocol	RTU Protocol
• Point-to-point	• Point-to-point
• Multidrop	• Multidrop
• GEnet	

System diagrams which follow show the basic structure of the various configurations. For details on the connecting cables, see section Cable Connectors and Specifications.

POINT-TO-POINT

In the point-to-point configuration only two devices can be connected to the same communication line. The communication line can be directly connected using RS-232D (50 feet, 15 meters maximum) or RS-422 (4000 feet, 1200 meters maximum). Modems can be used for longer distances.

The CCM protocol selection in point-to-point communications can be peer, for peer-to-peer protocol, or master or slave for master-slave protocol. In a peer-to-peer system composed of two CCMs, either of the devices can initiate communications. Several examples of the combination of elements possible with the point-to-point configuration are shown below.

Combination of Elements	Compatible Interface Types
• CCM or RTU mode to computer, process control system, color graphics terminal or other microprocessor based device	RS-232D, RS-422
• CCM to CCM mode	RS-232D, RS-422
• CCM or RTU mode to modem	RS-232D, RS-422
• CCM mode to Operator Interface Unit (OIU)	RS-422
• CCM mode to Dumb Terminal	RS-232D, RS-422
• GEnet to CCM mode	RS-232D, RS-422

CCM to CCM, Modem, Operator Interface Unit, or Dumb Terminal

All of these devices can be connected to the CCM in the same basic forms as shown below.

a42668

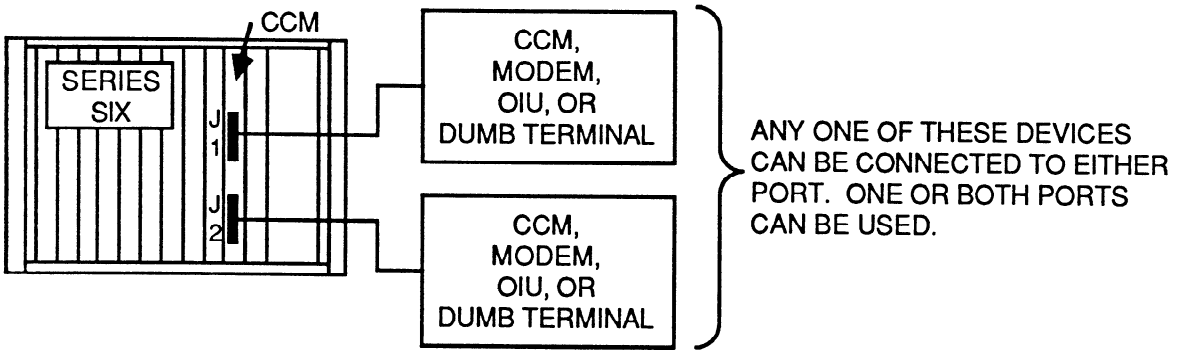


Figure 2.1
CCM TO CCM, MODEM, OIU, OR DUMB TERMINAL SYSTEM CONFIGURATION

CCM to Computer, Color-Graphics Terminal, or Microprocessor Based Device (Direct Connection)

Point-to-point connections between a CCM and a computer, color-graphics terminal, or microprocessor based device is similar to those shown above in Figure 2.1. In this case, however, the number of CCMs which can be connected depends on the communications hardware and software capability of the host device. The RTU mode of operation is capable of RS-232D and RS-422 connections, and either interface can be used as long as the host has the same capability.

a42669

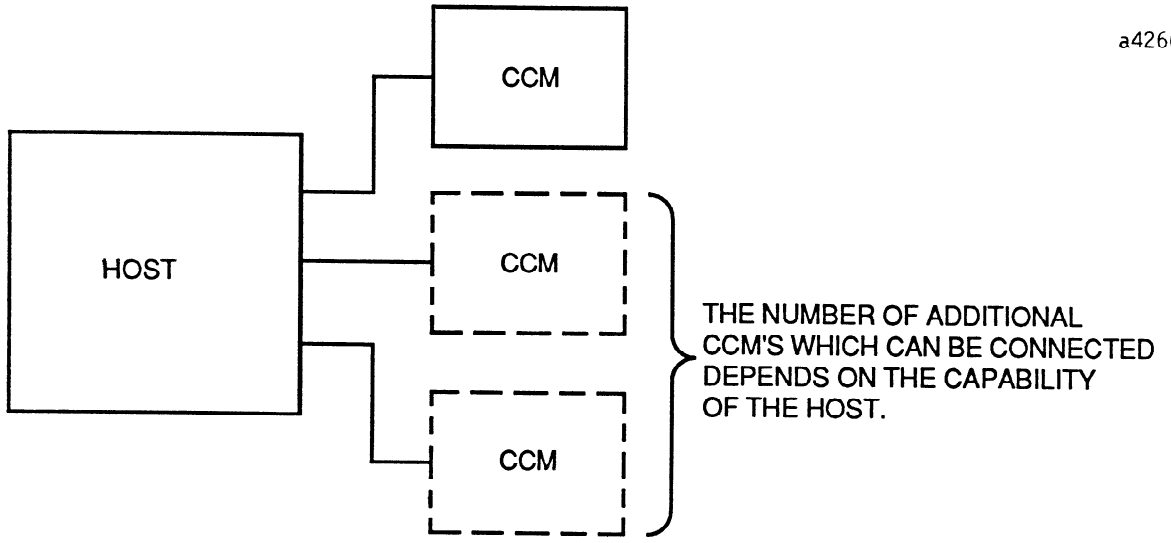


Figure 2.2
CCM TO COMPUTER, COLOR-GRAPHICS TERMINAL,
OR MICROPROCESSOR BASED DEVICE SYSTEM CONFIGURATION

MULTIDROP

In the multidrop configuration, for CCM mode, one CCM or host device is configured as the master and one or more CCMs are configured as slaves; only master-slave protocol can be used. A CCM configured as the master is capable of initiating communications; the slave is not. For the RTU mode of operation, a host device capable of emulating RTU protocol is the master and one or more CCMs using RTU mode are slaves.

Idle slaves continuously monitor the communication link to determine if the line is busy or idle. In the CCM mode, when the line is idle, the slaves will begin looking for new enquiry sequences. Since there is typically more than one slave device sharing the multidrop line, each slave will only recognize enquiry sequences containing its own CPU ID number. For RTU protocol, the slaves will look for a new request. Since there is typically more than one slave device sharing the multidrop line, each slave will process only requests containing its own CPU ID or a broadcast request which is sent to all slaves (CPU ID. 0).

There are three methods for connecting CCMs in the multidrop configuration.

- RS-422 direct
- RS-232D using modems
- RS-232D using modems and radio transmitters

RS-422 Direct

This method can be used when the maximum distance between the master and any slave does not exceed 4000 feet (1200 meters). This figure assumes good quality cables and a moderately "noisy" environment. A maximum of 8 slaves can be connected using RS-422 in a daisy chain or multidrop configuration. The RS-422 line may be of the 2-wire or 4-wire type as shown in the section, Cable and Connector Specifications.

a42670

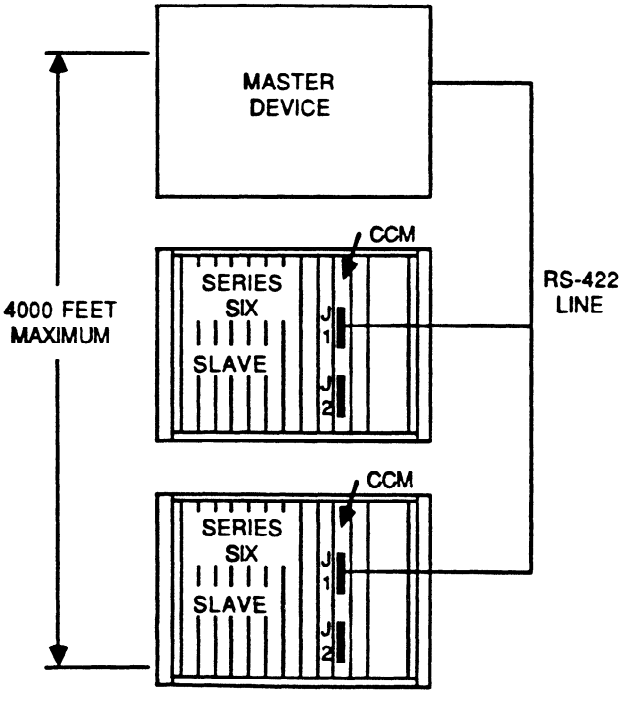


Figure 2.3 RS-422 MULTIDROP CONFIGURATION

RS-232D Using Modems

This configuration is used for long distance communication, primarily over telephone lines. The maximum number of slaves on the line is determined by the modem capabilies. A maximum of 90 slaves is possible with RS-232D using modems in the CCM mode, and 247 for RTU mode.

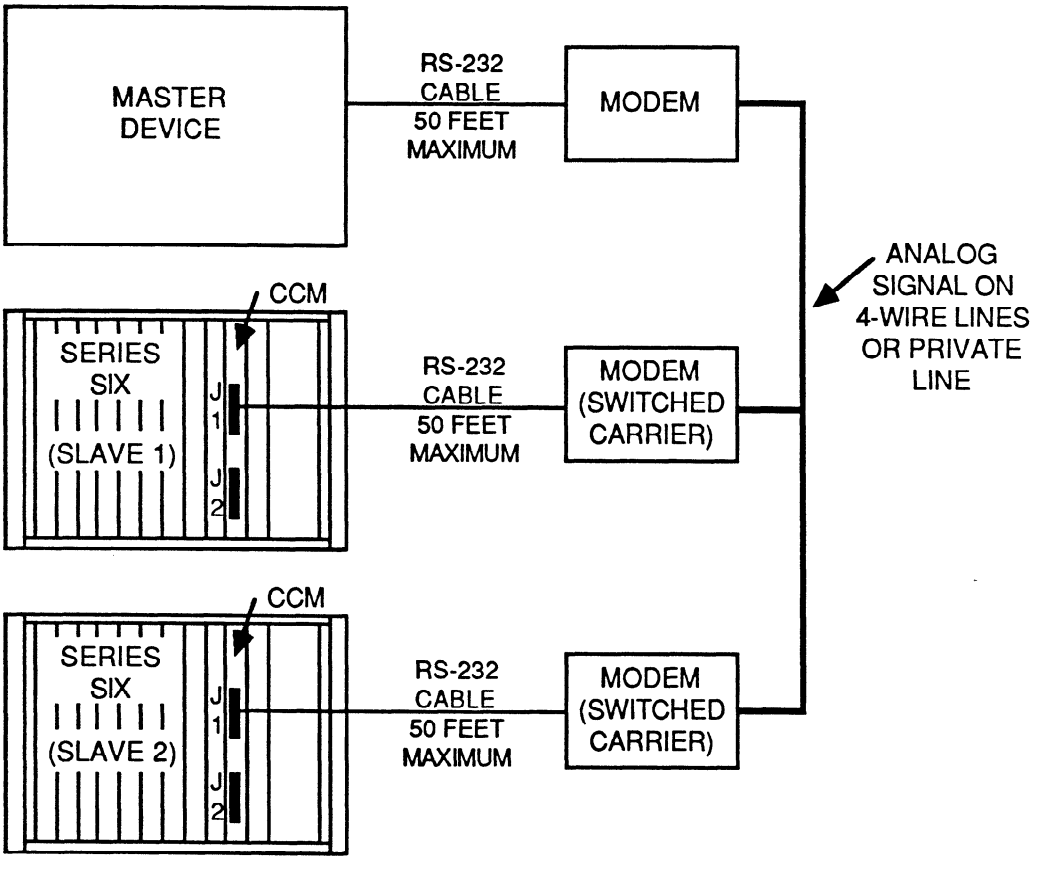


Figure 2.4 RS-232D MULTIDROP CONFIGURATION USING MODEMS

RS-232D Using Modems and Microwave or Radio Transmitters

This configuration is used where cables cannot be used between modems. The FCC normally requires the use of single frequency transmitters with short transmitter-on times. Therefore, a warm-up delay for the radio transmitter must be added before each transmission. The CCM keys the radio transmitter to warm up and wait a short time before actually transmitting the data. The various time-out values for the communication protocol are increased to include the added delay.

The wiring scheme, when using microwave or radio transmitters, depends on the particular modems and transmitters used. Consult your local GE Fanuc Automation salesperson or Application Engineering, for assistance.

GEnet LAN INTERFACE

GEnet is a Local Area Network (LAN) through which many devices can be interconnected. The Series Six PLC can be connected to network with either the GEnet LAN, CCM, or I/O CCM interface modules in the Series Six CPU rack or Bus Interface Unit (BIU).

Each Bus Interface Unit (BIU), which permits access to GEnet, can support a maximum of 16-CCM slaves. If it is desired to interconnect more CCMs, then additional BIUs can be used. A maximum of 254 Series Six PLCs with CCMs can be connected to GEnet.

Figure 1.4, in Chapter 1, shows an overview of the GEnet Factory LAN Interface and some of many devices that can be interconnected to communicate with the network.

For detailed information refer to:

- GEK-96608 *GEnet Factory LAN System User's Manual* provides information concerning the system components and network interconnection.
- GFK-0013 *GEnet Factory LAN Series Six PLC Network Interface User's Manual* provides detailed information for installing, programming and troubleshooting the network.

MODULE SPECIFICATIONS

Space Requirements:	One communications slot in either a Series Six CPU rack or Series Six Plus CPU rack.
Power Requirements:	+5 Vdc, +12 Vdc, -12Vdc (Rack CPU power supply)
	17 4 4 Units of load: CCM2/CCM3
Storage Temperature:	0° C to 70° C
Operating Temperature:	0° C to 60° C (ambient temperature)
Humidity:	5% - 95% (non-condensing)

DESCRIPTION OF THE CCM USER ITEMS

Faceplate

A. Single Pole/Double/Throw Center OFF Switch
B. Single Pole/Double/Throw Center OFF Switch
Switches A and B are used for CCM error diagnostics. (Both switches perform the same function in either the UP or DOWN position)
C. LED Indicators 1 to 4 (Refer to Table 2.10)
D. J1 Connector: 25-pin "D" type female connector for RS-232D and RS-422.
E. J2 Connector: 9-pin "D" type female connector for RS-232D and RS-422.

1. DIP Switches 9 to 16, Configuration Selection for J1 (Reference Table 2.1, 2.4)
2. DIP Switches 1 to 8, Configuration Selection for J2 (Reference Table 2.2, 2.5)
3. DIP Switches 18 to 20, and Miscellaneous Selections (Reference Table 2.3)

4. Jumper JP1: Always set in 1-2 position
5. Jumper JP2: Always set in 1-2 position
6. Jumper JP3: Always set in 1-2 position
7. Jumper JP5: Always set in 1-2 position
8. Jumper JP4: 1-2 position OIU DISABLE
Jumper JP4: 2-3 position OIU ENABLE
9. Jumper JP6: 1-2 position disconnects +5V from pin 20 of Port J1.
Jumper JP6: 2-3 position connects +5V from pin 20 of Port J1.
10. Jumper JP7: Always set in 1-2 position
11. Jumper JP8: Always set in 1-2 position

12. See installation of RS-422 interfaces for terminating resistor configuration
Jumper T2: J2, RS-422 receiver circuit
Jumper T4: RS-422 clock input
Jumper T6: J1, RS-422 receiver circuit
Jumper T8: Always set in storage position

a41538

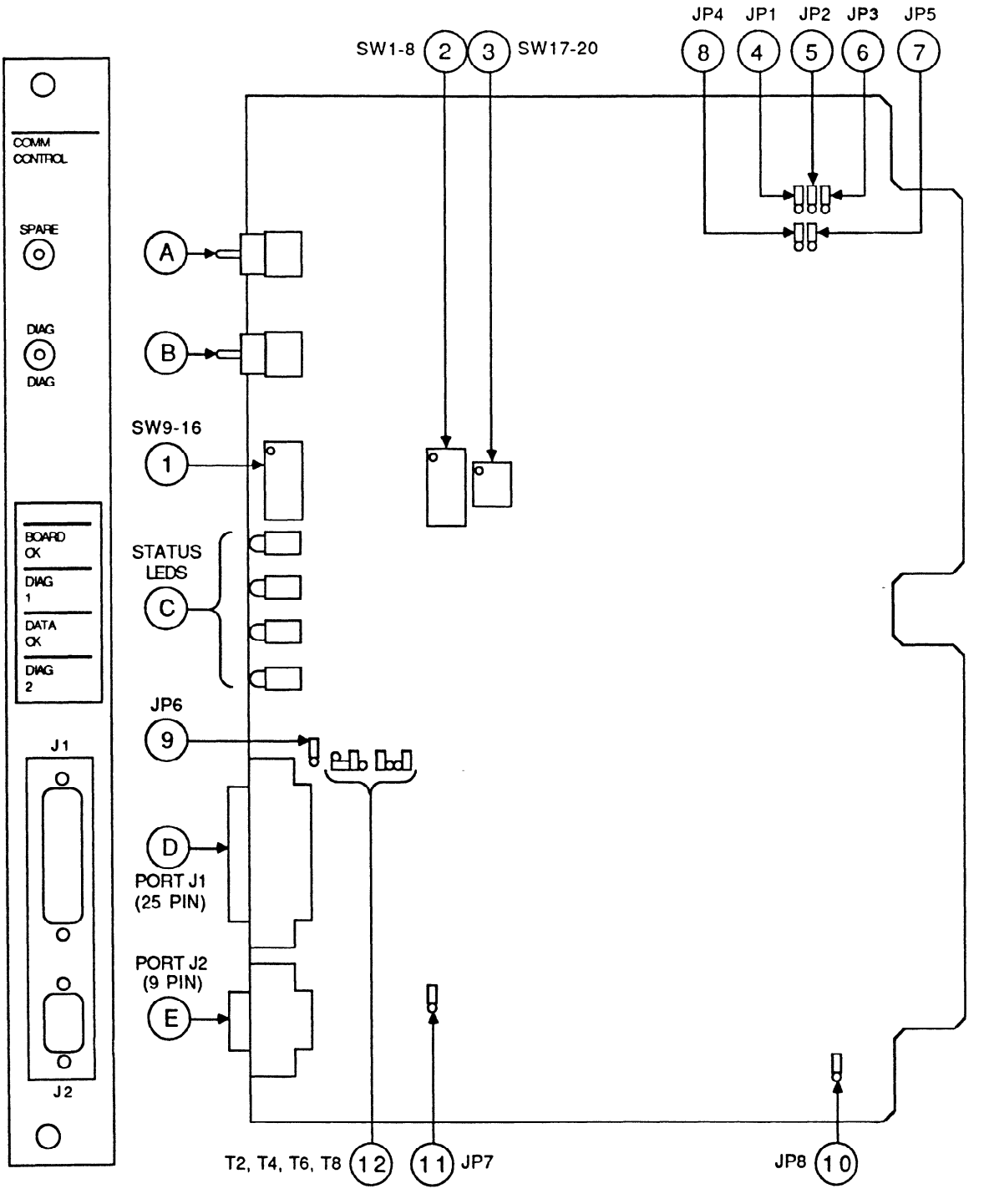


Figure 2.5 CCM LAYOUT AND USER ITEMS

DESCRIPTION OF MODULE FUNCTIONS

A brief description of the CCM communication characteristics is included in this section followed by a complete explanation of each of these functions in later portions of this chapter.

Also, refer to the Module Compatability information located in the Preface of this manual for more information concerning hardware/software features and module compatability.

The CCM communication characteristics may be selected as either hardware or software with the appropriate jumpers and DIP switch selection on the module. If the software configuration is selected, a Series Six programmer (e.g., the Workmaster™) is also required to complete the software configuration.

Selectable CCM module functions are:

- Data Rate (300 to 38.4 KBps)
- Protocol -- CCM and RTU
- Line Interface -- RS-232D, RS-422
- Turn-Around Delay (0 to 500 msec)
- Parity (Odd, Even, or None)

DATA RATE

The data rates available are as listed in tables starting with Table 2.1. Other data rates are provided for special purpose interfaces which include modems or radio transmitters which limit allowable rates.

- 300, 600, 1200, 2400, 4800, 9600,19.2K, 38.4 KBps
- The factory set position is 19.2 KBps.

PROTOCOL

The two-modes of communication are the CCM protocol for the CCM2/CCM3 module, and the RTU protocol for the CCM3 module.

CCM Protocol

The CCM protocol options are:

- Peer-to-Peer
- Master-Slave
- Test 1

Peer-To-Peer

A CCM module configured as peer for peer-to-peer communications can communicate with any other device configured as a peer. The peer-to-peer configuration allows either peer device to initiate a communication request.

Master-Slave

In the CCM mode, the CCM may be configured either as the master or slave device. When a CCM is configured as a master, for master-slave communications, the CCM can only communicate with another device or multiple devices configured as a slave. Only a master can initiate a communication request.

When the CCM is configured as a slave, the CCM can only communicate with another device configured as a master. A slave responds only to a communication request from a master.

Test 1

Test 1 is a special configuration used for test diagnostics. These diagnostics are explained in a later section; CCM Power-Up Diagnostic Tests.

RTU Protocol

RTU protocol is a master-slave protocol whereby the CCM3 module can be configured as a RTU slave. It is used on a link with a process controller, computer, or other intelligent device capable of emulating RTU master protocol.

Only the master can initiate a communications request when RTU protocol is used. There are, however, a limited number of serial communications requests which do not use the CCM protocol that can be initiated by the application program.

The RTU function options can be configured by hardware, using jumpers and DIP switches; or by software, using configuration registers R0247 and R0248.

LINE INTERFACES

The CCM line interface options are RS-232D and RS-422. Specific line interfaces for the CCM2 and CCM3 modules are as follows:

CCM2 Module	CCM3 Module
RS-232D	RS-232D
RS-422	RS-422
RS-422 with clocks	

RS-232D

The RS-232D interface may be selected for the CCM mode with either master-slave or peer-to-peer protocol, but slave protocol only for the RTU mode. When making direct connections using RS-232D, the CTS (clear to send) and RTS (request to send) lines can be used if connected to a device which supports them or they can be disabled by jumpering them together on both ends of the connecting cable. When connecting through modems, CTS and RTS might or might not be used depending on the type of modem. The RTS and CTS signals correspond to the standard Data Terminal Equipment (DTE) usage as explained as follow.

- When the CCM has nothing to transmit, the handshake output line (RTS) is in the false state.
- When the CCM has received a command to transmit some data, the handshake output line is set to true.
- After an optional turn-around delay, the CCM will check the handshake input line (CTS) and begin transmitting the data if the handshake input line is true.
- When the CCM has no more data to transmit, the handshake output line (RTS) will be set false after the last data character is transmitted.
- If the handshake input line (CTS) changes back to false before the CCM is finished transmitting, the CCM will stop transmitting at a character boundary and wait for the handshake input line (CTS) to change back to true.
- When flow control is used, the device implementing it must also guarantee that (CTS) will become false anytime (RTS) is set to false at the end of a data block.

These rules explain the transmit function only. The standard DTE data receive function is independent of the RTS and CTS handshake lines. The DTE is able to receive data at any time.

RS-422

The RS-422 interface may be selected for the CCM mode with either master-slave or peer-to-peer protocol, but slave potocol only for the RTU mode. This type of interface is used primarily for direct connection for both point-to-point and multidrop links. The total length of cable that can be used on either point-to-point links or multidrop links (including all drops) is 4000 feet (1200 meters).

The CTS/RTS flow control works for RS-422 links also. When making direct connections, the CTS/RTS lines may be jumpered together on both ends of the connecting cable.

RS-422 With Clock

This interface is supported for peer-to-peer protocol on a CCM2 module only. Only, CCM2 Port J1, provides for the use of external synchronizing clocks. These clock signals are used with synchronous modems. The CCM2 outputs a clock signal to the modem corresponding to the data rate. The CCM2 in turn uses the incoming clock signal from the modem to synchronize on incoming data.

TURN-AROUND DELAY

This refers to a delay in the amount of time before sending a control character, start of header, or start of a data block for the CCM protocol. The delay options for CCM protocol are as follows:

- 0 msec. for any CCM to CCM connection
- 10 msec. for situations causing slow response times
- 500 msec. for radio transmission
- 500 msec. with time-outs disabled for testing

Keying Signal

Pin 11 on the J1 port provides a keying signal for radio transmission. The keying signal allows the radio transmitter to warm up for the length of the turn-around delay before data begins flowing from the CCM. The CCM serial link time-outs are also lengthened by the amount of the turn-around delay.

Time-Outs Disabled

Time-outs are used on the serial link for error detection, error recovery, and to prevent the missing of end-of-block sequences. In the event of any link time-out, the CCM will abort the communication and send an end-of- transmission character (EOT). When the 500 msec. turn-around delay with time-outs disabled is selected, time-out error conditions are ignored.

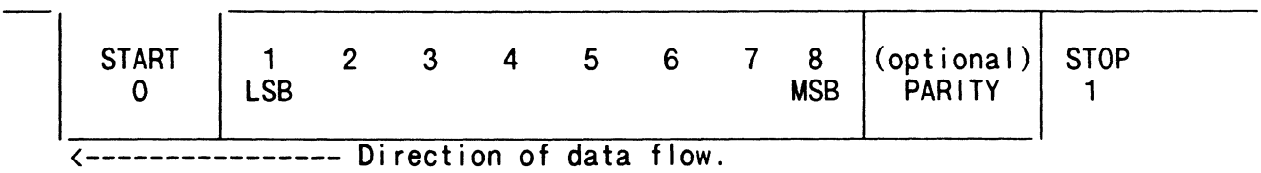
PARITY

The parity selection for serial data transmission is odd, even, or no parity for CCM and RTU protocol specified as follows:

CCM Protocol - Odd, None
RTU Protocol - Odd, Even or None

The data is divided into 8-bit bytes and transferred using an asynchronous format. This format consists of one start bit, 8 data bits, one parity bit (optional), and one stop bit. When enabled, the parity is odd for the CCM mode. When enabled for the RTU mode, the parity may be odd or even. If parity is disabled the parity bit is not transmitted.

The serial data format for both the CCM and RTU protocol is shown below.



OPERATOR INTERFACE UNIT (OIU)

The OIU is a handheld terminal which can monitor and change the contents of the CPU through the CCM. OIU interface selections are Enable, Disable, and Connect or Disconnect power to the OIU from the CCM. The OIU is supported for both CCM2 and CCM3 modules when operating in the CCM mode. Refer to the section, Operator Interface Unit (OIU), for more information.

NOTE

The Operator Interface Unit (OIU) is not supported for the CCM3 when the module is configured as RTU protocol. Refer to sections: Module Configuration and Configuring the CCM for OIU Operation.

MODULE CONFIGURATION

The CCM module functional options can be configured by hardware, using jumpers and Dual-In-Line (DIP) switches; or by software, using configuration registers R0247 and R0248. Selection of the CCM functional operation is explained in the tables on the following pages.

- Hardware Configuration
- Software Configuration

Complete the hardware/software module configuration prior to installing the CCM module into the Series Six CPU.

HARDWARE CONFIGURATION

Terminating resistors, hardware jumpers, and Dual-In-Line (DIP) switches located on the CCM are used to select desired option within each function. Before installing the module into the PLC rack, select the desired options.

- Set the on-board DIP Switches
- Verify Terminating Resistors

DIP Switch Settings

The CCM module DIP switches are used to select the desired option within each function. Hardware configuration tables on the following pages, shows the options available for the CCM and RTU modes of operation. All options except the required positions (as indicated) can be changed to meet user needs.

Refer to the Configuration Tables beginning with Table 2.1, and Figure 2.6 Hardware Configuration Diagram.

Terminating Resistors

The CCM module is also is supplied with a 150 Ohm terminating resistor in each RS-422 receiver circuit. If the module is at either end of an RS-422 multidrop or point-to-point link, these resistors should be in the circuit. If the module is an intermediate drop in the multidrop link, the appropriate resistors should be removed from the circuit by placing their jumpers in the storage position. (Refer to Table 2.3, CCM Hardware Configuration and the Description of the CCM User Items.)

The first column of the DIP switch configuration tables identifies the module function. Columns to the right define the position of the DIP switches for a particular option.

Switches are identified by the numbers located on the module to the left and right of the switch package. The switch numbers in parenthesis are located on the switch package itself and are included only as an aid in configuring the module.

Table 2.1 CCM PROTOCOL HARDWARE CONFIGURATION TABLE - PORT J1

FUNCTION PORT J1	O-Open C-Closed X-Don't Care	SWITCHES * 9 (1)	10 (2)	11 (3)	12 (4)	13 (5)	14 (6)	15 (7)	16 (8)	18 (2)	19 (3)	20 (4)
Data Rate	300	O	O	O								
	600	C	O	O								
	1200	O	C	O								
	2400	C	C	O								
	4800	O	O	C								
	9600	C	O	C								
	19.2K	O	C	C								
	38.4K	C	C	C								
Protocol	Master RS-232D				O	O	O					
	Master RS-422				C	O	O					
	Slave RS-232D				O	C	O					
	Slave RS-422				C	C	O					
	Peer RS-232D				O	O	C					
	Peer RS-422				C	O	C					
	Peer RS-422 With CLK (CCM2 only)				O	C	C					
	Software Config. Mode **				C	C	C					
Turn Around Delay												
	0 msec full duplex							O	O			
	10 msec half duplex							C	O			
	500 msec half duplex							O	C			
	500 msec with time-outs disabled							C	C			
Required Setting										X	X	O
Parity Selection	(Always odd when using hardware configuration. To select no parity for port J1, see, CCM Software Configuration.)											

* Numbers without parenthesis are the switch numbers shown on the board silk screen. Numbers in parenthesis are located on the dip switch package.

** Switch 17 must be CLOSED (C) for software configuration. (See Table 2.6)

Table 2.2 CCM PROTOCOL HARDWARE CONFIGURATION TABLE - PORT J2

FUNCTION PORT J2	O-Open C-Closed	SWITCHES * 1 (1)	2 (2)	3 (3)	4 (4)	5 (5)	6 (6)	7 (7)	8 (8)	17 (1)
Data Rate										
	300	O	O	O						
	600	C	O	O						
	1200	O	C	O						
	2400	C	C	O						
	4800	O	O	C						
	9600	C	O	C						
	19.2K	O	C	C						
	38.4K	C	C	C						
Protocol										
	Master RS-232D				O	O	O			
	Master RS-422				C	O	O			
	Slave RS-232D				O	C	O			
	Slave RS-422				C	C	O			
	Peer RS-232D				O	O	C			
	Peer RS-422				C	O	C			
	Test 1 (CCM2 only)				O	C	C			
Turn Around Delay										
	0 msec full duplex							O	O	
	10 msec half duplex							C	O	
	500 msec half duplex							O	C	
	500 msec with time-outs disabled							C	C	
Parity Selection										
	Odd									C
	None									O

* Numbers without parenthesis are the switch numbers shown on the board silk screen. Numbers in parenthesis are located on the dip switch package.

The first column of the configuration table identify the module function. Columns to the right define the jumper position for a particular option.

Resistor positioning: Resistor IN the circuit if the CCM module is at either end of an RS-422 multidrop or point-to-point link. Resistor REMOVED when the module is an intermediate drop in the multidrop link.

Table 2.3 HARDWARE CONFIGURATION TABLE (CCM and RTU MODE)

FUNCTION	SWITCHES *	
	18 (2) 19 (3) 20 (4)	
Required Settings (both ports) (x - Don't care)	x x 0	
	PINS JUMPERED	JUMPER
Required Settings	1-2	JP1
	1-2	JP2
	1-2	JP3
	1-2	JP5
	1-2	JP7
	1-2	JP8
OIU **		
Enabled	2-3	JP4
Disabled	1-2	
OIU Power (+5 v to pin 20 of J1)		
Connect	2-3	JP6
Disconnect	1-2	
Terminating Resistors (For RS-422 circuits)	JUMPER POSITION Resistor IN	Resistor OUT JUMPER
J2 RS-422 receiver		T2
J1 RS-422 clock input		T4
J1 RS-422 receiver		T6
Required setting		T8

* Numbers without parenthesis are the switch numbers shown on the board silk screen. Numbers in parenthesis are located on the dip switch package.

** Not supported when module configured for RTU mode of operation.

Table 2.4 RTU PROTOCOL HARDWARE CONFIGURATION TABLE - PORT J1

FUNCTION PORT J1	O-Open C-Closed	SWITCHES* 9 (1)	10 (2)	11 (3)	12 (4)	13 (5)	14 (6)	15 (7)	16 (8)	
Data Rate										
	300	O	O	O						
	600	C	O	O						
	1200	O	C	O						
	2400	C	C	O						
	4800	O	O	C						
	9600	C	O	C						
	19.2K	O	C	C						
	38.4K	C	C	C						
Protocol										
	RTU (CCM3 only)				O	C	C			NOTE: Switch 17
	Software Configuration				C	C	C			must be C for Software Configuration
Line Interface										
	RS-232D							O		
	RS-422							C		
Parity Selection										
	Odd								O	
	None								C	

* Numbers without parenthesis are the switch numbers shown on the board silk screen. Numbers in parenthesis are located on the dip switch package.

Table 2.5 RTU PROTOCOL HARDWARE CONFIGURATION TABLE - PORT J2

FUNCTION PORT J2	O-Open C-Closed	SWITCHES* 1 (1)	2 (2)	3 (3)	4 (4)	5 (5)	6 (6)	7 (7)	8 (8)	17 (1)
Data Rate										
	300	O	O	O						
	600	C	O	O						
	1200	O	C	O						
	2400	C	C	O						
	4800	O	O	C						
	9600	C	O	C						
	19.2K	O	C	C						
	38.4K	C	C	C						
Protocol										
	RTU (CCM3 only)				O	C	C			
Line Interface										
	RS-232D							O		
	RS-422							C		
Parity Selection										
	Odd								O	C
	Even								C	C
	None								O	O
	None								C	O

* Numbers without parenthesis are the switch numbers shown on the board silk screen. Numbers in parenthesis are located on the dip switch package.

NOTE

When using hardware configuration, the port OIU selections are as follows:

J1 Port - Operator Interface Unit: OIU enabled, OIU non-polled, OIU memory protect enabled.

J2 Port - Dumb Terminal: Dumb terminal enabled, dumb terminal non-polled, dumb terminal memory protect enabled.

84pc0036

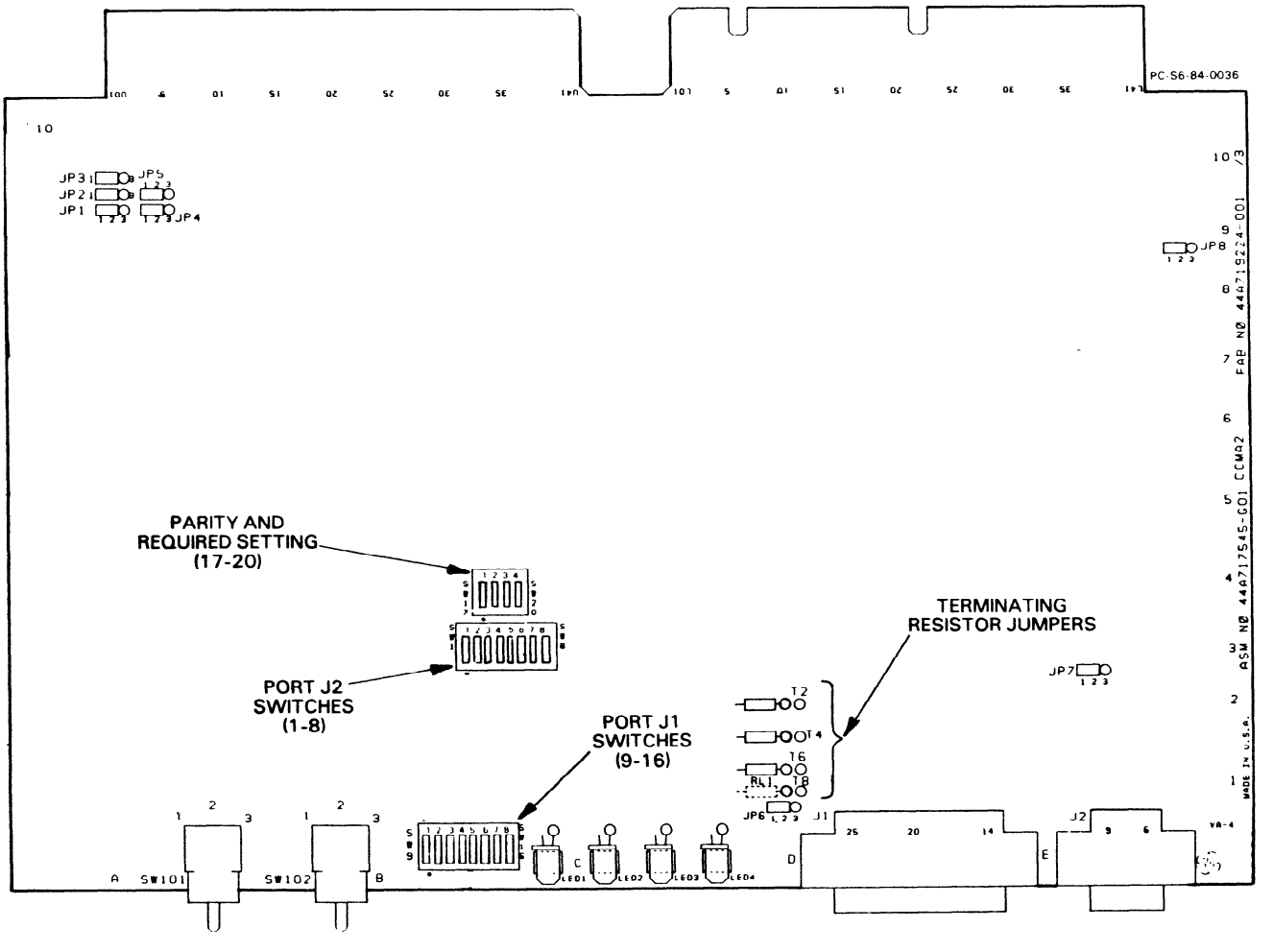


Figure 2.6 CCM HARDWARE CONFIGURATION DIAGRAM

SOFTWARE CONFIGURATION

The CCM can be configured by CPU registers R0247 and R0248 when in the Software Configuration Mode. To enter the Software Configuration Mode, first position module switches 12, 13, 14, and 17 as shown below.

Table 2.6 SOFTWARE CONFIGURATION MODE

FUNCTION	SWITCHES 12	13	14	17
Software Configuration Mode	C	C	C	
Odd Parity				C

C = Switch CLOSED

Also, ensure that the switches or jumpers shown in Table 2.3 for "Required Settings" are properly positioned and that the jumpers for "OIU Power" and "Terminating Resistors" are positioned according to the requirements of the user application.

When in the software configuration mode, register R0247 represents the configuration for serial port J1 and register R0248 represents the configuration for serial port J2. The format of the configuration data as shown below is exactly the same for both registers. Tables 2.7 and 2.8 shows the bit patterns required for selecting module options.

The format for the CCM protocol configuration data is:

16	15 14 13 12 11 10	9	8 7	6 5 4	bits 3 2 1
Port Enable/ Disable	OIU	Parity (optional)	Turn Around Delay	CCM Protocol	Data Rate

The format for the RTU protocol configuration data is:

16	15	14 13 12 11	10 9	8 7	6 5 4	bits 3 2 1
Port Enable/ Disable	Line Inter- face	Required Settings	Parity (optional)	Not Used	RTU Protocol	Data Rate

On-Line Reconfiguration

If the CCM is idle (i.e., not executing a SCREQ or serial conversation) the CCM will reinitialize the serial ports on a regular basis, once per second. When using software configuration, the reinitialize routine in the CCM will read the configuration data from the CPU registers to configure the serial ports.

In order to reconfigure the module on-line, the application program for the Series Six or external device must change the configuration registers and then ensure that the CCM is idle for a minimum of 3 seconds.

Table 2.7 CCM PROTOCOL SOFTWARE CONFIGURATION TABLE - BIT PATTERN

FUNCTION	BITS 16 15 14 13 12 11 10 9	8	7	6	5	4	3	2	1
Data Rate									
300							0	0	0
600							0	0	1
1200							0	1	0
2400							0	1	1
4800							1	0	0
9600							1	0	1
19.2K							1	1	0
38.4K							1	1	1
Protocol									
Master RS-232D				0	0	0			
Master RS-422				0	0	1			
Slave RS-232D				0	1	0			
Slave RS-422				0	1	1			
Peer RS-232D				1	0	0			
Peer RS-422				1	0	1			
Peer RS-422 With Clk (CCM2, J1 only, R0247)				1	1	0			
Test 1 (CCM2, J2 only, R0248)				1	1	0			
Turn Around Delay									
0 msec full duplex		0	0						
10 msec half duplex		0	1						
500 msec half duplex		1	0						
500 msec with time-outs disabled		1	1						

Table 2.7 (Continued)
CCM PROTOCOL SOFTWARE CONFIGURATION TABLE - BIT PATTERN

FUNCTION	* * BITS 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1
OIU Device[1]	
OIU	0
Dumb Terminal	1
OIU Enable/Disable	
Enable	0
Disable	1
OIU Polled/Non-Polled	
Non-Polled	0
Polled	1
OIU Memory Protect	
Enable	0
Disable	1
Port Enable/Disable	
Enable	0
Disable	1

1 When OIU or "dumb" terminal mode is selected, the CCM operates in a 7-bit even parity format. If a "dumb" terminal is used it must be configured as a 7-bit even parity device.

* Bits 11 and 12 are not used.

Table 2.8 RTU PROTOCOL SOFTWARE CONFIGURATION TABLE - BIT PATTERN

FUNCTION	BITS																
	17	16	15	14	13	12	11	10	9	8 **	7 **	6	5	4	3	2	1
Data Rate																	
300															0	0	0
600															0	0	1
1200															0	1	0
2400															0	1	1
4800															1	0	0
9600															1	0	1
19.2K															1	1	0
38.4K															1	1	1
Protocol*																	
RTU (CCM3)												1	1	0			
Parity Selection																	
Odd									0	1							
Even									1	1							
None									0	0							
None									1	0							
Required Settings																	
				0	1	0	0										
Line Interface																	
RS-232D			0														
RS-422			1														
Port Enable/Disable																	
Enable		0															
Disable		1															

* Refer to the note earlier in this chapter pertaining to the Operator Interface Unit (OIT).

** Bits 7 and 8 are not used.

INSTALLING THE CCM MODULE

The Communication Control Module (CCM) can be installed in a Series Six Plus, model 60, 600 or 6000 CPU rack.

- In the Series Six model 60, 600 or 6000 the second slot to the left of the power supply is reserved for the CCM module, and is the only position where the CCM can be installed.
- In the Series Six Plus PLC, slot 5 or 6 may be used for the CCM module.
- In the Series Six Plus II (13" or 19" shallow rack) PLC, either slot 5 or 6 may be used for the CCM module.

  When installing the CCM module into either the Series Six Plus or Series Six (shallow rack) PLC, set the backplane DIP switch package (all switches OPEN).

NOTE

Refer to the Module Compatability information located in the Preface of this manual for more information concerning hardware/software features and module compatability.

1. With CPU rack power turned off, install the CCM module into the logic rack using the extraction/insertion tool furnished with the Series Six PLC. Refer to The Series Six PLC rack layout (Figures 2.7 and 2.8)

84pc0016

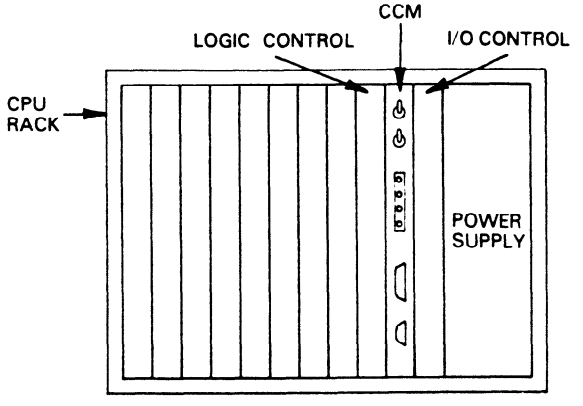


Figure 2.7 CCM LOCATION IN SERIES SIX PLC

INSTALLING THE CCM MODULE (continued)

The product structure for the Series Six Plus PLC is such that many different configurations, including combinations of I/O modules, may be contained in a single CPU rack. The following figure shows the Series Six Plus PLC rack layout.

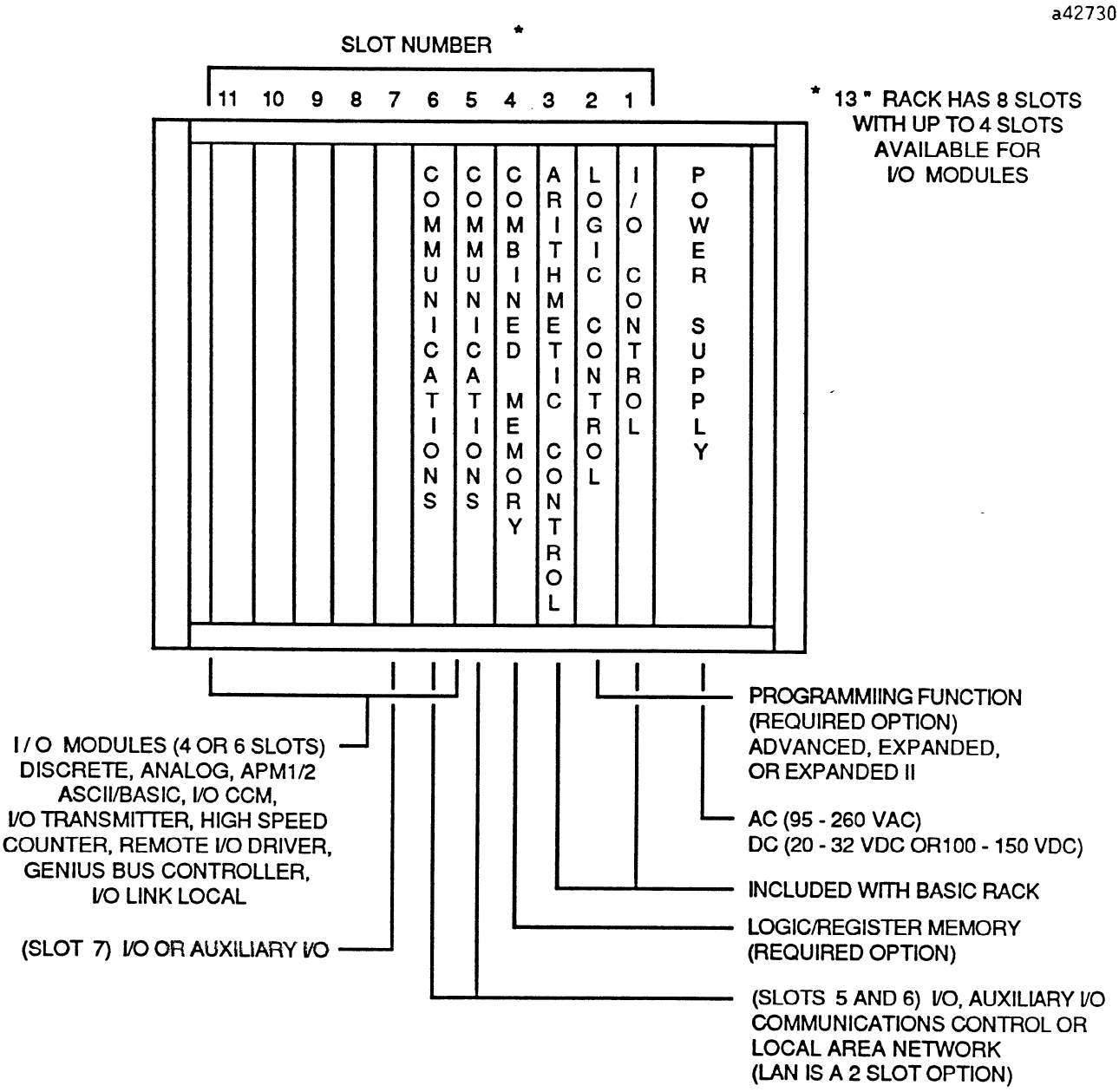


Figure 2.8 CCM LOCATION IN SERIES SIX PLUS PLC

INSTALLING THE CCM MODULE (continued)

2. Construct and install the CCM communication cable for port J1 or J2. Refer to Table 2.10 Port Characteristics (J1, J2) pin-out definition.

3. Power up and test the CCM to verify that the module is operating properly.

   To determine if the CCM is working properly, power up the module with factory settings for the jumpers and switches. This will cause a short diagnostic test to be performed by the CCM. The four lights on the faceplate cycle ON and OFF in a pattern indicating the progress and results of the diagnostic test. At the end of the test all lights should remain ON to show its successful completion. A further explanation of this test can be found in the following section, Power-Up Diagnostic Testing.

NOTE

Some older Series Six CPUs require a modification to operate with the CCM. If you have a Model 60 or 600 manufactured before fiscal week 38, 1981 or a Model 6000 manufactured before fiscal week 44, 1981, contact GE Fanuc Automation about the modification. To determine the date of manufacture, first locate the serial number on the CPU. The date of manufacture is indicated by the four numbers following C188, the first two of which indicate the year and the second two, the fiscal week.

Also, refer to the Module Compatability information located in the Preface of this manual for more information concerning hardware/software features and module compatability.

The ladder logic examples and programming information provided later in this chapter may also be used to verify that the CCM is communicating properly. Refer to the section later in this chapter, CPU/CCM Programming.

POWER-UP DIAGNOSTIC TESTING

The user accesses the CCM module through the faceplate controls which consist of two diagnostic switches for CCM error detection, 4 status and diagnostics indicator lights, and two serial ports, J1 and J2. (Refer to Figure 2.5 CCM Module Layout and User Items)

INDICATOR LIGHTS

The 4 LED indicator lights (BOARD OK, DIAG 1 and 2 and DATA OK) show port activity and module status. (Refer to Table 2.9 LED Indicator Power-Up Codes)

Table 2.9 LED INDICATOR POWER-UP CODES

LIGHT	CAUSE OF ERROR (o Light ON, • Light OFF)			
	CCM POWER-UP RAM TEST FAILED	CCM USART FAILED TO INITIALIZE	CCM PROM TEST FAILED	CPU/CCM COMMUNICATIONS FAILED
(1) BOARD OK	•	o	•	o
(2) DIAG 1	o	•	•	o
(3) DATA OK	o	o	o	•
(4) DIAG 2	o	o	o	o

1. BOARD OK (Module Status)

STATUS	DESCRIPTION
ON	Board has passed the self-check test and is operating properly.
FLASHING	Invalid configuration or invalid CPU ID number, 0 or greater than 90 for CCM slave mode, 0 or greater than 247 for RTU mode (CCM3 only). The configuration or the CPU ID must be changed and the module powered up again to recover.
OFF	Board has failed power-up test indicating a hardware failure or the CCM failed to communicate with the Series Six CPU. If the BOARD OK light goes off as a result of a major CCM error, further information about the specific cause of the error can be obtained by toggling the front panel switch. When this is done, the 4 indicator lights will create one of the patterns as in Table 2.9. • If it is a hardware failure then the CCM is inoperable and the LED will turn on again only after successful completion of the power-up test. • If at some time after a successful power-up there is a CCM/CPU communication failure, both the BOARD OK and the DATA OK LED will turn off. In this case additional information from the panel LEDs cannot be obtained. Both LEDs will turn on again upon successful communication with the CPU.

2. DIAG 1 (CCM Error Diagnostic)

STATUS	DESCRIPTION
ON	Passed powerup diagnostics. ON during normal operation.
	Cycles ON and OFF during powerup then remains ON.
OFF	May change states when toggling Switch A or B.

3. DATA OK (Serial Data Transmission Link)

STATUS	DESCRIPTION
ON	Data transmission normal.
FLASHING	The LED will flash as serial data is actually transmitted.
OFF	Data transmission is incorrect for one or more of the following reasons. - Parity, overrun, or framing errors. - Invalid header, data block, control character, or checksum. In those cases the LED will turn ON again after a successful session has been completed between the CCM and the external device or if the module power is cycled. A CCM/CPU communications failure will cause this LED and the BOARD OK LED to turn OFF. See BOARD OK.

4. DIAG 2 (CCM Error Diagnostic)

STATUS	DESCRIPTION
ON	Passed powerup diagnostics. ON during normal operation.
	Cycles ON and OFF during powerup then remains ON.
OFF	May change states when toggling Switch A or B.

ELECTRICAL INTERFACE CIRCUITS

The CCM module supports two types of system cable configurations Point-to Point and Multidrop.

In the Point-to-Point configuration only two devices can be connected to the same communication line. The communication line can be directly connected using RS-232D (50 feet, 15 meters maximum) or RS-422 (4000 feet, 1200 meters maximum). Modems can be installed for longer distances.

When configured for CCM mode, in the multidrop configuration, more than two devices can be connected to the same communication line. One CCM or host device is configured as a master and one or more CCMs are configured as slaves. In the RTU mode, a host computer is configured as a master and one or more CCMs are configured as slaves. A master is capable of initiating communications; a slave is not. There are three ways to connect CCMs in the multidrop configuration: RS-422 direct, RS-232D using modems, and RS-232D using modems and microwave or radio transmitters.

RS-422 Direct: This method can be used when the maximum distance between the master and the last slave does not exceed 4000 feet (1200 meters). This distance assumes good quality cables and a moderately "noisy" environment. A maximum of eight slaves can be connected using RS-422 in a daisy chain or multidrop configuration. The RS-422 line may be of the 2-wire or 4-wire type.

RS-232D Using Modems: This configuration is used for long distance communication, primarily over telephone lines. The number of slaves possible is determined by the modem capabilities.

RS-232D Using Modems and Microwave or Radio Transmitters: This configuration is used where cables cannot be used between modems. The FCC normally requires the use of single frequency transmitters with short transmitter-on times.

PORT CHARACTERISTICS

There are 2 ports on the CCM module. The J1 port is a 25-pin, female, D-type connector and the J2 port is a 9-pin, female, D-type connector. The pin definitions for each port are given below.

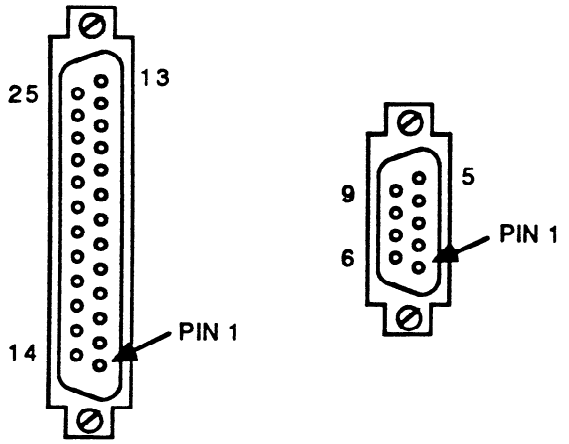


Figure 2.9 CONNECTOR CONFIGURATION (PORTS J1, J2)

Table 2.10 PORTS (J1, J2) PIN-OUT DEFINITIONS

Pin No.	Port J1	Pin No.	Port J2
1	*	1	RS-422 data out (+)
2	RS-232D data out	2	RS-232D data out
3	RS-232D data in	3	RS-232D data in
4	RS-232D request to send	4	RS-232D request to send
5	RS-232D clear to send	5	RS-232D clear to send
6	*	6	RS-422 data out (-)
7	Signal Ground	7	Signal Ground
8	*	8	RS-422 data in (+)
9	*	9	RS-422 data in (-)
10	*		
11	Keyout I/O		
12	+12 volts resistive		
13	RS-422 data in (+)		
14	RS-422 data in (-)		
15	*		
16	*		
17	RS-422 data out (-)		
18	RS-422 data out (+)		
19	OIU ground		
20	OIU +5 volts (fused at 5 A)		
21	RS-422 clock in (+)		
22	-12 volts resistive		
23	RS-422 clock in (-)		
24	RS-422 clock out (+)		
25	RS-422 clock out (-)		

* Do not connect

CABLE AND CONNECTOR SPECIFICATIONS

- Cable connector to CCM Port J1 - Male, D-Subminiature Type, Cannon DB25P (solder pot) with DB110963-3 Hood or equivalent (standard RS-232D connector)
- Cable connector to CCM Port J2 - Male, D-Subminiature Type, Cannon DE9P (solder pot) with DE110963-1 Hood or equivalent
- Length, Maximum - 50 feet (15 meters) for RS-232D
  4000 feet (1200 meters) for RS-422
- Overall shield
- 24 AWG (minimum)
- Connector to external device - specified by external device manufacturer
- Cable Selection

  The following cables provide acceptable operation at data rates up to 19.2K BPS for RS-232D and distances up to 4000 feet for RS-422.

  Belden 9184 Belden 9302 NEC 222P1SLCBT

  At shorter distances (under 1000 feet, 300 meters) almost any twisted pair or shielded twisted pair cable will work as long as the wire pairs are connected correctly.

  When using RS-422, the twisted pairs should be matched so that both transmit signals make up one twisted pair and both receive signals make up the other twisted pair. If this is ignored, then cross-talk can result from the mis-matching which may affect the performance of the communication system.

  Best results have been obtained with General Semiconductor Industries Transzorb SA series wired from each signal line to earth ground at both ends of the cable.

Grounding

Both the RS-232D and RS-422 require that the transmitter and receiver circuits be at the same ground potential (within a few hundred millivolts). On the CCM, none of the circuits are isolated from the Series Six chassis ground, which is also the "local" power supply ground. In many cases this is not a problem. However, the user should insure that the ground voltages are indeed within a few hundred millivolts of each other before connecting the devices together.

A problem will exist only if the local power supply is exceptionally noisy, or if the Series Six PLC rack or other device is floating with respect to this ground (which indicates an incorrect or very unusual configuration). If the user's configuration is such that the grounds do not meet the above conditions, then isolating modems will be required instead of a direct twisted pair hookup.

CAUTION

Communication cables should never be placed in the same trough or in close proximity with power carrying cables. A good rule of thumb is to allow at least one foot separation per 1000 watts (KVA) of power in the carrying cable.

When routing communication cables outdoors, transient suppression devices can be used to reduce the possibility of damage due to lightning or static discharge.

CAUTION

Care should be exercised to ensure that both the CCM module and the device to which it is connected are grounded to a common point. Failure to do so could result in damage to the equipment.

RS-232D Cables

Typical cable wiring for many CCM, RS-232D, applications is shown on the next few pages.

a41524

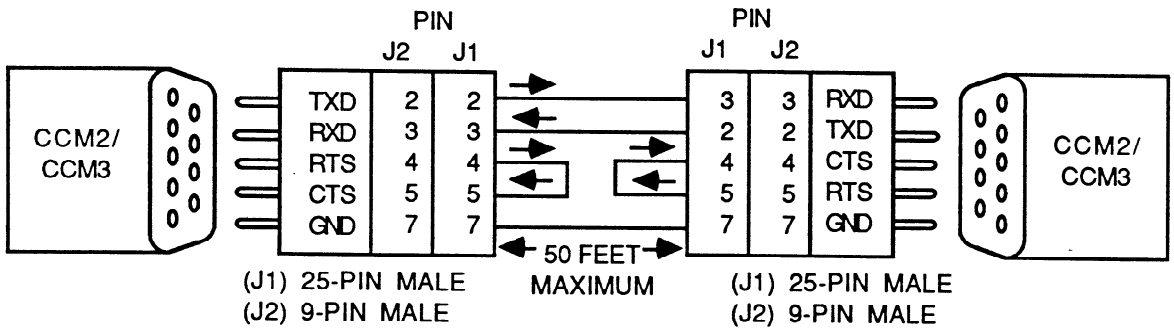


Figure 2.10 RS-232 CCM to CCM CONNECTION (CCM MODE ONLY)

a41525

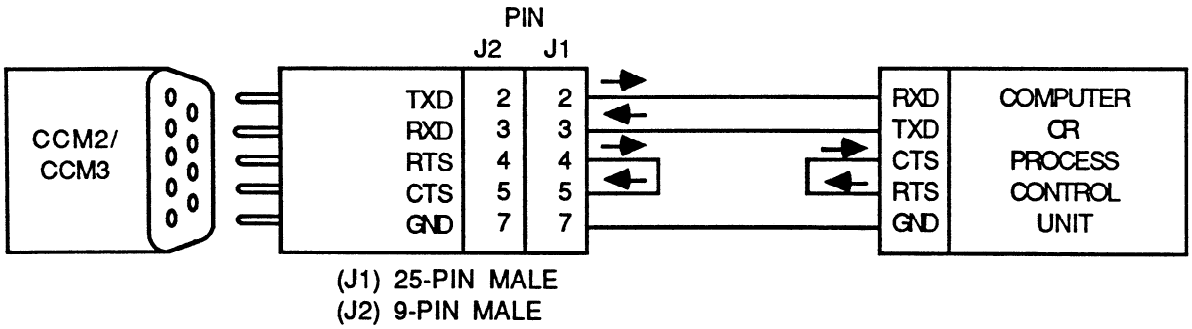


Figure 2.11 RS-232 (CCM OR RTU) TO COMPUTER OR OTHER INTELLIGENT DEVICE

a42637

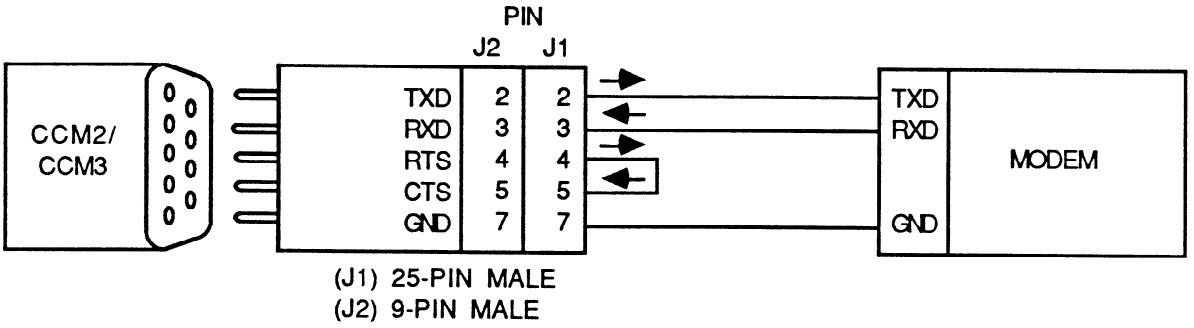


Figure 2.12 RS-232 CCM TO MODEM WITHOUT FLOW CONTROL

a42638

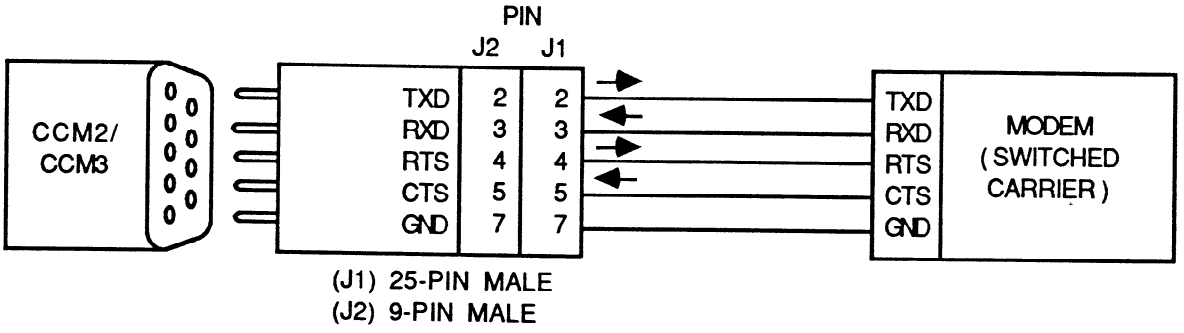


Figure 2.13 RS-232 CCM TO MODEM WITH FLOW CONTROL

a42640

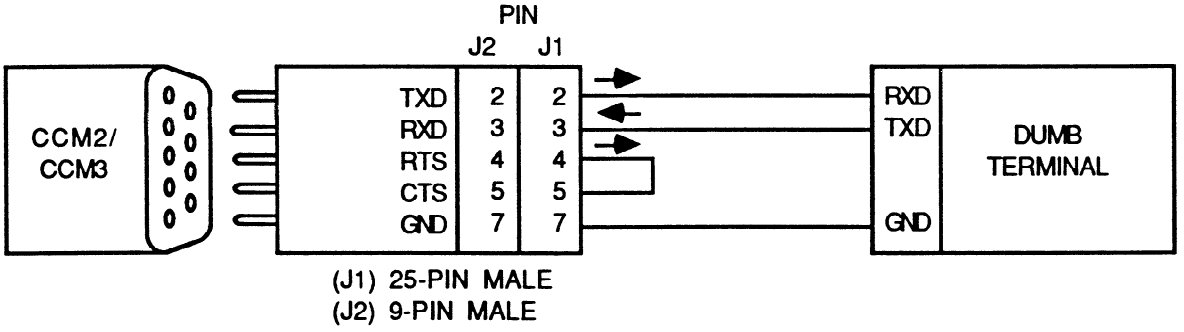


FIGURE 2.14 RS-232 CCM TO DUMB TERMINAL OR PRINTER

GEnet Factory LAN BIU RS-232D Connection

The CCM module may communicate with many other devices by connection to the GEnet LAN Network via the Bus Interface Unit (BIU). The interface connection can be made on J1 or J2 by using the appropriate 9-pin or 25-pin connectors. For detailed network connection information refer to GEK-96608 *GEnet Factory LAN System User's Manual.*

a41199

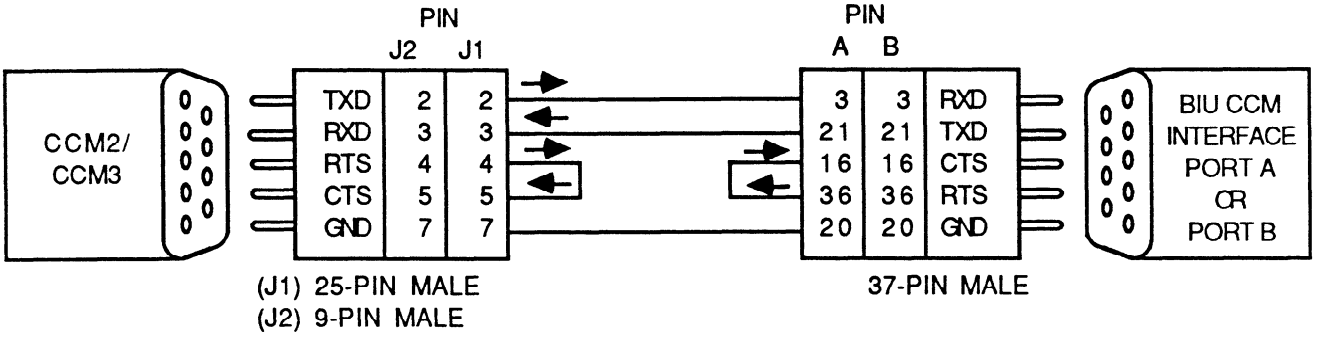


Figure 2.15 RS-232 CCM TO BIU (GEnet)

RS-422 Cables

The RS-422 interface can be used for distances up to 4000 feet (1200 meters) for point-to-point connections. On multidrop links the total length of cable used including all drops cannot exceed 4000 feet.

The RS-422 signal nomenclature used in this manual can be cross referenced to the RS-422 EIA standard as follows:

CCM SIGNAL NAME	RS-422 STANDARD SIGNAL NAME
RS-422 out + (TXD+)	B
RS-422 out - (TXD-)	A
RS-422 in + (RXD+)	B'
RS-422 in - (RXD-)	A'

During a mark condition (logic 1), B will be positive with respect to A. During a space condition (logic 0), B will be negative with respect to A.

When connecting the CCM to a non-Series Six device using the RS-422 standard, the non-Series Six device's line receiver must contain "fail safe" capability. This means that in an idle, open, or shorted line condition, the output of the line receiver chip must assume the "marking" state.

When using RS-422, the twisted pairs should be matched so that both transmit signals make up one twisted pair and both receive signals make up the other twisted pair.

Terminating Resistors

When implementing an RS-422 link, the user must properly include or exclude a 150 Ohm terminating resistor across the receiving circuits for optimum performance of the transmission line.

Devices at both ends of an RS-422 multidrop or point-to-point link should include the terminating resistor. Conversely, any device that is an intermediate drop in a multidrop link should not include the terminating resistor. The appropriate resistors should be removed from the circuit by placing the jumpers in the storage position. (See Table 2.3, CCM Hardware Configuration)

NOTE

Remove the terminating resistors for intermediate CCM modules in the RS-422 multidrop configuration. Refer to Figure 2.6 and to Tables 2.3

RS-232D to RS-422 Adapter Unit

If the host device does not contain RS-422 capability, a RS-232 to RS-422 adapter can be used to complete the interface. The RS-232 Adapter Unit (IC630CCM390) can be purchased from GE Fanuc Automation. The Adapter Unit has two, 25-pin ports. The mating cable connector to these ports must be terminated with a 25-pin, male, D-type connectors.

As many as eight CCMs can be connected in the RS-422 multidrop configuration using the Adapter Unit. The diagram below shows the cable configuration using the RS-232 Adapter Unit (IC630CCM390) which can be purchased from GE Fanuc Automation. Also refer to the documentation which accompanies your Adapter Unit for specific details.

a42697

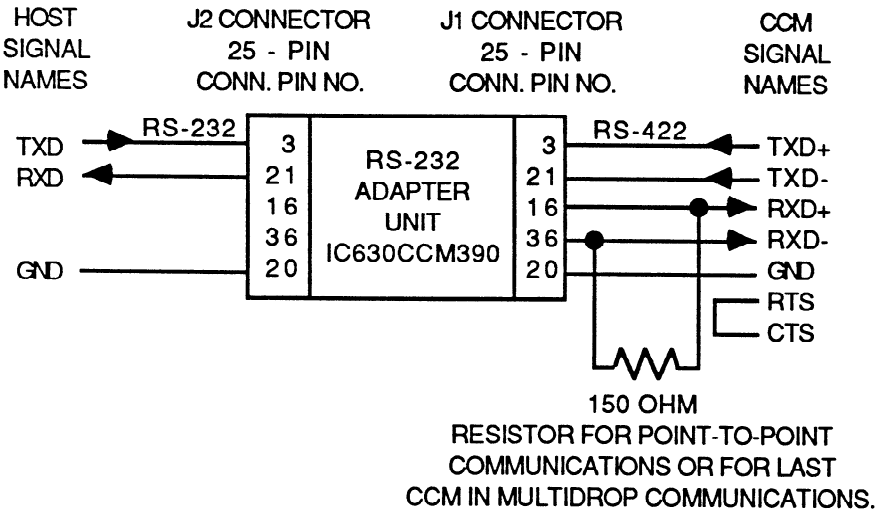


Figure 2.16 RS-232D TO RS-422 ADAPTER UNIT

a42698

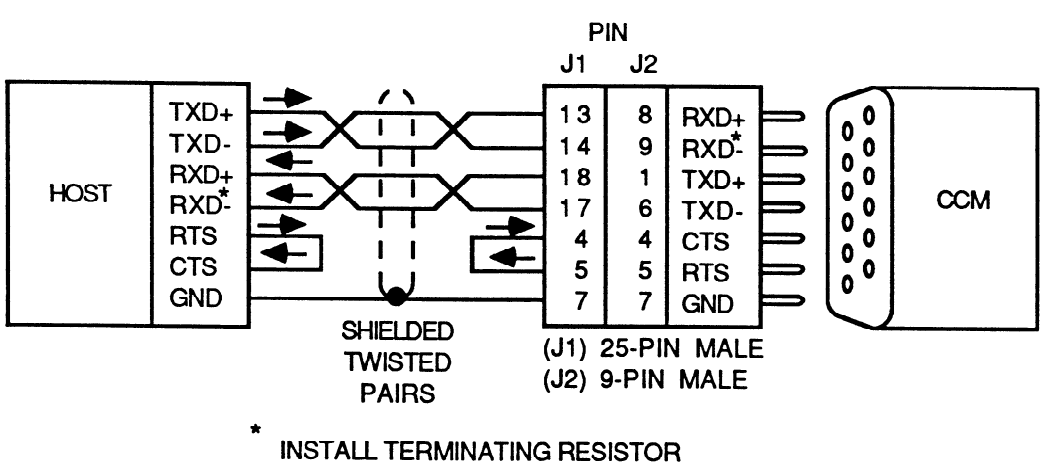


Figure 2.17 RS-422 HOST TO CCM

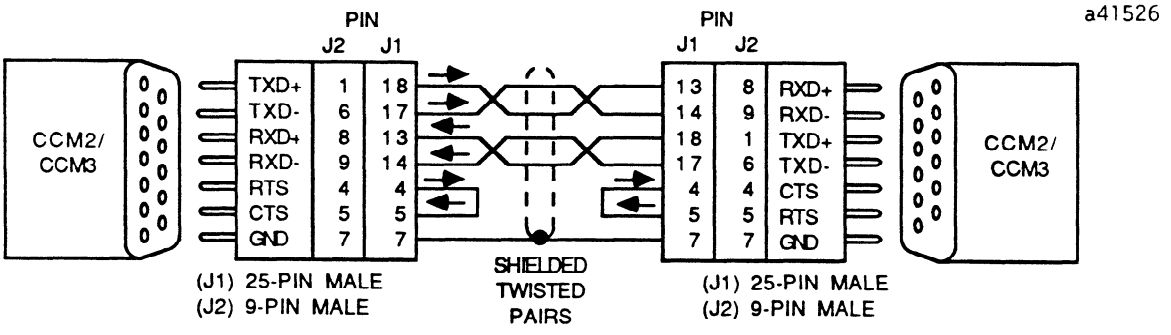


Figure 2.18 RS-422 CCM TO CCM CONNECTION

Operator Interface Unit (OIU) RS-422 Connection

When configured for CCM protocol, the CCM module communicates to the Operator Interface Unit (OIU) via Port J1, RS-422 receive and transmit signals at a data rate of 19.2 KBps, OIU Enabled, and OIU operating power connected. The J1 port also provides power to the OIU. For information concerning the OIU interface, refer to GEK-84866, *Operator Interface Unit Data Sheet.*

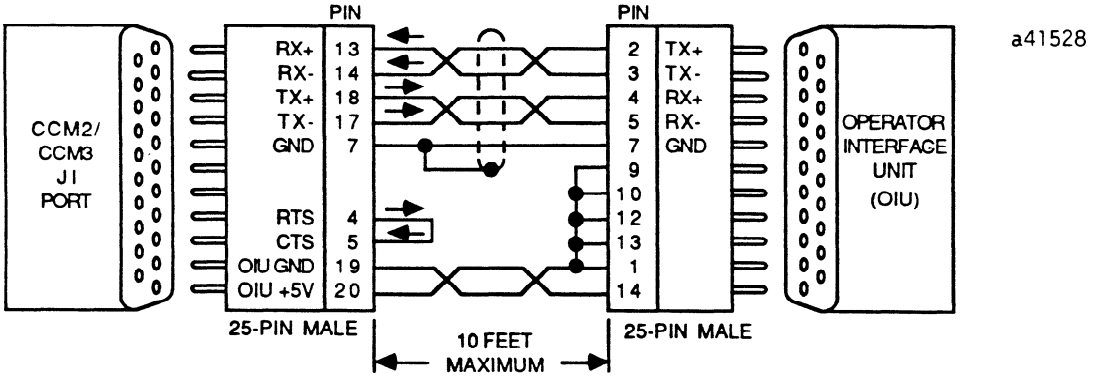


Figure 2.19 RS-422 DIRECT CCM TO OIU CONNECTION

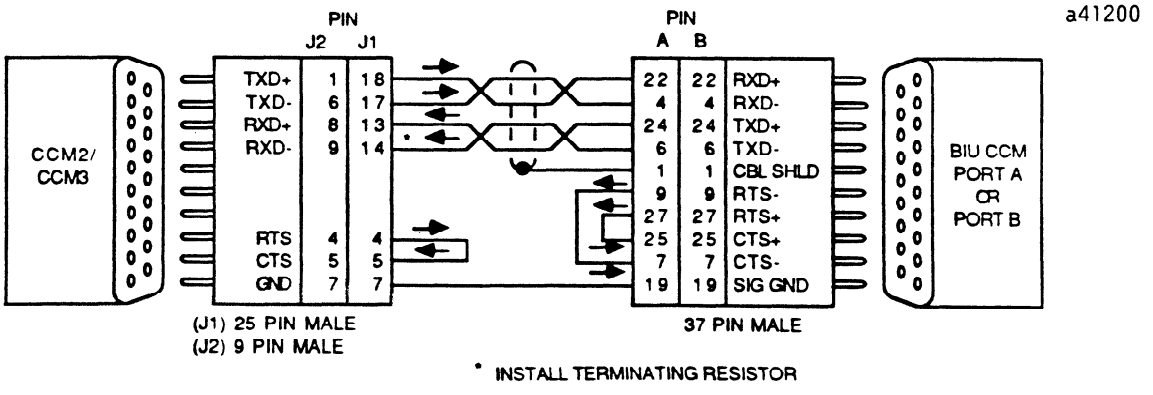


Figure 2.20 RS-422 4-WIRE CONNECTION, CCM TO GEnet BIU

CCM MULTIDROP CONNECTIONS

The diagrams that follow show how devices are normally connected in a multidrop configuration. Several examples are shown of RS-422 multidrop connections including the 4-wire and 2-wire configuration. Master-slave protocol must be used for multidrop connections. Modems of many types may also be used to set up a multidrop configuration.

CCM or Host Computer to Multiple CCMs Using Modems and Radio Transmitters (Multidrop)

The wiring scheme, when using microwave or radio transmitters, depends on the particular modems and transmitters used. Consult your local GE Fanuc Automation application engineer for assistance. (See section, Keying Signal Usage).

CCM or Host Computer to Multiple CCMs Using Modems (Multidrop)

The diagram below shows generally how modems are connected in a multidrop configuration. Due to the variety of modems available, it is not possible to include the unique aspects of each. The user is advised to consult the modem manual for additional information on modem configuration.

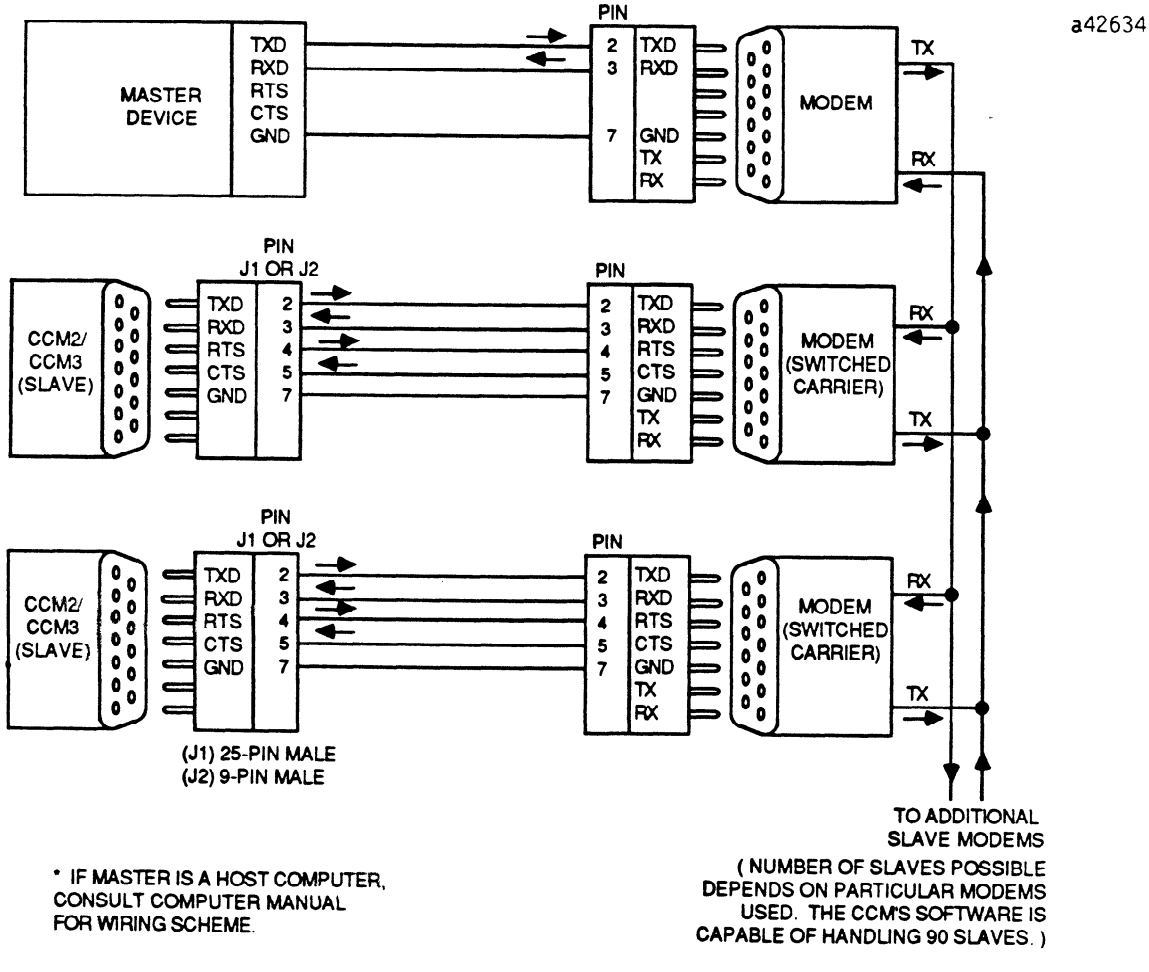


Figure 2.21 RS-232D CCM TO MULTIPLE CCMs USING MODEMS (MULTIDROP)

CCM to Multiple CCMs (4-Wire Multidrop)

a41533

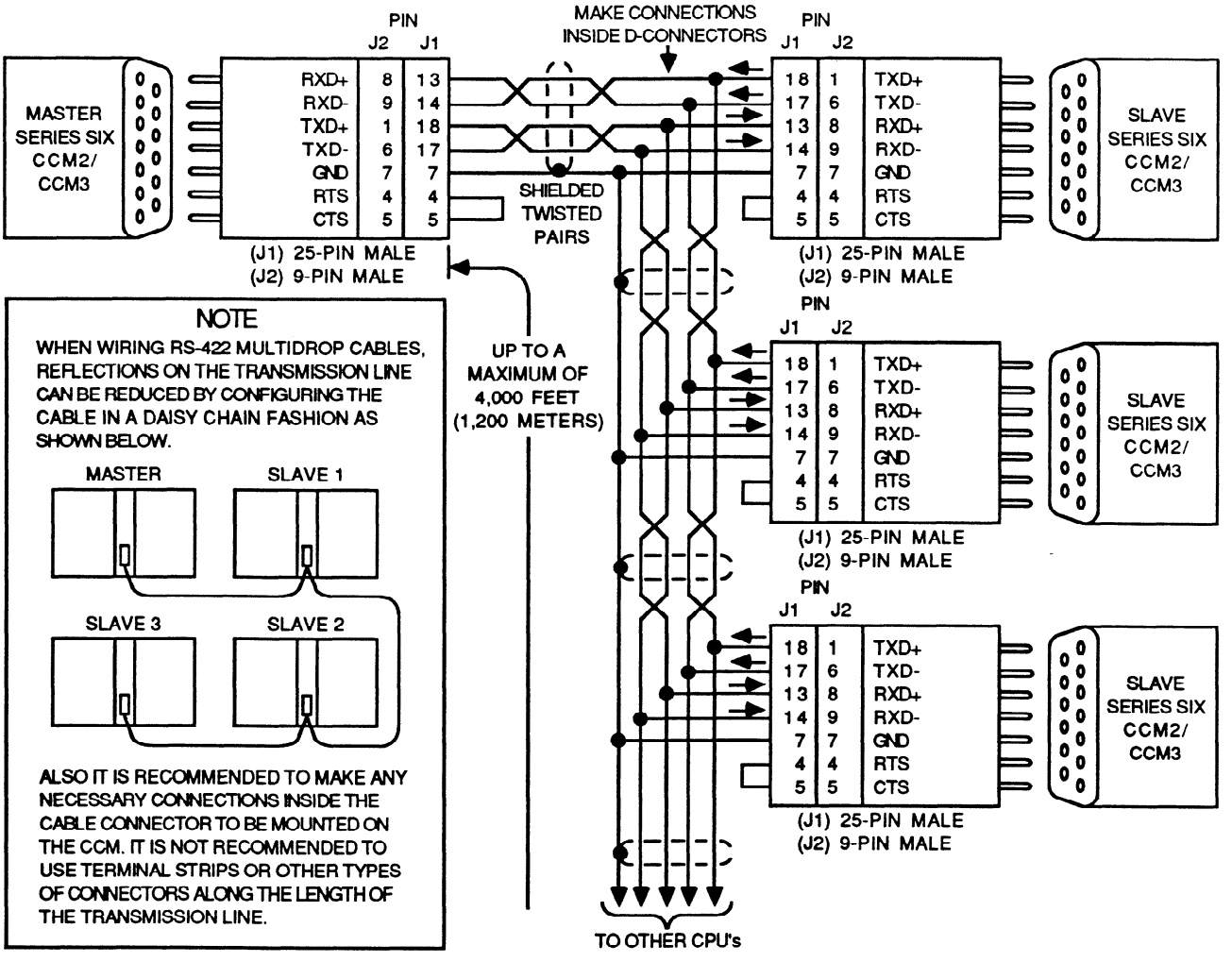


Figure 2.22 RS-422, 4-WIRE MULTIDROP CONNECTION

Host to Multiple CCM3s in RTU Mode (4-Wire Multidrop)

a42696

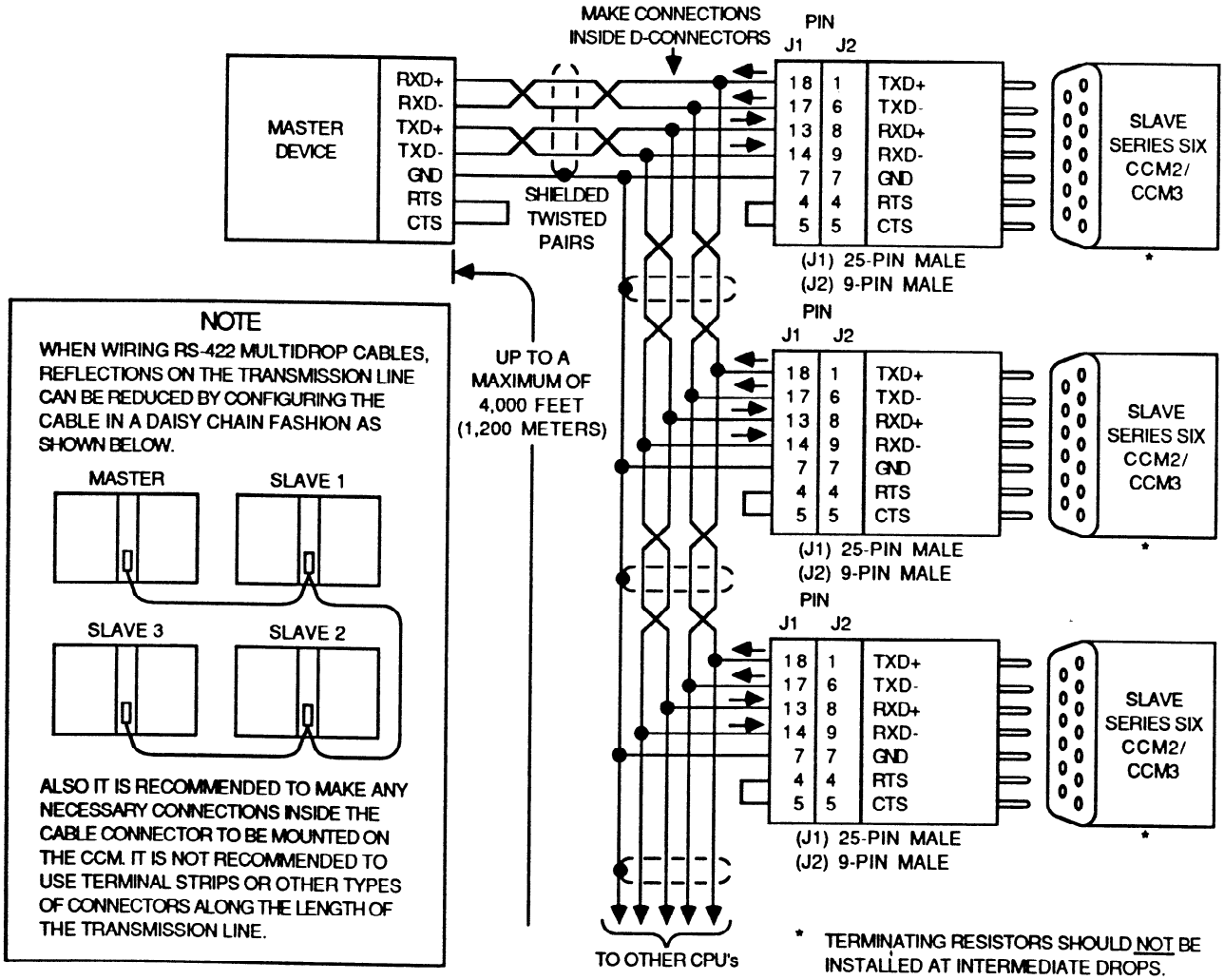


Figure 2.23 RTU RS-422, 4-WIRE MULTIDROP CONNECTION

CCM to Multiple CCMs (2-Wire Multidrop)

A two-wire RS-422 multidrop link can be implemented. To accomplish this, tie signals RXD+ and TDX+ together and tie RDX- and TDX- together at the CCM. This results in one signal path for a 2-wire RS-422 differential signal.

When implementing a 2-wire RS-422 link with a host as a master, the host must have a tri-state transmitter which maintains idle lines in a high impedence state. Also, some host equipment may not allow tying RXD and TDX together. In this case the user must use the 4-wire multidrop configuration.

a41534

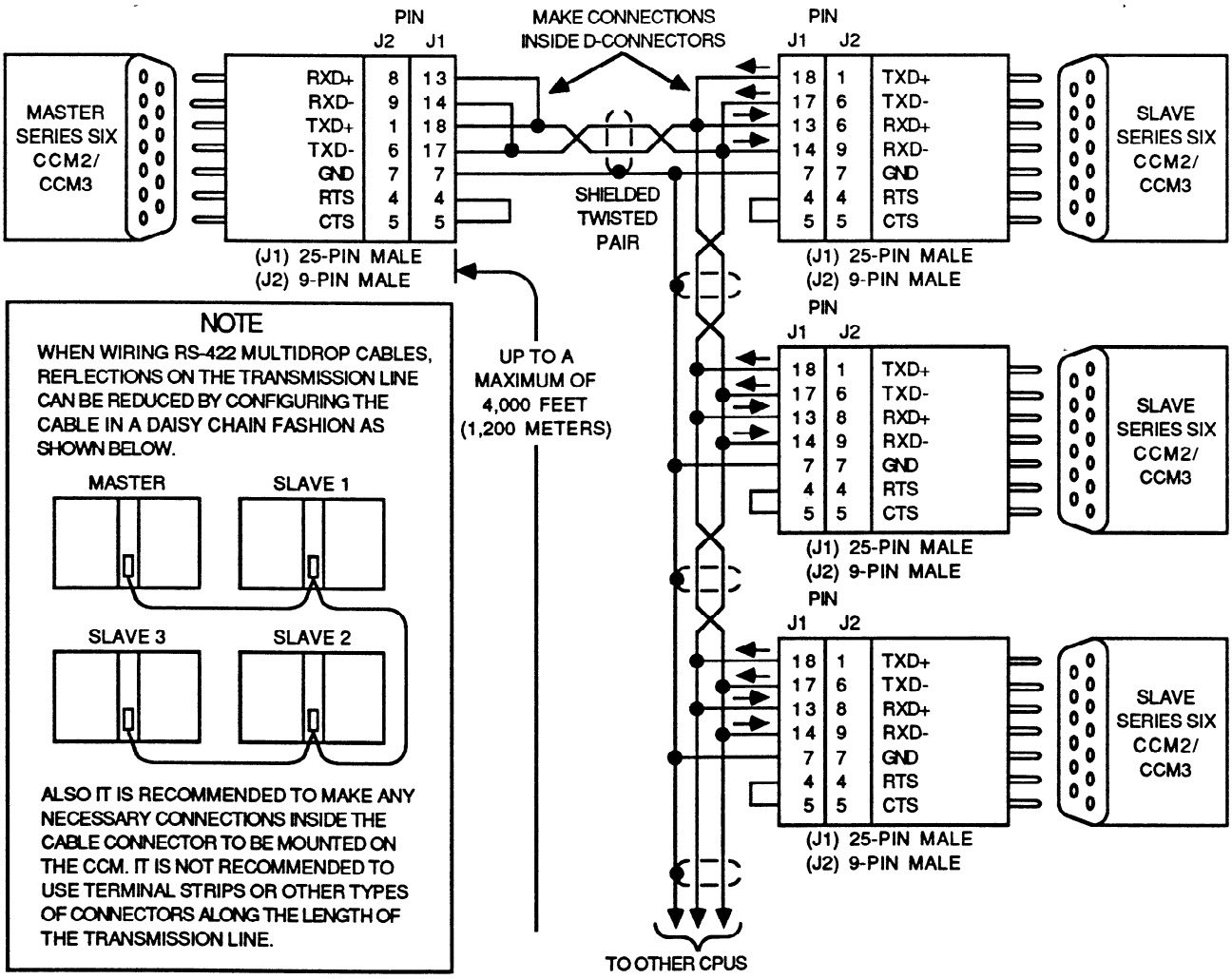


Figure 2.24 RS-422, 2-WIRE MULTIDROP CONNECTION

Host to Multiple CCM3s in RTU Mode (2-Wire Multidrop)

a42695

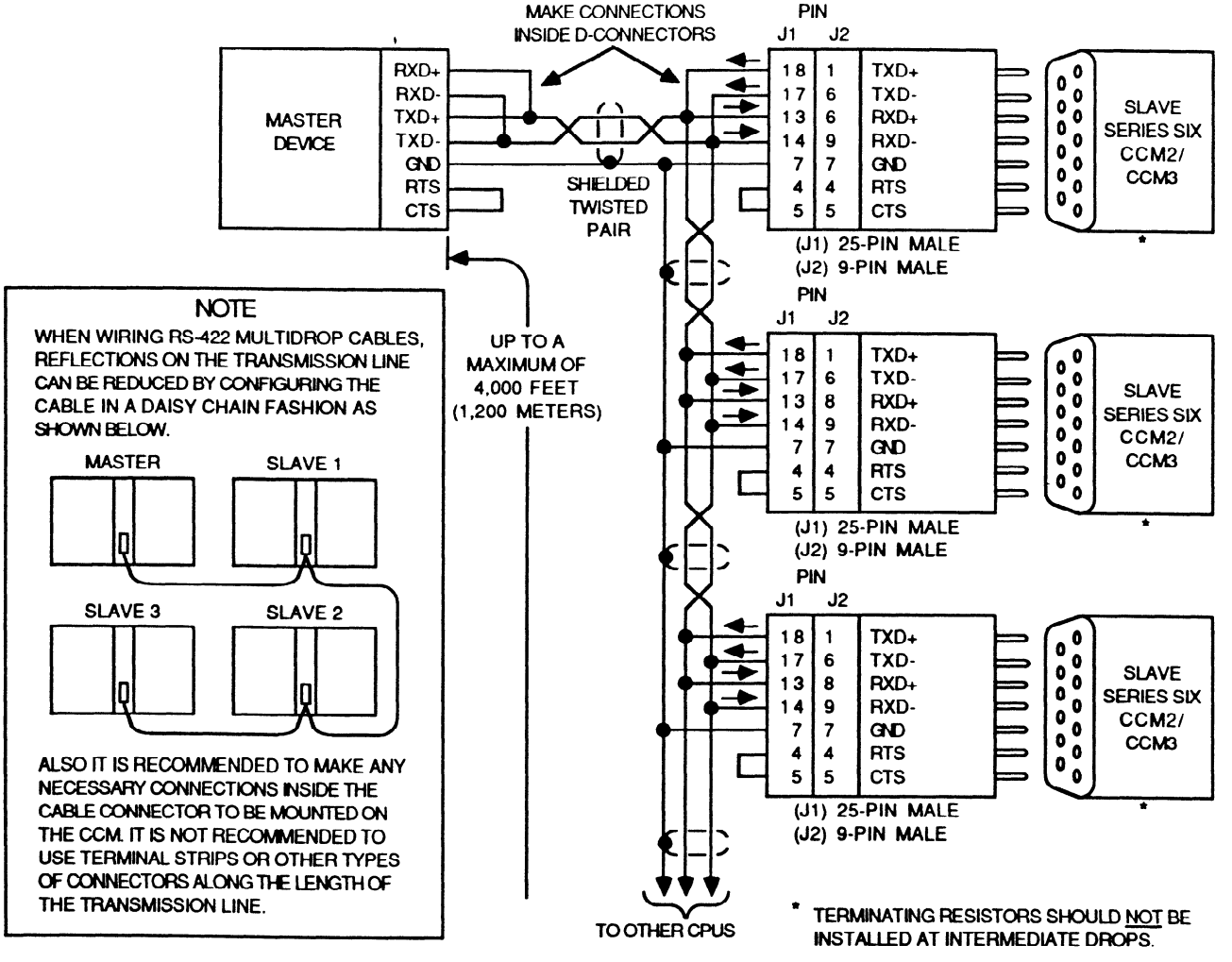


Figure 2.25 RTU, RS-422, 2-WIRE MULTIDROP CONNECTION

KEYING SIGNAL USAGE

Radio transmitters can be used to connect a Series Six PLC with another device when cables between modems are impractical or undesireable. The normal state of the transmitters is OFF and they must be turned ON before data is to be sent. The CCM module keying signal is used to turn transmitters on.

The keying signal allows the radio transmitter to warm up for the length of the turn-around delay before data begins flowing from the CCM. The circuit below shows generally how the keying signal is connected.

84pc0050

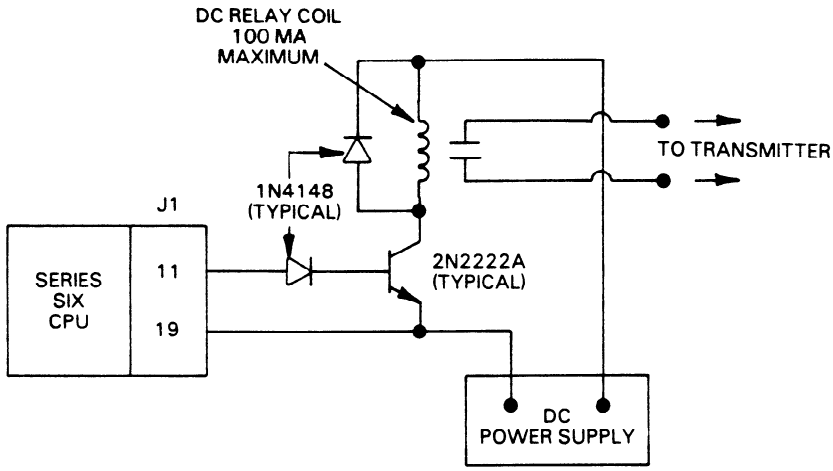


Figure 2.26 RADIO TRANSMITTER KEYING SIGNAL DIAGRAM

A typical radio transmitter has a keying pin which, when pulled to ground, causes the tramsmitter to turn on. One of the relay contact terminals shown in the circuit should be connected to the transmitter ground and the other terminal should be connected to the transmitter keying pin. For specific connection instructions consult the manual for the transmitter.

Grounding

CAUTION

Care should be exercised to ensure that both the CCM and the device to which it is connected are grounded to a common point. Failure to do so could result in damage to the equipment.

TEST DIAGNOSTICS

There are two types of diagnostics available to the user. The first type checks module operation and the second checks the physical interface line.

- Module Diagnostics
- Serial Interface Diagnostics (Test 1 Mode)

Test 1 option is available for the CCM2 module only. The hardware DIP switch settings on CCM3 are used to configure ports J1 and J2 for the RTU mode.

MODULE DIAGNOSTICS

When the CCM is powered-up a diagnostic test sequence is run which verifies whether or not the module is functioning properly. This power-up diagnostic sequence is as follows:

Power-Up Diagnostics

1. A write/read test is performed on all of the CCM RAM.
2. A checksum test is performed on all of the CCM PROM.
3. The 8253 timer chip and 7201 USART are programmed and checked for proper operation.
4. The module configuration is read to verify a valid configuration.
5. A write/read test is performed on the Series Six CPU.
6. A visual test of the indicators is then run to indicate that the previous steps of the test were successful.

If any of the Power-Up Diagnostics (Steps 1-5) above fail, the BOARD OK light turns off and the CCM will not operate. The specific error which occurred can be determined by pressing either of the front panel switches and observing the resulting pattern of the front panel lights (see section, Indicator Lights, Board OK).

Reinitialize Diagnostics

The reinitialize diagnostic occurs once every second when the module is powered-up and idle. The purpose of this diagnostic is to reprogram the timer and USART at regular intervals to prevent against accidental programming during a power glitch.

SERIAL INTERFACE DIAGNOSTICS (Test 1)

When the CCM2 is configured for Test 1 mode, the CCM2 will echo any characters that are received in either port. This test corresponds to the BERT test (Bit Error Rate Test).

This test checks the physical line connected to the CCM2 without requiring a Series Six user program to intitiate a data transfer. The user must supply a character generator such as a communications analyzer to send characters to the CCM2 and then observe the echo back from the CCM2.

When in this mode, the data rate and serial interface of both ports are determined by the J1 port switches (9-16).

CPU/CCM COMMUNICATIONS

Communications between the CPU and CCM do not occur continuously. The CPU must perform many tasks only one of which is CCM communications. The CPU performs these tasks, sequentially, in a CPU scan.

CPU SCAN

One CPU scan consists of the following:

- Housekeeping (part of the executive routine)
- Handling device communications "windows" for the programming device (Workmaster or PDT), and Communications Control Module (CCM)
- Solving the user logic program
- Updating the I/O based on that solution of the ladder logic program

The maximum CPU scan time is 200 msec (±50 msec) prior to version 130 microcode, and 300 msec (±22 msec) for microcode version 130 and later. The scan time is monitored by a hardware timer in the CPU, which is the Watchdog Timer. Figure 2.27 illustrates the scanning sequence.

a42635

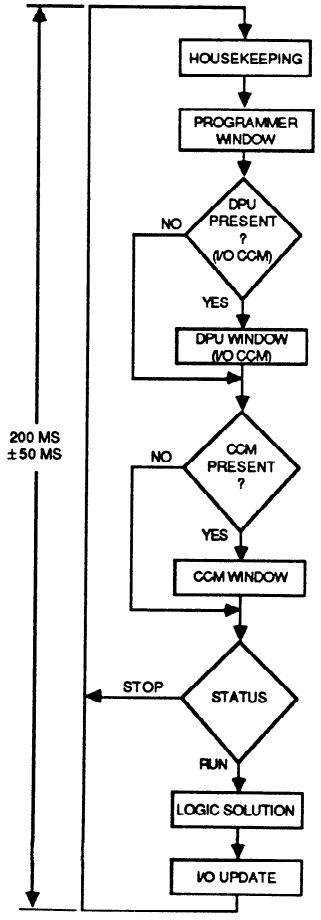


Figure 2.27 CPU SCAN

With the CPU in the RUN mode, the entire scanning sequence is repeated continually. Note that in the STOP mode, the logic solution and I/O update segments of the scan are omitted from the sequence.

The scan time is not fixed; its length depends on the content of the user program and whether or not the PDT, Workmaster, DPU, or CCM are connected and communicating with the CPU. Table 2.11 shows the time required by the CPU for execution of each element in its scan. Note the timing of the CCM window with data and with no data.

Table 2.11 CPU SCAN TIME

SCAN SEQUENCE	DEVICE PRESENT DATA	DEVICE PRESENT NO DATA	DEVICE NOT PRESENT
Housekeeping	0.1 millisec	N/A	N/A
PDT Executive Window	10 millisec (max)	0.56 millisec	0
DPU Executive Window	11 millisec	4 millisec	0
CCM Executive Window	24 millisec (max)	0	0
User Logic Solution	1 ms/k words (Basic) or 2.5 ms/k words (Extended) and -] [-[SCREQ]1	N/A	N/A
I/O Update	5.8 millisec (max)	N/A	N/A

1 If a Serial Communications REQuest is initiated in the user logic, a SCREQ window, with the same maximum length as the CCM window will open.

CCM COMMUNICATIONS WINDOWS

A window describes the mechanism by which the CPU communicates with the CCM. When a window is "open," communication can take place. In CPU to CCM communications there are two types of windows: the CCM executive window and the [SCREQ] window.

These two types of windows function essentially the same. The main difference between them is how and when they are initiated. The CCM executive window is executed automatically once per CPU scan as long as the CCM is present and not busy and the windows are enabled. Figure 2.27 shows the position in the scan that this window is executed. Table 2.11 shows the maximum time the CCM executive window can be open.

The [SCREQ] window, on the other hand, is executed by triggering the [SCREQ] in the user program. This window occurs only once for each [SCREQ] function activation. As before; the CCM must be present and not busy, and the windows enabled. The maximum length of this window is the same as the CCM executive window.

During the CCM communications windows, the CCM reads the CPU scratchpad for a pointer to the information necessary to execute a command. This information is passed from CPU to CCM through the 6 [SCREQ] function registers to be discussed in the next section. Actual data is also transferred from CPU to CCM or CCM to CPU during these windows. At the end of each window, the CCM status byte is passed to the CPU and the window is closed.

When a [SCREQ] function is initiated, the CPU updates the scratch pad containing the pointer to the 6 [SCREQ] registers. If the CCM is not busy with another request, the SCREQ window is opened and the pointer to the [SCREQ] registers is read. If the CCM is busy when a [SCREQ] function is initiated, the [SCREQ] window will not be executed. The pointer is written to the scratchpad by the CPU, but will not be read nor the command executed until the first executive window after the CCM becomes idle. RTU protocol communication cannot be initiated from the Series Six CPU via the SCREQ. Only a CCM configured as peer or master can initiate data transfer via a [SCREQ].

CPU [STATUS] FUNCTION

Upon power up of the Series Six CPU, the windows are enabled. If window control is desired, the user can enable or disable the CCM windows by programming the [STATUS] function using the programmer device (Workmaster or PDT). When the windows are enabled, they can be opened or closed at the normal time in the CPU scan (executive window) or by the execution of the [SCREQ] function ([SCREQ] window). When the windows are disabled, however, they cannot be opened at any time and will not be executed.

The CPU [STATUS] function operates by using the contents of bit 5 of its reference register to indicate DPU status and the contents of bit 6 to indicate CCM status.

Constant values can be entered in the CPU [STATUS] function to enable or disable one or both windows. The table below shows how the windows are controlled by using the [STATUS] function.

Table 2.12 CPU [STATUS] FUNCTION OPERATION

REFERENCE REGISTER CONTENTS				CCM WINDOWS	I/O CCM or DPU WINDOWS
BINARY		DEC	HEX		
BIT 6	BIT 5				
0	0	00000	0000	Enabled	Enabled
0	1	00016	0010	Enabled	Disabled
1	0	00032	0020	Disabled	Enabled
1	1	00048	0030	Disabled	Disabled

The CPU [STATUS] function is entered in a line of logic as shown in example below. For clarity only one permissive contact is shown in the example. The reference register can be any register from R0001 to R1024. It is recommended, however, that the CPU [STATUS] reference is R0006 when the DPU is used in conjunction with the CCM because registers R0001-R0005 are normally reserved for DPU system registers. Whether using the DPU or not it is convenient to use a register from R0001 to R0128 since the contents of these registers are reflected in auxiliary inputs or outputs and the individual bits or points can be easily monitored and used as interlocks.

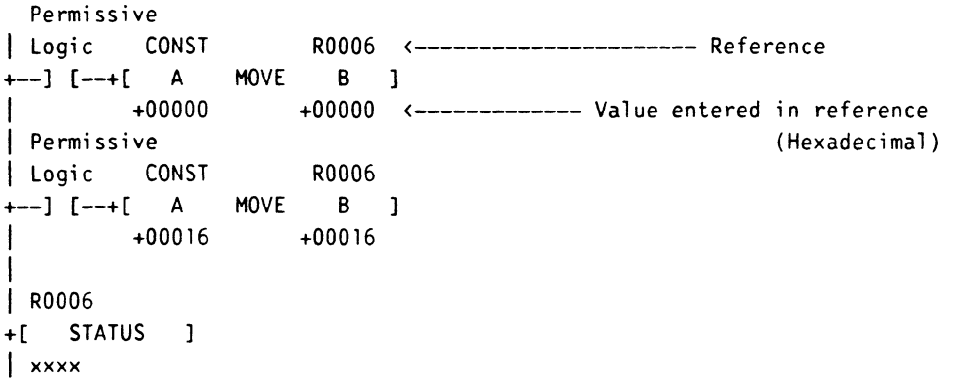


Figure 2.28 [STATUS] FUNCTION FORMAT

If there is power flow in one of the rungs containing the [A MOVE B] function, the constant specified by reference A will be copied into the CPU [STATUS] reference R0006. Since the [STATUS] function has power flow to it, the 5th and 6th bits of that constant will be copied into the CPU scratch pad memory locations and the function will control DPU window status and CCM window status respectively (disabled=1).

If the [STATUS] function were programmed with a permissive contact preventing power flow to it, the DPU window status and CCM window status would be copied from the CPU scratch pad to the 5th and 6th bits of the CPU [STATUS] reference and the function would not control the window status.

In either case, if R0006 is used as the STATUS reference, then the on/off states of auxiliary outputs AO0085 and AO0086 will reflect DPU window status and CCM window status respectively. Contacts assigned to these auxiliary outputs can be used as interlocks for [DPREQ] and [SCREQ] functions.

Refer to later sections of this chapter, CCM and RTU Status Byte Definition, for more information.

CPU/CCM PROGRAMMING

SCREQ commands are used to issue communication requests to the CCM module. This section discusses specific [SCREQ] commands and the CPU programming required to initiate them.

Three [SCREQ] port commands can be used with the RTU mode of operation. These commands are:

- Read Character String to Source Register Table
- Write Character String from Source Register Table
- Write then Read Immediate Character String

All other [SCREQ] commands described in this manual pertain to the CCM mode of operation only.

CCM [SCREQ] COMMAND USES AND CATEGORIES

The main characteristics of the [SCREQ] command categories are given below. For details of each type of [SCREQ] command, refer to the section, [SCREQ] Command Programming Examples.

Internal Commands

The internal [SCREQ] commands are numbered from 06000 to 06012. These commands provide the means for a CPU to access its resident: CCM Quick Access Buffer (QAB), Diagnostic Status Words, software memory protect function, and OIU timer and counter configuration function.

Port Commands

There is an identical set of commands for both the J1 and J2 ports. J1 port commands are numbered 06100-06128: J2 port commands are numbered 06200-06228. Four basic types of data transfer commands can be implemented through the ports.

CPU to CPU Transfer

In this transfer, information is passed from CPU memory in one Series Six to CPU memory in another Series Six, Series One, or Series Three PLC. The commands used to implement this transfer include command numbers (06101-06106, 06201-06206; and 06111-06117, 06211-06217) and take the general form of:

```
Read from Target  (CPU Memory Type)  to Source    (CPU Memory Type)

                          or

Write to Target   (CPU Memory Type)  from Source  (CPU Memory Type)
```

CCM to Remote CPU Transfer

The QAB is a 1024 byte buffer resident on the CCM module; the Diagnostic Status Words are also resident on the CCM and are used for communications error diagnostics. The CCM to remote CPU transfer enables data to be transferred in both directions between the CCM and an external CPU. The commands used to implement these transfers include command numbers (06101-06106; 06201-06206 and 06111-06117; 06211-06217) and take the form of:

```
Read from Target <          QAB              > to Source    (CPU Memory Type)
                            or
Write to Target  < Diagnostic Status Words > from Source  (CPU Memory Type)
```

These transfers are faster than the CPU to CPU transfer because they operate with the CCM directly and do not have to wait for data to be transferred from the CPU to the CCM. The QAB transfers operate in conjunction with internal commands, 06004-06009, for loading and reading the QAB of the resident CCM.

Q Response Transfer

This is the fastest type of data transfer from one Series Six to another; it requires the CCM master-slave protocol and transfers four 8-bit bytes of data at a time. An abbreviated protocol sequence and the small amount of data capable of being transmitted accounts for the speed of this transfer type. Command 06109, Read Q Response, is used to initiate the transfer. This command operates in conjunction with internal command, 06001, which loads new data for the next Q response.

Character String Transfer (Unformatted Data Transfer)

This transfer type allows any ASCII character to be written out to a printer or dumb terminal and for characters to be directly inputted from a dumb terminal. These characters are transmitted verbatim, that is, not within the peer-to-peer or master-slave protocol format. The commands used to implement this type of transfer are Read Character String, 06108, 06208; Write Character String, 06118, 06218; and Write then Read Immediate Character String, 06128, 06228.

[SCREQ] FUNCTION ACTIVATION

The Serial Communication REQuest [SCREQ] function initiates internal transfers as well as transfers of data from one Series Six to another. When the function is activated, the contents of 6 [SCREQ] registers, containing information required to execute the data transfer, are read from the CPU by the CCM. The user supplies the contents of the [SCREQ] registers based on the requirements of the transfer command. The figure below shows a simple ladder logic format which can be used to execute the [SCREQ] function.

```
                        < Rung No. 1 >
|
|   I0001     C       R0006
+---] [---+[ A  MOVE  B ]-                                          (  )
|         0
|
|                        < Rung No. 2 >
|
|   I0002     C       R0006
+---] [---+[ A  MOVE  B ]-                                          (  )
|          16
|
|                        < Rung No. 3 >
|
|     R0006
+---[ STATUS ]-                                                     (  )
|     0 or 16
|
|                        < Rung No. 4 >
|
|   I0003       R0100                                                 00001
+---] [----+--[                  BLOCK MOVE                        ]+----(OS)
|            xxxx  nnnn  nnnn   nnnn   nnnn   nnnn   nnnn  0000
|                  (Rn) (Rn+1) (Rn+2) (Rn+3) (Rn+4) (Rn+5)
|
|                        < Rung No. 5 >
|
|   00001   I1009     R0100
+---] [--+--]/[--+--[ SCREQ ]-                                      (  )
|
```

Figure 2.29 SIMPLIFIED [SCREQ] FUNCTION ACTIVATION

Rungs 1, 2, and 3 control the CCM windows.

Rung No. 4 contains a [BLOCK MOVE] function which is the easiest way to load the 6 [SCREQ] registers required by the [SCREQ] function. For this to work correctly the [BLOCK MOVE] reference register must be the same as the [SCREQ] reference register. When the [BLOCK MOVE] is activated by I0003, 7 constants are copied into 7 sequential locations starting at the specified reference (R0100 in this case) and power flow is outputted.

Since the CCM is looking only for 6 registers, the 7th constant in the [BLOCK MOVE] is ignored. The specific values of the [BLOCK MOVE] constants will be discussed shortly. The primary purpose of the [BLOCK MOVE] is to ensure that the proper information is loaded into the 6 [SCREQ] registers before the [SCREQ] function is activated. Upon execution of the [BLOCK MOVE] there is power flow and O0001 fires.

Rung No. 5 contains the [SCREQ] function and it is triggered by O0001. The normally closed contact, I1009, is used as an interlock. (See section, CCM Communication Request Status and Diagnostic Information). This input is automatically updated by the CCM and indicates whether the CCM is busy (no power flow) or idle (power flow). When there is power flow, the CCM is idle and the [SCREQ] function can be triggered by O0001; when there is no power flow, the [SCREQ] function cannot be executed. Upon activation, the [SCREQ] function sets up the information (in registers R0100-R0105 in this case) for the CCM to read from the CPU.

[SCREQ] REGISTER ASSIGNMENTS

Each of the 6 [SCREQ] registers are assigned a specific type of information about the serial communications request. The values in these registers must be kept constant until the CCM sets the CCM Busy Bit on (I1009), for one scan, to ensure proper operation of the [SCREQ] function. The register assignments are defined as follows:

Rn	Command Number
Rn + 1	Target[1] ID
Rn + 2	Target Memory Type
Rn + 3	Target Memory Address
Rn + 4	Data Length
Rn + 5	Source[2] Memory Address

[1] The target is the device which does not initiate the serial communications request.
[2] The source is the device which does initiate the serial communiations request.

The contents of the 6 [SCREQ] registers are stored in the [BLOCK MOVE] function in the order shown below.

```
  Rn
[                                  BLOCK MOVE                                  ]
  xxxx    Command  Target    Target    Target    Data      Source    0000
          Number     ID      Memory    Memory    Length    Memory
                             Type      Address             Address
           (Rn)   (Rn + 1)  (Rn + 2)  (Rn + 3)  (Rn + 4)  (Rn + 5)
```

A complete explanation of each [SCREQ] register is given on the following pages. The [SCREQ] registers of some commands do not follow the assignments exactly as shown above. Refer to the programming example for a particular command in the section, [SCREQ] Command Programming Examples, for exceptions to register assignments.

Rn: Command Number (Range: See Table 2.13)

The command number determines the target memory type, direction of data transfer, and whether the transfer is internal (between CCM and CPU) or external (through J1 and J2 ports). The following table lists all the CCM commands; it also shows which [SCREQ] registers are required for a particular command.

Table 2.13 [SCREQ] COMMANDS

COMMAND DEFINITION	[SCREQ] REGISTERS - Not Used X Required					
	Command Number Rn	Target ID Rn + 1	Target Memory Type Rn + 2	Target Memory Address Rn + 3	Data Length Rn + 4	Source Memory Address Rn + 5
Internal Commands						
NOOP	06000 (1770)*	-	-	-	-	-
Set Q Response	06001 (1771)	X	X	-	-	-
Clear CCM Diagnostic Status Words	06002 (1772)	-	-	-	-	-
Read CCM Diagnostic Status Words To Source Registers	06003 (1773)	-	-	X	X	X
Load CCM QAB From Source Register Table	06004 (1774)	-	-	X	X	X
Load CCM QAB From Source Input Table	06005 (1775)	-	-	X	X	X
Load CCM QAB From Source Output Table	06006 (1776)	-	-	X	X	X
Read CCM QAB To Source Register Table	06007 (1777)	-	-	X	X	X
Read CCM QAB To Source Input Table	06008 (1778)	-	-	X	X	X
Read CCM QAB To Source Output Table	06009 (1779)	-	-	X	X	X
Set CPU Memory Write Protect	06010 (177A)	-	X	X	X	-
Reinitialize CCM Timer and USART	06011 (177B)	-	-	-	-	-
Set OIU Timers and Counters	06012 (177C)	-	X	X	X	X

* Numbers in parenthesis are in hexadecimal

Table 2.13 [SCREQ] COMMANDS (continued)

COMMAND DEFINITION	[SCREQ] REGISTERS - Not Used X Required					
	Command Number Rn	Target ID Rn + 1	Target Memory Type Rn + 2	Target Memory Address Rn + 3	Data Length Rn + 4	Source Memory Address Rn + 5
J1 Port Commands						
NOOP	06100 (17D4)*	-	-	-	-	-
Read From Target To Source Register Table	06101 (17D5)	X	X	X	X	X
Read From Target To Source Input Table	06102 (17D6)	X	X	X	X	X
Read From Target To Source Output Table	06103 (17D7)	X	X	X	X	X
Read From Target To Source Input Override Table	06104 (17D8)	X	X	X	X	X
Read From Target To Source Output Override Table	06105 (17D9)	X	X	X	X	X
Read From Target To Source QAB	06106 (17DA)	X	X	X	X	X
Unused	06107					
Read Char. String To Source Register Table (Unformatted Read)	06108 (17DC)	-	-	X	X	X
Read Q Response To Source Register Table	06109 (17DD)	X	-	-	-	X
Single Bit Write	06110 (17DE)	X	X	X	-	-
Write To Target From Source Register Table	06111 (17DF)	X	X	X	X	X
Write To Target From Source Input Table	06112 (17E0)	X	X	X	X	X
Write To Target From Source Output Table	06113 (17E1)	X	X	X	X	X
Write To Target From Source Input Override Table	06114 (17E2)	X	X	X	X	X
Write To Target From Source Output Override Table	06115 (17E3)	X	X	X	X	X
Write To Target From Source QAB	06116 (17E4)	X	X	X	X	X
Write To Target From Source User Logic Memory	06117 (17E5)	X	X	X	X	X
Write Char. String From Source Register Table (Unformatted Write)	06118 (17E6)	-	-	-	X	X
Write Then Read Immediate Char. String From Source (Unformatted Write then Read)	06128 (17F0)	X	X	X	X	X
Set CCM Retries	06130 (17F2)	-	X	X	X	X
Set CCM Timeouts	06131 (17F3)	X	X	X	X	X

* Numbers in parenthesis are in hexadecimal

Table 2.13 [SCREQ] COMMANDS (continued)

COMMAND DEFINITION	[SCREQ] REGISTERS - Not Used X Required					
	Command Number Rn	Target ID Rn + 1	Target Memory Type Rn + 2	Target Memory Address Rn + 3	Data Length Rn + 4	Source Memory Address Rn + 5
J2 Port Commands						
NOOP	06200 (1838)*	-	-	-	-	-
Read From Target To Source Register Table	06201 (1839)	X	X	X	X	X
Read From Target To Source Input Table	06202 (183A)	X	X	X	X	X
Read From Target To Source Output Table	06203 (183B)	X	X	X	X	X
Read From Target To Source Input Override Table	06204 (183C)	X	X	X	X	X
Read From Target To Source Output Override Table	06205 (183D)	X	X	X	X	X
Read From Target To Source QAB	06206 (183E)	X	X	X	X	X
Unused	06207					
Read Char. String To Source Register Table (Unformatted Read)	06208 (1840)	-	-	X	X	X
Read Q Response To Source Register Table	06209 (1841)	X	-	-	-	X
Single Bit Write	06210 (1842)	X	X	X	-	-
Write To Target From Source Register Table	06211 (1843)	X	X	X	X	X
Write To Target From Source Input Table	06212 (1844)	X	X	X	X	X
Write To Target From Source Output Table	06213 (1845)	X	X	X	X	X
Write To Target From Source Input Override Table	06214 (1846)	X	X	X	X	X
Write To Target From Source Output Override Table	06215 (1847)	X	X	X	X	X
Write To Target From Source QAB	06216 (1848)	X	X	X	X	X
Write To Target From Source User Logic Memory	06217 (1849)	X	X	X	X	X
Write Char. String From Source Register Table (Unformatted Write)	06218 (184A)	-	-	-	X	X
Write Then Read Immediate Char. String From Source (Unformatted Write then Read)	06228 (1854)	X	X	X	X	X
Set CCM Retries	06230 (1856)	-	X	X	X	X
Set CCM Timeouts	06231 (1857)	X	X	X	X	X

* Numbers in parenthesis are hexadecimal

As can be seen, some commands do not use all 6 [SCREQ] registers. For clarity in programming, it is recommended that the constants in the [BLOCK MOVE] function associated with the unusued [SCREQ] registers should be set to 0.

Rn + 1: Target ID (Range: Peer-to-Peer, 1-255; Master-Slave, 1-90)

To execute a transfer of data between Series Six CPUs, one Series Six must request the transfer and the other must comply with the request. The device requesting or initiating the transfer is the source; the device complying with but not initiating the request is the target. Data may flow from source to target as well as from target to source.

The Target ID is the identification number of the target device; for a Series Six CPU, it is the CPU ID number. The user can assign this number with the programmer (e.g., Workmaster or PDT) via the Scratch Pad display. The value of the Target ID number can range from 1 to 255 when in peer-to-peer mode and from 1 to 90 in the master-slave mode. Target ID 0 is reserved. Any peer CPU regardless of its ID will respond to Target ID 255.

Rn + 2: Target Memory Type (Range: 0-11, 13-22)

This is the type of memory being accessed in the target device. There are 22 target types accessible to the user. The memory type associated with each number is shown below.

Number	Type Definition
0*	Absolute
1	Register Table
2	Input Table
3	Output Table
4*	Input Override Table
5*	Output Override Table
6*	CPU Scratch Pad Memory
7*	User Logic Memory
8	CCM Quick Access Buffer
9	CCM Diagnostic Status Word
10	Timers (Only valid for command 06012)
11	Counters - (Only valid for command 06012)
13	Bit Set Input Table
14	Bit Set Output Table
15	Bit Set Input Override Table
16	Bit Set Output Override Table
17	Bit Clear Input Table
18	Bit Clear Output Table
19	Bit Clear Input Override Table
20	Bit Clear Output Override Table
21	Bit Toggle Input Table
22	Bit Toggle Output Table

* Memory types 4, 5, 6 and 7 are protected by the CPU memory switch. Memory Type 0, Absolute Memory, includes types 4, 5, 6, 7 and these parts of Absolute Memory are also protected by the CPU memory switch. This switch must be in the WRITE position for writing into these memories.

Rn + 3: Target Memory Address (Range: See Table 2.14.)

The Target Address specifies the address within the target device where the transfer is to begin. The address ranges for each memory type are given in the following table.

Table 2.14 TARGET/SOURCE MEMORY ADDRESSES

Target/Source Memory Type	Description	Target/Source Address Range		
		Specific Mem. Type		Range*
0 - Absolute	Specifies the absolute memory address where the data transfer is to begin.	Transition Table	Outputs	01024-01151 (0400-047F)
			Inputs	01152-01279 (0480-04FF)
		Override Table	Outputs	02048-02175 (0800-087F)
			Inputs	02176-02303 (0880-08FF)
		Scratch Pad Mem.		04096-04351 (1000-10FF)
		Status Table	Outputs	08192-08319 (2000-207F)
			Inputs	08320-08447 (2080-20FF)
		Register Memory		16384-32767 **(4000-7FFF)
		User Logic Memory		32768-65535 **(8000-FFFF)
1 - Register Table	Specifies the register where the data transfer is to begin.	Register Memory Size		**
		256		1-256
		1K		1-1024
		8K		1-8192
		16K		1-16384
2 - Input Table	Specifies the Input or Output point where the data transfer is to begin. The number must begin on a byte boundary (i.e., 1, 9, 17. ...)			1-1024
3 - Output Table			***	1-32768
4 - Input Override Table				1-1024
			***	8193-9216
5 - Output Override Table				1-1024
			***	8193-9216
Aux I/O Override				
6 - CPU Scratch Pad Memory	Specifies the CPU Scratch Pad byte at which the data transfer is to begin.			00000-00255 (0000-00FF)

* Ranges without parenthesis are in decimal notation; with parenthesis are in hexadecimal notation.

** Range varies with CPU memory size

*** With Enhanced CCM (Refer to Appendix B, Expanded Functions)

Table 2.14 TARGET/SOURCE MEMORY ADDRESSES (Continued)

Target/Source Memory Type	Description	Target/Source Address Range	
7 - User Logic Memory	Specifies the user logic memory word at which the data transfer is to begin. The maximum value depends on the CPU memory size.	CPU Memory Size	Range
		2K Memory	00000-02047 (07FF)
		4K Memory	00000-04095 (0FFF)
		8K Memory	00000-08191 (1FFF)
		32K Memory	00000-32767 (7FFF)
	With CPU microcode >130	*** 64K Memory	00000-65535 (FFFF)
8 - CCM Quick Access Buffer	Specifies the CCM Quick Access Buffer byte where the data transfer is to begin.		00000-01023 (0000-03FF)
9 - CCM Diagnostic Status Words	Specifies the CCM Diagnostic Status Word at which the data transfer is to begin.		00001-00020 (0001-0014)
10 - OIU Timers 11 - OIU Counters	Specifies the number of timers or counters to be assigned for use by the OIU.		00000-00512 (0000-0200) combined total of timers and counters
Enhanced CCM Firmware only **			
13 - Bit Set for Input Table			1-32768 (1-8000)
14 - Bit Set for Output Table			1-32768 (1-8000)
15 - Bit Set for Input Override Table / Aux Input Override			1-1024 / 8193-9216 (1-400)/(2001-2400)
16 - Bit Set for Output Override Table / Aux Output Override			1-1024 / 8193-9216 (1-400)/(2001-2400)
17 - Bit Clear for Input Table			1-32768 (1-8000)
18 - Bit Clear for Output Table			1-32768 (1-8000)
19 - Bit Clear for Input Override Table / Aux Input Override			1-1024 / 8193-9216 (1-400)/(2001-2400)
20 - Bit Clear for Output Override Table / Aux Output Override			1-1024 / 8193-9216 (1-400)/(2001-2400)
21 - Bit Toggle for Input Table			1-32768 (1-8000)
22 - Bit Toggle for Output Table			1-32768 (1-8000)

* Ranges without parentheses are in decimal notation; with parentheses are in hexadecimal notation.

** Memory Types 13-22 may be used only with Enhanced CCM firmware.

*** Enhanced CCM (Refer to Appendix B, Expanded Functions)

Rn + 4: Data Length (Range: See Table 2.15)

This is the length of the data transfer. The units are determined by the source data type.

Table 2.15 DATA LENGTH

Source Data Type	Unit Length	Range
	Points	
Inputs	(Must be multiples of 8)	8-1024
Outputs		8-32768 ***
	(Must be in multiples of 16) If address is > 1024	16-32768 ***
Overrides	(Must be multiples of 8)	8-1024
	(Must be in multiples of 16) If address is > 1024	8193-9216 ***
Registers	Registers[1]	1-16384 **
User Logic	Words[2]	1 word to 64K - 1words depending on CPU memory size, and CPU microcode version >130
Quick Access Buffer	Bytes[3]	1-1024
Diagnostic Status Words	Words[2]	1-20

** Range varies with CPU memory size.

*** Enhanced CCM (Refer to Appendix B for Expanded Functions)

[1] A register consists of 16 bits.
[2] A word consists of 16 bits.
[3] A byte consists of 8 bits.

Rn + 5: Source Memory Address (Range: see Table 2.14)

The source address specifies the address within the source device where the transfer is to begin. The source memory address descriptions and ranges are the same as for target memory as shown in Table 2.14.

CCM COMMUNICATION REQUEST STATUS AND DIAGNOSTIC INFORMATION

In any communications system there are many possible causes for data to be transferred incorrectly or for the transfer to fail completely. These causes include hardware, software and human errors. The CCM provides two powerful tools to the user for monitoring and diagnosing errors in the transmission of data: The CCM status byte and the diagnostic status words.

CCM Status Byte

An 8-bit status byte is transferred from the CCM to CPU inputs I1009-I1016 during each CCM window. This byte indicates the status of the CCM and is not to be confused with the CPU [STATUS] function. Each bit of the CCM status byte is explained in the table below.

Table 2.16 STATUS BYTE DEFINITION (CCM and RTU)

Input Number	Bit	Status Bit Definition	Description And Use
1009	1	CCM busy with port [SCREQ]	Set to 1 when CCM accepts a [SCREQ] command from the CPU and reset to 0 upon completion. This bit can be used as an interlock to prevent a [SCREQ] command from being issued while another is in progress.
1010a	2	[SCREQ] complete without error.	Bits 2-6 are pulsed by the CCM when the condition causing the status change occurs. The pulse function ensures that the bit will be set to 1 for 3 windows minimum then will be set to 0 for 3 windows minimum. The pulse function for a particular status bit must be complete before another pulse function for the same status bit can be activated.
1011a	3	[SCREQ] complete with error.	
1012b	4	Externally initiated READ occurred successfully.	
1013b	5	Externally initiated WRITE occurred successfully.	
1014a	6	Q response sent. (CCM mode only)	The pulse function of bits 2, 4, 5, 6 can be used to sequence serial communications requests. The pulse function of bit 3 can be used to signal an error condition to be acted upon as the user desires.

For Input Numbers having a suffix (a, b), refer to the note on the next page.

Table 2.16 STATUS BYTE DEFINITION (CCM and RTU) (Continued)

Input Number	Bit	Status Bit Definition	Description And Use
1015	7	Spare (always 0)	
1016	8	CCM-CPU Communications OK	The bit is set to a 1 upon passing power-up test and once per CCM window as long as CCM/CPU communications remain OK. If CCM/CPU communications fail after power-up, the bit is not reset to 0, it remains in its last state. The user can periodically reset this bit to a 0 and later check to see if it has returned to a 1 to monitor CCM/CPU communications.

a The pulses can take longer to send to the CPU than the communication takes. In this case the pulses will be queued and will continue after the communication ends until the queue is exhausted. The queue is 255 pulses long and rolls over to zero.

b For these bits the pulses are not queued if a communication is completed before the pulse function has ended from a previous communication.

CCM Diagnostic Status Words

In addition to the status byte which is automatically transferred from the CCM to the CPU, there are 20 diagnostic status words which are maintained and updated in the CCM. These status words are not automatically transferred to the CPU; the internal [SCREQ] command 06003 (Read CCM Diagnostic Status Words to Source Registers) is used to transfer these status words to the CPU. An external device can access these registers using a READ command with target memory type 9 (Read CCM Target to Source). The following table explains the purpose of each diagnostic status word.

Table 2.17 CCM DIAGNOSTIC STATUS WORD DEFINITION

Diagnostic Status Word	Word Contents Bit 16	Bit 1	Notes
1	J2 Port Error Code	J1 Port Error Code	See Serial Port Error Code Table.
2	Number of Successful Conversations on J1 Port		
3	Number of Successful Conversations on J2 Port		
4	Number of Aborted Conversations on J1 Port		
5	Number of Aborted Conversations on J2 Port		
6	Number of Header Re-tries on J1 Port		
7	Number of Header Re-tries on J2 Port		
8	Number of Data Block Re-tries on J1 Port		
9	Number of Data Block Re-tries on J2 Port		
10	CCM Configuration on J1 Port **		** Reflects either the jumper and DIP switch configuration or if configuring from registers the bit patterns in these registers.
11	CCM Configuration on J2 Port **		
12	CCM Software Version Number		
13	[SCREQ] Error Code		See [SCREQ] Error Code Table.
14	Pointer to reference register of last [SCREQ] command which failed		

Table 2.17 CCM DIAGNOSTIC STATUS WORD DEFINITION (Continued)

Diagnostic Status Word	Word Contents Bit 16	Bit 1	Notes
15	Rn		Contents of 6 [SCREQ] registers for last [SCREQ] command which failed.
16	Rn + 1		
17	Rn + 2		
18	Rn + 3		
19	Rn + 4		
20	Rn + 5*		

*If an error occurs when executing port commands in which the source memory is an input table, output table, input override table, or output override table, diagnostic status word number 20 is incorrect for CCM PROM software revision 258 (102 Hex) or earlier. Instead of displaying the actual input or output value in Rn+5, it displays the byte in which the input or output occurs minus 1. Therefore, if input 17 is specified as the source memory address, diagnostic status word number 20 will contain a 2.

The following is a list of all of the error codes that are reported in the two bytes of CCM Diagnostic Status Word 1.

Table 2.18 CCM SERIAL PORT ERROR CODES (DIAGNOSTIC STATUS WORD 1)

ERROR CODE		DESCRIPTION
Dec	Hex	
0	00	Successful transfer.
1	01	A time out occurred on the serial link.
2	02	An external device attempted to write data to a section of the CPU Scratch Pad that is permanently write protected by the CCM .
3	03	An external device attempted to read or write a non-existent I/O point.
4	04	An external device attempted to access more data than is available in a particular memory type.
5	05	An external device attempted to read or write an odd number of bytes to Register memory, User Logic memory, or the Diagnostic Status Words.
6	06	An external device attempted to read or write one or more non-existent Registers.
7	07	An external device specified the transfer of zero data bytes.
8	08	An external device attempted to write to protected memory.
9	09	An external device attempted to transfer data to or from an invalid memory type or absolute source address.
10	0A	An external device attempted to read or write one or more non-existent Diagnostic Status Words.
11	0B	An external device attempted to transfer data beginning at an invalid User Logic memory, Scratch Pad, or Quick Access Buffer address.
12	0C	Serial communication was aborted after a data block transfer was retried three times.
13	0D	Serial communication was aborted after a header transfer was retried three times.
14	0E	Serial communication was aborted after a Q Response was retried three times.

Table 2.18 CCM SERIAL PORT ERROR CODES (Continued)
(DIAGNOSTIC STATUS WORD 1)

ERROR CODE Dec	ERROR CODE Hex	DESCRIPTION
20	14	One or more of the following errors occurred during a data block transfer: a) An invalid STX character was received; b) An invalid ETB character was received; c) An invalid ETX character was received; d) An invalid LRC character was received; e) A parity, framing, or overrun error occurred.
21	15	The CCM expected to receive an EOT character from an external device and did not receive it.
22	16	The CCM expected to receive an ACK or NAK character and did not receive either one.
23	17	Communication was aborted when the CCM did not receive a valid acknowledge to a master enquire sequence after 32 attempts.
24	18	Communication was aborted after a peer enquire was NAKed 32 times by the external device.
25	19	Communication was aborted when the CCM did not receive a valid response to a peer enquire after 32 attempts.
26	1A	A time out occurred during an attempt to transmit on a port due to CTS being in an inactive state too long.
29	1D	An error occurred when data was being transferred between the CCM and the Series Six CPU.
30	1E	A parity, framing, or overrun error occurred during a serial header transfer.
31	1F	A parity, framing, or overrun error occurred during a serial data block transfer.
48	30	A SCREQ attempted to initiate a conversation on a port being used by the OIU.

The following is a list of all of the error codes that are reported in Diagnostic Status Word 13.

Table 2.19 CCM [SCREQ] ERROR CODES
(DIAGNOSTIC STATUS WORD 13)

ERROR CODE		DESCRIPTION
Dec	Hex	
1	01	The command number is invalid.
2	02	The source address and/or data length is invalid.
3	03	The source address and/or data length is invalid when referring to the Diagnostic Status Words or the Quick Access Buffer.
4	04	The SCREQ specified a write to protected memory.
5	05	The SCREQ attempted to initiate a conversation on slave port J1.
6	06	The SCREQ attempted to initiate a conversation on slave port J2.
7	07	The SCREQ specified an invalid Target ID number when attempting to initiate a conversation on master port J1.
8	08	The SCREQ specified an invalid Target ID number when attempting to initiate a conversation on master port J2.
9	09	Memory protect SCREQ specified an invalid address and/or length.
10	0A	Memory protect SCREQ specified an invalid memory type.
11	0B	Communication, initiated by an SCREQ, was aborted when the CCM did not receive a valid acknowledge to a master enquire sequence after 32 attempts.
12	0C	Communication, initiated by an SCREQ, was aborted when the CCM did not receive a valid acknowledge to a peer enquire sequence after 32 attempts.
13	0D	A time out occurred when neither an ACK nor a NAK character was received in response to a header.
14	0E	A time out occurred during an attempt to transmit a header due to CTS being in an inactive state too long.
15	0F	A SCREQ attempted to initiate a conversation on a port that is being used by the OIU.
16	10	The CCM did not receive the ACK or NAK character that it expected to receive.

Table 2.19 CCM [SCREQ] ERROR CODES
(DIAGNOSTIC STATUS WORD 13) (Continued)

ERROR CODE Dec	ERROR CODE Hex	DESCRIPTION
17	11	A parity, framing, or overrun error occurred on the serial link during a data block transfer.
18	12	The CCM did not receive an EOT character that it was expecting.
19	13	A time out occurred on the serial link during the execution of a SCREQ.
20	14	An error occurred when data was being transferred between the CCM and the Series Six CPU.
21	15	The Read Character String SCREQ specified a non-existent Output point.
22	16	The Read Q Response SCREQ attempted to execute on a non-master port (Port J1).
23	17	The Read Q Response SCREQ attempted to execute on a non-master port (Port J2).
24	18	The serial communication was aborted after a Q Sequence was retried three times.
25	19	An error occurred on the serial link.
26	1A	A Write Then Read Immediate Character String SCREQ specified an invalid read register and/or an invalid read data count.
27 *	1B	Invalid retry count specified for ENQs.
28 *	1C	Invalid retry count specified for: Q sequence, Header, or Data Blocks.
29 *	1D	Invalid Timeout specified for: ACK / NAK for ENQ, Start of Header following ACK of ENQ, or EOT to close the link.
30 *	1E	Invalid Timeout specified for start to end of header.
31 *	1F	Invalid Timeout specified for ACK / NAK following Header
32 *	20	Invalid Timeout specified for: Start of Data following ACK of Header, or ACK /NAK following Data Block.
32 *	21	Invalid Timeout specified for Data to Finish.

* For more information, refer to Appendix B, Expanded Functions.

[SCREQ] COMMAND PROGRAMMING EXAMPLES

The following pages contain a full explanation and example of each command type (Internal and Port Commands). They are in numerical order beginning with the internal commands. The command sets for the Port commands (J1 and J2) are identical so the examples will not be duplicated. The command numbers are specified in decimal, with the hexadecimal equivalent shown in parenthesis.

INTERNAL COMMAND: 06001 (1771) SET Q RESPONSE

DESCRIPTION :

- A CCM setting the Q response must be configured as a slave.
- The execution of this command sets up the 4 data bytes that comprise the CCM data portion of the slave Q response.

  The Read Q Response to source Register Table command is used by a CCM master to read a CCM slave Q response data.
- Exceptions to SCREQ register definitions:
  Rn+1: Data bytes 1 and 2 of Q response
  Rn+2: Data bytes 3 and 4 of Q response
- Data byte format:

	Bit 16	Bit 1
Rn+1	DATA BYTE 2	DATA BYTE 1
Rn+2	DATA BYTE 4	DATA BYTE 3

PROGRAM EXAMPLE :Set Q response with the numbers 1,2,3,4.

Rn :	06001 (1771)	Command Number
Rn+1:	00513 (0201)	Data Bytes 1 and 2
Rn+2:	01027 (0403)	Data Bytes 3 and 4
Rn+3:	-	
Rn+4:	-	
Rn+5:	-	

CROSS REFERENCE : See port commands 06109/06209, Read Q Response to Source Register Table.

INTERNAL COMMAND: 06002 (1772) CLEAR CCM DIAGNOSTIC STATUS WORDS

DESCRIPTION : This command requires only the command number, Rn; it clears diagnostic status words 1 through 9 and 13 through 20.

PROGRAM EXAMPLE : Clear diagnostic status words

Rn :	06002 (1772)	Command Number
Rn+1:	-	
Rn+2:	-	
Rn+3:	-	
Rn+4:	-	
Rn+5:	-	

CROSS REFERENCE : See internal command 06003, Read CCM Diagnostic Status Words to Source Registers.

INTERNAL COMMAND: 06003 (1773) READ CCM DIAGNOSTIC STATUS WORDS TO SOURCE REGISTERS

DESCRIPTION : There are 20 consecutively numbered diagnostic status words which can be read by the CPU. A transfer of all or part of the diagnostic status words can be made to the CPU as long as they are in a consecutive block.

PROGRAM EXAMPLE : Read the first five diagnostic status words to source registers R0050-R0054.

Rn :	06003 (1773)	Command Number
Rn+1:	-	
Rn+2:	-	
Rn+3:	00001 (0001)	Target Memory Address
Rn+4:	00005 (0005)	Data Length (words)
Rn+5:	00050 (0032)	Source Register

CROSS REFERENCE : A Series Six PLC can read the diagnostic status words of another CCM using the port commands 06101-06106 or 06201-06206.

INTERNAL COMMAND: 06004-06006 (1774-1776) LOAD CCM QUICK ACCESS BUFFER FROM REGISTERS, INPUTS, OR OUTPUTS

DESCRIPTION:

- The QAB is 1024 8-bit bytes (512 registers) long; the user can structure the QAB in any manner he desires. The QAB is resident on the CCM module.
- When data is transferred to and from the QAB, it is all transferred during one window no matter what the data length. For a total of 1024 bytes, this will take approximately 24 msec. The transfer time for less than all 1024 bytes can be calculated as:

  Transfer time = 3.4 msec + 18 usec X no. of bytes transerred.

- Exceptions to SCREQ register definitions:
  Rn+3: QAB memory address (0 to 1023)
- QAB Data byte format
  Two QAB data bytes can be loaded from a single CPU register. Bits 1-8 of the register form the LSB and bits 9-16 form the MSB.
- The beginning data byte address of the QAB is 00000.

PROGRAM EXAMPLE : Load the first 4 data bytes of the CCM QAB from source registers R0050 and R0051 with the four numbers 1,2,3,4.

```
Rn   :    06004 (1774)   Command Number
Rn+1:     -
Rn+2:     -
Rn+3:     00000 (0000)   Target Memory Address
Rn+4:     00002 (0002)   Data Length (Registers = bytes/2)
Rn+5:     00050 (0032)   Source Memory Address
```

For the example to function properly, CPU registers R0050 and R0051 must contain the following values upon execution of the command example.

```
R0050:    00513 (0201)
R0051:    01027 (0403)
```

QAB data byte contents after execution of command example.

```
Data byte 0:   00001
Data byte 1:   00002
Data byte 2:   00003
Data byte 3:   00004
```

The four numbers in this example are packed into 2 registers requiring a data length of 2.

CROSS REFERENCE : The CPU can read the CCM'S QAB using internal commands 06007-06009. A CPU can also read the QAB or another CPU using the port commands 06101-06106 or 06201-06206.

INTERNAL COMMAND: 06007-06009 (1777-1779) READ CCM QUICK ACCESS BUFFER TO REGISTERS, INPUTS, OR OUTPUTS

DESCRIPTION : See commands 06004-06006.

PROGRAM EXAMPLE : Read CCM QAB data bytes 100-101 to source inputs I0256-I0271. QAB data byte 100 contains 001 and QAB data byte 101 contains 255.

```
Rn   : 06008 (1778)   Command Number
Rn+1 : -
Rn+2 : -
Rn+3 : 00100 (0064)   QAB Memory Address
Rn+4 : 00016 (0010)   Data Length
Rn+5 : 00256 (0100)   Source Memory Address
```

Status of inputs I0256-I0271 after execution of command example:

I0271															I0256
1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	1

The data length is the number of input points equivalent to 2 data bytes (16 input points).

CROSS REFERENCE : The CPU can load the CCM'S QAB using internal commands 06004-06006. A CPU can load another CPU'S QAB using port commands 06101-06106 or 06201-06206, along with specifying a target memory type equal to QAB (8).

INTERNAL COMMAND : 06010 (177A) SET CPU MEMORY WRITE PROTECT

DESCRIPTION :

- This command provides the user with a mechanism to protect all but a specified block of each CPU memory type from being overwritten by an external serial device such as another CPU or an Operator Interface Unit (OIU).

- Exceptions to SCREQ register definitions:
  Rn+2: Protected Memory Type
  Rn+3: Starting Memory Address of unprotected block
  Rn+4: Data Length of unprotected block

- If a data length of 00000 is specified then the entire memory type is write protected.

- The Set CPU Memory Write Protect function can be executed for each memory type. (Refer to Table 2. Status Byte Definition)

CCM Memory Type	CCM Target Table
0*	Absolute
1	Register Table
2	Input Table
3	Output Table
4*	Input Override Table
5*	Output Override Table
6*	CPU Scratch Pad Memory
7*	User Logic Memory
8	CCM Quick Access Buffer
9	CCM Diagnostic Status Words

  * Memory types 0, 4, 5, 6, and 7 are protected by the CPU memory switch.

- Cycling power on the CPU rack will remove the Write Protect settings.

PROGRAM EXAMPLE : Set the CPU Memory Write Protect so that only registers R0001-R0050 can be written to in the CPU.

```
Rn   :  06010 (177A)   Command Number
Rn+1:  -
Rn+2:  00001 (0001)    Protected Memory Type
Rn+3:  00001 (0001)    Starting Memory Address
Rn+4:  00050 (0032)    Unprotected data length
Rn+5:  -
```

NOTE

When using the Input, Output, Input Override, and Output Override tables, the memory address must begin on a byte boundary and the data length must be a multiple of 8.

INTERNAL COMMAND: 06011 (177B) REINITIALIZE CCM TIMER AND USART

DESCRIPTION :

- Execution of this command will cause the reinitialize diagnostic to occur. This diagnostic reads the CCM configuration information either from DIP switches or from Registers R0247 and R0248 and programs the timer and USART for the desired mode of operation.
- This command can be used when an error condition is detected or when doing on-line configuration. See section, Software Configuration.

PROGRAM EXAMPLE : Reinitialize CCM Timer and USART

Rn : 06011 (177B) Command Number
Rn+1: -
Rn+2: -
Rn+3: -
Rn+4: -
Rn+5: -

INTERNAL COMMAND: 06012 (177C) SET OIU TIMERS AND COUNTERS

DESCRIPTION :

- This command defines the location of timers and counters for the OIU function.
- The execution of this command will cause the CCM to define the location and number of registers used for the presets and accumulates for the OIU timers and counters.
- Exceptions to the SCREQ register definitions:
  Rn+2: Timer memory type = 10, Counter memory type = 11
  Rn+3: Address of first preset register
  Rn+4: Number of timers or counters
  Rn+5: Address of first accumulator register
- There must not be overlap in the address ranges defined for timer and counter preset and accumulate registers.
- A data length of 00000 specifies 0 counters or timers.
- The maximum number of any combination of timers and counters is 512.
- The table default values are as follows:

Number of timers	:	24
Timer presets	:	Registers R0011-R0034
Timer accumulators	:	Registers R0061-R0084
Number of counters	:	24
Counter presets	:	Registers R0036-R0059
Counter accumulators	:	Registers R0086-R0109

PROGRAM EXAMPLE : Assign 5 timers with presets beginning at R0200 and accumulators at R0205

Rn :	06012 (177C)	Command Number
Rn+1:	-	
Rn+2:	00010 (000A)	Timer Memory
Rn+3:	00200 (00C8)	First Timer Preset Register
Rn+4:	00005 (0005)	Number of Timers
Rn+5:	00205 (00CD)	First Counter Preset Register

NOTE

CCM PROM Revision 258 (102 Hex) or higher is required for Command 06012, Set OIU Timers and Counters, to work properly.

INTERNAL COMMAND: 06130, 06230 PROGRAMMABLE RETRIES FOR CCM
(17F2, 1856)

DESCRIPTION : • This command allows the user to program the maximum number of retries for the CCM protocol. Different retry values may be programmed for each port. Programmable ranges and default values for each retry that is programmable are listed below:

Entry	Description	Range	Default
(a)	Peer-Peer, Master/Slave ENQuiry, and Retry Count	0 to 32 times	32
(b)	Q Sequence Retry Count	0 to 5 times	3
(c)	Header Retry Count	0 to 5 times	3
(d)	Data Block Retry Count	0 to 5 times	3

SCREQ #	Rn + 1	Rn + 2	Rn + 3	Rn + 4	Rn + 5
6130	-	(a)	(b)	(c)	(d)
6230	-	(a)	(b)	(c)	(d)

PROGRAM EXAMPLE : Set the following retry values: ENQ retry count of 2, Header Retry count of 3, and Data Block retry count of 3. The port in use is port J2. The protocol is peer-to-peer.

Rn :	06230	(1856)	Command Number
Rn+1:	-		
Rn+2:	2	(000D)	ENQ Retry Count
Rn+3:	0	(0000)	Q Sequence Retry Count, Not Applicable for Peer-to-Peer
Rn+4:	3	(0005)	Header Retry Count
Rn+5:	3	(0003)	Data Block Retry Count

NOTE

Use of this command is recommended to solve the 18 second timeout condition when slaves are not present. Programming 2 retries for ACK to ENQ reduces the wait time by the master device to 1.63 seconds.

INTERNAL COMMAND: 06131, 06231 PROGRAMMABLE TIMEOUTS FOR CCM
(17F3, 1857)

DESCRIPTION : • This command allows the user to program timeout values for the CCM protocol. Different timeout values may be programmed for each port. Programmable ranges and default values for each timeout that is programmable are listed below:

Entry	Description	Range	Default
(a)	ACK/NAK for ENQ, Start of Header after ACK of ENQ, and EOT to Close Link	50 to 2000 msec	800
(b)	Header to Finish	50 to 3000 msec	*
(c)	ACK/NAK for Header	50 to 10000 msec	2000
(d)	Start of Data after ACK of ENQ, and ACK/NAK following Data Block	50 to 65000 msec	20000
(e)	Data to Finish	50 to 65000 msec	*

* Default value depends on selected data rate

SCREQ #	Rn + 1	Rn + 2	Rn + 3	Rn + 4	Rn + 5
6131	(a)	(b)	(c)	(d)	(e)
6232	(a)	(b)	(c)	(d)	(e)

PROGRAM EXAMPLE : Set the following timeout values: 100 msec timeout for ACK/NAK for ENQ; 300 msec timeout for header to finish; 200 msec for ACK/NAK for header; and 2000 msec timeout for data to finish. The port in use is port J1. The protocol is peer-to-peer.

```
Rn   :  06131 (17F3)   Command Number
Rn+1:  00100 (0064)   ACK/NAK for ENQ Timeout
Rn+2:  00300 (012C)   Timeout for Header to finish
Rn+3:  00200 (00C8)   ACK/NAK for Header Timeout
Rn+4:  00500 (01F4)   Timeout for Start of Data following
                      ACK of Header
Rn+5:  02000 (07D0)   Time for Data to Finish
```

PORT COMMAND : 06101-6106, 06201-6206 (17D5-17DA, 1839-183E) READ TARGET TO SOURCE MEMORY

DESCRIPTION : This set of commands is used to read information from the target device to one of the six source memory types listed below:

Register table	06101, 06201
Input table	06102, 06202
Output table	06103, 06203
Input override table	06104, 06204
Output override table	06105, 06205
Quick access buffer	06106, 06206

CCM Memory Type	CCM Target Table
0*	Absolute
1	Register Table
2	Input Table
3	Output Table
4*	Input Override Table
5*	Output Override Table
6*	CPU Scratch Pad Memory
7*	User Logic Memory
8	CCM Quick Access Buffer
9	CCM Diagnostic Status Words

* Memory types 0, 4, 5, 6, and 7 are protected by the CPU memory switch.

PROGRAM EXAMPLE: Read from target CCM diagnostic status words 1-9 to source registers R0936-R0944. The communication is to take place on port J2; the target ID is 36.

Rn :	06201 (1839)	Command Number
Rn+1:	00036 (0024)	Target ID
Rn+2:	00009 (0009)	Target Memory Type
Rn+3:	00001 (0001)	Target Memory Address
Rn+4:	00009 (0009)	Data Length
Rn+5:	00936 (03A8)	Source Memory Address

NOTE

When using the Input, Output, Input Override, and Output Override tables, the memory address must begin on a byte boundary and the data length must be a multiple of 8.

PORT COMMAND : 06108,06208 READ CHARACTER STRING TO SOURCE REGISTER TABLE (17DC,1840) (Unformatted Protocol)

DESCRIPTION :

- The execution of this command will cause the data bytes received on the specified port to be stored in the CPU register table. This command is commonly used to input characters from a terminal connected to the serial port.

- The standard CCM or RTU serial protocol is not used and the data is received according to the port configuration. Parity, data rate, and physical interface type (RS-232D, RS-422) must match between terminal and CCM.

- Exceptions to SCREQ register definitions:

  Rn+3: Output point number (O0001-O1024). User can terminate command by forcing selected output ON.

  Rn+4: Data length in registers (00002-00128). It is defined as the number of registers reserved for storing incoming data and the byte count register. When the characters read in fill the number of registers reserved by the data length register, the data transfer stops and the command is complete.

  Rn+5: Source address. It is the register number assigned to contain the count of the number of characters which are to be received through the port. As characters are received, the CCM automatically updates the byte count. The registers immediately following the register specified in Rn+5 are reserved for the actual characters read in.
  The maximum number of characters that can be transferred is:
  (128 reg - 1 reg for byte count) x 2 bytes/reg = 254 bytes

PROGRAM EXAMPLE : Read 4 characters through port J2 to CPU registers R0101-R0102.

Rn :	06208 (1840)	Command Number
Rn+1:	-	
Rn+2:	-	
Rn+3:	01024 (0400)	Output Point (terminator)
Rn+4:	00003 (0003)	Data Length
Rn+5:	00100 (0064)	Source Memory Address

If the characters S T O P (in caps) are entered via a terminal, then R0100-R0102 will contain the following after execution of the command:

R0100	4	Byte count (automatically updated)
R0101	54 53	Hex form of ASCII characters
	T S	ASCII characters
R0102	50 4F	Hex form of ASCII characters
	P 0	ASCII characters

If an odd number of bytes is to be read in, the command will not terminate automatically. Therefore, if in the example above it is desired to read only 3 characters, the command will not terminate automatically. To handle this situation, the user can monitor the byte count register and when it equals 3 (for 3 characters), trigger O1024 to terminate the command.

CROSS REFERENCES: See commands 06118, 06218 Write Character String From Source Register Table, and 06128-06228 Write Then Read Immediate Character String.

PORT COMMAND : 06109,06209 READ Q RESPONSE TO SOURCE REGISTER TABLE (17DD,1841)

DESCRIPTION :

- This command is used by a CCM master to read a CCM slave Q response data. A CCM slave sets the Q response data using command 6001.
- The target ID must be in the range from 1-90.

PROGRAM EXAMPLE : Read Q response through port J1 to source registers R0100-R0101. The data in the slave Q response is 1,2,3,4 as set by the example for command 06001, Set Q Response. The target ID is 33 (21).

```
Rn   :   06109 (177D)   Command Number
Rn+1:   00033 (0021)    Target ID
Rn+2:   -
Rn+3:   -
Rn+4:   -
Rn+5:   00100 (0064)    Source Memory Address
```

The data read into moved to R0100-R0101 is as follows:

	MSB	LSB
R0100	02	01
R0101	04	03

CROSS REFERENCE : See Internal Command, 06001, Set Q Response

PORT COMMAND: 06110, 06210 SINGLE BIT WRITE
(17DE, 1842)

DESCRIPTION : • This command allows the user to set, clear or toggle a single bit in the input or output table, and set or clear a single bit in the override tables of another CPU. The memory types for the single bit write function are listed below:

CCM Memory Type	CCM Target Table	Bit Operation
13	Input Table	Bit Set
14	Output Table	Bit Set
15	Input Ovrd Table	Bit Set
16	Output Ovrd Table	Bit Set
17	Input Table	Bit Clear
18	Output Table	Bit Clear
19	Input Ovrd Table	Bit Clear
20	Output Ovrd Table	Bit Clear
21	Input Table	Bit Toggle
22	Output Table	Bit Toggle

PROGRAM EXAMPLE : Clear Output 713 in the Series Six PLC with Target ID 25. The communication is to take place on port J1. The target ID is 25.

Rn :	06110 (17DE)	Command Number
Rn+1:	00025 (0019)	Target ID
Rn+2:	00018 (0012)	Memory Type/Function
Rn+3:	00713 (02C9)	Target Memory Address
Rn+4:	-	
Rn+5:	-	

NOTE

If a bit is manually toggled in the user program -- the toggle operation may produce unexpected results.

PORT COMMAND : 06111-06117, 06211-06217 WRITE TO TARGET FROM SOURCE (17DF-17E5, 1843-1849)

DESCRIPTION : This set of commands is used to write information to the target device from one of the 6 source memory types listed below:

Register table	06111, 06211
Input table	06112, 06212
Output table	06113, 06213
Input override table	06114, 06214
Output override table	06115, 06215
Quick access buffer	06116, 06216
User logic memory	06117, 06217

PROGRAM EXAMPLE : Write to target registers R0200-R0299 from source registers R0001-R0100. The target ID is 10. The communication takes place on port J1.

Rn :	06111 (17DF)	Command Number
Rn+1:	00010 (000A)	Target ID
Rn+2:	00001 (0001)	Target Memory Type
Rn+3:	00200 (00C8)	Target Memory Address
Rn+4:	00100 (0064)	Data Length
Rn+5:	00001 (0001)	Source Memory Address

NOTE

When using the Input, Output, Input Override, and Output Override tables, the memory address must begin on a byte boundary and the data length must be a multiple of 8.

PORT COMMAND : 06118, 06218 (17E6, 184A) WRITE CHARACTER STRING FROM SOURCE REGISTER TABLE (Unformatted Protocol)

DESCRIPTION :

- The execution of this command will cause data stored in the register table to be sent out of the specified port verbatim.
- The standard CCM or RTU serial protocol is not used and the data is transmitted according to the port configuration. Parity, data rate, turn-around delay, and the use of physical interface standards (RS-232D or RS-422) must match between terminal and CCM.
- Exceptions to SCREQ register definitions:

  Rn+4: Data length in registers (00002-00128). This length is defined as the number of registers reserved for storing the data and the byte count register.

  Rn+5: Source Memory Address. It is the register number assigned to contain the user defined count of the number of data bytes (characters) to be transmitted.
  The maximum number of characters that can be transmitted is:
  (128 reg - 1 reg for byte count) x 2 bytes/reg = 254 bytes
  The registers directly following the register specified in Rn+5 contain the actual data to be written out.

- The data must be transmitted within a specific length of time which is dependent upon the data rate. The data must be completely transmitted within the time defined by (10,000/data rate) + turn-around delay. Exceeding this time causes a serial time out to occur.

PROGRAM EXAMPLE :Write a 4-character string contained in R0101-R0102 out port J1.

Rn :	06118 (17E6)	Command Number
Rn+1:	-	
Rn+2:	-	
Rn+3:	-	
Rn+4:	00003 (0003)	Data Length
Rn+5:	00100 (0064)	Source Memory Address

If the characters S T O P (in caps) are to be written out, the data is stored in R0100-R0102 before execution of the command as follows:

R0100	4	user defined byte count
R0101	54 53	Hex form of ASCII characters
	T S	ASCII characters
R0102	50 4F	Hex form of ASCII characters
	P 0	ASCII characters

CROSS REFERENCES: See commands 06108,06208, Read Character String to Source Register Table and 06128,06228, Write Then Read Immediate Character String.

PORT COMMAND : 06128,06228 (17F0,1854) WRITE THEN READ IMMEDIATE CHARACTER STRING (Unformatted Protocol)

DESCRIPTION :

- The execution of this command produces the same result as the Write Character String command followed by the Read Character String command without any delay between the two commands. It is suggested that the user first become familiar with the Write Character String and Read Character String commands before attempting this command.

- As before the standard CCM or RTU serial protocol is not used and the data is transmitted and received according to the port configuration. Parity, data rate, turn-around delay, and the physical interface type (RS-232D, RS-422) must match between terminal and CCM.

- Exceptions to SCREQ register definitions:

  Rn+1: Data Length in registers of READ data plus the read byte count register.
  Rn+2: Source Memory Address (register) of READ data.
  Rn+3: Output Point Number. User can terminate READ data transfer by forcing output ON.
  Rn+4: Data Length in registers of WRITE data plus the write byte count register.
  Rn+5: Source Memory Address (register) of WRITE data.

PROGRAM EXAMPLE : Write the 4-byte character string S T O P located in R0101-R0102 out port J1 and then read the 2-byte character string G O back into R0151.

Rn :	06128 (17F0)	Command Number
Rn+1:	00002 (0002)	Data Length (READ data)
Rn+2:	00150 (0096)	Source Memory Address (READ data)
Rn+3:	01024 (0400)	Output Point Number
Rn+4:	00003 (0003)	Data Length (WRITE data)
Rn+5:	00100 (0064)	Source Memory Address (WRITE data)

Contents of registers from which data is written (upon execution of the command):

R0100	4	Byte count (user specified)
R0101	54 53	Hex form of ASCII characters
	T S	ASCII characters
R0102	50 4F	Hex form of ASCII characters
	P O	

Contents of registers to which data is written (after execution of the command):

R0150	2	Byte count (automatically updated)
R0151	4F 47	Hex form of ASCII characters
	O G	ASCII characters

CROSS REFERENCES: See commands 06108,06208 Read Character String to Source Register Table and 06118,06218 Write Character String from Source Register Table.

OPERATOR INTERFACE UNIT (OIU)

The Operator Interface Unit (OIU) is a hand-held device with the capability of monitoring and changing specified contents of the CPU.

CAPABILITIES OF THE OIU

The OIU can perform the following functions:

Display	:	Registers, inputs, outputs, and predefined timers and counters. (Maximum of 2 registers, timers, or counters at one time; maximum of 4 inputs or outputs at one time.)
Display Register Contents In	:	Decimal, hexadecimal, signed decimal, and double precision format.
Change	:	Register, timer, and counter values.
Force	:	Inputs or outputs ON or OFF.
Override	:	Inputs or outputs.
Search For	:	Inputs and outputs that are overridden.
Increment or Decrement Address of	:	Registers, timers, counters, inputs, or outputs being displayed.

There are two SCREQ commands directly associated with the OIU: 06010, Set CPU Memory Write Protect and 06012, Set OIU Timers and Counters. To implement these commands refer to the section, CPU/CCM Programming. Examples are given for both command types.

NOTE

CCM PROM Revision 258 (102 Hex) or higher is required for Command 06012, Set OIU Timers and Counters, to work properly.

CONFIGURING THE CCM FOR OIU OPERATION

See the section, Installation of the CCM, for a complete explanation of module hardware and software configuration.

Hardware Configuration

Jumpers and DIP switches are used to select the CCM interface characteristics. The table below shows the settings for OIU operation if the hardware configuration method is used.

Table 2.20 HARDWARE CONFIGURATION FOR THE OIU

FUNCTION	C-Closed O-Open	DIP SWITCHES							
		9	10	11	12	13	14	15	16
Port J1									
19.2 KBps Data Rate		0	C	C					
Peer RS-422 W/O Clocks					C	0	C		
0 msec Turn Around Delay								0	0
		1	2	3	4	5	6	7	8
Port J2									
19.2 KBps Data Rate		0	C	C					
Peer RS-422 W/O Clocks					C	0	C		
0 msec Turn Around Delay								0	0
		17	18	19	20				
Required Configuration Using Either Port									
Parity (odd)		C							
Required					0				
Don't Care			X	X					

Table 2.20 HARDWARE CONFIGURATION FOR THE OIU (Continued)

FUNCTION	JUMPER	PINS JUMPERED
OIU Enabled	JP4	2-3
OIU Power (+5 v to pin 20 of port J1)	JP6	
Enabled		2-3
Disabled		1-2
Required Positions	JP1	1-2
	JP2	1-2
	JP3	1-2
	JP5	1-2
	JP7	1-2
	JP8	1-2

NOTE

When using hardware configuration, the port OIU selections are as follows:

J1 Port - Operator Interface Device, OIU enabled, OIU non- polled, OIU memory protect enable.

J2 Port - Dumb Terminal, dumb terminal enabled, dumb terminal non-polled, dumb terminal memory protect enable.

Software Configuration

The CCM module can also be configured by CPU registers R0247 and R0248 when in the Software Configuration Mode. To enter the Software Configuration Mode place J1 port DIP switches 12, 13, 14 in the CLOSED position and ensure that J2 parity select switch 17 is in the (CLOSED) position (odd parity). Jumper JP6 for OIU power has no software equivalent and must be set using the jumper.

Register R0247 represents the configuration for serial port J1 and register R0248 represents the configuration for serial port J2. Table 2.21 shows the bit patterns for configuring either port.

Table 2.21 SOFTWARE CONFIGURATION FOR THE OIU

FUNCTION J1 (R0247) or J2 (R0248)	BITS 16	15	14	13	* 12	* 11	10	9	8	7	6	5	4	3	2	1
19.2 KBps Data Rate														1	1	0
Peer RS-422 W/O Clocks											1	0	1			
0 msec Turn Around Delay									0	0						
Parity								1								
OIU Device							0									
OIU Enable				0												
OIU Polled/Non-polled																
Non-polled			0													
Polled			1													
OIU Memory Protect																
Enabled		0														
Disabled		1														
Port Enabled/Disabled																
Enabled	0															
Disabled	1															

*Bits 11 and 12 are not used.

NOTE

When OIU or "dumb" terminal mode is selected, the CCM operates in a 7-bit even parity format. If a "dumb" terminal is used it must be configured as a 7-bit even parity device.

SIMULTANEOUS PORT OPERATIONS

Simultaneous operations are defined as communications taking place on both ports at the same time (both ports busy). In many cases, however, communications can appear simultaneous even if they are not. This occurs when the ports are alternately serviced in quick succession. Explained below are the cases in which true simultaneous operations are permissible and the action taken by the CCM when non-permissible simultaneous operations are requested.

CCM port communications can be initiated internally (CCM mode only) or by an external device. Series Six PLC applications programming should be used to prevent internally initiated port communications when a port is already busy. Failure to do this could result in a communication request being lost and not executed.

PERMISSIBLE SIMULTANEOUS OPERATIONS

A CCM residing in a CPU with an extended function set can properly perform simultaneous operations.

Table 2.22 PERMISSIBLE SIMULTANEOUS PORT OPERATIONS

ONE PORT	OTHER PORT
OIU	OIU
OIU	Q Sequence
CCM or RTU protocol	Q Sequence
Q Sequence	Q Sequence
Unformatted Write*	Q Sequence
Unformatted Read*	Q Sequence
Unformatted Write/Read*	Q Sequence

*These are CCM serial character string operations which do not use peer-to-peer or master-slave protocol.

As can be seen from the table, the Q Sequence operation can be performed regardless of the activity on the other port. Also, OIUs on both ports can perform simultaneous operations. No other combination of CCM serial operations, such as OIU and CCM protocol communications can be performed simultaneously.

ATTEMPTING NON-PERMISSIBLE SIMULTANEOUS OPERATIONS (CCM PROTOCOL)

If one port is busy (performing an operation) and a non-permissible simultaneous attempt is made by an external device to communicate on the other port, then the external device is NAKed by the CCM (indicating that the CCM is busy). When the CCM becomes idle again, priority is given (for 200 msec) to the port that NAKed the request.

If the external device is using CCM protocol, and it has received a NAK from the busy port, the external CCM will then delay 10 msec. or the turn-around delay (if it is not 0 msec.) and retry the enquiry sequence. It will do this until the port becomes idle and is able to respond or until it has retried the enquiry sequence 32 times (in peer-to-peer or normal sequence master-slave protocol) whichever comes first.

If the external device is a computer, the software driver should perform a retry sequence similar to the one the CCM uses as explained above and shown in the flow charts in Chapter 4, CCM Protocol.

CAUTION

If the simultaneous operations in Table 2.22 are attempted with a CCM residing in a CPU with the Basic Function Set, there is a possibility that it could cause the CPU to stop.

RTU PROTOCOL ON ONE PORT AND CCM PROTOCOL ON OTHER PORT

If one port is busy and an external request is made to the other port, the port receiving the request will not send a negative acknowledge to the external device. The incoming request enters a buffer and that request will be executed as soon as the other port is finished.

The user must be aware that the buffer does not stack external requests. If a second request is sent by the external device before the first request is serviced, the second request will not be serviced.

Care must be taken to ensure that a request by an external device is executed within the time required by the external device. A time-out could occur if the busy port is communicating at a slow data rate or even at a higher data rate if large amounts of data are being transmitted.

The only exception to the explanation above is when:

The RTU port is busy with a serial session and a Q sequence is initiated on the CCM port.

In this case, if the RTU port is busy at the time a Q Sequence arrives on the other port, the execution of the Q Sequence will be inter-leaved with the servicing of the RTU port. The Q Sequence uses an efficient protocol and only transfers 4 bytes of data at a time; therefore, the interruption should not present a timing problem on the other port.

RTU PROTOCOL ON BOTH PORTS

Normally, communications can occur on both ports at the same time. If the port is busy with an RTU request and a external request is received on the other port, the second request will be buffered until the busy port becomes idle. The user must be aware that the buffer does not stack external requests. If a third external request is sent before the second request (which is in the buffer) is serviced, the third request will not be serviced.

Care must be taken to ensure that a request received on one RTU port, when the other RTU port is busy, is executed within the time required by the external device.

CHAPTER 3
INPUT/OUTPUT COMMUNICATIONS CONTROL MODULE (I/O CCM)

INTRODUCTION TO THE I/O CCM

The Input/Output Communications Control Module (I/O CCM) provides a serial data link between a Series Six™ Programmable Logic Controller (PLC) and host computer, programmable terminal and many other intelligent devices. The I/O CCM resides in an I/O slot in the Series Six PLC, and more than one I/O CCM is allowed in a CPU configuration. Some devices which can be connected to the I/O CCM are:

- CCM2, CCM3, or I/O CCM in a Series Six PLC.
- Data Communications Unit (DCU) in a Series One™, or Series One Plus or Series One Junior PLC.
- Data Communications Module (DCM) in a Series Three PLC.
- WorkMaster™, VuMaster™ and FactoryMaster™ software running on the Workmaster computer.
- Intelligent devices such as a host computer.
- Process Control Systems.

The I/O CCM contains two independently configurable serial ports. Both ports support RS-232D and RS-422 serial interfaces, with Port 1 also supporting active/passive 20 mA current loop. Both ports support asynchronous serial communications with data rates of up to 19.2 Kbps. The user may select any of the following options using Dual-In-Line (DIP) switches.

- Data rate: 110 to 19.2 Kbps. Maximum data rate is limited to 4800 Kbps for current loop operation on Port 1.
- Protocol type: CCM - master, slave, or peer
  Remote Terminal Unit (RTU) - RTU slave
- Parity: even, odd, or none
- Turn-around delay: 0 or 500 msec (Port 2 only)

The I/O CCM can be used in communication systems using:

- Multidrop modem based links
- Multidrop RS-422 links
- Radio links (Port 2 only)

NOTE

As a master device port 1 or port 2 can be used in multidrop configurations. As a slave device only port 2 can be used in multidrop configurations.

The I/O CCM module provides isolation of the serial port receivers and transmitters and also provides 1500 volts of isolation protection from port to port and from the ports to the rest of the Series Six PLC system.

Six on-board Light-Emitting Diodes (LEDs) diagnostic and indicator lights show port activity and module status. These LEDs simplify troubleshooting and indicate correct data transfer. If the power-up diagnostics detect a failure, the BOARD OK LED will remain OFF and the lower five LEDs will provide an error code to specify the error. The CPU COMM LED blinks to indicate communications between the I/O CCM module and the Series Six CPU. The remaining four LEDs show port activity of the transmitters and receivers on both ports. They will BLINK when a port is communicating and will be OFF when an error occurs on a particular port. (See Tables 3.8 and 3.9 for the specific power-up error codes).

The user must provide Series Six CPU communication windows to the I/O CCM by use of the DPREQ instruction. Refer to later sections of this chapter on programming the I/O CCM.

The I/O CCM must be inserted in a High-Capacity I/O rack or a Series Six PLC rack I/O slot.

MODULE SPECIFICATIONS

Space Requirements:	One I/O slot in either a Series Six CPU rack, Series Six Plus CPU rack, or a High-Capacity I/O rack
Power Requirements:	+5 Vdc requirement is 1.5A -- 20 units of load +12 Vdc requirement is 300 mA -- 12 units of load (supplied by rack power)
Storage Temperature:	0° C to 70° C
Operating Temperature:	0° C to 60° C
Humidity:	5% - 95% (non-condensing)
Altitude:	Up to 6,600 feet (2,000 meters) above sea level (operating)
Isolation:	(Port to Port and either Port to Series Six common). Transient: 1500 Vac, 50/60 Hz for 1 minute maximum, non repetitive. Continuous: 240 Vdc or RMS ac, 50/60 Hz.
Noise & Transient:	Meets following specifications
Immunity:	Showering arcs per NEMA ICS 2,230.40 Surges per ANSI C37.90.9 5 W R.F. transmitter 27-450 Mhz

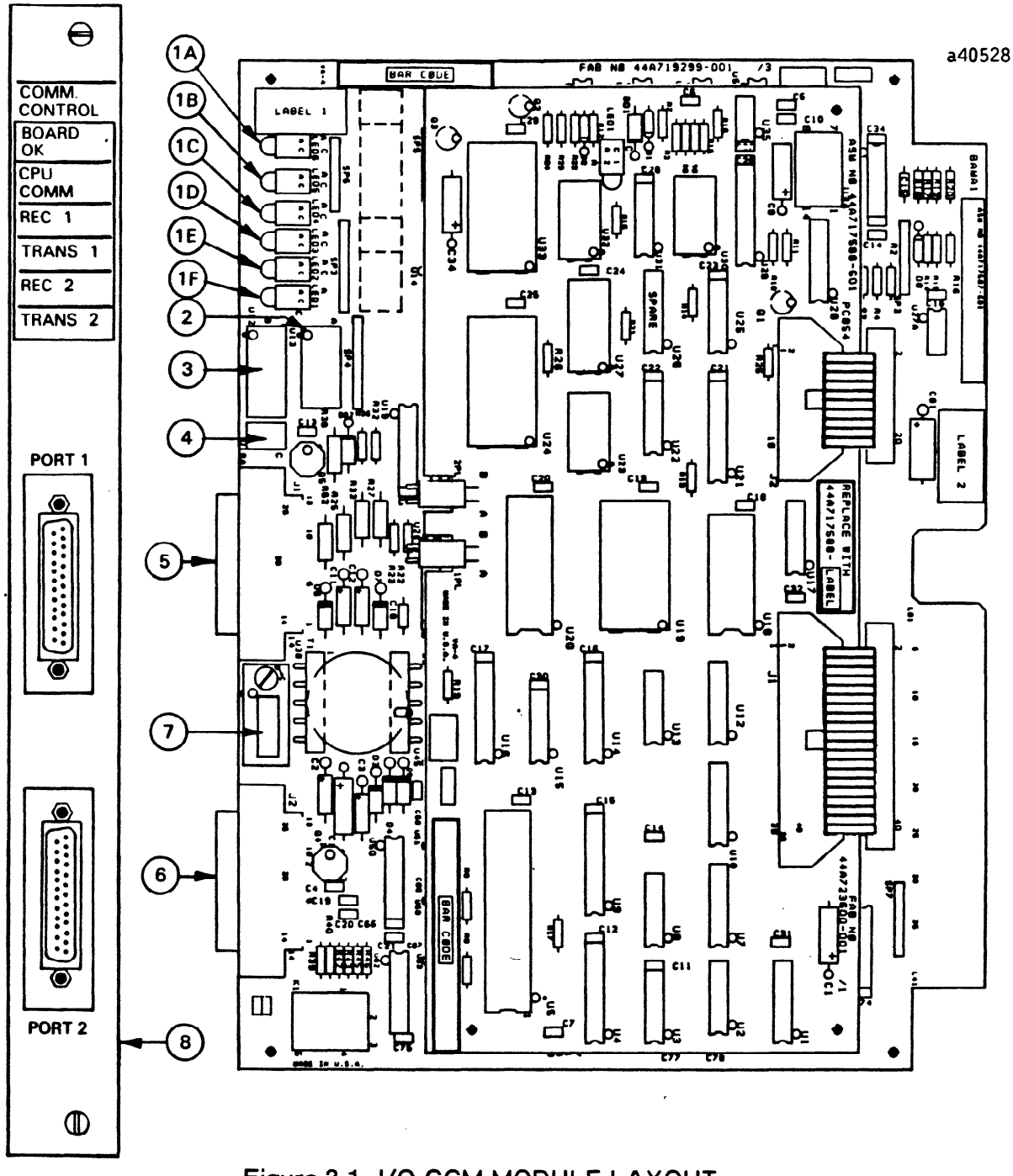


Figure 3.1 I/O CCM MODULE LAYOUT

1. LED Status Indicators (see Table 3.9).
2. Bank A DIP Switches.
3. Bank B DIP Switches.
4. Bank C DIP Switches.
5. J1 Connector: 25-pin D-type female connector (Communications Port 1).
6. J2 Connector; 25-pin D-type female connector (Communications Port 2)
7. J2 Communication selection DIP package: RS-232D or RS-422 configuration. (Read from top of imprinted label)
8. Faceplate

INSTALLING THE I/O CCM MODULE

Complete the steps as listed below to install and operate the I/O CCM module.

1. Calculate the total power requirements for the rack which will contain the I/O CCM. (Refer to "I/O CCM Power Requirements")

2. Configure the I/O CCM module.

   - Check the RS-232/RS-422 DIP package orientation -- for Port 2 only. (Reference Figure 3.2)

   - Configure the I/O CCM communication ports using the three on-board DIP switch packages: A, B and C. (Reference Tables 3.2, 3.3, 3.4)

3. Set the I/O CCM module address using the backplane DIP switch package. (Reference Figure 3.3, Table 3.1)

4. Insert the I/O CCM module into the rack.

5. Construct and install the I/O CCM port cable. (Reference Figures 3.4, 3.5, 3.6, and 3.7)

6. Power up and test the I/O CCM to verify that it is operating properly. (Reference Table 3.8)

7. Verify that the I/O CCM is communicating properly by use of the simple ladder logic examples and programming information provided later in this chapter. (Reference "Programming the I/O CCM")

NOTE

A special I/O terminator plug must be used when operating the I/O CCM module at the Data Processing Unit (DPU) Executive Window. The I/O Terminator Plug is dependent upon the operating environment.

I/O CCM POWER REQUIREMENTS

The I/O CCM may be installed in a Series Six CPU rack I/O slot, the Series Six High-Capacity I/O rack, or a Series Six Plus CPU rack.

The Series Six CPU rack can support a maximum of 300 units of load. A total of five I/O CCMs can be powered by the Series Six CPU rack, when no other loading exists for +12 Vdc. Alternately, four I/O CCMs and a normal CCM can be powered.

A maximum of five I/O CCM modules can be powered by a high capacity I/O rack. In this case there are 140 units of load remaining for I/O modules with +5v power only.

When other types of I/O modules are to be placed in the same rack as an I/O CCM, calculate the power requirements of all the modules to ensure that the maximum power of the rack is not exceeded. Refer to other sections of this chapter: "Module Specifications" and "Operational Information".

CONFIGURING THE I/O CCM MODULE

Configure the I/O CCM, prior to installing the module into the I/O rack.

Positioning the Hybrid DIP Package

The RS-422/RS-232 hybrid DIP package affects the operation of port 2 only. Verify the position of the configuration hybrid DIP package located between ports J1 and J2. It is marked "232" on one end and "422" on the other end and is mounted in a zero insertion force socket. Use a small screwdriver to turn the screw which releases the hybrid DIP package from the socket. Position the package with the desired interface type (RS-232 or RS-422) closest to port J1. See Figure 3.2 for proper package orientation.

a42442

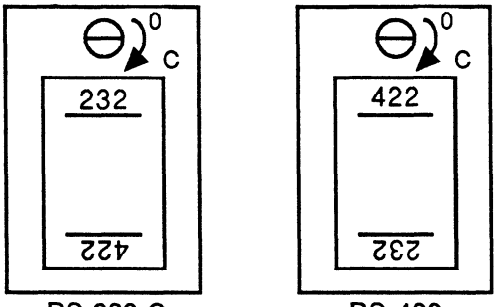


Figure 3.2 RS-232/RS-422 HYBRID DIP PACKAGE (FOR PORT 2)

Setting the Module Address

Before installing the module, set the backplane DIP switches (located adjacent to the card slot in the Series Six rack) to establish which group of eight consecutive input points in the CPU I/O tables will be used by the module. Figure 3.3 illustrates a typical I/O DIP switch set for address 673-680. Table 3.1 shows switch settings for all possible module addresses. Refer to a later section "Running at the DPU Executive Window", to set the I/O CCM module to run at the DPU Executive Window.

a42441

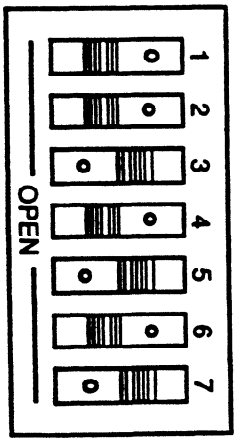


Figure 3.3 TYPICAL I/O BACKPLANE DIP SWITCH

Table 3.1 SETTING THE BACKPLANE DIP SWITCH TO ADDRESS THE I/O CCM

b40478

DPREQ REGISTER CONTENTS DECIMAL (HEX) *	I/O POINT	DIP SWITCH POSITION 7	6	5	4	3	2	1
1001 (03E9)	1- 8							X
1009 (03F1)	9- 16						X	
1017 (03F9)	17- 24						X	X
1025 (0401)	25- 32					X		
1033 (0409)	33- 40					X		X
1041 (0411)	41- 48					X	X	
1049 (0419)	49- 56					X	X	X
1057 (0421)	57- 64					X	X	
1065 (0429)	65- 72				X			X
1073 (0431)	73- 80				X		X	
1081 (0439)	81- 88				X		X	X
1089 (0441)	89- 96				X		X	
1097 (0449)	97- 104				X	X		X
1105 (0451)	105- 112				X	X		
1113 (0459)	113- 120				X	X	X	X
1121 (0461)	121- 128				X	X		
1129 (0469)	129- 136			X				X
1137 (0471)	137- 144			X			X	
1145 (0479)	145- 152			X			X	X
1153 (0481)	153- 160			X			X	
1161 (0489)	161- 168			X		X		X
1169 (0491)	169- 176			X		X		
1177 (0499)	177- 184			X		X	X	X
1185 (04A1)	185- 192			X		X	X	
1193 (04A9)	193- 200			X	X			X
1201 (04B1)	201- 208			X	X			
1209 (04B9)	209- 216			X	X		X	X
1217 (04C1)	217- 224			X	X		X	
1225 (04C9)	225- 232			X	X	X		X
1233 (04D1)	233- 240			X	X	X		
1241 (04D9)	241- 248			X	X	X	X	X
1249 (04E1)	249- 256			X	X	X	X	
1257 (04E9)	257- 264		X					X
1265 (04F1)	265- 272		X				X	
1273 (04F9)	273- 280		X				X	X
1281 (0501)	281- 288		X				X	
1289 (0509)	289- 296		X			X		X
1297 (0511)	297- 304		X			X		
1305 (0519)	305- 312		X			X	X	X
1313 (0521)	313- 320		X			X	X	
1321 (0529)	321- 328		X		X			X
1329 (0531)	329- 336		X		X			
1337 (0539)	337- 344		X		X		X	X
1345 (0541)	345- 352		X		X		X	
1353 (0549)	353- 360		X		X	X		X
1361 (0551)	361- 368		X		X	X		
1369 (0559)	369- 376		X		X	X	X	X
1377 (0561)	377- 384		X		X	X	X	
1385 (0569)	385- 392		X	X				X
1393 (0571)	393- 400		X	X				
1401 (0579)	401- 408		X	X			X	X
1409 (0581)	409- 416		X	X			X	
1417 (0589)	417- 424		X	X		X		X
1425 (0591)	425- 432		X	X		X		
1433 (0599)	433- 440		X	X		X	X	X
1441 (05A1)	441- 448		X	X		X	X	
1449 (05A9)	449- 456		X	X	X			X
1457 (05B1)	457- 464		X	X	X			
1465 (05B9)	465- 472		X	X	X		X	X
1473 (05C1)	473- 480		X	X	X		X	
1481 (05C9)	481- 488		X	X	X	X		
1489 (05D1)	489- 496		X	X	X	X		X
1497 (05D9)	497- 504		X	X	X	X	X	

DPREQ REGISTER CONTENTS DECIMAL (HEX) *	I/O POINT	DIP SWITCH POSITION 7	6	5	4	3	2	1
1505 (05E1)	505- 512		X	X	X	X	X	X
1513 (05E9)	513- 520	X						
1521 (05F1)	521- 528	X						X
1529 (05F9)	529- 536	X					X	
1537 (0601)	537- 544	X					X	X
1545 (0609)	545- 552	X				X		
1553 (0611)	553- 560	X				X		X
1561 (0619)	561- 568	X				X	X	
1569 (0621)	569- 576	X				X	X	X
1577 (0629)	577- 584	X			X			
1585 (0631)	585- 592	X			X			X
1593 (0639)	593- 600	X			X		X	
1601 (0641)	601- 608	X			X		X	X
1609 (0649)	609- 616	X			X	X		
1617 (0651)	617- 624	X			X	X		X
1625 (0659)	625- 632	X			X	X	X	
1633 (0661)	633- 640	X			X	X	X	X
1641 (0669)	641- 648	X		X				
1649 (0671)	649- 656	X		X				X
1657 (0679)	657- 664	X		X			X	
1665 (0681)	665- 672	X		X			X	X
1673 (0689)	673- 680	X		X		X		
1681 (0691)	681- 688	X		X		X		X
1689 (0699)	689- 696	X		X		X	X	
1697 (06A1)	697- 704	X		X		X	X	X
1705 (06A9)	705- 712	X		X	X			
1713 (06B1)	713- 720	X		X	X			X
1721 (06B9)	721- 728	X		X	X		X	
1729 (06C1)	729- 736	X		X	X		X	X
1737 (06C9)	737- 744	X		X	X	X		
1745 (06D1)	745- 752	X		X	X	X		X
1753 (06D9)	753- 760	X		X	X	X	X	
1761 (06E1)	761- 768	X		X	X	X	X	X
1769 (06E9)	769- 776	X	X					
1777 (06F1)	777- 784	X	X					X
1785 (06F9)	785- 792	X	X				X	
1793 (0701)	793- 800	X	X				X	X
1801 (0709)	801- 808	X	X			X		
1809 (0711)	809- 816	X	X			X		X
1817 (0719)	817- 824	X	X			X	X	
1825 (0721)	825- 832	X	X			X	X	X
1833 (0729)	833- 840	X	X		X			
1841 (0731)	841- 848	X	X		X			X
1849 (0739)	849- 856	X	X		X		X	
1857 (0741)	857- 864	X	X		X		X	X
1865 (0749)	865- 872	X	X		X	X		
1873 (0751)	873- 880	X	X		X	X		X
1881 (0759)	881- 888	X	X		X	X	X	
1889 (0761)	889- 896	X	X		X	X	X	X
1897 (0769)	897- 904	X	X	X				
1905 (0771)	905- 912	X	X	X				X
1913 (0779)	913- 920	X	X	X			X	
1921 (0781)	921- 928	X	X	X			X	X
1929 (0789)	929- 936	X	X	X		X		
1937 (0791)	937- 944	X	X	X		X		X
1945 (0799)	945- 952	X	X	X		X	X	
1953 (07A1)	953- 960	X	X	X		X	X	X
1961 (07A9)	961- 968	X	X	X	X			
1969 (07B1)	969- 976	X	X	X	X			X
1977 (07B9)	977- 984	X	X	X	X		X	
1985 (07C1)	985- 992	X	X	X	X		X	X
1993 (07C9)	993- 1000	X	X	X	X	X		

* For programming use only: add 1000 to I/O points.

[X] = Switch in OPEN Position (Depressed to the Left).

Configuring the Communications Ports

Set the DIP switch banks A, B, and C (Figure 3.1 items 2, 3 and 4) on the module to the required configurations (see Tables 3.2, 3.3, and 3.4).

Table 3.2 CONFIGURATION SWITCHES FOR PORT 1 (BANK A)

FUNCTION	SWITCH			O=OPEN C=CLOSED
Data Rate Selection	1	2	3	
110 bps	O	O	O	
300 bps	C	O	O	
600 bps	O	C	O	
1200 bps	C	C	O	
2400 bps	O	O	C	
4800 bps**	C	O	C	
9600 bps	O	C	C	
19.2 Kbps*	C*	C*	C*	
Protocol Selection	4	5	6	
CCM Master RS-232/RS-422	O	O	O	
CCM Master Current Loop**	C	O	O	
CCM Slave RS-232/RS-422	O	C	O	
CCM Slave Current Loop**	C	C	O	
CCM Peer RS-232/RS-422*	O*	O*	C*	
CCM Peer Current Loop**	C	O	C	
RTU Slave RS-232/RS-422	O	C	C	
RTU Slave Current Loop**	C	C	C	
Parity Selection	7	8		
No parity	O	O		
No parity	C	O		
Odd parity*	O*	C*		
Even parity	C	C		

* Indicates the factory-set default position.
** Maximum data rate for current loop operation is 4800 bps.

Table 3.3 CONFIGURATION SWITCHES FOR PORT 2 (BANK B)

FUNCTION	SWITCH			O=OPEN C=CLOSED
Data Rate Selection	1	2		
300 bps	O	O		
1200 bps	C	O		
9600 bps	O	C		
19.2 Kbps*	C*	C*		
Protocol Selection	3	4	5	
CCM Master RS-232D	O	O	O	
CCM Master RS-422	C	O	O	
CCM Slave RS-232D	O	C	O	
CCM Slave RS-422	C	C	O	
CCM Peer RS-232D*	O*	O*	C*	
CCM Peer RS-422	C	O	C	
RTU Slave RS-232D	O	C	C	
RTU Slave RS-422	C	C	C	
Turn Around Delay for CCM and RTU	6			
0 msec*	O*			
500 msec	C			
Parity Selection	7			
No parity	O			
Odd parity*	C*			
Module Operation	8			
Execute I/O CCM operational sw*	O*			
Execute factory test software	C			
Reset Switch				
I/O CCM module is enabled*	O*			
I/O CCM module is reset	C			

* Indicates the factory-set default position.

Table 3.4 CONFIGURATION SWITCHES FOR PORT 1 (BANK C)

FUNCTION	SWITCH	O=OPEN C=CLOSED
RS-232D Operation	1	
Disconnects Pins 15, 16 for Port 1 RS-232D	0	
Connects Pins 15 and 16 for Port RS-232D operation (use external jumper if desired across pins 15-16).	C*	

* Factory-set default position.

POSITIONING THE I/O CCM IN THE RACK

The I/O CCM may be installed in any of the I/O slots of the Series Six PLC, or in a High Capacity I/O rack. Use the extraction/insertion tool to position the module in the rack.

Guide the faceplate over the circuit board so that proper contact is made. Then secure the faceplate to the rack using the thumbscrews at the top and the bottom of the faceplate.

CABLE CONFIGURATION

Cable wiring for the I/O CCM will vary depending upon the desired configuration. A few of the more common applications are shown in the figures on the following pages.

General guidelines for cable construction are as follows:

- At short distances (under 1000 feet) almost any twisted shielded pair will work. The recommended cables will provide reliable operation at data rates up to 19.2 Kbps and distances up to 4000 feet.
- Good wiring practices must be observed. Twisted pairs must be matched so that both transmit signals make up one twisted pair and both receive signals make up the other twisted pair. If this is ignored, then cross-talk can result from the mis-matching which may affect the performance of the communication system.
- The transmitter and receiver signal ground should be within a few millivolts of each other for RS-422 communication.

WARNING

Verify that the RS-422 transmitter and receiver ground are within a few millivolts of each other or damage to the transmitter and receiver may result.

- When routing communication cables outdoors transient suppression devices should be used to reduce the possibility of damage due to lightning or static discharge.

  Best results have been obtained with General Semiconductor Industries Transzorb SA series wired from each signal line to earth ground at both ends of the cable.

CABLE SPECIFICATIONS

Table 3.5 RS-232D/RS-422 CABLE SPECIFICATIONS

Maximum Length	50 feet (15 meters) for RS-232D 4000 feet (1.2Km) for RS-422 1,000 feet (305 meters) for current loop.
Overall Shield Recommended	
24 AWG Minimum	
Mating connector to Port 1 or Port 2 is a D-Subminiature Type. Cannon DB25P (Solder Pot) with DB11096B-3 Hood or Equivalent. (Standard RS-232D male connector)	
The following cables provide acceptable operation at data rates up to 19.2 Kbps and distances up to 4000 feet. Belden - 9184 Belden - 9302 NEC - 222P1SLCBT	

PORT CHARACTERISTICS AND WIRING (J1, J2)

Table 3.6 CONNECTOR PIN-OUT FOR PORTS (J1, J2)

PIN	COMMUNICATION PORT (J1)	COMMUNICATION PORT (J2)
1	NC	NC
2	Data Out RS-232D	Data Out RS-232D
3	Data In RS-232D	Data In RS-232D
4	NC	RTS (RS-232D)
5	NC	CTS (RS-232D)
6	NC	NC
7	Ground	Ground
8	Data Out (+) Current Loop	NC
9	Ground	Ground
10#	Data Out (+) RS-422	Data Out (+) RS-422
11	Data In (+) RS-422	Data In (+) RS-422
12	Current Source (+) Rxd	NC
13	Current Source (+) Txd	NC
14	NC	Output Relay - Normally Closed
15+	RS-232D JMP 1	Output Relay - Normally Open
16+	RS-232D JMP 2	Output Relay - Common
17	Terminate Rxd RS-422	Terminate Rxd RS-422
18	Data In (+) Current Loop	NC
19	Data In (-) Current Loop	NC
20	NC	NC
21	Data Out (-) Current Loop	NC
22#	Data Out (-) RS-422	Data Out (-) RS-422
23	Data In (-) RS-422	Data In (-) RS-422
24	Current Source (-) Rxd	NC
25	Current Source (-) Txd	NC

\+ Optional connection for Port 1 only, switch in DIP bank C can be set to make this connection.

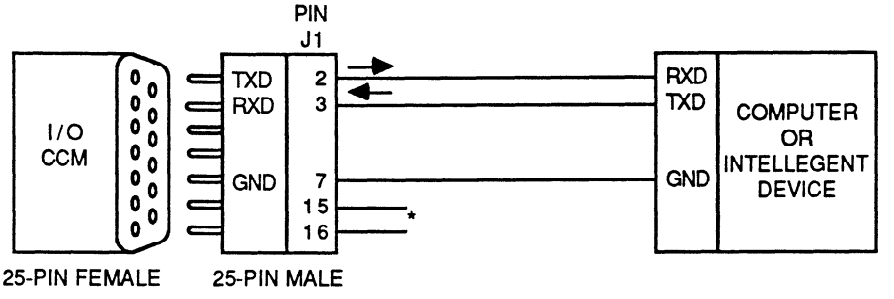
\# RS-422 transmit signals for communications port J2 only are tri-stated for multidrop links when the transmitter is inactive.

CABLE DIAGRAMS

The diagrams that follow include basic RS-232D, RS-422, multidrop, and current loop cable configuration. For more information on RS-232D and RS-422 connections and for connections to the CCM2 or CCM3, refer to Chapter 2 of this manual.

RS-232D Cables

a42700



* PINS 15 AND 16 MUST BE CONNECTED ON PORT 1. THIS MAY BE DONE ON THE CONNECTOR, IN THE CABLE, OR VIA THE DIP SWITCH C SETTING.

Figure 3.4 RS-232D POINT-TO-POINT CONNECTION (PORT 1)

a42722

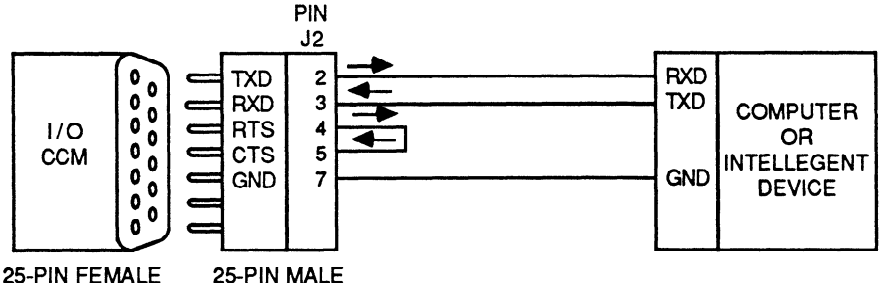


Figure 3.5 RS-232D POINT-TO-POINT CONNECTION (PORT 2)

RS-422 Cables

a42723

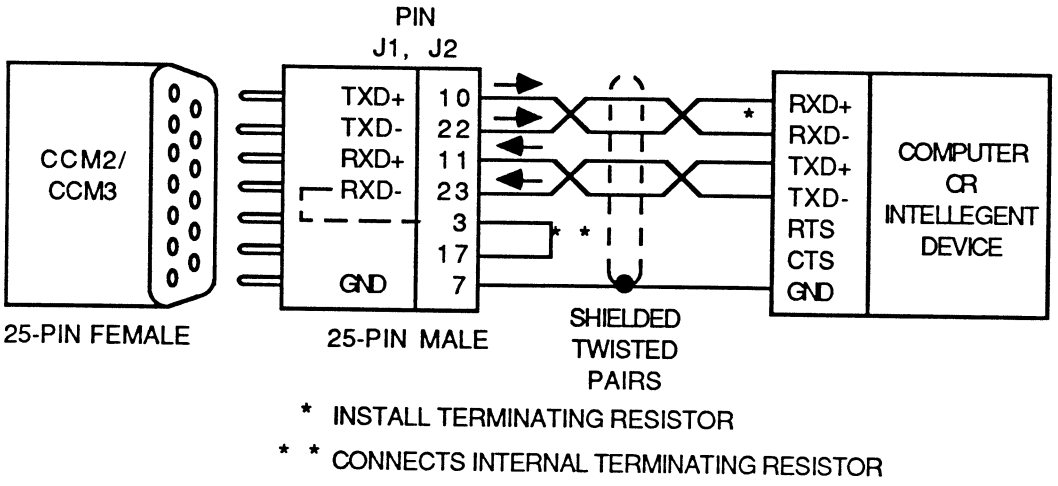


Figure 3.6 RS-422 POINT-TO-POINT CONNECTION

a42724

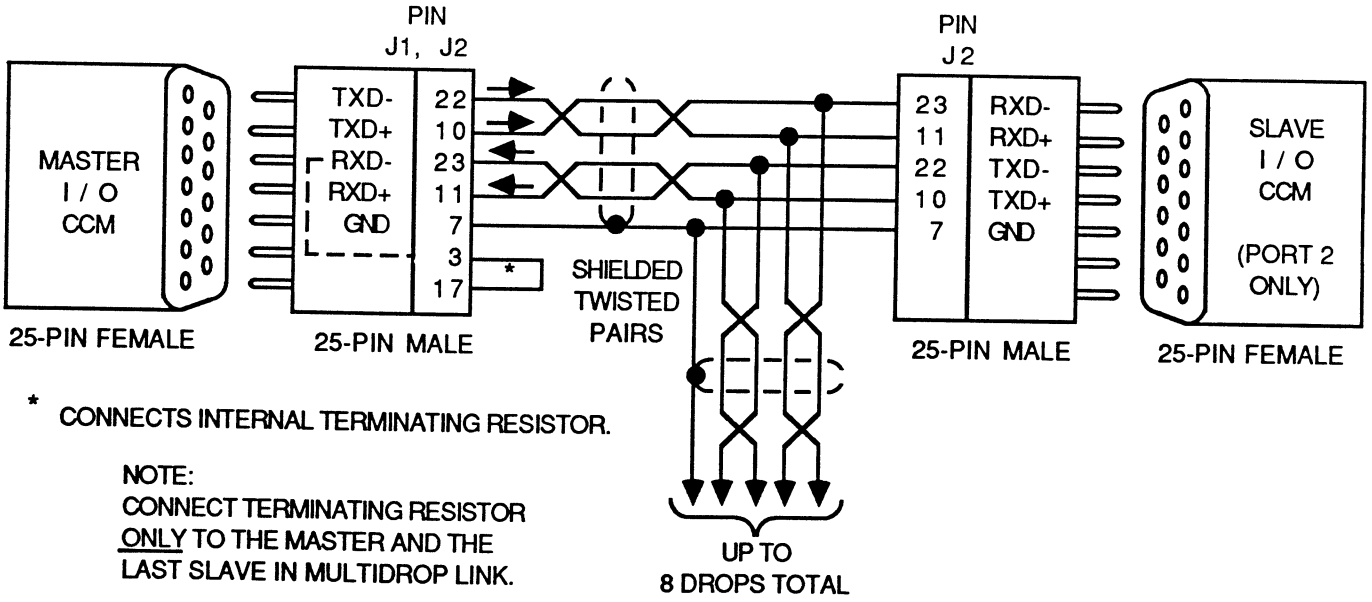


Figure 3.7 RS-422 MULTIDROP CONNECTION

The RS-422 signal nomenclature is cross referenced to the RS-422 EIA standard as follows:

Table 3.7 RS-422 SIGNAL CROSS-REFERENCE TO THE EIA STANDARD

CCM SIGNAL NAME	RS-422 STANDARD SIGNAL NAME
RS-422 out + (TXD+)	B
RS-422 out - (TXD+)	A
RS-422 in + (RXD+)	B'
RS-422 IN - (RXD-)	A'

During a mark condition (logic 1), B will be positive with respect to A.
During a space condition (logic 0), B will be negative with respect to A.

Current Loop Cables

a40529

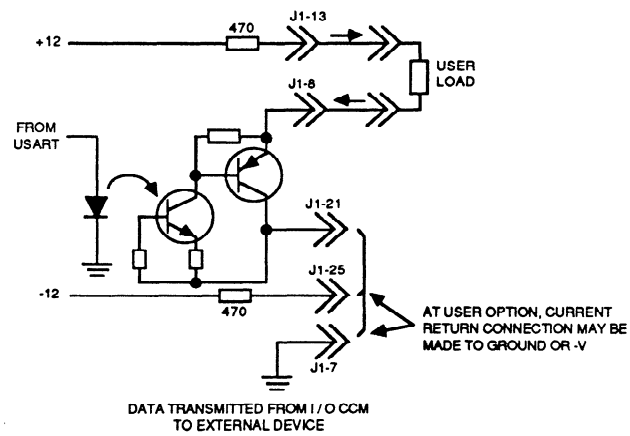


Figure 3.8 ACTIVE CURRENT LOOP DATA TRANSMIT

a42439

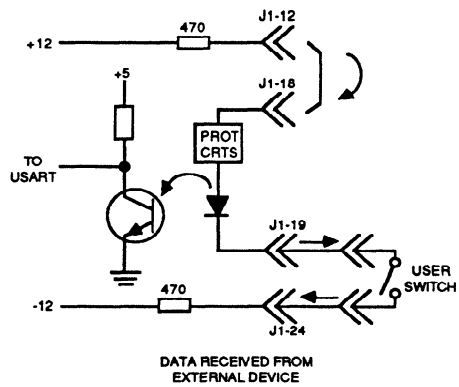


Figure 3.9 ACTIVE CURRENT LOOP DATA RECEIVE

a40530

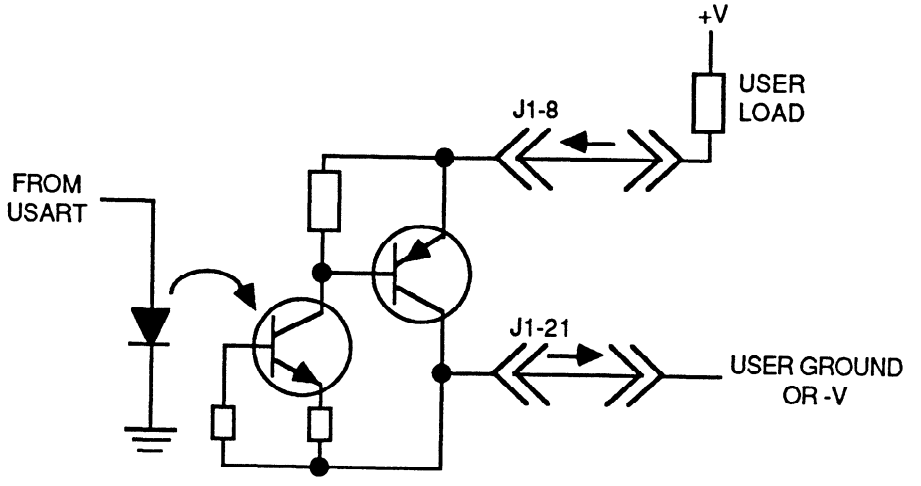


Figure 3.10 PASSIVE CURRENT LOOP DATA TRANSMIT

a42440

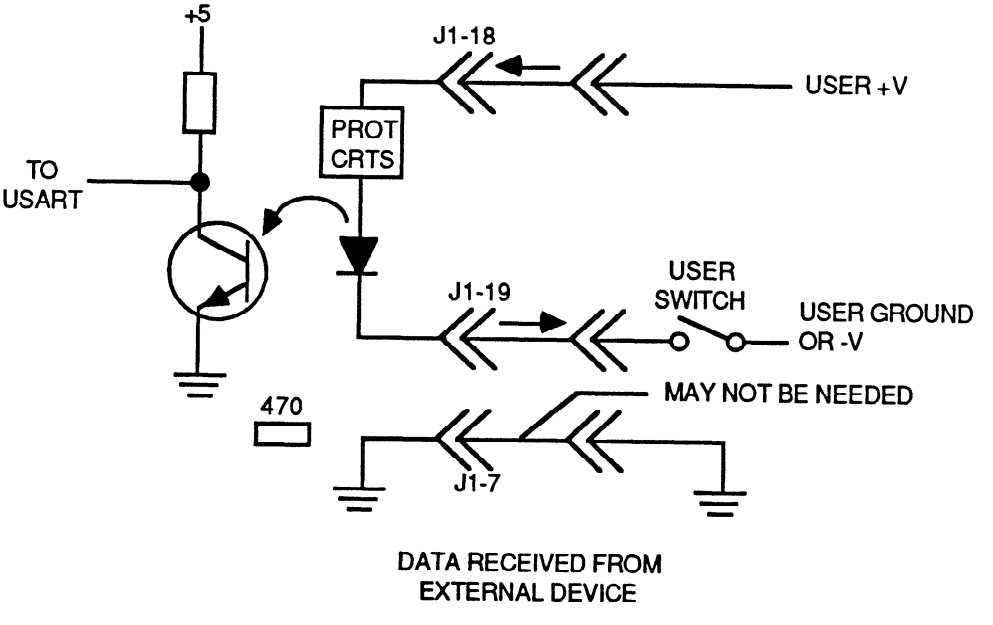


Figure 3.11 PASSIVE CURRENT LOOP DATA RECEIVE

POWERUP AND DIAGNOSTIC TESTING

Power may now be applied to the I/O CCM module and other external devices connected to the ports. After power up diagnostics, the indicator lights should all turn ON.

If the BOARD OK light turns OFF after the power up self-diagnostics routine, the indicator lights will create one of the patterns below:

Table 3.8 LED POWER-UP ERROR CODES (● light on, o light off)

LED	CODE 1	CODE 2	CODE 3	CODE 4	CODE 5	CODE 6	CODE 7
BOARD OK	o	o	o	o	o	o	o
CPU COMM	o	o	o	o	o	o	o
REC 1	o	o	o	o	o	o	●
TRANS 1	o	o	o	o	●	●	●
REC 2	o	o	●	●	o	o	o
TRANS 2	o	●	o	●	o	●	●

Code	Description
1	Processor test failed
2	Timer 0 test failed
3	Timer 1 test failed
4	Timer 2 test failed
5	EPROM test failed
6	RAM test failed (E000-FFFF); board location, U20
7	RAM test failed (C000-DFFF); board location, U19

Table 3.9 LED POWER-UP STATUS INDICATORS DESCRIPTION

LEDs		DESCRIPTION
(1A) BOARD OK	ON	Board has passed self-diagnostics and is operating properly.
	FLASHING	Invalid slave ID when either port is configured as a slave.
	OFF	Board has failed self-diagnostics. See Table 3.8
(1B) CPU COMM	FLASHING	Board is communicating with the Series Six CPU properly. The rate of blink indicates the frequency of CPU communication windows.
	ON	No communication between the Series Six CPU and the board. (Check the backplane DIP switches for the I/O slot and the ladder program if using a DPREQ instruction for window communication.)
(1C) REC 1	ON	Port 1 serial data communications normal.
	FLASHING	Serial data being received on Port 1.
	OFF	Port 1 serial data communications error occurred due to parity errors, bad blocks, or serial link timeout.
(1D) TRANS 1	ON	Port 1 serial data communications normal.
	FLASHING	Serial data being transmitted on Port 1.
	OFF	Port 1 serial data communications error occurred due to parity errors, bad blocks, or serial link timeout.
(1E) REC 2	ON	Port 2 serial data communications normal.
	FLASHING	Serial data being received on Port 2.
	OFF	Port 2 serial data communications error occurred due to parity errors, bad blocks, or serial link timeout.
(1F) TRANS 2	ON:	Port 2 serial data communications normal.
	FLASHING	Serial data being transmitted on Port 2.
	OFF	Port 2 serial data communications error occurred due to parity errors, bad blocks, or serial link timeout.

PROGRAMMING THE I/O CCM

This section describes the two methods of generating window communications between the I/O CCM and the CPU:

- DPREQ Windows
- DPU Executive Window

PROGRAMMING THE DPREQ

The ladder logic program grants communication windows to the I/O CCM through the programmed DPREQ or WINDOW instruction. The ladder logic programs initiates serial data transfers to another device by loading a command into the I/O CCM command registers.

- Program the [DPREQ] or [WINDOW] instruction to establish windows between the I/O CCM and the CPU. The [WINDOW] instruction is valid for CPU microcode Version 130 and thereafter.

- Program the registers containing the communications command and parameters for the required transfer of data if the I/O CCM is to initiate communications.

Establishing I/O CCM to CPU Communications Windows

The CPU provides a window to the I/O CCM using the DPREQ instruction (or WINDOW instruction) as shown below. When properly programmed, the CPU COMM LED will start blinking to indicate that windows are occurring.

An example ladder logic rung for programming the DPREQ instruction is as follows:

```
Oxxxx   Rnnnn                    Oyyyy
-] [---[DPREQ]-------------------(  )
        HHHH
```

In this program, the I/O CCM will receive a CPU communications window if output Oxxxx is on. The contents of register Rnnnn must correspond to the first I/O point address of the I/O CCM plus 1000 decimal. If the I/O CCM address is for inputs 1-8, then HHHH equals 03E9H (decimal 1001).

When the I/O CCM services the CPU communications window without fault, output Oyyyy will remain off. If a fault occurs during the CPU communication window, Oyyyy will turn on.

The I/O CCM does not process serial transfers until the first window is received after the module has powered up. The module needs the first window to determine the CPU ID number and the CPU register and user logic size.

The CPU COMM LED blink rate will show the frequency of DPREQ windows. The LED blinking means that the module detects that the window opened and closed successfully. (The module may or may not have transferred data during that window).

The frequency of DPREQ windows to the I/O CCM module affects the performance (time to complete a message) of the serial links. Therefore, the user should guarantee that the module receives windows on a regular and timely basis. For the fastest response times on the serial link, the module can be given a window once per scan or even multiple windows per scan.

The I/O CCM has a 5-second timeout on waiting for a window to transfer data to or from the Series Six CPU. If the timeout occurs, the I/O CCM will abort the serial link communication (sends an EOT or an error response).

RUNNING AT THE DPU EXECUTIVE WINDOW

With the enhanced I/O CCM (Version 203 Hex, or thereafter), it is possible to get Data Processing Unit (DPU) windows without having a DPREQ in the ladder logic. This feature allows program uploads and downloads while the CPU is stopped.

The following steps are required to set-up the I/O CCM to run at the DPU address.

1. Power-down the unit.

2. Set the backplane DIP switch for Inputs 1009-1016 to be addressed (7E hexadecimal). (Switch 1 CLOSED, all other switches OPEN -- Refer to Figure 3.12)

a42729

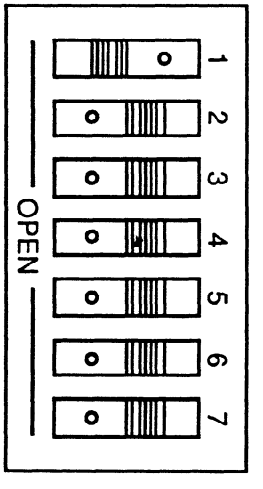


Figure 3.12 BACKPLANE DIP SWITCH SETTING FOR RUNNING AT DPU WINDOW

3. Connect the I/O terminator plug. (Reference "I/O Terminator Plug")

4. Power-up the unit.

I/O Terminator Plug (DPU)

A special I/O terminator plug may be required when operating the I/O CCM module at the DPU Executive Window. The I/O Terminator Plug requirement is dependent upon: whether the I/O CCM is placed in a CPU rack or an I/O rack, and whether type IOI4 or IOI5, I/O Controller, card is installed in the CPU rack.

Installing the I/O CCM in a CPU Rack

When the I/O CCM is installed in a CPU rack (e.g., Series Six Plus or Series 60) along with the IOI4 card, an I/O terminator plug (wired as show below) must be used.

Pin No.	Signal		Jumper Connection
30	FIN+	<----------+	Pins: 30, 35, 37
31	FIN-	<----------|----+	Pins: 31, 34, 36
35	DPE+	<----------+ |	
36	DPE-	<----------|----+	
34	+5V	<----------|----+	
37	GND	<----------+	

Figure 3.13 I/O TERMINATOR PLUG (CPU RACK)

Position the 37-pin, male, connector plug on the I/O port of the I/O Controller (IOI4) card in the Series Six CPU -- Slot 1.

Installing the I/O CCM in an I/O Rack

When the I/O CCM is installed in an I/O rack along with an IOI4 card in the CPU rack, the I/O terminator plug (wired as shown below) must be used. (Verify that the CPU connector end of the I/O cable is wired as shown below -- a wiring modification may be required).

Pin No.	Signal		Jumper Connection
35	DPE+	<----------+	Pins: 34, 36
			Pins: 35, 37
36	DPE-	<----------|----+	
34	+5V	<----------|----+	
37	GND	<----------+	

Figure 3.14 I/O TERMINATOR PLUG (I/O RACK)

A jumper setting for the IOI4 card is also required -- Position Jumper ABC in A-B position.

Installing the I/O CCM with I/O Controller (IOI5)

When the I/O CCM is installed in either a CPU rack or I/O rack along with the I/O Controller (IOI5) card, the I/O terminator plug is NOT required.

Position the IOI5 card jumper (Jumper ABCK) in A-K position.

COMMUNICATIONS COMMAND AND PARAMETER REGISTERS

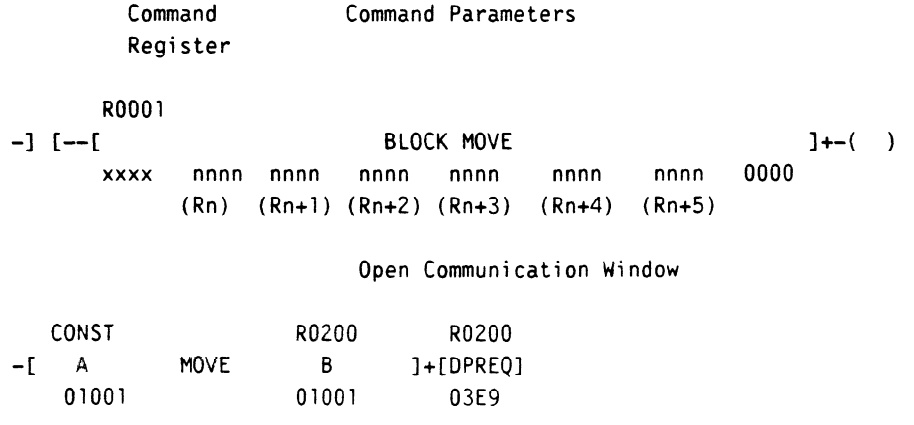
Each I/O CCM has an associated communications command register. This register is monitored by the I/O CCM for communication commands which the user program wants to initiate. The command register corresponds to the register number of the first input point of the module address. For example, if the I/O CCM is addressed (using the backplane DIP switches) at Inputs 9-16, then the communications command register in the Series Six CPU is Register 9.

The format of these commands and the command parameters is the same as for the CCM2 and CCM3. The main difference is that for the I/O CCM, the command register reference must always correspond to the module address. Therefore, if using the DPREQ windows, the reference for the DPREQ register must not be the same as the command register reference. If running at the DPREQ Executive Window, the DPREQ instruction is not required for serial communications.

When the user sets up one of these commands for execution, the I/O CCM will read the communications command number and the command parameters. It will then zero the communications command register to notify the user that the command was read by the I/O CCM. The I/O CCM status byte indicates when the command is in progress and when the command has completed.

In the example below, the module is addressed for I/O points 1-8 (01001 dec or 03E9 hex). The CPU communications window is opened once each scan. The example below shows the logic necessary to initiate a serial request using the BLOCK MOVE function in the Series Six CPU.

Refer to *Chapter 2,* for definitions of the command and parameter registers, and for programming examples.



Command Register for DPU Executive Window

The command register to be used when operating the I/O CCM at the DPU Executive Window is R1009 (3F1 hex). This corresponds to Input/Output points I1009-1016 that are translated from the DIP switch position 7E (Not shown in Table 3.1)

NOTE

This address is valid only for the I/O CCM module.

I/O CCM STATUS BYTE

The eight input points in the Series Six CPU which correspond to the address of the I/O CCM module are used to provide the CPU with the status of the module. The I/O CCM status byte has the same format as the CCM status bytes and is updated in the same way as the CCM status bytes. The module guarantees that the pulsed status bits will be pulsed a minimum of three windows.

DPU Executive Windows

When running at the DPU Executive Window the I/O CCM status byte is located at Input locations I0993 - I1000. In this way, the I/O CCM status byte will not be in conflict with the CCM2/3 status byte.

EXPANDED MEMORY MAPPING

Expanded Memory Mapping is a feature in later versions of the Series Six PLC Communications Control (CCM2, CCM3 and I/O CCM) modules . Only a brief listing of the features of the CCM expanded memory mapping is given in this section. Refer to Appendix B, Expanded Functions for detailed information concerning the CCM expanded memory mapping.

- CCM module hardware and software identification
- Expanded programming information
- Expanded I/O Reference
- Expanded User Memory Reference
- Single Bit Write
- Programmable Timeouts and Retries

OPERATIONAL INFORMATION

I/O CCM operational information which may be of interest to users familiar with CCM is listed below.

1. An external device can perform program uploads and downloads using the enhanced I/O CCM module firmware.

   When using the I/O CCM module firmware (Version 203 Hex, or later) uploads and downloads may be performed when the I/O CCM is placed at locations I/O 1009-1016.

2. The user is not restricted from executing CCM protocol functions to write to memory areas which might stop the Series Six CPU (i.e., subroutine vector addresses and User Logic). This could result in error conditions in the I/O CCM. The I/O CCM receives windows from the CPU only if the CPU is running if it does not use the DPU executive window.

3. The software version number as read from Diagnostic Status Word 12 for the I/O CCM starts with 512 (200H) and increments by one (1) for each revision. This relates to the CCM2 and CCM3 as follows:

Board	Diagnostic Status Word 12 Software Version # Range
CCM2	1 - 255 (1 - 0FFH)
CCM3	256 - 511 (100H - 1FFH)
I/O CCM	512 - 767 (200H - 2FFH)

4. If a serial protocol error occurs when using the CCM protocol on the I/O CCM, both the Txd and Rxd LEDs for the associated port will turn OFF. When the next successful message is sent or received, the LEDs will turn ON again. The Rxd and Txd LEDs will reflect the reception and transmission of characters.

5. The I/O CCM cannot be configured from registers.

6. The I/O CCM does not perform tape or OIU operations.

7. The I/O CCM does not use a battery.

8. The port 2 relay and RTS are turned on before all serial transmissions on Port 2. The port 2 relay can be heard opening and closing when communications are occurring on port 2; this is normal.

9. The RTU protocol can be selected to use the 500 msec. turn-around delay on the J2 port.

10. The I/O CCM module will check for commands (in the communications command register) between communications with serial devices and continually when idle.

11. The maximum data rate for current loop operation is 4800 bps.

NOTE

If commands are not going to be initiated from the I/O CCM, a value of zero should be placed in the command register. The five successive command parameter registers can then be used as desired.

CHAPTER 4
CCM SERIAL INTERFACE PROTOCOLS

INTRODUCTION TO CCM PROTOCOL

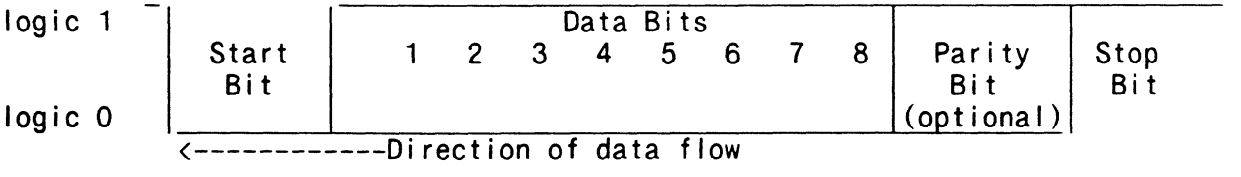
The purpose of this chapter is to provide complete information on CCM protocol and timing to allow the user to write a serial communications driver for a host computer or microprocessor.

Communications Control Module protocol was defined in Chapter 1 as a set of rules governing the establishment of a communications link and the flow of data between a target PLC and a source PLC. In addition, this protocol governs any other communication element in the configuration. If a host computer or control device is to be a part of a system configuration, it must communicate based on CCM protocol.

The CCM is capable of both peer-to-peer and master-slave protocols. The protocol selection for CCM can be made by DIP switches or by using selected CPU registers as explained in the section, Module Configuration, in Chapter 2.

ASYNCHRONOUS DATA FORMAT

Communications Control Module serial interface protocol is based on ANSI Standard X3.28 implementing asynchronous character transfers using an 8-bit binary or ASCII format with optional parity as shown below.



The 8 data bits can contain either ASCII characters or uncoded binary numbers. Parity on the CCM can be specified as either odd or none.

CONTROL CHARACTER CODING

The ASCII control characters used for both peer-to-peer and master-slave protocol are shown below.

Table 4.1 ASCII CONTROL CHARACTERS FOR CCM PROTOCOL

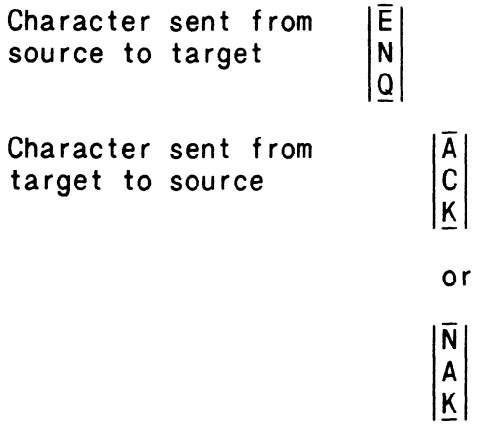
ABBREVIATION	HEX VALUE	MEANING
SOH	01	Start of Header
STX	02	Start of Text
ETX	03	End of Text
EOT	04	End of Transmission
ENQ	05	Enquire
ACK	06	Acknowledge
NAK	15	Negative Acknowledge
ETB	17	End of Block

PEER-TO-PEER PROTOCOL

Peer-to-peer protocol is used in the point-to-point system configuration where only 2 devices share a single communication line. In peer-to-peer protocol either device can initiate a communication. The device initiating the communication is known as the source; the other device, the target. For example, peer-to-peer protocol is used when connecting Series Six™ PLCs to GEnet through the Bus Interface Unit (BIU).

ENQUIRY SEQUENCE

When a device intitiates a communication using peer-to-peer protocol, an enquiry sequence consisting of the ASCII control character, ENQ, is sent on an idle communication line (channel) to the target device. As shown below, if the receiving device is not busy, it responds with the ASCII control character, ACK; if it is busy, with the ASCII control character, NAK. ACK and NAK are the only acceptable responses to ENQ in this mode.



If the target response to a peer enquiry is invalid, the source will delay a short time and retry the enquiry. The source will retry the enquiry 32 times before aborting the communication.

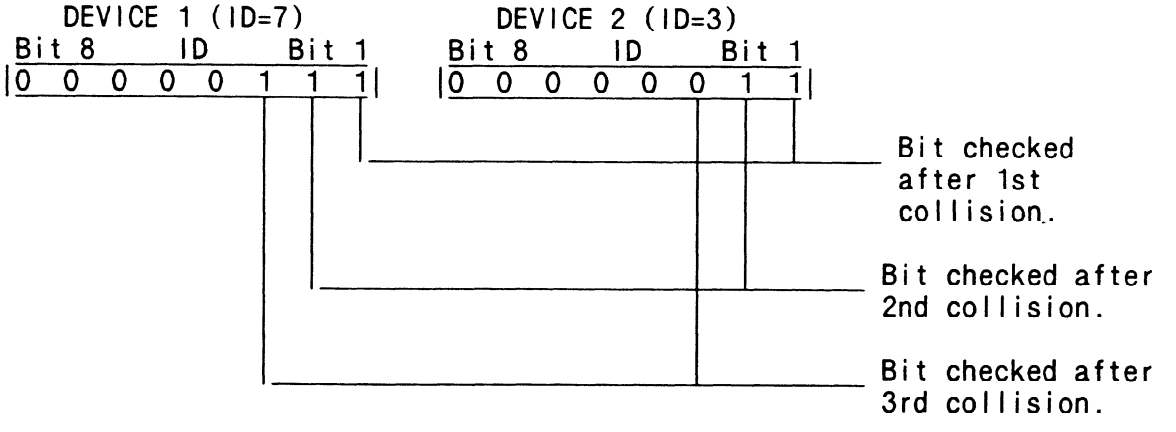
ENQUIRY COLLISION

In peer-to-peer protocol, a collision occurs whenever both devices on the same communication line request communications at the same time. Upon collision, each device will back off an amount of time depending on the data rate and the device's own 8-bit ID number. Each device will begin by checking whether bit 1 of its own ID is 1 or 0. If after any back-off and retry there is another collision, then the next bit of each device is checked. Since device IDs are unique, there will always be at least one pair of corresponding bits that are not the same and which will produce a different back-off delay for each device, thus, preventing a collision on the next retry. The back-off times are shown in Table 4.2.

Table 4.2 BACK-OFF TIMES

DATA RATE	ID BIT=0 (time in millisec)	ID BIT=1 (time in millisec)
300	300	440
600	140	220
1200	80	120
2400	80	120
9600	80	120
19200	80	120
38400	80	120

The illustration below shows the sequence for checking device ID bits after a collision.



PEER-TO-PEER PROTOCOL FORMAT

The general format for a successful communication is shown below. Figure 4.1 shows a data transfer from the source device to the target device and Figure 4.2 shows a data transfer from the target device to the source device. The source device is always the initiator of the request; the target device receives the request.

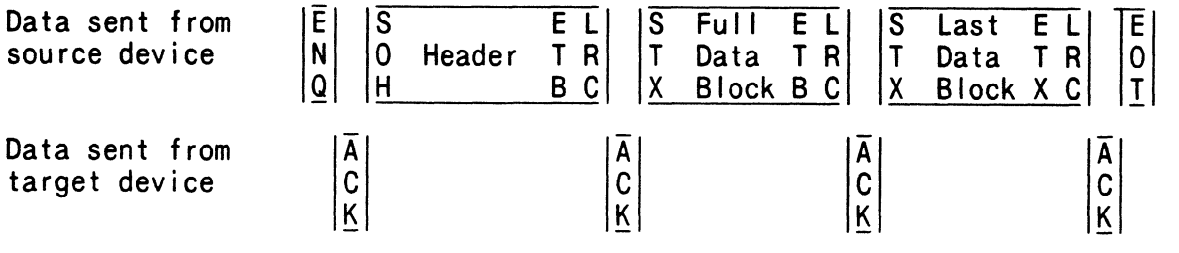


Figure 4.1 DATA TRANSFER FROM SOURCE TO TARGET (PEER-TO-PEER)

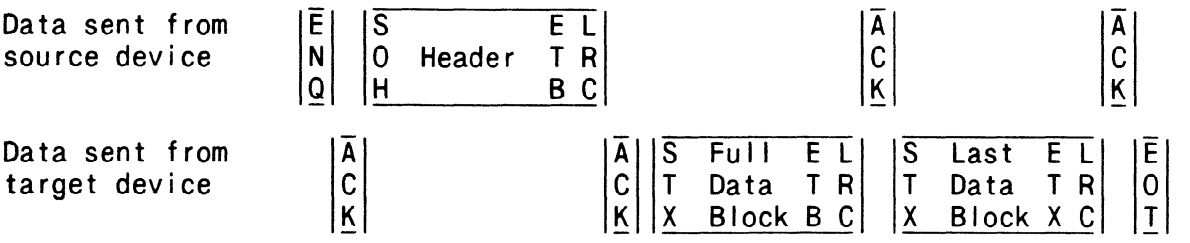


Figure 4.2 DATA TRANSFER FROM TARGET TO SOURCE (PEER-TO-PEER)

PEER-TO-PEER FLOW CHARTS

The general format above explains the protocol sequence only when no errors occur during transmission. To understand the protocol sequence when errors do occur, see the flow charts and accompanying explanation on the following pages.

Peer Request Initiate Sequence, Source Device (See Figure 4.3).

Start request initiate sequence.
Set collision back-off flag to false.
Has ENQ been retried 32 times?
    If YES, send EOT and exit request initiate sequence.
    If NO, send ENQ.
Is collision back-off flag still false?
    If NO, read ENQ response with timeout = 4 char. times + turn around delay.
    Is there a time-out on the response?
        If YES, go to "Read ENQ With Backoff-Delay...".
        If NO, go to "Is Response ACK?".
    If YES, read ENQ response with time-out. (Condition 1, Table 4.5)
    Is there a time-out on response?
        If YES, go to "Set Back-Off Flag to True".
        If NO, go to "Is Response ACK?".
Is Response ACK?
    If NO, is response NAK?
        If YES, increment NAK count and wait for other device to become free (10 msec or turn around delay if it is not 0 msec) and return to "ENQ Retried or NAKed 32 Times?".
        If NO, then set back-off flag to true and read ENQ with time-out based on CPU ID. (See section, Enquiry Collision).
            Is ENQ received?
                If NO, increment ENQ retry count and go to ENQ retry.
                If YES, service other device and act as a target device.
    If YES, send header.
Read response to header.

(Explanation continued on page 4-9)

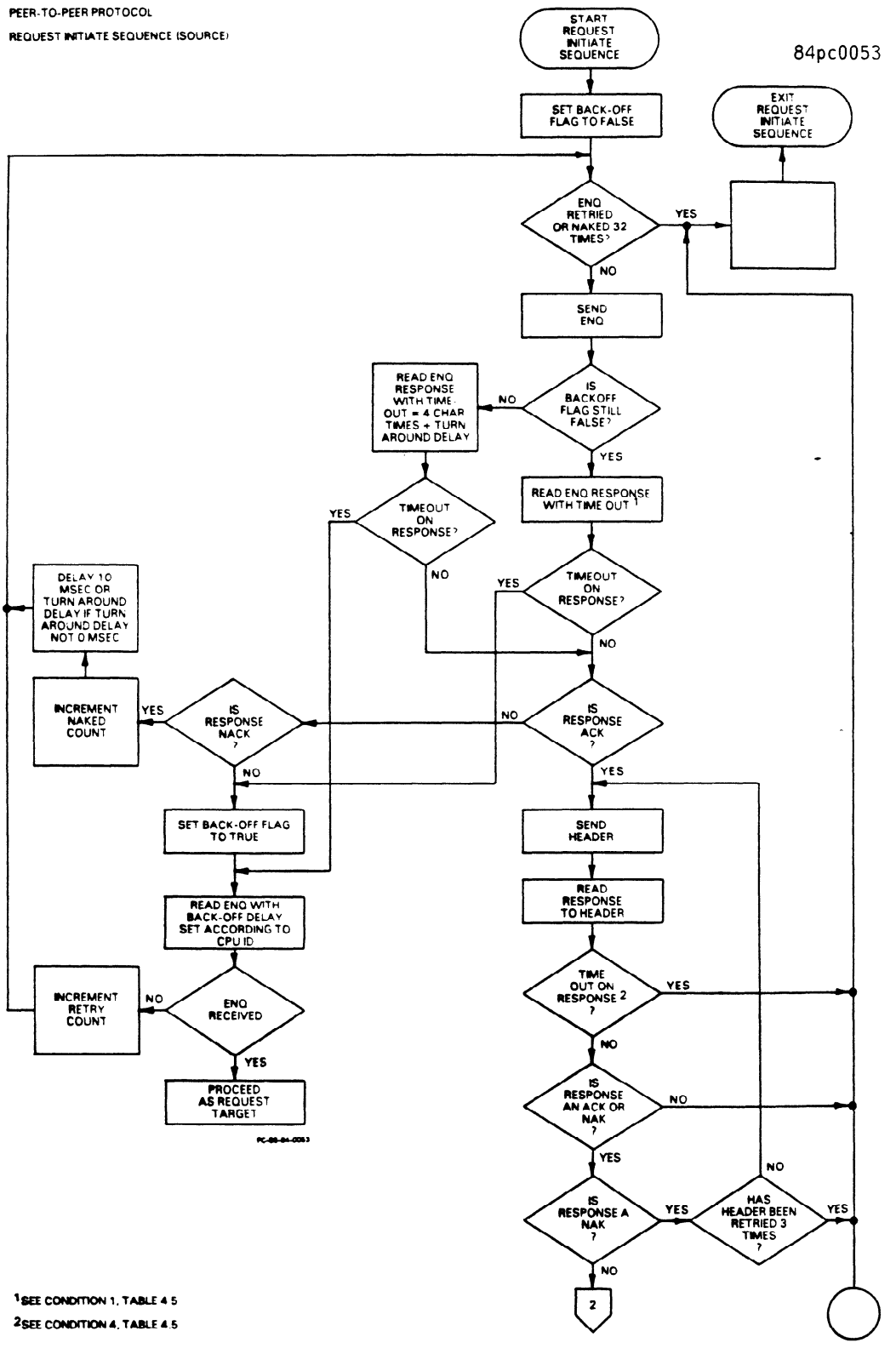
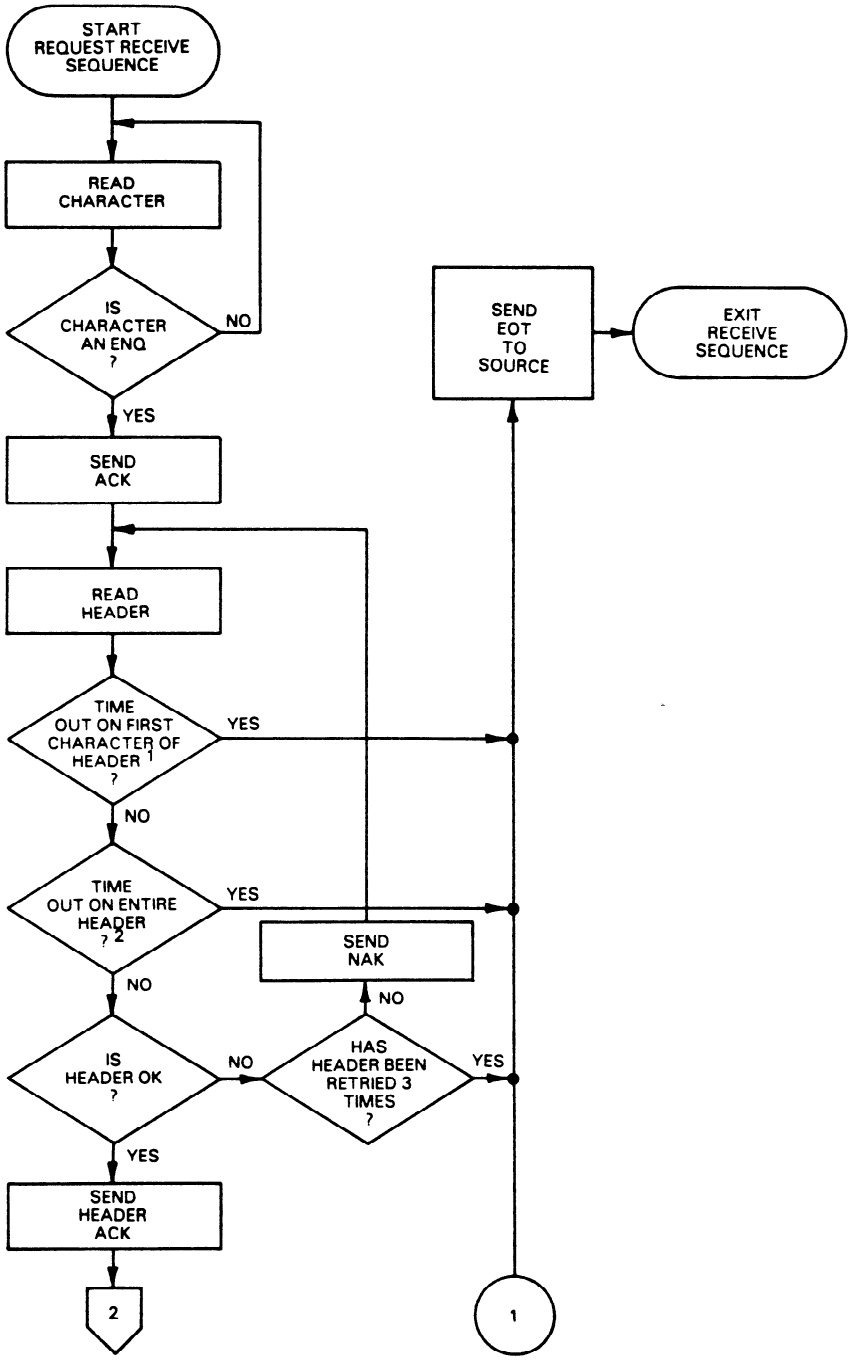


Figure 4.3 PEER REQUEST INITIATE SEQUENCE, SOURCE DEVICE

84pc0054

PEER-TO-PEER PROTOCOL

REQUEST RECEIVE SEQUENCE (TARGET)



[1]SEE CONDITION 2, TABLE 4.5
[2]SEE CONDITION 3, TABLE 4.5

Figure 4.4 PEER REQUEST RECEIVE SEQUENCE, TARGET DEVICE

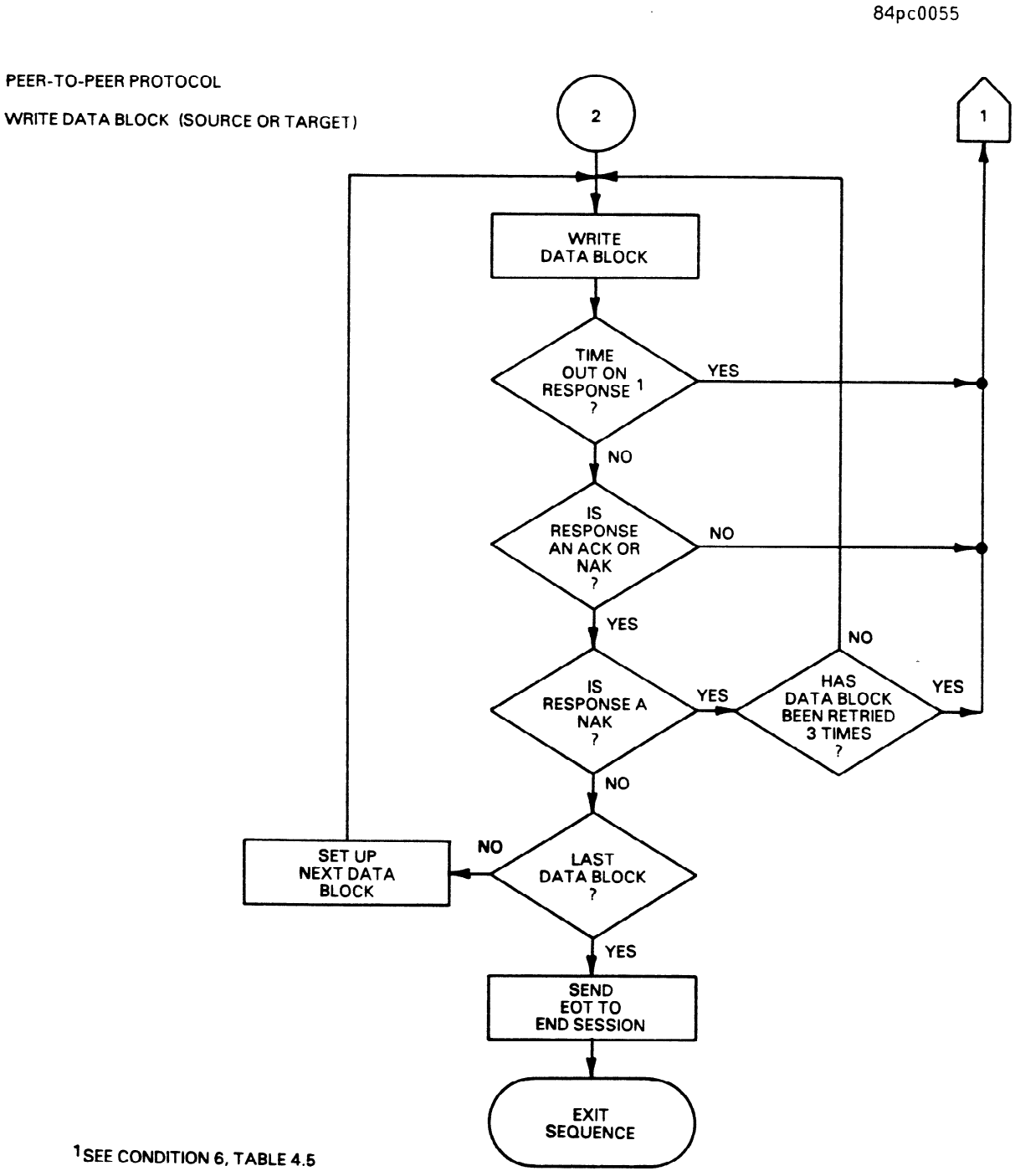


Figure 4.5 PEER WRITE DATA BLOCKS, SOURCE OR TARGET DEVICE

84pc0056

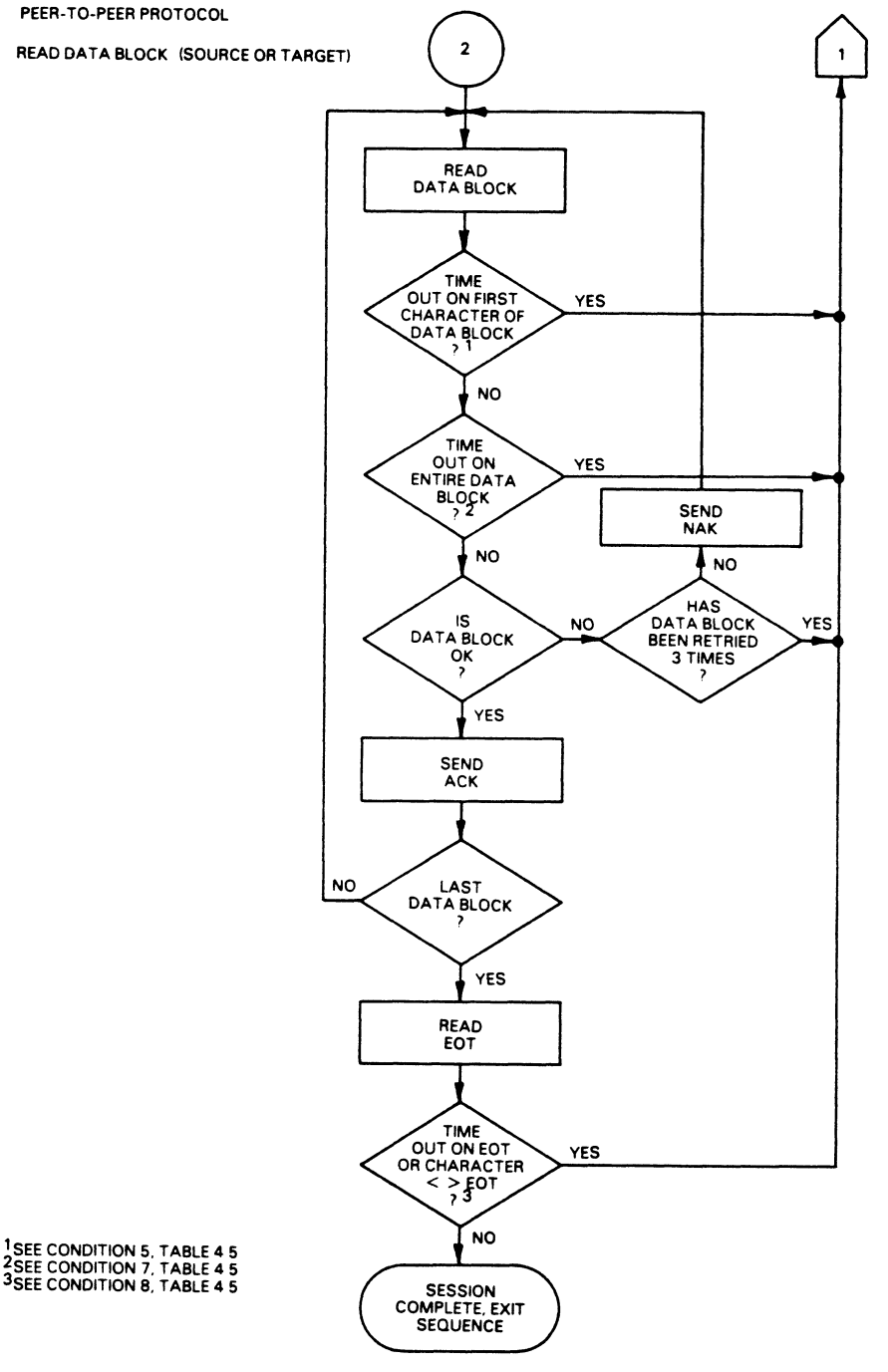


Figure 4.6 PEER READ DATA BLOCKS, SOURCE OR TARGET DEVICE

Is there a time-out on the response? (Condition 4, Table 4.5)
If YES, send an EOT and exit the initiate sequence.
If NO, is response an ACK or NAK?
If not ACK or NAK, send EOT and exit initiate sequence.
If ACK or NAK is it NAK?
If YES, has header been retried 3 times?
If YES, send EOT and exit initiate sequence.
If NO, return to "Send Header".
If NO, go to "Read or Write Data Blocks" depending on the direction of data transfer.

Peer Request Receive Sequence, Target Device (See Figure 4.4).

Read character.
Is character an ENQ?
If NO, go to read character.
If YES, send ACK.
Read header.
Is there a time-out between ENQ response and the first character of the header? (Condition 2, Table 4.5)
If YES, send EOT and exit.
If NO, is there a time-out on entire header? (Condition 3, Table 4.5)
If YES, send EOT and exit.
If NO, is header OK?
If NO, has header been retried 3 times?
If YES send EOT and exit.
If NO, send NAK and return to "Read Header".
If YES, send ACK and go to "Read or Write Data Blocks" depending on the direction of data transfer.

Peer Write Data Blocks, Source or Target Device (See Figure 4.5).

Write data block.
Is there a time-out on the data block response? (Condition 6, Table 4.5)
If YES, send EOT to other device and exit.
If NO, is data block response ACK or NAK?
If not ACK or NAK, send EOT to other device and exit.
If ACK or NAK, is it a NAK?
If YES, has data block been retried 3 times?
If YES, send EOT and exit.
If NO, return to "Write Data Block".
If NO, is it last data block?
If NO, set up next data block and return to "Write Data Block".
If YES, send EOT to end session and exit sequence.

Peer Read Data Blocks, Source or Target Device (See Figure 4.6).

Read data block.
Is there a time-out on the first character of the data block? (Condition 5, Table 4.5)
If YES, send an EOT and exit.
If NO, is there a time-out on the entire data block? (Condition 7, Table 4.5)
If YES, send and EOT and exit.
If NO, is the data block OK?
If NO, has the data block been retried 3 times?
If YES, send EOT and exit.
If NO, send NAK and return to "Read Data Block".
If YES, send ACK.
Is it the last data block?
If NO, return to "Read Data Block".
If YES, read EOT.
Is there a time-out on the EOT or is the character not an EOT? (Condition 8, Table 4.5)
If there is a time-out or character is not EOT, send EOT and exit the sequence.
If EOT is OK, the session is complete. Exit sequence.

MASTER-SLAVE PROTOCOL

Master-slave protocol is typically used in a multidrop system configuration. It can be used, however, in the point-to-point configuration. In master-slave protocol there is one master and one or more slaves. Only the master can initiate communications.

The enquiry sequence for master-slave protocol differs from that for peer-to-peer. In peer-to-peer protocol there are only 2 devices connected to the communication line. When one of the devices initiates the communication, there is only one other device that can be the target, therefore, the enquiry sequence needs no ID for the target. As stated before, in the master-slave protocol there may be more than one slave which can respond to an enquiry sequence. Because of this, in master-slave protocol the enquiry sequence must include the target address for identifying the target device.

There are two forms of master-slave protocol: Normal (N) Sequence and Quick (Q) Sequence. Both forms require that master-slave protocol be selected on the CCM2, CCM3, or I/O CCM module. Q Sequence protocol is used only for serial communications using the CCM commands 06109 or 06209, Read Q Response. All other master-slave serial communications use the Normal Sequence form.

ENQUIRY RESPONSE DELAY

The enquiry response delay is a delay between the receipt of an enquiry sequence from a master and the response by a slave. A delay will exist so that idle slaves, which monitor any active link between the master and a slave will not be confused by enquiry sequences occurring during transmission of the data text. When an idle slave recognizes an apparent enquiry sequence it starts an internal timer of 10 msec plus 4 character times. If any other character is received before the timer times out, the idle slave disregards the enquiry. Therefore, any device transmitting data text on a multidrop link should ensure that there will be no gaps in the text greater than 2 character times so an idle slave will not misinterpret data as an enquiry sequence.

NORMAL SEQUENCE, MASTER-SLAVE

The form of the Normal (N) Enquiry Sequence from the master to the target slave and the response by the target slave is shown as follows:

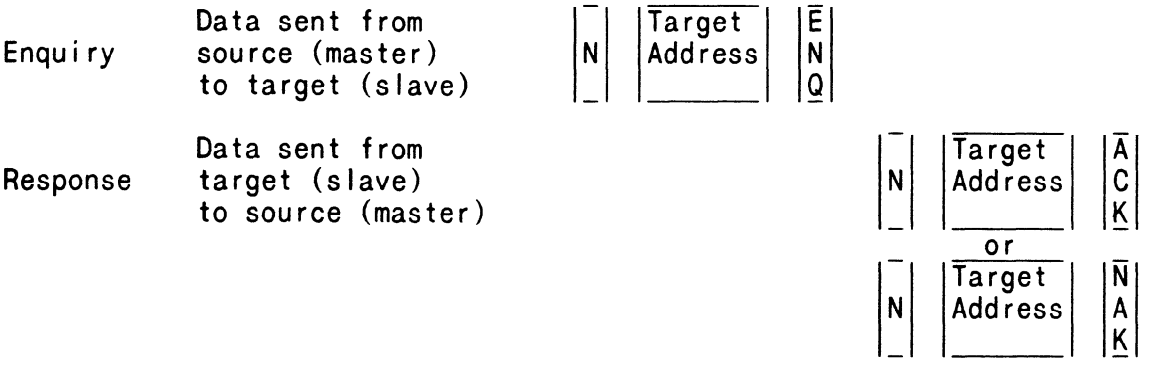


Figure 4.7 NORMAL ENQUIRY SEQUENCE

N : ASCII coded "N" used to specify Normal Sequence operation as opposed to a "Q" for Q Sequence operation.

Target Address : Target address is the target ID number to which the master is attempting communications plus 20H. The target address is a single byte which may have the hexadecimal value 21 through 7A (ASCII "!" through "z").

ENQ : ASCII control character meaning enquire.

ACK or NAK : Response from slave meaning acknowledge or negative acknowledge.

If the slave response to a master enquiry is invalid, the master will delay a short time and retry the enquiry. The master will retry the enquiry 32 times before aborting the communication.

Normal Sequence Protocol Format

The general format for a successful communication is shown below. Figure 4.8 shows a data transfer from the source device to the target device and Figure 4.9 shows a data transfer from the target device to the source device. The source device is always the initiator of the request; the target device receives the request.

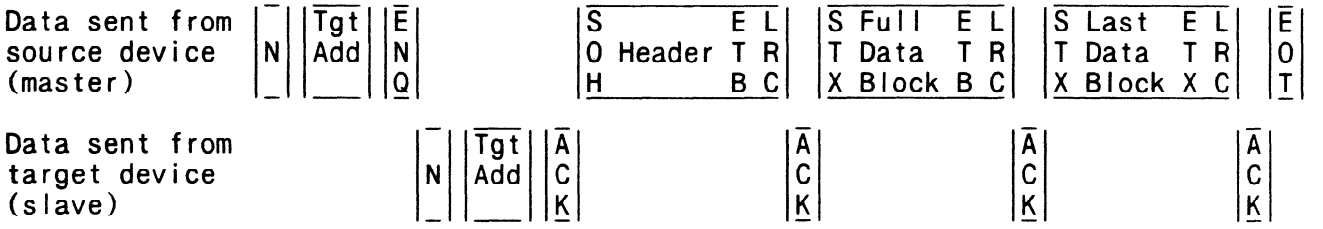


Figure 4.8 DATA TRANSFER FROM MASTER TO SLAVE

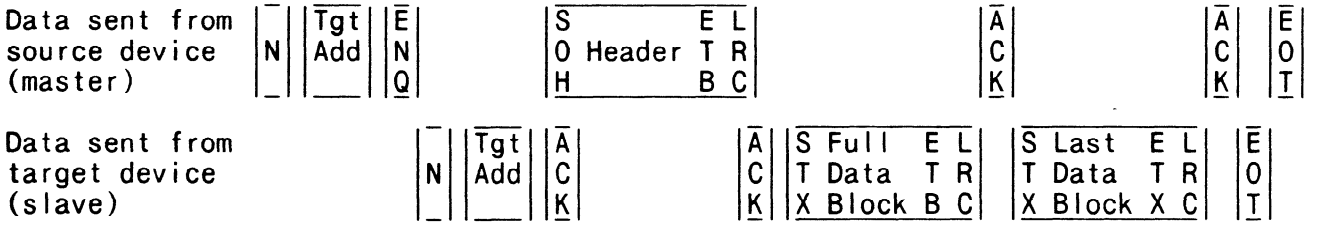


Figure 4.9 DATA TRANSFER FROM SLAVE TO MASTER

Master-Slave Normal Sequence Flow Charts

To fully understand how the protocol operates under error conditions see the flow charts and accompanying explanation.

Normal Sequence, Master (See Figure 4.10)

Start N Sequence.
Start N Enquiry.
Has enquiry been retried 32 times?
    If YES, send EOT to slave and exit N Sequence.
    If NO, send N Enquiry (N, Target Address, ENQ).
Read N Enquiry response.
Is there a time-out or error in response (response not an ACK or a NAK)? (Condition 1, Table 4.5)
    If YES, delay 10 msec or the turn around delay if it is not 0 msec, increment the N Enquiry retry count, and return to "Start N Enquiry".
    If NO, send the header to the slave.
Read response to header.

Is there a time-out on the response? (Condition 4, Table 4.5)
If YES, send an EOT and exit the initiate sequence.
If NO, is response an ACK or NAK?
If not ACK or NAK, send EOT and exit initiate sequence.
If ACK or NAK is it NAK?
If YES, has header been retried 3 times?
If YES, send EOT and exit initiate sequence.
If NO, return to "Send Header".
If NO, go to "Read or Write Data Blocks" depending on the direction of data transfer.

Normal Response, Slave (See Figure 4.11)

Start N Response.
Read N Enquiry.
Is N Enquiry sequence correct?
If NO, return to "Read N Enquiry".
If YES, start timer of 10 msec plus 4 character times.
Is timer done?
If NO, have any characters arrived?
If NO, go to "Is Timer Done?".
If YES, go to "Read N Enquiry".
If YES, send N Enquiry Response.
Read header.
Is there a time-out between ENQ response and the first character of the header? (Condition 2, Table 4.5)
If YES, send EOT and exit.
If NO, is there a time-out on entire header? (Condition 3, Table 4.5)
If YES, send EOT and exit.
If NO, is header OK?
If NO, has header been retried 3 times?
If YES send EOT and exit.
If NO, send NAK and return to "Read Header".
If YES, send ACK and go to "Read or Write Data Blocks" depending on the direction of data transfer.

Write Data Blocks, Master or Slave (See Figure 4.12)

Write data block.
Is there a time-out on the data block response? (Condition 6, Table 4.5)
If YES, send EOT to other device and exit.
If NO, is data block response ACK or NAK?
If not ACK or NAK, send EOT to other device and exit.
If ACK or NAK, is it a NAK?
If YES, has data block been retried 3 times?
If YES, send EOT and exit.
If NO, return to "Write Data Block".
If NO, is it last data block?
If NO, set up next data block and return to "Write Data Block".
If YES, send EOT to end session.

(Explanation continued on page 4-18).

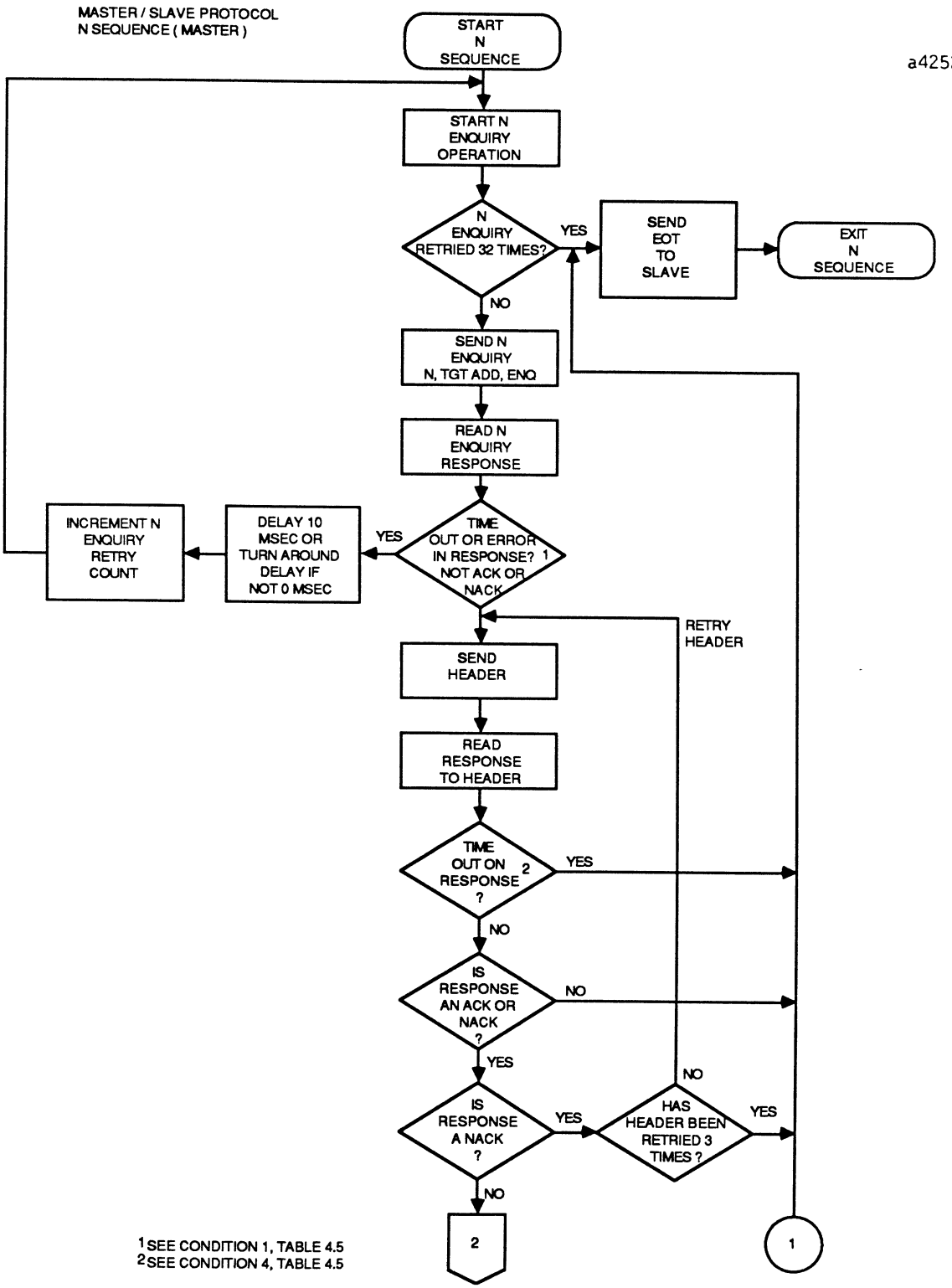


Figure 4.10 N SEQUENCE, MASTER

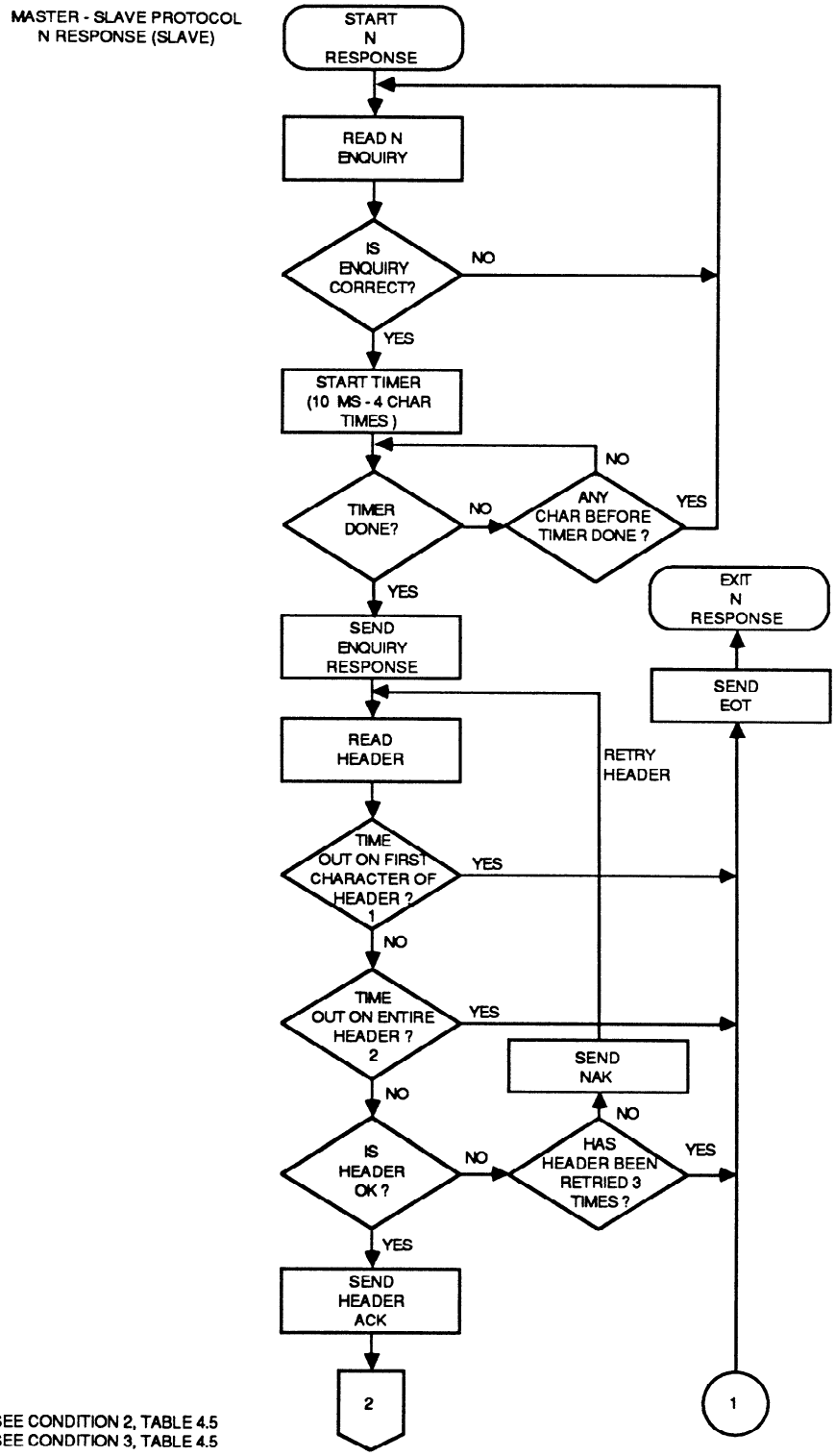


Figure 4.11 N RESPONSE, SLAVE

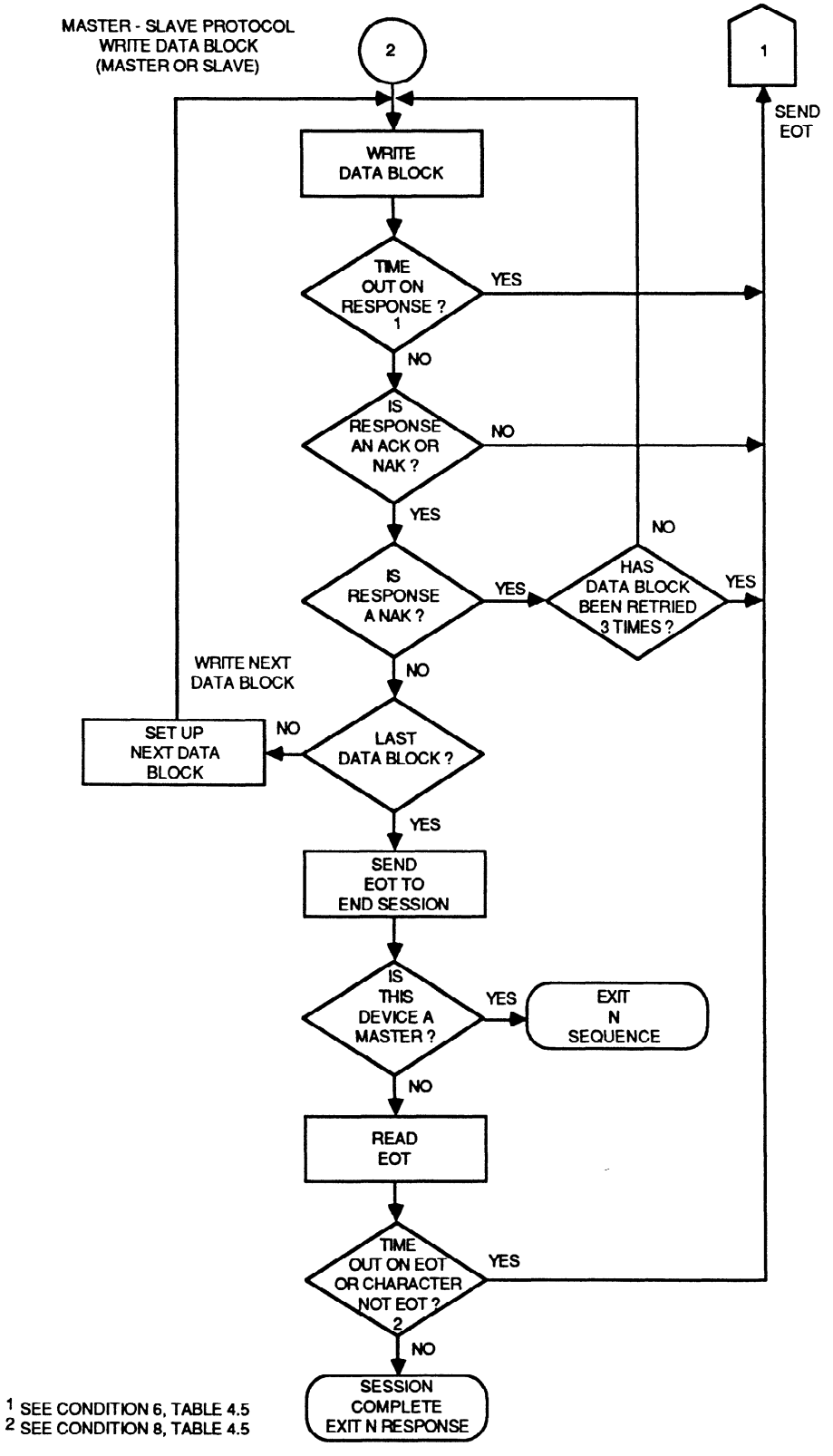


Figure 4.12 WRITE DATA BLOCKS, MASTER OR SLAVE

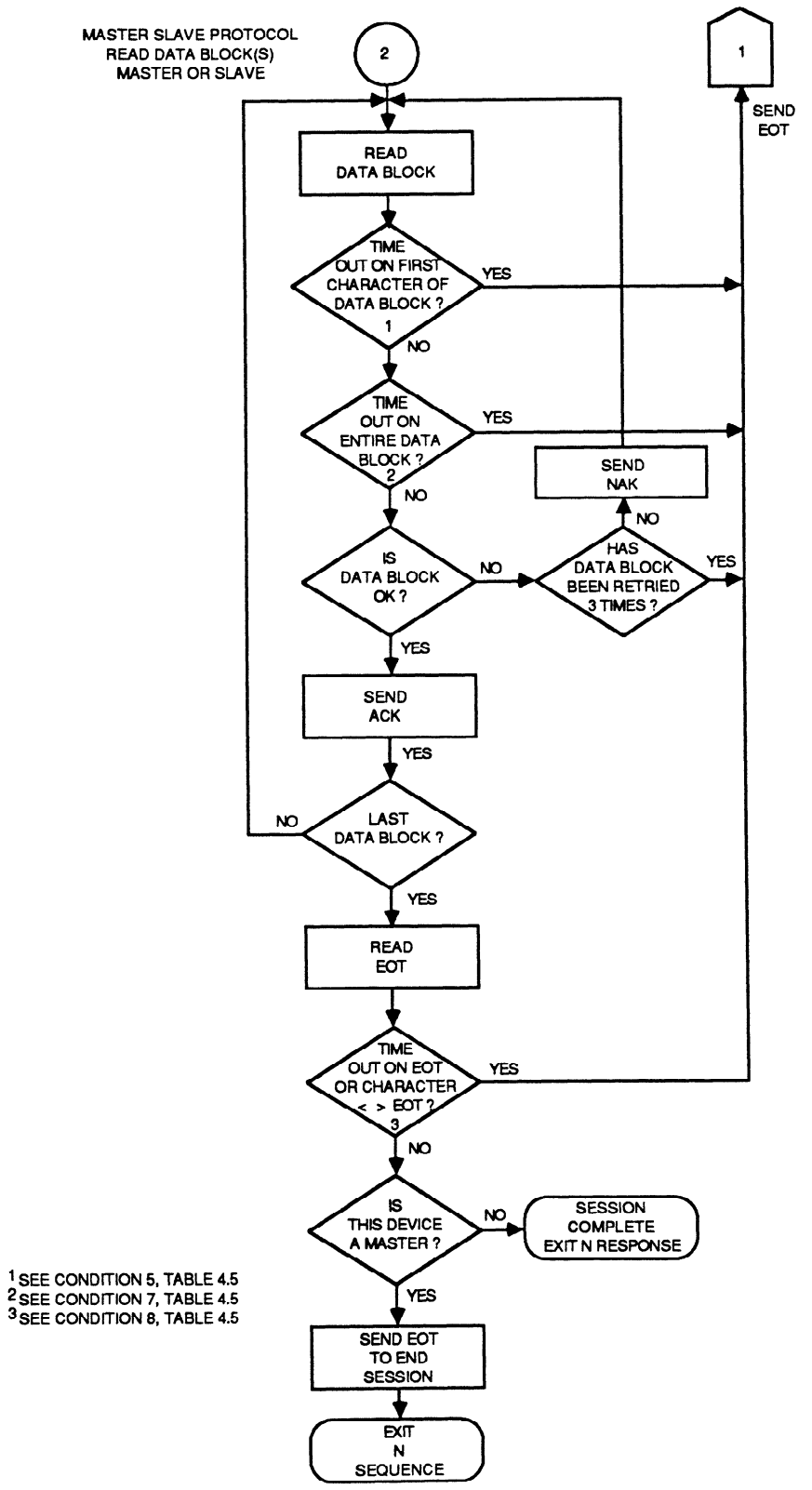


Figure 4.13 READ DATA BLOCKS, MASTER OR SLAVE

Is this device a Master?
If YES, exit N Sequence.
If NO, read EOT.
Is there a time-out on EOT or is character not an EOT? (Condition 8, Table 4.5)
If there is a time-out or character is not EOT, send EOT and exit N Response.
If EOT is OK, session is complete. Exit N Response.

Read Data Blocks, Master or Slave (See Figure 4.13)

Read data block.
Is there a time-out on the first character of the data block? (Condition 5, Table 4.5)
If YES, send an EOT and exit.
If NO, is there a time-out on the entire data block? (Condition 7, Table 4.5)
If YES, send and EOT and exit.
If NO, is the data block OK?
If NO, has the data block been retried 3 times?
If YES, send EOT and exit.
If NO, send NAK and return to "Read Data Block".
If YES, send ACK.
Is it the last data block?
If NO, return to "Read Data Block".
If YES, read EOT.
Is there a time-out on the EOT or is the character not an EOT? (Condition 8, Table 4.5)
If there is a time-out or character is not EOT, send EOT and exit.
If EOT is OK, is this device a master?
If NO, the session is complete, exit N Response.
If YES, send EOT to end session, exit N Sequence.

Q SEQUENCE, MASTER-SLAVE

The Q sequence operation can be used to poll and transfer 4 bytes of data from slaves without having to send a 17-byte header. To do this the CCM commands 06109 or 06209, Read Q Response, are used. The Q Sequence protocol format is shown below.

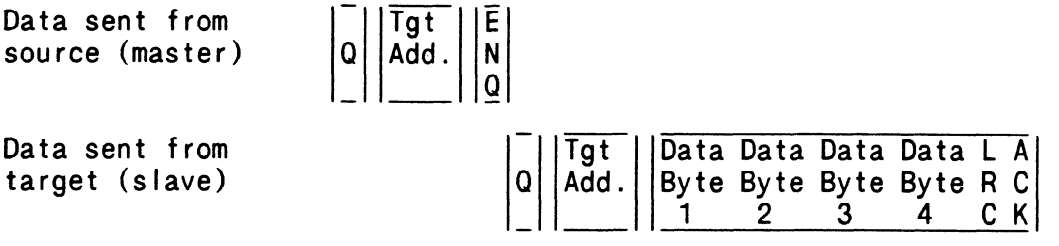


Figure 4.14 Q SEQUENCE PROTOCOL FORMAT

- ASCII coded "Q" signifying Q Sequence operation is sent by the master and returned by the slave.
- Slave target ID + 20H is sent by the master and returned by slave.
- ASCII control character ENQ for enquiry by Master.
- Data byte 1 sent by slave.
- Data byte 2 sent by slave.
- Data byte 3 sent by slave.
- Data byte 4 sent by slave.
- LRC - Longitudinal redundancy check sent by slave (XOR of Data Bytes 1-4 only).
- ACK - Acknowledge sent by slave.

This is the entire protocol format for Q Sequence operation. Only 4 data bytes can be transferred at a time and the direction is always from slave to master. After the Q Response is sent by the slave, it returns to the idle state without the need for an End of Transmission control character (EOT).

If the slave response to a master enquiry is invalid, the master will retry the enquiry. The master will retry the enquiry 3 times before aborting the communication.

Q Sequence Flow Charts

To fully understand how the protocol operates under error conditions see the flow charts and accompanying explanation.

Q Sequence, Master (See Figure 4.15)

Start Q Sequence.
Start Q Enquiry.
Has Q Enquiry been retried 3 times?
    If YES, exit Q Sequence.
    If NO, send Q Enquiry Sequence (Q, Target Address, ENQ).
Read Q Response.
Is there a time-out or error in the Q Response? (Condition 1, Table 4.5)
    If YES, increment retry count and return to "Start Q Enquiry".
    If NO, valid response has been received, exit Q Sequence.

Q Response, Slave (See Figure 4.16)

Start Q Response.
Read Q Enquiry Sequence.
Is Q Enquiry correct?
    If NO, return to "Read Q Enquiry Sequence".
    If YES, start timer (10 msec plus 4 character times).
Is timer done?
    If NO, have any characters arrived?
        If YES, return to "Read Q Enquiry Sequence".
        If NO, return to "Is Timer Done?".
    If YES, send Q Response and exit.

84pc0061

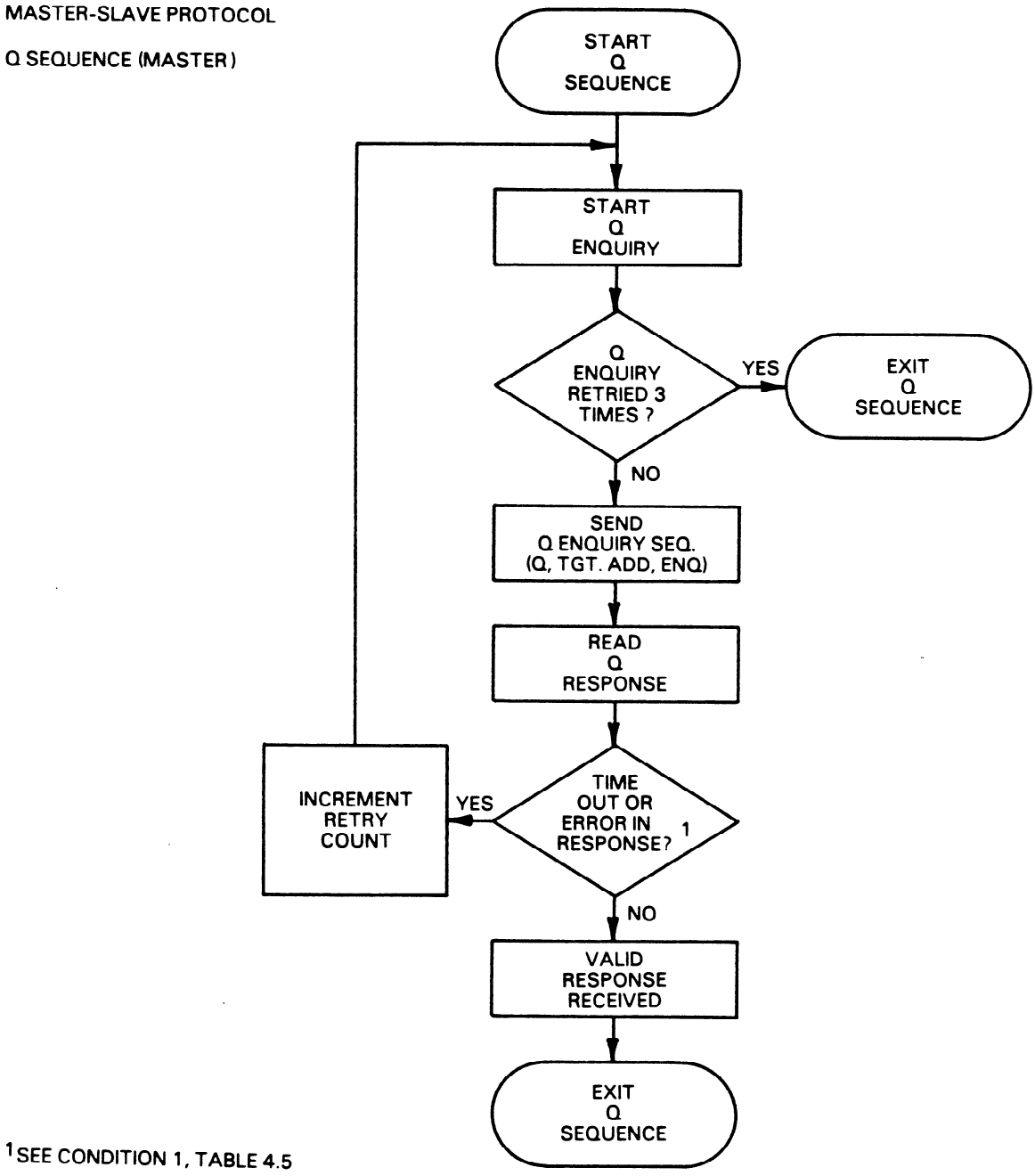


Figure 4.15 Q SEQUENCE, MASTER

a42699

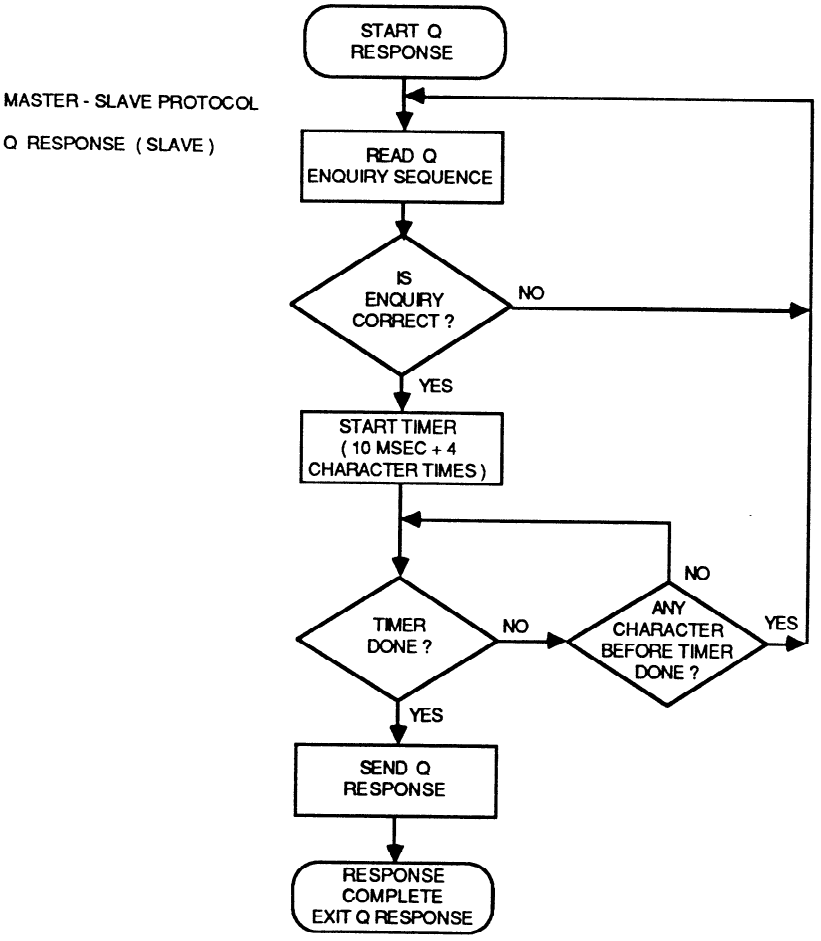


Figure 4.16 Q RESPONSE, SLAVE

HEADER BLOCKS

The header block format and valid responses to the header block are the same for both peer-to-peer protocol and master-slave protocol. The header block is sent from the source device to the target device. The specific contents of the header are shown below.

SOH	Target ID	Data Flow Dir & Tgt Mem	Target Memory Type	Target Memory Address MSB	Target Memory Address LSB	No. of Complete Data Blocks	No. of Bytes in Last Block	Source ID	ETB	LRC
1	2 3	4	5	6 7	8 9	10 11	12 13	14 15	16	17

Byte 1 : SOH (01H)
Bytes 2, 3 : Target ID
Byte 4 : Data flow direction (read or write)/Target memory type (Most Significant Byte)
Byte 5 : Target memory type (Least Significant Byte)
Bytes 6, 7 : Target memory address (Most Significant Byte)
Bytes 8, 9 : Target memory address (Least Significant Byte)
Bytes 10, 11: Number of full 256 byte data blocks
Bytes 12, 13: Number of bytes in last data block if less than 256 bytes
Bytes 14, 15: Source ID
Byte 16 : ETB (17H)
Byte 17 : LRC (XOR of bytes 2-15)

Figure 4.17 HEADER BLOCK FORMAT

The information in bytes 2-15 are ASCII coded hexadecimal. Valid ASCII coded hexadecimal values are 30H-39H (0-9) and 41H-46H (A-F). For fields requiring more than one byte, the most significant byte is transmitted first. For example, if the target ID were 254 (FEH), byte 2 would contain 46H (F) and byte 3 would contain 45H (E); byte 2 would be transmitted first.

TARGET ID: Bytes 2, 3

The Target ID is the identification number of the target device. For a Series Six CPU, it is the CPU ID number. It can range from 1 to 255 (ASCII coded hexadecimal, 01 to FF) when using peer-to-peer protocol. When using peer-to-peer protocol, target ID 255 is recognized by any CPU no matter what its ID number. When using master-slave protocol, valid IDs range from 1-90 (decimal).

These two bytes representing ASCII coded hexadecimal values from 01 to FF and are not encoded the same as the target address in the enquiry sequence.

DATA FLOW DIRECTION AND TARGET MEMORY TYPE: (Bytes 4 and 5)

Bytes 4 and 5 supply the target memory type. Byte 4 also supplies the data direction (read or write).

The contents of byte 4 may range from ASCII 0 (Hex 30), to ASCII 9 (Hex 39), and ASCII A (Hex 41) to ASCII F (Hex 46). Table 4.3 lists the memory types currently supported by the CCM module.

Byte 4 also contains information on which memory type is being accessed. If byte 4 is an ASCII 0 (Hex 30), the request is a read request. If byte 4 is an ASCII 8 (Hex 38) or 9 (Hex 39), the request is a write request.

Byte 5 is the Least Significant Byte of the target memory type. Table 4.3 shows the valid memory types for CCM read and write requests. Refer to Appendix B, Expanded Functions for detailed information concerning expanded memory mapping.

Table 4.3 TARGET MEMORY TYPES

TARGET MEMORY TYPE (decimal)	BYTE 4 * HEX RD/WR	BYTE 4 * ASCII RD/WR	BYTE 5 HEX	BYTE 5 ASCII	TARGET MEMORY TYPE
0	30/38	0/8	30	0	CPU Absolute Memory Address
1	30/38	0/8	31	1	CPU Register Table
2	30/38	0/8	32	2	CPU Input Table
3	30/38	0/8	33	3	CPU Output Table
4	30/38	0/8	34	4	CPU Input Override Table
5	30/38	0/8	35	5	CPU Output Override Table
6	30/38	0/8	36	6	CPU Scratchpad
7	30/38	0/8	37	7	CPU User Logic
8	30/38	0/8	38	8	CCM Quick Access Buffer
9	30/38	0/8	39	9	CCM Diagnostic Status Words
13	38	8	44	D	Input Table Bit Set
14	38	8	45	E	Output Table Bit Set
15	38	8	46	F	Input Ovrd Table Bit Set
16	39	9	30	0	Output Ovrd Table Bit Set
17	39	9	31	1	Input Table Bit Clear
18	39	9	32	2	Output Table Bit Clear
19	39	9	33	3	Input Ovrd Table Bit Clear
20	39	9	34	4	Output Ovrd Table Bit Clear
21	39	9	35	5	Input Table Bit Toggle
22	39	9	36	6	Output Table Bit Toggle

* Bit functions can only be write requests.

TARGET MEMORY ADDRESS: Bytes 6, 7, 8, 9

These bytes inform the target device at which address the read or write is to begin. For example, if the target memory address is 00986 (03DAH), each hexadecimal digit is converted to ASCII coded hexadecimal for transmission as shown below.

	BYTE			
	6	7	8	9
Target Memory Address (hexadecimal)	0	3	D	A
Target Memory Address (converted to ASCII coded hexadecimal)	30	33	44	41

Therefore, byte 6 contains 30; byte 7, 33; byte 8, 44; and byte 9, 41.

Refer to Chapter 2, Table 2.14, for a complete listing of the target memory addresses.

NUMBER OF COMPLETE DATA BLOCKS: Bytes 10, 11

These bytes specify the number of complete 256 byte data blocks to be transferred following the header. This number can range from 0-255 (00-FF).

NUMBER OF BYTES IN LAST DATA BLOCK: Bytes 12, 13

These bytes specify the number of bytes in the final data block transmitted if the final block contains less than 256 bytes. This number can range from 0-255 (00-FF). If the number of complete data blocks is zero, this number specifies the total number of bytes to be transferred.

SOURCE ID: Bytes 14, 15

The source ID is the identification number of the source device. For a Series Six CPU, it is the CPU ID number. The limits for the source ID are the same as for the target ID.

DATA TEXT BLOCKS

Data text is broken up into blocks with a maximum size of 256 bytes. The contents of a data text block is shown below.

Full data block (except last)	STX	256 Data Bytes	ETB LRC
Last data block	STX	256 or fewer data bytes	ETX LRC

- STX - ASCII control character, Start of Text preceeding each data block.
- 256 or fewer data bytes of binary data.
- After the data text an End of Block (ETB) character is inserted unless it is the last data block in transmission when an End of Text (ETX) character is inserted.
- A Longitudinal Redundancy Check (LRC) is inserted after each ETB or ETX. The LRC is an XOR of all the data text bytes; it does not include STX, ETX, or ETB. See section, Longitudinal Redundancy Check in Chapter 1.

When 16 bit information (registers or user logic) is being transferred in a data text block, the least significant byte is transferred first, followed by the most significant byte.

CCM HEADER EXAMPLE

In the following example, the source device (ID = 02) reads Register 00986 from the target device ID = 01.

Table 4.4 CCM HEADER EXAMPLE

BYTE	BINARY	HEX	ASCII	POSITION
Start of Header	0000 0001	01	SOH	1
Target ID - MSB	0011 0000	30	0	2
Target ID - LSB	0011 0001	31	1	3
Data Direction	0011 0000	30	0	4
Target Memory Type	0011 0001	31	1	5
Target Memory Address - MSB	0000 0000	30	0	6
Target Memory Address - NMSB	0011 0011	33	3	7
Target Memory Address - NMSB	0100 0100	44	D	8
Target Memory Address - LSB	0100 0001	41	A	9
Complete Block - MSB	0011 0000	30	0	10
Complete Block - LSB	0011 0000	30	0	11
Bytes Last Block - MSB	0011 0000	30	0	12
Bytes Last Block - LSB	0011 0010	32	2	13
Source ID - MSB	0011 0000	30	0	14
Source ID - LSB	0011 0010	32	2	15
End Transfer Block	0001 0111	17	ETB	16
Block Check Character	0000 0110	06	LRC *	17

* The LRC value is the vertical XOR result of bytes 2 - 15. Any like numbers cancel each other to zero.

MSB = Most Significant Byte, NMSB = Next Most Significant Byte

SERIAL LINK TIME-OUTS

A time-out occurs on a serial link when a CCM does not receive a response, a header, or data from another device within a required amount of time. Time-outs are used on the serial link for error detection, error recovery, and to prevent missing end of block sequences. Whenever a serial link time-out occurs, the CCM will abort the communication and send an EOT to the other device.

The time-outs are listed in Table 4.5 and apply to both directions on the serial link. The turn-around delay is added by the CCM when used. When the user writes communications software, he must ensure that data, headers, and responses are transmitted within the time allowed to avoid an error condition.

Later versions of the CCM module support programmable serial link timeout values. For more information concerning programmable timeout and retry, refer to Appendix B, Expanded Functions.

Table 4.5 SERIAL LINK TIME-OUTS

CONDITION	TIME-OUT IN MSEC WITH TURN-AROUND OF 0 MSEC	10 MSEC	500 MSEC
1 Wait on ACK/NAK following ENQ	800	810	1300
2 Wait on start of header following ACK of ENQ	800	810	1300
3 Wait on header to finish			
Data Rate			
300	2670	2680	3170
600	1340	1350	1840
1200	670	680	1170
2400	670	680	1170
4800	670	680	1170
9600	670	680	1170
19200	670	680	1170
38400	670	680	1170
4 Wait on ACK/NAK following header	2000	2010	2500
5 Wait on start of data following ACK of header	20000	20010	20500
6 Wait on ACK/NAK following data block	20000	20010	20500

Table 4.5 SERIAL LINK TIME-OUTS (Continued)

CONDITION	TIME-OUT IN MSEC WITH TURN-AROUND OF 0 MSEC	10 MSEC	500 MSEC
7 Wait on data block to finish			
Data Rate			
300	33340	33350	33840
600	16670	16680	17170
1200	8340	8350	8840
2400	8340	8350	8840
4800	8340	8350	8840
9600	8340	8350	8840
19200	8340	8350	8840
38400	8340	8350	8840
8 Wait on EOT to close link	800	810	1300

TURN-AROUND DELAYS

Turn-around delay options for CCM2 and CCM3, are 0, 10, and 500 msec. Turn-around delay options for I/O CCM are 0 and 500 msec. Turn-around delays on the serial communications line are introduced when using modems or radio transmitters in the half-duplex mode of operation. This delay allows a computer or Series Six the time needed to signal the modem or radio transmitter to warm up before actual transmission of data. When a turn-around delay is selected, the time is automatically added to the serial time-outs. Turn-around delays can be selected by DIP switches or the appropriate CPU registers on the CCM2 and CCM3 module. See section in Chapter 2, Module Configuration.

PROGRAMMABLE RETRIES AND TIMEOUTS FOR CCM

Later versions of the CCM allows the user to select CCM protocol timeouts and retries by appropriate programming of the Series Six PLC. Refer to Appendix B, Expanded Functions for more information. The figure below lists programmable ranges for particular portions of the CCM protocol.

Table 4.6 PROGRAMMABLE TIME-OUTS FOR CCM

CONDITION	RANGE
Wait on ACK/NAK following ENQ	50 to 2000 msec
Wait on start of header following ACK of ENQ	50 to 2000 msec
Wait on header to finish	50 to 3000 msec
Wait on ACK/NAK following header	50 to 10000 msec
Wait on Start of data following ACK of header	50 to 65000 msec
Wait on ACK/NAK following data block	50 to 65000 msec
Wait on data block to finish	50 to 65000 msec
Wait on EOT to close link	50 to 2000 msec

SCREQs have been allocated to allow programmable timeouts to be set for each port configured for CCM protocol. If a turn-around delay has been selected, it will be added to the selected serial time-out.

SERIAL LINK COMMUNICATION ERRORS

Serial link communication errors can be divided into 4 categories.

- Invalid header
- Invalid data
- Invalid ACK, NAK, EOT
- Serial link time-out

Each of the errors in the four categories is detected by the CCM. The CCM reports errors through the Diagnostic Status Words. The error codes are listed in Chapter 2.

INVALID HEADER

- Target ID number does not match ID of device receiving header except when ID is 255 in peer-to-peer
- Incorrect header LRC
- Missing or invalid SOH
- Missing or invalid ETB
- Invalid memory type
- Transfer across a memory boundary
- Invalid header character (not 0-9, A-F)
- Invalid address for specified memory type
- Number of complete blocks and number of bytes in last block both equal 0
- Number of bytes in last block not an even number if memory type is register, user logic memory, or diagnostic status words
- Invalid CPU write command (trying to write to user logic memory, I/O override table, or CPU scratch pad with the memory switch in the PROTECT position or with memory protected by software memory protect)
- Invalid CPU scratch pad write
- Parity, overrun, or framing error

If any of the above errors occur, a NAK is sent to the external serial device. This signals the device to retransmit the header. The DATA OK light on the CCM is turned off.

The header is retried a maximum of three times unless programmed otherwise. If the header still has one of the above errors, the CCM will abort the communication and send an EOT to the external device. The CCM then waits for an ENQ to start a new communication. The DATA OK light is turned on after the next successful communication.

INVALID DATA

If any of the following errors occur, the same retry procedure is followed as for an invalid header.

- Incorrect LRC
- Missing or invalid STX
- Missing or invalid ETB or ETX
- Parity, overrun, or framing error

INVALID NAK, ACK, or EOT

If the CCM is expecting one of these control characters in response to a header or data block, and a character is received that is not one of these, the CCM aborts the session and sends an EOT to the other device.

SERIAL LINK TIME-OUT

If at any time during the communication after the enquiry sequence the CCM times out waiting for the other device, the communication is aborted and an EOT is sent to the other device.

WRITING TO CPU SCRATCH PAD

There are only 2 fields within the CPU Scratch Pad to which a remote device is permitted to write data: the CPU Run and Status field and the Subroutine Vector Address field.

Table 4.7 SCRATCH PAD FIELDS

FIELD	ADDRESS	
	ABSOLUTE MEM.	SCRATCH PAD MEM.
CPU Run and Command Status	1000H- 1001H	0000H- 0001H
Subroutine Vector Addresses	1060H- 107FH	0060H- 007FH

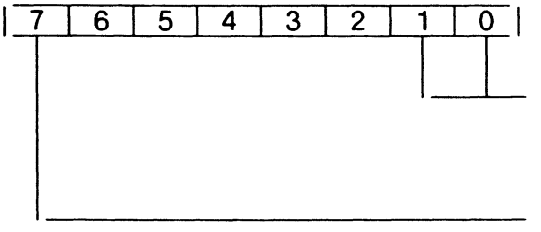
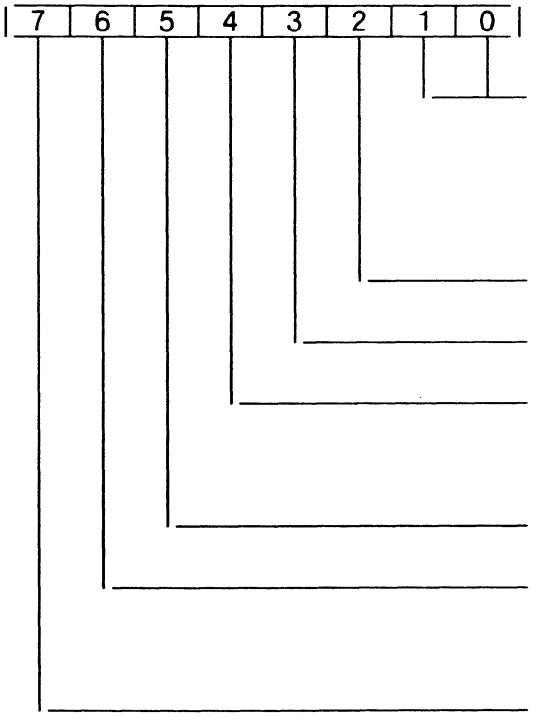
CPU RUN AND COMMAND STATUS

To stop the CPU, 128 (80H) is written to both 4096 and 4097 (1000H and 1001H) of Absolute Memory or 0000H and 0001H of Scratch Pad Memory. To start the CPU, 01H is written to both locations.

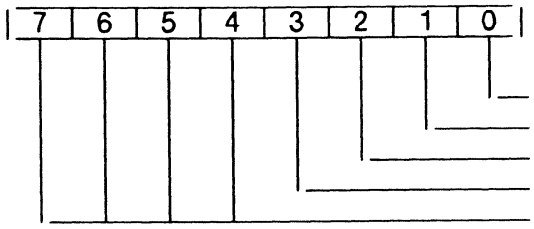
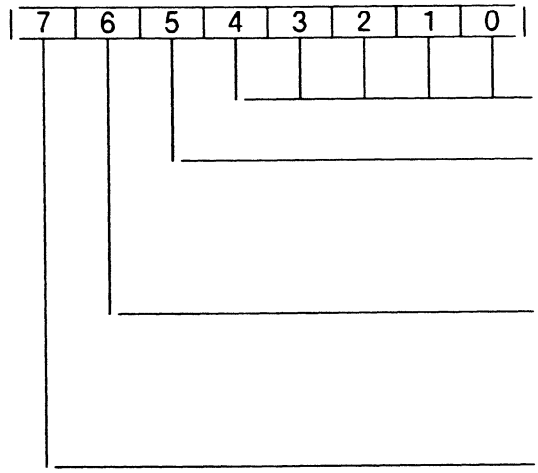
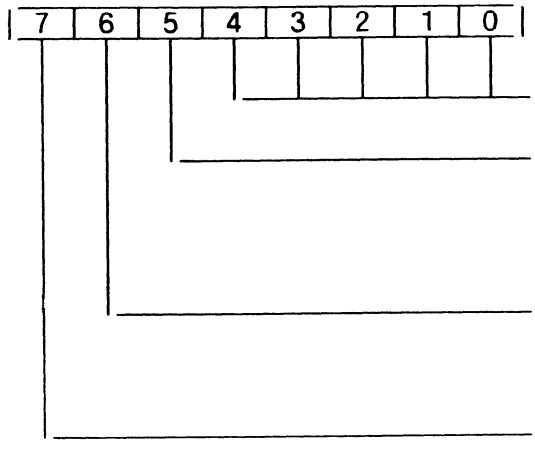
SUBROUTINE VECTOR ADDRESSES

If a host computer is used to develop a Series Six logic program with subroutines, the subroutine vector addresses must be written to the CPU Scratch Pad.

SCRATCH PAD MEMORY ALLOCATION

HEX ADDRESS	BITS	MEANING
00		CPU Run Status 00 = Stop 01 = Start 11 = Run 0 = Outputs Enabled 1 = Outputs Disabled
01	(Same Bit Pattern as Address 00)	CPU Command Status
06		CPU Options Register Size 00 = 16K 01 = 8K 10 = Reserved 11 = 1K 1 = DPU Present 1 = CCM Present Memory Protect Switch 0 = Protect 1 = Write 1 = Expanded Functions CPU Model 0 = 256 Registers 1 = Check bits 0 and 1 Run/Stop Switch 0 = Run 1 = Stop
07		First byte of CPU Flags (leftmost)
08		Second byte of CPU Flags
09		Third byte of CPU Flags
0A		Fourth byte of CPU Flags (rightmost)
	Note:	CPU Flags may be viewed on the bottom of the Scratch Pad Menu of the Logicmaster Six PLC
0B-0E		CPU Logic Memory Map - Each bit set represents 1K of User Logic

SCRATCH PAD MEMORY ALLOCATION (continued)

HEX ADDRESS	BITS	MEANING
0F		Instruction Set 1 = Basic 1 = Extended 1 = Advanced 1 = Expanded 0 = Always 0
13		CPU Software Version
16		CPU ID Number
59		I/O Diagnostic Flags Don't Care 1 = Don't stop on I/O Parity Failure 0 = Stop if given number of retries fail 1 = Modify I/O Parity Error Retries 0 = Leave retries a 1 after power up Don't Care
5A		I/O parity Error Retry Count
5B		Always 0 1 = I/O Parity Error and all retries bad 0 = No I/O Cycle used max retries 1 = I/O Parity Error since last read 0 = No Parity Error Always 1
60 -7F		Subroutine vector address

CHAPTER 5
RTU COMMUNICATIONS PROTOCOL

INTRODUCTION

The Communications Control Modules (CCM3 and I/O CCM) use two protocols, CCM and Remote Terminal Unit (RTU). The CCM protocol is explained in Chapter 4 of this manual. When the CCM3 or I/O CCM module (CCM device) is configured as an RTU slave, it uses the protocol as explained in this chapter.

RTU protocol is a query-response protocol used for communication between the CCM device and a host computer which is capable of communicating using RTU protocol. The host computer is the master device and it transmits a query to a RTU slave which responds to the master. The CCM device, as an RTU slave, cannot query; it can only respond to the master.

The RTU data transferred consists of 8-bit binary characters with or without parity. No control characters are used to control the flow of data, there is, however, an error check (Cyclic Redundancy Check) included as the final field of each query and response to ensure accurate transmission of data.

MESSAGE FORMAT

The general formats for RTU message transfers are shown below.

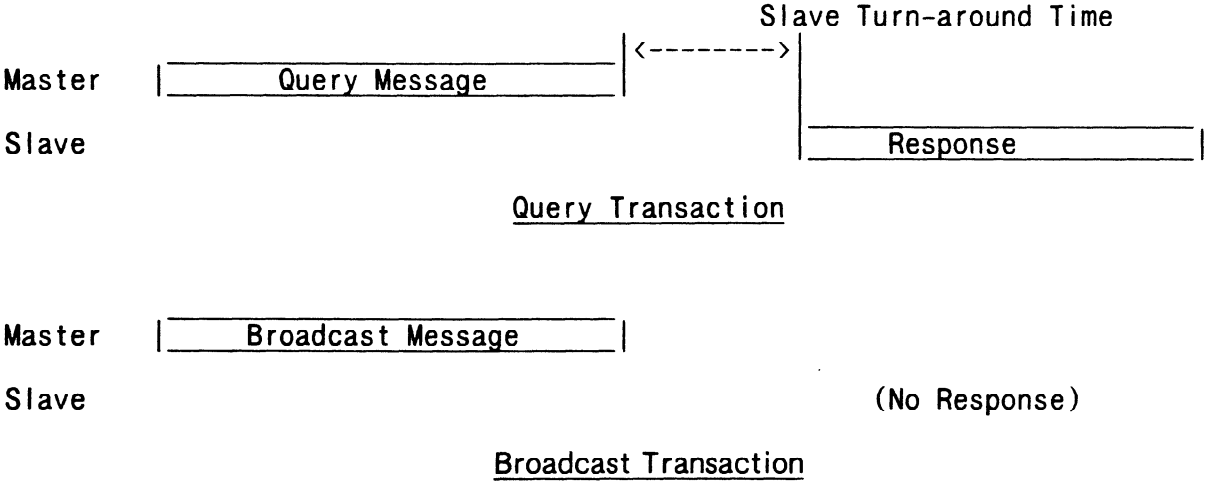


Figure 5.1 RTU MESSAGE TRANSFERS

A distinction is made between two communicating devices. The device which initiates a data transfer is called the master and the other device is called the slave. The CCM device can only be a RTU slave.

The master device begins a data transfer by sending a query or broadcast request message. A slave completes that data transfer by sending a response message if the master sent a query message addressed to it. No response message is sent when the master sends a broadcast request. The time between the end of a query and the beginning of the response to that query is called the slave turn-around time.

MESSAGE TYPES

The RTU protocol has four message types; query, normal response, error response, and broadcast.

Query

The master sends a message address to a single slave.

Normal Response

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

Error Response

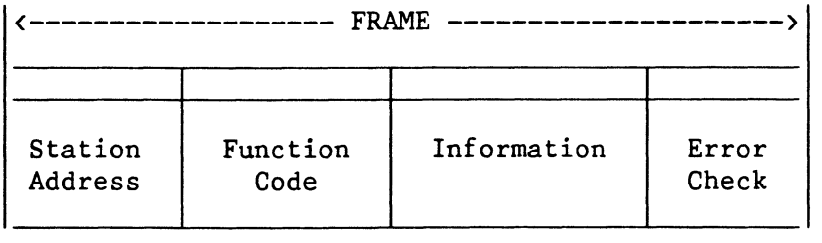
The slave receives the query, but for some reason it cannot perform the requested function. The slave sends back an error response which indicates the reason the request could not be processed. (No error message will be sent for certain types of errors. For more information see section, Communication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

Broadcast

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

MESSAGE FIELDS

The message fields for a typical message are shown below.



Station Address

The station address is the address of the slave station selected for this data transfer. It is one byte in length and has a value from 0 to 247 inclusive. An address of 0 selects all slave stations, and indicates that this is a broadcast message. An address from 1 to 247 selects a slave station with that station address. The CCM device (module) address is equal to the CPU ID of the attached Series Six™ PLC.

Function Code

The function code identifies the command being issued to the station. It is one byte in length and is defined for the values 0 to 255 as follows:

0	Illegal Function
1	Read Output Table
2	Read Input Table
3	Read Registers *
4	Read Registers *
5	Force Single Output
6	Preset Single Register
7	Read Exception Status
8	Loopback Maintenance
9-14	Unsupported Function
15	Force Multiple Outputs
16	Preset Multiple Registers
17	Report Device Type
18-64	Unsupported Function
65	Read Output Override Table
66	Read Input Override Table
67	Read Scratch Pad Memory
68	Read User Logic
69	Write Output Override Table
70	Write Input Override Table
71	Write Scratch Pad Memory
72	Write User Logic
73-127	Unsupported Function
128-255	Reserved for Exception Responses

* These two functions are identical.

Information Field

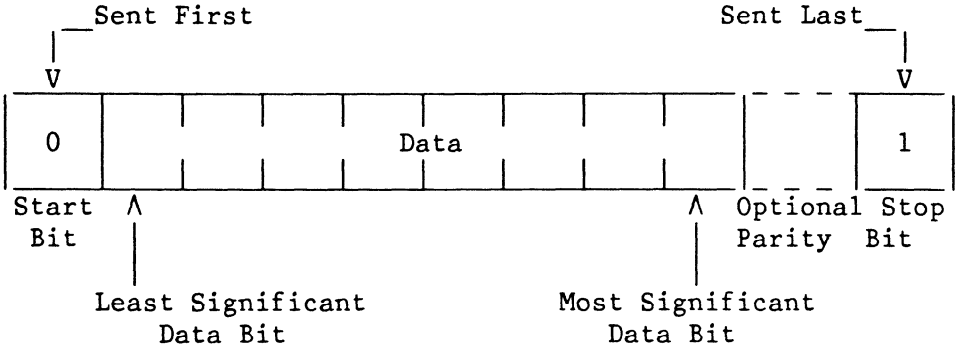
The information field contains all of the other information required to further specify or respond to a requested function. Detailed specification of the contents of the information field for each message type--broadcast, query, normal response, and error response--and each function code is found in the section, Message Descriptions.

Error Check Field

The error check field is two bytes in length and contains a cyclic redundancy check (CRC-16) code. Its value is a function of the contents of the station address, function code, and information field. The details of generating the CRC-16 code are in the section, Cyclic Redundancy Check (CRC). Note that the information field is variable in length. In order to properly generate the CRC-16 code, the length of frame must be determined. See section, Calculating the Length of Frame, to calculate the length of a frame for each of the defined function codes.

CHARACTER FORMAT

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



MESSAGE TERMINATION

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

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

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

TIME-OUT USAGE

Time-outs are used on the serial link for error detection, error recovery, and to prevent the missing of the end of messages and message sequences. Note that although the module allows up to three character transmission times between each character in a message that it receives, there is no more than half a character time between each character in a message that the module transmits.

The slave turn-around times listed in Table 5.1 are the guaranteed maximum times for the communication module. In many cases the actual turn-around times will be much less.

Table 5.1 RTU TURN-AROUND TIME

DESCRIPTION		RTU TURN-AROUND TIME* (MILLISECONDS)
Normal Responses		
Function Code	1	500
	2	500
	3	500
	4	500
	5	500
	6	500
	7	500
	8	500
	15	500
	16	500
	17	500
	65	500
	66	500
	67	500
	68	500
	69	500
	70	500
	71	500
	72	500
Error Responses		
Error Code	1	500
	2	500
	3	500
	4	500

* Times are given for one port busy. If both ports are busy double the times given.

CYCLIC REDUNDANCY CHECK (CRC)

The Cyclic Redundancy Check (CRC) is one of the most effective systems for checking errors. The CRC consists of 2 check characters generated at the transmitter and added at the end of the transmitted data characters. Using the same method, the receiver generates its own CRC for the incoming data and compares it to the CRC sent by the transmitter to ensure proper transmission.

A complete mathematic derivation for the CRC will not be given in this section. This information can be found in a number of texts on data communications. The essential steps which should be understood in calculating the CRC are as follows:

- The data bits which make up the message are multiplied by the number of bits in the CRC.
- The resulting product is then divided by the generating polynomial (using modulo 2 with no carries). The CRC is the remainder of this division.
- Disregard the quotient and add the remainder (CRC) to the data bits and transmit the message with CRC.
- The receiver then divides the message plus CRC by the generating polynomial and if the remainder is 0, the transmission was transmitted without error.

A generating polynomial is expressed algebraically as a string of terms in powers of X such as $X^3 + X^2 + X^0$ (or 1) which can in turn be expressed as the binary number 1101. A generating polynomial could be any length and contain any pattern of 1s and 0s as long as both the transmitter and receiver use the same value. For optimum error detection, however, certain standard generating polynomials have been developed. RTU protocol uses the polynomial $X^{16} + X^{15} + X^2 + 1$ which in binary is 1 1000 0000 0000 0101. The CRC this polynomial generates is known as CRC-16.

The discussion above can be implemented in hardware or software. One hardware implementation involves constructing a multi-section shift register based on the generating polynomial.

a40473

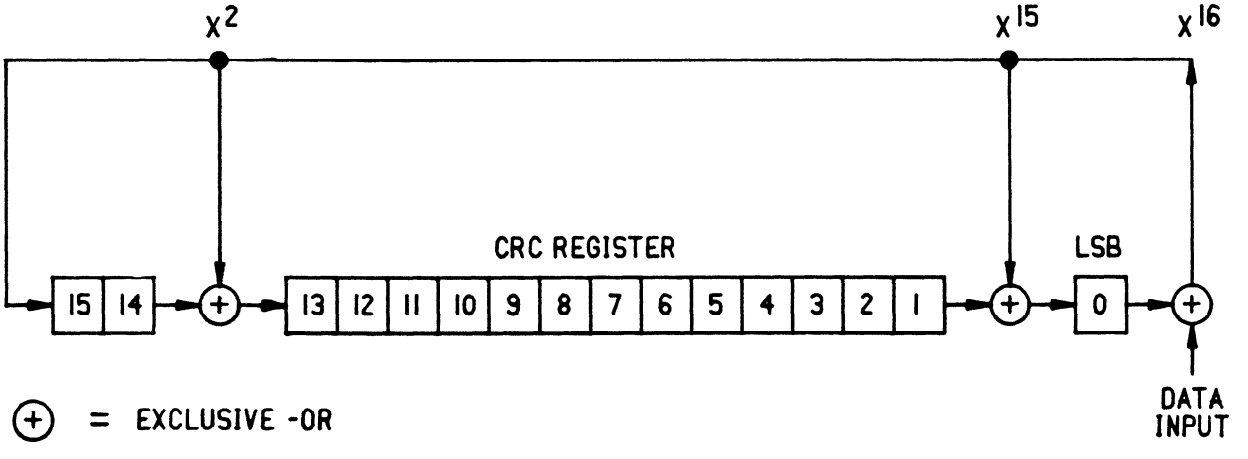


Figure 5.2 CYCLIC REDUNDANCY CHECK (CRC) REGISTER

To generate the CRC, the message data bits are fed to the shift register one at a time. The CRC register contains a preset value. As each data bit is presented to the shift register, the bits are shifted to the right. The LSB is XORed with the data bit and the result is: XORed with the old contents of bit 1 (the result placed in bit 0), XORed with the old contents of bit 14 (and the result placed in bit 13), and finally, it is shifted into bit 15. This process is repeated until all data bits in a message have been processed. Software implementation of the CRC-16 is explained in the next section.

CALCULATING THE CRC-16

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

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

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

EXAMPLE CRC-16 CALCULATION

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

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

Address	Func	CRC-16
01	07	

In this example we are querying device number 1 (address 01). We need to know the amount of data to be transmitted and this information can be found for every message type in the section, Calculating the Length of Frame. For this message the data length is 2 bytes.

TRANSMITTER
CRC-16 ALGORITHM

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

RECEIVER*
CRC-16 ALGORITHM

	MSB			LSB	Flag
Rcvr CRC after data	1110	0010	0100	0001	
XOR 1st byte Trns CRC	0000	0000	0100	0001	
Current CRC	1110	0010	0000	0000	
Shift 1	0111	0001	0000	0000	0
Shift 2	0011	1000	1000	0000	0
Shift 3	0001	1100	0100	0000	0
Shift 4	0000	1110	0010	0000	0
Shift 5	0000	0111	0001	0000	0
Shift 6	0000	0011	1000	1000	0
Shift 7	0000	0001	1100	0100	0
Shift 8	0000	0000	1110	0010	0
XOR 2nd byte trns CRC	0000	0000	1110	0010	
Current CRC	0000	0000	0000	0000	
Shift 1-8 yields	0000	0000	0000	0000	

ALL ZEROES FOR RECEIVER
FINAL CRC-16 INDICATES
TRANSMISSION CORRECT!

EXAMPLE MESSAGE
Refer to the example of a transmitted message shown on the following page.

*As stated before, the receiver processes incoming data through the same CRC algorithm as the transmitter. The example for the receiver starts at the point after all the data bits but not the transmitted CRC have been received correctly. Therefore, the receiver CRC should be equal to the transmitted CRC at this point. When this occurs, the output of the CRC algorithm will be zero indicating that the transmission is correct.

The transmitted message with CRC would then be:

```
1110 0010   0100 0001   0000 0111   0000 0001
  E    2      4    1      0    7      0    1
^          Order of transmission --->       ^
|                                           |
Transmitted last                  Transmitted first
```

CALCULATING THE LENGTH OF FRAME

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

Table 5.2 RTU MESSAGE LENGTH

FUNCTION CODE AND NAME	QUERY OR BROADCAST MESSAGE LENGTH LESS CRC CODE	RESPONSE MESSAGE LENGTH LESS CRC CODE
0	Not Defined	Not Defined
1 Read Output Table	6	3 + 3rd byte *
2 Read Input Table	6	3 + 3rd byte *
3 Read Registers	6	3 + 3rd byte *
4 Read Registers	6	3 + 3rd byte *
5 Force Single Output	6	6
6 Preset Single Register	6	6
7 Read Exeption Status	2	3
8 Loopback/Maintenance	6	6
9-14	Not Defined	Not Defined
15 Force Multiple Outputs	7 + 7th byte *	6
16 Preset Multiple Registers	7 + 7th byte *	6
17 Report Device Type	2	3 + 3rd byte
18-64	Not Defined	Not Defined
65 Read Output Override Table	6	3 + 3rd byte *
66 Read Input Override Table	6	3 + 3rd byte *
67 Read Scratch Pad Memory	6	3 + 3rd byte *
68 Read User Logic	6	3 + 3rd byte *
69 Write Output Override Table	7 + 7th byte *	6
70 Write Input Override Table	7 + 7th byte *	6
71 Write Scratch Pad Memory	7 + 7th byte *	6
72 Write User Logic	7 + 7th byte *	6
73-127	Not Defined	Not Defined
128-255	Not Defined	3

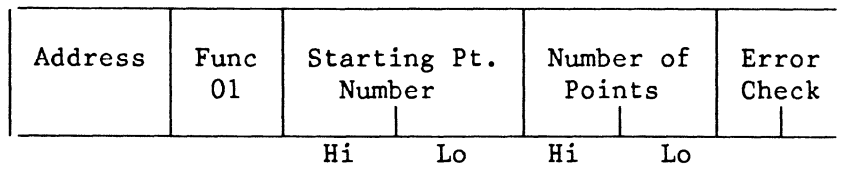
* The value of this byte is the number of bytes contained in the data being transmitted.

MESSAGE DESCRIPTIONS

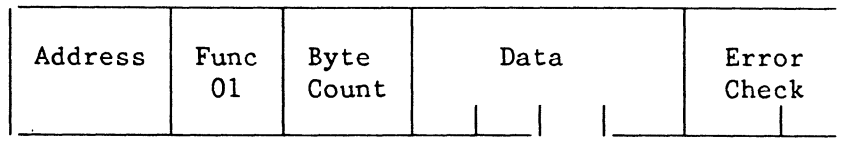
The following pages explain the format and fields for each RTU message.

MESSAGE (01): **READ OUTPUT TABLE**

FORMAT:



Query



Normal Response

QUERY:

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

RESPONSE:

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

MESSAGE (02): **READ INPUT TABLE**

FORMAT:

Address	Func 02	Starting Pt. Number		Number of Points		Error Check
		Hi	Lo	Hi	Lo	

Query

Address	Func 02	Byte Count	Data	Error Check

Normal Response

QUERY:

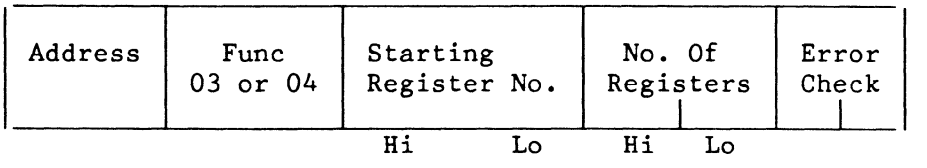
- An address of 0 is not allowed as this cannot be a broadcast request.
- The function code is 02.
- The starting point number is two bytes in length and may be any value less than the highest input point number available in the attached Series Six CPU. The starting point number is equal to one less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

RESPONSE:

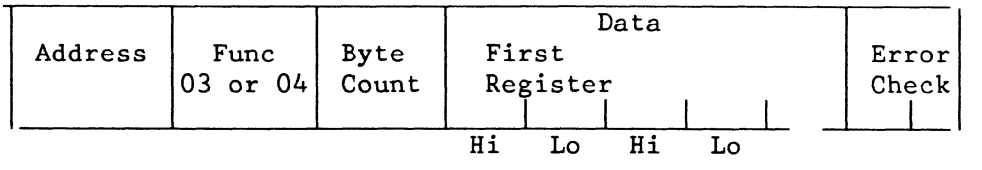
- The byte count is a binary number from 1 to 256 (0 = 256). It is the number of bytes in the normal response following the byte count and preceeding the error check.
- The data field of the normal response is packed input status data. Each byte contains 8 input point values. The least significant bit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

MESSAGE (03, 04): **READ REGISTERS**

FORMAT:



Query



Normal Response

QUERY:

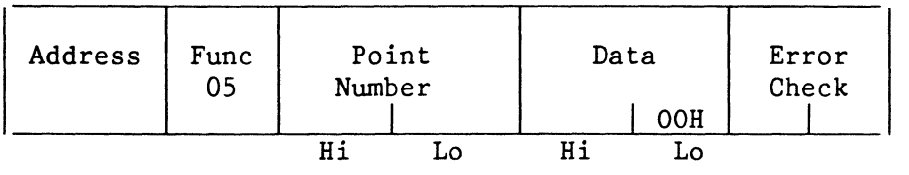
- (An address of 0 is not allowed as this request cannot be a broadcast request.
- The function code is equal to either 3 or 4.
- The starting register number is two bytes in length. The starting register number may be any value less than the highest register number available in the attached Series Six CPU. It is equal to one less than the number of the first register returned in the normal response to this request.
- The number of registers value is two bytes in length. It must contain a value from 1 to 125 inclusive. The sum of the starting register value and the number of registers value must be less than or equal to the highest register number available in the attached Series Six CPU. The high order byte of the starting register number and number of registers fields is sent as the first byte in each of these fields. The low order byte is the second byte in each of these fields.

RESPONSE:

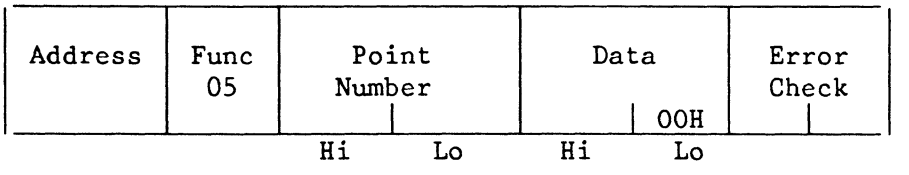
- The byte count is a binary number from 2 to 250 inclusive. It is the number of bytes in the normal response following the byte count and preceeding the error check. Note that the byte count is equal to two times the number of registers returned in the response. A maximum of 250 bytes (125) registers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

MESSAGE (05): **FORCE SINGLE OUTPUT**

FORMAT:



Query



Normal Response

QUERY:

- An address of 0 indicates a broadcast request. All slave stations process a broadcast request and no response is sent.
- The function code is equal to 05.
- The point number field is two bytes in length. It may be any value less than the highest output point number available in the attached Series 6 CPU. It is equal to one less than the number of the output point to be forced on or off.
- The first byte of the data field is equal to either 0 or 255 (FFH). The output point specified in the point number field is to be forced off if the first data field byte is equal to 0. It is to be forced on if the first data field byte is equal to 255 (FFH). The second byte of the data field is always equal to zero.

RESPONSE:

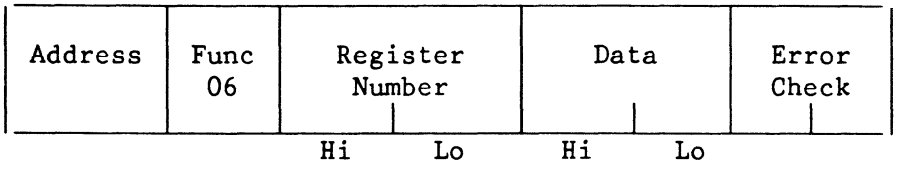
- The normal response to a force single output query is identical to the query.

NOTE

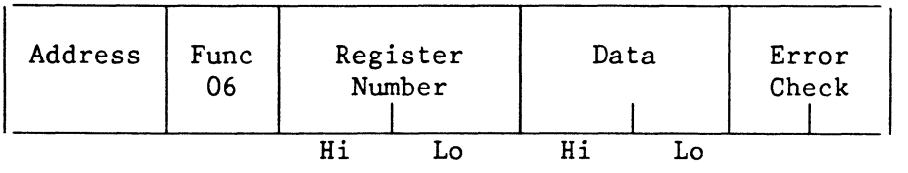
The force single output request is not an output override command. The output specified in this request is insured to be forced to the value specified only at the beginning of one sweep of the Series Six user logic.

MESSAGE (06): **PRESET SINGLE REGISTER**

FORMAT:



Query



Normal Response

QUERY:

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

RESPONSE:

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

MESSAGE (07): **READ EXCEPTION STATUS**

FORMAT:

Address	Func 07	Error Check

Query

Address	Func 07	Data	Error Check

Normal Response

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

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

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

MESSAGE (08): **LOOPBACK/MAINTENANCE (GENERAL)**

FORMAT:

Address	Func 08	Diagnostic Code 0, 1, or 4	Data DATA1 | DATA2	Error Check

Query

Address	Func 08	Diagnostic Code 0, 1, or 4	Data DATA1 | DATA2	Error Check

Normal Response

QUERY:

- The function code is equal to 8.
- The diagnostic code is two bytes in length. The high order byte of the diagnostic code is the first byte sent in the diagnostic code field. The low order byte is the second byte sent. The loopback/maintenance command is defined only for the diagnostic code equal to 0, 1, or 4. All other diagnostic codes are reserved.
- The data field is two bytes in length. The contents of the two data bytes are defined by the value of the diagnostic code.

RESPONSE:

- See descriptions for individual diagnostic codes.

DIAGNOSTIC CODE (00): **Return Query Data (Loopback/Maintenance)**

- A loopback/maintenance query with a diagnostic code equal to 0 is called a return query data request.
- An address of 0 is not allowed for the return query data request.
- The values of the two data field bytes in the query are arbitrary.
- The normal response is identical to the query.
- The values of the data bytes in the response are equal to the values sent in the query.

DIAGNOSTIC CODE (01): **Initiate Communication Restart (Loopback/Maintenance)**

A loopback/maintenance request (query or broadcast) with a diagnostic code equal to 1 is called an Initiate Communication Restart request.

- An address of 0 indicates a broadcast request. All slave stations process a broadcast request and no response is sent.
- This request disables the listen-only mode (enables responses to be sent when queries are received so that communications can be restarted).
- The value of the first byte of the data field (DATA1) must be 0 or FF. Any other value will cause an error response to be sent. The value of the second byte of the data field (DATA2) is always equal to 0.
- The normal response to an Initiate Communication Restart query is identical to the query.

DIAGNOSTIC CODE (04): **Force Listen-Only Mode (Loopback/Maintenance)**

A loopback/maintenance request (query or broadcast) with a diagnostic code equal to 4 is called a Force Listen-Only Mode request.

- An address of 0 indicates a broadcast request. All slave stations process a broadcast request.
- After receiving a Force Listen-Only mode request, the CCM device will go into the listen-only mode and will not send either normal or error responses to any queries. The listen-only mode is disabled when the CCM device receives an Initiate Communication Restart request and when the CCM device is powered up.
- Both bytes in the data field of a Force Listen-Only Mode request are equal to 0. The CCM device never sends a response to a Force Listen-Only Mode request.

NOTE

Upon power up, the CCM device disables the listen-only mode and is configured to continue sending responses to queries.

MESSAGE (15): **FORCE MULTIPLE OUTPUTS**

FORMAT:

Address	Func 15	Starting Point No.	Number Of Points	Byte Count	Data	Error Check

Query

Address	Func 15	Starting Point No.	Number Of Points	Error Check

Normal Response

QUERY:

- An address of 0 indicates a broadcast request. All slave stations process a broadcast request and no response is sent.
- The value of the function code is 15.
- The starting point number is two bytes in length and may be any value less than the highest output point number available in the attached Series Six CPU. The starting point number is equal to one less than the number of the first output point forced by this request.
- The number of points value is two bytes in length. The sum of the starting point number and the number of points value must be less than or equal to the highest output point number available in the attached Series Six CPU. The high order byte of the starting point number and number of bytes fields is sent as the first byte in each of these fields. The low order byte is the second byte in each of these fields.
- The byte count is a binary number from 1 to 256 (0 = 256). It is the number of bytes in the data field of the force multiple outputs request.
- The data field is packed data containing the values that the outputs specified by the starting point number and the number of points fields are to be forced to. Each byte in the data field contains the values that eight output points are to be forced to. The least significant bit (LSB) of the first byte contains the value that the output point whose number is equal to the starting point number plus one is to be forced to. The values for the output points are ordered by number starting with the LSB of the first byte of the data field and ending with the most significant bit (MSB) of the last byte of the data field. If the number of points is not a multiple of 8, then the last data byte contains zeros in one to seven of its highest order bits.

RESPONSE: • The description of the fields in the response are covered in the query description.

NOTE

The force multiple outputs request is not an output override command. The outputs specified in this request are ensured to be forced to the values specified only at the beginning of one sweep of the Series Six user logic.

MESSAGE (16): **PRESET MULTIPLE REGISTERS**

FORMAT:

Address	Func 16	Starting Register Number	Number Of Registers	Byte Count	Data	Error Check

Query

Address	Func 16	Starting Register Number	Number Of Registers	Error Check

Normal Response

QUERY:

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

RESPONSE:

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

MESSAGE (17): **REPORT DEVICE TYPE**

FORMAT:

Address	Func 17	Error Check

Query

Address	Func 17	Byte Count 5	Device Type 60	Slave Run Light	Data	Error Check

Normal Response

QUERY: The Report Device Type query is sent by the master to a slave in order to learn what type of programmable control or other computer it is. All models of the Series Six return a device type 60 when this request is received.

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

RESPONSE:
- The byte count field is one byte in length and is equal to 5.
- The device type field is one byte in length and is equal to 60.
- The slave run light field is one byte in length. The slave run light byte is equal to OFFH if the Series Six CPU is running. It is equal to 0 if the Series Six CPU is not running.

- The data field contains three bytes.

  The first byte is called the system configuration byte and is shown below. Bit 1 (the least significant bit) indicates whether or not the attached Series Six CPU user logic memory is write protected. Bit 2 indicates whether or not a Data Processing Unit (DPU) is connected to the attached Series Six CPU. Bit 3 indicates whether the attached Series Six CPU contains a basic or an extended instruction set. Bits 4 and 5 indicate how many registers the attached Series Six CPU contains. Bits 6,7 and 8 are reserved for future use and are equal to 0.

  The second and third data bytes specify the size of the attached Series Six PLC user logic memory. The second data byte contains the high order byte of the number of words of user logic memory (in units of 1024 words, commonly called kilowords or K words).
  The third data byte contains the low order byte of the number of K words of user logic memory in the attached Series Six CPU.

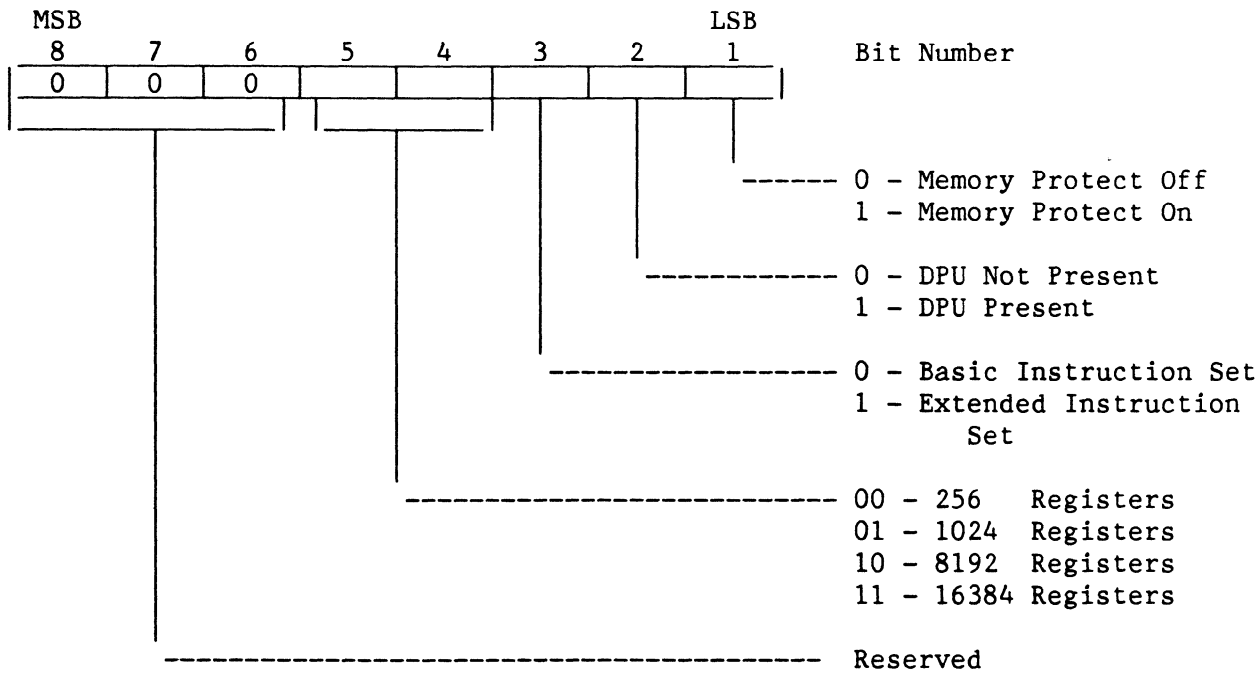


Figure 5.3 SYSTEM CONFIGURATION BYTE

MESSAGE (65): **READ OUTPUT OVERRIDE TABLE**

FORMAT:

Address	Func 65	Starting Point No.	Number Of Points	Error Check

Query

Address	Func 65	Byte Count	Data	Error Check

Normal Response

QUERY:

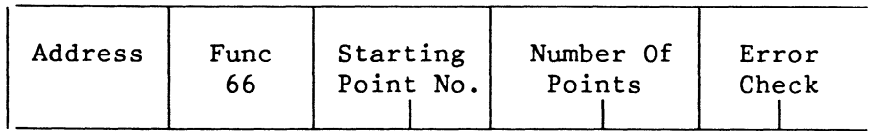
- An address 0 is not allowed as this cannot be a broadcast request.
- The function code is equal to 65.
- The starting point number is two bytes in length and may be any value less than the highest output point number available in the attached Series Six CPU. The starting point number is equal to one less than the number of the first output point whose override status is returned in the normal response to this request.
- The number of points value is two bytes in length. It specifies the number of output points whose override status are returned in the normal response. The sum of the starting point number and the number of points values must be less than or equal to the highest output point number available in the attached Series Six CPU. The high order byte of the starting point number and number of points fields is sent as the first byte in head of these fields. The low order byte is the second byte in each of these fields.

RESPONSE:

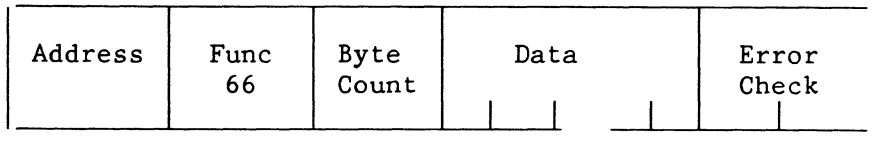
- The byte count is a binary number from 1 to 256 (0 = 256). It is the number of bytes in the data field of the normal response.
- The data field of the normal response is packed output override table data. Each byte contains the override status of eight output points. The least significant bit (LSB)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

MESSAGE (66): **READ INPUT OVERRIDE TABLE**

FORMAT:



Query



Normal Response

QUERY:

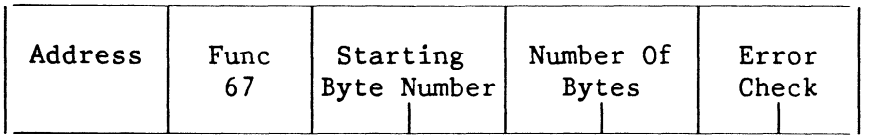
- An address 0 is not allowed as this cannot be a broadcast request.
- The function code is equal to 66.
- The starting point number is two bytes in length and may be any value less than the highest input point number available in the attached Series Six CPU. The starting point number is equal to one less than the number of the first input point whose override status is returned in the normal response to this request.
- The number of points value is two bytes in length. It specifies the number of input points whose override status are returned in the normal response. The sum of the starting point number and the number of points values must be less than or equal to the highest input point number available in the attached Series Six CPU. The high order byte of the starting point number and number of points fields is sent as the first byte in head of these fields. The low order byte is the second byte in each of these fields.

RESPONSE:

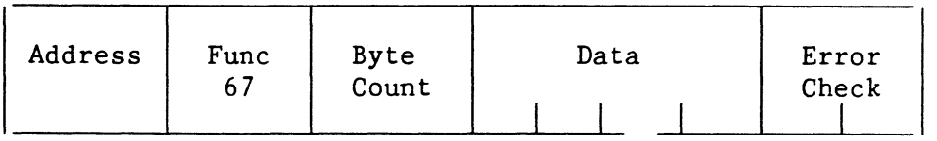
- The byte count is a binary number from 1 to 256 (0 = 256). It is the number of bytes in the data field of the normal response.
- The data field of the normal response is packed input override table data. Each byte contains the override status of eight input points. The least significant bit (LSB) of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

MESSAGE (67): **READ SCRATCH PAD MEMORY**

FORMAT:



Query



Normal Response

QUERY:

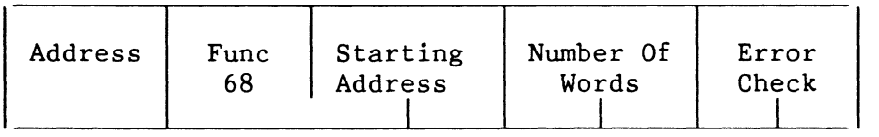
- An address of 0 is not allowed as this cannot be a broadcast request.
- The function code is equal to 67.
- The starting byte number is two bytes in length and may be any value less than or equal to the highest scratch pad memory address available in the attached Series Six CPU. The starting byte number is equal to the address of the first scratch pad memory byte returned in the normal response to this request.
- The number of bytes value is two bytes in length. It specifies the number of scratch pad memory locations (bytes) returned in the normal response. The sum of the starting byte number and the number of bytes values must be less than two plus the highest scratch pad memory address available in the attached Series Six CPU. The high order byte of the starting byte number and number of bytes fields is sent as the first byte in each of these fields. The low order byte is the second byte in each of the fields.

RESPONSE:

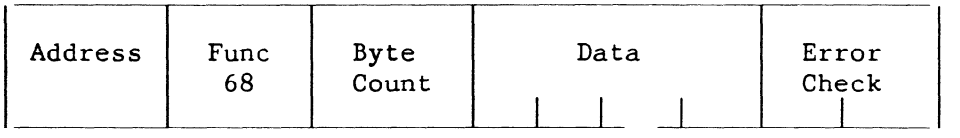
- The byte count is a binary number from 1 to 256 (0 = 256). It is the number of bytes in the data field of the normal response.
- The data field contains the contents of the scratch pad memory requested by the query. The scratch pad memory bytes are sent in order of address. The contents of the scratch pad memory byte whose address is equal to the starting byte number is sent in the first byte of the data field. The contents of the scratch pad memory byte whose address is equal to one less than the sum of the starting byte number and number of bytes values is sent in the last byte of the data field.

MESSAGE (68): **READ USER LOGIC**

FORMAT:



Query



Normal Response

QUERY:

- An address of 0 is not allowed as this cannot be a broadcast request.
- The function code is equal to 68.
- The starting address is two bytes in length and may be any value less than or equal to the highest user logic memory address available in the attached Series Six CPU. The starting address is equal to the address of the first user logic memory word returned in the normal response to this request.
- The number of words value is two bytes in length. It contains a value from 1 to 125. It specifies the number of user logic memory words returned in the normal response. The sum of the starting address and the number of words values must be less than two plus the highest user logic memory address available in the attached Series Six CPU. The high order byte of the starting address and number of words fields is sent as the first byte in each of these fields. The low order byte is the second byte in each of these fields.

RESPONSE:

- The byte count is a binary number from 2 to 250. It is the number of bytes in the data field of the normal response.
- The contents of the user logic memory are returned in the data field in order of address. The lowest address contents are returned in the first two bytes and the highest address contents are returned in the last two bytes. The address of the first user logic memory contents returned in the data field is equal to the starting address. The high order byte of each user logic memory address is sent before the low order byte of that address.

MESSAGE (69): **WRITE OUTPUT OVERRIDE TABLE**

FORMAT:

Address	Func 69	Starting Point No.	Number Of Points	Byte Count	Data	Error Check

Query

Address	Func 69	Starting Point No.	Number Of Points	Error Check

Normal Response

QUERY:

- An address of 0 indicates a broadcast request. All slave stations process a broadcast request and no response is sent.
- The value of the function code is 69.
- The starting point number is two bytes in length and may be any value less than the highest output point number available in the attached Series Six CPU. The starting point number is equal to one less than the number of the first output point whose override status is returned in the normal response to this request.
- The number of points value is two bytes in length. It specifies the number of output points whose override status are returned in the normal response. The sum of the starting point number and the number of points values must be less than or equal to the highest output point number available in the attached Series Six CPU. The high order byte of the starting point number and number of points fields is sent as the first byte in each of these fields. The low order byte is the second byte in each of these fields.

- The byte count is a binary number from 1 to 256 (0 = 256). It is the number of bytes in the data field of the normal response.

- The data field of the normal response is packed output override table data. Each byte contains the override status of eight output points. The least significant bit (LSB) of the first byte contains the override status of the output point whose number is equal to the starting point number plus one. The override status of the output points are ordered by number starting with the LSB of the first byte in the data field and ending with the most significant bit (MSB) of the last byte of the data field. If the number of points is not a multiple of eight, then the last data byte contains zeros in one to seven of its highest order bits.

RESPONSE: The description of the response fields are all covered in the description of the query fields.

NOTE

The output override table cannot be written to when the memory switch of the attached Series Six CPU is in the protect position.

MESSAGE (70): **WRITE INPUT OVERRIDE TABLE**

FORMAT:

Address	Func 70	Starting Point No.	Number Of Points	Byte Count	Data	Error Check

Query

Address	Func 70	Starting Point No.	Number Of Points	Error Check

Normal Response

QUERY:

- An address of 0 indicates a broadcast request. All slave stations process a broadcast request and no response is sent.
- The function code is equal to 70 for write input override table.
- The starting point number is two bytes in length and may be any value less than the highest input point number available in the attached Series Six CPU. The starting point number is equal to one less than the number of the first input point whose override status is returned in the normal response to this request.
- The number of points value is two bytes in length. It specifies the number of input points whose override status are returned in the normal response. The sum of the starting point number and the number of points values must be less than or equal to the highest input point number available in the attached Series Six CPU. The high order byte of the starting point number and number of points fields is sent as the first byte in head of these fields. The low order byte is the second byte in each of these fields.

- The byte count is a binary number from 1 to 256 (0 = 256). It is the number of bytes in the data field of the normal response.

- The data field of the normal response is packed input override table data. Each byte contains the override status of eight input points. The least significant bit (LSB) of the first byte contains the override status of the input point whose number is equal to the starting point number plus one. The override status of the input points are ordered by number starting with the LSB of the first byte in the data field and ending with the most significant bit (MSB) of the last byte of the data field. If the number of points is not a multiple of eight, then the last data byte contains zeros in one to seven of its highest order bits.

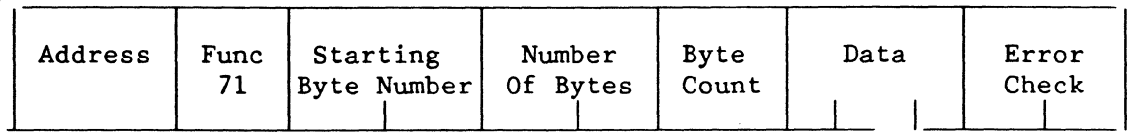
RESPONSE: The description of the response fields are covered in the description of the query fields.

NOTE

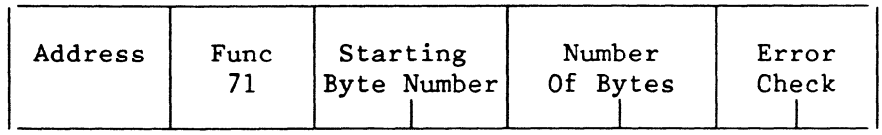
The input override table cannot be written to when the memory switch of the attached Series Six CPU is in the protect position.

MESSAGE (71): **WRITE SCRATCH PAD MEMORY**

FORMAT:



Query



Normal Response

QUERY: An address of 0 indicates a broadcast request. All slave stations process a broadcast request and no response is sent.

- The value of the function code is 71.
- The starting byte number, number of bytes, byte count, and data fields are described in the read scratch pad memory.
- The starting byte number is two bytes in length and may be any value less than or equal to the highest scratch pad memory address available in the attached Series Six CPU. The starting byte number is equal to the address of the first scratch pad memory byte returned in the normal response to this request.
- The number of bytes value is two bytes in length. It specifies the number of scratch pad memory locations (bytes) returned in the normal response. The sum of the starting byte number and the number of bytes values must be less than two plus the highest scratch pad memory address available in the attached Series Six CPU. The high order byte of the starting byte number and number of bytes fields is sent as the first byte in each of these fields. The low order byte is the second byte in each of the fields.

- The byte count is a binary number from 1 to 256 (0 = 256). It is the number of bytes in the data field of the normal response.

- The data field contains the contents of the scratch pad memory requested by the query. The scratch pad memory bytes are sent in order of address. The contents of the scratch pad memory byte whose address is equal to the starting byte number is sent in the first byte of the data field. The contents of the scratch pad memory byte whose address is equal to one less than the sum of the starting byte number and number of bytes values is sent in the last byte of the data field.

RESPONSE: The description of the response fields are covered in the query description.

REMARKS: Only 2 writes are allowed to the CPU's Scratch Pad from an external device:

A	Address 0 to 1	CPU RUN and COMMAND STATUS
B	Address 60H to 7FH	SUBROUTINE VECTOR ADDRESSES

Writing to CPU Scratch Pad Addresses 0 and 1 provides for stopping and starting the CPU. To stop the CPU, 80H is written to both locations. To start the CPU, 01H is written to both locations.

The Subroutine Vector Addresses are used in conjunction with the User Logic programs stored in the CPU. Even addresses are most significant bytes and odd addresses are least significant bytes that make up subroutine vector addresses. Subroutine 0 address starts at 60H and subroutine F address ends at 7FH.

NOTE

The scratch pad memory cannot be written to when the memory protect switch of the attached Series Six CPU is in the protect position.

When an external device writes to the CPU scratch pad, the CCM device will first place the CPU in stop mode.

MESSAGE (72): **WRITE USER LOGIC**

FORMAT:

Address	Func 72	Starting Address	Number Of Words	Byte Count	Data	Error Check

Query

Address	Func 72	Starting Address	Number Of Words	Error Check

Normal Response

QUERY:

- An address of 0 indicates a broadcast request. All slave stations process a broadcast request and no response is sent.
- The function code is equal to 72.
- The starting address is two bytes in length and may be any value less than or equal to the highest user logic memory address available in the attached Series Six CPU. The starting address is equal to the address of the first user logic memory word returned in the normal response to this request.
- The number of words value is two bytes in length. It contains a value from 1 to 125. It specifies the number of user logic memory words returned in the normal response. The sum of the starting address and the number of words values must be less than two plus the highest user logic memory address available in the attached Series Six CPU. The high order byte of the starting address and number of words fields is sent as the first byte in each of these fields. The low order byte is the second byte in each of these fields.

- The byte count is a binary number from 2 to 250. It is the number of bytes in the data field of the normal response.

- The contents of the user logic memory are sent in the data field in order of address with the lowest address contents in the first two bytes and the highest address contents in the last two bytes. The address of the first user logic memory contents returned in the data field is equal to the starting address. The high order byte of each user logic memory address is sent before the low order byte of that address.

RESPONSE: The description of the response fields are covered in the query description.

NOTE

User logic memory cannot be written to when the memory switch of the attached Series Six CPU is in the protect position.

REMARKS: The following procedure is recommended when writing to user logic:

1. Read scratch pad memory (function 67) addresses 6 thru 14 (OEH). These scratch pad addresses allow the master to check if it can load a program into the Series Six CPU attached to the CCM device, and if the program is compatible with that CPU. Scratch pad address 6 indicates the state of the memory switch (protect or write), the type of instruction set (basic or extended), and the number of registers in the attached Series Six CPU. Scratch pad addresses 11 thru 14 (OBH thru OEH) indicate the amount of user logic memory in the attached Series Six CPU.

2. Replace the first two words of the user logic program with a SUSPEND I/O and a ENDSW instruction and write the logic program to the communication module (using one or more write user logic requests.). The presence of the SUSPEND I/O and ENDSW instructions at the beginning of the user logic programs prevents the execution of a partly loaded program.

3. If the user logic program uses any subroutines write the user program subroutine vector addresses to scratch pad memory with a write scratch pad memory request.

4. Load any initial register, input/output, or input/output override values that are required by the user logic program.

5. Write the first two words of the user logic program into the first two user logic memory addresses.

NOTE

When an external device writes to the user logic the CCM device will first place the CPU in stop mode.

COMMUNICATION ERRORS

Serial link communication errors are divided into three groups:

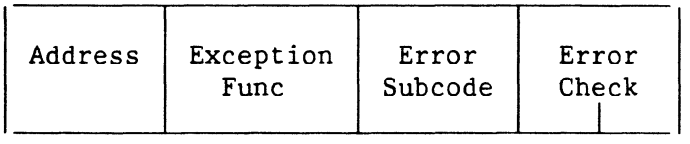
- Invalid Query Message
- Serial Link Time Outs
- Invalid Transaction

INVALID QUERY MESSAGE

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

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

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



An address of 0 is not allowed as there is no response to a broadcast request. The exception function code is equal to the sum of the function code of the query which the error response is a response to plus 128. The error subcode is equal to 1, 2, 3, or 4. The value of the subcode indicates the reason that the properly received query could not be processed.

Invalid Function Code Error Response (1)

An error response with a subcode of 1 is called an invalid function code error response. This response is sent by a slave if it receives a query whose function code is not equal to 1 through 8, 15, 16, 17, or 65 through 72.

Invalid Address Error Response (2)

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

1. The starting point number and number of points fields specify output status points or input status points that are not available in the attached Series Six CPU (returned for function codes 1, 2, 15, 65, 66, 69, 70).

2. The starting register number and number of registers fields specify registers that are not available in the attached Series Six CPU (returned for function codes 3, 4, 16).

3. The point number field specifies an output status point not available in the attached Series Six CPU (returned for function code 5).

4. The register number field specifies a register not available in the attached Series Six CPU (returned for function code 6).

5. The diagnostic code is not equal to 0, 1, or 4 (returned for function code 8).

6. The starting byte number and number of bytes fields specify a scratch pad memory address that is not available in the attached Series Six CPU (returned for function code 67).

7. The starting byte number and number of bytes fields specify a write to a scratch pad memory address other than addresses 0, 1, 60H thru 7FH, and 5CH thru 5FH (returned for function code 71).

8. The starting address and number of words fields specify a user logic memory address not available in the attached Series Six CPU (returned for function codes 68, 72).

Invalid Data Value Error Response (3)

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

The first byte of the data field is not equal to 0 or 255 (FFH) or the second byte of the data field is not equal to 0 for the force single output request (functon code 5) or the initiate communication restart request (function code 8, diagnostic code 1).

NOTE

Although there are no checks for invalid data when the subroutine vector addresses are written to scratch pad memory addresses 96 (60H) to 127 (7FH), a subroutine vector address should never be set equal to 0.

Query Processing Failure Error Response (4)

An error response with a subcode of 4 is called a query processing failure response. This error response is sent by a CCM device if it properly receives a query but communication between the associated Series Six CPU and the CCM device fails.

SERIAL LINK TIME-OUT

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

- After sending a query message, the master should wait the length of the turn-around time before assuming that the slave did not respond to its request. See Table 5.1 for turn-around times using the various function codes.

- The master must also consider the activity occurring on the CCM device port to which the master is not connected. If there is activity occurring on the J2 port when an RTU query message is sent to the J1 port, the query message will not be processed until after the J2 port becomes idle. The time it takes for the port to become idle must be allowed for by the master to prevent the master from timing out. More information on dual port activity with the CCM device can be found in Chapter 2, section, Simultaneous Port Operations.

INVALID TRANSACTIONS

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

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

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

CHAPTER 6
COMMUNICATION APPLICATIONS

INTRODUCTION

This chapter includes several application programs for using the features of the CCM2 and CCM3 (CCM) communications module. The programs present basic programming techniques which the user can tailor to his specific needs. The following programs, applicable to the CCM are included:

- Using the CCM Status Byte for SCREQ Interlocks and Sequencing
- Using the CCM Diagnostic Status Words
- Multidrop Polling Routine

TITLE: USING THE CCM STATUS BYTE FOR SCREQ INTERLOCKS AND SEQUENCING

INTRODUCTION: The CCM Status Byte consists of 8 bits of status information as shown below which are transferred from the CCM to CPU inputs I1009-I1016 during each CCM communications window.

Input No.	Bit	Definition
I1009	1	CCM Port Busy with [SCREQ]
I1010	2	[SCREQ] complete without error
I1011	3	[SCREQ] complete with error
I1012	4	Externally initiated READ occurred successfully
I1013	5	Externally initiated WRITE occurred successfully
I1014	6	Q response sent
I1015	7	Spare (always 0)
I1016	8	CCM-CPU communications OK

Bit 1 is set to a 1 when the CCM accepts a port command from the CPU and resets to 0 upon completion.

Bits 2-6 are pulsed by the CCM when the condition causing the status change occurs. The pulse function ensures that the bit will be set to 1 for 3 windows minimum then will be set to 0 for 3 windows minimum. The pulse function for a particular status bit will be completed before another pulse function for the same status bit is activated.

Bit 8 is explained in the Theory of Operation section later in this application.

This instructional program will show how bits 1 and 8 can be used as SCREQ interlocks to prevent improper activation of the SCREQ function and how bits 1, 2, 4, 5, and 6 can be used to sequence a series of SCREQ functions. Bit 3 indicates an error in the execution of an SCREQ. An example program for using this bit is presented later in the chapter.

EQUIPMENT USED: 1 - CPU with extended functions
1 - CCM (all SCREQs in this example are internal commands)
Series Six I/O (optional)

MODULE CONFIGURATION: Any valid configuration is acceptable since the SCREQs in this program are internal.

THEORY OF OPERATION:

SEQUENCER

This program sequentially executes 2 internal requests: 06004, Load QAB and 06007, Read QAB. In the first request, bytes 0-3 of the QAB are loaded with the contents of R0050 and R0051, then in the second request the same QAB bytes, 0-3, are read into R0052 and R0053. A shift register, which is reset and initialized manually and advanced by the pulsing of bit 2 of the status byte (I1010), controls the sequencing. The shift register consists of a block of outputs O0001-O0016. When input I0001 is active the shift register is first cleared and then output O0001 is set to a 1:

```
00016                          00001
| 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1|
```

This triggers the execution of the first SCREQ, 06004, Load QAB (which loads the QAB from registers R0050, R0051). Upon completion of 06004, bit 2 (I1010, [SCREQ] complete without error) of the status byte pulses on and off which triggers the shift register to shift 1 bit to the left.

```
00016                          00001
| 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0|
```

This triggers the execution of the second SCREQ, 06007, Read QAB (which reads the QAB to registers R0052, R0053). Upon completion of 06007, Bit 2 pulses on and off and triggers the shift register to shift 1 bit to the left again. Since there are no more SCREQs in the program the sequence stops. With this type of shift register as many as 16 different SCREQs could be sequenced. Larger shift registers can be programmed using the extended shift functions.

This program consists of internal commands only; when port commands are included, bits 1, 4, 5, and 6 can be used as triggers to sequence SCREQs as well as other functions of the program.

INTERLOCKS

Two interlocks are used in this program. One of the interlocks--bit 1, I1009, of the status byte indicating CCM busy with [SCREQ]--can be used as a normally closed contact permitting power flow to the [SCREQ] function only when a CCM2 port is not busy.

The other interlock is based on bit 8 of the status byte, I1016, indicating CCM-CPU communications is OK. This bit, however, cannot be used directly as is the CCM busy bit. Bit 8 is set to a 1 if the CCM passes power up indicating good communication between CCM and CPU. After power up, a 1 is written to bit 8 during each window. If communications between CCM and CPU fail, a 1 is not written because no window occurs.

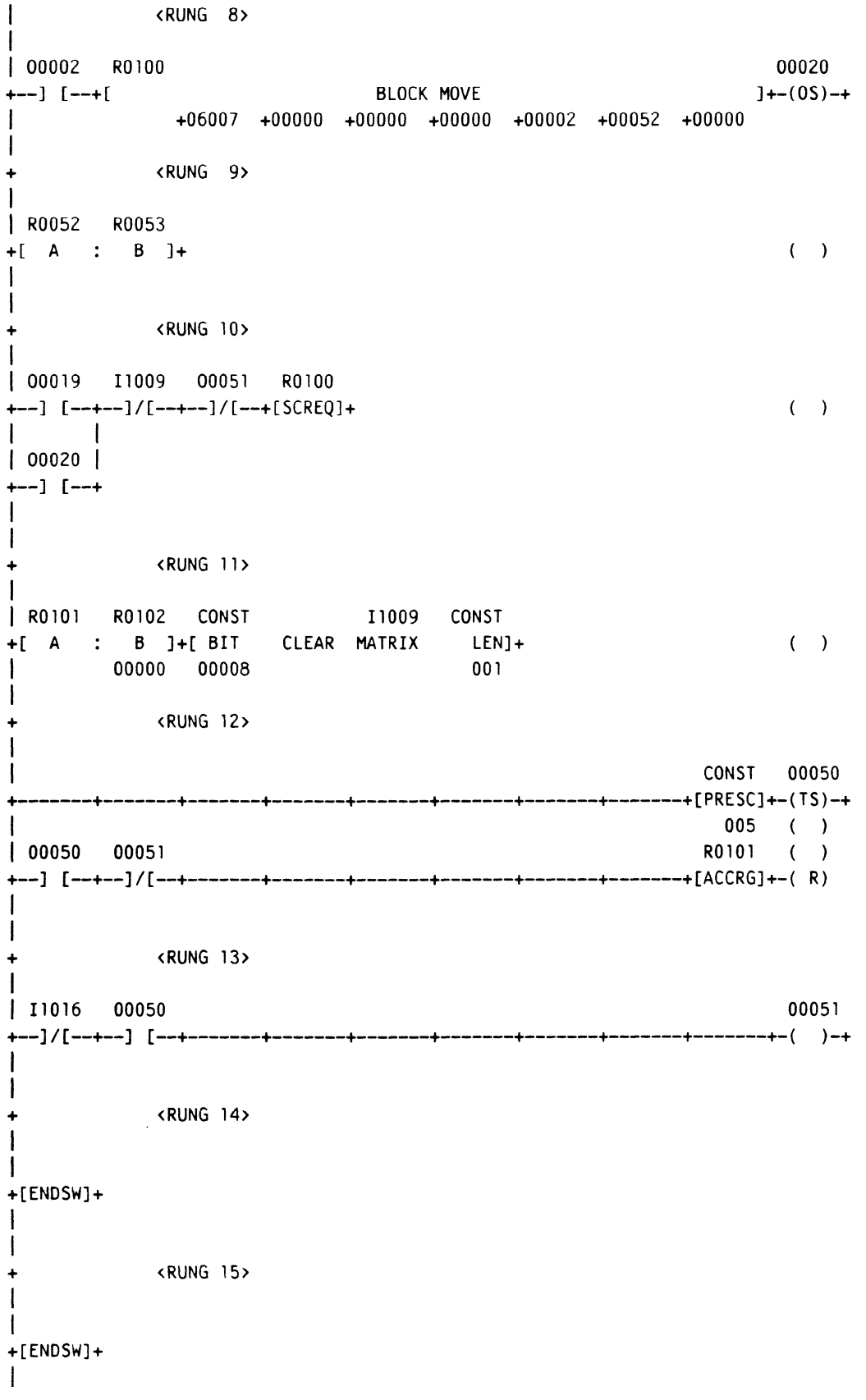
To use bit 8 as an interlock, periodically reset the bit to a 0, wait a period of time, and check to see if the bit has returned to a 1. If the bit has not been set to a 1 again, then communications between the CCM and CPU has failed. The length of time needed to wait must be longer than the time required by the longest transmission: Rule of Thumb:

$$\frac{\text{No. Char. x 10}}{\text{Data Rate (bits/sec)}} + \text{Longest Response Timeout}$$

PROGRAM 1

(For a rung by rung explanation see the annotation following the program.)

```
|              <RUNG  0>
+[NO OP]+                                                              (  )
|
|
+              <RUNG  1>
|
| CONST       R0006   R0006
+[  A   MOVE    B  ]+[  STATUS  ]+                                      (  )
| 0000
|
+              <RUNG  2>
|
| I0001                                                                00017
+--] [--+-------+-------+-------+-------+-------+-------+-------+-------+-(OS)-+
|
|
+              <RUNG  3>
|
| 00017   CONST           00001
+--] [--+[  A     MOVE      B   ]+                                       (  )
|        +00001
|
|
+              <RUNG  4>
|
| I1010                                                                00018
+--] [--+-------+-------+-------+-------+-------+-------+-------+-------+-(OS)-+
|
|
+              <RUNG  5>
|
| 00018  00001
+--] [--+[SHIFT]+                                                       (  )
|
|
+              <RUNG  6>
|
| 00001   R0100                                                        00019
+--] [--+[                        BLOCK MOVE                          ]+-(OS)-+
|               +06004  +00000  +00000  +00000  +00002  +00050  +00000
|
+              <RUNG  7>
|
| R0050   R0051
+[  A   :   B  ]+                                                       (  )
|
```


<RUNG 8>
00002 R0100
BLOCK MOVE
00020
+06007 +00000 +00000 +00000 +00002 +00052 +00000
<RUNG 9>
R0052 R0053
A : B
<RUNG 10>
00019 I1009 00051 R0100
SCREQ
00020
<RUNG 11>
R0101 R0102 CONST I1009 CONST
A : B BIT CLEAR MATRIX LEN
00000 00008 001
<RUNG 12>
CONST 00050
PRESC
TS
005
00050 00051 R0101
ACCRG
R
<RUNG 13>
I1016 00050 00051
<RUNG 14>
ENDSW
<RUNG 15>
ENDSW

ANNOTATION OF PROGRAM:

Rung No. 1 ensures the CCM windows are enabled by zeroing the [STATUS] function register. (Note: A value of zero in the [STATUS] function enables DPU as well as CCM windows).

Rung No. 2 triggers the shift register initialization.

Rung No. 3 places a 1 in O0001 turning it ON and causing the first SCREQ to be executed.

Rung No. 4 triggers the [SHIFT] function when bit 2 (I1010) of the status byte, indicating [SCREQ] complete without error, pulses ON and OFF.

Rung No. 5 shifts one bit to the left when triggered.

Rung No. 6 loads the [SCREQ] registers for command 06004 when output O0001 is ON. This request loads QAB bytes 0-3 with the contents of the 2 registers R0050, R0051.

Rung No. 7 allows for loading R0050 and R0051 from the user program without going to the register tables. (For user convenience only).

Rung No. 8 loads the [SCREQ] registers for command 06007 when output O0002 is ON. This request reads the contents of QAB bytes 0-3 into 2 registers R0052, R0053.

Rung No. 9 permits monitoring R0052 and R0053 from the user program without using the register tables. When commands 06004 and 06007 in this example are executed in sequence, the contents of R0050 and R0051 are copied into the QAB and then copied back out to R0052 and R0053. (For user convenience only).

Rung No. 10 is the [SCREQ] rung containing permissive contacts O0019 and O0020 from the [BLOCK MOVE] functions and containing interlocks I1009 (bit 1 of the status byte) and O0051 (derived from I1016, which is bit 8 of the status byte, as shown in rungs 11, 12, and 13).

Rung No. 11 is used with rungs 12 and 13 to provide an interlock for the [SCREQ] rung indicating the status of CPU/CCM communications. Rung 11 zeroes bit 8 of the status byte when the accumulator register of the 5 second timer in rung 12 is 0. The theory of programming this interlock is explained earlier in this application in the section, Interlocks.

Rung No. 12 is a timer which runs continuously as long as CPU/CCM communications are good. Its length is determined by the longest serial transmission likely to occur in the application.

Rung No. 13 signals a failure in CPU/CCM communications if bit 8 does not return to a 1 before the timer times out. Output O0051 will turn on if communications have failed; and in rung 10 power to the [SCREQ] function will be broken.

TITLE: USING THE CCM DIAGNOSTIC STATUS WORDS

INTRODUCTION: The CCM Diagnostic Status Words (defined in Table 2.20, CCM Diagnostic Status Word Definition) are powerful tools which allow the user to monitor and analyze SCREQ or serial port errors. Unlike the CCM status byte which is automatically transferred from the CCM to the CPU once each window, the Diagnostic Status Words must be read from the CCM using an SCREQ.

This application program shows the user how to:

- Read the host CCM Diagnostic Status Words
- Clear the host CCM Diagnostic Status Words
- Read the remote CCM Diagnostic Status Words
- Clear the remote CCM Diagnostic Status Words
- Analyze error codes of Diagnostic Status Words 1 and 13

EQUIPMENT USED: 2 - CPUs with extended functions
2 - CCMs
Series Six I/O (optional)
CCM to CCM, RS-232D cable configured as shown in Chapter 2, Section, Cable and Connector Specifications.

CCM AND CPU CONFIGURATION:

CCM Software Configuration
- R0247 = 00038 (0026) for both Series Six CPUs
- RS-232D
- Peer-to-peer protocol
- 19.2 Kbps
- 0 msec turn-around delay
- No parity

CPU ID Configuration
- Host CPU ID = 1
- Remote CPU ID = 2

All port SCREQs use Port J1

THEORY OF OPERATION: There are 2 main parts to this application program which resides in the host CPU:

Trial SCREQ - which emulates a port SCREQ between the host and remote devices occurring in a user program.

Diagnostic Status Word SCREQs - which read and clear the Diagnostic Status Words in both the host and remote device.

The trial SCREQ is used as a vehicle to introduce errors in an SCREQ to cause the Diagnostic Status Word SCREQs to display the Diagnostic Status Words. When a communication error occurs, an SCREQ is activated to read the host Series Six Diagnostic Status Words to registers R0201 - R0220. If further analysis is needed, the remote Series Six Diagnostic Status Words can be read to registers R0201 - R0220.

To illustrate the usefulness of the Diagnostic Status Words, several trial SCREQs using command 06101, Read from Target to Source Registers were made. Intentional errors were introduced into the SCREQ registers or the communication line to simulate errors in the user program.

Table 6.1 shows the error introduced into each trial SCREQ and the resulting Diagnostic Status Words from R0201 - R0220 for the host Series Six PLC and the remote Series Six PLC where applicable. The error code definitions for Diagnostic Status Words 1 (Serial Port Errors, Table 2.20) and (SCREQ Error Codes for Status Word 13, Table 2.21) are also included for each trial.

Table 6.1 TRIAL SCREQS USING COMMAND 06101, READ FROM TARGET TO SOURCE REGISTERS

TRIAL NUMBER	ERROR INTRODUCED	CONTENTS OF SCREQ REG. USED IN TRIAL
1	NONE	Rn: 6101 Rn+1: 2 Rn+2: 1 Rn+3: 52 Rn+4: 2 Rn+5: 50

HOST (SOURCE) DIAGNOSTIC STATUS WORDS (DSW) FROM R0201 - R0220 AND ERROR DEFINITIONS

R0201 DSW1	R0202 DSW2	R0203 DSW3	R0204 DSW4	R0205 DSW5	R0206 DSW6	R0207 DSW7	R0208 DSW8	R0209 DSW9	R0210 DSW10
0	1	0	0	0	0	0	0	0	38

R0211 DSW11	R0212 DSW12	R0213 DSW13	R0214 DSW14	R0215 DSW15	R0216 DSW16	R0217 DSW17	R0218 DSW18	R0219 DSW19	R0220 DSW20
0	3	0	0	0	0	0	0	0	0

ERROR DEFINITIONS (DSW 1: PORT ERRORS, TABLE 2.20)
(DSW 13: SCREQ ERRORS, TABLE 2.21)

DSW 1: NONE

DSW 13: NONE

REMOTE (TARGET) DIAGNOSTIC STATUS WORDS (DSW) FROM R0201 - R0220 AND ERROR DEFINITIONS

R0201 DSW1	R0202 DSW2	R0203 DSW3	R0204 DSW4	R0205 DSW5	R0206 DSW6	R0207 DSW7	R0208 DSW8	R0209 DSW9	R0210 DSW10
0	2	0	0	0	0	0	0	0	38

R0211 DSW11	R0212 DSW12	R0213 DSW13	R0214 DSW14	R0215 DSW15	R0216 DSW16	R0217 DSW17	R0218 DSW18	R0219 DSW19	R0220 DSW20
0	2	0	0	0	0	0	0	0	0

ERROR DEFINITIONS (DSW 1: PORT ERRORS, TABLE 2.20)
(DSW 13: SCREQ ERRORS, TABLE 2.21)

DSW 1: NONE

DSW 13: NONE

Table 6.1 TRIAL SCREQS USING COMMAND 06101, READ FROM TARGET TO SOURCE REGISTERS (Continued)

TRIAL NUMBER	ERROR INTRODUCED	CONTENTS OF SCREQ REG. USED IN TRIAL
2	INVALID COMMAND	Rn: (5000) Rn+1: 2 Rn+2: 1 Rn+3: 52 Rn+4: 2 Rn+5: 50

HOST (SOURCE) DIAGNOSTIC STATUS WORDS (DSW) FROM R0201 - R0220 AND ERROR DEFINITIONS

R0201 DSW1	R0202 DSW2	R0203 DSW3	R0204 DSW4	R0205 DSW5	R0206 DSW6	R0207 DSW7	R0208 DSW8	R0209 DSW9	R0210 DSW10
0	0	0	0	0	0	0	0	0	38

R0211 DSW11	R0212 DSW12	R0213 DSW13	R0214 DSW14	R0215 DSW15	R0216 DSW16	R0217 DSW17	R0218 DSW18	R0219 DSW19	R0220 DSW20
0	3	1	100	5000	2	1	52	2	50

ERROR DEFINITIONS (DSW 1: PORT ERRORS, TABLE 2.20)
(DSW 13: SCREQ ERRORS, TABLE 2.21)

DSW 1: NONE

DSW 13: 1 - Command number is invalid.

REMOTE (TARGET) DIAGNOSTIC STATUS WORDS (DSW) FROM R0201 - R0220 AND ERROR DEFINITIONS

R0201 DSW1	R0202 DSW2	R0203 DSW3	R0204 DSW4	R0205 DSW5	R0206 DSW6	R0207 DSW7	R0208 DSW8	R0209 DSW9	R0210 DSW10
0	1	0	0	0	0	0	0	0	38

R0211 DSW11	R0212 DSW12	R0213 DSW13	R0214 DSW14	R0215 DSW15	R0216 DSW16	R0217 DSW17	R0218 DSW18	R0219 DSW19	R0220 DSW20
0	2	0	0	0	0	0	0	0	0

ERROR DEFINITIONS (DSW 1: PORT ERRORS, TABLE 2.20)
(DSW 13: SCREQ ERRORS, TABLE 2.21)

DSW 1: NONE

DSW 13: NONE

Table 6.1 TRIAL SCREQS USING COMMAND 06101, READ FROM TARGET TO SOURCE REGISTERS (Continued)

TRIAL NUMBER	ERROR INTRODUCED	CONTENTS OF SCREQ REG. USED IN TRIAL
3	INVALID MEMORY ADDRESS	Rn: 6101 Rn+1: 2 Rn+2: 1 Rn+3:(2000) Rn+4: 2 Rn+5: 50

HOST (SOURCE) DIAGNOSTIC STATUS WORDS (DSW) FROM R0201 - R0220 AND ERROR DEFINITIONS

R0201 DSW1	R0202 DSW2	R0203 DSW3	R0204 DSW4	R0205 DSW5	R0206 DSW6	R0207 DSW7	R0208 DSW8	R0209 DSW9	R0210 DSW10
22	0	0	1	0	3	0	0	0	38

R0211 DSW11	R0212 DSW12	R0213 DSW13	R0214 DSW14	R0215 DSW15	R0216 DSW16	R0217 DSW17	R0218 DSW18	R0219 DSW19	R0220 DSW20
0	3	16	100	6101	2	1	2000	2	50

ERROR DEFINITIONS (DSW 1: PORT ERRORS, TABLE 2.20)
(DSW 13: SCREQ ERRORS, TABLE 2.21)

DSW 1: 22 - CCM expected to receive ACK or NAK and did not receive either one.
DSW 13: 16 - CCM did not receive ACK or NAK that it expected to receive.

REMOTE (TARGET) DIAGNOSTIC STATUS WORDS (DSW) FROM R0201 - R0220 AND ERROR DEFINITIONS

R0201 DSW1	R0202 DSW2	R0203 DSW3	R0204 DSW4	R0205 DSW5	R0206 DSW6	R0207 DSW7	R0208 DSW8	R0209 DSW9	R0210 DSW10
4	1	0	1	0	3	0	0	0	38

R0211 DSW11	R0212 DSW12	R0213 DSW13	R0214 DSW14	R0215 DSW15	R0216 DSW16	R0217 DSW17	R0218 DSW18	R0219 DSW19	R0220 DSW20
0	2	0	0	0	0	0	0	0	0

ERROR DEFINITIONS (DSW 1: PORT ERRORS, TABLE 2.20)
(DSW 13: SCREQ ERRORS, TABLE 2.21)

DSW 1: 4 - An external device attempted to access more data than is available in a particular memory type.

DSW 13: NONE

Table 6.1 TRIAL SCREQS USING COMMAND 06101, READ FROM TARGET TO SOURCE REGISTERS (Continued)

TRIAL NUMBER	ERROR INTRODUCED	CONTENTS OF SCREQ REG. USED IN TRIAL
4	INVALID DATA LENGTH	Rn: 6101 Rn+1: 2 Rn+2: 1 Rn+3: 52 Rn+4:(2000) Rn+5: 50

R0201 DSW1	R0202 DSW2	R0203 DSW3	R0204 DSW4	R0205 DSW5	R0206 DSW6	R0207 DSW7	R0208 DSW8	R0209 DSW9	R0210 DSW10
0	0	0	0	0	0	0	0	0	38

R0211 DSW11	R0212 DSW12	R0213 DSW13	R0214 DSW14	R0215 DSW15	R0216 DSW16	R0217 DSW17	R0218 DSW18	R0219 DSW19	R0220 DSW20
0	3	2	100	6101	2	1	52	2000	50

ERROR DEFINITIONS (DSW 1: PORT ERRORS, TABLE 2.20)
(DSW 13: SCREQ ERRORS, TABLE 2.21)

DSW 1: NONE

DSW 13: 2 - Source address and/or data length is invalid.

REMOTE (TARGET) DIAGNOSTIC STATUS WORDS (DSW) FROM R0201 - R0220 AND ERROR DEFINITIONS

R0201 DSW1	R0202 DSW2	R0203 DSW3	R0204 DSW4	R0205 DSW5	R0206 DSW6	R0207 DSW7	R0208 DSW8	R0209 DSW9	R0210 DSW10
0	1	0	0	0	0	0	0	0	38

R0211 DSW11	R0212 DSW12	R0213 DSW13	R0214 DSW14	R0215 DSW15	R0216 DSW16	R0217 DSW17	R0218 DSW18	R0219 DSW19	R0220 DSW20
0	2	0	0	0	0	0	0	0	0

ERROR DEFINITIONS (DSW 1: PORT ERRORS, TABLE 2.20)
(DSW 13: SCREQ ERRORS, TABLE 2.21)

DSW 1: NONE

DSW 13: NONE

Table 6.1 TRIAL SCREQS USING COMMAND 06101, READ FROM TARGET TO SOURCE REGISTERS (Continued)

TRIAL NUMBER	ERROR INTRODUCED	CONTENTS OF SCREQ REG. USED IN TRIAL
5	DISCONNECTED LINE AT SOURCE DEVICE	Rn: 6101 Rn+1: 2 Rn+2: 1 Rn+3: 52 Rn+4: 2 Rn+5: 50

HOST (SOURCE) DIAGNOSTIC STATUS WORDS (DSW) FROM R0201 - R0220 AND ERROR DEFINITIONS

R0201 DSW1	R0202 DSW2	R0203 DSW3	R0204 DSW4	R0205 DSW5	R0206 DSW6	R0207 DSW7	R0208 DSW8	R0209 DSW9	R0210 DSW10
26	0	0	0	0	0	0	0	0	38

R0211 DSW11	R0212 DSW12	R0213 DSW13	R0214 DSW14	R0215 DSW15	R0216 DSW16	R0217 DSW17	R0218 DSW18	R0219 DSW19	R0220 DSW20
0	3	19	100	6101	2	1	52	2	50

ERROR DEFINITIONS (DSW 1: PORT ERRORS, TABLE 2.20)
(DSW 13: SCREQ ERRORS, TABLE 2.21)

DSW 1: 26 - A time-out occurred during an attempt to transmit on a port due to CTS being in an inactive state too long.
DSW 13: 19 - A time-out occurred on the serial link during the execution of an SCREQ.

REMOTE (TARGET) DIAGNOSTIC STATUS WORDS (DSW) FROM R0201 - R0220 AND ERROR DEFINITIONS

R0201 DSW1	R0202 DSW2	R0203 DSW3	R0204 DSW4	R0205 DSW5	R0206 DSW6	R0207 DSW7	R0208 DSW8	R0209 DSW9	R0210 DSW10
-	-	-	-	-	-	-	-	-	-

R0211 DSW11	R0212 DSW12	R0213 DSW13	R0214 DSW14	R0215 DSW15	R0216 DSW16	R0217 DSW17	R0218 DSW18	R0219 DSW19	R0220 DSW20
-	-	-	-	-	-	-	-	-	-

ERROR DEFINITIONS (DSW 1: PORT ERRORS, TABLE 2.20)
(DSW 13: SCREQ ERRORS, TABLE 2.21)

DSW 1: N/A

DSW 13: N/A

Table 6.1 TRIAL SCREQS USING COMMAND 06101, READ FROM TARGET TO SOURCE REGISTERS (Continued)

TRIAL NUMBER	ERROR INTRODUCED	CONTENTS OF SCREQ REG. USED IN TRIAL
6	DISCONNECTED LINE AT TARGET DEVICE	Rn: 6101 Rn+1: 2 Rn+2: 1 Rn+3: 52 Rn+4: 2 Rn+5: 50

HOST (SOURCE) DIAGNOSTIC STATUS WORDS (DSW) FROM R0201 - R0220 AND ERROR DEFINITIONS

R0201 DSW1	R0202 DSW2	R0203 DSW3	R0204 DSW4	R0205 DSW5	R0206 DSW6	R0207 DSW7	R0208 DSW8	R0209 DSW9	R0210 DSW10
25	0	0	1	0	0	0	0	0	38

R0211 DSW11	R0212 DSW12	R0213 DSW13	R0214 DSW14	R0215 DSW15	R0216 DSW16	R0217 DSW17	R0218 DSW18	R0219 DSW19	R0220 DSW20
0	3	12	100	6101	2	1	52	2	50

ERROR DEFINITIONS (DSW 1: PORT ERRORS, TABLE 2.20)
(DSW 13: SCREQ ERRORS, TABLE 2.21)

DSW 1: 25 - Communication was aborted when the CCM did not receive a valid response to a peer enquire after 32 attempts.
DSW 13: 12 - Communication, initiated by an SCREQ, was aborted when the CCM did not receive a valid acknowledge to a peer enquire sequence after 32 attempts.

REMOTE (TARGET) DIAGNOSTIC STATUS WORDS (DSW) FROM R0201 - R0220 AND ERROR DEFINITIONS

R0201 DSW1	R0202 DSW2	R0203 DSW3	R0204 DSW4	R0205 DSW5	R0206 DSW6	R0207 DSW7	R0208 DSW8	R0209 DSW9	R0210 DSW10
-	-	-	-	-	-	-	-	-	-

R0211 DSW11	R0212 DSW12	R0213 DSW13	R0214 DSW14	R0215 DSW15	R0216 DSW16	R0217 DSW17	R0218 DSW18	R0219 DSW19	R0220 DSW20
-	-	-	-	-	-	-	-	-	-

ERROR DEFINITIONS (DSW 1: PORT ERRORS, TABLE 2.20)
(DSW 13: SCREQ ERRORS, TABLE 2.21)

DSW 1: N/A
DSW 13: N/A

PROGRAM 2

Using the Program:

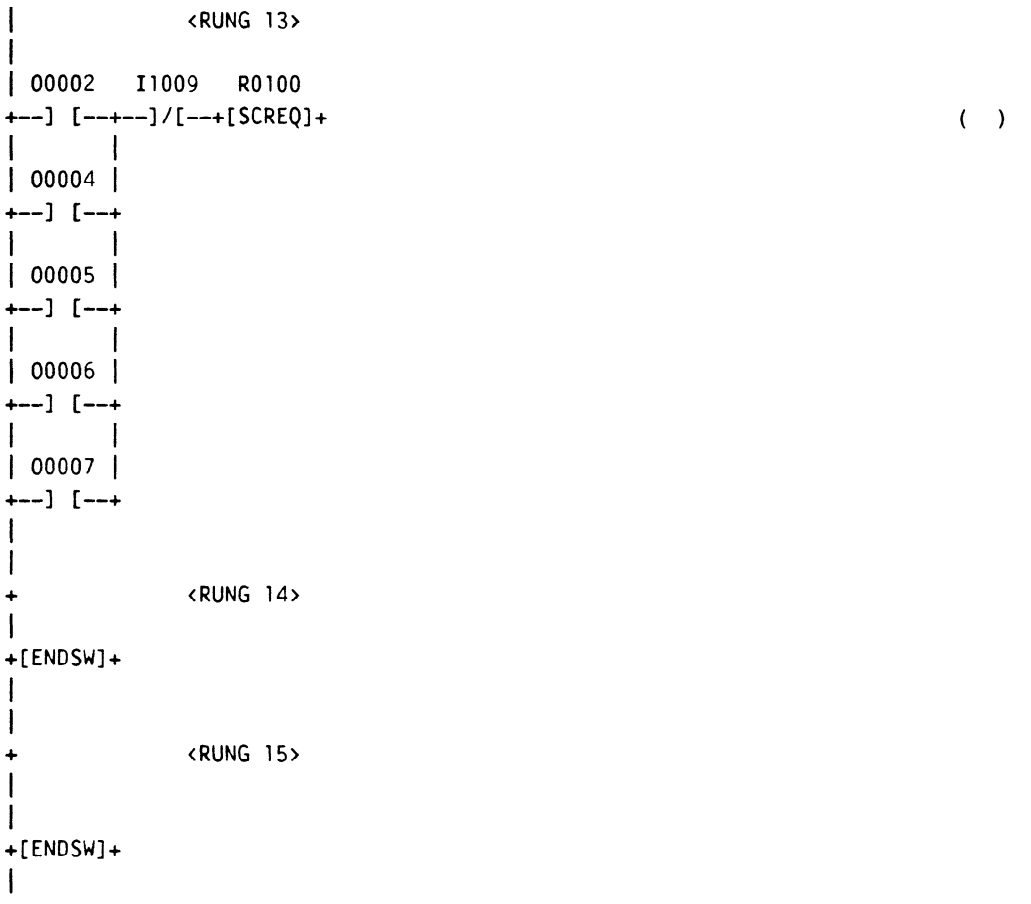
- Enter trial SCREQ register values in [BLOCK MOVE] in rung No. 3.
- Closing I0001 initiates trial SCREQ
- Closing I0002 reads host Diagnostic Status Words
- Closing I0003 clears host Diagnostic Status Words
- Closing I0004 reads remote Diagnostic Status Words
- Closing I0005 clears remote Diagnostic Status Words

(For a rung by rung explanation see the annotation following the program).

```
|            <RUNG  0>
|
+[NO OP]+                                                          (  )
|
|
+            <RUNG  1>
|
| CONST      R0006   R0006
+[  A   MOVE   B  ]+[  STATUS  ]+                                   (  )
| 0000
|
|
+            <RUNG  2>
|
| I0001                                                            00001
+--] [--+-------+-------+-------+-------+-------+-------+-------+-------+-(  )-+
|
|
+            <RUNG  3>
|
| 00001   R0100                                                    00002
+--] [--+[                      BLOCK MOVE                        ]+-(OS)-+
|              +06101  +00002  +00001  +00052  +00002  +00050  +00000
|
+            <RUNG  4>
|
| R0050   R0051
+[  A   :   B   ]+                                                 (  )
|
|
+            <RUNG  5>
|
| I1011                                                            00003
+--] [--+-------+-------+-------+-------+-------+-------+-------+-------+-(  )-+
|
```

```
|            <RUNG  6>
|
| O0003   R0100                                                    O0004
+--] [--+[                        BLOCK MOVE                    ]+-(OS)-+
|       |       +06003  +00000  +00000  +00001  +00020  +00201  +00000
| I0002 |
+--] [--+
|
|
+            <RUNG  7>
|
| I0003   R0100                                                    O0005
+--] [--+[                          BLOCK MOVE                  ]+-(OS)-+
|              +06002  +00000  +00000  +00000  +00000  +00000  +00000
|
+            <RUNG  8>
|
| I0004   R0100                                                    O0006
+--] [--+[                          BLOCK MOVE                  ]+-(OS)-+
|              +06101  +00002  +00009  +00001  +00020  +00201  +00000
|
+            <RUNG  9>
|
| I0005   R0100                                                    O0007
+--] [--+[                          BLOCK MOVE              *   ]+-(OS)-+
|              +06111  +00002  +00009  +00001  +00020  +00070  +00000
|
+            <RUNG 10>
|
| R0201   R0202   R0203   R0204   R0205   R0206   R0207   R0208
+[  A   :   B  ]+[  A   :   B  ]+[  A   :   B  ]+[  A   :   B  ]+        ( )
|
|
+            <RUNG 11>
|
| R0209   R0210   R0211   R0212   R0213   R0214   R0215   R0216
+[  A   :   B  ]+[  A   :   B  ]+[  A   :   B  ]+[  A   :   B  ]+        ( )
|
|
+            <RUNG 12>
|
| R0217   R0218   R0219   R0220
+[  A   :   B  ]+[  A   :   B  ]+                                       ( )
|
```

* R0070-R0089 must all contain zeroes.



ANNOTATION OF PROGRAM:

Rung number 1 ensures the CCM windows are enabled by zeroing the [STATUS] function register. (Note: A value of zero in the [STATUS] function enables DPU as well as CCM windows).

Rung number 2 initiates a trial SCREQ when I0001 is closed.

Rung number 3 loads the SCREQ registers for the trial SCREQ. Each trial SCREQ in this example was based on command 06101, Read Target (Registers R0052 and R0053) to Source Registers (R0050 and R0051).

Rung number 4 is for display purposes only. A successful execution of command 06101 as explained in rung number 3 will read the contents of Target registers R0052 and R0053 to source registers R0050 and R0051 and will be displayed in this rung.

Rung number 5 initiates the SCREQ for reading the host Diagnostic Status Words to R0201 - R0220 when I1011 of the Status Byte, SCREQ complete with error, is pulsed on and off.

Rung number 6 loads the SCREQ registers for SCREQ 06003, Read Diagnostic Status Words, when either I1011 is pulsed or I0002 is closed.

Rung number 7 loads the SCREQ registers for SCREQ 06002, Clear Diagnostic Status Words, when I0003 is closed.

Rung number 8 loads the SCREQ registers for SCREQ 06101 which is used in this case to read the remote Series Six Diagnostic Status Words when I0004 is closed.

Rung number 9 loads the SCREQ registers for SCREQ command 06111, which is used in this case to write zeroes to the remote Series Six Diagnostic Status Words, when I0005 is closed.

Rung numbers 10–12 are used to display the contents of the Diagnostic Status Words.

Rung number 13 is the SCREQ rung with permissive contacts for activation and with the interlock I1009 to prevent execution of the SCREQ when a CCM port is busy.

TITLE: MULTIDROP POLLING ROUTINE

INTRODUCTION: A multidrop configuration is one in which a Series Six PLC or host computer is a master controller and two or more Series Six PLCs are slaves to the master controller.

The master controller typically receives data from the slave devices and transmits control information back to them. A polling routine, whereby each slave is either written to or read from in succession, is often used to pass information between master and slaves, and that is the type of routine shown in this example.

EQUIPMENT USED: 2 or more CPUs with extended functions
2 or more CCMs
Series Six I/O (optional)
RS-422 multidrop cable configured as shown in Chapter 2 section: Cable and Connector Specifications.

CCM AND CPU CONFIGURATION:

CCM Software Configuration	CPU ID Configuration
Master R0247 = 00006 (0006H)	Master CPU ID = 1
Slave R0247 = 00022 (0016H)	Slave 1 CPU ID = 2
• RS-422	Slave 2 CPU ID = 3
• Master-Slave Protocol	Slave 3 CPU ID = 4
• 19.2 Kbps	
• 0 msec turn-around delay	
• No parity	

All port SCREQs use Port J1.

THEORY OF OPERATION: A sequence of SCREQS that reads 10 registers from each slave is triggered every 5 seconds. A shift register is used for controlling the sequence of requests. The operation of a shift register for sequencing is explained in the Theory of Operation section in the application program, Using the CCM Status Byte for SCREQ Interlocks and Sequencing.

The block diagram illustrates the transfer of registers from slaves to master.

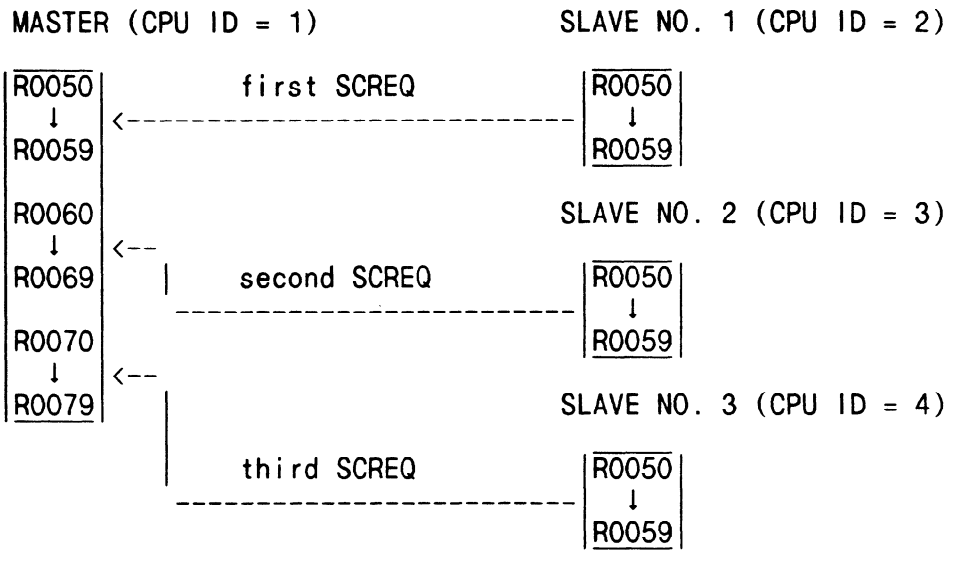


Figure 6.1 REGISTER TRANSFER FROM SLAVE TO MASTER

The master and slaves are identified by their CPU ID number which is configured through the CPU Scratchpad.

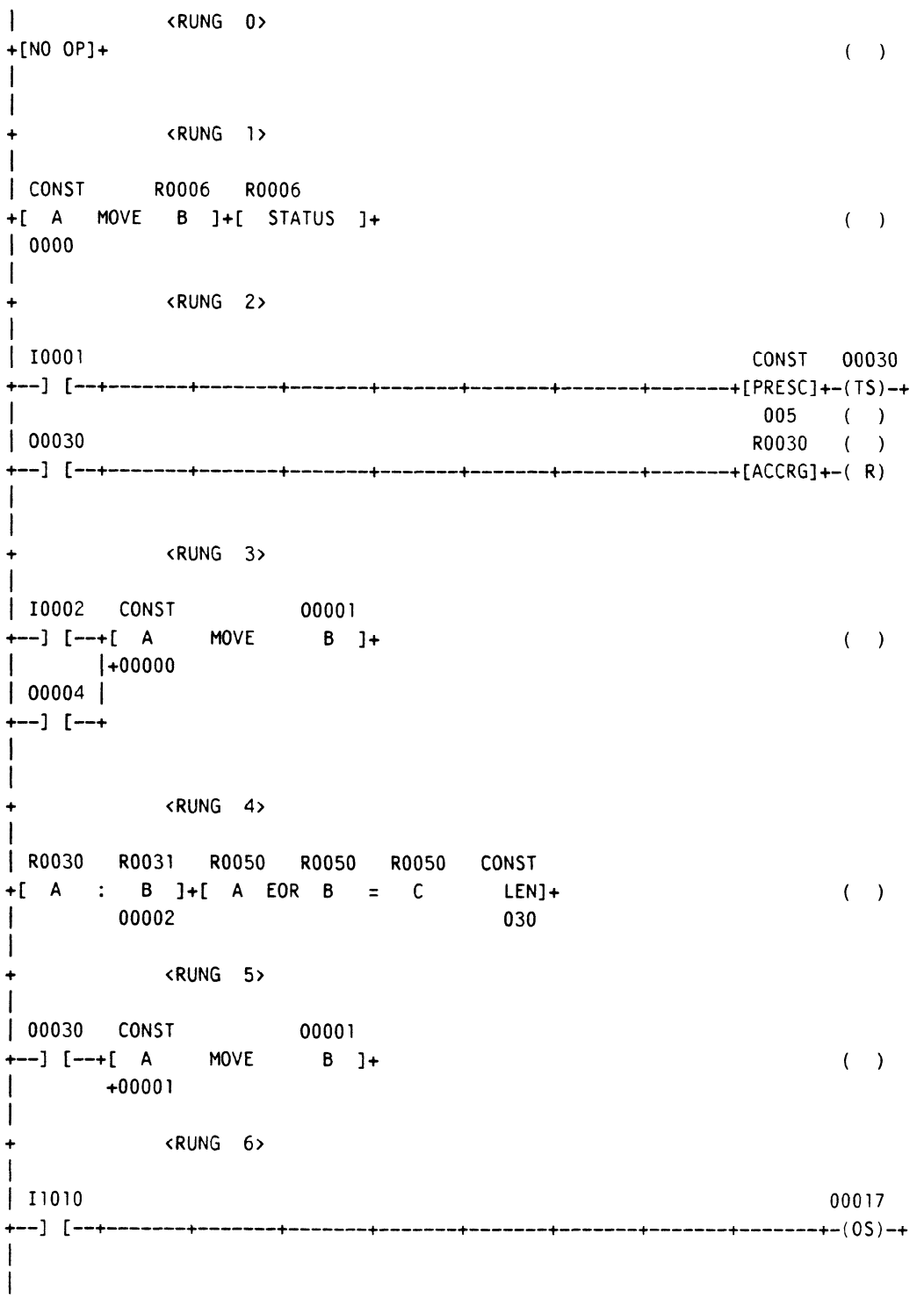
The first SCREQ executed reads registers R0050-R0059 from slave number 1 to registers R0050-R0059 of the master; the second SCREQ reads R0050-R0059 from slave number 2 to R0060-R0069 of the master; and the third SCREQ reads R0050-R0059 from slave number 3 to R0070-R0079. This sequence is repeated every time the timer times out.

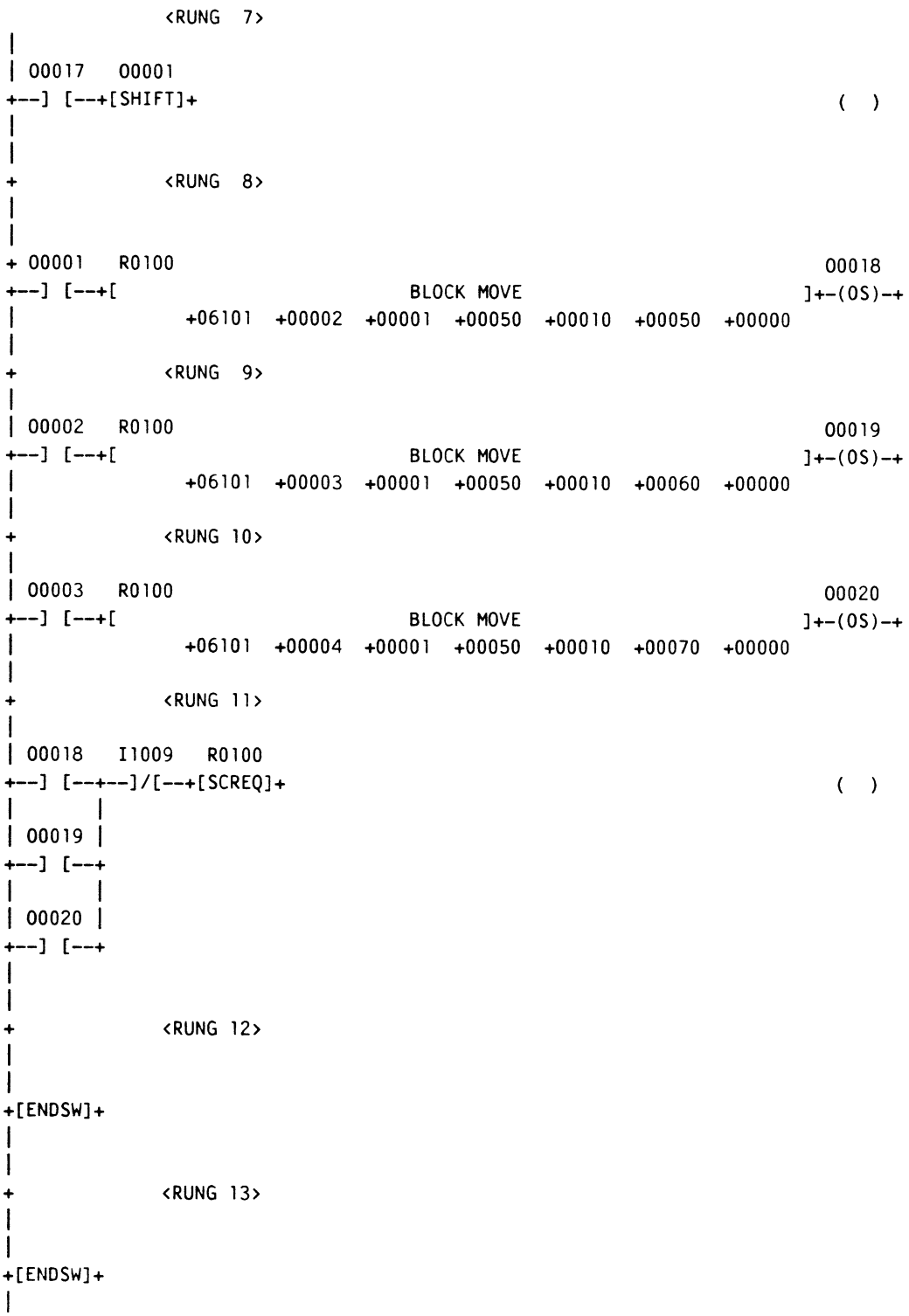
To see the polling routine operate, first place known values in registers R0050-R0059 of each slave. The transfer can then be seen by monitoring the register table of the master Series Six. Before the sequence of SCREQs begins, the matrix function [A AND B = C LEN] zeroes registers R0050-R0079 in the master Series Six allowing repetitive polling sequences to be easily monitored.

The program which follows was written for 1 master and 3 slaves. It can, however, be easily modified to work for 2 slaves or more than 3 slaves by changing the length of the shift register and adding or deleting SCREQs to slave Series Sixes.

PROGRAM 3

(For a rung by rung explanation see the annotation following the program).




<RUNG 7>
00017 00001
[SHIFT]
( )
<RUNG 8>
00001 R0100
BLOCK MOVE
00018
(OS)
+06101 +00002 +00001 +00050 +00010 +00050 +00000
<RUNG 9>
00002 R0100
BLOCK MOVE
00019
(OS)
+06101 +00003 +00001 +00050 +00010 +00060 +00000
<RUNG 10>
00003 R0100
BLOCK MOVE
00020
(OS)
+06101 +00004 +00001 +00050 +00010 +00070 +00000
<RUNG 11>
00018 I1009 R0100
[SCREQ]
( )
00019
00020
<RUNG 12>
[ENDSW]
<RUNG 13>
[ENDSW]

ANNOTATION OF PROGRAM:

Rung number 1 ensures the CCM windows are enabled by zeroing the [STATUS] function register. (Note: A value of zero in the [STATUS] function enables DPU as well as CCM windows).

Rung number 2 is a 5-second timer that runs continuously as long as input I0001 is closed. When the timer times out, it initiates the polling sequence.

Rung number 3 resets the shift register specified in rung No. 7 manually when input I0002 is closed or automatically when the shift register contains:

```
00016                00001
| 0000 0000 0000 1000 |
```

Rung number 4 zeroes registers R0050-R0079 in the master Series Six before each polling sequence (when the timer accumulator equals 2 seconds).

Rung number 5 moves a 1 to the first bit of the shift register specified in rung number 7. This turns output O0001 ON causing the SCREQ in rung number 8 to execute.

Rung number 6 triggers the shift register to shift one bit to the left when I1010 (SCREQ complete without error) transitions from OFF to ON.

Rung number 7 contains the shift register

Rung number 8 loads the SCREQ registers to execute a read command from registers in slave number 1. (CPU ID = 2).

Rung number 9 loads the SCREQ registers to execute a read command from registers in slave number 2. (CPU ID = 3).

Rung number 10 loads the SCREQ registers to execute a read command from registers in slave number 3. (CPU ID = 4).

Rung number 11 is the SCREQ rung with permissive contacts for activation and with the interlock I1009 to prevent execution of the SCREQ when a CCM port is busy.

APPENDIX A
HOST COMPUTER COMMUNICATION INTERFACE SOFTWARE

INTRODUCTION

Host computer communication interface software allows a host computer to communicate with one or more Series Six™ Programmable Logic Controllers (PLCs) equipped with the Communications Control Module (CCM). This interface software generally provides network and communications control, debugging and network event messages, and interface routines for user application programs to transfer data to and from the PLCs. With the communications interface software handling these requirements, the user can concentrate on application programming specific to his needs instead of communications programming.

DEC COMMUNICATION INTERFACE SOFTWARE PACKAGES

GE Fanuc Automation - NA has developed communication interface software for use on Digital Equipment Corporation (DEC) VAX* computers. The main features of this package is summarized below.

FEATURES OF DEC SOFTWARE PACKAGES

- Comprehensive communication package allowing the user to work on the application task, not communications.
- Includes software drivers for CCM protocol.
- Supports all CCM2, CCM3, and I/O CCM system configurations:

  Point-to-point
  Point-to-multipoint (GEnet™)
  Multidrop (includes polling routine).

- Data transfers initiated from the host computer are made by FORTRAN application programs using subroutine calls supplied as a part of this software.
- Accepts normal or interrupt driven data transfers initiated by Series Six PLCs.
- Includes a terminal interface for configuring the network and for accessing system performance data.
- Includes diagnostics for troubleshooting and maintenance.
- Includes a simulator to verify application programs.
- Can handle up to 16 channels.

* Trademarks of Digital Equipment Corporation.

- Can accommodate a total of 254 Series Six PLCs.
- Can accommodate 60 application tasks.

ORDERING SOFTWARE

Types of Licenses

Three types of licenses are offered.

1. SINGLE COMPUTER LICENSE: for use on one computer (registered by DEC serial number on licensing agreement). This license provides the customer with the software on the specified media, the user's manual, and technical support.

2. COPY LICENSE: allows the customer to copy the software for use on an additional computer. Only a copy of the user's manual is supplied. The customer is responsible for copying the software; no technical support is provided. If required, technical support can be ordered separately. For a customer to have obtained this license, he must have previously ordered a Single Computer License.

   This type of license is intended for use by customers having multiple computer installations of which only one site is supported or for OEMs that do not pass support to their customers.

3. CORPORATE LICENSE: Unrestricted use within a company division.

Forms of Software

There are three forms in which software is supplied.

1. SOURCE CODE: This is the form of the software that a human can read and is the form used when writing the software. Source software can easily be modified by a user if he is skilled in programming.

2. OBJECT CODE (Binary): This is the form of the software, generated from source code, that a computer can read. Object software cannot be modified and is the form usually supplied.

3. EXECUTABLE CODE: This is the form of software that the computer uses to perform the job. Executable software is created from object software on the particular computer on which it will be used.

The communication interface software package is offered as a combined source and object code distribution. The package includes a command file which will build the executable code from the source or object code.

Hardware and Software Requirements for VAX Computers

- Any valid VMS system configuration.
- For version 1.3 of the communication interface software, the following software is also required.

  VMS, Version 4.*
  FORTRAN-77, Version 5.0

- Full duplex terminal drivers for connecting to Series Six CCMs supporting 8-bit data with 9th bit parity: e.g. DL-11, DZ-11, and DH-11.

Memory Requirements for DEC Communications Interface Software

The DEC software package consists of 8 system components as listed below.
The approximate task sizes (in 16 bit words) for the system components are as follows:

```
System Control Program (SCP)     :  28K words
Communication Manager  (COMMAN)  :  36K words
Network Event Logger   (NETLOG)  :  16K words
Configurator Program   (CFG)     :  24K words
Configuration Database           :  Application dependent
Simulator              (SIMLTR)  :  24K words
FORTRAN Interface Routines       :  Application dependent
```

The components--COMMAN and the database region--must be in memory to use the software. Therefore, the memory size required for the software is 36K + the data base region. The other components--SCP, CFG, NETLOG, and SIMLTR--require memory only when called.

Catalog Numbers for Ordering Software Packages

Table A.1 CATALOG NUMBERS FOR VAX SOFTWARE

SOFTWARE AND LICENSE TYPE	CATALOG NUMBER	MEDIA
Single Computer License	IC601V001B1B	Magtape 1600BPI 9-Track
Copy License	IC601V001B3B	None Supplied

GEK-25377 User's Manual, Object Code

DESCRIPTION OF DEC SOFTWARE OPERATION

The DEC software package consists of several major system components tied together to perform as a comprehensive communications controller. The primary components are:

System Control Program
Communication Manager
Network Event Logger
Event Processor
Database Configurator Program
System Database
Simulator
FORTRAN Interface Routines

All of these components serve particular roles and will be described on the following pages. Figure A.1 below illustrates the system components and their interaction.

84pc0110

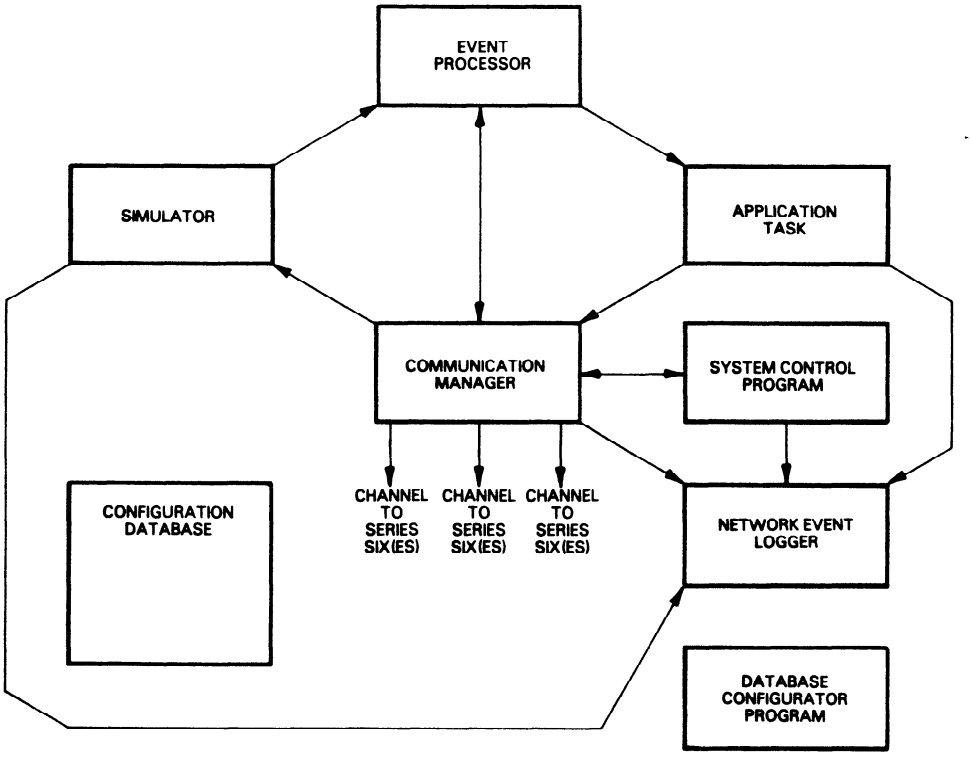


Figure A.1 SYSTEM COMPONENT INTERACTION

Description of Components

System Control Program

The System Control Program (SCP) is an interactive utility program that accepts terminal commands to monitor, test, and control the network. Most SCP commands consist of a command name, a component upon which the command acts, and selected parameters for that component. SCP commands perform the following functions.

- Upline copy programs
- Downline load programs
- Setting channel parameters
- Control of the data logger
- Displaying status data
- Setting remote parameters

The channel and remote parameters are used to configure the network and specify timing parameters.

Communication Manager

The Communication Manager (COMMAN) is a stand alone task that provides the communication network control and protocol functions. COMMAN performs all the communication to and from the Series Six PLCs. COMMAN services the requests from the application tasks, the System Control Program and the Event Processor. In addition, COMMAN maintains status information and requests the logging of network events.

Network Event Logger

The network event logger allows a user to selectively record the activities of the network. It records two types of information: debugging messages and network event messages.

The debugging messages are generated indirectly by application programs. These messages enable the user to monitor the activities of an application program. The messages trace application task subroutine calls, report a routine's completion status, and log the type, size, and direction of data transmission.

The network event messages record changes and problems in the network as they occur. The information logged by these messages includes: any change of Series Six status, bad data transmission, illegal data requests, read and write request failures, Series Six allocation status, and network logger status.

The network loggers are controlled from the System Control Program. These types of messages have been placed in categories which can be selectively enabled. This allows the event logger to be tailored to obtain specific types of information.

Event Processor

The Event Processor informs application tasks or the System Control Program the results of their service requests to the Communication Manager. The Event Processor is included within the Communication Manager.

Database Configurator Program

The Database Configurator Program configures and tailors the configuration database to the user's specific requirements.

The size of the configuration database is determined by the number of channels and remotes the user desires to be serviced by the Communication Manager.

System Database

The system database consists of a group of parameters and counters. The parameters define how the network is configured and will perform. The counters store information describing actual system configuration and performance. Information is available on a system, channel, or remote basis. Parameters and counters may be accessed from the System Control Program.

Simulator

The simulator allows computer application tasks to be tested and debugged without connecting to a Series Six CPU. The programmer develops a script of responses to communications. An application task that is in the simulator mode will then access the script for data or the location to send data. The results then may be easily analyzed.

FORTRAN Interface Routines

The FORTRAN Interface Routines are a series of subroutine calls available to the computer application task. These include:

- Allocating a remote Programmable Logic Controller (PLC)
- Copying a remote PLC program
- De-allocating a remote PLC
- Getting a channel's parameters
- Getting a remote PLC's parameters
- Getting the system's parameters
- Retrieving the computer's memory tables
- Loading a program to a remote PLC
- Initiating a computer data request from a remote PLC
- Receiving an externally initiated exception message
- Receiving an interrupt message
- Sending data to a remote

After execution, each subroutine will respond with a completion code.

Privileges

The VAX software uses a privilege account system. A privileged account is required to be able to change communication parameters, load Series Six programs, or modify contents of Series Six memory locations.

A non-privileged account can examine the status and configuration of the communication parameters and can examine Series Six memory locations, but cannot modify them.

Allowable Hardware System Configurations

The DEC interface software will support 3 configurations of the Series Six and a DEC VAX computer: point-to-point, point-to-multipoint (GEnet), and multidrop. All connections are made to the Series Six CCM1 or CCM2/3 modules.

Any combination of configurations may coexist on the same computer, but only one configuration is allowed per channel.

84pc0063

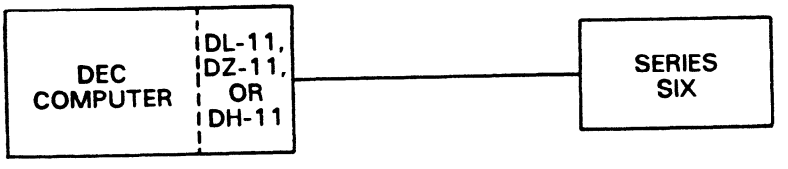


Figure A.2 POINT-TO-POINT CONNECTION

The computer can initiate a message; the Series Six PLC can also initiate a message if a CCM2/3 is used. The maximum number of devices which can be connected to the computer is determined by the number of channels the computer hardware and software can support. The DEC software package can support a maximum of 16 channels.

84pc0064

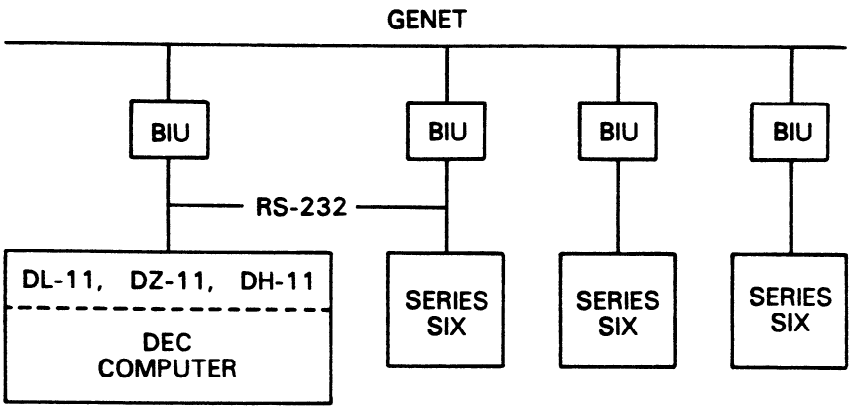


Figure A.3 POINT-TO-MULTIPOINT (GEnet) NETWORK

The DEC Communication Interface Software will support communications to Series Six PLCs accross GEnet. Any device may initiate a message to any other, except Series Six PLCs with a CCM1 interface which respond only to another device.

84pc0065

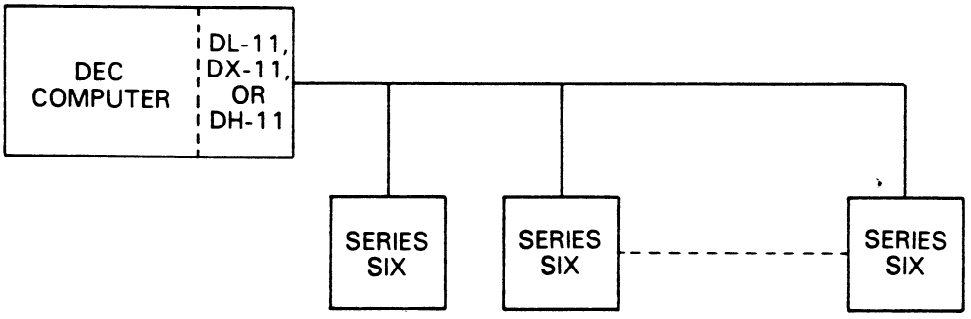


Figure A.4 MULTIDROP NETWORK CONNECTION

This configuration is supported only by the CCM option. The computer serves as a master in a multidrop network. The computer is the only device which can initiate a message in this configuration. A polling routine is provided to perform polling of the remotes. Consult CCM literature for maximum number of devices on the remote link (typically 8 without modems using the RS422 electrical interface; 90 with modems).

APPENDIX B
EXPANDED FUNCTIONS

INTRODUCTION

The following pages explain the the Series Six™ Communications Control Module (CCM2, CCM3, and I/O CCM) expanded functions and the CCM module hardware and software identification.

CCM modules that perform the extended functions are listed below. Those versions listed or later versions may be used.

CCM Module	Hardware Id.	Software ID.
I/O CCM	IC600BF948	203 (hex), 515 (decimal)
CCM2	IC600CB536K	006
CCM3	IC600CB537K	104 (hex), 260 (decimal)

HARDWARE IDENTIFICATION

CCM2, CCM3 hardware versions IC600CB536, IC600CB537 are a single-PROM module. This new single-PROM module replaces either a 6-PROM or single-PROM module for CCM2, and a 7-PROM or single-PROM module for CCM3.

The CCMA3 module (for both CCM2 and CCM3) is identified as follows:

Hardware Id. CCMA3, 44A717545-G02 R02 or later.

The I/O CCM module is identified as follows:

Hardware Id. BAMA, 44A717588-G01 R02 or later.

NOTE

Refer to the Module Compatability information located in the Preface of this manual for more information concerning hardware/software features and module compatability.

EXPANDED FUNCTIONS OVERVIEW

Several additional features and enhancements are available to the user with the appropriate hardware/software release as listed above. A brief description of the Series Six Communication Control Module (CCM) features and enhancements are as follows:

EXPANDED I/O REFERENCE

A new method of addressing the I/O points within the Expanded Instruction Set has been devised to allow access of additional I/O points. This feature allows addressing of channelized I/O points available with the Series Six expanded instruction set. The I/O points can be accessed by both the CCM protocol and Remote Terminal Unit (RTU) protocol for CCM3 and I/O CCM, and the CCM protocol only for CCM2. CCM protocol also supports addressing of the Auxiliary I/O Override table. Refer to the attached documentation, Table B.1, which shows the I/O addressing for CCM and RTU protocols.

EXPANDED USER MEMORY REFERENCE

The expanded II instruction set allows memory addressing up to 64K of the user logic memory. The expanded user logic memory is supported by the CCM protocol.

SINGLE BIT WRITE

The CCM offers a single bit write feature that may be used on the input, output, auxiliary input, auxiliary output and auxiliary override tables in the Series Six PLC. This feature has been added to the CCM protocol, and will permit the user to set, clear, or toggle a bit. Refer to Table B.2, which lists the new memory types allocated for the single bit write feature.

PROGRAMMABLE TIMEOUTS AND RETRYS

This feature allows timeout and retry value programming for the CCM protocol. Four SCREQs have been defined to allow timeouts and retrys to be programmed for both ports. Refer to Table B.5 which shows the format of the new SCREQs allocated for this feature.

EXPANDED I/O TRANSLATION

This section provides an easy algorithm for translating Expanded I/O references to and from absolute bit offsets within the I/O table. Examples are also provided to clearly show how the algorithm works. Refer to Table B.1 to find the CCM and RTU point mapping for the first point within each I/O channel.

Table B.1 SERIES SIX PLUS I/O CHANNEL AND POINT MAPPING

SERIES SIX PLUS I/O CHANNEL AND POINT - RTU/ CCM POINT MAPPING							
<---------- CCM3 and I/O CCM ---------->							
<---------- CCM2 ---------->							
SERIES SIX PLUS - CCM MAPPING				SERIES SIX PLUS - RTU MAPPING			
I/O START		I/O END		I/O START		I/O END	
S6 PLUS CHAN & PT	CCM POINT	S6 PLUS CHAN & PT	CCM POINT	S6 PLUS CHAN & PT	RTU POINT	S6 PLUS CHAN & PT	RTU POINT
REAL I/O							
O(I) 1	1	O(I) 1024	1024	O(I) 1	0	O(I) 1024	1023
O(I) 1+1	1025	O(I) 1+1024	2048	O(I) 1+1	1024	O(I)1+1024	2047
O(I) 2+1	2049	O(I) 2+1024	3072	O(I) 2+1	2048	O(I)2+1024	3071
O(I) 3+1	3073	O(I) 3+1024	4096	O(I) 3+1	3072	O(I)3+1024	4095
O(I) 4+1	4097	O(I) 4+1024	5120	O(I) 4+1	4096	O(I)4+1024	5119
O(I) 5+1	5121	O(I) 5+1024	6144	O(I) 5+1	5120	O(I)5+1024	6143
O(I) 6+1	6145	O(I) 6+1024	7168	O(I) 6+1	6144	O(I)6+1024	7167
O(I) 7+1	7169	O(I) 7+1024	8192	O(I) 7+1	7168	O(I)7+1024	8191
AO(I) 1 *1	8193	AO(I) 1024 *1	9216	AO(I) +1	*****	AO(I) +1024	*****
O(I) 9+1	9217	O(I) 9+1024	10240	O(I) 9+1	9216	O(I)9+1024	10239
O(I) A+1	10241	O(I) A+1024	11264	O(I) A+1	10240	O(I)A+1024	11263
O(I) B+1	11265	O(I) B+1024	12288	O(I) B+1	11264	O(I)B+1024	12287
O(I) C+1	12289	O(I) C+1024	13312	O(I) C+1	12288	O(I)C+1024	13311
O(I) D+1	13313	O(I) D+1024	14336	O(I) D+1	13312	O(I)D+1024	14335
O(I) E+1	14337	O(I) E+1024	15360	O(I) E+1	14336	O(I)E+1024	15359
O(I) F+1	15361	O(I) F+1024	16384	O(I) F+1	15360	O(I)F+1024	16383
INTERNAL I/O(I)							
O(I) 0-1	16385	O(I) 0-1024	17408	O(I) 0-1	16384	O(I)0-1024	17407
O(I) 1-1	17409	O(I) 1-1024	18432	O(I) 1-1	17408	O(I)1-1024	18431
O(I) 2-1	18433	O(I) 2-1024	19456	O(I) 2-1	18432	O(I)2-1024	19455
O(I) 3-1	19457	O(I) 3-1024	20480	O(I) 3-1	19456	O(I)3-1024	20479
O(I) 4-1	20481	O(I) 4-1024	21504	O(I) 4-1	20480	O(I)4-1024	21503
O(I) 5-1	21505	O(I) 5-1024	22528	O(I) 5-1	21504	O(I)5-1024	22527
O(I) 6-1	22529	O(I) 6-1024	23552	O(I) 6-1	22528	O(I)6-1024	23551
O(I) 7-1	23553	O(I) 7-1024	24576	O(I) 7-1	23552	O(I)7-1024	24575
O(I) 8-1	*****	O(I) 8-1024	*****	O(I) 8-1	*****	O(I)8-1024	*****
O(I) 9-1	25601	O(I) 9-1024	26624	O(I) 9-1	25600	O(I)9-1024	26623
O(I) A-1	26625	O(I) A-1024	27648	O(I) A-1	26624	O(I)A-1024	27647
O(I) B-1	27649	O(I) B-1024	28672	O(I) B-1	27648	O(I)B-1024	28671
O(I) C-1	28673	O(I) C-1024	29696	O(I) C-1	28672	O(I)C-1024	29695
O(I) D-1	29697	O(I) D-1024	30720	O(I) D-1	29696	O(I)D-1024	30719
O(I) E-1	30721	O(I) E-1024	31744	O(I) E-1	30720	O(I)E-1024	31743
O(I) F-1	31745	O(I) F-1024	32768	O(I) F-1	31744	O(I)F-1024	32767

*1 CCM point value for Aux Input and Output Overrides.

***** Indicates no CCM or RTU points assigned for channel

SERIES SIX PLUS I/O POINT - CCM/RTU POINT MAPPING

The CCM or RTU point corresponding to any Series Six Plus input or output point may be found by following the steps listed below:

1. Select desired channel and point.
2. Find CCM or RTU point for first point within desired channel.
3. Add the desired point to value from step 2.
4. Deduct 1 from total in step 3.

The value in step (4) is the CCM or RTU point corresponding to the desired channel and point.

EXAMPLE 1:

Find CCM point for "O7 + 578".

-> CCM point for O7 + 1 = 7169.
-> 7169 + 578 = 7747.
-> 7747 - 1 = 7746.
-> CCM point for O7 + 578 = 7746.

EXAMPLE 2:

Find RTU point for "IA - 213".

-> RTU point for IA - 1 = 26624.
-> 26624 + 213 = 26837.
-> 26837 - 1 = 26836.
-> RTU point for IA - 213 = 26836.

CCM SINGLE BIT WRITE

The CCM protocol includes a single bit write feature that may be used on the input, output, aux. input, aux. output, and override tables in the Series Six. This feature will support bit set, bit clear, and bit toggle functions.

The bit set operation allows a single point to be turned on in normal or expanded I/O, Aux I/O, or override tables. The bit clear operation allows a single bit to be cleared in normal or expanded I/O, Aux I/O, or override tables. The bit toggle function allows change of the current state of a single bit in normal or expanded I/O or Aux I/O tables. The bit toggle function will not be supported for the override tables.

Any of the bit write functions may be invoked by issuing a CCM write to one of the following memory types defined for CCM. The memory types, which define the target table and bit write operation, are listed in Table B.2.

Table B.2 MEMORY TYPES FOR CCM BIT WRITE FUNCTION

CCM Memory type	CCM Target Table	Bit Operation
13	Input table	Bit Set
14	Output table	Bit Set
15	Input Override table	Bit Set
16	Output Override table	Bit Set
17	Input table	Bit Clear
18	Output table	Bit Clear
19	Input Override table	Bit Clear
20	Output Override table	Bit Clear
21	Input table	Bit Toggle
22	Output table	Bit Toggle

Two SCREQ command numbers have been reserved for the bit write function, one for each port. The ladder logic program may invoke the desired bit write function by issuing the new SCREQ supplying the information defined in Table B.3

Table B.3 NEW SCREQs FOR SINGLE BIT WRITE FUNCTION

Single Bit Write Function SCREQ						
Port	Command Number	Rn+1	Rn+2	Rn+3	Rn+4	Rn+5
Port 1 Port 2	6110 6210	(a) (a)	(b) (b)	(c) (c)	x x	x x
SCREQ entry	Description					
(a) (b) (c) x	Target I/D Target Memory Type Target Memory Address Field not required					

SINGLE BIT WRITE DATA FLOW

The following example shows the flow of the CCM protocol processing a bit write function.

The CCM protocol processing the bit write function with memory type 17 (11H) clear input.

This example shows a write request to CPU ID 11(0BH) to clear bit 41 (29H) of the Input table.

The high bit of header byte 4 is set for the write function and leaves 7 bits free for the memory type.

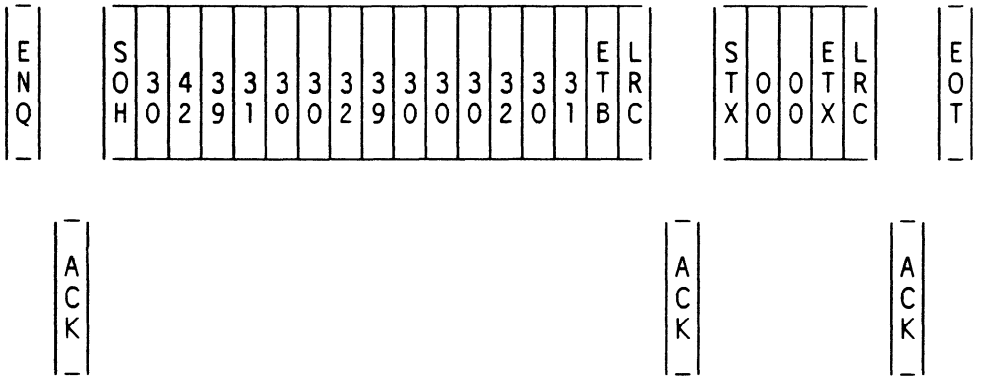


Figure B.1 SINGLE BIT WRITE DATA FLOW

In order to maintain consistency within the CCM protocol, each bit write request will be required to supply a 2 byte dummy data field. The data field that must be supplied is shown in Table B.4

Table B.4 REQUIRED DATA FIELD FOR CCM BIT WRITE FUNCTION

STX	0	0	ETX	LRC

PROGRAMMABLE TIMEOUT AND RETRY

Timeout values and retry values for Series Six CCM protocol may be programmed. RTU does not have a programmable timeout or retry feature.

Four separate SCREQs have been defined to allow the user to set retry or timeout values for each port. Table B.5 shows these SCREQs. It also lists the default values for each timeout, and the valid range for each value that is programmable.

Table B.5 NEW SCREQs AND DEFAULT VALUES

Programmable Retries and Timeouts SCREQs							
Command	Port	SCREQ #	Rn+1	Rn+2	Rn+3	Rn+4	Rn+5
Set Retries	1 2	6130 6230	x x	(a) (a)	(b) (b)	(c) (c)	(d) (d)
Set Timeouts	1 2	6131 6231	(e) (e)	(f) (f)	(g) (g)	(h) (h)	(i) (i)
Entry	Range	Default	Description				
(a)	0 to 32 times	32 times	Peer-Peer or Master/Slave ENQ retry count				
(b)	0 to 5 times	3 times	Q Sequence retry				
(c)	0 to 5 times	3 times	Header retry count				
(d)	0 to 5 times	3 times	Data Block retry				
(e)	50 to 2000 ms	800 ms	ACK/NAK for ENQ, Start of Header following ACK, and EOT to close link				
(f)	50 to 3000 ms	*	Time for Header to finish				
(g)	50 to 10000 ms	2000 ms	ACK/NAK for Header				
(h)	50 to 65000 ms	20000 ms	Start of Data after Header Ack ACK/NAK following Data Block				
(i)	50 to 65000 ms	*	Data to finish				
x			Field not required				

* Default value depends on selected data rate.

NOTE

It is possible to set timeouts so that the communications will not execute properly.

APPENDIX C
GLOSSARY OF TERMS

Address - A series of decimal numbers assigned to specific program memory locations and used to access those locations.

Analog - A numerical expression of physical variables such as rotation and distance to represent a quantity.

Application program - The ladder logic program executing in a PLC or user program in computer.

ASCII - An 8-level code (7 bits plus 1 parity bit) commonly used for exchange of data which is the American Standard Code for Information Interchange.

Asynchronous - Transmission of data in which time intervals between transmitted characters may be of unequal length. Asynchronous transmission is controlled by start and stop bits at the beginning and end of each character.

Backplane - A group of connectors physically mounted at the back of a rack so that printed circuit boards can be mated to them.

Baud - A unit of data transmission speed equal to the number of code elements per second.

Binary - A numbering system that uses only the digits 0 and 1. This system is also called base 2.

Bit - The smallest unit of memory. Can be used to store only one piece of information that has two states (for example, a One/Zero, On/Off, Good/Bad, Yes/No, etc.). Data that requires more than two states (for example, numerical values 000-999) will require multiple bits.

Broadband Network - A network which can handle medium-to-large size applications with up to several hundred stations as a typical number which might be attached. Broadband technology is used in larger networking systems and requires a headend remodulator.

Bus - An electrical path for transmitting and receiving data.

Bus Interface Unit (BIU) - A functional unit that interconnects a local area network (LAN) with another device or network that uses different protocols.

Byte - A group of binary digits operated on as a single unit. In the Series Six PLC, a byte is made up of 8 bits.

Carrierband Network - A network designed to handle small-to medium-size applications with 6-20 stations as a typical number of stations which might be attached.

Communication Control Module (CCM2, CCM3) - The Communications Control Module provides a serial interface between the Series Six PLC and other devices on the network which can initiate communications based on the CCM protocol.

Communication Windows - Communication between the ladder logic program and the local interface module which takes place during the PLC scan.

CPU (Central Processing Unit) - The central device or controller that interprets user instructions, makes decisions and executes the functions based on a stored program. This program specifies actions to be taken to all possible inputs.

Current Loop - There is no true standard for the current loop interface. The current loop interface is normally used in environments where excessive electrical noise from machinery is a problem.

Data Link - The equipment including interface modules and cables that allow transmission of information.

Diagnostic Status Words - A group of 20 words which provide detailed information about the operation and configuration of the CCM module, and used for monitoring and diagnosing transmission errors. The status words are maintained and updated in the CCM module.

DIP Switch - An acronym for Dual-In-Line Package, which is a group of miniature toggle or slide switches arranged side-by-side in a single package. Commonly used as the physical device for setting the configuration of various parameters necessary to the operation of electronic equipment.

Data Processing Request (DPREQ) - The Data Processing REQuest is an instruction in the ladder logic program which opens a communications window between the Series Six CPU and the I/O CCM. The DPREQ Allows the I/O CCM to execute the communication function specified in the request.

DPU Executive Window - The Data Processing Unit (DPU) executive window is a part of the PLC scan which provides a window for the I/O CCM. The window is enabled by setting hardware jumpers on the module.

CCM Executive Window - A part of the PLC scan which provides a mechanism for the CCM to read and write PLC memory. The window is executed automatically once per PLC scan as long as the CCM Interface module is installed and the windows have been enabled by the STATUS instruction.

Firmware - A series of instructions contained in ROM (Read Only Memory) which are used for internal processing functions only. These instructions are transparent to the user.

Hardware - All of the mechanical, electrical and electronic devices that comprise the Series Six programmable controller and its application(s).

Hexadecimal - A numbering system, having 16 as a base, represented by the digits 0 through 9, then A through F.

Initiating Station - The station from which communication originates.

Input - A signal, typically ON or OFF, that provides information to the PLC. Inputs are usually generated by devices such as limit switches and pushbuttons.

Input Module - An I/O module that converts signals from user devices to logic levels used by the CPU.

Interface - To connect a Programmable Logic Controller with its application devices, communications channels, and peripherals through various modules and cables.

I/O (Input/Output) - That portion of the PLC to which field devices are connected.

I/O Module - A printed circuit assembly (I/O CCM) that interfaces between user devices and the Series Six programmable logic controller.

I/O Scan - A method by which the CPU monitors all inputs and controls all outputs within a prescribed time.

ISO Standards - The International Standards Organization (ISO) for Open System Interconnection (OSI).

ISO Reference Model for Open System Interconnection - An international standard for network architectures which define a seven layer model. The intent is to provide a network design framework to allow equipment from different vendors to be able to communicate.

Isolation - A method of separating field wiring from logic level circuitry. Typically accomplished through the use of optical isolation devices.

K - An abbreviation for kilo or exactly 1024 in the world of computers. Usually related to 1024 words of memory.

Ladder Diagram - A representation of control logic relay systems. The user programmed logic is expressed in relay equivalent symbology.

LED - An acronym for Light-Emitting-Diode, which is a solid state device commonly used as a visual indicator in electronic equipment.

Local Area Network (LAN) - A communication network covering a limited physical space, and having intermediate data transport capability.

Logic - A fixed set of responses (outputs) to various external conditions (inputs). All possible situations for both synchronous and non-synchronous activity must be specified by the user. Also referred to as the program.

Logic Memory - In the Series Six PLC, dedicated CMOS RAM memory accessible by the user for storage of user ladder diagram programs.

Manufacturing Automation Protocol (MAP) - MAP communication protocol is specified by the Manufacturing Automation Protocol (MAP) specification. MAP is a "Connection-oriented" protocol; that is, stations residing on a network are able to transfer information only after establishing a logical connection much like two people using the telephone system.

Memory - A grouping of physical circuit elements that have data entry, storage and retrieval capability.

Memory Protect - A hardware capability that prevents user memory from being altered by an external device. This capability is controlled by a key switch on the CPU power supply.

Microprocessor - An electronic computer processor consisting of integrated circuit chips that contain arithmetic, logic, register, control and memory functions.

Microsecond (μs) - One millionth of a second. 1×10^{-6} or 0.000001 second.

Millisecond (ms) - One thousandth of a second. 1×10^{-3} or 0.001 second.

Mnemonic - An abbreviation given to an instruction, usually an acronym formed by combining initial letters or parts of words.

Modules - A replaceable electronic subassembly usually plugged in and secured in place but easily removable in case of fault or system redesign. In the Series Six PLC , a combination of a printed circuit board and its associated faceplate which when combined form a complete assembly.

Nanosecond (ns) - One billionth of a second. 1×10^{-9} or 0.000000001 second.

Noise - Undesirable electrical disturbances to normal signals, generally of high frequency content.

Non-Volatile Memory - A memory capable of retaining its stored information under no-power conditions (power removed or turned off).

OFF-Line - Equipment or devices that are not connected to a communications line; for example, the Workmaster computer, when off-line, operates independent of the Series Six CPU.

ON-Line - Descriptive of equipment or devices that are connected to the communications line.

Optical Isolation - Use of a solid state device to isolate the user input and output devices from internal circuitry of an I/O module and the CPU.

Output - Information transferred from the CPU, through a module for level conversion, for controlling an external device or process.

Output Devices - Physical devices such as motor starters, solenoids, etc. that receive data from the Programmable Logic Controller.

Output module - An I/O module that converts logic levels within the CPU to a usable output signal for controlling a machine or process.

Outputs - A signal typically ON or OFF, originating from the PLC with user supplied power, that controls external devices based upon commands from the CPU.

Parity - The anticipated state, either odd or even, of a set of binary digits.

Parity Bit - A bit added to a memory word to make the sum of the bits in a word always even (even parity) or always odd (odd parity).

Parity Check - A check that determines whether the total number of ones in a word is odd or even.

Parity Error - A condition that occurs when a computed parity check does not agree with the parity bit.

Peer-Peer - Communication between stations at the same level or layer in the hierarchy.

Peripheral Equipment - External units that can communicate with a PLC, for example, programmers, printers, etc.

PLC - Commonly used abbreviation for Programmable Logic Controller.

Program - A sequence of functions entered into a Programmable Logic Controller to be executed by the processor for the purpose of controlling a machine or process.

Programmable Logic Controller or Programmable Controller - A solid-state industrial control device which receives inputs from user supplied control devices such as switches and sensors, implements them in a precise pattern determined by ladder diagram based programs stored in the user memory, and provides outputs for control of processes or user supplied devices such as relays and motor starters.

Programmer - A device for entry, examination and alteration of the PLC's memory, including logic and storage areas.

PROM - An acronym for Programmable Read Only Memory. A retentive digital device programmed at the factory and not readily alterable by the user.

Protocol - A set of rules for exchanging messages between two communicating processes.

Q Sequence - The Q sequence protocol format is used to poll and transfer 4 bytes of data from a slave to a master without issuing the 17-byte header.

Quick Access Buffer (QAB) - The QAB is a 1024 byte buffer resident on the CCM module used for faster data transfer than the CPU to CPU transfer.

RAM - An acronym for Random Access Memory. A solid-state memory that allows individual bits to be stored and accessed. This type of memory is volatile; that is, stored data is lost under no power conditions, therefore a battery backup is required. The Series Six PLC uses a Lithium Manganese Dioxide battery or an optional external back-up battery for this purpose.

Read - To have data entered from a storage device.

Reference - A number used in a program that tells the CPU where data is coming from or where to transfer the data.

Register Memory - In the Series Six PLC, dedicated CMOS RAM memory accessible by the user for data storage and manipulation.

Remote Terminal Unit (RTU) - RTU protocol is a query-response mode of operation used for communication between the CCM device and host computer. The host computer transmits the query to the RTU slave which can only respond to the master.

RS-232D - A standard specified by the Electronics Industries Association (EIA) for the mechanical and electrical characteristics of the interface for connecting Data Communications Equipment (DCE) and Data Terminal Equipment (DTE).

RS-422 - A recommended standard defining electrical interface characteristics to connect Data Terminal Equipment (DTE) or Data Circuit-Transmitting Equipment (DCE). The RS-422 standard permits longer range and faster transmission rate than the RS-232D standard.

RUN Light - An LED indicator on the Arithmetic Control module which, when on, indicates that the execution sequence of the PLC is proceeding normally and the I/O scan is completed at least once every 200 milliseconds, ±50 milliseconds.

Rung - A sequence or grouping of PLC functions that control one coil. One or more rungs form a ladder diagram.

Scan - The technique of examining or solving all logic steps specified by the program in a sequential order from the first step to the last.

Serial Communication - A method of data transfer within a PLC, whereby the bits are handled sequentially rather than simultaneously as in parallel transmission.

Serial Communication Request (SCREQ) - Instruction which, when executed by the ladder logic program, opens a window between the Series Six Plus CPU and the CCM module, allowing the CCM to execute the communication function specified in the request.

Significant Bit - A bit that contributes to the precision of a number. The number of significant bits is counted beginning with the bit contributing the most value, referred to as the Most Significant Bit (MSB), and ending with the bit contributing the least value, referred to as the Least Significant Bit (LSB).

Status Byte - Indicates overall status of the CCM module and the communication network.

Status Instruction - A ladder logic program instruction which enables and disables the communication windows between the communications module and the PLC.

Storage - Used synonymous with memory.

Synchronous - Transmission in which data bits are transmitted at a fixed rate, with the transmitter and receiver synchronized by a clock. This eliminates the need for start and stop bits.

Terminator - A device or load connected to the output end of a transmission line to terminate or end the signals on that line. In the Series Six PLC, DIP shunts and jumper packs connect on-board resistors which terminate the I/O chain signals on an I/O Receiver or Advanced I/O Receiver if it is the last Receiver in any I/O chain.

Unit of Load - An expression used to describe the load placed on a power supply by an I/O module or a CPU module. Also the amount of current or load capacity available from a power supply.

User Memory - Term commonly used when referring to the memory circuits within the PLC used for storage of user ladder diagram programs.

Volatile Memory - A memory that will lose the information stored in it if power is removed from the memory circuit devices.

Word - A measurement of memory length, usually 4, 8, or 16 bits long (16 bits for the Series Six PLC).

Write - To transfer, record, or copy data from one storage device to another.

INDEX

INDEX

C

D

E

F, G

H

INDEX

I

J, K, L

M

INDEX

N

O

P

Q

INDEX

R

S

INDEX

T

U V

W

GE Fanuc Automation North America, Inc., Charlottesville, Virginia