

Micro850 Modbus TCP to MicroLogix 1400

This guide will show you how to set a Micro850 as a Modbus TCP Master in order to talk to a ML1400 and share data

Hardware (Used for this guide):

- Micro850 controller (FRN 4.11 or greater required).
- ML1400 Series B (Only ML1400 units shipping with Series B FRN, shipping since late 2010, will support Modbus TCP).

Software (Used for this guide):

- Connected Components Workbench (4.00 or greater)
- RSLogix 500 version 8.4 (version 8.3 or greater required)

Reference:

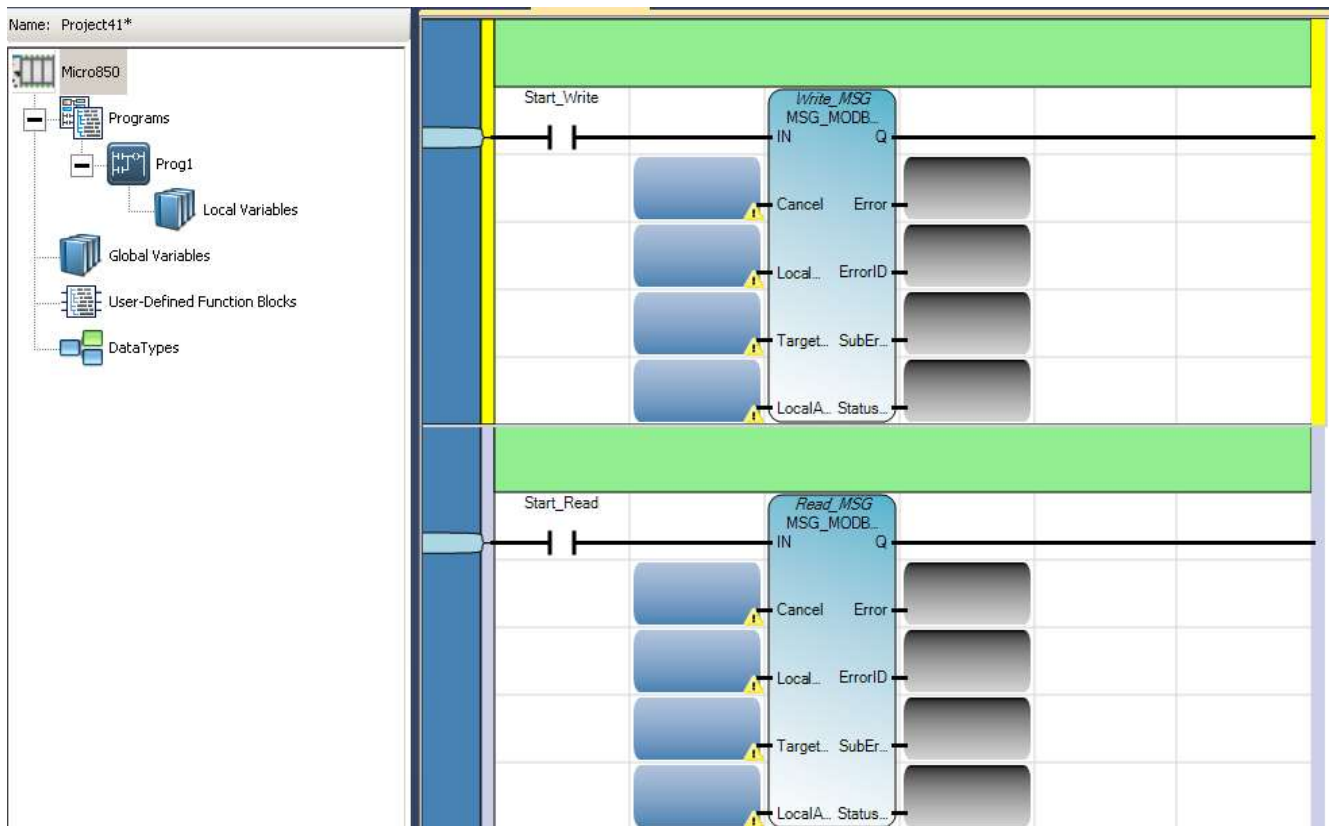
- Getting Started with CIP Client Messaging
http://literature.rockwellautomation.com/idc/groups/literature/documents/gs/2080-gs002_-en-e.pdf
- KnowledgeBase Article 20543
https://rockwellautomation.custhelp.com/app/answers/detail/a_id/20543
- Micro800 reference Manual
http://literature.rockwellautomation.com/idc/groups/literature/documents/rm/2080-rm001_-en-e.pdf

In this guide, we will set the Micro850 as a Modbus TCP Master to talk to a ML1400 Modbus TCP slave.

Micro850 Modbus TCP Master

We will create a program in CCW that will write to a ML1400 and read data from it.

- 1) Start a new project in CCW and in the Program Ladder, place a Direct Contact (normally open contact) and assign a Bool Variable for it. Following the contact, place a MSG_MODBUS2 function block, and do the same on a second rung. We will call the MSG_Modbus2 in rung one "Write_MSG" and the MSG_MODBUS2 in rung two "Read_MSG". The program should now look as follows:



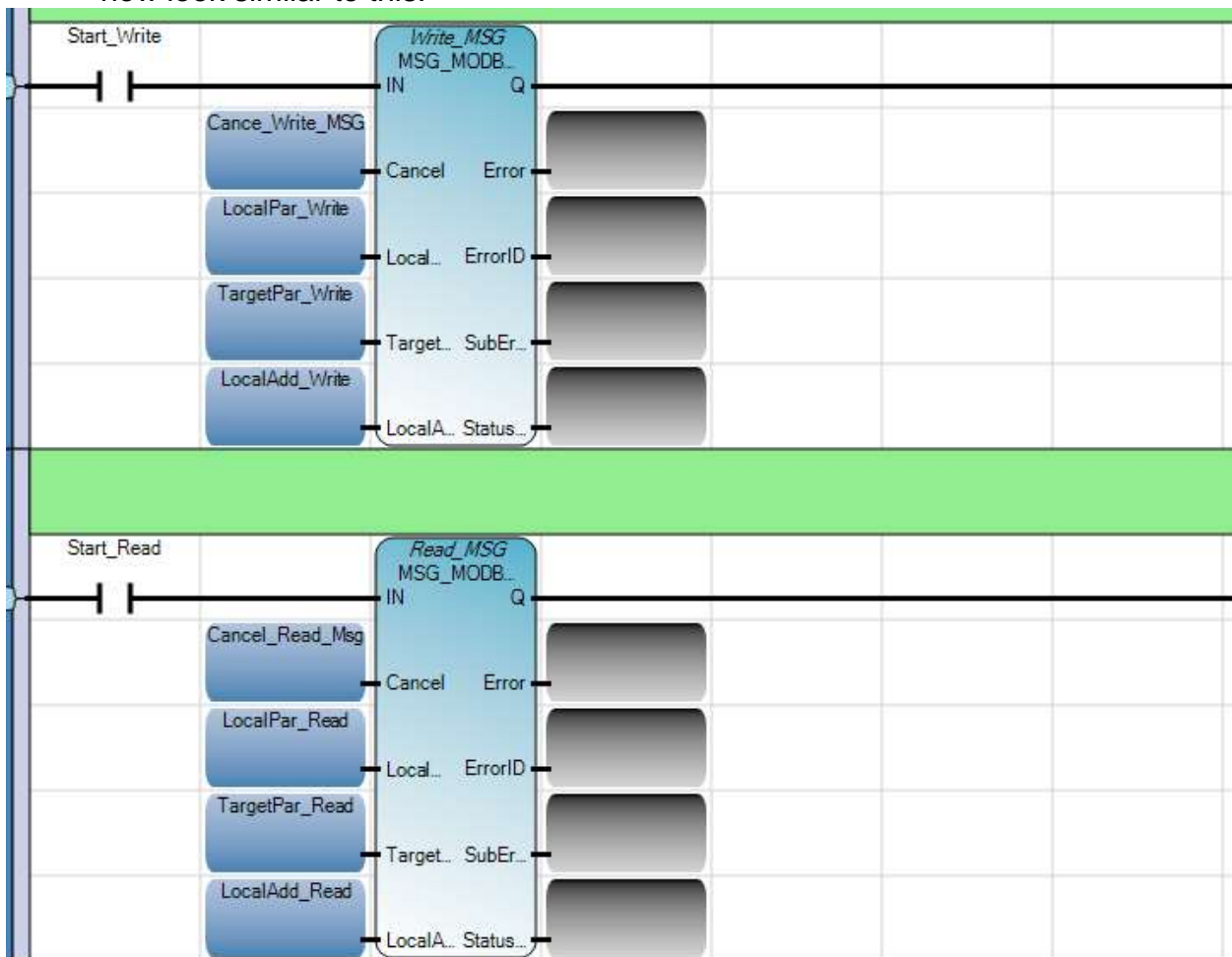
- 2) Create the variables (Local or Global) that will be assigned to the MSG_MODBUS2 sub-blocks. Keep in mind the each MSG block will need its own individual variable.
So for the Write MSG, we will create the following variables for this guide:

Name	Data Type	Dimension	Initial Valu	Alias	Attribute
Cancel_Write_MSG	BOOL				Read/Write
+ LocalPar_Write	MODBUS2LOCPARA		...		Read/Write
+ TargetPar_Write	MODBUS2TARPARA		...		Read/Write
+ LocalAdd_Write	MODBUSLOCADDR		...		Read/Write
Start_Write	BOOL				Read/Write

And for the Read MSG, we will create the following variables:

Cancel_Read_Msg	BOOL				Read/Write
+ LocalPar_Read	MODBUS2LOCPARA		...		Read/Write
+ TargetPar_Read	MODBUS2TARPARA		...		Read/Write
+ LocalAdd_Read	MODBUSLOCADDR		...		Read/Write
Start_Read	BOOL				Read/Write

3) Assign the variables to the specific MSG_Modbus sub-blocks. The ladder should now look similar to this:



4) Next, we will configure the variables we created in order to be able to read/write to the specified ML1400.

- We will start with the **LocalPar_Write** and **LocalPar_Read**:

Name	Data Type	Initial Value	Comment
Cance_Write_MSG	BOOL		
LocalPar_Write	MODBUS2LOCP#	...	
LocalPar_Write.Channel	UINT	4	Local Channel number
LocalPar_Write.TriggerType	UDINT	0	0 = Trigger once, n = Cyclic Trigger
LocalPar_Write.Cmd	USINT	16	Modbus command
LocalPar_Write.ElementCnt	UINT	2	No. of elements to Read/Write
LocalPar_Read	MODBUS2LOCP#	...	
LocalPar_Read.Channel	UINT	4	Local Channel number
LocalPar_Read.TriggerType	UDINT	0	0 = Trigger once, n = Cyclic Trigger
LocalPar_Read.Cmd	USINT	3	Modbus command
LocalPar_Read.ElementCnt	UINT	2	No. of elements to Read/Write

Parameter	Data type	Description
Channel	UINT	Local Ethernet port number: <ul style="list-style-type: none"> • 4 for Micro850 & Micro820 embedded Ethernet port
TriggerType	UDINT	Message trigger type: <ul style="list-style-type: none"> • 0: Msg Triggered Once (when IN goes from False to True) • 1 to 65535 - Cyclic trigger value in milliseconds. Message triggered periodically when IN is true and the previous message execution completes. • Set the value to 1 to trigger messages as quickly as possible.
Cmd	USINT	Modbus command: <ul style="list-style-type: none"> • 01: Read Coil Status (0xxxx) • 02: Read Input Status (1xxxx) • 03: Read Holding Registers (4xxxx) • 04: Read Input Registers (3xxxx) • 05: Write Single Coil (0xxxx) • 06: Write Single Register (4xxxx) • 15: Write Multiple Coils (0xxxx) • 16: Write Multiple Registers (4xxxx) • Others: See MODBUS2LOCPARA custom command support
ElementCnt	UINT	Limits <ul style="list-style-type: none"> • For Read Coil/Discrete inputs: 2000 bits • For Read Register: 125 words • For Write Coil: 1968 bits • For Write Register: 123 words

- Then the **TargetPar_Write** and **TaretPar_Read**:

	Name	Data Type	Initial Value	Comment
	TargetPar_Write	MODBUS2TARPARA	...	
	TargetPar_Write.Addr	UDINT	1	Target's Modbus data address
	TargetPar_Write.NodeAddress	MODBUS2NODEADDR	...	Target node address
	TargetPar_Write.NodeAddress[0]	USINT	130	
	TargetPar_Write.NodeAddress[1]	USINT	88	
	TargetPar_Write.NodeAddress[2]	USINT	214	
	TargetPar_Write.NodeAddress[3]	USINT	8	
	TargetPar_Write.Port	UINT	502	Target TCP port number
	TargetPar_Write.UnitId	USINT	255	Unit Identifier
	TargetPar_Write.MsgTimeout	UDINT	3000	Message time out (in milliseconds)
	TargetPar_Write.ConnTimeout	UDINT	3000	Connection timeout (in milliseconds)
	TargetPar_Write.ConnClose	BOOL	FALSE	Connection closing behavior
	TargetPar_Read	MODBUS2TARPARA	...	
	TargetPar_Read.Addr	UDINT	1	Target's Modbus data address
	TargetPar_Read.NodeAddress	MODBUS2NODEADDR	...	Target node address
	TargetPar_Read.NodeAddress[0]	USINT	130	
	TargetPar_Read.NodeAddress[1]	USINT	88	
	TargetPar_Read.NodeAddress[2]	USINT	214	
	TargetPar_Read.NodeAddress[3]	USINT	8	
	TargetPar_Read.Port	UINT	502	Target TCP port number
	TargetPar_Read.UnitId	USINT	255	Unit Identifier
	TargetPar_Read.MsgTimeout	UDINT	3000	Message time out (in milliseconds)
	TargetPar_Read.ConnTimeout	UDINT	3000	Connection timeout (in milliseconds)
	TargetPar_Read.ConnClose	BOOL	FALSE	Connection closing behavior

*****Note: Target's Modbus Data Address is the beginning address you want to read/write to. For example: If I want to start writing to address 40001, then the Target's Modbus data address is 1. *****

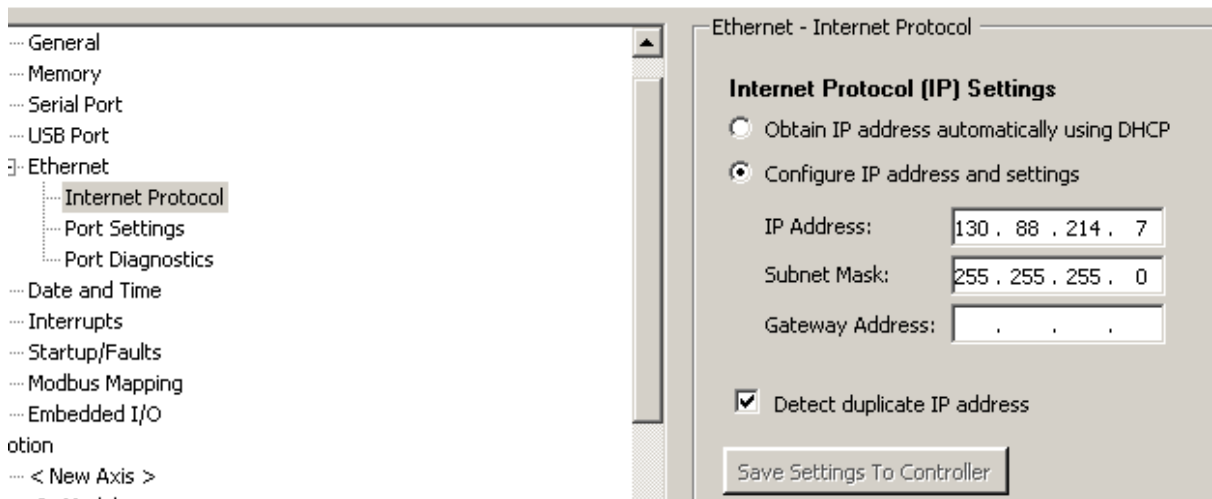
Parameter	Data type	Description
Addr	UDINT	Target device's Modbus data address: <ul style="list-style-type: none"> • 1 - 65536. • Decreases by one when sending. • Firmware uses low-word of address if the address value is greater than 65536.
NodeAddress[4]	USINT	Target device's IP address. The IP address should be a valid unicast address and cannot be 0, multicast, broadcast, local address or loop back address (127.x.x.x). For example, to specify 192.168.2.100: <ul style="list-style-type: none"> • NodeAddress[0]= 192 • NodeAddress[1]= 168 • NodeAddress[2]= 2 • NodeAddress[3]= 100
Port	UINT	Target TCP port number. Standard Modbus/TCP port is 502. 1 - 65535 Set to 0 to use the default value 502
UnitId	USINT	Unit Identifier. Used to communicate with slave devices through a Modbus bridge. Refer Modbus specification for more details. Note that Mikro800 shall not attempt to validate this value. 0 - 255 Set to 255 if Target device is not a bridge.
MsgTimeOut	UDINT	Message timeout (in milliseconds). Amount of time to wait for a reply for an initiated command. <ul style="list-style-type: none"> • 250 - 10,000 • Set to 0 to use the default value 3000. • A value less than 250 (minimum) will be set to 250. • A value greater than 10,000 (maximum) will be set to 10,000. See also Modbus/TCP message timeout timers (on page 213).
ConnTimeOut	UDINT	TCP Connection establishment timeout (in milliseconds). Amount of time to wait for establishing successful TCP connection to the Target device. <ul style="list-style-type: none"> • 250 - 10,000 • Set to 0 to use the default value 3000. • A value less than 250 (minimum) will be set to 250. • A value greater than 10,000 (maximum) will be set to 10,000. See also Modbus/TCP message timeout timers (on page 213).
ConnClose	BOOL	TCP connection closing behavior. <ul style="list-style-type: none"> • True: Close the TCP connection upon message completion. • False: Do not close the TCP connection upon message completion [Default]. See also Modbus/TCP message connections (on page 213).

- And finally, we will give the data value we want to write to the ML1400 under **LocalAdd_Write**:

Name	Data Type	Initial Value
LocalAdd_Write	MODBUSLOCADDR	...
LocalAdd_Write[1]	WORD	12345
LocalAdd_Write[2]	WORD	6789

LocalAddr	Input	MODBUSLOCADDR	MODBUSLOCADDR data type is a 125 Word array that is used by Read commands to store the data (1-125 words) returned by the Modbus slave and by Write commands to buffer the data (1-125 words) to be sent to the Modbus slave.
-----------	-------	---------------	---

- Go to the Internet Protocol Settings under the General Micro850 settings window and assign the Micro850 an IP address within the same network as the ML1400. In This sample we assigned the Micro850 an IP address of 130.88.214.7.



- Build, Save, then Download the program into the Micro850
- Open RSLogix500 and create a new project for the ML1400 controller
- Go to Channel 1 in Channel Configuration and set the IP address of the controller (in this sample it is 130.88.214.8). Then check the checkbox for Modbus TCP Enable in order to allow the ML1400 to accept Modbus TCP protocol.

Channel Configuration

General | Channel 0 | Channel 1 | Channel 2 | Chan. 1 - Modbus

Driver: Ethernet

Hardware Address: 00:00:BC:67:23:B9 Network Link ID: 0

IP Address: 130 . 88 . 214 . 8

Subnet Mask: 255 . 255 . 255 . 0

Gateway Address: 0 . 0 . 0 . 0

Default Domain Name:

Primary Name Server: 0 . 0 . 0 . 0

Secondary Name Server: 0 . 0 . 0 . 0

User Provided Web Pages:

Starting Data File Number: 0

Number of Pages: 1

Protocol Control:

BOOTP Enable DHCP Enable Msg Connection Timeout (x 1mS): 15000

SNMP Server Enable SMTP Client Enable Msg Reply Timeout (x 1mS): 3000

HTTP Server Enable DNP3 over IP Enable Inactivity Timeout (x Min): 30

Modbus TCP Enable

Disable EtherNet/IP Incoming Connections

Auto Negotiate Disable Duplicate IP Address Detection

Port Setting: 10/100 Mbps Full Duplex/Half Duplex

- 9) Go to the Chan. 1 – Modbus tab that appears in the Channel Configuration and assign an unused Data File number to the registers or coils you wish to read/write. For this guide, we will assign a value of 9 to Holding Registers (4xxxx) in order to create a Data File number 9 that will contain our registers. Hit enter, and then assign the Size (in # of words) to a value of 2, since we would like to write/read 2 registers from the ML1400.

Channel Configuration

General | Channel 0 | Channel 1 | Chan. 1 - Modbus | Channel 2

Modbus TCP configuration:

Modbus Data Table File Numbers:

Coils (0xxxx) 0 Input Registers (3xxxx) 0

Contacts (1xxxx) 0 Holding Registers (4xxxx) 9 Expanded

Enable Access Control for IP Addresses

Client IP0: 0 . 0 . 0 . 0 Local Port Number TCP: 502

Client IP1: 0 . 0 . 0 . 0 Diagnostic File: 0

Client IP2: 0

Client IP3: 0

Client IP4: 0

Create Modbus Holding Table (Data File #9)?

Yes

No

Size (in # of words): 2

Integer

Binary

10) Verify the project and download it to the ML1400 (you will need to power cycle the ML1400 in order for the Ethernet changes to take effect). Go online with the ML1400 and open the N9 – Holding Data file that was created and monitor the values.

11) Put the Micro800 into Debug mode after having downloaded a program and toggle the **Start_Write** contact. You should see the *Write_MSG* FB complete without errors and the ML1400 should now display the values under **N9:0** and **N9:1**.

The screenshot displays a PLC programming environment. At the top, a ladder logic diagram shows a normally open contact labeled 'Start_Write' connected to the 'IN' input of a function block named 'Write_MSG'. The function block has several outputs: 'Cancel' (False), 'Error' (False), 'Local_ ErrorID' (0), 'Target_ SubEr...' (0), and 'LocalA_ Status...' (17). Below the function block, four parameters are listed: 'Cancel_Write_MSG' (False), 'LocalPar_Write' (4), 'TargetPar_Write' (1), and 'LocalAdd_Write' (12345).

Overlaid on the right is a 'Variable Monitoring' window. It shows a table of variables with their logical and physical values:

Name	Logical Value	Physical Value	Lock
Start_Write	<input checked="" type="checkbox"/>	N/A	<input type="checkbox"/>
Read_MSG	<input type="checkbox"/>
Write_MSG	<input type="checkbox"/>
Cancel_Read_Msg	<input type="checkbox"/>	N/A	<input type="checkbox"/>
LocalPar_Read	<input type="checkbox"/>

At the bottom, a 'Data File N9 (dec)' window shows a table of data points:

Offset	0	1	2	3	4	5	6	7	8	9
N9:0	12345	6789								

The 'Data File N9 (dec)' window also includes a search field with 'N9:0' entered, a 'Radix' dropdown set to 'Decimal', and a 'Columns' dropdown set to '10'. Buttons for 'Properties', 'Usage', and 'Help' are visible at the bottom of the window.

12) Now do the same for the *Read_MSG* FB and you should see the value that was read from the ML1400 be displayed under **LocalAdd_Read** in the Micro850 variables.

2

Start_Read

Read_MSG
MSG_MODB...

IN Q

Cancel_Read_Msg
False

Cancel Error False

LocalPar_Read
4

Local... ErrorID 0

TargetPar_Read
1

Target... SubEr... 0

LocalAdd_Read
12345

LocalA... Status... 17

Variable Monitoring

User Global Variables · Local Variables · Prog1 System Variables · Micro

Name	Logical Value	Physical Value	Lock
Cancel_Read_Msg	<input type="checkbox"/>	N/A	<input type="checkbox"/>
LocalPar_Read	<input type="checkbox"/>
TargetPar_Read	<input type="checkbox"/>
LocalAdd_Read	<input type="checkbox"/>
Start_Read	<input checked="" type="checkbox"/>	N/A	<input type="checkbox"/>

Close

Prog1-VAR × Prog1-POU

Name	Logical Value	Physical Value	Lock	Data Type
LocalAdd_Read	<input type="checkbox"/>	MODBUSLOCADI
LocalAdd_Read[1]	12345	N/A	<input type="checkbox"/>	WORD
LocalAdd_Read[2]	6789	N/A	<input type="checkbox"/>	WORD