



RS-485 Modbus Using the 5300 with an Eaton MVX9000 Drive

The information in this document is current as of the following Hardware and Firmware revision levels. Some features may not be supported in earlier revisions. See www.ctc-control.com for the availability of firmware updates or contact CTC Technical Support.

Model Number	Hardware Revision	Firmware Revision
BC5311-01B	All	All



WARNING: Use of CTC Controllers and software is to be done only by experienced and qualified personnel who are responsible for the application and use of control equipment like the CTC controllers. These individuals must satisfy themselves that all necessary steps have been taken to assure that each application and use meets all performance and safety requirements, including any applicable laws, regulations, codes and/or standards. The information in this document is given as a general guide and all examples are for illustrative purposes only and are not intended for use in the actual application of CTC product. CTC products are not designed, sold, or marketed for use in any particular application or installation; this responsibility resides solely with the user. CTC does not assume any responsibility or liability, intellectual or otherwise for the use of CTC products.

The information in this document is subject to change without notice. The software described in this document is provided under license agreement and may be used and copied only in accordance with the terms of the license agreement. The information, drawings, and illustrations contained herein are the property of Control Technology Corporation. No part of this manual may be reproduced or distributed by any means, electronic or mechanical, for any purpose other than the purchaser's personal use, without the express written consent of Control Technology Corporation.

TABLE OF CONTENTS

Overview.....	5
Wiring the RS-485 Ports.....	6
Addressing	8
Eaton MVX-9000 Drive Settings.....	10
CTC 5300 Settings.....	11
RS-485 Communications Settings.....	11
Communications Registers for Serial Settings	11
5300 Modbus Master Settings	12
Modbus Register Overview	12
Writing to the Eaton Drive Parameters.....	14

Blank Page

Overview

This document shows you how to setup communications between a CTC 5300 Controller and an Eaton MVX900 drive over RS-485. The BC5311-01B has an RS-485 port that can be accessed via communications port 3. The BC5311-01A only offers RS-232 communications and would require an RS-232 to RS-485 convertor.

We will be discussing the use of a BC5311-01B in this Tech Note.

It is assumed you have a basic understanding of QuickBuilder. If you do not please refer to the following manuals first:

Quick Builder QuickStart Guide

http://www.ctc-control.com/customer/techinfo/docs/5300_951/951-530030.pdf

5300 Quick Register Guide

http://www.ctc-control.com/customer/techinfo/docs/5300_951/951-530006.pdf

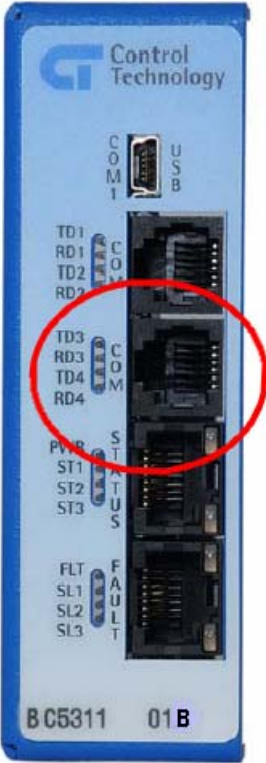
5300 Enhancements Guide

http://www.ctc-control.com/customer/techinfo/docs/5300_951/951-530001.pdf

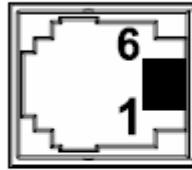
Wiring the RS-485 Ports

The RS-485 port on the BC5311-01B is accessed via communication port 3 shown below:

CPU module

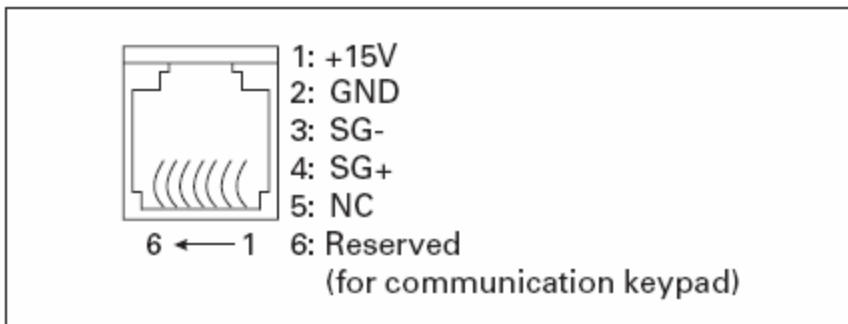


COM3 and COM4 RS232/RS485 pinouts



Pin #	Signal
1	TxD COM4
2	TXD COM3/A (+RS485)
3	Common
4	Common
5	RXD COM3/B(-RS485)
6	RXD COM4

The RS-485 port on the Eaton MVX-9000 is accessed via a RJ-12 Jack at the top of the drive near the Input power. The pin configuration is shown below:

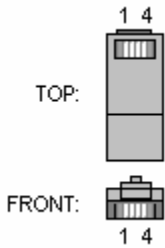


Since Pins 1 and 6 are not used on either side of the cable a 4 pin RJ-11 connector can be used.

Modbus RS-485 Communications Between 5300 and Eaton MVX9000

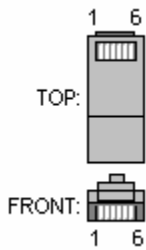
The cable pin out between the CTC BC5311-01B and the Eaton MVX9000 would be as follows:

Using 4 Pin RJ-11 connectors (male connector configuration shown below):



	CTC	Eaton
Pin	1 B/485-	3 SG-
	4 A/485+	2 SG+
	2 Common	1 GND

Using 6 Pin RJ-12 connectors (male connector configuration shown below):



	CTC	Eaton
Pin	2 B/485-	4 SG-
	5 A/485+	3 SG+
	3 Common	2 GND

Note that you may find it necessary to add termination resistor(s) depending on your environment and the length of your cable. Refer to RS-485 network specifications for more information on this.

Addressing

Addressing between the CTC and Eaton drive works as follows:

You will select a start address of Parameters you want to get from the Eaton Drive and then select how many consecutive Parameters you would like.

For example you may want to access all 9 of the parameters from the Basic Grouping:

MVX9000 Parameter Listing						
Table B-1: 20 — BASIC GROUPING (Quick Start)						
Modbus	Groups	Page #	Description	Range	Default	User Settings
0000H	20.01 50.05	5-5	Motor Nameplate Frequency	10.0 to 400.0 Hz	60.0	
0001H	20.02 50.06	5-5	Motor Nameplate Voltage	115/230V drives: 1.0 to 255V	230	
				460V drives: 1.0 to 510V	460	
				575V drives: 1.0 to 637V	575	
0002H	20.03 50.01	5-5	Source of Master Frequency	00: Master frequency determined by digital keypad up/down	01	
				01: Master frequency determined by keypad potentiometer		
				02: Master frequency determined by 0 to +10V input on AI terminal		
				03: Master frequency determined by 4 to 20 mA input on AI terminal		
				04: Master frequency determined by RS-485 communication interface		
0003H	20.04 50.02	5-5	Source of Operation command	00: Operation commands determined by digital keypad	00	
				01: Operation commands determined by external control terminals, keypad STOP is effective		
				02: Operation commands determined by external control terminals, keypad STOP is ineffective		
				03: Operation commands determined by RS-485 interface, keypad STOP is effective		
				04: Operation commands determined by RS-485 interface, keypad STOP is ineffective		
0004H	20.05 60.01	5-6	Motor Rated Current	30 to 120%	FLA	
0005H	20.06 50.09	5-6	Minimum Output Frequency	0.0 to 20.0 Hz	1.5	
0006H	20.07 50.04	5-6	Maximum Output Frequency	50.0 to 400.0 Hz	60.0	
0007H	20.08 50.12	5-6	Acceleration Time 1	0.01 to 600.0 sec	10.0	
0008H	20.09 50.13	5-6	Deceleration Time 1	0.01 to 600.0 sec	10.0	

Modbus RS-485 Communications Between 5300 and Eaton MVX9000

In this case the start address of 0000H from the MVX9000 would be considered 1 to CTC since our addressing always starts at one. The number of parameters or sequential registers to get would be 9.

If you wanted to get 20 of the Drive Control Parameters (shown below) you would use a start address of 769. 0300H is a hexadecimal number. This needs to be converted to decimal, which is 768. The final value of 769 is due to the offset of 1 because the MVX addressing starts at zero and the 5300 starts at 1. The number of sequential registers would be 20.

Table B-4: 50 — DRIVE CONTROL

Modbus	Groups	Page #	Description	Range	Default	User Settings
0300H	50.01	5-31	Source of Master Frequency	00: Master Frequency determined by digital keypad up/down	01	
				01: Master Frequency determined by keypad potentiometer		
				02: Master Frequency determined by 0 to +10V input on AI1 terminal		
				03: Master Frequency determined by 4 – 20 mA input on AI2 terminal		
				04: Master Frequency determined by RS-485 communication interface		
0301H	50.02	5-31	Source of Operation Command	00: Operation command determined by digital keypad	00	
				01: Operation command determined by external control terminals, keypad STOP is effective		
				02: Operation command determined by external control terminals, keypad STOP is ineffective		
				03: Operation command determined by RS-485 interface, keypad STOP is effective		
				04: Operation command determined by RS-485 interface, keypad STOP is ineffective		
0302H	50.03	5-31	Stop Methods	00: Ramp to Stop	00	
				01: Coast to Stop		
0303H	50.04	5-32	Maximum Output Frequency	50.0 to 400.0 Hz	60.0	
0304H	50.05	5-32	Motor Nameplate Frequency	10.0 to 400.0 Hz	60.0	
0305H	50.06	5-32	Motor Nameplate	115/230V 0.1 to 255.0V	230.0	

Eaton MVX-9000 Drive Settings

When setting the communication parameters it is important to make sure the RS-485 settings are compatible between the two devices.

Modbus	Groups	Page #	Description	Range	Default	User Settings
0700H	90.01	5-63	Communication Protocol	00: MODBUS ASCII mode < 7 data bits, no parity, 2 stop bits >	00	
				01: MODBUS ASCII mode < 7 data bits, even parity, 1 stop bit >		
				02: MODBUS ASCII mode < 7 data bits, odd parity, 1 stop bit >		
				03: MODBUS RTU mode < 8 data bits, no parity, 2 stop bits >		
				04: MODBUS RTU mode < 8 data bits, even parity, 1 stop bit >		
				05: MODBUS RTU mode < 8 data bits, odd parity, 1 stop bit >		
0701H	90.02	5-63	RS-485 Communication	01 to d 254	01	
0702H	90.03	5-63	Transmission Speed	00: 4800 baud	01	
				01: 9600 baud		
				02: 19200 baud		
				03: 38400 baud		

Above are the basic communication parameters available on the MVX-9000.

In this example we set them as follows:

Parameter	Description	Setting
90.01	Comm. Protocol	03 : Modbus RTU mode 8 data bits, no parity 2 stop bits
90.02	Modbus Address	01 (each device in a multi-drop system must be unique)
90.03	Transmission Speed	02: 19200 Baud

We used default for the remaining settings. Refer to the MVX-9000 manual for more information on these settings.

CTC 5300 Settings

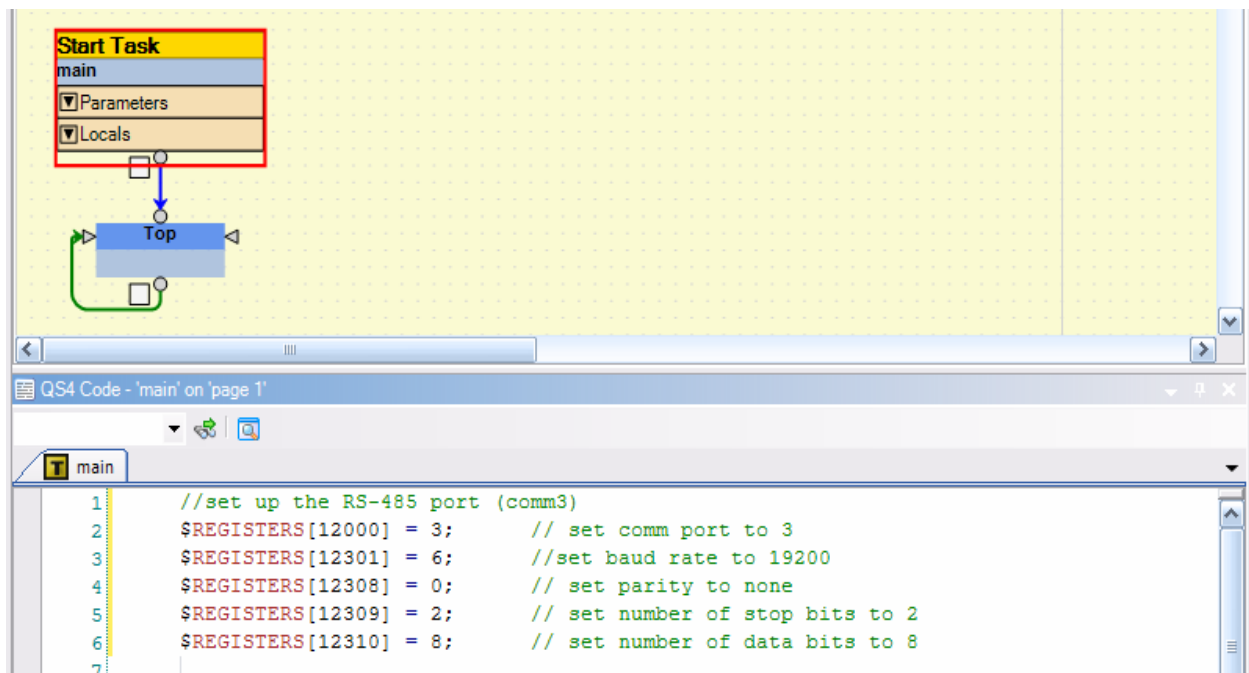
RS-485 Communications Settings

The first thing we need to do on the CTC 5300 is set-up the RS-485 parameters for port 3. The Registers used to setup communication for the serial ports are as follows:

Communications Registers for Serial Settings

REG 12000: Com port selection, 1=COM1, 2=COM2, 3 thru 7 = TCP raw socket
 REG 12000: Selected port status 0=not busy, 1=busy
 REG 12301: Baud Rate (2=1.2K, 3=2.4K, 4=4.8K, 5=9.6K, 6=19.2K, 7=38.4k)
 REG 12308: Serial Port Parity, 0=none (default), 1=odd, 2=even
 REG 12309: Serial Port Stop Bits, 1 (default), 2
 REG 12310: Serial Data Bits, 7 or 8 (default)

We can setup the registers in the beginning of the start task of our program as shown below:



For your convenience here is the same code available for Cut and Paste into your project:

```
//set up the RS-485 port (comm3)
$REGISTERS[12000] = 3; // set comm port to 3
$REGISTERS[12301] = 6; //set baud rate to 19200
$REGISTERS[12308] = 0; // set parity to none
$REGISTERS[12309] = 2; // set number of stop bits to 2
$REGISTERS[12310] = 8; // set number of data bits to 8
```

5300 Modbus Master Settings

We will now discuss the basics of setting up the 5300 as a Modbus Master.

More in-depth information on setting up the 5300 as a Modbus master can be found here:
http://www.ctc-control.com/customer/techinfo/docs/5200_951/951-520002.pdf

The 5300 controller can run numerous Modbus TCP Master connections and a single RTU/ASCII Serial connection at the same time, to differing devices, limited only by the performance desired.

Modbus Register Overview

REG 21000-21299: Modbus parameters are organized in groups of 10

- 21xx0-21xx3:** IP address of target controller
- 21xx4:** Start register in target (first register to read in target controller)
- 21xx5:** Number of sequential registers to read, 1 to 100
- 21xx6:** Poll time (time between reads) in ms. 50mS is min, 0 reads once
- 21xx7:** Status, 0=offline, 1=ok, -1=fail, -2=busy connecting, -3=busy reading
-4=busy writing, -5=timed out, -10=aborted
- 21xx8:** Index offset register, pointer to stack parameters, see address below
 - 1000 – Peer Request Time-Out
 - 1001 – Peer Request Failed
 - 1002 – Peer Request Retry Counter
 - 1003 – Protocol Index Register
 - 1004 – TCO Client Support Register
 - 1005 – Modbus Master Unit ID
 - 1006 – Modbus Master Exception
 - 1007 – Register Remapping Start (23000 – 24999)
 - 1008 – Modbus Master MAX Retries
 - 1009 – Modbus Master Retry Counter
 - 1010 – Modbus Master Timeout
 - 1011 – Modbus Master Block Size
 - 1999 – Peer Request Initiate
 - 2000 – 2099 – Peer Request Write Block
- 21xx9:** Data for the 21xx8 index pointer. Point w/ index → store data here

Since we are looking using a serial RS-485 connection we can set the 21xx0 registers to any value.

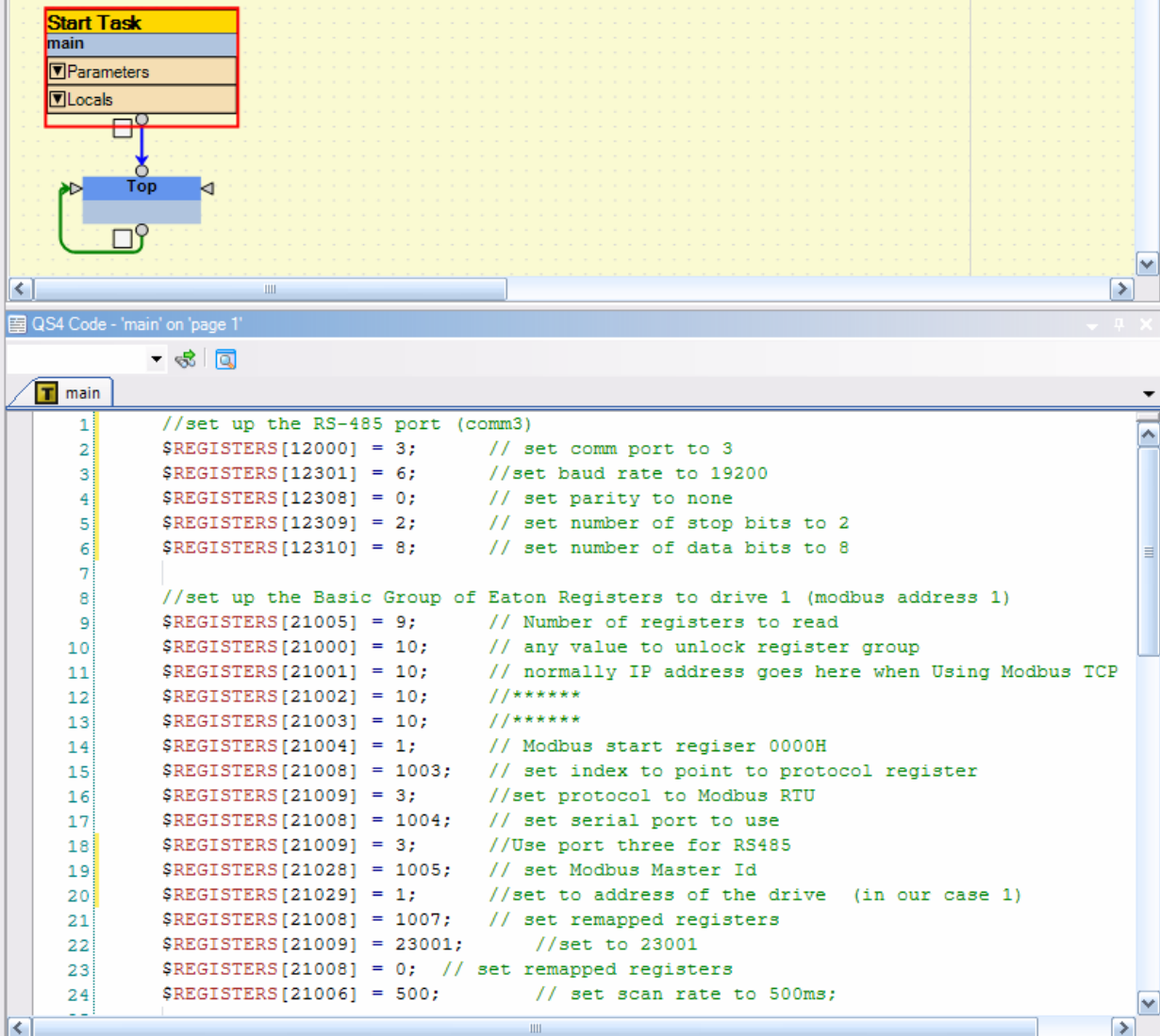
The Modbus Master Unit ID offset Register tells the Modbus Master which device ID to connect and communicate to.

The Register Remapping Start offset Register assigns which registers within the 5300 to use to access the Eaton parameters.

Modbus RS-485 Communications Between 5300 and Eaton MVX9000

For our example we will be do the following within QuickBuilder to set-up our Modbus connection:

Note that we are just adding the Modbus Register Settings right after our Communication Port Settings:



```
1 //set up the RS-485 port (comm3)
2 $REGISTERS[12000] = 3; // set comm port to 3
3 $REGISTERS[12301] = 6; //set baud rate to 19200
4 $REGISTERS[12308] = 0; // set parity to none
5 $REGISTERS[12309] = 2; // set number of stop bits to 2
6 $REGISTERS[12310] = 8; // set number of data bits to 8
7
8 //set up the Basic Group of Eaton Registers to drive 1 (modbus address 1)
9 $REGISTERS[21005] = 9; // Number of registers to read
10 $REGISTERS[21000] = 10; // any value to unlock register group
11 $REGISTERS[21001] = 10; // normally IP address goes here when Using Modbus TCP
12 $REGISTERS[21002] = 10; //*****
13 $REGISTERS[21003] = 10; //*****
14 $REGISTERS[21004] = 1; // Modbus start register 0000H
15 $REGISTERS[21008] = 1003; // set index to point to protocol register
16 $REGISTERS[21009] = 3; //set protocol to Modbus RTU
17 $REGISTERS[21008] = 1004; // set serial port to use
18 $REGISTERS[21009] = 3; //Use port three for RS485
19 $REGISTERS[21028] = 1005; // set Modbus Master Id
20 $REGISTERS[21029] = 1; //set to address of the drive (in our case 1)
21 $REGISTERS[21008] = 1007; // set remapped registers
22 $REGISTERS[21009] = 23001; //set to 23001
23 $REGISTERS[21008] = 0; // set remapped registers
24 $REGISTERS[21006] = 500; // set scan rate to 500ms;
```

For your convenience, the code used to setup the Modbus Master as shown above is available for Cut and Paste into your project on the next page.

Modbus RS-485 Communications Between 5300 and Eaton MVX9000

Code available for Cut and Paste into your project:

```
//set up the Basic Group of Eaton Parameters to drive 1 (Modbus address 1)
$REGISTERS[21005] = 9;           // Number of registers to read
$REGISTERS[21000] = 10;          // any value to unlock register group
$REGISTERS[21001] = 10;          // IP address goes here when Using Modbus TCP
$REGISTERS[21002] = 10;          //*****
$REGISTERS[21003] = 10;          //*****
$REGISTERS[21004] = 1;           // Modbus start register 0000H
$REGISTERS[21008] = 1003;        // set index to point to protocol register
$REGISTERS[21009] = 3;           //set protocol to Modbus RTU
$REGISTERS[21008] = 1004;        // set serial port to use
$REGISTERS[21009] = 3;           //Use port three for RS485
$REGISTERS[21028] = 1005;        // set Modbus Master Id
$REGISTERS[21029] = 1;           //set to address of the drive (in our case 1)
$REGISTERS[21008] = 1007;        // set remapped registers
$REGISTERS[21009] = 23001;       //set to 23001
$REGISTERS[21008] = 0;           //
$REGISTERS[21006] = 500;         // set scan rate to 500ms;
```

The code above will remap the Basic Group of Eaton Parameters in the Eaton drive to the 5300's 23000 registers as follows:

Eaton Modbus Address	Parameter Number	Description	CTC Address
0000H	20.01	Motor Nameplate Frequency	23001
0001H	20.02	Motor Nameplate Voltage	23002
0002H	20.03	Source of Master Frequency	23003
0003H	20.04	Source of Operation Command	23004
0004H	20.05	Motor Rated Current	23005
0005H	20.06	Minimum Output Frequency	23006
0006H	20.07	Maximum Output Frequency	23007
0007H	20.08	Acceleration Time 1	23008
0008H	20.09	Deceleration Time 1	23009

Writing to the Eaton Drive Parameters

Now we will show you example code that will change the value of the Eaton Modbus register 0007H which is parameter number 20.08 to set the Acceleration Time 1 to a value of 3.5 seconds. To do this we will check to make sure the Modbus status is ready for a read or write, and then set our register 23008 to a value of 35 (Modbus uses integer values so the decimal place is omitted). Note that if we wanted to set it to 3.0 we would have set it to a value of 30.

Modbus RS-485 Communications Between 5300 and Eaton MVX9000

The screenshot displays a software interface for editing a task. At the top, a 'Start Task' block is shown with a 'main' sub-task. Below it, a 'Parameters' and 'Locals' section is visible. The main task contains a sequence of steps: a 'Set_Accel_Time' step (highlighted with a red box) followed by a 'Move' step. Below the task structure, a code window titled 'QS4 Code - Set_Accel_Time on page 1' is open, showing the following code:

```
S Set_Accel_Time
1  if $REGISTERS[21007]==1 then // check the Modbus Status Register
2  {
3      $REGISTERS[23007] = 35; // if modbus is ready set the Accel Time to 3.5 seconds
4      goto Move; //goto the next step Move
5  }
6  goto Set_Accel_Time; //if modbus is not Ready back to Set_Accel_Time
```

For your convenience here is the same code shown above available for Cut and Paste into your project:

```
if $REGISTERS[21007]==1 then //check the Modbus Status Register
{
    $REGISTERS[23007] = 35; //if Modbus is ready set Accel Time to 3.5 sec
    goto Move; //goto the next step Move
}
goto Set_Accel_Time; //if Modbus is not Ready back to Set_Accel_Time
```