Programmable Logic Controllers

Using ModBus to Interface PLCs with Mach3

3/21/2014

Everything you need to know to interface Mach3 with your programmable logic controller.

MachMotion.com
http://www.machmotion.com
14518 County Road 7240, Newburg, MO 65550
(573) 368-7399 • Fax (573) 341-2672

Purpose

MachMotion's CNC software, Mach3, is designed to serially communicate with ModBus devices. For extra I/O or tool changers, PLCs can be easily interfaced with our control.

 In this manual you will learn how to setup the ModBus protocol to communicate with your PLC. Also, you will examine the addressing scheme and the I/O mapping used to easily access the external I/O and internal registers in your PLC. Although some of this manual may be challenging to understand, the examples included should help clear up any uncertainty.

Note: If your PLC uses 484 or 584/984 addressing modes, your PLC addressing scheme is completely different than shown in this manual.
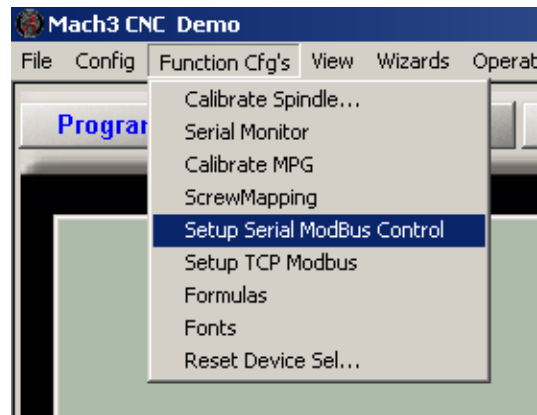
Overview

Let's begin with an overview of how we are going to communicate with the PLC. Every PLC has a section of memory that can be used for general applications. By writing and reading data to these locations, we can control external I/O or just transfer information to and from a PLC. You can designate a certain range of memory inside the PLC for inputs (data to Mach3) and a certain range for outputs (data from Mach3).

To ensure that the PLC is communicating correctly, the PLC and Mach3 are continually toggling two bits. If the PLC doesn't respond for a specific amount of time, the control's emergency stop will activate. After your PLC is programmed with the communication check (as described above), you are ready to read and write to memory inside your PLC.

To simplify your job of accessing external I/O, our control maps user LEDs and pin numbers of port 0 to 4 different registers. This allows for a maximum of 64 inputs and 64 outputs. If you write those registers to the PLC I/O, reading I/O is as simple as simple as setting up ports and pins or accessing user LEDs.

Viewing the Serial ModBus Configuration

To open the serial modbus control, select **Function Cfg's** from the menu bar and then click on **Setup Serial Modbus Control** as shown below.

You should see the following window:

Configuring the ModBus Registers

The two fields you may need to update are the **Address ModBus(Var)** and the **# of Registers**. The
Address ModBus(Var) is the decimal equivalent of the V memory location in the PLC. Automation direct
PLCs automatically calculate the offsets for the input and output holding registers. Therefore, writing or
reading from a register is as simple as placing the desired register (in decimal) in this column.

_____

EXAMPLE 1: Read from register V400 and write to register V600 in the PLC.

Step 1: Convert the V memory locations to decimal using the Window's calculator.
400 octal = 256 in decimal
600 octal = 384 in decimal

Step 2: Enter those values into the serial modbus control window.

	Enabled On/Off	Comment or Device	Port/Address	Slave # 0-10	Refresh 25ms Incr.	Address ModBus(Var)	# of Registers	Direction Input Output
Cfg #0	☒	PLC - Inputs	1	5	25	256	7	Input-Holding
Cfg #1	☒	PLC - Output	1	5	25	384	7	Output-Holding

Step 3: Press **Apply** and your changes will be updated.

In the configuration above, registers V400-V407 are being read from the PLC and registers V600-V607 are being written to the PLC.

_____

The maximum number of registers you can send or receive is 100. If more are needed, you will have to by-pass the MachMotion plugin and do the data processing directly in Mach3.

WARNING:

Make sure to leave the first Cfg as inputs and the second Cfg as outputs. The ModBus communications will not work if you move the CFG positions.

_____

EXAMPLE 2: Read from registers V1200-V1215 and write to registers V1400-V1415.

Step 1: Convert the V memory locations to decimal and find the range.
1215-1200 in octal is 15 or in decimal, 13.
1200 octal = 640 in decimal
1400 octal = 768 in decimal

Step 2: Enter these numbers into the serial modbus control window.

Step 3: Press **Apply** and your changes will be updated.

_____

Programming the Communication Check

To ensure that your control does not continue to operate after your PLC is disconnected, two bits are toggled back and forth. The first bit of the register of the inputs and of the outputs is used for this communication check. For an example, assume that the ModBus Configuration is setup as follows:

	Enabled On/Off	Comment or Device	Port/Address	Slave # 0-10	Refresh 25ms Incr.	Address ModBus(Var)	# of Registers	Direction Input Output
Cfg #0	☒	PLC - Inputs	1	5	25	640	5	Input-Holding
Cfg #1	☒	PLC - Output	1	5	25	740	5	Output-Holding
Cfg #2	☐		1	1	50	0	1	Input Reg

The program below shows how the toggling works with the above configuration.

NOTE: The first 5 registers of the inputs and outputs are reserved for external I/O, E-stop, and the communication check.

Register Mapping

The MachMotion plugin writes the data from a PLC to user DROs. Inputs to Mach3 are mapped to DRO numbers 1600 – 1699. Outputs from Mach3 are mapped to DRO numbers 1500-1599. DROs 1500-1504 and 1600-1604 are reserved for PLC I/O and the communication check. See the tables below.

Outputs from Mach3				Inputs to Mach3			
V Memory Address (Octal)	**DROs**	**Function**	**# of Bytes**	**V Memory Address (Octal)**	**DROs**	**Function**	**# of Bytes**
Base + 0	1500	Communication Check	1	Base + 0	1600	Communication Check	1
Base + 1	1501	Outputs 0-15	2	Base + 1	1601	Inputs 0-15	2
Base + 2	1502	Outputs 16-31	3	Base + 2	1602	Inputs 16-31	3
Base + 3	1503	Outputs 32-47	4	Base + 3	1603	Inputs 32-47	4
Base + 4	1504	Outputs 48-63	5	Base + 4	1604	Inputs 48-63	5
Base + 5	1505	Data	6	Base + 5	1605	Data	6
Base + 6	1506	Data	7	Base + 6	1606	Data	7
Base + 7	1507	Data	8	Base + 7	1607	Data	8
Base + 10	1508	Data	9	Base + 10	1608	Data	9
Base + 11	1509	Data	10	Base + 11	1609	Data	10
Base + 12	1510	Data	11	Base + 12	1610	Data	11
Base + 13	1511	Data	12	Base + 13	1611	Data	12
Base + 14	1512	Data	13	Base + 14	1612	Data	13
Base + 15	1513	Data	14	Base + 15	1613	Data	14
.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.
Base + 142	1598	Data	99	Base + 142	1698	Data	99
Base + 143	1599	Data	100	Base + 143	1699	Data	100

The Base is the V memory location in octal. The number of bytes is the number of registers to be transferred. Notice that you must use DRO 1505 or greater for regular output data to the PLC. In the same way you must use DRO 1605 or greater to read data from the PLC.

Note: The DROs 1500-1599 and 1600-1699 are used regardless of the V memory location.

_____

EXAMPLE 3: Based on the configuration used in example 2, write to V1206 and read from V1407.

Example 2 used V memory 1200 to 1215 as Mach3 outputs (writing to the PLC) so our base is V1200. Therefore V1206 is equivalent to Base + 6. Using the table above, Base + 6 is mapped to DRO 1506. So writing to DRO 1506 writes that word (16 bits) to the PLCs memory location V1206. You can write to it in VB using SetUserDRO(1506,Value) where Value is the number you want to write.

Subsequently, example 2 used V memory 1400 to 1415 as inputs from Mach3 so our base is V1400. Therefore V1407 is equivalent to Base + 7. Using the table above, Base + 7 is mapped to DRO 1607. So reading DRO 1607 should give us the word (16 bits) from the PLCs memory location V1407. You can read from V1407 in VB by using GetUserDRO(1607).

_____

I/O Mapping

As shown above DROs 1601-1604 and 1501-1504 are used for I/O. A total of 64 inputs and 64 outputs can be accessed in these 4 registers. User LEDs and port 0 and pin 0-63 are used to write to the data inside Mach3. To utilize the I/O from Mach3, the V memory locations Base + 1 through Base + 4 must be written to the actual outputs on the PLC.

_____

EXAMPLE 4: Based on the configuration used in example 2, program the PLC to use the first 16 inputs and outputs.

From the table above we know that the first sixteen inputs and outputs are written to Base + 1. For the Mach3 outputs we load this register (V1401) and output it to Y0-Y17. For the Mach3 inputs we read the input port and output it to register V1201. See the program below:

_____
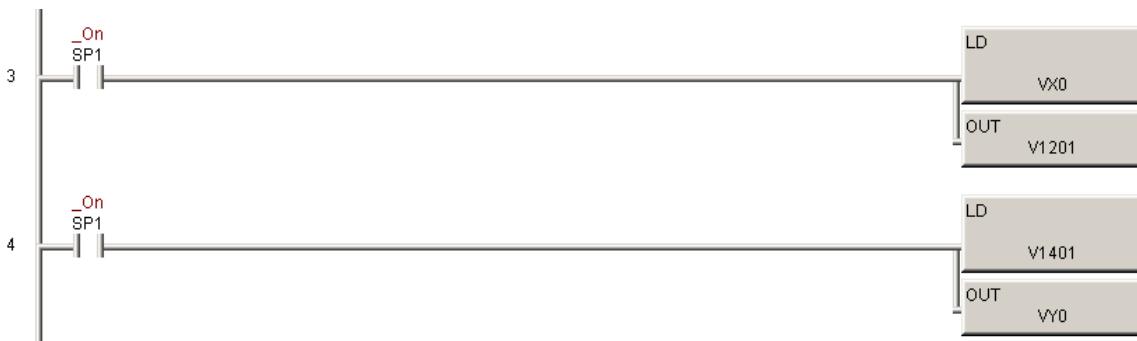
The tables below show the actual mapping of the user LEDs and the pin numbers.

Inputs						Outputs					
User LEDs	Port	Pin	PLC Output Register	Bit Number	PLC Input	User LEDs	Port	Pin	PLC Input Register	Bit Number	PLC Output
2200	0	0	Base + 1	0	X0	2100	0	0	Base + 1	0	Y0
2201	0	1	Base + 1	1	X1	2101	0	1	Base + 1	1	Y1
2202	0	2	Base + 1	2	X2	2102	0	2	Base + 1	2	Y2
2203	0	3	Base + 1	3	X3	2103	0	3	Base + 1	3	Y3
2204	0	4	Base + 1	4	X4	2104	0	4	Base + 1	4	Y4
2205	0	5	Base + 1	5	X5	2105	0	5	Base + 1	5	Y5
2206	0	6	Base + 1	6	X6	2106	0	6	Base + 1	6	Y6
2207	0	7	Base + 1	7	X7	2107	0	7	Base + 1	7	Y7
2208	0	8	Base + 1	8	X10	2108	0	8	Base + 1	8	Y10
2209	0	9	Base + 1	9	X11	2109	0	9	Base + 1	9	Y11
2210	0	10	Base + 1	10	X12	2110	0	10	Base + 1	10	Y12
2211	0	11	Base + 1	11	X13	2111	0	11	Base + 1	11	Y13
2212	0	12	Base + 1	12	X14	2112	0	12	Base + 1	12	Y14
2213	0	13	Base + 1	13	X15	2113	0	13	Base + 1	13	Y15
2214	0	14	Base + 1	14	X16	2114	0	14	Base + 1	14	Y16
2215	0	15	Base + 1	15	X17	2115	0	15	Base + 1	15	Y17
2216	0	16	Base + 2	0	X20	2116	0	16	Base + 2	0	
2217	0	17	Base + 2	1	X21	2117	0	17	Base + 2	1	
2218	0	18	Base + 2	2	X22	2118	0	18	Base + 2	2	
2219	0	19	Base + 2	3	X23	2119	0	19	Base + 2	3	
2220	0	20	Base + 2	4		2120	0	20	Base + 2	4	
2221	0	21	Base + 2	5		2121	0	21	Base + 2	5	
2222	0	22	Base + 2	6		2122	0	22	Base + 2	6	
2223	0	23	Base + 2	7		2123	0	23	Base + 2	7	
2224	0	24	Base + 2	8		2124	0	24	Base + 2	8	
2225	0	25	Base + 2	9		2125	0	25	Base + 2	9	
2226	0	26	Base + 2	10		2126	0	26	Base + 2	10	
2227	0	27	Base + 2	11		2127	0	27	Base + 2	11	
2228	0	28	Base + 2	12		2128	0	28	Base + 2	12	
2229	0	29	Base + 2	13		2129	0	29	Base + 2	13	

2230	0	30	Base + 2	14		2130	0	30	Base + 2	14	
2231	0	31	Base + 2	15		2131	0	31	Base + 2	15	
2232	0	32	Base + 3	0		2132	0	32	Base + 3	0	
2233	0	33	Base + 3	1		2133	0	33	Base + 3	1	
2234	0	34	Base + 3	2		2134	0	34	Base + 3	2	
2235	0	35	Base + 3	3		2135	0	35	Base + 3	3	
2236	0	36	Base + 3	4		2136	0	36	Base + 3	4	
2237	0	37	Base + 3	5		2137	0	37	Base + 3	5	
2238	0	38	Base + 3	6		2138	0	38	Base + 3	6	
2239	0	39	Base + 3	7		2139	0	39	Base + 3	7	
2240	0	40	Base + 3	8		2140	0	40	Base + 3	8	
2241	0	41	Base + 3	9		2141	0	41	Base + 3	9	
2242	0	42	Base + 3	10		2142	0	42	Base + 3	10	
2243	0	43	Base + 3	11		2143	0	43	Base + 3	11	
2244	0	44	Base + 3	12		2144	0	44	Base + 3	12	
2245	0	45	Base + 3	13		2145	0	45	Base + 3	13	
2246	0	46	Base + 3	14		2146	0	46	Base + 3	14	
2247	0	47	Base + 3	15		2147	0	47	Base + 3	15	
2248	0	48	Base + 4	0		2148	0	48	Base + 4	0	
2249	0	49	Base + 4	1		2149	0	49	Base + 4	1	
2250	0	50	Base + 4	2		2150	0	50	Base + 4	2	
2251	0	51	Base + 4	3		2151	0	51	Base + 4	3	
2252	0	52	Base + 4	4		2152	0	52	Base + 4	4	
2253	0	53	Base + 4	5		2153	0	53	Base + 4	5	
2254	0	54	Base + 4	6		2154	0	54	Base + 4	6	
2255	0	55	Base + 4	7		2155	0	55	Base + 4	7	
2256	0	56	Base + 4	8		2156	0	56	Base + 4	8	
2257	0	57	Base + 4	9		2157	0	57	Base + 4	9	
2258	0	58	Base + 4	10		2158	0	58	Base + 4	10	
2259	0	59	Base + 4	11		2159	0	59	Base + 4	11	
2260	0	60	Base + 4	12		2160	0	60	Base + 4	12	
2261	0	61	Base + 4	13		2161	0	61	Base + 4	13	
2262	0	62	Base + 4	14		2162	0	62	Base + 4	14	
2263	0	63	Base + 4	15		2163	0	63	Base + 4	15	

_____

EXAMPLE 5: Turn on output Y3 and read from X5 of example 4.

For Y3 is bit 3 of Base + 1. That means you can use LED 2103 or port 0 pin 3 to turn on Y3. SetUserLED(2103,1) will turn it on.

X5 is bit 5 of Base + 1. That means that you can use LED 2205 or port 0 pin 5 to read the state of X5.

_____

_____

EXAMPLE 6: Setup a D0-06 PLC as external I/O.

Below is the serial modbus configuration:

Below is the PLC program:

1

Communication Check
Toggle
B1400.0
——| |——

Communication Check
Toggle1
B1200.0
——(RST)

2

Communication Check
Toggle
B1400.0
——|/|——

Communication Check
Toggle1
B1200.0
——(SET)

3

_On
SP1
——| |——

```
LDD
      VX0
```
```
OUTD
      V1201
```

4

_On
SP1
——| |——

```
LD
      V1401
```
```
OUT
      VY0
```

5

——(END)

_____