ACL Manufacturing
A Division of ClearRushCo

Modbus Manual

for the

CSC200 Controller

V 1.6

5406, Township Rd 325B Sundre, Alberta, Canada T0M 1X0
(403) 638-2287 info@clearrushco.com www.clearrushco.com

Table of Contents

Introduction and Summary

The CSC200 Controllers are able to communicate remotely with Modbus Master Devices. A Modbus Master Device may be a Programmable Logic Controller, a PC, or another device. The CSC200 Controller is a Modbus Slave Device that implements the Modbus RTU protocol on an RS-485, half-duplex, physical connection.

The CSC200 has a hardware revision of 2A and current firmware version of 3.7 (minimum).

The default Modbus communication parameters are 9600 baud, 8 data bits, no parity bits, one stop bit ("8N1"), Modbus Slave ID (Modbus address) 2.

CSC200 Controller Modbus Quick Summary	
Protocol	RTU
Physical Connection	RS485, half-duplex
Hardware Revision	2A
Firmware Version (minimum)	3.7
Default Settings	
Baud rate	9600
Number of data bits	8
Parity bit setting	None
Stop bits	1
Slave ID (Modbus Address)	2

THIS EQUIPMENT IS SUITABLE FOR USE IN CLASS1 DIVISION 2, GROUPS A,B,C & D OR NONHAZARDOUS LOCATIONS ONLY

WARNING - EXPLOSION HAZARD - SUBSTITUTION OF COMPONENTS MAY IMPAIR THE SUITABILITY FOR CLASS 1 DIVISION 2

WARNING: EXPOSURE TO SOME CHEMICALS MAY DEGRADE THE SEALING PROPERTIES OF MATERIALS USED IN THE FOLLOWING DEVICES:

Four position DIP switch SW2
Relays K1 – K5, K7, K8
Twelve-position DIP switch S1
Four-position DIP switch S2

Additional Documents

The following additional documents for the CSC200 Combustion Safety Controller are available.

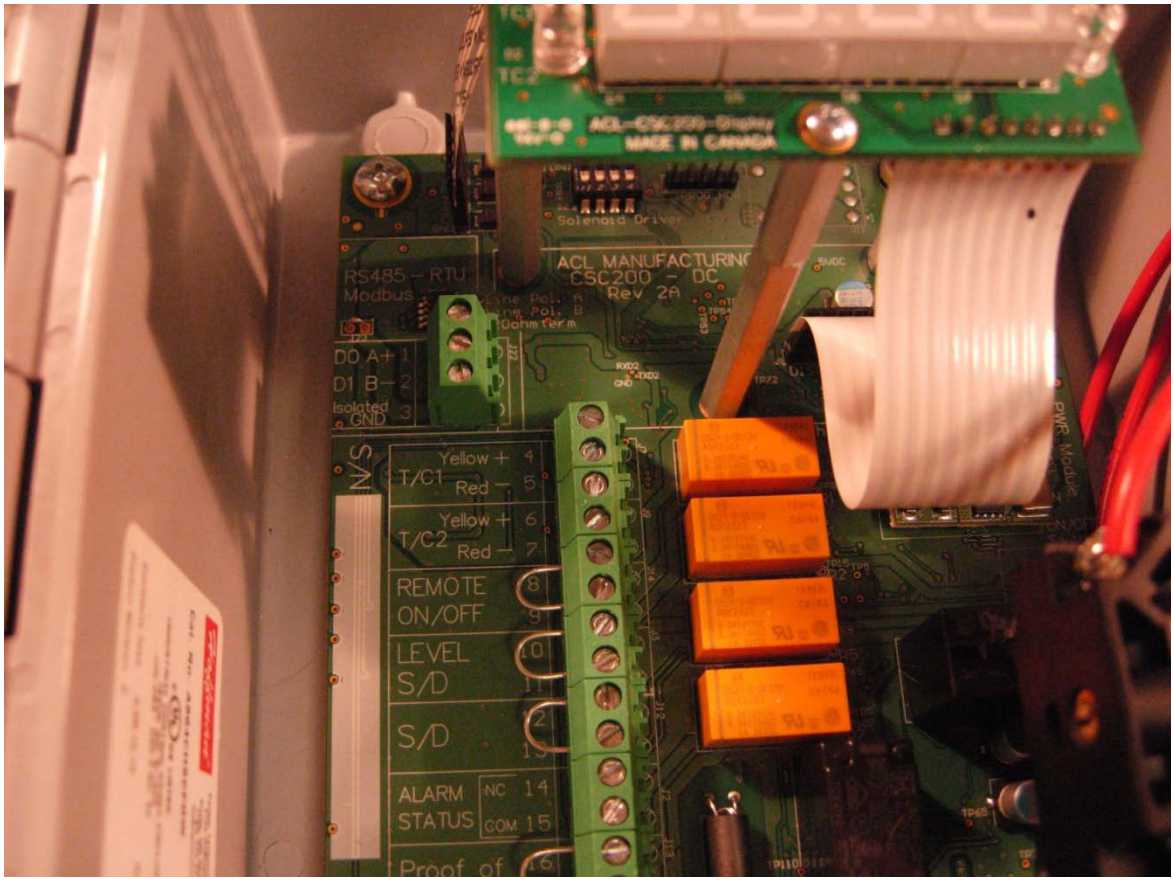
Document Filename	Document Description
CSC200_Rev_2A_Installation_Manual.pdf	CSC200 installation information and quickstart instructions.

Quickstart Installation Procedure

CSC200 Controller - Modbus

> The Quickstart Installation Instructions assumes the user has some familiarity with Modbus and Modbus cabling and communications.

Figure 1 – Final Assembly, Zoomed-out Front-Top View



Programming a New Modbus Slave ID Using a PC

1) Connect one end of a USB to RS-485 cable to the three screw terminals of the CSC200 Controller (refer to Appendix D for details if necessary). Connect the USB end to a PC. This CSC200 should be the only device attached to the RS-485 bus while changing the Slave ID (address) to avoid potential conflicts. Additional details may be found in Appendices E and D.

2) Run the desired Modbus Master software (examples are Modnet for Modbus or Modbus Constructor) and connect to the COM port used by the USB-to-RS485 cable. Default serial settings for the CSC200 are 9600 baud, 8N1, Modbus RTU.

3) Select the unique Slave ID for the CSC200 to communicate to (default Slave ID for a new CSC200 is “2”). Issue a Write Single Holding Register command to Modbus Holding Register Address 4 (“Unlock Slave ID register”) using the value 0x55AA (21930). This command unlocks the Slave ID for changing it. This is used as a safety precaution to prevent inadvertent Slave ID changing. See the section on page 7 titled "Commonly Used CSC200 Modbus Registers" for more information on the modbus registers needed.

General Modbus Command:			
Command to Write	Modbus Function Code	Write Address	Value to write
Write Single Holding Register	0x06	4 (“Unlock Slave ID register”)	0x55AA (21930)
Modnet for Modbus RTU Software:			
Function		Write Address	Value to write
(06) Write Single Register		4 (“Unlock Slave ID register”)	21930

4) Issue a Write Single Holding Register command to Modbus Holding Register Address 5 (“Slave ID register”) using the new desired Modbus Slave ID (address) that you want to assign to this CSC200. Values between 0x0001 and 0x00F7 are allowed. Note that the Modbus specification says that at least 32 Modbus devices can reside on one RS-485 bus (without repeaters). Testing needs to be done by the installer to ensure adequate signal integrity if more than 32 devices are placed on one Modbus RS-485 bus.

General Modbus Command:			
Command to Write	Modbus Function Code	Write Address	Value to write
Write Single Holding Register	0x06	5 (“Slave ID register”)	Desired Modbus address value between 0x0001 and 0x00F7 (between 1 and 247)
Modnet for Modbus RTU Software:			
Function		Write Address	Value to write
(06) Write Single Register		5 (“Slave ID register”)	Desired Modbus address value between 1 and 247 (between 0x0001 and 0x00F7)

Modbus/RS-485 Cable Connections – Field Installations

Special Notes
Ensure that only industrial-rated equipment is used for field installations, with appropriate measures for handling noisy environments.

If using a PC with USB-to-RS485 connectivity for field installations, use an industrial-rated USB hub (preferably one with a metal case) for connecting the PC to the USB-to-RS485 cable.

Refer to Appendix C for additional Modbus cabling technical details.

Cabling

Connect a cable from a PLC (Programmable Logic Controller) or a PC to the 3-pin terminal strip of the CSC200 labeled "Modbus", observing proper connections:

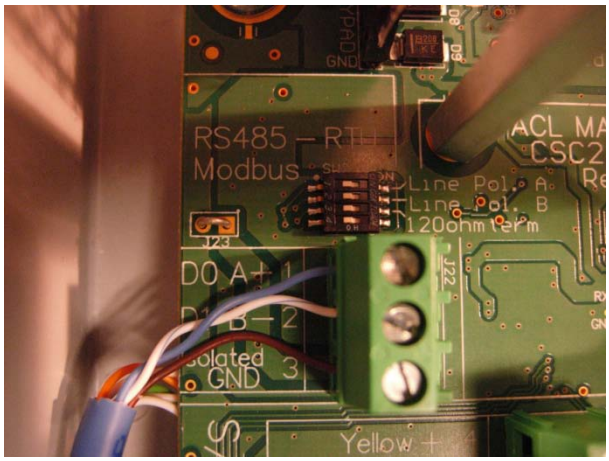
- The RS-485 standard suggests using twisted pair type cables (CAT5E or a shielded twisted pair with ground) for connecting devices together. This is definitely a requirement for longer cable runs (25m to 1000m) and for use in noisy environments like industrial or commercial installations.

- The RS485 signal naming convention used in this document and by many RS485 transceiver vendors is reversed from what the EIA/TIA-485 specification states:

CSC200 Modbus/RS485 Documentation	EIA/TIA-485 Naming Convention	Modbus Specification Name	Description
A (“RS485 A +” or "D0 A+")	B	D1	Non-Inverting, Transceiver Terminal 1, V1 voltage (V1 > V0 for binary 1 (OFF) state
B (“RS485 B –” or "D1 B-")	A	D0	Inverting, Transceiver Terminal 0, V0 voltage (V0 > V1 for binary 0 (ON) state
Isolated GND (or common GND)	C	Common	Signal and Optional Power Supply common ground

- Ensure that the “Isolated Ground” terminals are all attached together on all RS485 devices on the bus. This ground should be connected to earth ground at one point along the bus, preferably at the Master.

Figure 2 - Example CAT5E Cable Connection



- If using a CAT5E (or similar) cable with unused wires, do not leave them "floating". Connect these wires at one point on the cable to the ground (or "Isolated GND") terminal at the CSC200, or at the master's ground terminal.

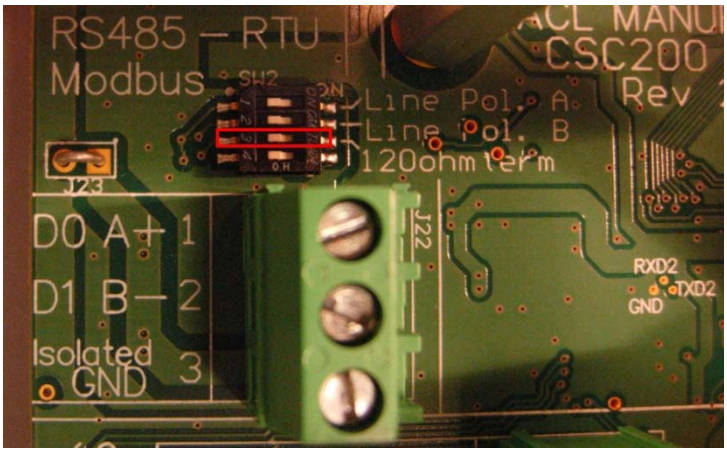
- A USB-to-RS485 cable may also have unused wires if the provided Terminator resistor wires are not used (the FTDI Chip cable as an example). These should be connected to ground as well, to reduce noise propagation.

Termination

An RS-485 bus should only be terminated at each end of the cable (at each device at the end of the cable). No other devices in-between the two devices at each end should have termination resistors installed or enabled. If there are 20 devices on an RS-485 bus in a daisy-chain, the 120 ohm termination resistors should only be enabled at the first device and at the 20th device.

The CSC200 Controller has a 4-pin DIP switch with the third switch from the top labeled "120ohm term". This can be used to connect a built-in 120 ohm resistor. Simply push the third DIP switch to the right and the 120ohm termination resistor will be connected.

Figure 3 - 120 ohm Termination Resistor DIP Switch

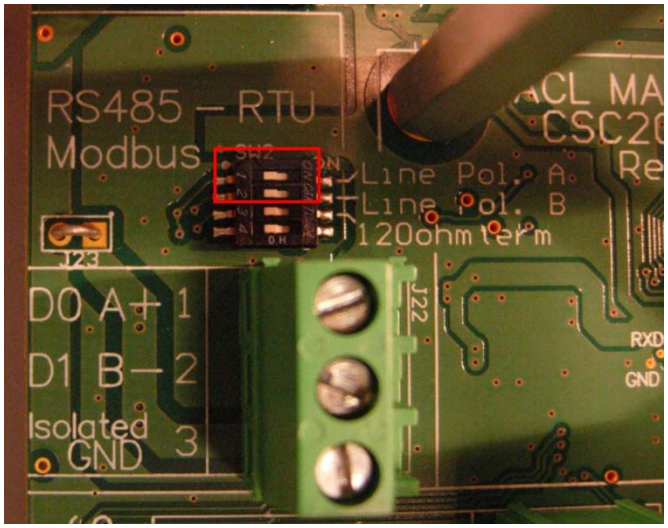


Line Polarization

If Line Polarization is not available on the Master device and is required for the RS-485 bus in this installation, two "Line Polarization" DIP switches on the CSC200 Controller are available. To enable the Line Polarization terminations, move them

to the right (towards the “Line Pol…” text) as shown in the picture below (Figure 4). If the DIP switches are moved towards the left, the Line Polarization terminations are removed from the RS-485 bus on this CSC200 device.

Figure 4 – Line Polarization DIP Switches



“Line Polarization” enables a pullup resistor on the “Data A +” signal and a pulldown resistor on the “Data B –“ signal. It ensures that the bus is put into a known state with the “Data A +” signal High and the “Data B -” signal Low.

Line Polarization should only be enabled on one device on the RS485 bus, if necessary. Usually this is done at the end of the bus where the master device resides.

Isolated (or Common) Ground

The “Isolated Ground” terminal on each CSC200 Controller is isolated from the onboard CSC200 ground. This isolated ground connection should be used to connect all common ground connections on all RS-485 devices on the bus. This common ground should be connected to earth or protective ground at one end of the RS-485 cable only (preferably), usually at the master device.

Due to the potential for large amounts of noise to be conducted onto the RS485 cable, an option is provided to connect the RS485 isolated ground to the CSC200 earth ground to shunt noise away locally instead of at the Modbus master. A solid ground connection should be made between a CSC200 earth ground terminal to an earth ground external to the CSC200 using a minimum 16AWG wire.

Commonly Used CSC200 Modbus Registers

Notes:

- SCADAPack Register Addresses are listed for reference when programming SCADAPack PLC units.
- See Appendix A - Full CSC200 Modbus Registers List for additional registers and specific details about reading and writing registers.

Function Code 0x01 - Read Coils

Function used to read the state of each relay. Read Coil function code 0x01 can read all relay coils in one byte.

SCADAPack Register Address	Coil #	Modbus Coil Address	Description	Type	Notes
1	1	0	Reserved		
2	2	1	Reserved		
3	3	2	Pilot solenoid relay	Solenoid relay	
4	4	3	Main solenoid relay	Solenoid relay	
5	5	4	Alarm relay	Control relay	
6	6	5	Not used	N/A	Reserved
7	7	6	Proof of closure relay	Control relay	
8	8	7	Temperature Main solenoid relay	Solenoid relay	

Function Code 0x02 - Read Discrete Inputs

This Function is used to read the state of each input. 1 = ON, 0 = OFF (unless otherwise stated)

SCADA Pack Register Address	Input #	Modbus Discrete Input Address	Inputs Byte	Input Bit	Description	Notes
10001	1	0	0	0 (LSB)	Igniter Alarm input	1 = Alarm signal high (Alarm indicated)
10002	2	1	0	1	Igniter Valve input	1 = Valve signal high
10003	3	2	0	2	Main solenoid	1 = Main solenoid is on
10004	4	3	0	3	Pilot solenoid	1 = Pilot solenoid is on
10005	5	4	0	4	T/Main solenoid	1 = T/Main solenoid is on
10006	6	5	0	5	On/Off switch "minus" input	1 = On/Off switch is On (12VDC present), 0 = OFF
10007	7	6	0	6	POC relay output	1 = POC relay output is High (12VDC present)
10008	8	7	0	7 (MSB)	POC minus terminal	1 = POC "minus" terminal is High (12VDC present)
10009	9	8	1	0 (LSB)	Shutdown input	1 = Shutdown input is High (12VDC present, shutdown sensor not tripped)
10010	10	9	1	1	Remote Reset input	1 = Remote Reset switch is On/Closed (12VDC present)
10011	11	10	1	2	HT input: On/Off switch "plus" input (output of TC2 "R2" relay)	1 = High Temp R2 relay output is High (12VDC is present, not in High Temp shutdown), 0 = high temp shutdown
10012	12	11	1	3	Output of TC1 "R1" relay (input to POC relay)	1 = "Low" Temp R1 relay output is High (12VDC is

						present), 0 = TC1 temp is in shutdown (if in Intermittent Pilot mode)
10013	13	12	1	4	PWR fail condition (only on briefly upon powerup)	
10014	14	13	1	5	PWR fail latch condition	1 = Latch is on presently
10015	15	14	1	6	HT/HT latch condition	1 = Latch is on presently
10016	16	15	1	7 (MSB)	SD/SD latch condition	1 = Latch is on presently
10017	17	16	2	0 (LSB)	Thermocouple 1 open/fault	1 = TC fault, 0 = no fault
10018	18	17	2	1	Thermocouple 2 open/fault	1 = TC fault, 0 = no fault
10019	19	18	2	2	Modbus Remote Stop condition	1 = Modbus Remote Stop is active (CSC200 is stopped via Modbus)
10020	20	19	2	3	Level Shutdown input	1 = Level Shutdown input is High (12VDC present, shutdown sensor not tripped)
10021	21	20	2	4	Pilot Solenoid Fault (Short)	1 = Solenoid fault (short), 0 = no fault
10022	22	21	2	5	Main Solenoid Fault (Short)	1 = Solenoid fault (short), 0 = no fault
10023	23	22	2	6	TMain Solenoid Fault (Short)	1 = Solenoid fault (short), 0 = no fault
10024	24	23	2	7 (MSB)	TC1 calibration ratio error (out of acceptable range)	1 = an error was detected in the calibration ratio for TC1. A recalibration is needed
					DIP Switches, first byte	
10025	25	24	3	0 (LSB)	Power Fail Latch Select	1 = Power Fail Latch Select is ON
10026	26	25	3	1	High Temp Latch Select	1 = High Temp Latch Select is ON
10027	27	26	3	2	Shutdown Latch Select	0 = Shutdown Latch Select is ON
10028	28	27	3	3	Intermittent / Continuous Pilot Select	0 = Intermittent Pilot, 1 = Continuous Pilot Select
10029	29	28	3	4	Power Save	1 = Power Save ON (dim LED display after 2 min of no button adjustments)
10030	30	29	3	5	TC2 High / Low Range Select	1 = TC2 Low range select, 0 = TC2 High range select
10031	31	30	3	6	TC2 Disable / Enable	0 = TC2 Disable, 1 = Enable
10032	32	31	3	7 (MSB)	TC1 High / Low Range Select	1 = TC1 Low range select, 0 = TC1 High range select
					DIP Switches, second byte	
10033	33	32	4	0 (LSB)	Button Disable / Enable	
10034	34	33	4	1	Deadband 2	DB2,DB1 = 0,0 --> 5 deg C, 10 deg F

10035	35	34	4	2	Deadband 1	DB2,DB1 = 0,1 --> 3 deg C, 6 deg F
						DB2,DB1 = 1,0 --> 2 deg C, 4 deg F
						DB2,DB1 = 1,1 --> 1 deg C, 2 deg F
10036	36	35	4	3	Deg C / Deg F	0 = Display Temp in Deg C, 1 = Display Temp in Deg F
10037	37	36	4	4	Low Power Solenoid: 40%	1 = OFF, 0 = LP sol mode ON, solenoid driven at 40% (power driving solenoid is the sum of percentages turned ON, eg: 40% and 10% ON = solenoid driven at 50%)
10038	38	37	4	5	Low Power Solenoid: 20%	1 = OFF, 0 = LP sol mode ON, solenoid driven at 20% (power driving solenoid is the sum of percentages turned ON, eg: 20% and 10% ON = solenoid driven at 30%)
10039	39	38	4	6	Low Power Solenoid: 10%	1 = OFF, 0 = LP sol mode ON, solenoid driven at 10%
10040	40	39	4	7 (MSB)	TC2 calibration ratio error (out of acceptable range)	1 = an error was detected in the calibration ratio for TC2. A recalibration is needed

Function Code 0x03 - Read Holding Registers

Holding registers are 16-bit values (2 bytes)
Register bytes are read back as MSB then LSB

SCADA Pack Register Address	Register #	Modbus Holding Register Address	Description	Notes
40001	1	0	TC1 temp setpoint (deg C)	
40002	2	1	TC2 temp setpoint (deg C)	
40003	3	2	TC1 temp setpoint (deg F)	
40004	4	3	TC2 temp setpoint (deg F)	

Function Code 0x04 - Read Input Registers

Input registers are 16-bit values (2 bytes)
Register bytes are read back as MSB then LSB

SCADA Pack Register Address	Register #	Modbus Inputs Register Address	Description	Notes
30001	1	0	TC1 current temp (deg C)	
30002	2	1	TC2 current temp (deg C)	
30003	3	2	TC1 current temp (deg F)	
30004	4	3	TC2 current temp (deg F)	

Function Code 0x05 – Write Single "Coil" (or Setting)

The individual coils can't actually be written to, they're influenced by the temperature.
Remote Stop and Remote Start are allowed though.
Remote Stop will turn off all relays in the CSC200. CSC200 can only be started again by a Remote Start command, or by turning ON/OFF switch to OFF, then back to ON.

SCADAPack Register Address	Coil #	Modbus Write Coil Address	Description	Type	Notes
	1-8		(No direct write access to relays)		
9	9	8	Increment TC1 setpoint		ON = increment TC1 setpoint, OFF = no effect
10	10	9	Decrement TC1 setpoint		ON = increment TC1 setpoint, OFF = no effect
11	11	10	Increment TC2 setpoint		ON = increment TC1 setpoint, OFF = no effect
12	12	11	Decrement TC2 setpoint		ON = increment TC1 setpoint, OFF = no effect
13	13	12	Remote Stop		ON = Stop, OFF = no effect
14	14	13	Remote Start		ON = Start, OFF = no effect

Notes:

- Write Single "Coil" (or setting) function code 0x05 can increment/decrement the setpoint temperatures of either thermocouple, and can also trigger a Remote Stop or Remote Start command.
- "0xFF00" (or 65280 in decimal) turns a "coil" ON, "0x0000" turns a coil "OFF"
- For our "coils" or settings, 0x0000 or OFF, has no effect on the Setpoints or Remote Stop/Start settings.
- Remote Stop disables all power going to ignition module and closes all three valve solenoids
- Remote Stop can be cleared by a physical toggling of the ON/OFF switch or the Remote Reset power rung
- Remote Stop can also be cleared by receiving a Modbus message turning Remote Start ON
- Remote Start enables the CSC200 to be turned on
- Remote Start can be interrupted if ON/OFF switch is OFF, if Remote Reset is open, or if Shutdown is open, or if POC is still open
- Remote Start can also be cleared by receiving a Modbus message turning Remote Stop ON

Function Code 0x06 - Write Holding Registers

Holding registers are 16-bit values (2 bytes)
Register bytes are written as MSB then LSB

SCADA Pack Register Address	Register #	Modbus Holding Register Address	Description	Notes
40001	1	0	TC1 temp setpoint (deg C)	Writing a value to TC1 in degrees C, also writes to the TC1 degrees F register (after conversion)
40002	2	1	TC2 temp setpoint (deg C)	Writing a value to TC2 in degrees C, also writes to the TC2 degrees F register (after conversion)
40003	3	2	TC1 temp setpoint (deg F)	Writing a value to TC1 in degrees F, also writes to the TC1 degrees C register (after conversion)
40004	4	3	TC2 temp setpoint (deg F)	Writing a value to TC2 in degrees F, also writes to the TC2 degrees C register (after conversion)
40005	5	4	Unlock Slave ID and / or Baud Rate and Serial Format register	Write a "0x55AA" (21930) to this register to unlock the Slave ID for changing. Write a "0x775D" (30557) to this register to unlock the Baud rate or Serial Format registers for changing.
40006	6	5	Slave ID register	Write the new Slave ID value to use for this CSC200 unit to this register once it's been "unlocked" using the previous register (register address 4)
				(ID change is made after the response is sent)
				(Unlock Slave ID register (reg # 5, address 4) is also reset to zero after the Slave ID is changed)

Appendix A - Full CSC200 Modbus Registers List

Supported Modbus Function Code Commands for the CSC200

Function Code		Sub-Function Code		Function Name	Length (bits)	Description of Use With CSC200
(Dec)	(Hex)	(Dec)	(Hex)			
1	0x01			Read Coils	1	Used to read the state of each relay
2	0x02			Read Discrete Inputs	1	Used to read the state of each input
3	0x03			Read Holding Registers	16	Used to read the holding registers
4	0x04			Read Input Registers	16	Used to read the input registers
5	0x05			Write Single “Coil” (or setting)	1	Used to increment/decrement temperature setpoint and controls Remote Start/Stop
6	0x06			Write Single Holding Register	16	Used to write values to individual holding registers for setup or control
7	0x07			Read Exception Status		Unused at the moment
8	0x08			Diagnostics		
		0	0x00	Return Query Data (loopback)		Echoes the request back to the Master
		1	0x01	Restart Communications Option		Restart communications port and brings device out of Listen Only mode if currently in it
		4	0x04	Force Listen Only Mode		Device will not respond to requests if put in this mode
		10	0x0A	Clear Counters and Diagnostic Register		Clear Counters and Diagnostic Register
		11	0x0B	Return Bus Message Count		Returns number of messages on the bus since last restart, clear counters operation, or powerup (even if not addressed to this device)
		12	0x0C	Return Bus Communication Error Count		Returns number of CRC errors since last restart, clear counters operation, or powerup
		13	0x0D	Return Bus Exception Error Count		Returns number of exception responses sent back to the Master since last restart, clear counters operation, or powerup
		14	0x0E	Return Slave Message Count		Returns number of messages addressed to this device since last restart, clear counters operation, or powerup
11	0x0B			Get Communication Event Counter		Used to get a status word and an event count from the communication event counter
12	0x0C			Get Communication Event Log		Used to get a status word, event count, message count, and a field of event bytes from the CSC200. The status word and event counts are identical to that returned by the Get Communications Event Counter function (11, 0B hex).
17	0x11			Report Slave ID		Used to read the Slave ID, the description of the type, the current status, and other information specific to

						the CSC200.
43	0x2B	14	0x0E	Read Device Identification		Allows reading the identification and additional information relative to the physical and functional description of the CSC200

Notes:

- "Length" refers to the number of bits used for each value. For example, a coil is 1 bit in length (either a zero or a one) whereas a Holding Register is 16 bits in length (values are from 0 to 65535 (0xFFFF))

Specific Modbus Function Code Register Details

Notes:

- SCADAPack Register Addresses are listed for reference when programming SCADAPack PLC units.

Function Code 0x01 - Read Coils

Function used to read the state of each relay

SCADAPack Register Address	Coil #	Modbus Coil Address	Description	Type	Notes
1	1	0	Reserved		
2	2	1	Reserved		
3	3	2	Pilot solenoid relay	Solenoid relay	
4	4	3	Main solenoid relay	Solenoid relay	
5	5	4	Alarm relay	Control relay	
6	6	5	Not used	N/A	Reserved
7	7	6	Proof of closure relay	Control relay	
8	8	7	Temperature Main solenoid relay	Solenoid relay	

Notes:

- Read Coil function code 0x01 can read all relay coils in one byte.

Recommended Modbus Read Coils request message sent to CSC200 (PDU, protocol data unit):

0x01 0x00 0x00 0x00 0x08

Function - Read Coils	0x01
Starting Address Hi	0x00
Starting Address Lo	0x00
Quantity of Outputs Hi	0x00
Quantity of Outputs Lo	0x08

Modbus Read Coils response message sent back to Master from CSC200 (PDU, protocol data unit):

0x01 0x01 0xXX

Function - Read Coils	0x01
Byte Count	0x01
Output (Coil) Status	0xXX where XX is the byte holding the current status of the coils in the same configuration as above

Function Code 0x02 - Read Discrete Inputs

Function used to read the state of each input
1 = ON, 0 = OFF (unless otherwise stated)

SCADA Pack Register Address	Input #	Modbus Discrete Input Address	Inputs Byte	Input Bit	Description	Notes
10001	1	0	0	0 (LSB)	Igniter Alarm input	1 = Alarm signal high (Alarm indicated)
10002	2	1	0	1	Igniter Valve input	1 = Valve signal high
10003	3	2	0	2	Main solenoid	1 = Main solenoid is on
10004	4	3	0	3	Pilot solenoid	1 = Pilot solenoid is on
10005	5	4	0	4	T/Main solenoid	1 = T/Main solenoid is on
10006	6	5	0	5	On/Off switch "minus" input	1 = On/Off switch is On (12VDC present), 0 = OFF
10007	7	6	0	6	POC relay output	1 = POC relay output is High (12VDC present)
10008	8	7	0	7 (MSB)	POC minus terminal	1 = POC "minus" terminal is High (12VDC present)
10009	9	8	1	0 (LSB)	Shutdown input	1 = Shutdown input is High (12VDC present, shutdown sensor not tripped)
10010	10	9	1	1	Remote Reset input	1 = Remote Reset switch is On/Closed (12VDC present)
10011	11	10	1	2	HT input: On/Off switch "plus" input (output of TC2 "R2" relay)	1 = High Temp R2 relay output is High (12VDC is present, not in High Temp shutdown), 0 = high temp shutdown
10012	12	11	1	3	Output of TC1 "R1" relay (input to POC relay)	1 = "Low" Temp R1 relay output is High (12VDC is present), 0 = TC1 temp is in shutdown (if in Intermittent Pilot mode)
10013	13	12	1	4	PWR fail condition (only on briefly upon powerup)	
10014	14	13	1	5	PWR fail latch condition	1 = Latch is on presently
10015	15	14	1	6	HT/HT latch condition	1 = Latch is on presently
10016	16	15	1	7 (MSB)	SD/SD latch condition	1 = Latch is on presently

10017	17	16	2	0 (LSB)	Thermocouple 1 open/fault	1 = TC fault, 0 = no fault
10018	18	17	2	1	Thermocouple 2 open/fault	1 = TC fault, 0 = no fault
10019	19	18	2	2	Modbus Remote Stop condition	1 = Modbus Remote Stop is active (CSC200 is stopped via Modbus)
10020	20	19	2	3	Level Shutdown input	1 = Level Shutdown input is High (12VDC present, shutdown sensor not tripped)
10021	21	20	2	4	Pilot Solenoid Fault (Short)	1 = Solenoid fault (short), 0 = no fault
10022	22	21	2	5	Main Solenoid Fault (Short)	1 = Solenoid fault (short), 0 = no fault
10023	23	22	2	6	TMain Solenoid Fault (Short)	1 = Solenoid fault (short), 0 = no fault
10024	24	23	2	7 (MSB)	TC1 calibration ratio error (out of acceptable range)	1 = an error was detected in the calibration ratio for TC1. A recalibration is needed
					DIP Switches, first byte	
10025	25	24	3	0 (LSB)	Power Fail Latch Select	1 = Power Fail Latch Select is ON
10026	26	25	3	1	High Temp Latch Select	1 = High Temp Latch Select is ON
10027	27	26	3	2	Shutdown Latch Select	0 = Shutdown Latch Select is ON
10028	28	27	3	3	Intermittent / Continuous Pilot Select	0 = Intermittent Pilot, 1 = Continuous Pilot Select
10029	29	28	3	4	Power Save	1 = Power Save ON (dim LED display after 2 min of no button adjustments)
10030	30	29	3	5	TC2 High / Low Range Select	1 = TC2 Low range select, 0 = TC2 High range select
10031	31	30	3	6	TC2 Disable / Enable	0 = TC2 Disable, 1 = Enable
10032	32	31	3	7 (MSB)	TC1 High / Low Range Select	1 = TC1 Low range select, 0 = TC1 High range select
					DIP Switches, second byte	
10033	33	32	4	0 (LSB)	Button Disable / Enable	
10034	34	33	4	1	Deadband 2	DB2,DB1 = 0,0 --> 5 deg C, 10 deg F
10035	35	34	4	2	Deadband 1	DB2,DB1 = 0,1 --> 3 deg C, 6 deg F
						DB2,DB1 = 1,0 --> 2 deg C, 4 deg F
						DB2,DB1 = 1,1 --> 1 deg C, 2 deg F
10036	36	35	4	3	Deg C / Deg F	0 = Display Temp in Deg C, 1 = Display Temp in Deg F
10037	37	36	4	4	Low Power Solenoid: 40%	1 = OFF, 0 = LP sol mode ON, solenoid driven at 40%

						(power driving solenoid is the sum of percentages turned ON, eg: 40% and 10% ON = solenoid driven at 50%)
10038	38	37	4	5	Low Power Solenoid: 20%	1 = OFF, 0 = LP sol mode ON, solenoid driven at 20% (power driving solenoid is the sum of percentages turned ON, eg: 20% and 10% ON = solenoid driven at 30%)
10039	39	38	4	6	Low Power Solenoid: 10%	1 = OFF, 0 = LP sol mode ON, solenoid driven at 10%
10040	40	39	4	7 (MSB)	TC2 calibration ratio error (out of acceptable range)	1 = an error was detected in the calibration ratio for TC2. A recalibration is needed

Notes:

- Read discrete inputs function code 0x02 can read all inputs used for decision making and DIP switches.

Recommended Modbus Read Discrete inputs request message sent to CSC200 (PDU, protocol data unit):

0x02 0x00 0x00 0x00 0x28 (a read includes the reserved input bits)

Function - Read Discrete Inputs	0x02
Starting Address Hi	0x00
Starting Address Lo	0x00
Quantity of Outputs Hi	0x00
Quantity of Outputs Lo	0x28

Modbus Read Discrete Inputs response message sent back to Master from CSC200 (PDU, protocol data unit):

0x02 0x05 0xXX 0xXX 0xXX 0xXX 0xXX

Function - Read Discrete Inputs	0x02	
Byte Count	0x05	
Inputs Status Byte 0	0xXX	Inputs byte 0
Inputs Status Byte 1	0xXX	Inputs byte 1
Inputs Status Byte 2	0xXX	Inputs byte 2
Inputs Status Byte 3	0xXX	Inputs byte 3 (DIP Switches, first byte)
Inputs Status Byte 4	0xXX	Inputs byte 4 (DIP Switches, second byte)

Function Code 0x03 - Read Holding Registers

Holding registers are 16-bit values (2 bytes)
Register bytes are read back as MSB then LSB

SCADA Pack Register Address	Register #	Modbus Holding Register Address	Description	Notes
40001	1	0	TC1 temp setpoint (deg C)	
40002	2	1	TC2 temp setpoint (deg C)	

40003	3	2	TC1 temp setpoint (deg F)	
40004	4	3	TC2 temp setpoint (deg F)	
40005	5	4	Unlock Slave ID (and / or Baud Rate and Serial Format) register	Reads as "0x55AA" (21930 decimal) or "0x775D" (30557 decimal) if unlocked, "0x0000" otherwise
40006	6	5	Slave ID register	Current Slave ID of this CSC200
40007	7	6	Baud rate selection	(see Description of values in Function Code 0x06 - Write Single Holding Register)
40008	8	7	Serial Format selection	(see Description of values in Function Code 0x06 - Write Single Holding Register)
40009	9	8	Reset serial communication settings to default	Reads as "0x0000" always
40010	10	9	Temperature log: Enable/Disable	"0x00" = disable Temp logging
				"0x01" = enable Temp logging but only when not in shutdown (OFF, HT, SD, RR, Remote Stop, POC) (Default)
				"0x11" = enable Temp logging, even when in shutdown (OFF, HT, SD, RR, Remote Stop, POC)
40011	11	10	Temperature log: Overwrite Type setting	Value of "0" = Save log, do not overwrite if full
				Value of "1" = Allow overwriting, CSC200 only keeps the most recent data (default)
40012	12	11	Temperature log: Record Rate setting	(see Description of values in Function Code 0x06 - Write Single Holding Register)
40013	13	12	Temperature log: Reset log	Reads as "0x0000" always
40014	14	13	Temperature log: Total Count	Holds the number of temperature measurements currently in each log (TC1 and TC2). Max size is currently 512
40015	15	14	Temperature log: Temperature Format	Value of "0" = store temp in currently selected format (eg: deg C if deg C selected by degC/degF DIP switch)
				Value of "1" = save Temp in degrees Celsius
				Value of "2" = save Temp in degrees Fahrenheit
40016	16	15	Shutdown log: Overwrite Type setting	Value of "0" = Save log, do not overwrite if full
				Value of "1" = Allow overwriting, CSC200 only keeps the most recent data
40017	17	16	Shutdown log: Clear/Reset	Reads as "0x0000" always
40018	18	17	Shutdown log: Total Count	Holds the number of shutdowns detected stored currently in the log. Max size is currently 128
40019	19	18	Shutdown log: Mask register	Selects the type of shutdowns to store in the shutdown log. Uses lower byte of holding register. A "1" enables the selected shutdown to be stored in the Shutdown log. Eg: 0x13 (binary 0001 0011) only enables storing Power Fails, Shutdown power rung, and High-Temp shutdowns.
				Bit 0: High-Temp

				Bit 1: Shutdown power rung
				Bit 2: Remote reset power rung
				Bit 3: Modbus remote stop
				Bit 4: Power Fails
				Bit 5: On/Off Switch
				Bit 6: Flame Fails
				Bit 7: Flame Fail Retries
40020	20	19	Shutdown Count: Flame Fail Retries	All shutdown counts are 16-bits (range is 0 to 65535)
40021	21	20	Shutdown Count: Flame Fails	
40022	22	21	Shutdown Count: On/Off Switch	
40023	23	22	Shutdown Count: Power Fails	
40024	24	23	Shutdown Count: Modbus Remote Stops	
40025	25	24	Shutdown Count: Remote Reset	
40026	26	25	Shutdown Count: Shutdown Power rung	
40027	27	26	Shutdown Count: High-Temp shutdowns	
40028	28	27	Shutdown Count: Level Shutdown Power rung	
40029	29	28	Shutdown Counts: Clear/Reset	Reads as "0x0000" always
40030	30	29	TMain time on, days	Indicates the number of total days the TMain valve has been open/ON. Full result = days, hours, minutes
40031	31	30	TMain time on, hours	Indicates the number of hours the TMain valve has been open/ON.
40032	32	31	TMain time on, minutes	Indicates the number of minutes the TMain valve has been open/ON.
40033	33	32	TMain time on, Clear/Reset	Clears/Zeros the TMain ON time in all variables (days, hours, minutes), and in the EEPROM.
40257 – 40768	257 – 768	0x100 – 0x2FF	Temperature log: TC1 Temperature Values read access	Max log size is 1024 bytes for TC1: 512 16-bit temperature values (512 = 0x200)
		(256 – 767)		(A Reset log command (address 12 or 0x0C) is needed to clear the buffer once all the data has been read out)
				Temperature log contents are lost if a power failure occurs. Temperature log settings are saved though.
41281 - 41792	1281 - 1792	0x500 – 0x6FF	Temperature log: TC2 Temperature Values read	Max log size is 1024 bytes for TC2: 512 16-bit temperature values (512 = 0x200)

			access	
		(1280 – 1791)		(A Reset log command (address 12 or 0x0C) is needed to clear the buffer once all the data has been read out)
				Temperature log contents are lost if a power failure occurs. Temperature log settings are saved though.
42049 - 42176	2049 - 2176	0x800 – 0x87F	Shutdown log: Log Values read access	Max log size is 128 bytes (records a max of 128 shutdowns): 128 8-bit shutdown values (read as 16-bits, upper 8 bits are zeros)
		(2048 – 2175)		Values are read back as 16-bit values due to the nature of Modbus registers: MSB is always 00
				Max value is 125 (125 (0x7D) 16-bit values = 250 bytes) for each read command.
				If there's more values in the log, the Master must adjust the starting Address to read from, the number of value to read and issue another read command
				(A Reset log command (address 16 or 0x10) is needed to clear the buffer once all the data has been read out)
				Shutdown log contents are lost if a power failure occurs. Shutdown log settings are saved though.
				Shutdown log byte organization (in LSB):
				MSB LSB
				15 … 8 7 … 0
				Bit 0: High-Temp
				Bit 1: Shutdown power rung
				Bit 2: Remote reset power rung
				Bit 3: Modbus remote stop
				Bit 4: Power Fail
				Bit 5: On/Off Switch
				Bit 6: Flame Fail
				Bit 7: Flame Fail Retry

Notes:

- Read Holding registers function code 0x03 can read the internal register settings for the CSC200 and the Temperature and Shutdown logs.
- Some registers are used as "Write-only" registers (see function code 0x06, Write Single Holding Register, for descriptions of the write only registers)

Recommended Modbus Read Holding Registers request message sent to CSC200 (PDU, protocol data unit):

0x03 0x00 0x0D 0x00 0x01

Function - Read Holding Registers	0x03	
Starting Address Hi	0x00	
Starting Address Lo	0x0D	0x0D = 13 : Temperature log: Total Count
Number of Registers Hi	0x00	
Number of Registers Lo	0x01	

Modbus Read Holding Registers response message sent back to Master from CSC200 (PDU, protocol data unit):

0x03 0x02 0x00 0x37

Function - Read Input Registers	0x03	
Byte Count	0x02	
Register Value Hi Byte	0x00	Value = 0x0037 = 55 values available for reading in each Temperature log (TC1 and TC2)
Register Value Lo Byte	0x37	

Function Code 0x04 - Read Input Registers

Input registers are 16-bit values (2 bytes)
Register bytes are read back as MSB then LSB

SCADA Pack Register Address	Register #	Modbus Inputs Register Address	Description	Notes
30001	1	0	TC1 current temp (deg C)	
30002	2	1	TC2 current temp (deg C)	
30003	3	2	TC1 current temp (deg F)	
30004	4	3	TC2 current temp (deg F)	

Notes:

- Read input registers function code 0x04 can read the temperature of both thermocouples in either degrees C or F.

Recommended Modbus Read Input Registers request message sent to CSC200 (PDU, protocol data unit):

0x04 0x00 0x02 0x00 0x01

Function - Read Input Registers	0x04	
Starting Address Hi	0x00	
Starting Address Lo	0x02	
Quantity of Outputs Hi	0x00	
Quantity of Outputs Lo	0x01	

Modbus Read Input Registers response message sent back to Master from CSC200 (PDU, protocol data unit):

0x04 0x02 0xYY 0xXX

Function - Read Input Registers	0x04	
Byte Count	0x02	
Input Reg. 3 Hi Byte	0xYY	Input register #3, Hi byte
Input Reg. 3 Lo Byte	0xXX	Input register #3, Lo byte

Function Code 0x05 – Write Single "Coil" (or Setting)

The individual coils can't actually be written to, they're influenced by the temperature.
Remote Stop and Remote Start are allowed though.
Remote Stop will turn off all relays in the CSC200. CSC200 can only be started again by a Remote Start command, or by turning ON/OFF switch to OFF, then back to ON.

SCADAPack Register Address	Coil #	Modbus Write Coil Address	Description	Type	Notes
1	1	0	Reserved		
2	2	1	Reserved		
3	3	2	Pilot solenoid relay	Solenoid relay	
4	4	3	Main solenoid relay	Solenoid relay	
5	5	4	Alarm relay	Control relay	
6	6	5	Not used	N/A	Reserved
7	7	6	Proof of closure relay	Control relay	
8	8	7	Temperature Main solenoid relay	Solenoid relay	
9	9	8	Increment TC1 setpoint	Control	ON = increment TC1 setpoint, OFF = no effect
10	10	9	Decrement TC1 setpoint	Control	ON = increment TC1 setpoint, OFF = no effect
11	11	10	Increment TC2 setpoint	Control	ON = increment TC1 setpoint, OFF = no effect
12	12	11	Decrement TC2 setpoint	Control	ON = increment TC1 setpoint, OFF = no effect
13	13	12	Remote Stop	Control	ON = Stop, OFF = no effect
14	14	13	Remote Start	Control	ON = Start, OFF = no effect

Notes:

- Write Single "Coil" (or setting) function code 0x05 can increment/decrement the setpoint temperatures of either thermocouple, and can also trigger a Remote Stop or Remote Start command.
- "0xFF00" (or 65280 in decimal) turns a "coil" ON, "0x0000" turns a coil "OFF"
- For our "coils" or settings, 0x0000 or OFF, has no effect on the Setpoints or Remote Stop/Start settings.
- Remote Stop disables all power going to ignition module and closes all three valve solenoids
- Remote Stop can be cleared by a physical toggling of the ON/OFF switch or the Remote Reset power rung
- Remote Stop can also be cleared by receiving a Modbus message turning Remote Start ON

- Remote Start enables the CSC200 to be turned on
- Remote Start can be interrupted if ON/OFF switch is OFF, if Remote Reset is open, or if Shutdown is open, or if POC is still open
- Remote Start can also be cleared by receiving a Modbus message turning Remote Stop ON

Recommended Modbus Single "Coil" (or setting) request message sent to CSC200 (PDU, protocol data unit):

0x05 0x00 0x0B 0xFF 0x00

Function - Write "Coil" (or setting)	0x05	
"Coil" or setting Address Hi	0x00	
"Coil" or setting Address Lo	0x0B	This example (0x0B = 11) command decrements the TC2 setpoint by one degree for each command sent to the device from the master
"Coil" or setting Value Hi	0xFF	
"Coil" or setting Value Lo	0x00	

Modbus Single "Coil" (or setting) response message sent back to Master from CSC200 (PDU, protocol data unit):

0x05 0x00 0x0B 0xFF 0x00

Function - Write "Coil" (or setting)	0x05
"Coil" or setting Address Hi	0x00
"Coil" or setting Address Lo	0x0B
"Coil" or setting Value Hi	0xFF
"Coil" or setting Value Lo	0x00

Function Code 0x06 - Write Holding Registers

Holding registers are 16-bit values (2 bytes)
Register bytes are written as MSB then LSB

SCADA Pack Register Address	Register #	Modbus Holding Register Address		Description	Notes
40001	1	0		TC1 temp setpoint (deg C)	Writing a value to TC1 in degrees C, also writes to the TC1 degrees F register (after conversion)
40002	2	1		TC2 temp setpoint (deg C)	Writing a value to TC2 in degrees C, also writes to the TC2 degrees F register (after conversion)
40003	3	2		TC1 temp setpoint (deg F)	Writing a value to TC1 in degrees F, also writes to the TC1 degrees C register (after conversion)
40004	4	3		TC2 temp setpoint (deg F)	Writing a value to TC2 in degrees F, also writes to the TC2 degrees C register (after conversion)
40005	5	4		Unlock Slave ID and / or Baud Rate and Serial Format register	Write a "0x55AA" (21930) to this register to unlock the Slave ID for changing. Write a "0x775D" (30557) to this register to unlock the Baud rate or Serial Format registers for changing.
40006	6	5		Slave ID register	Write the new Slave ID value to use for this CSC200 unit to this register once it's been "unlocked" using the previous register (register address 4)
					(ID change is made after the response is sent)
					(Unlock Slave ID register (reg # 5, address 4) is also reset to zero after the Slave ID is changed)
40007	7	6		Baud rate selection	Write one of the following code values to select a new baud rate to use once it has

					been unlocked using Holding Register Address 4. Write 0x775D (30557) to Holding Register Address 4 to unlock.
					Value of "0" = 300 baud
					Value of "1" = 1200 baud
					Value of "2" = 2400 baud
					Value of "3" = 4800 baud
					Value of "4" = 9600 baud (default)
					Value of "5" = 19200 baud
					Value of "6" = 38400 baud
					(All changes are made after the response is sent)
40008	8	7		Serial Format selection	Write one of the following code values to select a new serial formal to use once it has been unlocked using Holding Register Address 4. Write 0x775D (30557) to Holding Register Address 4 to unlock.
					Value of "0" = 8-N-1 (8 bits, no parity bits, 1 stop bit) (default)
					Value of "1" = 8-E-1 (8 bits, even parity, 1 stop bit)
					Value of "2" = 8-O-1 (8 bits, odd parity, 1 stop bit)
					Value of "3" = 8-N-2 (8 bits, no parity bits, 2 stop bits)
					(All changes made after the response is sent)
40009	9	8		Reset serial communication settings to default	"0xFFFF" resets serial communication settings to default after the response is sent, all other values have no effect
					- Resets serial communication settings to 9600 baud and 8N1 format
40010	10	9		Temperature log: Enable/Disable	"0x00" = disable Temp logging
					"0x01" = enable Temp logging but only when not in shutdown (OFF, HT, SD, RR, Remote Stop, POC) (Default)
					"0x11" = enable Temp logging, even when in shutdown (OFF, HT, SD, RR, Remote Stop, POC)
40011	11	10		Temperature log: Overwrite Type setting	Value of "0" = Save log, do not overwrite if full
					Value of "1" = Allow overwriting, CSC200 only keeps the most recent data (default)
40012	12	11		Temperature log: Record Rate setting	Value of "0" = save Temp every 5 minutes

					Value of "1" = save Temp every 10 minutes
					Value of "2" = save Temp every 15 minutes
					Value of "3" = save Temp every 20 minutes
					Value of "4" = save Temp every 30 minutes
					Value of "5" = save Temp every 60 minutes (default)
					Value of "6" = save Temp every 120 minutes
					Value of "7" = save Temp every 3 hours
					Value of "8" = save Temp every 4 hours
					Value of "9" = save Temp every 6 hours
					Values of "10" and above are reserved for debugging and future updates
40013	13	12		Temperature log: Reset log	Writing a "0xFFFF" here resets (zeros) the Temp log, all other values have no effect
40014	14	13	No	Temperature log: Total Count (no write access)	Holds the number of temperature measurements currently in each log (TC1 and TC2). Max size is currently 512. (no write access)
40015	15	14		Temperature log: Temperature Format	Value of "0" = store temp in currently selected format (eg: deg C if deg C selected by degC/degF DIP switch)
					Value of "1" = save Temp in degrees Celsius
					Value of "2" = save Temp in degrees Fahrenheit
40016	16	15		Shutdown log: Overwrite Type setting	Value of "0" = Save log, do not overwrite if full
					Value of "1" = Allow overwriting, CSC200 only keeps the most recent data
40017	17	16		Shutdown log: Clear/Reset	Writing a "0xFFFF" here resets (zeros) the Shutdown log, all other values have no effect
40018	18	17	No	Shutdown log: Total Count (no write access)	Holds the number of shutdowns detected stored currently in the log. Max size is currently 128. (no write access)
40019	19	18		Shutdown log: Mask register (Default value is 0x00D3: all shutdowns are logged except for ON/OFF switch toggling, Remote Reset, and modbus Remote Stop)	Selects the type of shutdowns to store in the shutdown log. Uses lower byte of holding register. A "1" enables the selected shutdown to be stored in the Shutdown log. Eg: 0x13 (binary 0001 0011) only enables storing Power Fails, Shutdown power rung, and High-Temp shutdowns.
					A "1" in the selected bit position enables that type of shutdown to be recorded into the shutdown log.

					A "0" in the selected bit position means that that type of shutdown is NOT recorded in the shutdown log. It is still counted though in its corresponding count register
					Bit 0: High-Temp
					Bit 1: Shutdown power rung
					Bit 2: Remote reset power rung
					Bit 3: Modbus remote stop
					Bit 4: Power Fails
					Bit 5: On/Off Switch
					Bit 6: Flame Fails
					Bit 7: Flame Fail Retries
40020	20	19	No	Shutdown Count: Flame Fail Retries	All shutdown counts are 16-bits (range is 0 to 65535) and have no write access other than the "Shutdown Counts: Clear/Reset" register
40021	21	20	No	Shutdown Count: Flame Fails	
40022	22	21	No	Shutdown Count: On/Off Switch	
40023	23	22	No	Shutdown Count: Power Fails	
40024	24	23	No	Shutdown Count: Modbus Remote Stops	
40025	25	24	No	Shutdown Count: Remote Reset	
40026	26	25	No	Shutdown Count: Shutdown Power rung	
40027	27	26	No	Shutdown Count: High-Temp shutdowns	
40028	28	27	No	Shutdown Count: Level Shutdown Power rung	
40029	29	28		Shutdown Counts: Clear/Reset	Clears/Zeros the shutdown counts in all shutdown count variables, and in EEPROM
					Writing a "0xFFFF" here resets (zeros) the Shutdown counters, all other values have no effect
40030	30	29	No	TMain time on, days (no write access)	Indicates the number of total days the TMain valve has been open/ON. Full result = days, hours, minutes
40031	31	30	No	TMain time on, hours (no write access)	Indicates the number of hours the TMain valve has been open/ON.
40032	32	31	No	TMain time on, minutes (no write access)	Indicates the number of minutes the TMain valve has been open/ON.
40033	33	32		TMain time on, Clear/Reset	Clears/Zeros the TMain ON time in all variables (days, hours, minutes) and in the EEPROM.
					Writing a "0xFFFF" here resets (zeros) the TMain ON timers, all other values have no effect.

40257 – 40768	257 – 768	0x100 – 0x2FF		Temperature log: TC1 Temperature Values read access (no write access)	Max log size is 1024 bytes for TC1: 512 16-bit temperature values (512 = 0x200)
		(256 – 767)			(A Reset log command (address 12 or 0x0C) is needed to clear the buffer once all the data has been read out)
					Temperature log contents are lost if a power failure occurs. Temperature log settings are saved though.
41281 - 41792	1281 - 1792	0x500 – 0x6FF		Temperature log: TC2 Temperature Values read access (no write access)	Max log size is 1024 bytes for TC2: 512 16-bit temperature values (512 = 0x200)
		(1280 – 1791)			(A Reset log command (address 12 or 0x0C) is needed to clear the buffer once all the data has been read out)
					Temperature log contents are lost if a power failure occurs. Temperature log settings are saved though.
42049 - 42176	2049 - 2176	0x800 – 0x87F		Shutdown log: Log Values read access (no write access)	Max log size is 128 bytes (records a max of 128 shutdowns): 128 8-bit shutdown values (read as 16-bits, upper 8 bits are zeros)
		(2048 – 2175)			Values are read back as 16-bit values due to the nature of Modbus registers: MSB is always 00
					Max value is 125 (125 (0x7D) 16-bit values = 250 bytes) for each read command.
					If there's more values in the log, the Master must adjust the starting Address to read from, the number of value to read and issue another read command
					(A Reset log command (address 16 or 0x10) is needed to clear the buffer once all the data has been read out)
					Shutdown log contents are lost if a power failure occurs. Shutdown log settings are saved though.
					Shutdown log byte organization (in LSB):
					MSB LSB
					15 … 8 7 … 0
					Bit 0: High-Temp
					Bit 1: Shutdown power rung
					Bit 2: Remote reset power rung
					Bit 3: Modbus remote stop
					Bit 4: Power Fail
					Bit 5: On/Off Switch
					Bit 6: Flame Fail
					Bit 7: Flame Fail Retry

					(accessing an address outside the area containing valid data in the shutdown log will return two bytes of 0x00 0x00)

Notes:

- Write Holding registers function code 0x06 can write the internal register settings for the CSC200 and the Temperature and Shutdown log settings.

Recommended Modbus Write Holding Registers request message sent to CSC200 (PDU, protocol data unit):

0x06 0x00 0x03 0x01 0xF4

Function - Write Holding Register	0x06	
Register Address Hi	0x00	
Register Address Lo	0x03	This example (register address 0x03) sets the setpoint of TC2 in degrees F to a value of 500 degrees F (0x01F4).
Register Value Hi	0x01	
Register Value Lo	0xF4	

Modbus Single "Coil" (or setting) response message sent back to Master from CSC200 (PDU, protocol data unit):

0x06 0x00 0x03 0x01 0xF4

Function - Write Holding Register	0x06
Register Address Hi	0x00
Register Address Lo	0x03
Register Value Hi	0x01
Register Value Lo	0xF4

Function Code 0x07 - Read Exception Status

(This function code is not used by the CSC200 at the moment)

SCADA Pack Register Address	Bit #	Modbus Read Exception Status Register Address (bit)	Description	Notes
	1	0		
	2	1		
	3	2		
	4	3		
	5	4		
	6	5		
	7	6		
	8	7		

Notes from the Modbus Application Protocol Document:

- "This function code is used to read the contents of eight Exception Status outputs in a remote device."

- "The contents of the eight Exception Status outputs are device specific."

Function Code 0x08 - Diagnostics

Sub-Function Code		Function Name	Length (bits)	Description of Use With CSC200
(Dec)	(Hex)			
0	0x00	Return Query Data (loopback)		Echoes the request back to the Master
1	0x01	Restart Communications Option		Restart communications port and brings device out of Listen Only mode if currently in it
4	0x04	Force Listen Only Mode		Device will not respond to requests if put in this mode
10	0x0A	Clear Counters and Diagnostic Register		Clear Counters and Diagnostic Register
11	0x0B	Return Bus Message Count		Returns number of messages on the bus since last restart, clear counters operation, or powerup (even if not addressed to this device)
12	0x0C	Return Bus Communication Error Count		Returns number of CRC errors since last restart, clear counters operation, or powerup
13	0x0D	Return Bus Exception Error Count		Returns number of exception responses sent back to the Master since last restart, clear counters operation, or powerup
14	0x0E	Return Slave Message Count		Returns number of messages addressed to this device since last restart, clear counters operation, or powerup
		Get Communication Event Counter		Used to get a status word and an event count from the communication event counter
		Get Communication Event Log		Used to get a status word, event count, message count, and a field of event bytes from the CSC200. The status word and event counts are identical to that returned by the Get Communications Event Counter function (11, 0B hex).
		Report Slave ID		Used to read the Slave ID, the description of the type, the current status, and other information specific to the CSC200.
14	0x0E	Read Device Identification		Allows reading the identification and additional information relative to the physical and functional description of the CSC200

Function Code 11 (0x0B) – Get Communication Event Counter

This function code is used to get a status word and an event count from the remote device's communication event counter. Device's event counter is incremented once for each successful message completion.

Recommended Modbus "Get Communication Event Counter" request message sent to CSC200 (PDU, protocol data unit):

0x0B

Function - "Get Comm. Event …" 0x0B

Modbus "Get Communication Event Counter" response message sent back to Master from CSC200 (PDU, protocol data unit):

0x0B 0x00 0x00 0xXX 0xXX

Function - "Get Comm. Event …"	0x0B	
Status Hi	0x00	Status word is 0xFFFF if busy with a previous command, otherwise the response is 0x0000
Status Lo	0x00	
Event Counter Hi	0xXX	where XX XX is the 2 bytes holding the current event count
Event Counter Lo	0xXX	

Function Code 12 (0x0C) – Get Communication Event Log

This function code is used to get a status word, event count, message count, and a field of event bytes from the remote device. The status word and event counts are identical to that returned by the Get Communications Event Counter function (11, 0B hex).
The message counter contains the quantity of messages processed by the remote device since its last restart, clear counters operation, or power–up.
The remote device enters the events into the field in chronological order. Byte 0 is the most recent event.

Recommended Modbus "Get Communication Event Log" request message sent to CSC200 (PDU, protocol data unit):

0x0C

Function - "Get Comm. Event …"	0x0C

Modbus "Get Communication Event Log" response message sent back to Master from CSC200 (PDU, protocol data unit):

0x0C 0x08 0x00 0x00 0xXX 0xXX 0xZZ 0xZZ 0xEV 0xEV

Function - "Get Comm. Event …"	0x0C	
Byte Count	0x08	
Status Hi	0x00	Status word is 0xFFFF if busy with a previous command, otherwise the response is 0x0000
Status Lo	0x00	
Event Counter Hi	0xXX	where 0xXXXX is the 2 bytes holding the current event count
Event Counter Lo	0xXX	The status word and event counts are identical to that returned by the Get Communications Event Counter function (11, 0B hex).
Message Counter Hi	0xZZ	where 0xZZZZ is the 2 bytes holding the current message count
Message Counter Lo	0xZZ	
Event 0	0xEV	where 0xEVEV is an example showing the event log
Event 1	0xEV	The most recent communications event is shown in Event 0 byte. The previous event is shown in Event 1 byte. The total number of event bytes is 0 - 64

Function Code 17 (0x11) – Report Slave ID

This function code is used to read the description of the type, the current status, and other information specific to a remote device.

Recommended Modbus "Report Slave ID" request message sent to CSC200 (PDU, protocol data unit):

0x11

Function - "Report Slave ID"	0x11

Modbus "Report Slave ID" response message sent back to Master from CSC200 (PDU, protocol data unit):

0x11 0x02 0xXX 0xFF

Function - "Report Slave ID"	0x11	
Byte Count	0x04	
Slave ID	0xXX	Current ID byte of this slave device
Run Indicator Status - On/Off Switch	0xFF	0x00 = OFF, 0xFF = ON "ON" = CSC200 is ON and running, "OFF" = ON/OFF switch is OFF or Remote Stop has been triggered
Run Indicator Status - SD	0xFF	0x00 = OFF, 0xFF = ON

		Shutdown input current state:"ON" = CSC200 is ON and running, "OFF" = Shutdown has triggered, ON/OFF switch is OFF or Remote Stop has been triggered
Run Indicator Status - POC relay	0xFF	0x00 = OFF, 0xFF = ON POC relay current state:"ON" = POC relay is on and system is running, "OFF" = POC is open or relay 7 isn't sending power to the Ignition module, Shutdown has triggered, ON/OFF switch is OFF or Remote Stop has been triggered

Function Code 43 / 14 (0x2B / 0x0E) - Read Device Identification

This function code allows reading the identification and additional information relative to the physical and functional description of a remote device.

Appendix B - CSC200 Rev 2A Modbus Technical Specifications

Notes:

- Receivers are designed to fail-safe to a logic high output state if inputs (terminals A and B) are left un-driven or shorted. If the bus is un-driven for long periods of time, the receivers are designed to not require line polarization on the bus (adding a pullup resistor to "A" and a pulldown resistor to "B"). Line polarization may be enabled (via the two DIP switches on the top of the CSC200 Controller) for use with other devices on the same RS-485 bus.
- Drivers are protected from excess current flow caused by bus contention or output short-circuits by both an internal current limit and a thermal-overload shutdown.
- RS-485 inputs (terminals A and B) are protected against ESD events up to +/- 15kV (Air-Gap and Human Body Model) and up to +/- 8kV Contact Discharge (IEC61000-4-2).
- All components on the CSC200 Controller are RoHS compliant.

Specification	Default Value	Possible Values
Modbus Protocol	Modbus RTU	Modbus RTU
Modbus Slave ID (address)	2	1 - 247
Modbus/RS-485 Serial Settings:		
Baud rate	9600	300, 1200, 2400, 4800, 9600, 19200, 38400
Number of data bits	8	8
Parity bit setting	None	None, Even, Odd
Stop bits	1	1, 2 (only with parity set to "None")
Operating Temperature		-40°C to 60°C
RS-485 Signals:		
Input voltage on A and B signals		-7 VDC to +12 VDC
Driver Short Circuit Current Limit		+/- 250mA maximum
Differential Driver Output, No Load		5 VDC
Differential Driver Output, R_L = 54ohms		1.5 VDC minimum 2.7 VDC typical 5 VDC maximum
Receiver Input Resistance		96kohm minimum (1/8th of a Modbus "Unit Load")
Receiver Differential Threshold (VA – VB)		-200mV minimum -125mV typical -40mV maximum
Receiver Input Hysteresis		25mV typical
Termination		None or 120ohms (2-pin jumper may be installed by user)
Line Polarization Resistors		560 ohms +/- 1%, selectable by user via two DIP switches
Line Polarization Pullup voltage		5 VDC +/- 1% (5% max)
Line Polarization Pulldown voltage		RS-485 Isolated or Common GND (0V)
Physical Dimensions:		

Length		6.750” (171.45mm)
Width		4.850” (123.19mm)
Height, maximum (from bottom of components on bottom layer to top of components on top layer)		1.130” (28.70mm)

Appendix C - Modbus/RS-485 Cabling Technical Details

Refer to the Modbus documentation available at www.modbus.org:

RS-485 Signal Naming Conventions

The RS485 signal naming convention used in this document and by many RS485 transceiver vendors is reversed from what the EIA/TIA-485 specification states:

CSC200 Modbus/RS485 Documentation	EIA/TIA-485 Naming Convention	Modbus Specification Name	Description
A ("Data A +")	B	D1	Non-Inverting, Transceiver Terminal 1, V1 voltage (V1 > V0 for binary 1 (OFF) state
B ("Data B –")	A	D0	Inverting, Transceiver Terminal 0, V0 voltage (V0 > V1 for binary 0 (ON) state
Isolated GND (or common GND)	C	Common	Signal and Optional Power Supply common ground

Half-Duplex vs Full-Duplex

Half-duplex communication allows only one device to communicate over the 2 RS-485 wires (one differential pair). Full-duplex communication adds another pair of wires to allow bi-directional communication to occur simultaneously.

For the Modbus protocol, the Master pair would be used by the master to communicate to the slave devices on the full-duplex connection, and the Slave Pair would be used by slaves for transmitting messages back to the master. This could happen simultaneously.

Cable Types

Master Used	Cable Type To Use For Testing	Notes
PC	USB to RS485 cable	RS485 cable should have stripped wires for connecting to terminal blocks on the CSC200 Controller
PLC – Programmable Logic Controller (eg: SCADAPack, ROC800 series)	CAT5E	Use a matched twisted pair for RS485A+/B- Eg: Blue for RS485A+ Blue with white stripe for RS485B-

Allowable Pairings of CAT5E Cable

Signal	CAT5E Cable Wire Color Twisted Pairs	Notes
RS485A + or Data +	Blue	
RS485B - or Data -	Blue with white stripe	
RS485A + or Data +	Green	
RS485B - or Data -	Green with white stripe	
RS485A + or Data +	Orange	

RS485B - or Data -	Orange with white stripe	
RS485A + or Data +	Brown	
RS485B - or Data -	Brown with white stripe	

The common ground connection should use a wire from an unused pair in the CAT5E cable.

Examples of USB to RS485 cables

Manufacturer	Part #	Length	Website	Available at
Moxa	UPort 1130/1130I or UPort 1150/1150I		www.moxa.com	www.moxa.com
FTDI Chip	USB-RS485-WE-5000-BT	5m	www.ftdichip.com	www.digikey.com, www.mouser.com
FTDI Chip	USB-RS485-WE-1800-BT	1.8m	www.ftdichip.com	www.digikey.com, www.mouser.com
Startech	ICUSB422	6ft	www.startech.com	www.startech.com

Industrial-Rated USB Hubs

Manufacturer	Part #	Website	Available at
Startech	ST4200USBM	www.startech.com	www.startech.com
Moxa	UPort 404, UPort 407	www.moxa.com	www.moxa.com

Wiring topology

For connecting multiple Modbus devices on to the same RS-485 bus, a “daisy-chain” wiring topology should be used (one long cable with short “stub” connections to each device). Ensure that short “stub” connections are made at each device to the main RS485 cable to reduce signal reflections and interference.

A “star” or “ring” wiring topology should not be used. An example of a “star” configuration would be separate, multiple cables branching out from the Master to each individual slave device. Only one cable should be connected at the Master end.

Line Polarization

Line Polarization enables a pullup resistor on the “Data A +” signal and a pulldown resistor on the “Data B –“ signal. It ensures that the bus is put into a known state with the “Data A +” signal High and the “Data B -” signal Low. Some RS485 receivers are susceptible to external noise or interference if the RS485 bus is not driven to a known state when the bus is idle (no device is driving a signal on the bus).

Line Polarization should only be enabled on one device on the RS485 bus, if necessary. Usually this is done at the end of the bus where the master device resides. The CSC200 Controller allows the implementation of Line Polarization via two DIP switches located on the top of the board.

Some PC software (or other Masters) will work with Line Polarization off, while others may need the non-inverting signal to be driven high during idle times on the RS485 bus. For example, the PC software Modnet for Modbus RTU will work with Line Polarization off but it shows an extra "0x00" byte received at the beginning and end of a Modbus packet. However, the Modbus Reader PC software shows a Frame Error received by the CSC200 Controller if no Line Polarization is turned on.

Termination

This type of termination refers to bus termination between the pairs, not the termination resistors used for Line Polarization. This termination connects signal "Data A +" to "Data B –" through a 120 ohm resistor.

An RS-485 bus should only be terminated at each end of the cable (at each device at the end of the cable). No other devices in-between the two devices at each end should have termination resistors installed or enabled.

The CSC200 Controller has a 4-pin DIP switch with the third switch from the top labeled "120ohm term". This can be used to connect a built-in 120 ohm resistor. Simply push the third DIP switch to the right and the 120ohm termination resistor will be connected.

Number of Allowed Devices on the RS-485

The number of devices allowed on an RS-485 bus depends on a variety of factors: the total length of the wire, the wire gauge, the signaling characteristics or the "Unit Load" of each device on the bus (receiver input impedance, capacitance).

The CSC200 Controller uses newer RS485 transceivers with advanced fail-safe features. Due to these newer transceivers, the theoretical maximum number of devices allowed on the bus is 256 because the receiver's input impedance is 96kohm which is $1/8^{th}$ the input impedance of older transceivers at 12kohm ($1/8^{th}$ of a "Unit Load"). The Modbus specification limits this theoretical maximum further to 247 devices allowed on an RS-485 bus.

Any Modbus system allows a minimum of 32 devices on the RS-485 bus without use of a repeater. More devices may be allowed depending on the characteristics of all devices on the RS-485 bus.

The CSC200 Controller allows more than 32 devices to be present on the RS-485 bus due to each transceiver occupying $1/8^{th}$ of a Unit Load on the bus. Since each installation is different, with different cable lengths and the potential for other devices to be present on the bus, the user needs to test out the maximum number of devices that can be placed on each RS-485 bus.

Slew Rate

The CSC200 Controller incorporates RS-485 transceivers with slew rate limited drivers. Slew rate refers to the speed at which a signal changes state from a 0 (Low) to a 1 (High) or from a High to a Low state. Slew rate limited drivers slow down the rise and fall times of a signal which help with reducing signal reflections, reducing EMI emissions, and possibly allowing a bus to work without termination resistors.

Unfortunately, with slower rise and fall times, the maximum communication speed (or baud rate) is reduced. The drivers on the CSC200 Controller can operate at a maximum rate of 115kbps but the maximum setting allowed in the CSC200 firmware is 38.4kbps (38400 baud, or raw bits per second).

Isolated (or Common) Ground

The "Isolated Ground" terminal on each CSC200 Controller is isolated from the onboard CSC200 ground. This isolated ground connection should be used to connect all common ground connections on all RS-485 devices on the bus. This common ground should be connected to earth or protective ground at one end of the RS-485 cable only (preferably), usually at the master device.

Due to the potential for large amounts of noise to be conducted onto the RS485 cable, an option is provided to connect the RS485 isolated ground to the CSC200 earth ground to shunt noise away locally instead of at the Modbus master. A solid ground connection should be made between a CSC200 earth ground terminal to an earth ground external to the CSC200 using a minimum 16AWG wire.

Appendix D - Modbus Communication Tests

The Modbus communication between a Master device and the CSC200 Slave should be tested once the CSC200 Controller Modbus cabling is installed to ensure proper operation. Each CSC200 Modbus Slave device should also have its Modbus Slave ID (address) changed to a unique value before field installation takes place.

Connect Modbus test cabling between the CSC200 Controller and a PLC (Programmable Logic Controller) or a PC, referring to the following tables:

Cable Connections to Use Depending on the Master Used For Testing

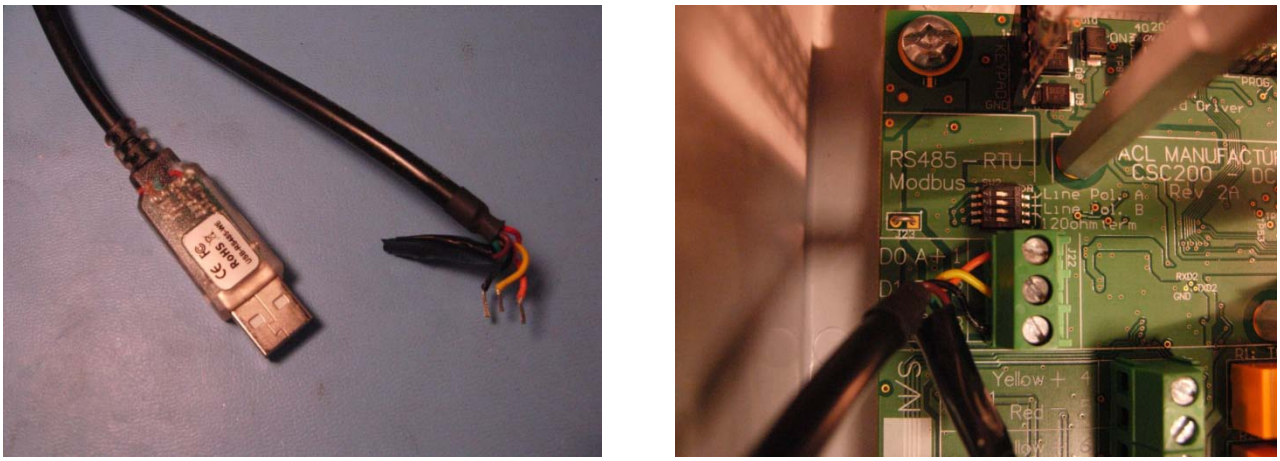
Master Used	Cable Connections	CSC200 Controller Terminal
PC	Data + (A)	"RS485A +" or "D0 A+"
	Data – (B)	"RS485B -" or "D1 B-"
	Ground	GND
PLC – Programmable Logic Controller (eg: SCADAPack, ROC800 series)	CAT5E pair + (Eg. Blue wire)	"RS485A +" or "D0 A+"
	CAT5E pair - (Eg. Blue with white stripe wire)	"RS485B -" or "D1 B-"
	CAT5E wire from an unused CAT5E pair	GND

Example Cable Connection – PC Master

Cable used: FTDI Chip USB-RS485-WE-5000-BT, 5m, USB to RS-485 cable.

Cable Signal	FTDI Chip USB-RS485-WE-5000-BT (double-check these wire colors with the cable received)
Data + (A)	Orange wire
Data – (B)	Yellow wire
Ground	Black wire
Terminator 120ohm, pin 1	Brown wire
Terminator 120ohm, pin 2	Green wire

Figure 5 - FTDI Chip USB-RS485-WE-5000-BT USB to RS485 Cable, Installed with CSC200 Controller



Note that in this example, the unused wires are insulated from shorting to other parts of the CSC200 by using electrical tape to cover them.

If the termination resistor connections (brown and green wires) are not used on the FTDI Chip cable, it may be necessary to connect these two wires to the same “Isolated GND” ground terminal that the black ground wire is connected to. This prevents these wires from “floating” and potentially propagating noise down the RS485 cable.

Example Cable Connection – SCADAPack PLC Master

Figure 6 - CAT5E cable used

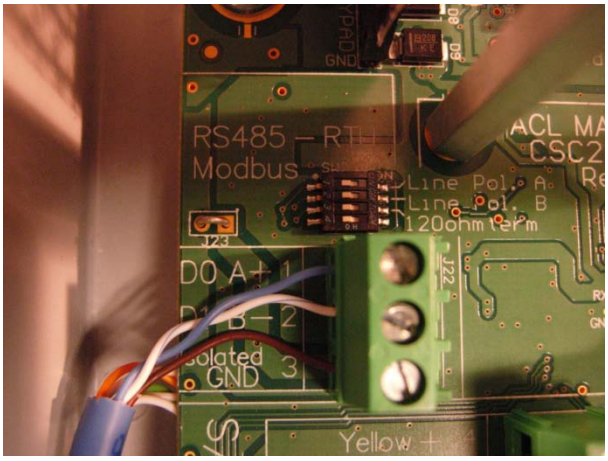


Figure 7 – SCADAPack 100

(SCADAPack 100 picture courtesy http://www.controlmicrosystems.com)

Figure 8 – SCADAPack 100 with CAT5E Cable Attached to COM1 (RS485 capable serial port)

Modbus Communication Test Using a PC Master

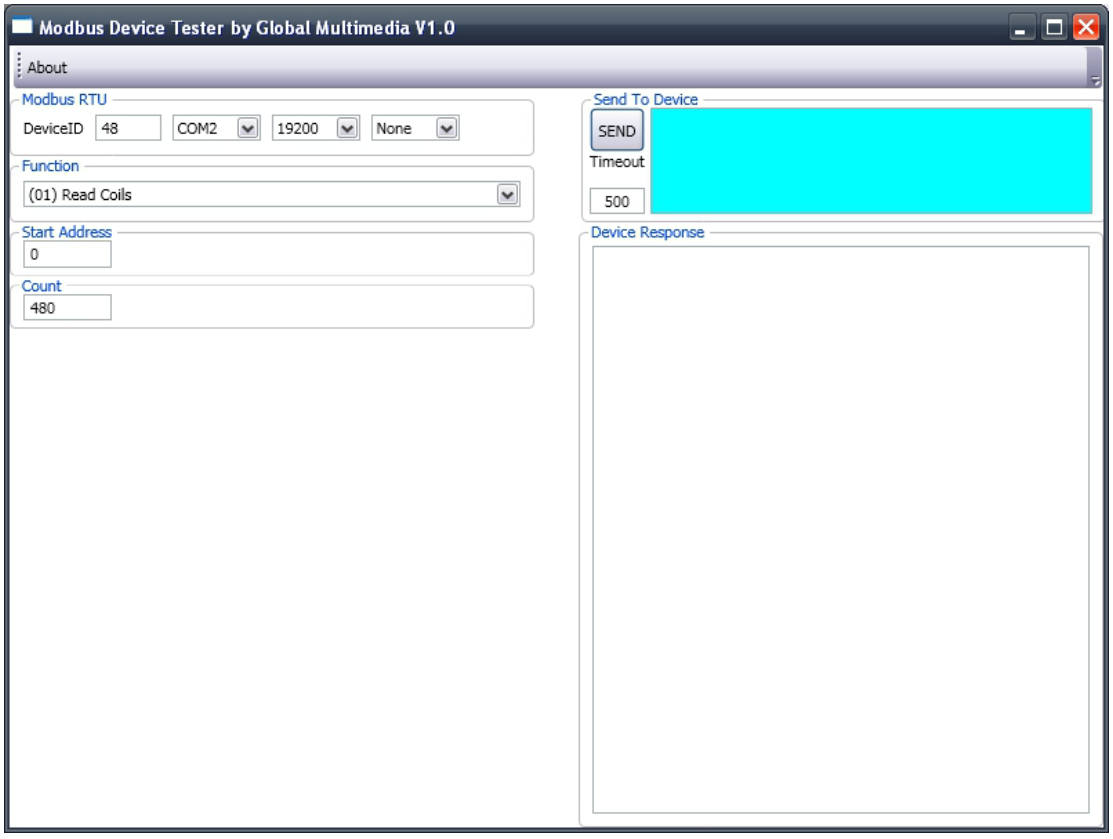
A variety of test programs are available for the PC for testing Modbus communications. A few are listed below:

PC Test Software	Company	License Type	Website
Modnet for Modbus RTU	Global Multimedia Private Ltd	Freeware	http://www.globalmultimedia.in/modnet.htm
Modbus Poll	Modbus Tools	30-day trial	http://www.modbustools.com/modbus_poll.asp
Modbus Constructor	KurySoft	30-day trial	http://www.kurysoft.com/products.shtml
Modbus Reader	KurySoft	Freeware	http://www.kurysoft.com/products.shtml

Additional technical resources for modbus can be found at the official Modbus Organization website: http://www.modbus.org/tech.php

The following procedure uses the PC software Modnet for Modbus RTU. The testing was done using a PC running Windows XP SP3 32-bit.

Figure 9 - PC Software Modnet for Modbus RTU



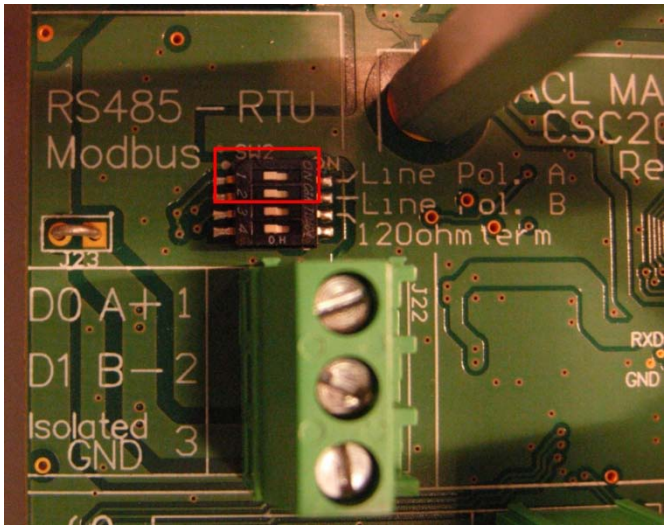
The test cable used was the FTDI Chip USB-RS485-WE-5000-BT, 5m, USB to RS-485 cable.

Figure 10 - FTDI Chip USB-RS485-WE-5000-BT, 5m, USB to RS-485 Cable

Test Preamble

For this test, the two "Line Polarization" DIP switches on the CSC200 Controller were turned ON, by moving them to the right (towards the "Line Pol…" text) as shown in the picture below (Figure 11).

Figure 11 – Line Polarization DIP Switches



"Line Polarization" enables a pullup resistor on the "Data A +" signal and a pulldown resistor on the "Data B –" signal. It ensures that the bus is put into a known state with the "Data A +" signal High and the "Data B -" signal Low.

Line Polarization should only be enabled on one device on the RS485 bus.

The PC software Modnet for Modbus RTU will work with Line Polarization on or off but there's a small difference in the response data received from the CSC200 using this software. Referring to the pictures below (Figure 12 and Figure 13), the data received shows an extra "0x00" at the beginning and at the end of the response packet.

Figure 12 - Modbus Response Data From CSC200 with Line Polarization ON

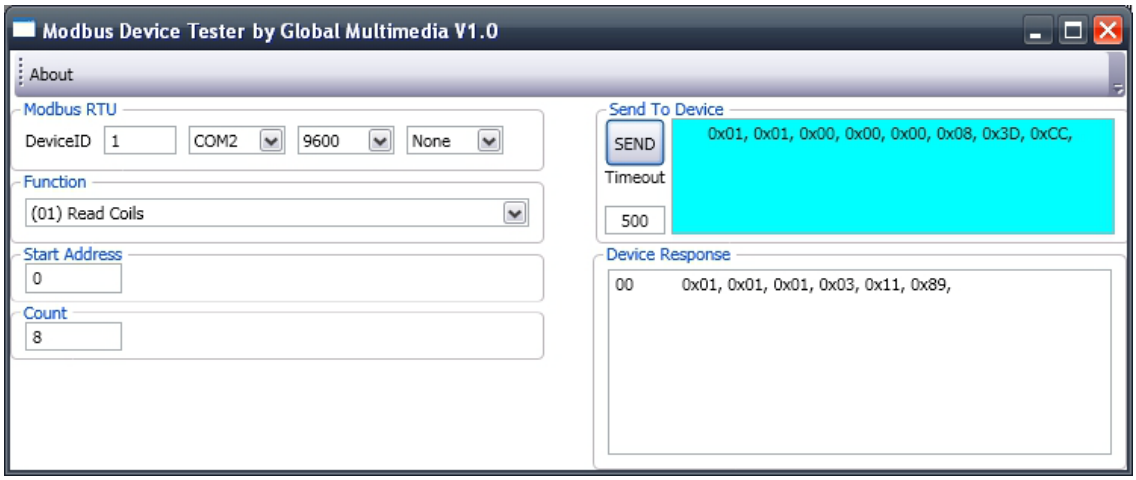
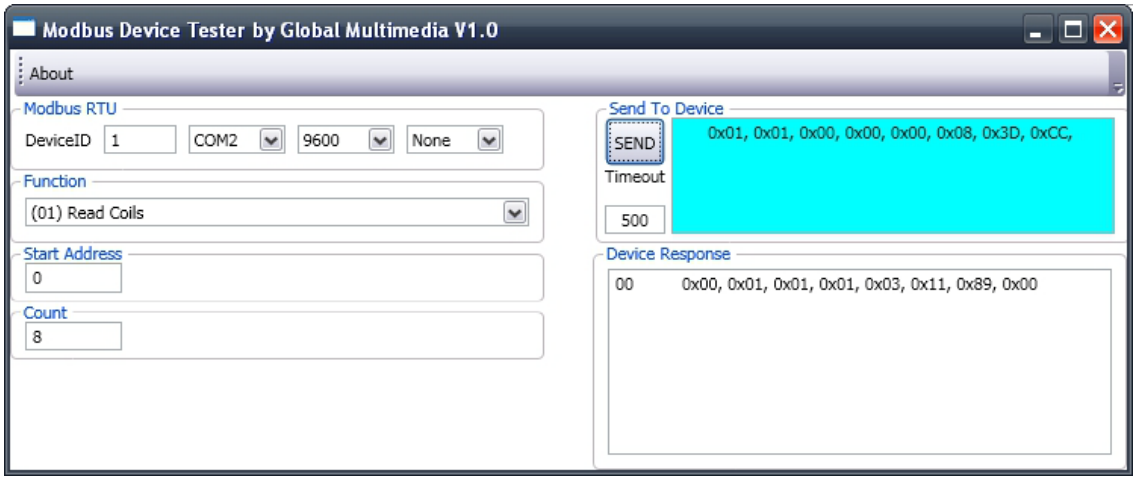


Figure 13 - Modbus Response Data From CSC200 with Line Polarization OFF

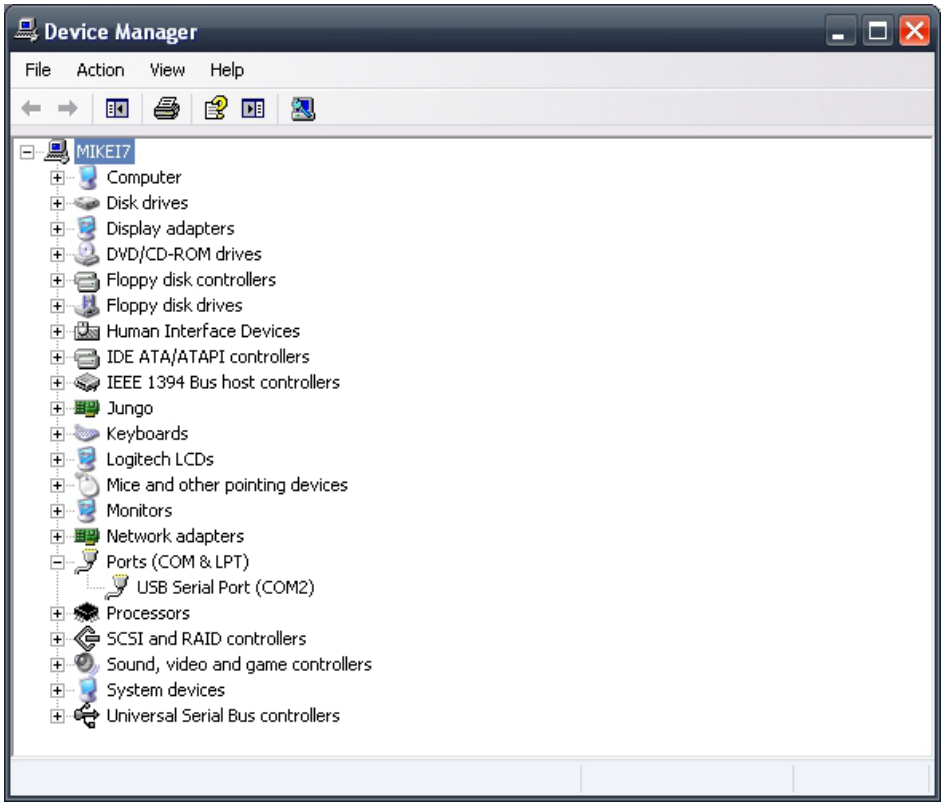


Test Procedure

1) Connect a USB-to-RS485 cable between the CSC200 Controller and the PC (refer to "Figure 5 - …. USB to RS485 Cable, Installed with CSC200 Controller" for details).

2) Ensure that the driver software for the USB to RS485 cable is installed and that the cable shows up as a virtual COM port in the Device Manager:
   - Press and hold the Left Windows Key, then press the Pause/Break key to display the System Properties window.
   - Click on the "Hardware" tab, then click on the "Device Manager" button. You should see the Device Manager window open, similar to the window shown in Figure 14.

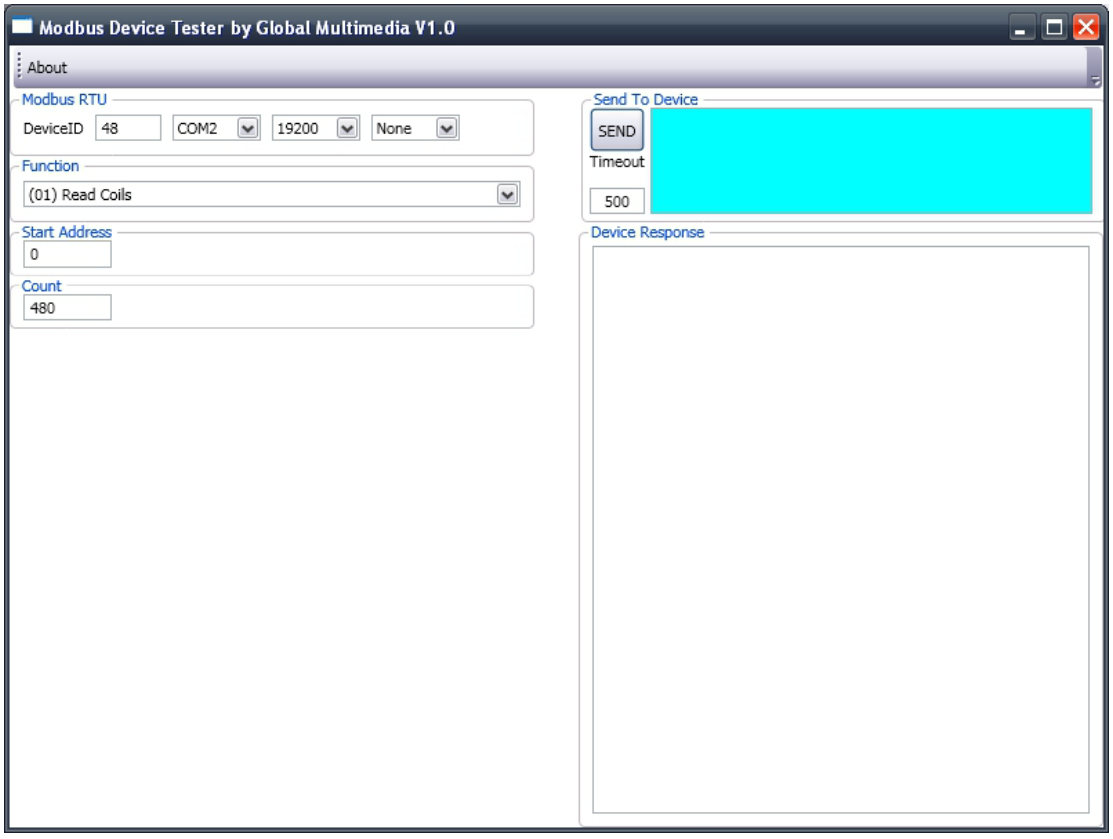
Figure 14 – Windows XP's Device Manager Window



"USB Serial Port (COM2)" is shown under "Ports (COM & LPT)"

3) Run the PC software Modnet for Modbus RTU. You should see a window similar to the one pictured in Figure 15 below.

Figure 15 - Modnet for Modbus RTU Software

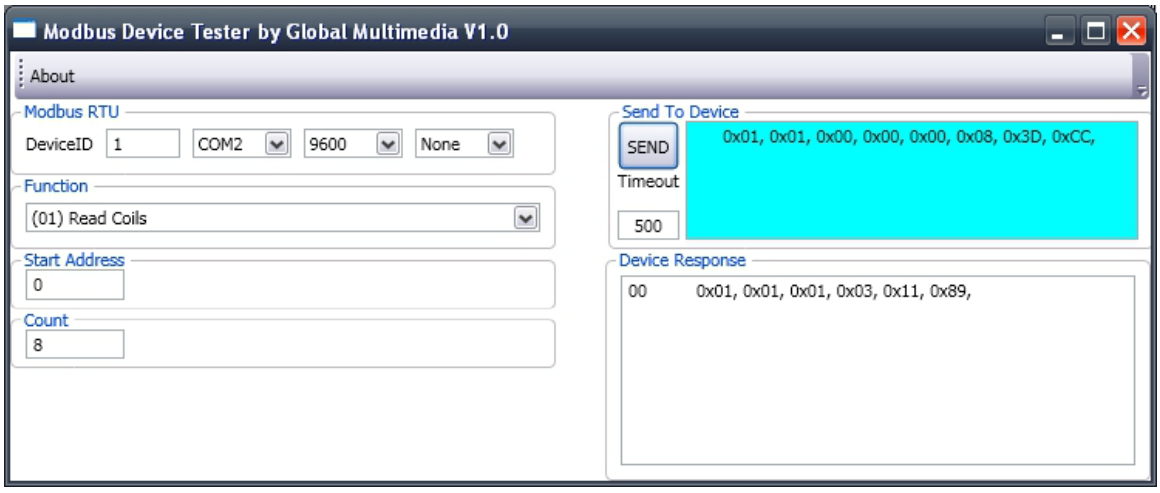


4) Change the default settings of the software to the following:
- Change DeviceID from “48” to “2”
- Make sure the same COM port shown in the Device Manager is selected here in the Modnet software (“COM2” in this case)
- Change the baud rate from “19200” to “9600”
- Change “Count” from “480” to “8”
- Leave all other settings at the defaults

5) Apply power to the CSC200. Wait for it to progress through its startup sequence. Once it shows a temperature on the LED display, proceed to the next step.

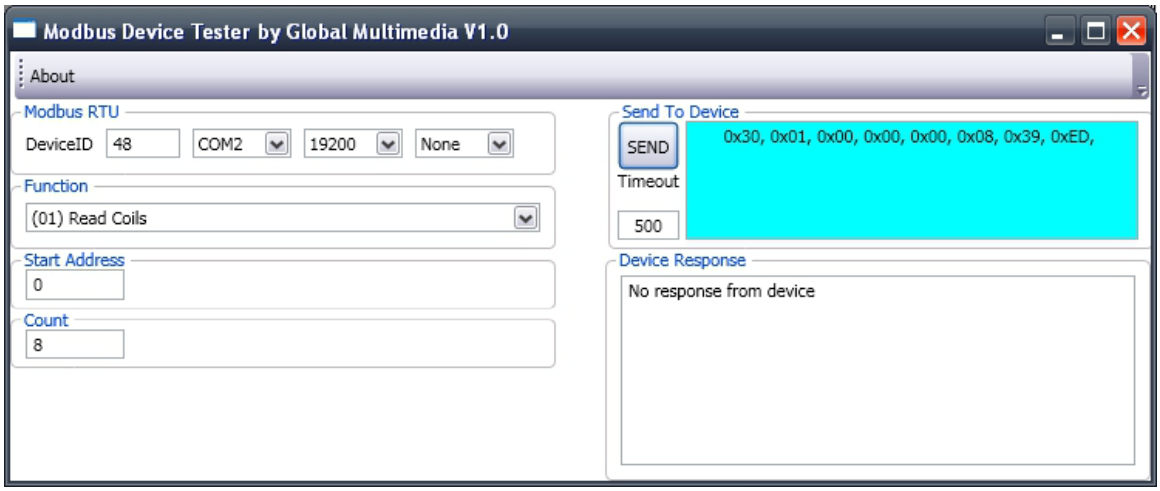
6) Click on the “Send” button. You should see a response from the CSC200 similar to the one shown in Figure 16.

Figure 16 – CSC200 Modbus Response to “Read Coils” Function Code



If you received no response, you will see something similar to the picture in Figure 17. (Notice that the settings weren’t changed after first running the modnet software).

Figure 17 – Result from No Response to “Read Coils” Function Code



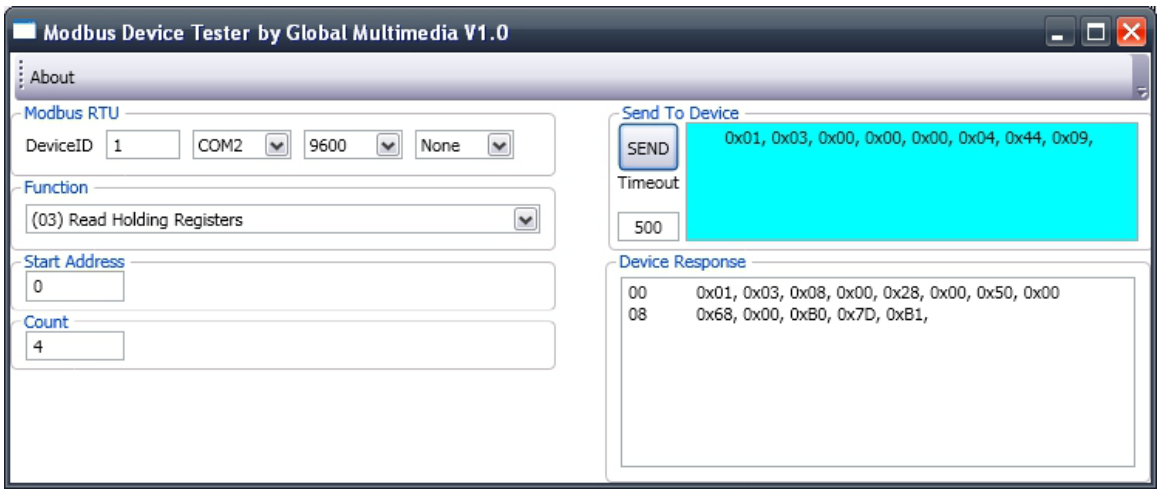
7) If a response was received where the first three bytes are all “0x01” (as seen in Figure 16), then the CSC200 Controller Modbus connections have been installed properly. You may proceed to the section titled “Programming a New Modbus Slave ID (Address)” to change the modbus Slave ID to the desired ID before installing this CSC200 in a field installation. Alternatively, you may also continue on to step 8, performing additional tests using the Modnet for Modbus software.

If no response was received, double-check the wiring connections and the serial port settings. The default serial settings for the CSC200 Rev 2A Controller are 9600 baud, 8 data bits, no parity bits, and one stop bit. The default modbus Slave ID (address) is “2”.

8) Change the settings of the modnet software to the following:
- Change the “Function” to “(03) Read Holding Registers”
- Change “Count” to “4”
- Leave all other settings as they are

9) Click on the “Send” button. You should see a response from the CSC200 similar to the one shown in Figure 18.

Figure 18 - CSC200 Modbus Response to “Read Holding Registers” Function Code



The response shows four 16-bit values returned which are the current setpoint temperatures in degrees Celsius and Fahrenheit :

Value in Hex	Value in decimal	Description
0x0028	40	TC1 temp setpoint (deg C)
0x0050	80	TC2 temp setpoint (deg C)
0x0068	104	TC1 temp setpoint (deg F)
0x00B0	176	TC2 temp setpoint (deg F)

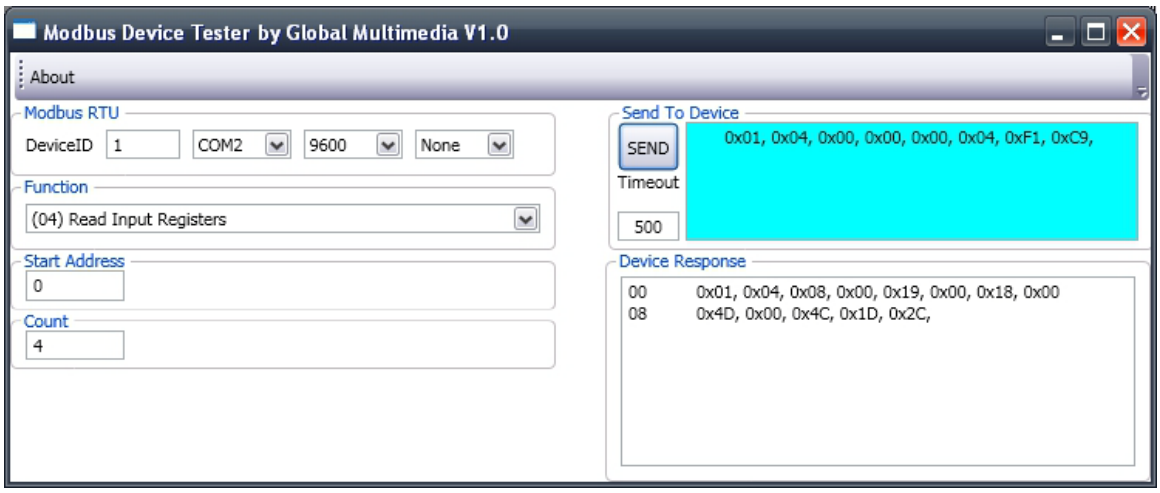
These values correspond to the current setpoint temperatures that we are viewing on the CSC200’s LED display, so we know that the modbus communication was successful.

Note that the last two bytes are for CRC (cyclic redundancy check) error checking.

10) Change the settings of the modnet software to the following:
- Change the “Function” to “(04) Read Input Registers”
- Leave all other settings as they are

11) Click on the “Send” button. You should see a response from the CSC200 similar to the one in Figure 19.

Figure 19 - CSC200 Modbus Response to “Read Input Registers” Function Code



The response shows four 16-bit values returned which are the current temperatures measured by the two thermocouples in degrees Celsius and Fahrenheit :

Value in Hex	Value in decimal	Description
0x0019	25	TC1 current temp (deg C)
0x0018	24	TC2 current temp (deg C)
0x004D	77	TC1 current temp (deg F)
0x004C	76	TC2 current temp (deg F)

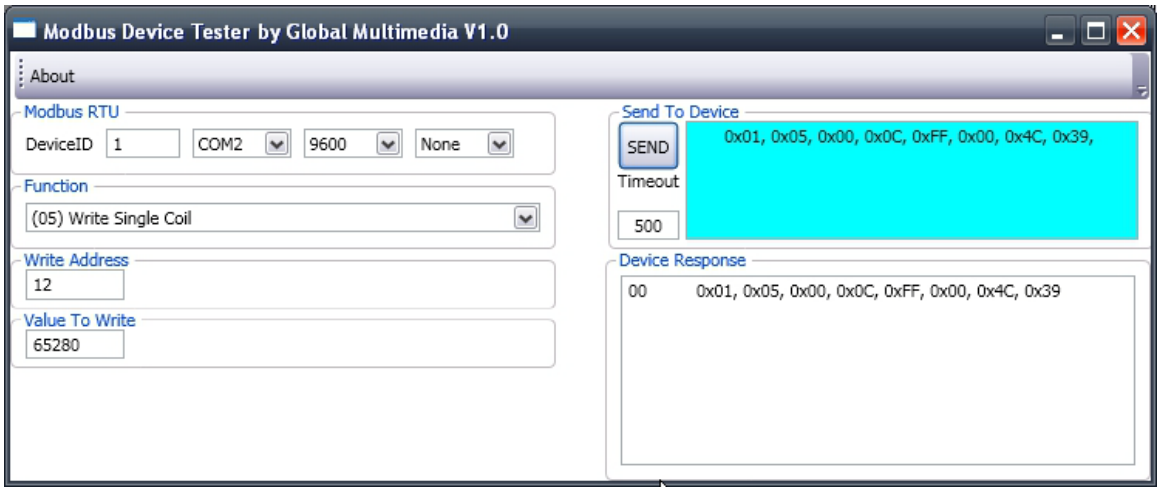
These values correspond to the current measured temperatures that we are viewing on the CSC200’s LED display, so we know that the modbus communication was successful.

Note that the last two bytes are for CRC (cyclic redundancy check) error checking.

12) Change the settings of the modnet software to the following:
- Change the “Function” to “(05) Write Single Coil”
- Change the “Write Address” field to 12
- Change the “Value To Write” field to 65280 (0xFF00)
- Leave all other settings as they are

13) Click on the “Send” button. You should see a response from the CSC200 similar to the one in Figure 20.

Figure 20 - CSC200 Modbus Response to “Write Single Coil” Function Code



This function writes 0xFF00 to the “Coil” address for issuing a Remote Stop command to the CSC200. If the CSC200 was turned on, this command should’ve turned off the CSC200.

This test procedure showed how a variety of commands could be sent and received to the CSC200 using a PC with a USB-to-RS485 cable and the modnet for modbus diagnostic test software.

Modbus Communication Test Using a SCADAPack 100 PLC and Telepace Studio

A SCADAPack 100:1024k was used for testing Modbus communications between a PLC and the CSC200.

Telepace Studio version 5.0.3 from Schneider Electric/Control Microsystems Inc. was used for testing. Due to the size of the sample project used for this test procedure, a full license of Telepace Studio was required for this test. Smaller projects have been created to test individual commands sent to the CSC200 from the SCADAPack 100 using the evaluation version of Telepace Studio.

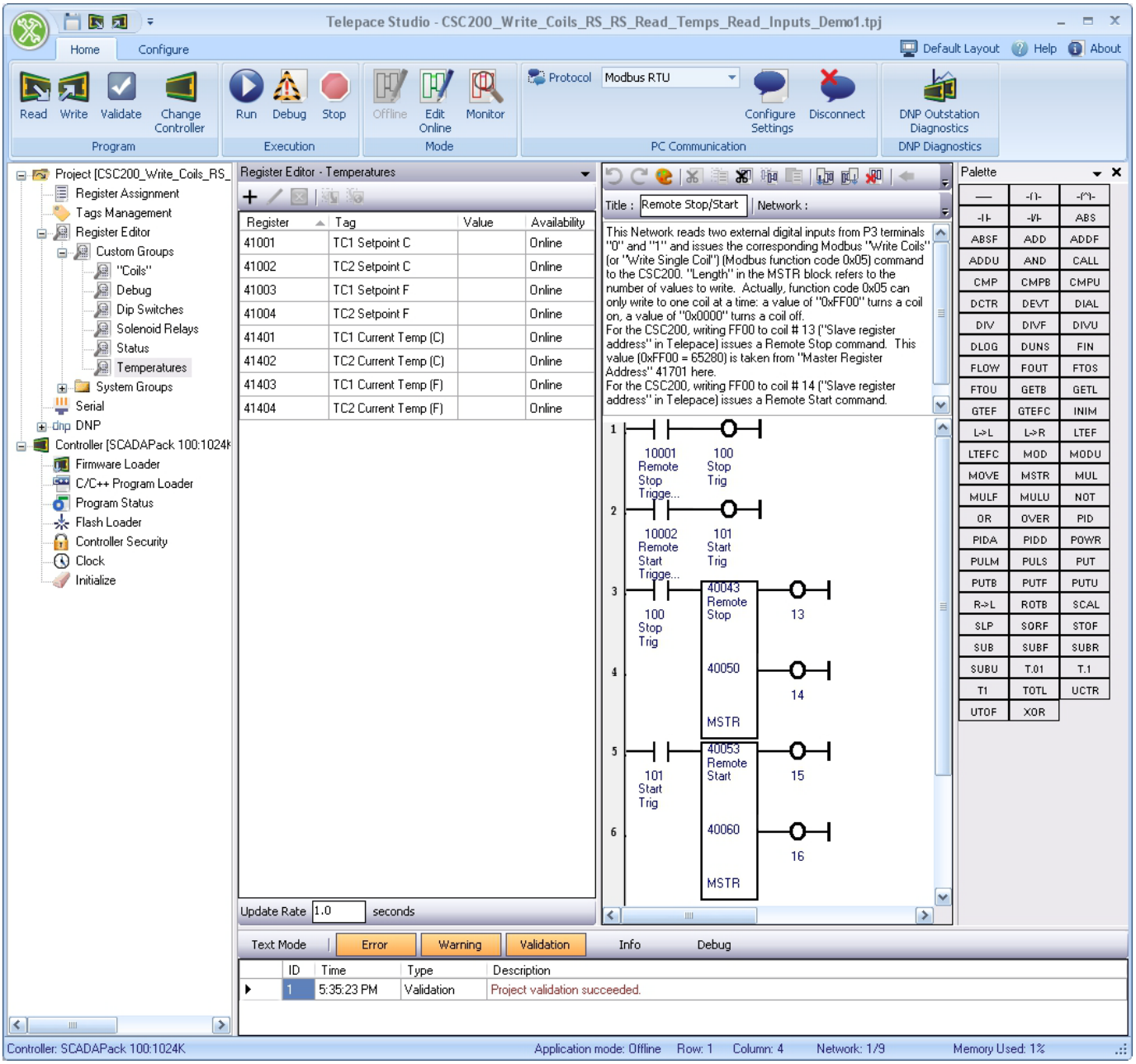
Figure 21 - SCADAPack 100 With Attached CAT5E Cable

The Telepace project used demonstrates all the commonly needed information read back from the CSC200. It periodically polls the CSC200 to read the coil status, the input registers, the discrete inputs, and some of the holding registers.

A SCADAPack 100:1024k Controller was used for this demonstration.

A CAT5E test cable between the SCADAPack and the CSC200 Controller was attached. Refer to Appendix C for additional details.

Figure 22 - Communication Test Demonstration Using a SCADAPack and Telepace Studio



Appendix E - Programming a New Modbus Slave ID (Address)

The default Modbus Slave ID for a new CSC200 is "2".

Summary

#	Command to Perform (Modbus Function Code)	SCADAPack Register Address	Register Number	Modbus Holding Register Address	Register Description	Value to Write
1	Write Single Holding Register	40005	5	4	Unlock Slave ID register	0x55AA (21930)
2	Write Single Holding Register	40006	6	5	Slave ID register	New Desired Slave ID (0x0001 to 0x00F7)

Procedure When Using a PC Master to Change the Modbus Slave ID (Address)

1) Connect one end of a USB to RS-485 cable to the three screw terminals of the CSC200 Controller (refer to Appendix D for details if necessary). Connect the USB end to a PC. This CSC200 should be the only device attached to the RS-485 bus while changing the Slave ID (address) to avoid potential conflicts.

2) Run the desired Modbus Master software (examples are Modnet for Modbus or Modbus Constructor) and connect to the COM port used by the USB-to-RS485 cable. Default serial settings for the CSC200 are 9600 baud, 8N1, Modbus RTU.

3) Select the unique Slave ID for the CSC200 to communicate to (default Slave ID for a new CSC200 is "2"). Issue a Write Single Holding Register command to Modbus Holding Register Address 4 ("Unlock Slave ID register") using the value 0x55AA (21930). This command unlocks the Slave ID for changing it. This is used as a safety precaution to prevent inadvertent Slave ID changing.

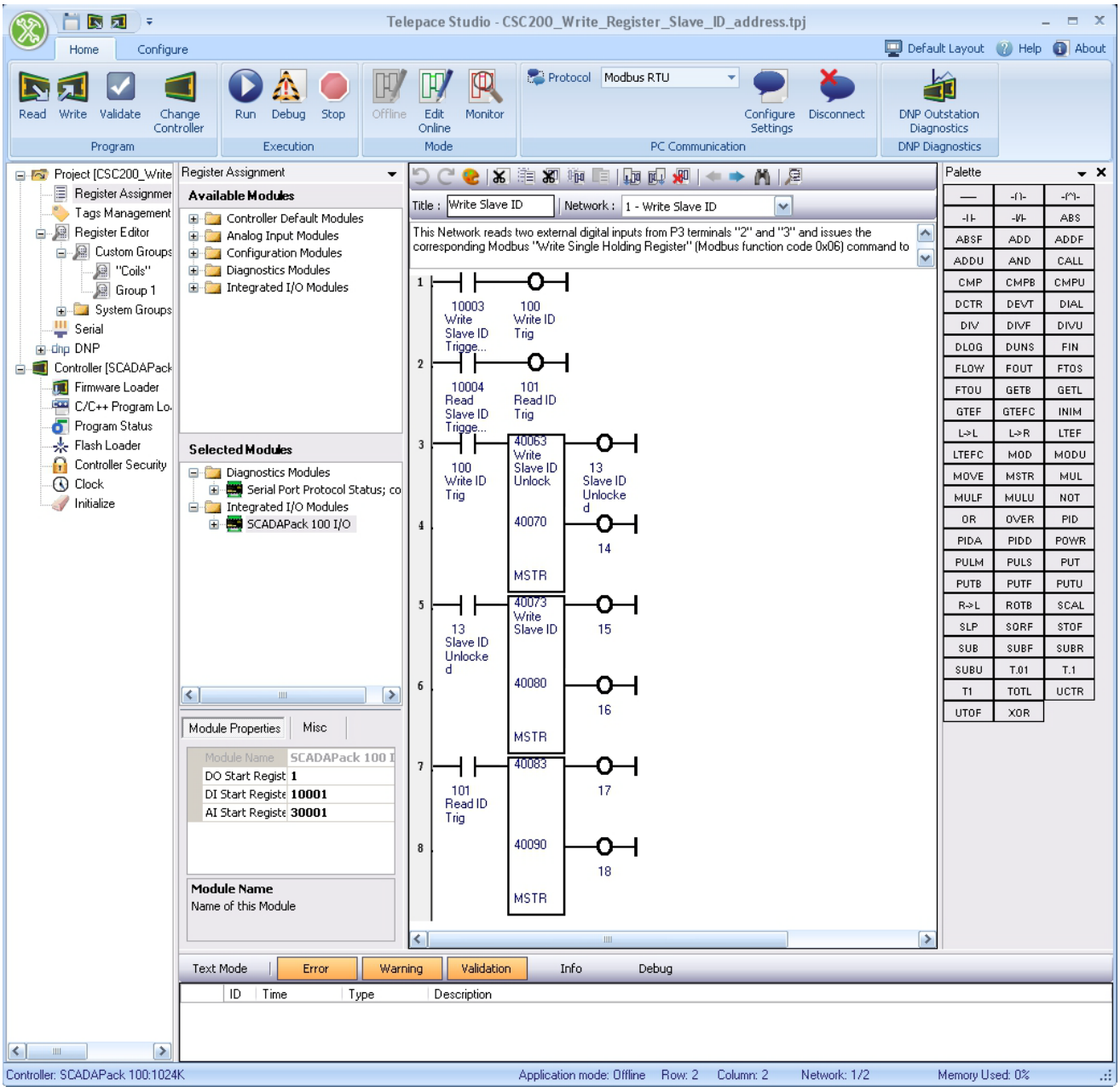
General Modbus Command:			
Command to Write	Modbus Function Code	Write Address	Value to write
Write Single Holding Register	0x06	4 ("Unlock Slave ID register")	0x55AA (21930)
Modnet for Modbus RTU Software:			
Function		Write Address	Value to write
(06) Write Single Register		4 ("Unlock Slave ID register")	21930

4) Issue a Write Single Holding Register command to Modbus Holding Register Address 5 ("Slave ID register") using the new desired Modbus Slave ID (address) that you want to assign to this CSC200. Values between 0x0001 and 0x00F7 are allowed. Note that the Modbus specification says that at least 32 Modbus devices can reside on one RS-485 bus (without repeaters). Testing needs to be done by the installer to ensure adequate signal integrity if more than 32 devices are placed on one Modbus RS-485 bus.

General Modbus Command:			
Command to Write	Modbus Function Code	Write Address	Value to write
Write Single Holding Register	0x06	5 ("Slave ID register")	Desired Modbus address value between 0x0001 and 0x00F7 (between 1 and 247)
Modnet for Modbus RTU Software:			
Function		Write Address	Value to write
(06) Write Single Register		5 ("Slave ID register")	Desired Modbus address value between 1 and 247 (between 0x0001 and 0x00F7)

Sample Project When Using a SCADAPack PLC to Change the Modbus Slave ID (Address)

Figure 23 - Changing the Modbus Slave ID Using a SCADAPack and Telepace Studio



A Telepace Studio sample project was created to write the Unlock Slave ID value, then the new Slave ID to use. It is used as an example of ladder logic programmable control of the CSC200's specific Slave ID.

Referring to Figure 23 above, it can be seen that this ladder logic network utilizes two external trigger inputs: one at SCADAPack address 10003 to trigger the writing of the Unlock Slave ID value and the writing of the new Slave ID, and the other at SCADAPack address 10004 to trigger a read back of the ID using the new Slave ID just written.

A second network (not shown) was used to store two unsigned values in the SCADAPack: one holding the Unlock Slave ID value to write (0x55AA or 21930), and the other holding the new Slave ID to write.

Appendix F - PC Communication Test Demonstration: Modbus Reader Software

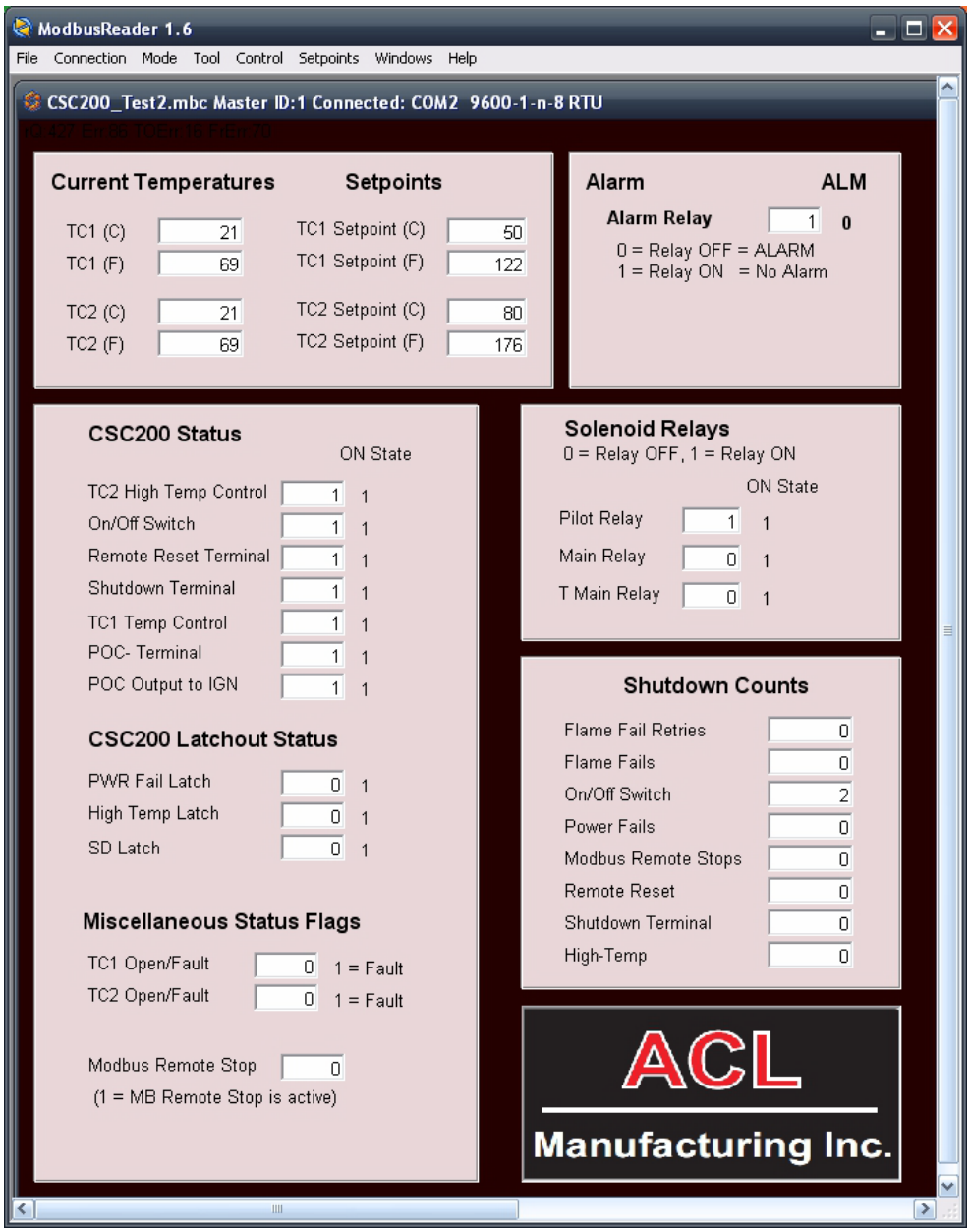
The program Modbus Constructor (includes Modbus Reader) was used to generate the project shown below in Figure 24. The license for Modbus Constructor is a 30-day free trial. It can be downloaded here http://www.kurysoft.com/products.shtml

Modbus Reader can be downloaded for free from http://www.kurysoft.com/products.shtml

1) This project demonstrates all the commonly needed information read back from the CSC200. It periodically polls the CSC200 to read the coil status, the input registers, the discrete inputs, and some of the holding registers.

2) The serial settings can be changed in the "Connection" menu then "COM Parameters" sub menu. The Slave ID (address) to communicate with can be changed in the menu item "Mode" --> "Master Settings".

Figure 24 - CSC200_Test3.mbc Modbus Reader Sample Project



Appendix G - Modbus/RS-485 References

The Modbus protocol specification can be viewed here http://www.modbus.org/specs.php
"Modbus Protocol Specification", filename "Modbus_Application_Protocol_V1_1b.pdf")

The Modbus serial line protocol and implementation guide can be viewed here http://www.modbus.org/specs.php
"Modbus Serial Line Protocol and Implementation Guide", filename "Modbus_over_serial_line_V1_02.pdf")

Additional technical resources for modbus can be found at the official Modbus Organization website:
http://www.modbus.org/tech.php

Appendix H - Troubleshooting

#	Issue	Possible Reason	Corrective Action
1	Modbus Master can't read temperature values from CSC200 (or any other data)	RS485 cable isn't connected properly	Ensure the wires for the RS485 cable are connected properly at the CSC200 and at the master and that the screw terminals are gripping the metal wire, not the insulation. Wires may also become damaged with frequent bending or if they've been pinched. Ensure that the RS485 signal wires haven't been broken by testing continuity.
		Modbus Slave ID (address) is different than the address used for the CSC200	Verify that the address used by the master to communicate with the CSC200 matches the address set in the CSC200. Try using the default address: "2". The master may need to poll a variety of modbus addresses (from 1 to 247) to find slaves that respond.
		Power to the CSC200 may have been interrupted	Verify the CSC200 has power locally.
		Inappropriate, non-twisted pair cable has been used for the RS485, for long distances	Ensure that an appropriate twisted-pair cable (like CAT5e cable, or other appropriate cable) is used for the RS485 bus.
2	Modbus communication interrupted, noise issues suspected	Inadequate or ineffective grounding	Ensure that an adequate connection has been made between the earth ground terminal on the CSC200 and an appropriate earth ground external to the CSC200 (eg: thick spike in the ground, underground water pipes, earth ground pin on an AC wall outlet).
			Ensure that unused, non-power sourcing wires in any RS485cable are grounded.
			Connect the "Isolated GND" terminal on the CSC200 Controller to the CSC200 earth ground terminal to provide a localized ground path for noise. (Attach GND jumper on Rev 2B cards and later)
		Power to the CSC200 may have been interrupted	Verify the CSC200 has power locally.
3	Modbus PC Master communication with CSC200 interrupted	If a USB-to-RS485 conversion cable has been used, the PC test software may have lost connection to the virtual COM port, or noise may have interfered with USB communications.	Unplug the USB-to-RS485 conversion cable from the USB port on the PC, wait 10 seconds, then plug it back in. Retry connecting to the COM port in the test software.
			Add an industrial-rated USB hub between the PC and the USB-to-RS485 cable. Ensure that the hub is powered locally, not bus-powered from the PC.
			Refer to Troubleshooting item # 2 for additional grounding notes
		Power to the CSC200 may have been interrupted	Verify the CSC200 has power locally.
4	Modbus communication works for writing Remote Stop, Remote Start, but no values are being read back	CSC200's ignition module may be "sparking".	The CSC200 will not respond to Modbus requests when the ignition module is powering its high-voltage sparker to ignite the Pilot.
		If a USB-to-RS485 conversion	Unplug the USB-to-RS485 conversion cable from the

		cable has been used, the PC test software may have lost connection to the virtual COM port, or noise may have interfered with USB communications.	USB port on the PC, wait 10 seconds, then plug it back in. Retry connecting to the COM port in the test software.

Notes:

Notes:

Notes: