User Manual for the
HE693RTM705 and
HE697RTM700

RTU Master Communication Module

Seventh Edition
14 November 2003

MAN0076-07

PREFACE

This manual explains how to use the RTU Master Communication Module.

Information in this document is subject to change without notice and does not represent a commitment on the part of Horner APG, LLC.

Series 90-30, Series 90-70, and Logicmaster are trademarks of GE Fanuc Automation North America, Inc.

Alspa 8000 and P8 are trademarks of CEGELEC.

Modbus is a trademark of Modicon.

For user manual updates, contact Horner APG,

North America:
(317) 916-4274
www.heapg.com

Europe:
(+) 353-21-4321-266
www.horner-apg.com

LIMITED WARRANTY AND LIMITATION OF LIABILITY

Horner APG, LLC. ("HE") warrants to the original purchaser that RTU Master Communication Module manufactured by HE is free from defects in material and workmanship under normal use and service. The obligation of HE under this warranty shall be limited to the repair or exchange of any part or parts which may prove defective under normal use and service within two (2) years from the date of manufacture or eighteen (18) months from the date of installation by the original purchaser whichever occurs first, such defect to be disclosed to the satisfaction of HE after examination by HE of the allegedly defective part or parts. THIS WARRANTY IS EXPRESSLY IN LIEU OF ALL OTHER WARRANTIES EXPRESSED OR IMPLIED INCLUDING THE WARRANTIES OF MERCHANTABILITY AND FITNESS FOR USE AND OF ALL OTHER OBLIGATIONS OR LIABILITIES AND HE NEITHER ASSUMES, NOR AUTHORIZES ANY OTHER PERSON TO ASSUME FOR HE, ANY OTHER LIABILITY IN CONNECTION WITH THE SALE OF THIS RTU Master Communication Module. THIS WARRANTY SHALL NOT APPLY TO THIS RTU Master Communication Module OR ANY PART THEREOF WHICH HAS BEEN SUBJECT TO ACCIDENT, NEGLIGENCE, ALTERATION, ABUSE, OR MISUSE. HE MAKES NO WARRANTY WHATSOEVER IN RESPECT TO ACCESSORIES OR PARTS NOT SUPPLIED BY HE. THE TERM "ORIGINAL PURCHASER", AS USED IN THIS WARRANTY, SHALL BE DEEMED TO MEAN THAT PERSON FOR WHOM THE RTU Master Communication IS ORIGINALLY INSTALLED. THIS WARRANTY SHALL APPLY ONLY WITHIN THE BOUNDARIES OF THE CONTINENTAL UNITED STATES.

In no event, whether as a result of breach of contract, warranty, tort (including negligence) or otherwise, shall HE or its suppliers be liable of any special, consequential, incidental or penal damages including, but not limited to, loss of profit or revenues, loss of use of the products or any associated equipment, damage to associated equipment, cost of capital, cost of substitute products, facilities, services or replacement power, down time costs, or claims of original purchaser's customers for such damages.

To obtain warranty service, return the product to your distributor with a description of the problem, proof of purchase, post paid, insured and in a suitable package.

ABOUT PROGRAMMING EXAMPLES

Any example programs and program segments in this manual or provided on accompanying diskettes are included solely for illustrative purposes. Due to the many variables and requirements associated with any particular installation, Horner APG cannot assume responsibility or liability for actual use based on the examples and diagrams. It is the sole responsibility of the system designer utilizing RTU Master Communication Module to appropriately design the end system, to appropriately integrate the RTU Master Communication Module and to make safety provisions for the end equipment as is usual and customary in industrial applications as defined in any codes or standards which apply.

Note: The programming examples shown in this manual are for illustrative purposes only. Proper machine operation is the sole responsibility of the system integrator.

Revisions to This Manual

This version (MAN0076-07) of the **RTU Master Communication Module User Manual** contains the following revisions and additions:

1. Added note in opening paragraph in Chapter 1: Introduction – RTM705 and RTM700 do not support Modbus PLUS.

2. Revised Section 1.2. Neither RTM705 nor RTM700 operate in an expansion rack. Added power consumption for RTM700.

3. Revised LED operation in Section 1.4 (RTM705) and Section 1.6 (RTM700).

4. Revised Figure 2.1 – uses VersaPro screens.

5. Added new Figures 2.7 – 2.10 and Figure 2.14. Re-numbered previous Figures 2.7, 2.8, and 2.9 as Figures 2.11, 2.12 and 2.13.

6. Revised Section 3.2.1.

7. Revised Section 3.2.2.

8. Revised Table 3.3.

9. Revised Section 3.2.3, SCB Number, Pointer type, and Offset.

10. Revised Section 3.3.1.

11. Added note to Table 3.9.

12. Added new Section 3.8: Advanced Functions.

13. Added Index.

TABLE OF CONTENTS

NOTES

CHAPTER 1: INTRODUCTION

The HE**693**RTM**705** / HE**697**RTM**700** provides a PLC with a flexible communications interface to **RTU/Modbus** networks. The RTM700/705 functions as a master that allows data to be read/written to one or more slave devices. The RTM705 resides in a single Series 90-30 local slot and is compatible with the CPU331 or larger. The RTM700 resides in a single Series 90-70 local slot and is compatible with Series 90-70 CPUs.

Note: The HE**693**RTM**705** and HE**697**RTM**700 do not support Modbus PLUS**.

1.1 Product Description

The RTM700/705 provides two Modbus/RTU master channels which may be controlled by the 90-30/90-70 to access data from remote slave RTU's. Each channel can be individually programmed for dial in, dial out, radio modem, multi-drop (RS422/RS485) and direct connection (RS232) operation. Each channel can send or receive Analog or Discrete data using standard RTU commands (i.e. Read Coil). Multiple commands can be sent to each slave with each command requesting either a single point or multiple contiguous points up to 250 bytes. Data to be sent or stored is placed in the 90-30/90-70 reference data tables.

Also stored in the 90-30/90-70 reference data tables are command lists, which control what RTU commands are sent to slave RTUs. Once started, the RTM700/705 uses these command lists to run in a semi-unsupervised state to transfer data between the 90-30/90-70 data tables and the slave RTUs. Additionally, it may run in a single command mode where new commands can be issued during operation. The length of time between updates to each slave RTU can be individually adjusted.

Each slave RTU has an individual status field, which reflects the condition of communications with that RTU. By continuously monitoring each status field, the 90-30/90-70 can detect configuration or communication problems on a slave by slave basis.

The RTM700/705 also supports report-by-exception when used with Horner APG Modbus slaves. A slave may originate a call to the RTM700/705 when an exception occurs. The RTM700/705 then automatically determines the slave ID of the caller and executes the command list associated with that caller to retrieve and/or send the appropriate data.

Configuration of the RTM700/705 port and initialization parameters is easily accomplished through a COM_REQ command in the 90-30/90-70 CPU. Baud rates of up to 19.2k are supported in either RS232 or RS422/RS485 mode. Most popular handshaking, parity, stop bits and RTU transfer (ANSII [7-bit] or ASCII [8-bit]) modes are available.

1.2 System Requirements

The RTM705 requires CPU331 or higher Series 90-30 CPU; its power consumption is 425mA @ 5VDC. The RTM700 requires a Series 90-70 CPU; its power consumption is 1000mA @5VDC. The RTM700 and the RTM705 do not operate in an expansion rack.

1.3 Physical Layout of RTM705

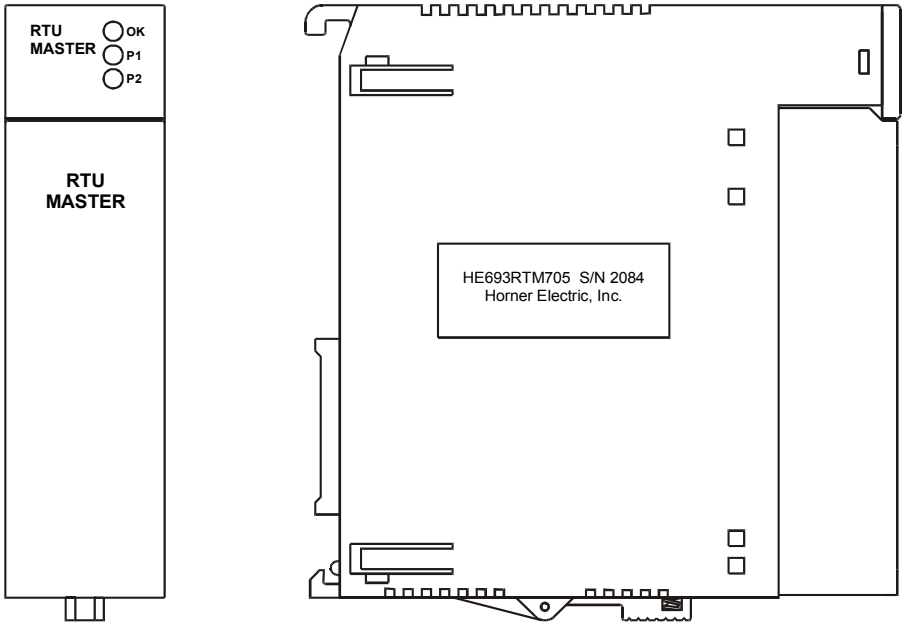


Figure 1-1. RTM705 Module

1.4 LED Operation of RTM705

The RTM705 module features three indicating LEDs; OK, P1 (port 1), and P2 (port 2). At power-up, the OK light flashes to indicate that the module is cycling through its initialization self-tests. After the self-tests are complete, the OK LED remains solidly lit. The P1 and P2 port lights become solidly lit after hardware initialization. After the PLC ladder logic initializes a port with a COM_REQ, it checks for slaves. If a slave(s) is found and is working correctly, the LEDs go out, and thereafter, flicker with port activity. If a fatal error occurs, the P1 LED flash an error code. Errors are listed in Table 3.7 (Page 31).

1.5 Physical Layout of RTM700

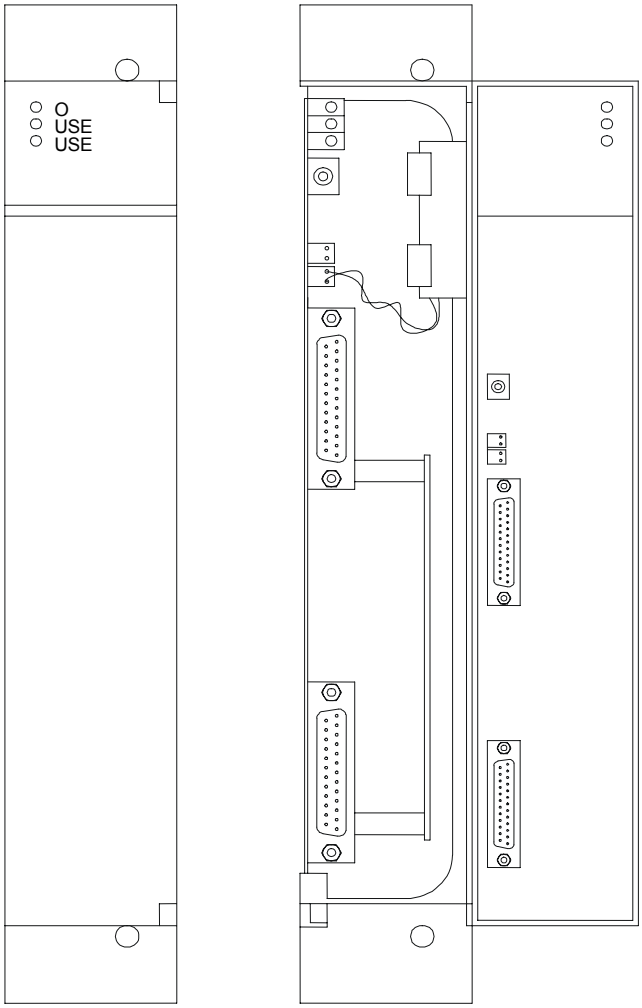


Figure 1.2 - RTM705 Module

1.6 LED Operation of RTM700

The RTM700 module features three indicating LEDs; MODULE OK, USER 1 (port 1), and USER 2 (port 2). At power-up, the MODULE OK light flashes to indicate that the module is cycling through its initialization self-tests. After the self-tests are complete, the MODULE OK LED remains solidly lit. The USER 1 and USER 2 port light becomes solidly lit after hardware initialization. After the PLC ladder logic initializes a port with a COM_REQ, it checks for slaves. If slave(s) is found and is working correctly, the LEDs will go out and thereafter flicker with port activity. If a fatal error should occur, the P1 LED will flash an error code. Errors listed in Table 3.7 (page 31).

1.7 Technical Support

For assistance, contact Technical Support at the following locations:

North America:

(317) 916-4274
www.heapg.com

Europe:

(+) 353-21-4321-266
www.horner-apg.com

CHAPTER 2: INSTALLATION

2.1 RTM705 Mounting Requirements

The RTM705 Module is designed to plug into any series 90-30 local slot. The RTM705 module is not compatible with the Series 90-30 embedded CPUs (CPU311/313/323), but requires a CPU331 model or higher. Please refer to the 90-30 Installation manual for information on installing the module.

2.2 LogicMaster/VersaPro Configuration for RTM705

The RTM705 is configured in LogicMaster/VersaPro as an IC693PCM301, Programmable Coprocessor Module (PCM). Set the cfg mode parameter to “PCM Configuration”. A sample VersaPro configuration screen is shown in Figure 2.1 below:

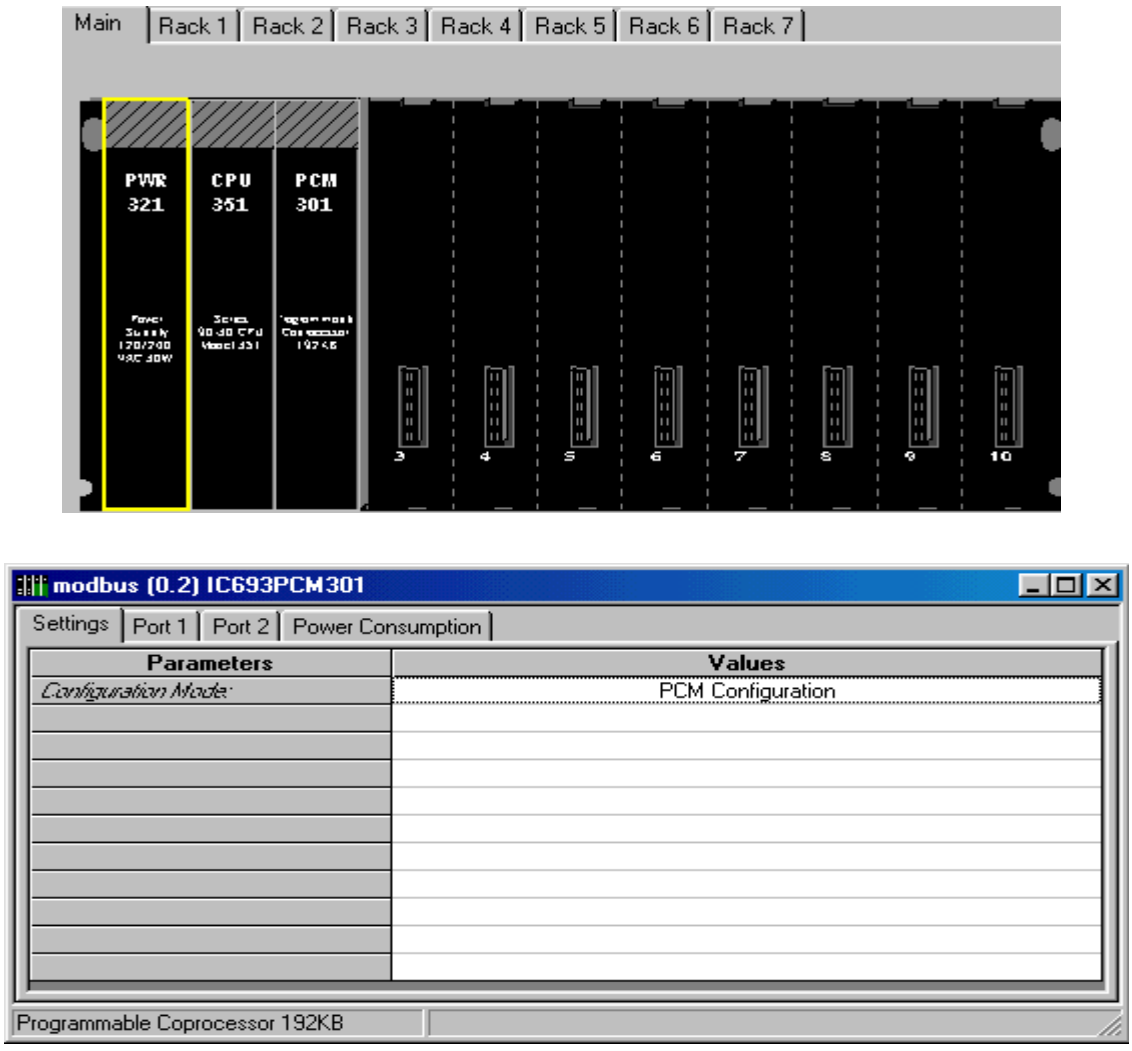


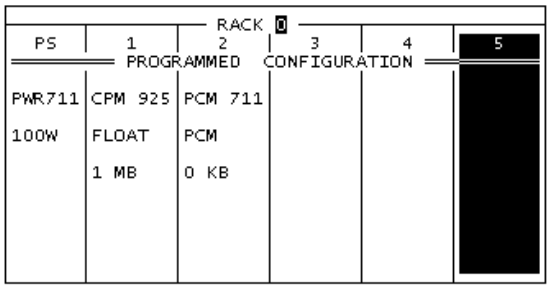
Figure 2.1 - RTM VersaPro Configuration Rack View (top) and Config. Parameter (bottom).

2.3 RTM700 Mounting Requirements

The RTM700 Module is designed to plug into any series 90-70 local slot. The RTM700 requires a 90-70 CPU. Please refer to the 90-70 Installation manual for information on installing the module.

2.4 LogicMaster Configuration for RTM700

The RTM700 is configured in Logicmaster as a IC697PCM711, Programmable Coprocessor Module (PCM). From the LogicMaster Configuration package, select “I/O Configuration” (F1), “Other” (F8), then “PCM” (F1). After selecting the IC697PCM711, press <Enter>. Set the cfg mode parameter to “PCM CFG”. A sample LogicMaster configuration screen is shown in Figure 2.2 below:



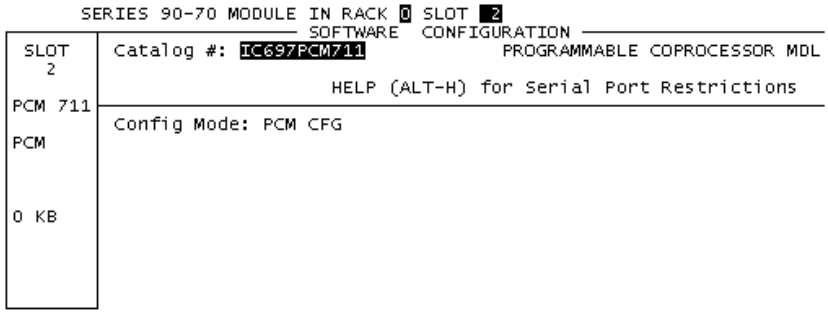


Figure 2.2 - RTM Logicmaster Configuration Rack View (top) and Zoom View (bottom)

2.5 Serial Port Pin-outs

a. Both the RTM700 and the RTM705 feature two ports, which utilize a 25-pin D-sub for each port.
b. With the exception of RTM705 port 1, both ports support RS-232 and RS-485.
Note: RTM705 port 1 only supports RS-232.
c. The RTM705 requires the included 'Y' cable in order to access its two ports.
Note: The diagram for the RTM705 multiplied port (module connector) is provided for custom wiring.
d. Ports 1 and 2 on the RTM700 **contain signals for both** RS-232 and RS-422/RS-485 types of communication. The diagrams on the following pages apply to the RTM705 as well as the RTM700 (taking into consideration that the communication ports on the RTM700 contain signals for both RS-232 and RS-422/RS-485 types of communication).

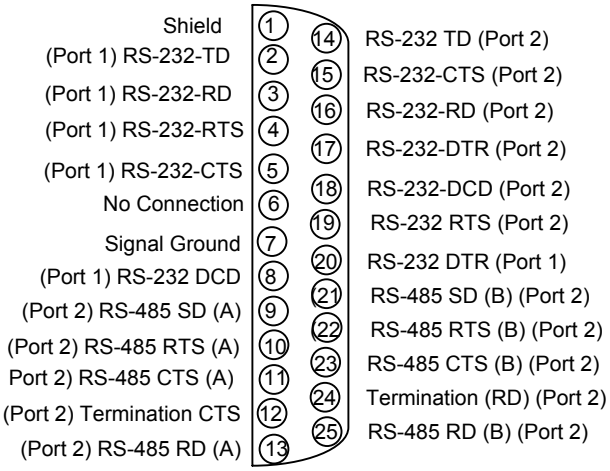
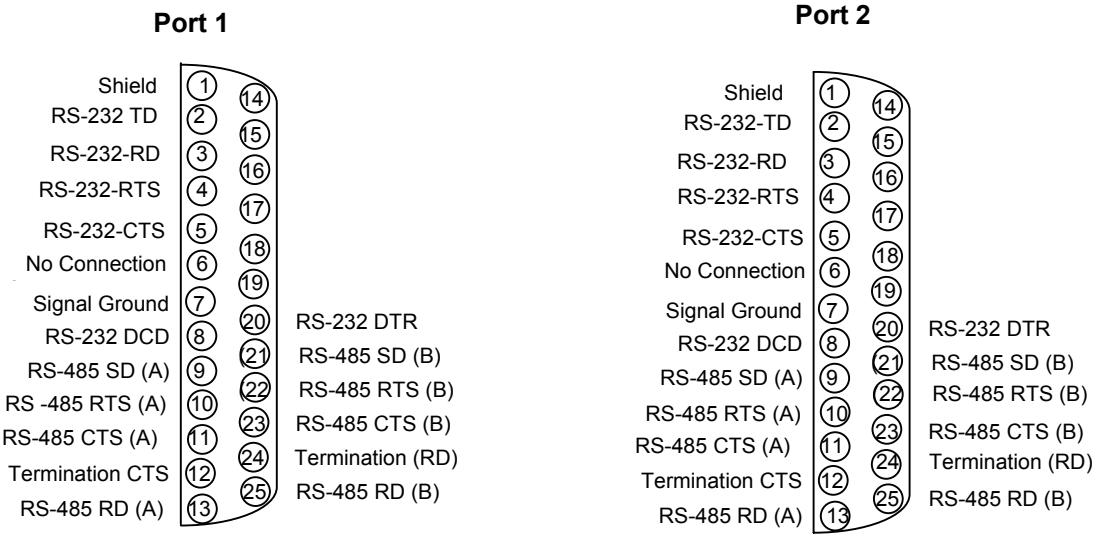


Figure 2.3 - Main Port Pinout for RTM705 Module Connector

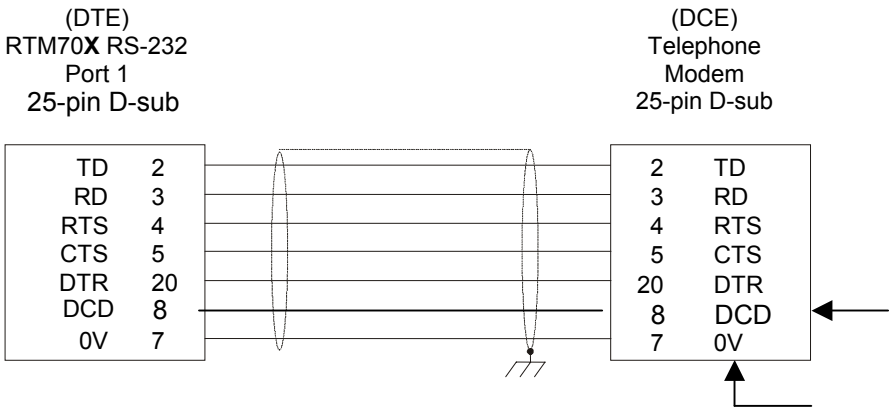


Note: For Port 1, Pins 9-13 and 21-25 are available for RTM**700** only.

Figure 2.4 - Ports 1 and 2 Pin-outs for RTM700/705

Note: (**RTM700**: Ports are physically located on the RTM700.) (**RTM705**: Ports are physically located on the "Y" cable included with the RTM705.)

2.5.1 RS-232 Connections



Note: Modem operation requires all of the displayed handshake signals.

Figure 2.5 - RTM700/705 RS-232 to Telephone Modem Wiring

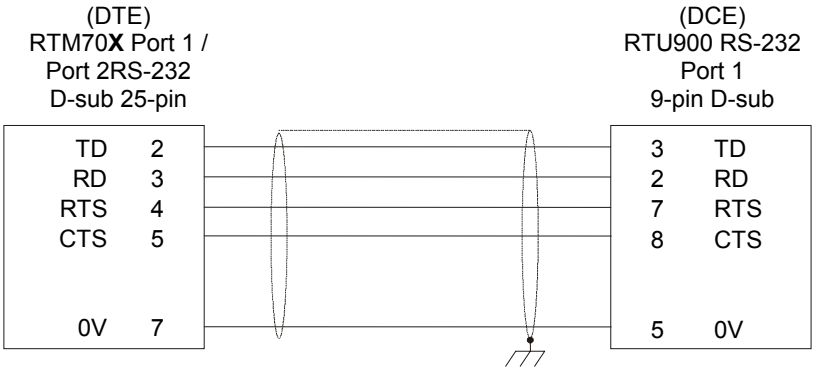


Figure 2.6 - RTM700/705 RS-232 to RTU900 RS-232 (Port 1) Wiring

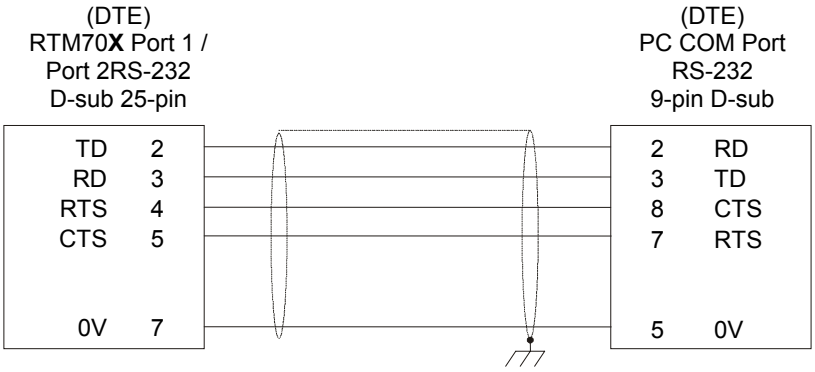


Figure 2.7 - RTM700/705 RS-232 to PC COM Port Wiring

2.5.2 RTM to OCS

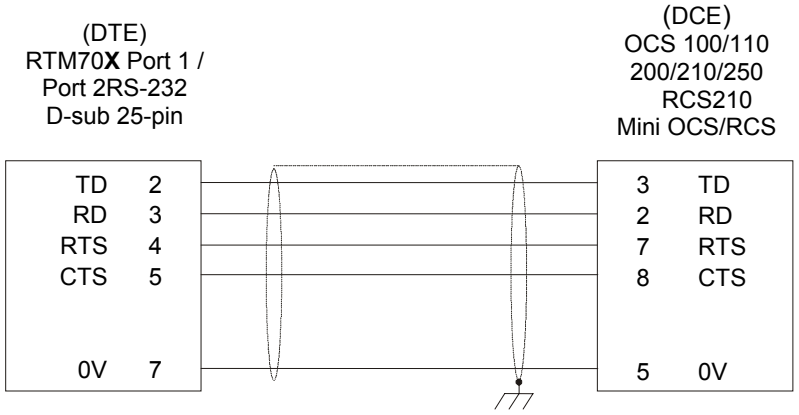
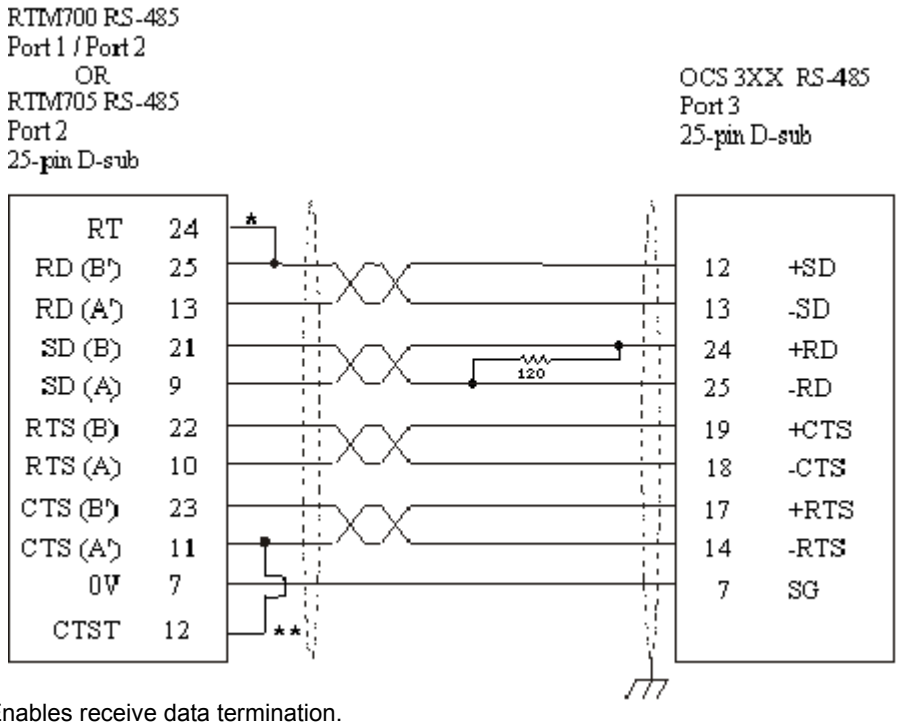


Figure 2.8 - RTM700/705 RS-232 to OCS/RCS



* Enables receive data termination.
** Enables CTS termination. RTS/CTS handshake optional.

Figure 2.9 - RTM700/705 RS-485 to OCS3XX Port 3

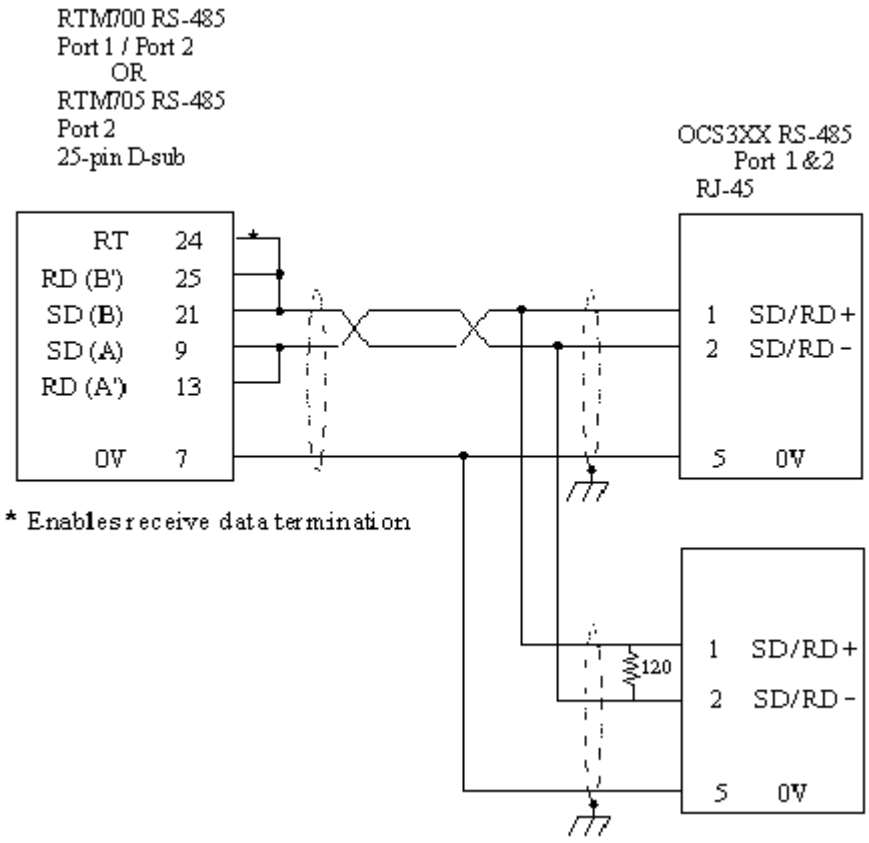


Figure 2.10 - RTM700/705 RS-485 to OCS3XX Port 1 & 2

2.5.3 RS-422 Connections

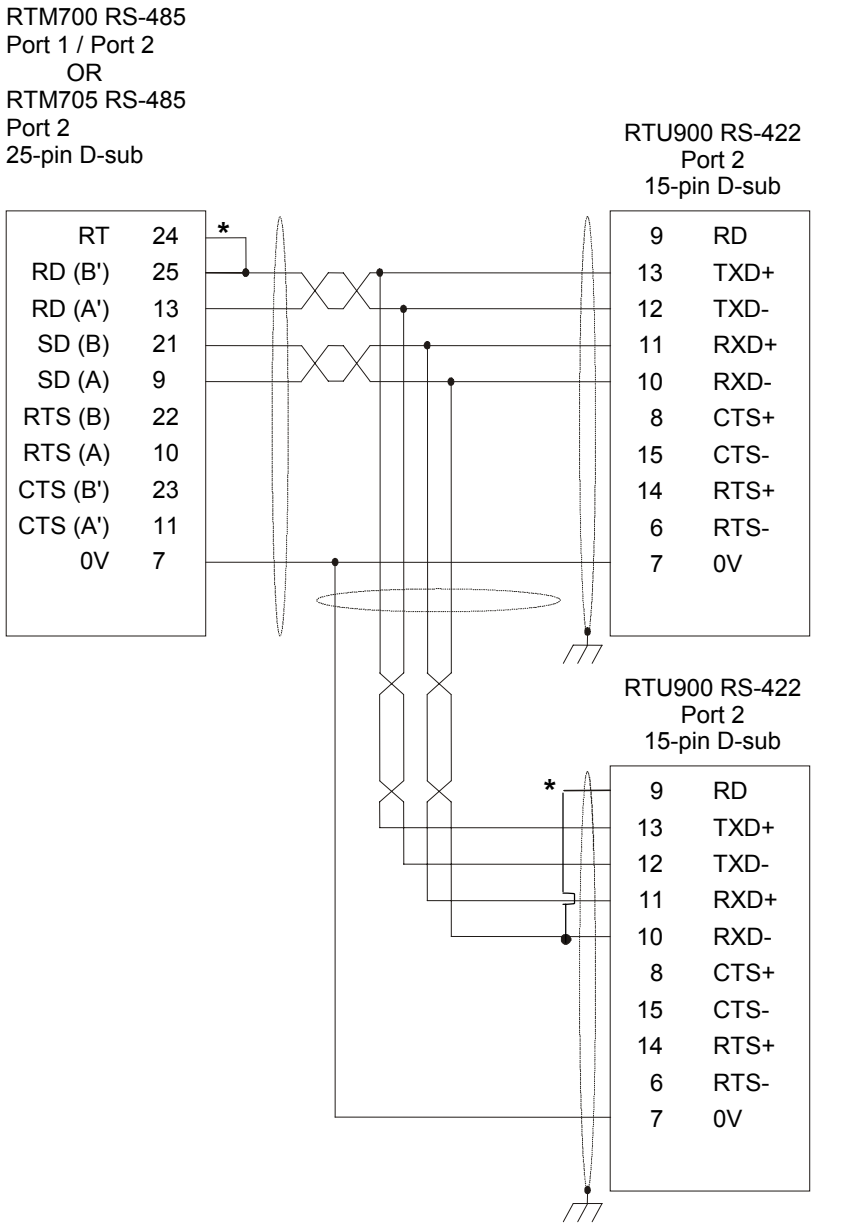
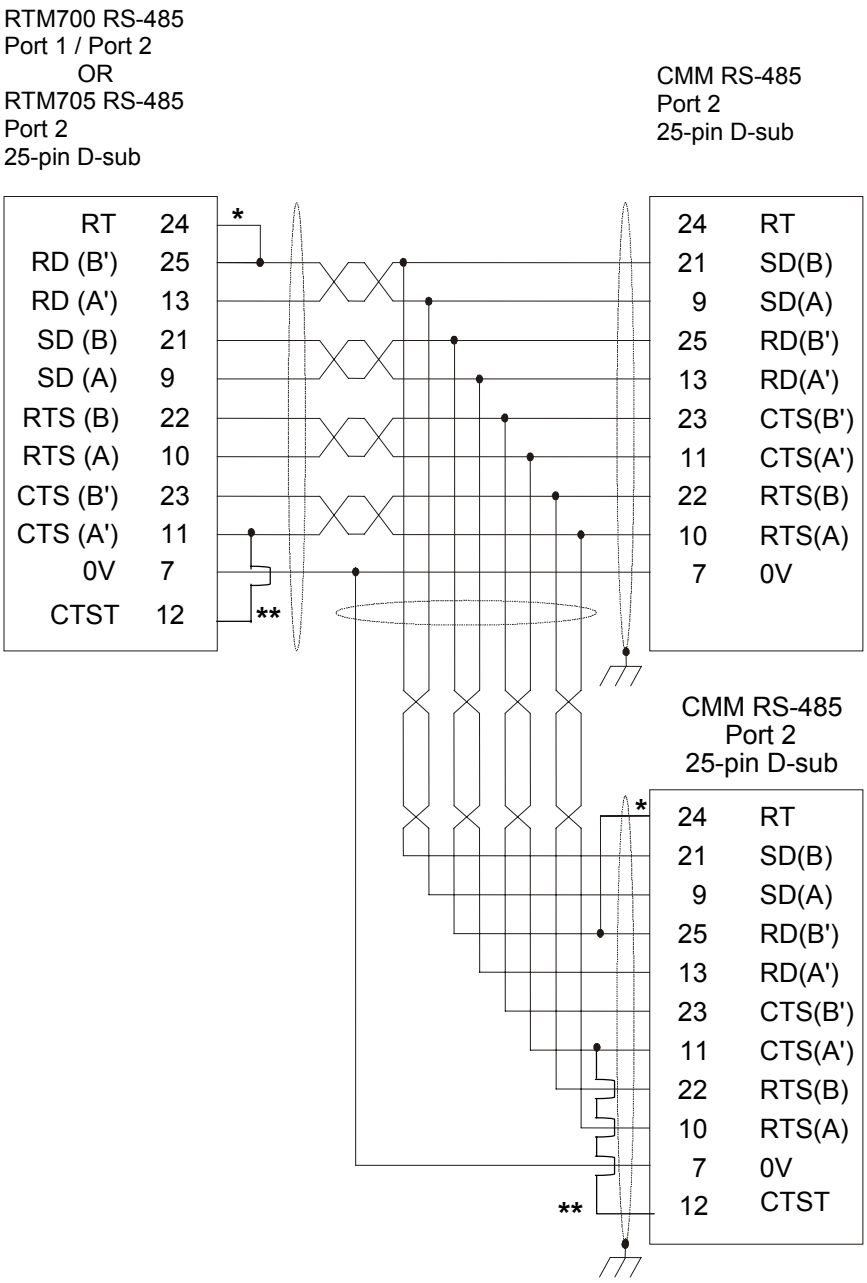


Figure 2-11. RTM700/705 RS-422/RS-485 to RTU900 RS-422 Wiring (4-Wire Multi-drop)



* Enables receive data termination

**Enables CTS termination. RTS/CTS handshake optional.

Figure 2-12. RTM700/705 RS-422/RS-485 to CMM311 RS-422/RS-485 Wiring (8-Wire Multi-drop)

2.5.4 Two-wire RS-485 Connections

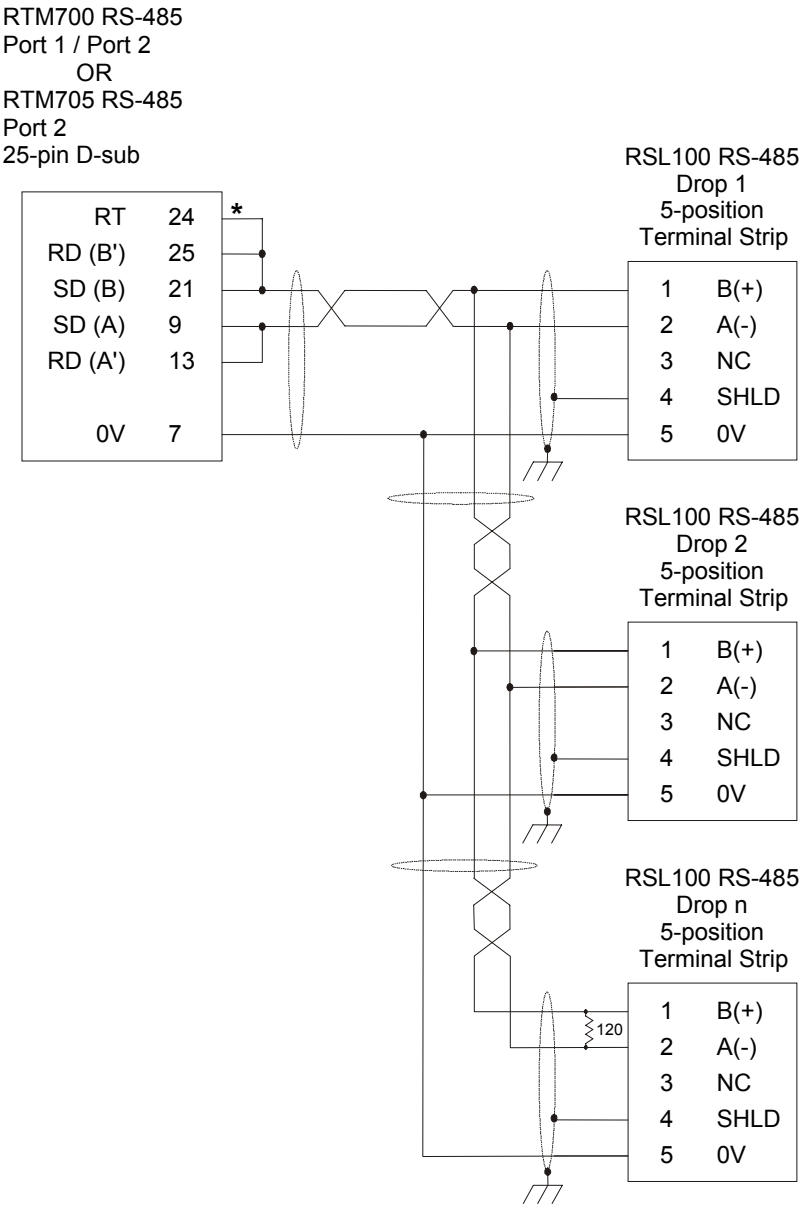


Figure 2-13. RTM700/705 to RSL100 RS-485 (2-wire multi-drop) Wiring (The RSL100 is an RTU option card for the GE AF-300E$ Adjustable Frequency Drive).

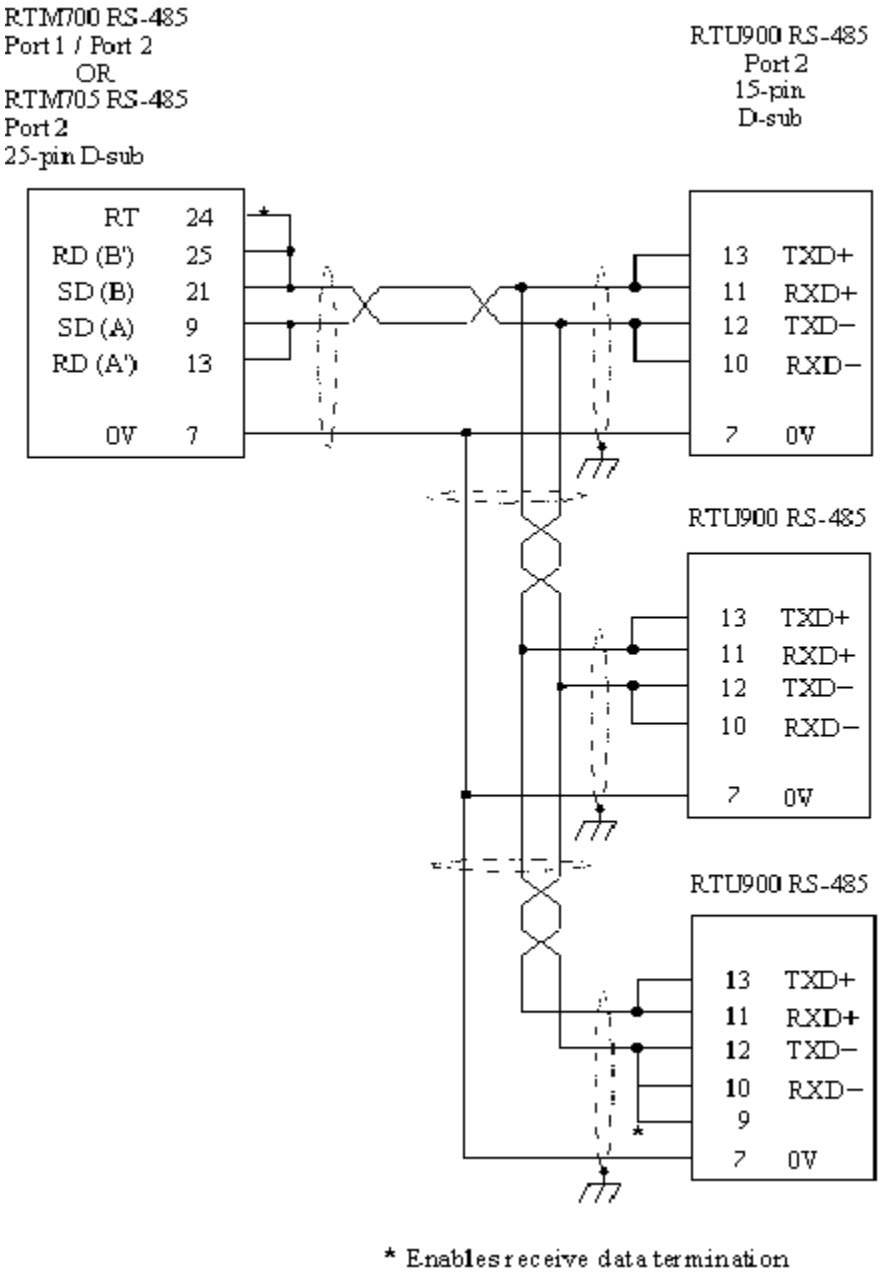


Figure 2-14. RTM700/705 RS-485 to RTU900 RS-485 wiring (2-Wire Multi-drop)

CHAPTER 3: OPERATION

3.1 RTU/Modbus Protocol

The Modbus/RTU protocol uses a Master/Slave protocol that can support a common bus of one master and up to 247 Modbus Slaves. That common bus can be extended from direct wiring to radio and telephone modems. As a RTU master, commands are constructed and issued to an addressed slave. Only the addressed slave is expected to respond to the command. Message integrity is assured through use of checksums included in a message. Should a slave receive a message with a bad checksum, no response is returned. This master provides a configurable modem turn around time-out parameter and retries sending the message twice. Note: When the manual refers to the RTM, this includes the RTM700 **and** RTM705.

Before the RTM can send acceptable RTU commands, it must be configured to the frame protocol used by the slave RTU's (or interfacing modems).

3.2 Initialization

On power up, the RTM waits for an initialization through a COM_REQ before beginning to process RTU commands. While waiting for a COM_REQ, the associated port LED remains solidly lit. However, before the COM_REQ can be issued, a pair of lists, SCB(Slave Control Block) and MCB(Message Control Block) must be preloaded in an appropriate PLC reference data table [%R, %AI, %AQ] with scan information. These lists specify the Slaves, which are scanned and the RTU command(s) which are sent to the slave RTUs. Each communications port requires its own set of lists and associated COM_REQ.

3.2.1 Slave Control Blocks (SCBs)

A Slave Control Block (SCB) is a group of 15 words, which must be configured (with the exception of answer mode) for each slave RTU to be accessed. These SCB's are all stored together in a contiguous space of PLC reference data words, which may be %R, %AI or %AQ. So for example, if you have three slaves you need three SCBs. If you start the first SCB at address %R301, the next SCB should start at %R316 and the next at %R331 to be contiguous. Each SCB provides such information as the slave RTU ID, delay between updates (Secs) and the call string (if dial out mode). The number of SCB's or stations accessible is limited to 32.

The following descriptions refer to Table 3.1.

1. **Status Indicator**. The Slave Control Block (SCB) Status Indicator under normal successful operation is changing at some rate between a value of 0 and 1. If a problem of some nature occurs, an error code greater than 1 is written to the SCB Status Indicator. These error codes are listed in Table 3.9.

To restart a SCB that has erred, a zero is written to the Status Indicator.

2. **Failed MCB Index**. The failed MCB index word specifies the number of the MCB that failed starting with zero. A 0 indicates the first MCB failed, a 1 indicates the second MCB failed and so on.
3. **Number of Retries**. This word contains the number of retries to access the slave. Each retry increments the count.
4. **Station ID**. The Station ID is the slave ID number, which is placed in the RTU message. A slave RTU with that ID is expected to respond.
5. **Update Delay**. The update delay specifies the amount of time between accesses to the slave of this SCB. A 0 = no delay between updates. 1 = 6 seconds. For example, a 2 in this register will cause the slave to be accessed every 12 seconds. Since each slave has its own update delay entry, priorities can be established.
6. **Dial String Characters**. The dial string is used when a telephone modem is attached. This specifies the telephone number and control characters used for dialing.

The status_indicator, mcb_index and number_of_retries may be written during operation and are reset during initialization.

Table 3.1 – Slave Control Block (SCB)				
Address	**Description**		**Address**	**Description**
starting address	Status Indicator		address + 8	Dial String Characters 8 & 7
address + 1	Failed MCB Index		address + 9	Dial String Characters 10 & 9
address + 2	Number of Retries		address + 10	Dial String Characters 12 & 11
address + 3	Station ID		address + 11	Dial String Characters 14 & 13
address + 4	Update Delay		address + 12	Dial String Characters 16 & 15
address + 5	Dial String Characters 2 & 1		address + 13	Dial String Characters 18 & 17
address + 6	Dial String Characters 4 & 3		address + 14	Dial String Characters 20 & 19
address + 7	Dial String Character 6 & 5			

The least significant byte in word 6 is the 1st digit to be dialed. The most significant byte in word 6 is the next. This storage pattern continues up to word 11. Digits are entered as a ANSII hexadecimal digit (i.e., '6' is entered as a 0x36)

3.2.2 Message Control Blocks (MCBs)

A Message Control Block (MCB) is a group of 6 words, which must be configured for each RTU command which is sent. These MCB's are all stored together in a contiguous space of PLC reference data words, which may be %R, %AI or %AQ. So for example, if you have three MCBs and If you start the first MCB at address %R801, the next MCB should start at %R807 and the next at %R813 to be contiguous. If multiple MCB's are defined for a particular ID, they are executed in the order encountered and do not need to be grouped together as belonging to the same node ID. The whole of the MCBs still need to be contiguous. All the commands for a currently selected slave are sent before accessing the next slave. Each MCB specifies the associated slave RTU ID, the RTU command, RTU data offset, RTU data length, and the type and offset of the PLC reference data which is accessed. The number of MCB's is limited to 1024 per channel.

Table 3.2 - Message Control Block (MCB)	
Address	**Description**
Starting address	Station ID
address + 1	RTU Command (See table 3.3)
address + 2	RTU Reference Offset
address + 3	RTU Reference Length
address + 4	PLC Reference Type
address + 5	PLC Reference Offset

The station ID is used to determine which slave is associated with this command.

The **RTU Command** specifies the Modbus type command the slave should respond to. The commands (word values) in the table below are supported. Any other command generates an error.

Table 3.3 – Supported Modbus/RTU Commands

Decimal Value	Description		Traditional Modbus Addressing	Extended Modbus Addressing
1	Read Coil Status		1	1
2	Read Input Status		10,001	100,001
3	Read Holding Register		40,001	400,001
4	Read Input Register		30,001	300,001
5	Force Single Coil		1	1
6	Preset Single Holding Register		40,001	400,001
7	Read Exception Status		----------	----------
15	Force Multiple Coils		1	1
16	Preset Multiple Holding Registers		40,001	400,001

The **RTU reference offset** specifies the starting data offset on the slave RTU to access.

The **RTU reference length** specifies the number of data items on the slave RTU to access. (up to 250 bytes per MCB).

The **PLC reference type** specifies the local data reference table type where data on the PLC is accessed. The types (word values) in the table below are supported. Any other type value generates an error. (Byte oriented cards must be referenced to bit references)

Table 3.4 – Supported PLC Reference Types and Offset Ranges

Decimal Value	Description		Decimal Value	Description
70	%I		8	%R
72	%Q		10	%AI
74	%T		12	%AQ
76	%M			

The **PLC reference offset** specifies the local data reference table offset where the data on the PLC is accessed. Specifying an offset, which exceeds the reference data table size, generates an error.

A **Single MCB Mode** is available which provides the ability to dynamically change the MCB fields while the RTM is running. To activate this mode the Interactive bit (bit 5 Fig. 3.3) in the Port Parameter Word (COMM_REQ buf) must be set when the COMM_REQ is submitted. To change the MCB you simply create a new block move in ladder to move the new values into the MCB registers. Moving a zero into the command word shuts the MCB off. Note that this mode increases - both the amount of time used by the PLC I/O scan and the response time of the RTM. Moving a new command into the MCB should be synchronized. The entire SCB should be written in a single scan. Changing in the middle of a comm transfer may have unexpected data momentarily.

One Shot MCB. When using the One Shot method, The MCB is set up as described previously with one exception. The high bit of the command word must be set high. For example, if you want to do a command 2, you would enter a 16# 8002. A command 6 would be a 16# 8006 etc. The interactive bit must be set. If the high bit is set, but the Interactive bit is not set in the Port Parameter Word, an error

165 will occur in the SCB status word. The RTM will execute the command that was originally placed in the MCB during the start up routine. Once that command has been successfully carried out and a proper response has been returned from the slave, the command word is set to zero. A zero in the command word essentially turns the MCB off and signifies that the MCB is ready for a new command. When sending a new command, the other values of the MCB can be re- written as well. The new values must all occur within one scan or the action will result in error.

3.2.3 Initialization Communications Request (COM_REQ)

Once the SCB and MCB lists are initialized, a channel needs to be opened between the RTM port and the PLC. The starting address of each list along with other initialization parameters are passed to the RTM through a COM_REQ. Each of the communication ports requires its own set of lists and associated COM_REQ. The PLC selects which port a COM_REQ is addressed to through the TASK input to the function block. Selecting task 101 (16#65) initializes port 1 and task 102 (16#66) initializes port 2 on the RTM. Since the RTM is slower to power up than the PLC, a delay of 5 seconds should be implemented in the ladder code before the COM_REQ is called. Additionally, the PLC should set the provided status reference to zero before making the call. Only the first comm_req is accepted.

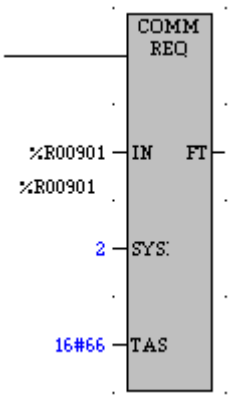


Figure 3.1 – COM_REQ Function

IN The starting register of the COM_REQ Data

SYS A hexadecimal value which gives the rack and slot location of the RTM module. It is in the following format:

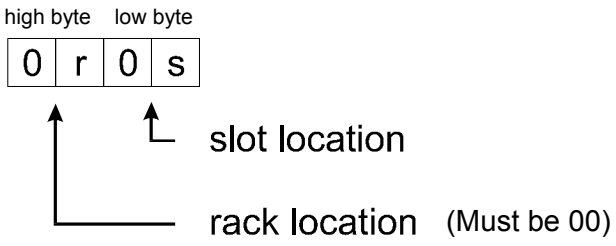


Figure 3.2 – COM_REQ Function

TASK Identifies which port is being initialized. The following values are valid:

Table 3.5 – Port Initialization		
Port	**Hex Value**	**Decimal Value**
1	0065	101
2	0066	102

After the COM_REQ call is made in NO_WAIT mode, the status field should be sampled for a non-zero response. If the RTM initializes without errors, an OK [0x0001] status is returned. If an error occurs, the associated value is placed in the status field. If no status response is received from the COM_REQ after waiting one second, the RTM should be considered unreachable or in fatal error. The COM_REQ FT output can also provide an indication of fault conditions.

Table 3.6 – COM_REQ Data Format				
Address	**Description**		**Address**	**Description**
start address	Data Block Length (11)		address + 9	SCB Pointer Offset
address + 1	No Wait (0)		address + 10	Number of MCBs
address + 2	Status Pointer Type		address + 11	MCB Pointer Type
address + 3	Status Pointer Offset		address + 12	MCB Pointer Offset
address + 4	Idle Timeout (0)		address + 13	Port Baud Rate
address + 5	Maximum Comm Time (0)		address + 14	Port Parameter Word
address + 6	RTM Mode		address + 15	Modem Turnaround Time (mS)
address + 7	Number of SCBs		address + 16	Radio CTS Delay Time (ms)
address + 8	SCB Pointer Type			

Numbers in parentheses are required entries.

Data Length

Specifies the number of additional data words in the COM_REQ. This value must be 11.

No Wait

You must specify No Wait (0) to guarantee PLC does not wait indefinitely should the RTM not respond to the COM_REQ.

Status Pointer Type and Offset

Specifies the PLC word reference, which receives the RTM status after initialization. This reference should be monitored by the PLC even after initialization. This is where, should there be any, the com_req errors are put. (See table 3.8). A one in this address specifies that the com_req has executed successfully. The offset number entered is the address number minus one. So if you wanted to use %R100 as the status word you would enter 99 for the offset.

RTM Mode

Specifies the type of communication.

a. Direct (1) mode provides RS232 direct communication between the RTM and a single slave RTU. RTS is driven high during transmission to activate any external device. CTS is ignored. Although not typical in this mode, multiple stations can be scanned through a multi-drop line converter device.

b. Multidrop (2) mode provides RS485 communications between the RTM and multiple slave RTU's. This mode activates the RTM RS-485 drivers on the communications port. The transmission driver (while using half duplex) is only active while sending commands.

c. Radio modem (3) mode provides RS232 communications to an external radio modem. RTS is asserted when the RTM has a message to transmit. The RTM then samples the CTS line which should be asserted by the radio modem when the transmitter has come up to power. A configurable radio modem time-out-timer is provided that releases the RTS line after a timeout with no CTS response and immediately issues a Modem-did-not-respond error. The receive line is monitored immediately after a send and does not depend on DCD.

d. Originate (4) mode provides RS232 communications to an external modem which may be used to originate calls to remote slaves. An entry is provided in the SCB for the user to provide a packed string of ANSII hexadecimal represented numbers and standard 'ATDT' control characters. The RTM issues the call string and wait for a modem response. The modem must be compatible with verbal extended 'ATV0X4' commands. Once the connection is made, the MCB is processed as normal. When all messages for that station is complete, the connection is broken.

e. Answer (5) mode provides RS232 communications to an external modem which may be used to answer calls placed by the remote slaves (compatible only with Horner APG Modbus slaves). Once a call is received, the RTM issues a command to determine the station ID of the calling slave. Then the messages for that station is issued. Once all the messages for that station is complete, the connection is broken.

SCB Number, Pointer type and offset

Specifies the number of SCBs, the pointer type, and the starting location in PLC memory. Since addressing in the Modbus protocol starts at zero, the number entered for the offset should be the starting address minus one.

MCB Number, Pointer type and offset

Specifies the number of MCBs, the pointer type, and the starting location in PLC memory. Since addressing in the Modbus protocol starts at zero, the number entered for the offset should be the starting address minus one.

Port BAUD rate

Specifies the baud rate of the associated RTM port. The following rates are supported: 300, 600, 1200, 2400, 4800, 9600, 14400, 19200. The values are entered as the unsigned decimal value of the baud rate.

Port Parameter word

Specifies the frame protocol:

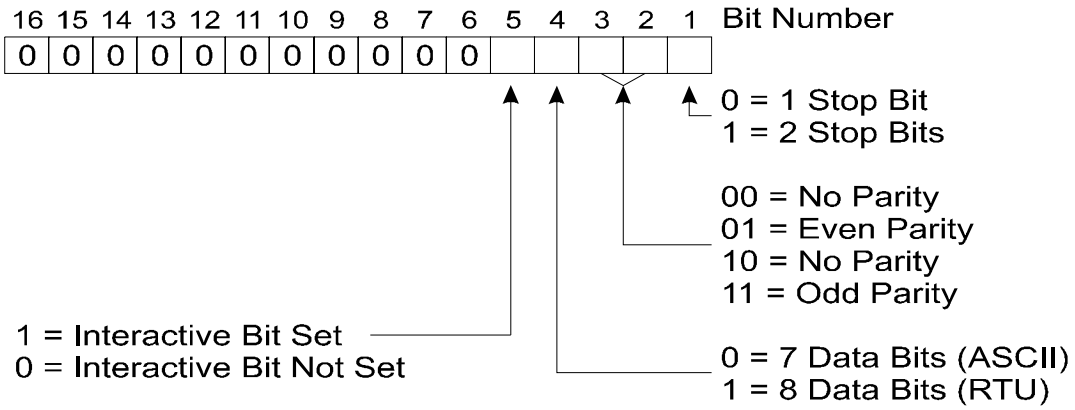


Figure 3.3 – Frame Protocol Modem Turn Around Timer

Specifies the amount of time in milliseconds (i.e. 2000=2sec) that the master waits for a response from the slave before recording a timeout error. Setting this value to zero defaults to 2.5 Sec.

Radio CTS turn around timer (radio modem mode only)

Specifies the amount of time in milliseconds (i.e. 100=100mS) that the master waits for CTS (transmitter ready) after asserting RTS. Setting this value to zero defaults to 250mSec.

Once the COM_REQ is issued, the RTM is initialized and a status is returned in the provided status field.

3.3 Normal Operation

Upon receiving a valid COM_REQ, the RTM initializes the associated port. The port is initialized to proper frame protocol and if a modem is attached, it is sent an initialization string which sets the connection and response operation. Next, the associated MCB list is read from PLC memory into local memory. Changes to the SCB list affect normal operation.

3.3.1 Scanning the Slaves

After the RTM is initialized, it begins to scan the SCB's. An SCB is ready when its associated Update Delay time has passed since the last time this slave was serviced. If the user entered an SCB Update Delay of zero, the station is serviced immediately each time it is scanned. Once the RTM selects a SCB to service, the associated status field is set to zero to indicate that this slave is being serviced. If the RTM is in originate mode, it attempts to call that slave. If connected successfully, the entire MCB list is scanned for a matching slave ID. Those MCB's with a matching ID are used to construct individual RTU messages which are sent to that slave in the order encountered. The speed at which the MCBs are executed is approx. 10 - 12 PLC scans per MCB.

If a response to an RTU message is corrupted or not returned, the RTM retries the command twice. If there is still no valid response, an error is registered in the status field and that slave no longer is serviced until the ladder program clears the status field in the associated SCB. Each retry increments a Retry Count in the SCB, which can be used to determine how 'clean' the connection is with the slave. Logical errors such as Bad-RTU-Command or Data error, or Bad-PLC-Type or Address error is <u>not</u> retried. In addition, the index to the associated MCB (which caused the logical error) is entered in the SCB. If all the MCB's for an SCB are sent without any errors, the status field in the associated SCB is set to a '1' and indicates that this slave has been serviced without any errors.

3.3.2 Status Registers

While the RTM is running, the ladder program should continue to sample the status reference provided in the COM_REQ. This provides an indication of a system level fault such as the modem failing to respond. Additionally, the ladder program should also monitor each SCB's control block status. This allows the user to access the current state of a slave. If a slave is marked in error (value > 1), that error must be cleared before communications is continued with that slave. When the RTM is running, the PLC status is being sampled. Should the PLC be taken out of run mode, the RTM halts operation and wait for re-initialization. Pressing the reset button for less than 5 sec. (soft reset) halts operation and cause it to wait for re-initialization through new COM_REQ's:

> **Warning: Pressing the reset button for longer than 5 sec. (hard reset) causes the RTM to halt. To restart from halted state, press the reset button for less than 5 sec. or power cycle the rack. Resetting the RTM DOES <u>NOT</u> reinitialize the ports. That has to be done in ladder.**

3.4 Originate Mode Considerations

At start-up, the RTM issues an initialization string to the modem. If the appropriate response is not received from the modem, a MODEM DID NOT RESPOND error is recorded in the system status field. Thereafter, the RTM continues issuing the initialization string until the proper response is received. Once the modem response properly from the initialization string, the system status is set to 1 (OK). The initialization string sent to the modem at start-up includes the parameter "ATS7=55". This programs the modem to wait 55 seconds for a response from the remote. If the modem timer times out without a response from the remote, a NO CARRIER response is issued from the modem and recorded in the SCB status field. However, if no response is received from the modem within 60 Sec of a call, the RTM registers a Modem-Bad-Response in the SCB status field. The initialization string also includes "ATV0X4" such that 'Busy' or 'No Answer' conditions are recognized and registered in the SCB Status field. Should any of these error be registered, the associated slave no longer is serviced until the associated SCB's Status field is cleared.

Upon scanning and when the RTM determines that a slave needs service in 'Originate' mode, it sends the dial string defined in the SCB's Dial string field. This string is prefixed with 'ATDT' and sent to the modem at the defined baud and frame configuration. The defined string must be terminated in the SCB with a single byte of zero value. The characters may be those used by typical modems. An invalid dial string generates an Invalid_dial_string_error (8Bh).

Once the 'CONNECTED' response code is received from the modem, the RTM waits 1 second for the remote to settle and begins sending associated MCB messages. Bad responses are retried as described above. When the RTM has completed transmission of the MCB's or a communications error has occurred, the DTR line is cleared for 500mSec. to 'hang-up' the modem and the appropriate status is written to the SCB Status field.

In either Originate or Answer modes, the RTM samples the CTS to verify that the modem is powered up and on-line. Should the CTS line from the modem drop at any time after the RTM705 is initialized, the RTM goes into a continuous loop waiting for the CTS to be restored. While in that loop, SCB's are not be scanned and a NO CTS error is written to the system status field. Once CTS is restored, the modem is re-initialized, the system status field is updated with '1' (OK) and the SCB'S are scanned.

3.5 Answer Mode Considerations

In 'Answer' mode, only the first SCB is used. It reflects the status and slave address of the last (or current if on-line) slave that called in. This SCB can be monitored by the ladder logic program to determine if a particular slave is having communications problems. If an error occur, the status is read as soon as possible since the next incoming message clears the ID, Retry and Status fields. Incoming calls are not ignored if the status field contains an error value.

On initialization, the modem is sent an 'ATS0=1' command to place the modem in answer mode. Then the modem ring signal is monitored waiting for an incoming call. When a slave 'calls in', a connection is established and the SCB Status field is set to '0' to indicate busy. After a 1 second delay, the RTM sends a custom RTU message to determine the ID of the calling slave.

Once determined, the first SCB is filled with the ID of the slave calling in and the Retry field set to zero. Thereafter, the MCB list is scanned for entries associated with that station. Bad responses are retried as described above. When the RTM has completed transmission of the MCB's or a communications error has occurred, the DTR line are cleared for 500mSec. to 'hang-up' the modem and the appropriate status are written to the SCB Status field.

3.6 Pager Considerations

When in 'Originate' mode, an SCB may be configured to call a pager utilizing the special dial commands of the modem. The SCB ID field must be set to 0FFH and the SCB Delay field is still active to provide a forced delay between calls. A call to the pager is activated by setting the SCB Status field to zero. The RTM then attempts to call the pager. A SCB Status of '1' indicates that the call was successful; otherwise, the Status field contains the error.

A pager call string to call a pager supplier and issue a 'pin' number should provide the following string.

xxx-xxxx@yyyy;

where 'x' is the number of pager.
where '@' causes the mode to wait for an answer and then 5 sec of silence.
where 'y' is the 'pin' number of the pager.
where ';' is required to hangup after the pin number is sent.

Generally 'y' can be any hexadecimal representation of number. Additionally, a comma (,) can be inserted to cause two second delays between numbers.

Should the remote not answer or provide 5 sec. of silence, the SCB status indicates NO ANSWER.

3.7 Error Codes

3.7.1 Fatal Errors

If a fatal error is encountered, the LED for the First COM port flashes a code annunciating the error. The error codes are listed in Table 3.7.

Table 3.7 – Fatal Errors P1 LED Blinks Code

Code	Description		Code	Description
1 count	Allocate Timer Fault		5 counts	Com Device Configuration Fault
2 counts	Device Open Fault		6 counts	Backplane Device Configuration Fault
3 counts	Device Read Fault		7 counts	PLC Write-Protected (password)
4 counts	Device Write Fault		8 counts	Com write buffer overflow

3.7.2 COM_REQ Errors

The COM_REQ Status Register under normal successful operation contains a value of 1. If a problem results from the initialization of the RTM, or if a modem error occurs, a diagnostic error code is written to the COM_REQ status register. These codes are listed in Table 3.8.

Table 3.8 – COM_REQ Errors						
Code (Hex)	**Code (Dec)**	**Description**		**Code (Hex)**	**Code (Dec)**	**Description**
00C0	192	Specified bad mode of operation		00C6	198	Specified bad MCB address
00C1	193	Specified bad number Of SCBs		00C7	199	Specified baud rate not supported
00C2	194	Specified bad SCB type		00C8	200	Specified invalid COM_REQ size
00C3	195	Specified bad SCB address		00B0	176	Modem dropped CTS unexpectedly
00C4	196	Specified bad number Of MCBs		00B1	177	Modem did not Respond to initial string
00C5	197	Specified bad MCB type				

3.7.3 Slave Control Block (SCB) Errors

The Slave Control Block (SCB) Status Indicator under normal successful operation is changing at some rate between a value of 0 and 1. If a problem of some nature occurs, an error code greater than 1 is written SCB Status Indicator. These error codes are listed in Table 3.9. To restart a SCB that has erred, a zero is written to the Status Indicator.

Table 3.9 – SCB Errors						
Code (Hex)	**Code (Dec)**	**Description**		**Code (Hex)**	**Code (Dec)**	**Description**
Local Modbus Frame Errors						
00A0	160	No MCB found matching slave ID of on-line station		00A3	163	PLC Reference type doesn't match data type
00A1	161	Unsupported Modbus Command		00A4	164	PLC Reference in MCB exceeds table size
00A2	162	Specified data length Exceeds Modbus frame size		00A5	165	High bit in command field while in scan mode
				00A6	166	Invalid Broadcast Command (Trying to do a read with a broadcast command)
Remote Modbus Frame Errors						
0091	145	Unsupported Modbus Command		0094	148	Failure in remove device
0092	146	Invalid Modbus data offset				
0093	147	Invalid Modbus data				
Transport Frame Errors						
0081	129	Last Failed Response was Timeout		0087	135	Busy
0082	130	Last Failed Response was Bad Checksum (corrupted incoming data)		0088	136	No Answer

Table 3.9 Continued		Transport Frame Errors				
0083	131	No carrier (No answer)		0089	137	Uart did not receive expected stop bit or parity error
0084	132	Bad string sent to modem		008A	138	Message Exceeded Buffer Size (Too much data received)
0085	133	Modem did not respond to command		008B	139	Bad Character in Dial String
0086	134	No dialtone		008C	140	Lost Connection
Note: Any error code above hex number 94 is generated by the slave and Slave documentation needs to be consulted. The 9 in 94 is added by the RTM. So for example, in an error 94, the slave is generating an error 4.						

3.8 Advanced functions

3.8.1 Broadcast Mode

A broadcast mode is available with the RTM. In broadcast mode, it is possible to communicate to all of the slaves by creating a SCB for address zero (station ID=0 for SCB & MCBs). Writes to all of the slaves are possible by utilizing this feature, but reading data from all of the slaves concurrently produces an error code (A6h - invalid cmd for broadcast). **Note: A broadcast must time-out according to the COMM_REQ.turn-around-time before transmitting the next message.**

3.8.2 Station Locking

An optional lock-on-connection is available for this module. 'Locking' is the capability of causing the master to connect to a specified station, remain connected to that station indefinitely and only sending the MCB's associated to that station. This optional mode is enabled by setting the Interactive Bit high and writing a 16#FFFF to a SCB.update delay field. At the end of normal processing of MCB's for a remote station, an update timer is started using the value in the SCB.update field and the current connection is broken. However, if a 16#FFFF is detected in the SCB.update field at that time, the remote remains connected and a new snapshot and update of the associated MCB's are performed. No other remote stations are processed and updating of the current 'locked' remote is continuous until a typical update delay value is written back to the associated SCB.update field. **Note** that even if all the associated MCB commands are the zero command, the connection is maintained. Also **note** that this may 'lock' the connection of a station calling-in in Answer mode. If a fault occurs in 'lock' mode, the connection is maintained; however, no messages are processed until the SCB status field is cleared. With the lock command, a telephone connection can be 'locked' at initialization if a system only has one remote station. Should DCD drop off while connected, indicating a break in the communications path, a E_LOST_CARRIER (8Ch) is registered in the SCB status field. DCD detection is active in both locked and non-locked modes.

To determine if a marked station is currently locked on a connection and is good, locked on a connection and is faulted, or if a lock was tried and failed because of a connection fault, the rules on the next page must be followed.

a. Station Locking Rules

1. A station can only be 'locked' once a connection is established. If a modem mode is being used, the modem must complete its 'connect xxxx' phase with the remote modem before the associated SCB can be locked. Additionally, if ANSWER mode is being used, the Master must also complete interrogation of the remote station for its ID and verify that associated MCB's exist before it can be locked. **Failure to complete a connection on a SCB marked for locking results in taking the station off-line for any further attempts until the connect fault value is cleared from its associated status field. Furthermore, the SCB.status 'locked' bit as defined below is not be set.**

2. A non-modem mode is not considered connected until an associated MCB is found. If an associated MCB cannot be found for a station, the station is taken off-line until the UNKNOWN_ID fault is cleared. Furthermore, the SCB.status 'locked' bit as defined below is not be set (Null command MCB's are acceptable, and a station can be 'locked' if only one associated null command MCB exists).

3. However, once a connection is established and 'locked', certain faults can thereafter occur without breaking the connection. These faults are MCB bad field values and transport faults. To allow the user to recognize that the connection is 'locked' but in error, the high bit in the SCB.status field in addition to the fault value is set. Under this condition, the Master is in a connect idle state waiting for the user to clear the status field to restart the scanning of that particular station.

4. If a modem mode is being used and a station is currently locked, a failure in the connection (such as telco line noise) breaks the locked connection and produce a normal fault in the status field without the high 'locked' bit indication. That station is no longer locked.

5. Once a station is 'locked' and running without faults, the high bit in the status field is set high to indicate locked. The lower status bit values either indicate 1(OK) or 0(BUSY) as described below.

6. If an SCB.update field is set to 0xFFFF and is unable to connect or is interrupted during connection, the SCB.update field still contains an illegal time-out value. Therefore, it is retried upon clearing the associated status field immediately without any additional delay. However, should a 'locked' station fail and other station SCB's exist, they are scanned in order as usual. When the 'locked' station fault status is cleared, it is retried in the order scanned.

7. A 'locked' station rechecks the value of its associated SCB.update delay value at the end of its associated MCB scan. If the value returns to a valid update delay value, the station disconnects, and the high bit of status field is cleared.

3.8.3 *Report by Exception*

Report by Exception (RBE) is a non standard extension of RTU protocol which allows a slave to contact the system master unsolicited via modem. RBE is primarily used for reporting alarm conditions at the slave location that might otherwise not be seen by the RTM for several minutes or longer. It can also be used in cases where communications between master and slave is desired on an exception only basis. This may be due to the expense of the telephone calls involved, where the number of calls is to be minimized to a safe, cost effective level. RBE will only work with Horner APG slaves. The host RTM module detects the call, establishes a connection, and sends out a special command which identifies the slave calling in. Then the messages (set up with the answering port) for that station are issued. Once all the messages for that station are complete, the connection is broken.

Both ports of the RTM need to be used. One for normal polling of slaves and the other to receive calls from slaves. Since two RTM ports are used, two Com Requests must be used. One port set up for originate mode (4) and the other port set for answer mode (5). One SCB is used with the answering port. The station ID of the slave calling in is automatically loaded into that SCB. A zero in the status word of the SCB indicates that the RTM and RTU are connected. Then the corresponding MCB(s) is executed. There should be at least one MCB for each slave that could call in.

An example program for initializing both ports for use with RBE is provided in the ladder logic examples (page 50).

NOTES

CHAPTER 4: LADDER LOGIC EXAMPLES

Basic ladder example.

4.1 Ladder Logic Tasks

There are three primary tasks to be performed by the PLC ladder logic in regards to the RTM Modbus Master Module:

a. Initialize the RTM Module
b. Read / Write Modbus/RTU Data
c. Monitor Communications Status

The first task, initializing the module, is performed once at power-up or Stop to Run transition and is never repeated. The second tasks are not performed until after the RTM is initialized, and then they are performed on a constant basis. This chapter covers in varying degrees of detail the coding of these three ladder logic tasks.

4.2 Initialization

Initialization of the RTM module is performed after a power-cycle or after a stop-to-run transition. The purpose of the initialization is to set the mode of operation of the RTM. Following successful RTM initialization, the module indefinitely operates as instructed until the next reset condition.

4.2.1 Subroutine Call

It is convenient to utilize a subroutine (or program block) in the coding of the initialization process. This subroutine is executed from the initial PLC scan until the RTMs COM_REQ instruction(s) returns a successful result. From that point, the subroutine no longer requires execution. An example ladder logic rung integrating an initialization subroutine is shown below:

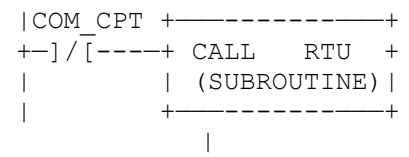


Figure 4.1 – Subroutine Example

The “RTU” subroutine shown above is executed as long as COM_CPT (a temporary internal bit) is not set. The contents of the subroutine is a series of rungs which initialize blocks of memory (usually %R) with SCB, MCB, and COM_REQ data,

4.2.2 Subroutine Contents

The contents of the subroutine data is a series of rungs which perform the following tasks:

a. Initialize the Slave Control Block (SCB) data,
b. Initialize the Message Control Block (MCB) data,
c. Initialize the COM_REQ data,
d. Execute the COM_REQ function after a 5 second delay, and
e. Monitor the COM_REQ for completion and disable the subroutine.

4.2.3 Contents of the Subroutine: Initializing the SCBs

The Slave Control Block (SCB) is a block of registers which defines the number and types of slaves with which the RTM is communicating. One SCB, 15 registers long, is required for each slave to be polled by the RTM. Therefore, for a system with 6 slaves, a block of ninety (6 x 15) contiguous registers is required for SCB1 through SCB6.

Following is an example data list for a SCB in a non-dialup application:

Table 4.1 – SCB1		
Description	**Value**	**PLC Register**
Status Indicator	0*	%R701
Failed MCB Index	0*	%R702
Number of Retries	0*	%R703
Station ID	1	%R704
Update Delay	0	%R705
Dial String	0….0	%R706 to %R715
* This indicates the initial value as set by the PLC. After initialization, the RTM module writes status information to these registers.		

Note that the PLC registers listed do not need to be located at %R701 for every application but are used in this example. For an application with six slaves, the table above would indicate just one of six total SCBs. The most straight forward way to code Slave Control Blocks (SCBs) with the Series 90-30 PLC is to utilize block move and integer move commands. The ladder logic below utilizes these functions to initialize SCB1:

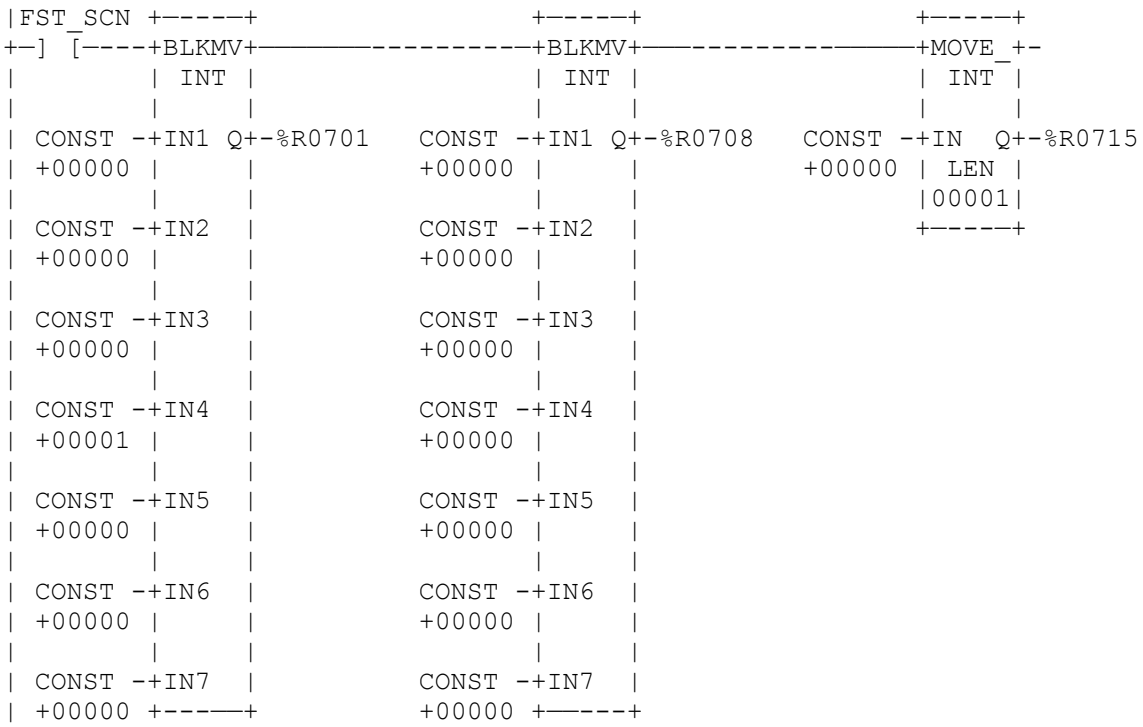


Figure 4.2 – SCB Ladder Logic (one SCB)

A rung similar to the above is required for each of the slave control blocks (SCBs). So in the example with six SCBs, six rungs similar to the above are required.

Despite the fact that the logic shown above is the most straight forward, it is not the most efficient. This inefficiency has a greater effect on program size than scan rate. The scan rate is not significantly affected simply because the rung is executed on the first scan only.

Due to the fact that a majority (all but one, in this case) of the register values in the SCB example are set to zero, a more efficient means of coding SCB1 is shown below:

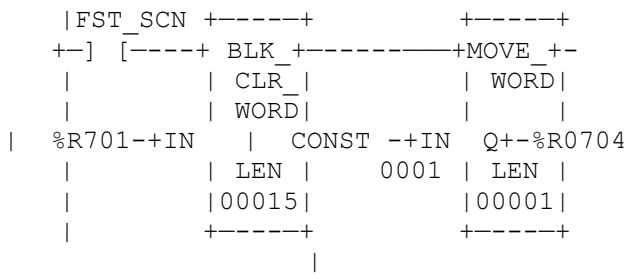


Figure 4.3 – SCB1 Ladder Logic Example

The logic above first sets registers %R701-%R715 to zero, then writes the lone non-zero value (%R704) with a move command. %R704 in the above example indicates slave ID 1.

4.2.4 Contents of the Subroutine: Initializing the MCBs

A Message Control Block (MCB) is a block of registers which defines the Modbus/RTU message commands that are to be executed by the RTM with each slave. A block of six registers is required for each MCB. If a single read command and single write command is to be executed with each of six slaves, a total register block of 72 registers (6 slaves x 6 registers x (1 read + 1 write)) must be allocated for MCBs. As with SCBs, block moves are the most straight forward coding approach for MCBs. Following is a pair of MCBs which represent a “read holding registers” command, 7 words long, and a “preset multiple registers” command, 2 words long. Both of these commands are directed at the slave with Station ID 1.

Table 4.2 – MCB Examples

Description	Value	PLC Register		Description	Value	PLC Register
Slave ID	1	%R801		Slave ID	1	%R807
RTU Command	3	%R802		RTU Command	16	%R808
RTU (slave) Ref. Offset	101	%R803		RTU (slave) Ref. Offset	95	%R809
RTU (slave) Ref. Length	7	%R804		RTU (slave) Ref. Length	2	%R810
PLC (master) Ref Type	8	%R805		PLC (master) Ref Type	8	%R811
PLC (master) Ref Offset	0	%R806		PLC (master) Ref Offset	7	%R812

Coding MCBs in ladder logic is very similar to SCBs, utilizing Block Move commands. The one thing to keep in mind is that block moves affect seven consecutive registers, and MCBs are only six registers long. One could utilize standard integer move functions, but that would require six separate integer moves functions for each MCB. It is much easier to utilize overlapping block moves for multiple MCBs as is shown in Figure 4.4.

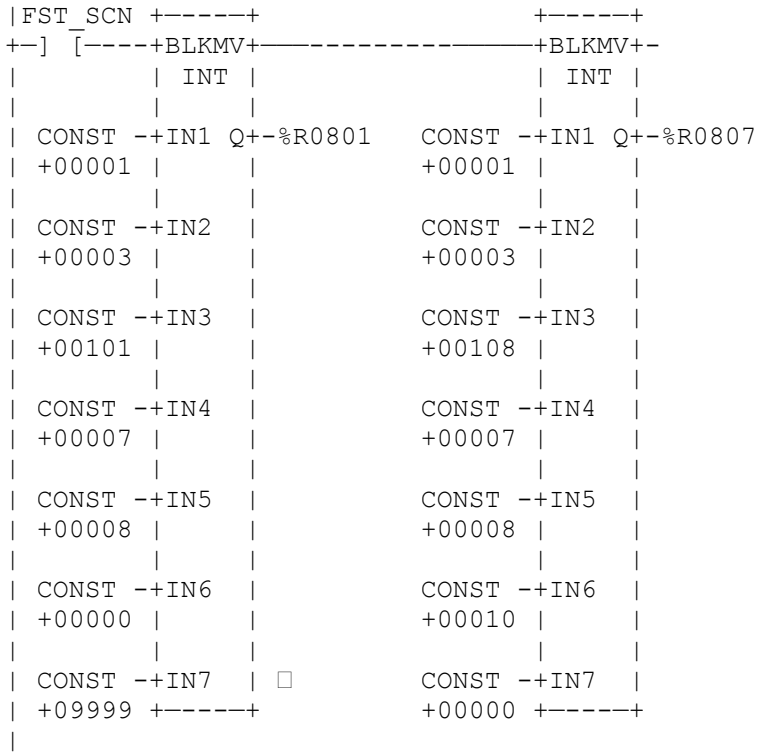


Figure 4.4 – MCB Ladder Logic Example

The last starting address of the second block move (%R807) overlaps with the last register of the first block move (for clarity, the overlapped register in the first block move is shown with a value of “9999” to make it easier to pick out). If more MCBs are to be programmed, additional block moves can be added in the same overlapping fashion, one block move per MCB. Our example stated that for a six slave system with one read command and one write command, a total of 72 registers would be allocated for MCBs. If they are coded as shown above, six block moves would be required affecting a total of 73 registers. The extra (73rd) register is affected by the seventh register of the last block move which is not overwritten by a subsequent block move.

4.2.5 Contents of the Subroutine: Initializing the COM_REQ Data Registers

The COM_REQ Data Block is a block of 17 registers which is referenced by the COM_REQ command in the initialization of the RTM module. An example COM_REQ data block is shown in Table 4.3.

Table 4.3 – COMM_REQ Data Block

PLC Register	Value	Description		PLC Register	Value	Description
%R901	11	Data Size		%R910	700	SCB Pointer Offset
%R902	0	No Wait		%R911	2	Number of MCBs
%R903	8 (type R)	Status Pointer Type		%R912	8	MCB Pointer Type
%R904	99 (puts in address 100)	Status Pointer Offset		%R913	800	MCB Pointer Offset
%R905	0	n/a		%R914	19200	Baud Rate
%R906	0	n/a		%R915	8	Port Parameter Word
%R907	2	RTM Mode		%R916	0	Modem Turnaround Time
%R908	1	Number of SCBs		%R917	0	Radio CTS Turnaround Time
%R909	8	SCB Pointer Type				

A COM_REQ Data Block is required for each active port on the RTM module. In an application which only utilizes either port 1 or port 2, only one COM_REQ data block (17 registers long) is required. If both ports are going to be utilized, then two COM_REQ data blocks (for a total of 34 registers) are required.

Like SCBs and MCBs, block moves are the simplest way to code the COM_REQ data block. Because 17 registers must be preset, three consecutive block moves (7 registers each) are required. If the block moves are not overlapped, a total of 21 registers are affected by the block moves. The following ladder logic utilizes non-overlapped block moves:

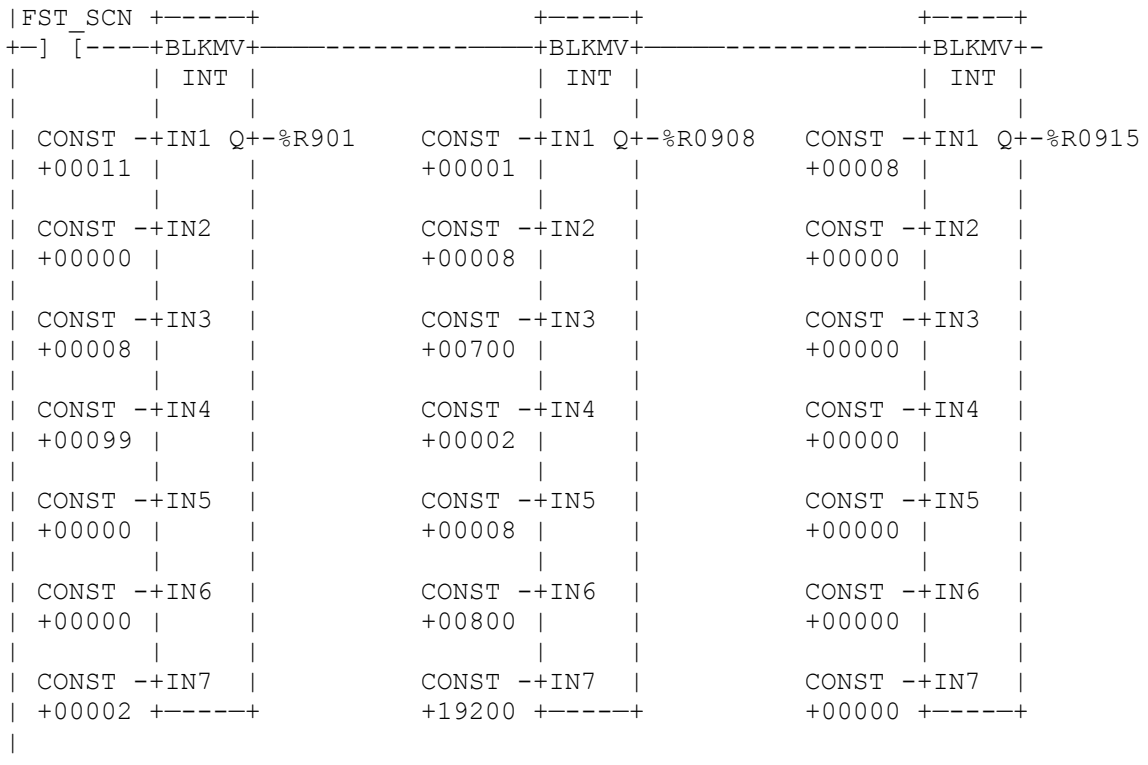


Figure 4.5 – COM_REQ Data Block Ladder Logic Example

To minimize the amount of extra registers overwritten by the block moves, they can be overlapped as shown in the following example:

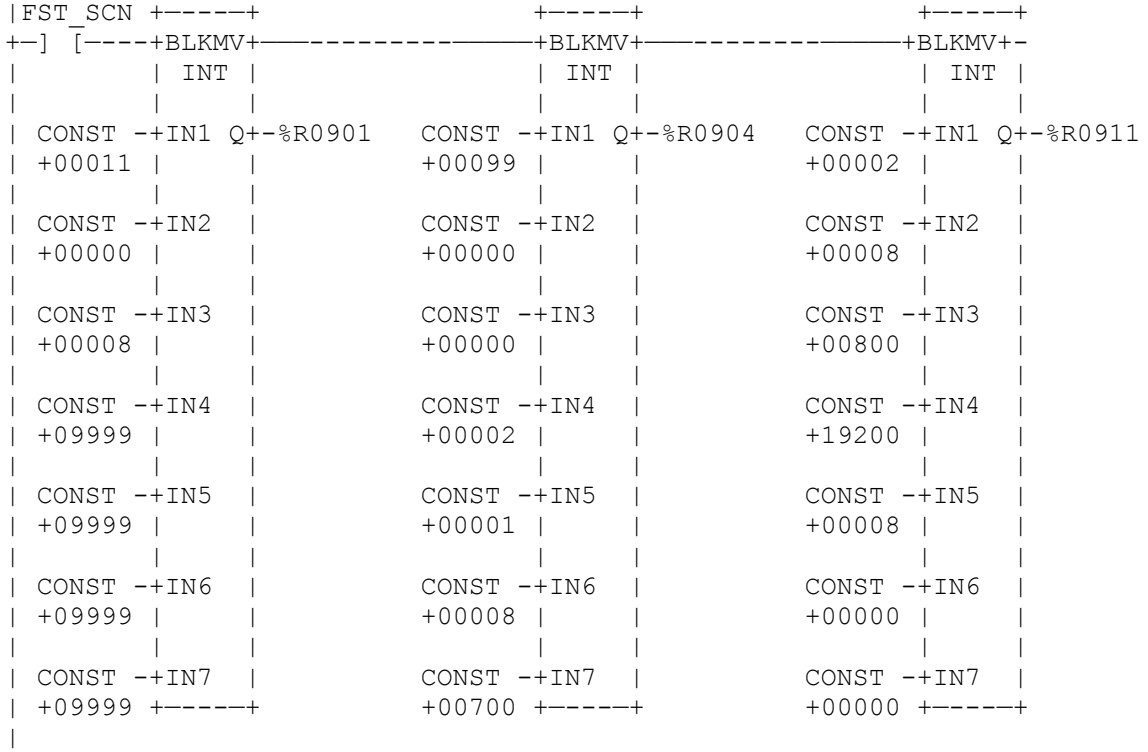


Figure 4.6 – COM_REQ Data Block Ladder Logic Example

The block moves above are overlapped such that only the desired 17 registers (in this case, %R901-%R917) are affected by the rung. The overlap occurs between the first two block moves. For clarity, the overlapped registers in the first block move are shown with values of "9999" to make them easier to pick out.

4.2.6 Contents of the Subroutine: Executing the COM_REQ

After the SCBs, MCBs, and COM_REQ data blocks have been set, the COM_REQ logic is ready for execution. As is often the case with other modules, the COM_REQ directed at the RTM must be delayed until at least 5 seconds after power-up.

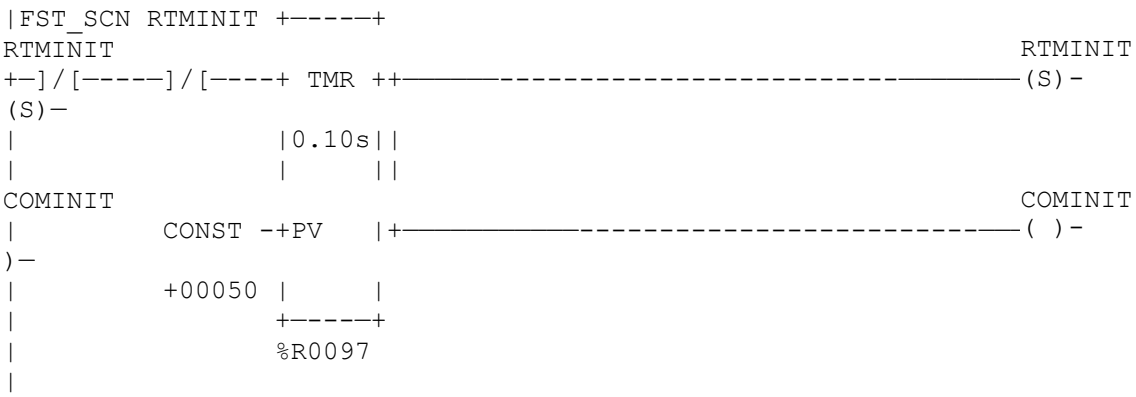


Figure 4.7 – COM_REQ Data Block Ladder Logic Example

The timer shown in Figure 4.7 is reset on the first scan. On the second scan, it starts timing. After 5 seconds, it energizes a bit called "COMINIT", which is used to trigger the COM_REQ. It also energizes a bit called "RTMINIT". On the next scan, "COMINIT" is turned off, and the RTMINIT bit causes the timer to be disabled indefinitely. Temporary (%T) data types are used for these two bits, so that their state is not retained through a reset.

After the COM_REQ has been executed, it writes status data to the status register specified in the COM_REQ data block. This register is monitored by the ladder logic to determine the successful (or unsuccesful) result of the COM_REQ. It is good practice to clear the status register just prior to execution of the COM_REQ, ensuring the validity of future data. The following rung shows the execution of the COM_REQ:

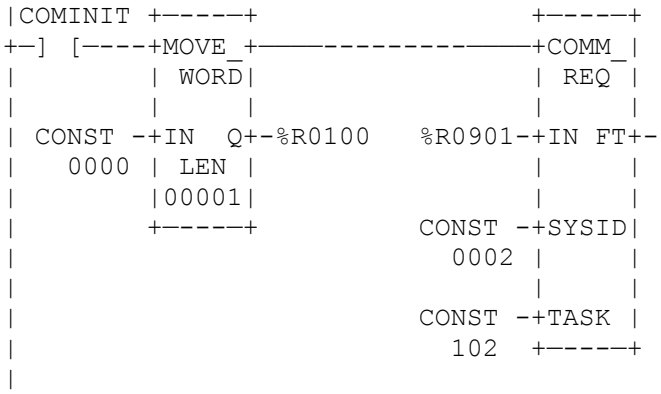


Figure 4.8 – COM_REQ Execution Ladder Logic Example

Notice that the “COMINIT” bit set by the timer in the previous rung triggers the rung. It is important to note that this bit (“COMINIT”) is on for one scan only. The COM_REQ status register (in this case %R100) is cleared just prior to execution of the COM_REQ. The COM_REQ is then executed, using the data block that starts with %R901, with a SYSID of 0002H, and a TASK of 0066H. The SYSID indicates the RTM module is installed in Rack 0, Slot 2. The TASK of 0066H (102 decimal) indicates that the COM_REQ is directed at Port 2 of the RTM module.

4.2.7 Contents of the Subroutine: Monitoring the COM_REQ

Now that the COM_REQ has been executed, the ladder logic monitors the COM_REQ status register. When that register reaches a non-zero value, the COM_REQ is complete and the entire subroutine no longer requires execution. Now COM_CPT is set which disables the subroutine. (Fig. 4.1)

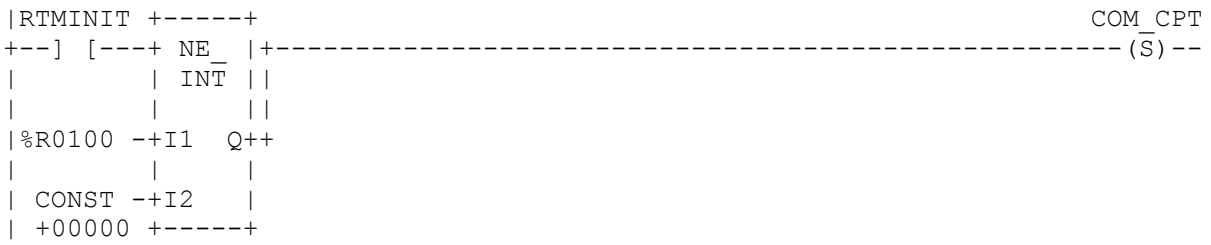


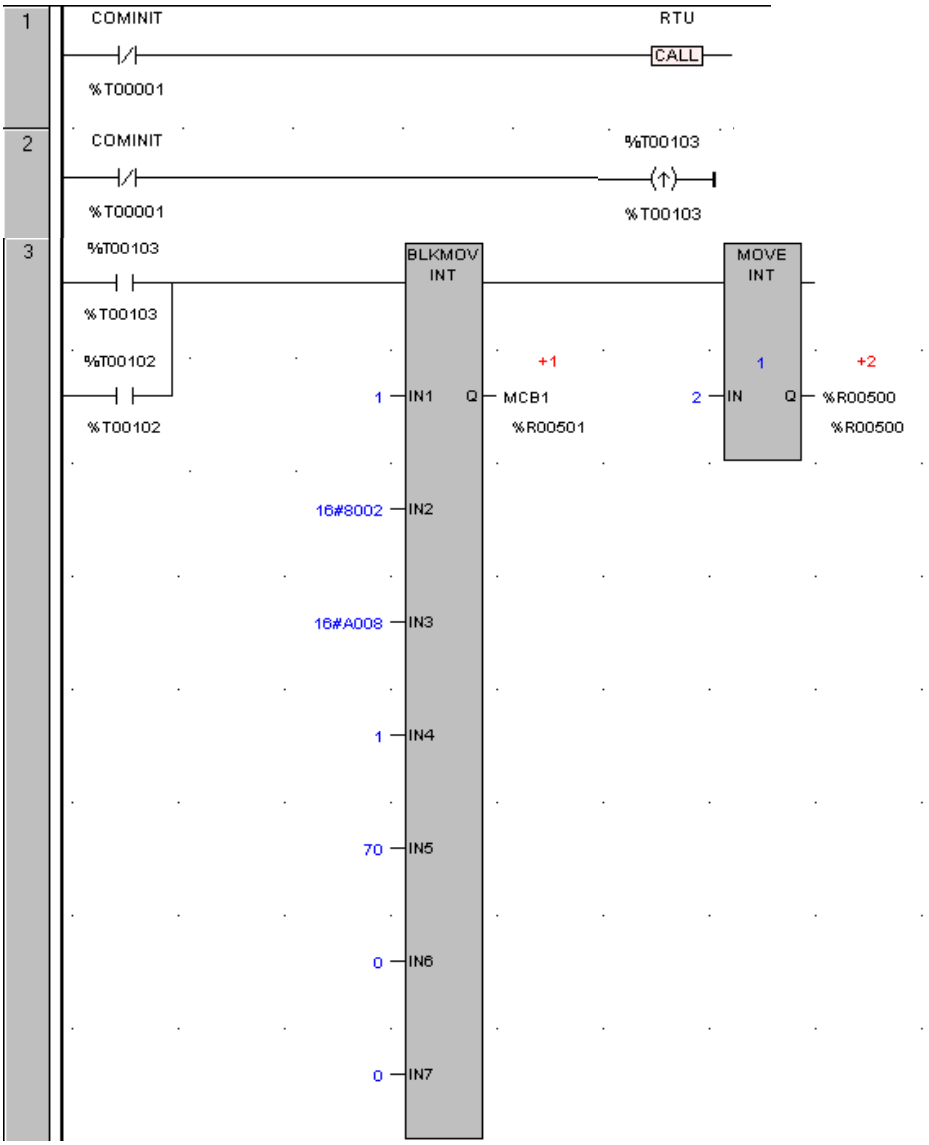
Figure 4.9 - COM_REQ Execution Ladder Logic Example

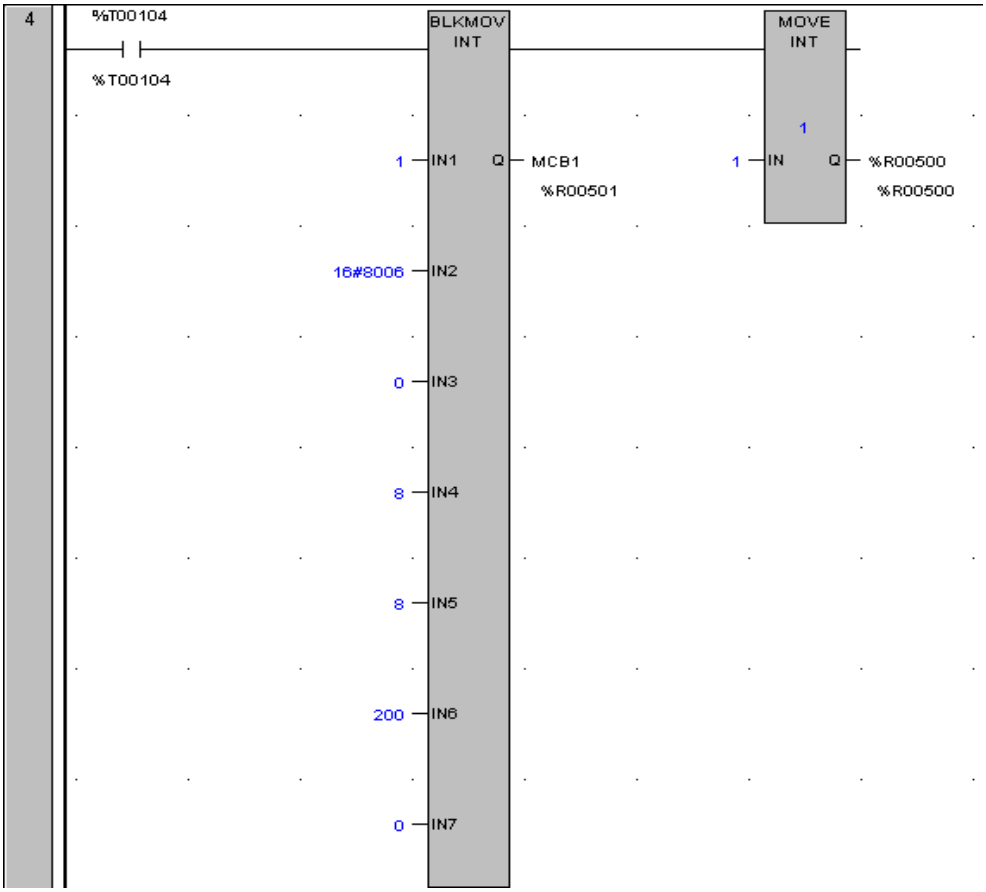
Ladder logic example utilizing the Interactive bit and two one shot MCBs

The following example has two sections, the Main Program and the RTU subroutine. The MCBs are located in the main program instead of the subroutine so that they can be executed on the fly. There is a time delay set up so that the first MCB (rung 3) executes, then five seconds later switches to the second MCB (rung 4). The execution of the MCBs continue indefinitely.

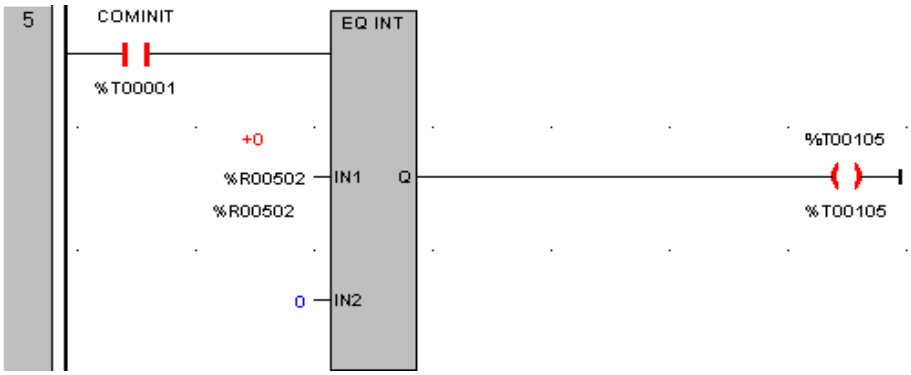
Main program.

Rung 3. "16#8002" is placed in the command word. The 8 in 8002 sets bit 16 high which makes the MCB a one shot MCB. After a One Shot MCB is executed, it sets the command word (IN2) to zero.

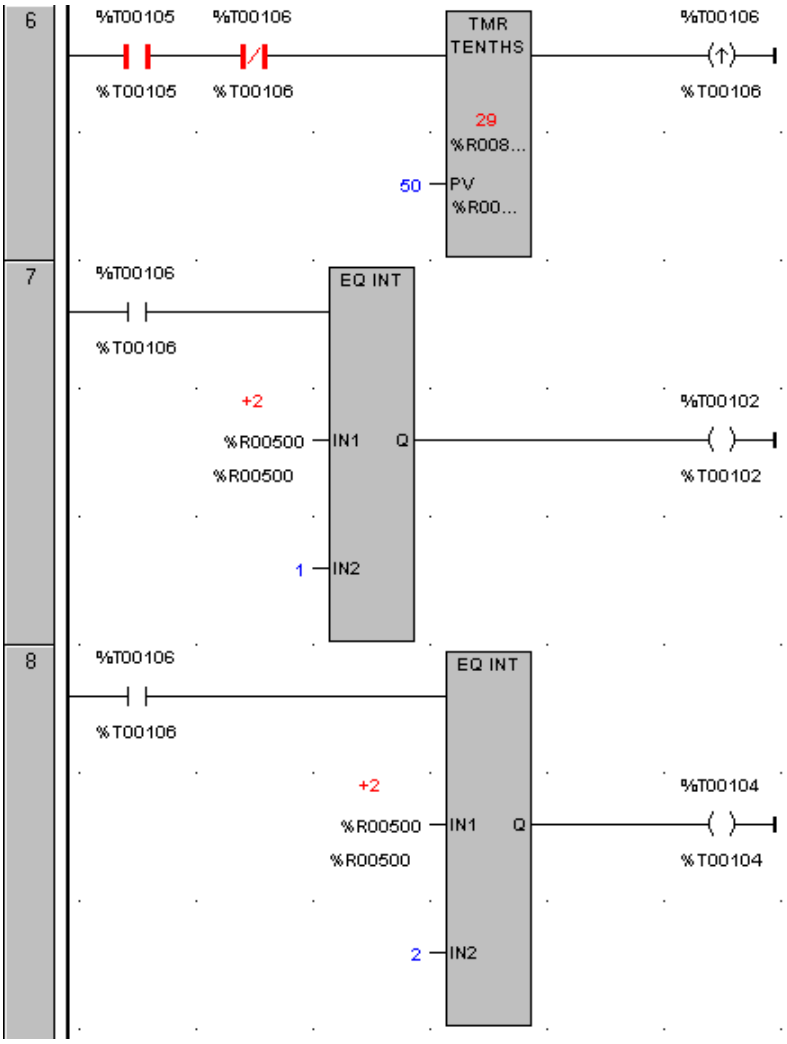




Rung 5 checks for a zero in %R502 which is the command word of the MCBs.



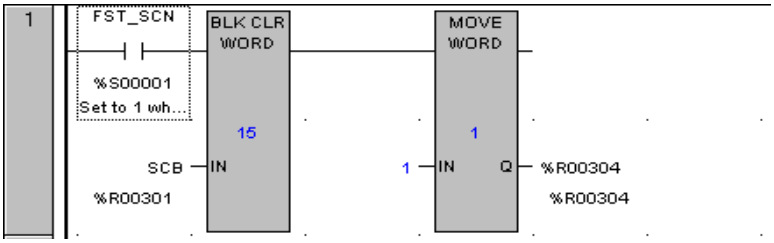
Rung 6 creates a five second delay in execution of MCBs. Rung 7 and 8 decide which MCB is executed.



RTU Subroutine.

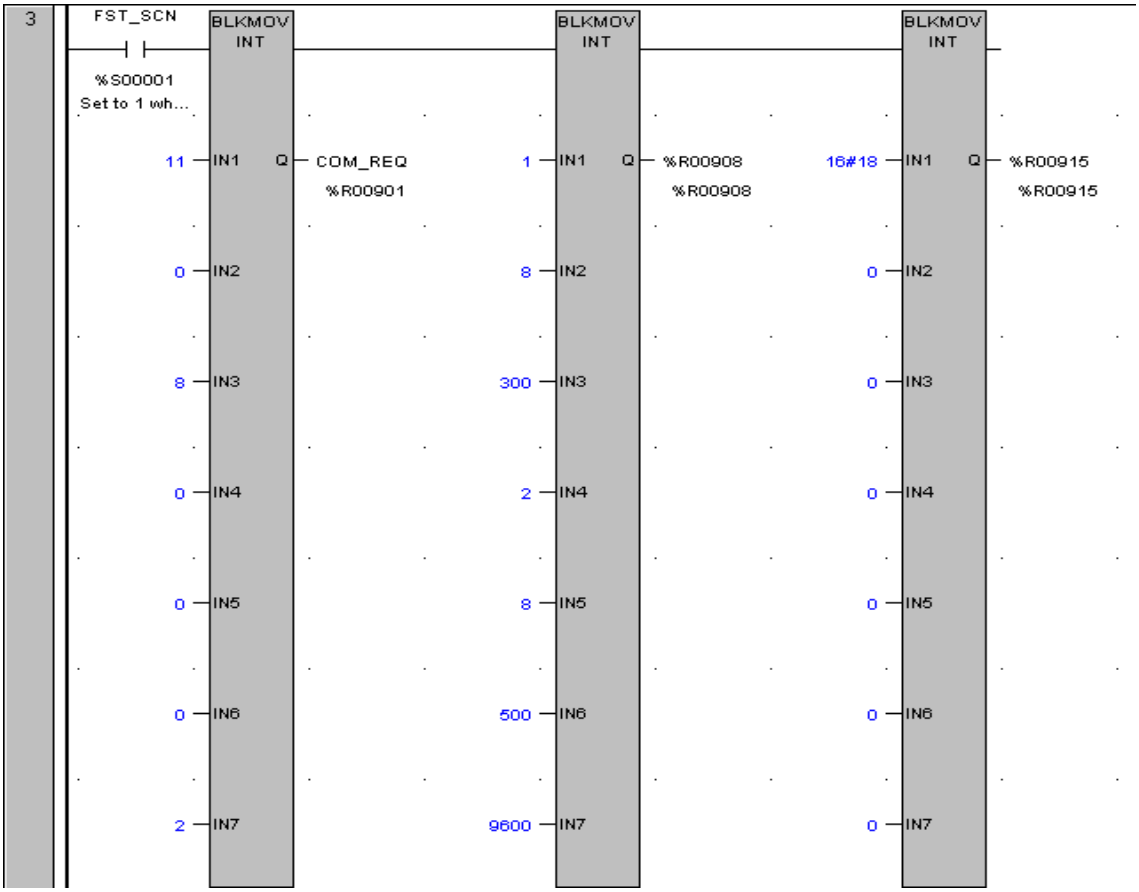
The RTU subroutine is basically the same as the first example except that the MCBs are located outside of the subroutine. Any combination of MCBs could still be used here, but were not necessary for this example.

Rung 1. SCB



Rung 3. Com Request setup.

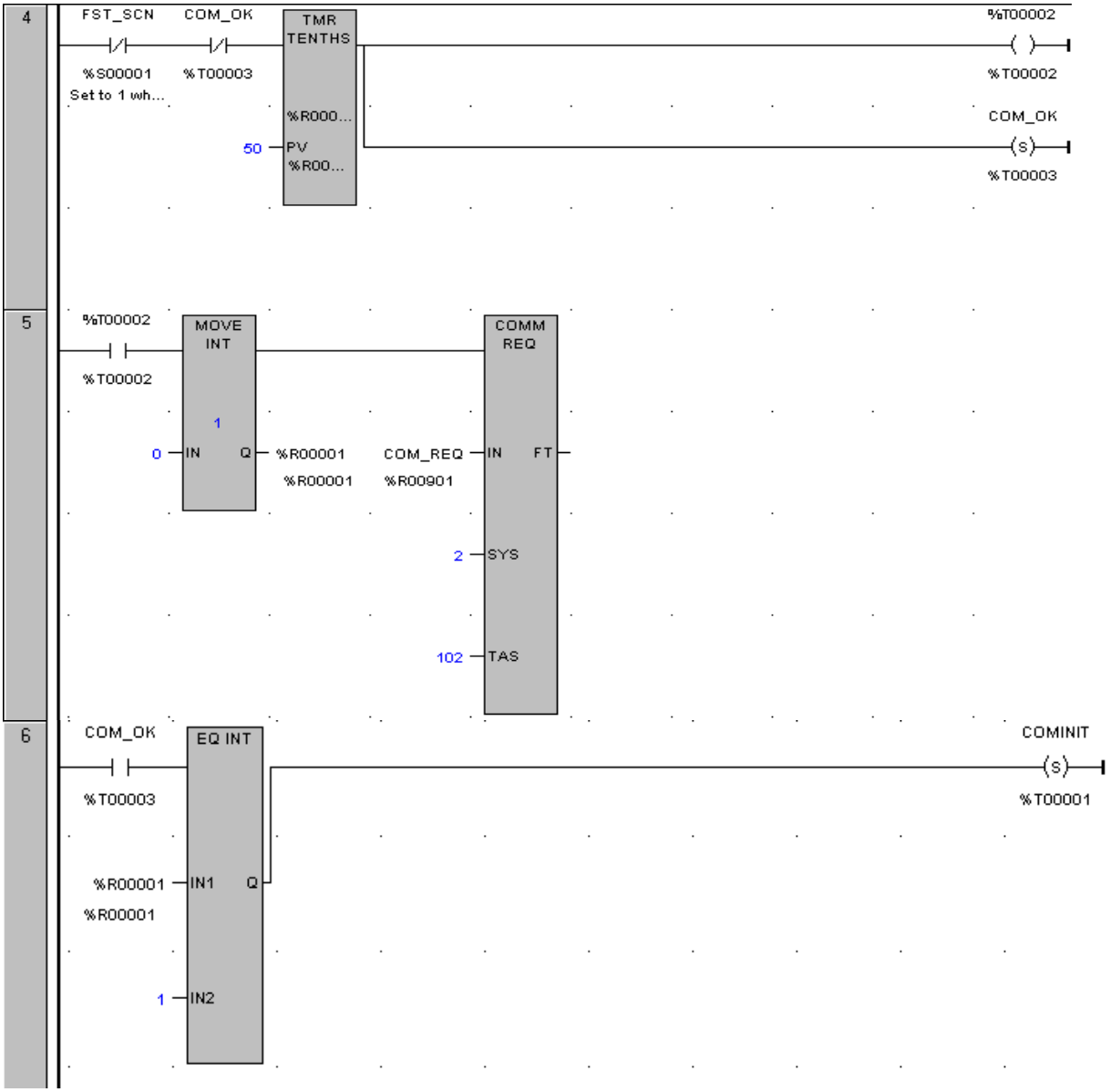
The Interactive bit must be set for this program to work correctly. The Interactive bit is set with bit 5 in the Port Parameter Word. (Refer to Fig 3.3 [page 28]; %R915 is in this example.)



Rung 4. After the SCBs, MCBs, and COM_REQ data blocks have been set, the COM_REQ logic is ready for execution. As is often the case with other modules, the COM_REQ directed at the RTM must be delayed until at least 5 seconds after power-up.

Rung 5. After the COM_REQ has been executed, it writes status data to the status register specified in the COM_REQ data block. This register is monitored by the ladder logic to determine the successful (or unsuccesful) result of the COM_REQ. It is good practice to clear the status register just prior to execution of the COM_REQ, ensuring the validity of future data. The COM_REQ shows the RTM is in slot 2 of the rack and port2 of the RTM is being used.

Rung 6. If COM_REQ status word %R1 equals 1 then end subroutine.

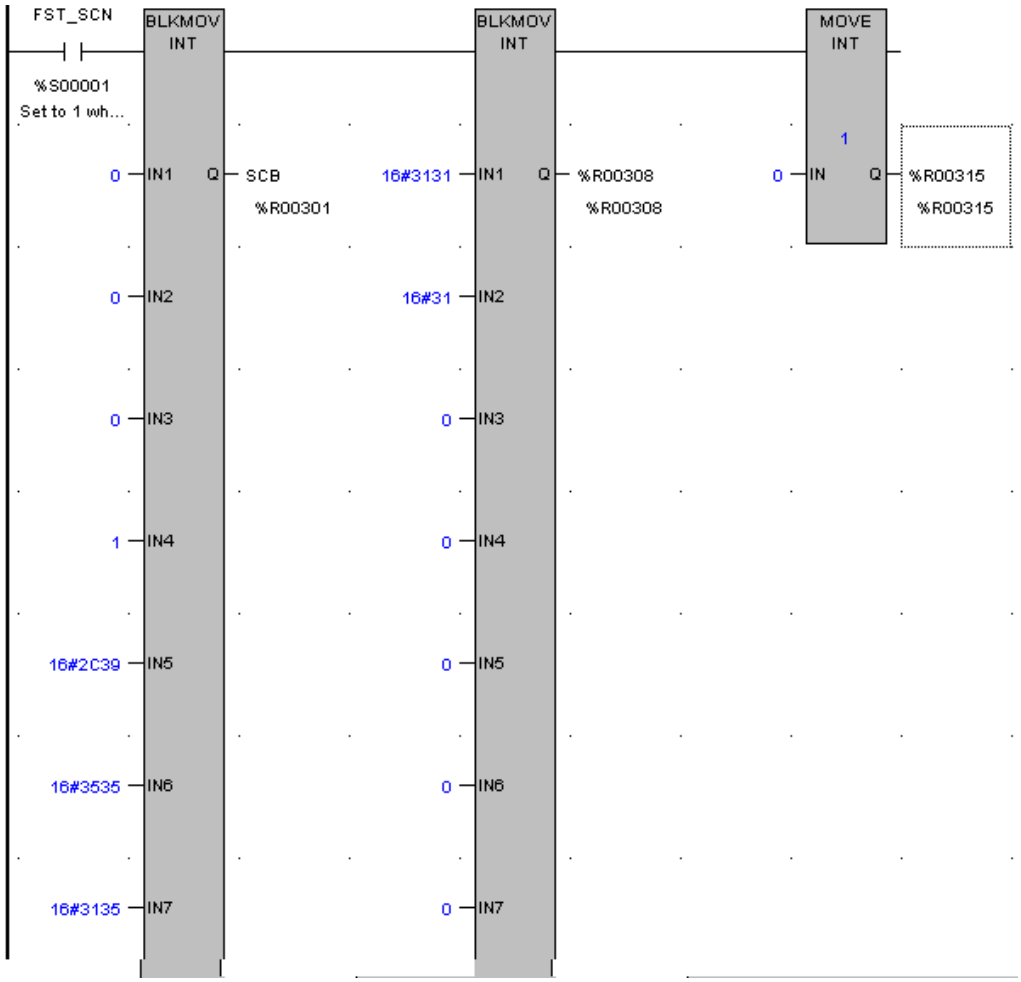


Ladder logic example utilizing Report by Exception

The following example is written for three slaves. The first set of SCBs, MCBs and Com_Req is for port 2 set to originate mode (4). The second set of SCB, MCBs, and Com_Req is for port 1 set to answer mode (5). The SCB bellow dials out to slave ID 1. The example number is 9,555-1111.
The numbers are entered as ASC values converted to hex.
To get the hex value of a number simply add 30.

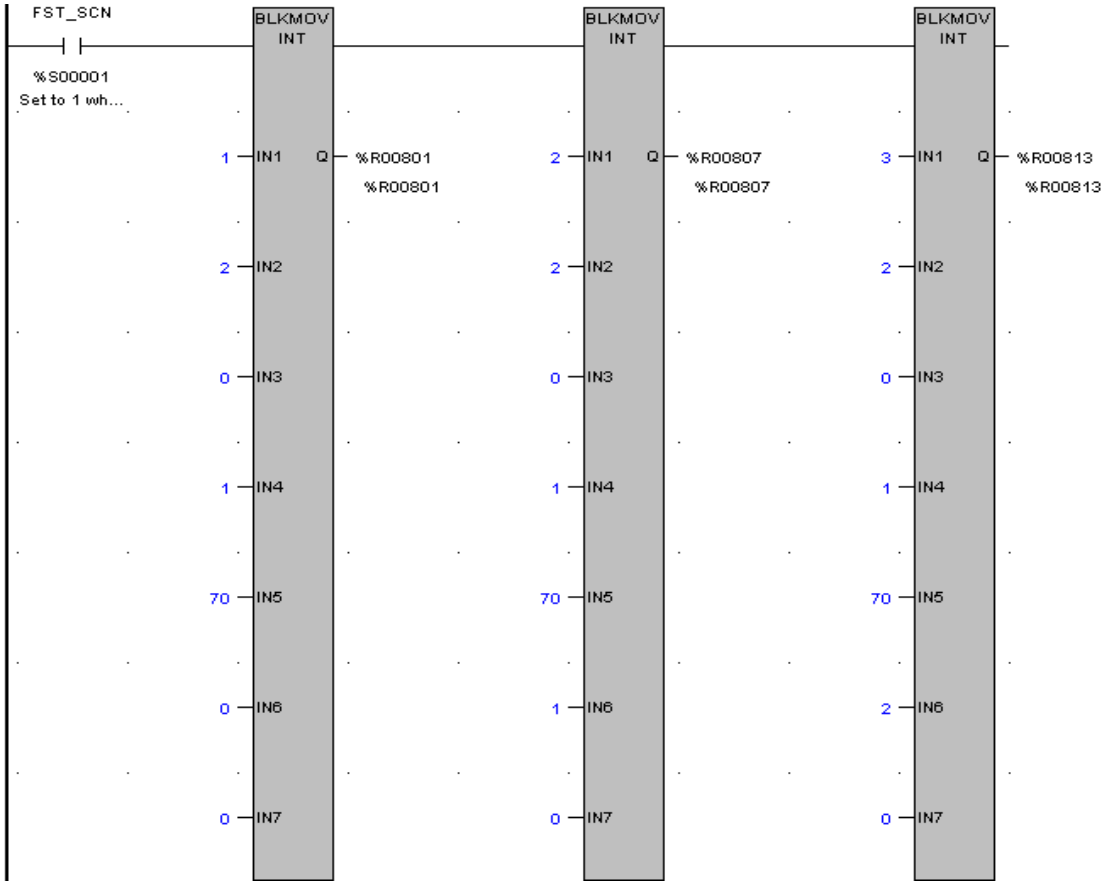
Where 9 = 39 hex
, = 2C
5 = 35
1 = 31

SCB 1

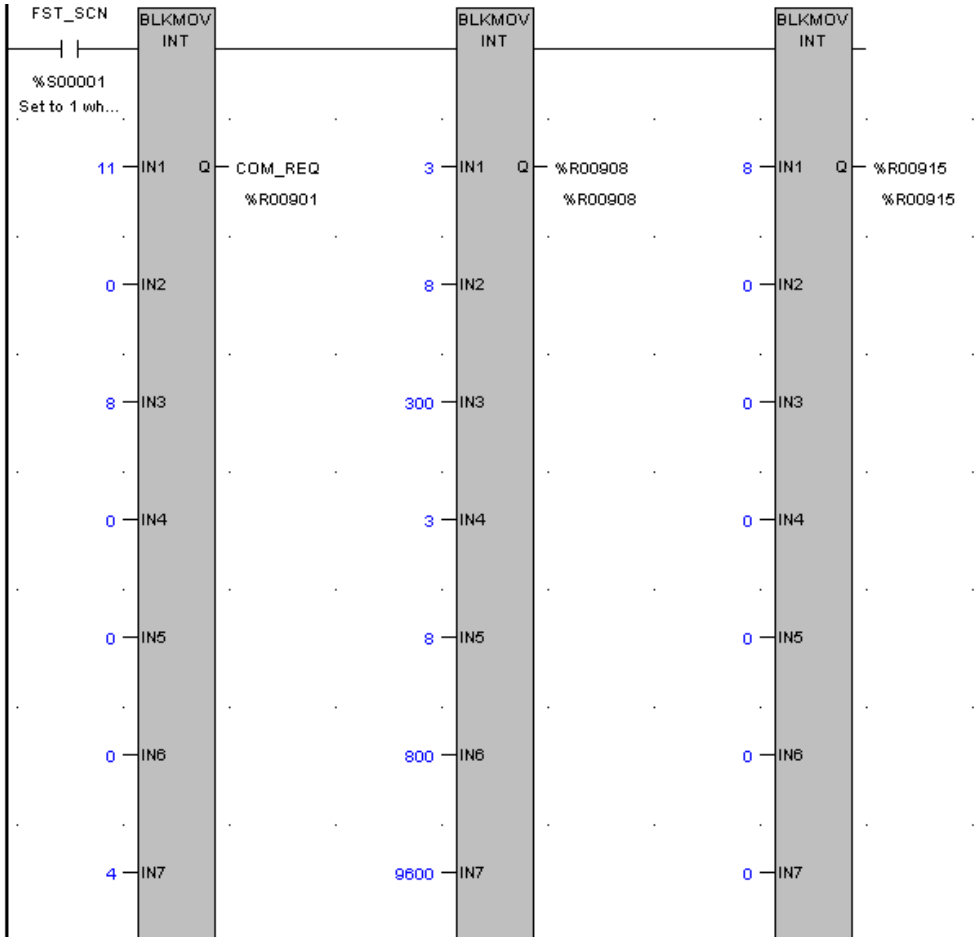


SCB 2 and SCB 3 for Port 2 are not shown. SCB 2 should start at %R316 and SCB 3 at %R331.

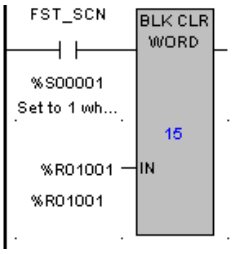
Port 2 MCBs for slaves 1,2, and 3. The MCBs read the first input value of the slaves and put the data in %I1, 2, and 3 respectively.



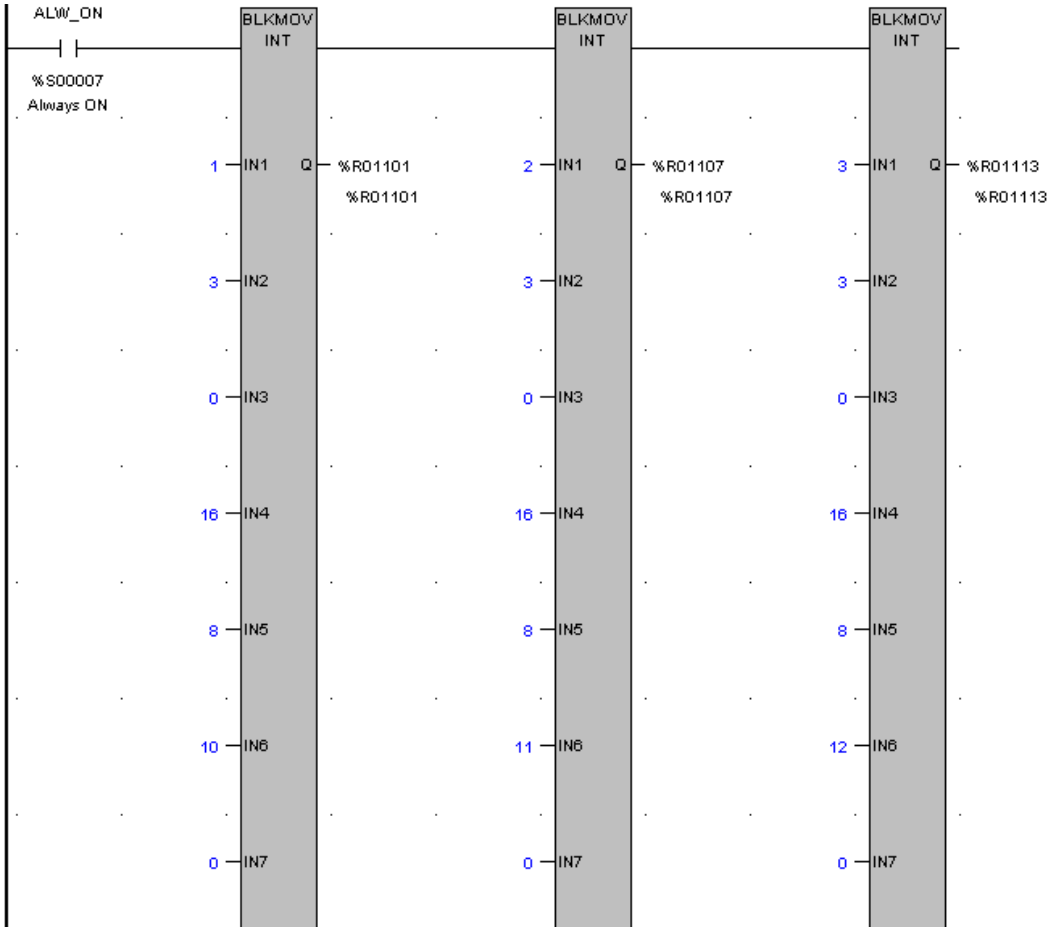
Port 2 Com_Req data (originate mode (4))



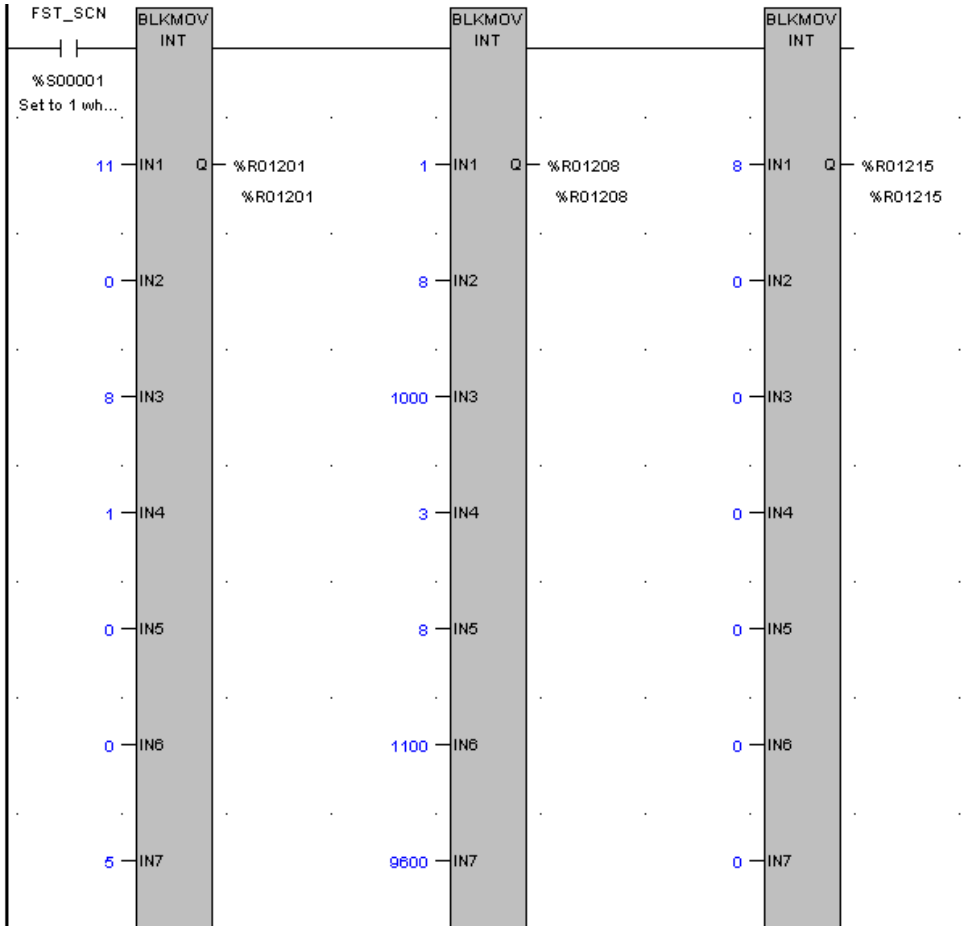
Port 1 SCB. Slave ID is automatically placed in the SCB by the Calling slave.



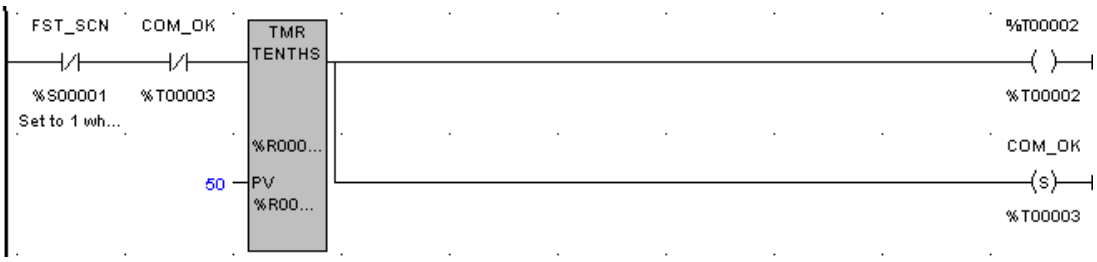
Port 1 MCBs. The correct MCB is executed based on the station ID that is placed in the Port 1 SCB. After the MCB has been executed, the connection between RTM and slave is broken. The MCBs read the first holding register of the slaves and put the data in %R11, 12, and 13 respectively.



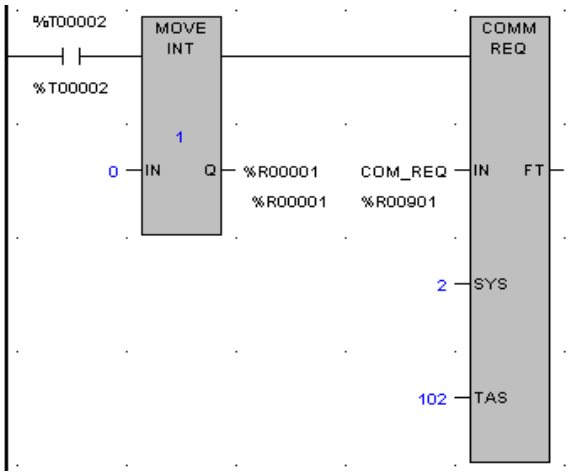
Port 1 Com_Req data (answer mode (5)).



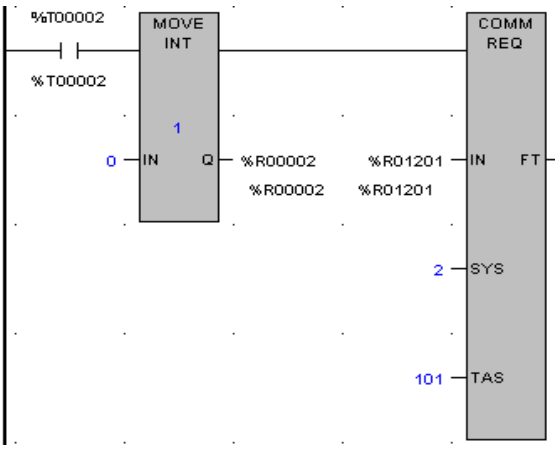
The rung below assures that the RTM has time to initialize.



Com_Req for Port 2. Opens Port 2 for communication.

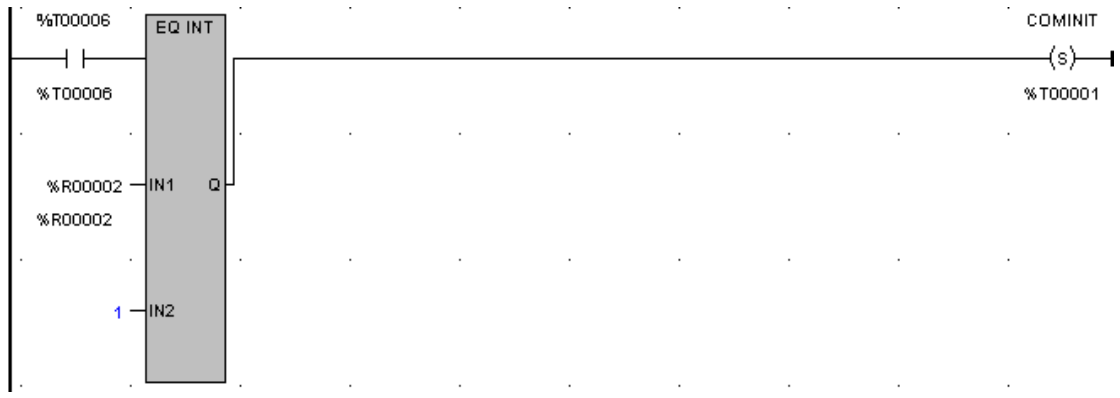


Com_Req for Port 1. Opens Port 1 for communication.

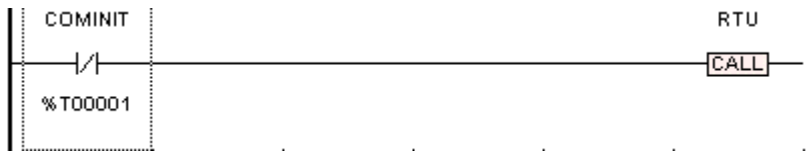


This rung ensures that Port 2 initialized successfully. %R1 is the status register for Port 2. A one in %R1 indicates that the port is ready and without errors.

This rung ensures that Port 1 initialized successfully. %R2 is the status register for Port 1. A one in %R2 indicates that the port is ready and without errors. If %R2 = 1 then %T1 turns on Which ends the subroutine.



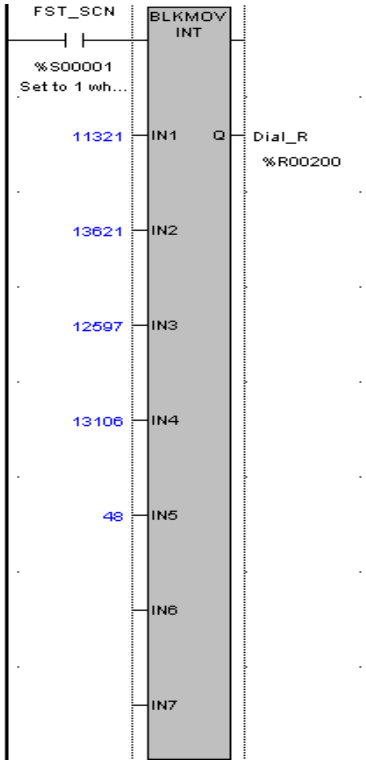
Subroutine call. When %T1 is high, the subroutine is terminated.



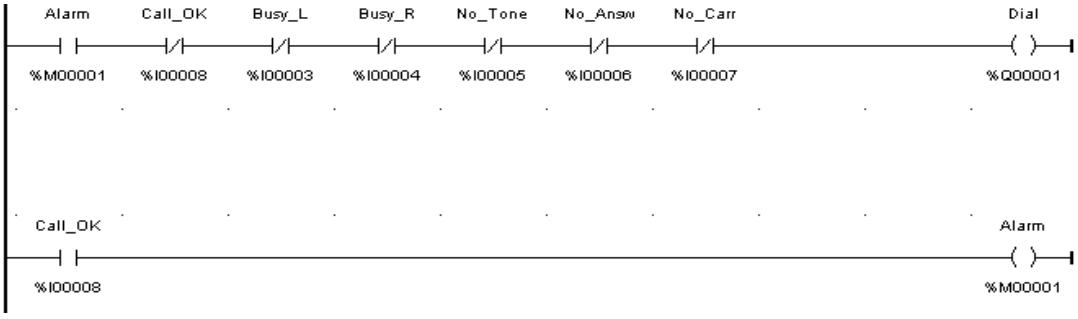
Report by Exception example using a HE693RTU940.

This example is for an HE693RTU940 modbus slave. The rung below sets the telephone number for the slave to dial in an alarm (Report by Exception) condition. The telephone number dialed represents the number of the answer port for the RTM master (Port 1 in the above example). The number dialed is 9,555-1230. The block move shows the number in decimal format. The hex values are as follows:

%R200	16#2C39
%R201	16#3535
%R202	16#3135
%R203	16#3332
%R204	16#0030



The next two rungs initiate the call and then reset the alarm.



NOTES

INDEX