Using Modbus with Mach3

By
Peter Homann

Eventually a new user to Mach3 will hear the term "Modbus" being bandied about. This article explains how it can be used with Mach3. But, before diving in and showing how to use it in Mach3, a basic understanding of what Modbus is, and how it works is required.

The Modbus Protocol

Modbus is arguably one of the most widely used communication interfaces in industrial control applications. It was developed in the late seventies and has stood the test of time. The nice thing about the Modbus standard is its flexibility and the ease with which it can be implemented and used.

Its primary purpose of Modbus is to exchange data between slave devices and a master. To put this into context, Mach3 is the Master, ModIO™s, VFDs, and other devices are the slaves.

Modbus implements a number of protocols, namely, serial RTU/ASCII and Modbus/TCP. The serial protocols may be transmitted over standard RS232 and/or RS485 interfaces. Modbus/TCP is transmitted over Ethernet.

Serial RTU is the most common protocol used with Mach3, although Modbus/TCP is gaining popularity. Both Protocols may be used simultaneously with Mach3. It should be noted that Serial/ASCII is not implemented in Mach3.

Figure 1 below illustrates an Architecture Overview of a Mach3 Modbus setup.

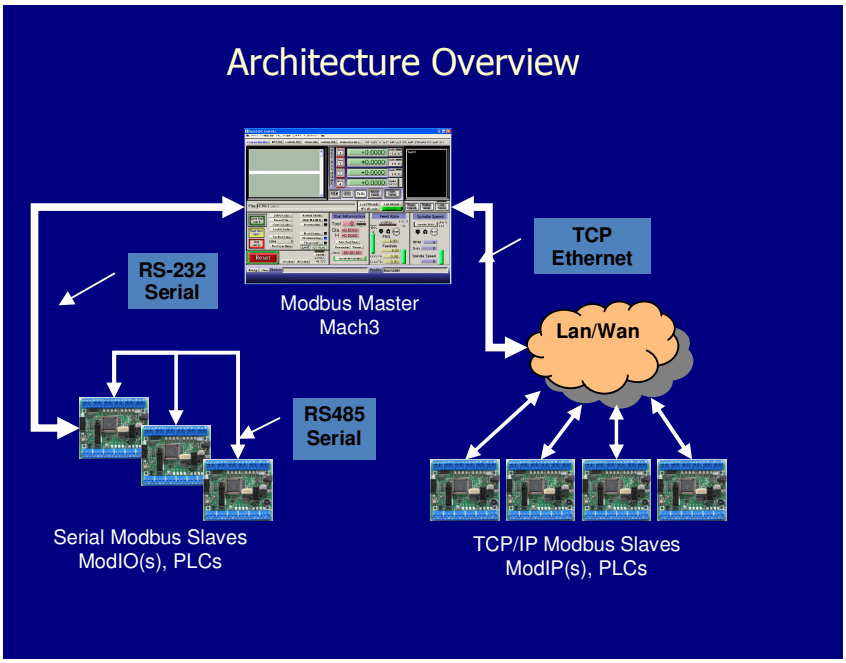


Figure 1 Architecture Overview

Query/Response Cycle

Modbus works on a query/response cycle. Only the master (Mach3) can initiate communication. The slaves just respond to the masters queries. The master transmits a query frame and one of the slaves (ModIO™) returns a response frame back to the master. Each frame that the master sends out contains the address of the slave that it wants the response from.

It is important to note that the query and response messages all have a known starting and end point. This allows the receiver to know when a message has been received, if it has been received in full and without error and determine if the message is for them.

The Query / Response cycle is depicted in Figure 4 below.

Modbus Message Structure

Modbus messages are framed to allow the receiver to detect the start and end of the message. Modbus RTU uses time gaps of silence to separate the message frames. Each message must be preceded with a time gap equivalent to **3.5** characters. When the receiver detects this gap after receiving characters, it knows that a message has been received and can process it.

The frame of a Modbus RTU message looks like this;

Field	Description
Device address	Address of the receiver
Function Code	Code defining the message type
8-bit Data Bytes	Block containing additional data
Checksum	Checksum to validate message

Device Address - This identifies who the message is addressed to. Valid addresses are **0-247** with address **0** being used as the broadcast address. Broadcast messages are a special case, and are processed by all slaves with no response message being returned. Addresses **1-247** are assigned to individual slave devices. When a slave responds to a request it uses the same address that was in the request. This allows the master to detect that the slave has responded the request.

Function Code – The function code identifies the message type and the action that is to be performed by the slave. The function codes implemented by Mach3 are;

Code	Data Type Description
01	Read Coils
02	Read Discrete Inputs
03	Read Holding Registers
04	Read Input Registers
15	Write multiple Coils
16	Write multiple Holding Registers

The register variables are 16 bit. Discretes and coils are single bit values. Coils and holding registers may be read and/or written to. Discretes and Input registers are read only.

8-bit data Bytes – This block contains any additional information in the message. For the read commands, it contains the address and the number of registers to read. For the write commands it contains the data to be written, the number of registers to write the address that the data will be written to.

Checksum – The checksum is a 16-bit CRC code that is used to validate the message. The receiver uses this to ensure that the message is not corrupted. If the message is deemed to be invalid, the receiver just ignores it and does not respond with a reply.

Mach3 and Modbus a practical example

Figure 2 below depicts a simple example for using a ModIO™ via Modbus to perform a number of simple Input/Output functions.

Three push buttons will be used to perform the Cycle Start, Feed-Hold and Cycle Stop input functions. On the output side a three indicator light tower will be used to show the mache/Mach3 status.

The ModIO™ is connected to the PC via the serial port connector. In this example, Serial Port 1 is used.

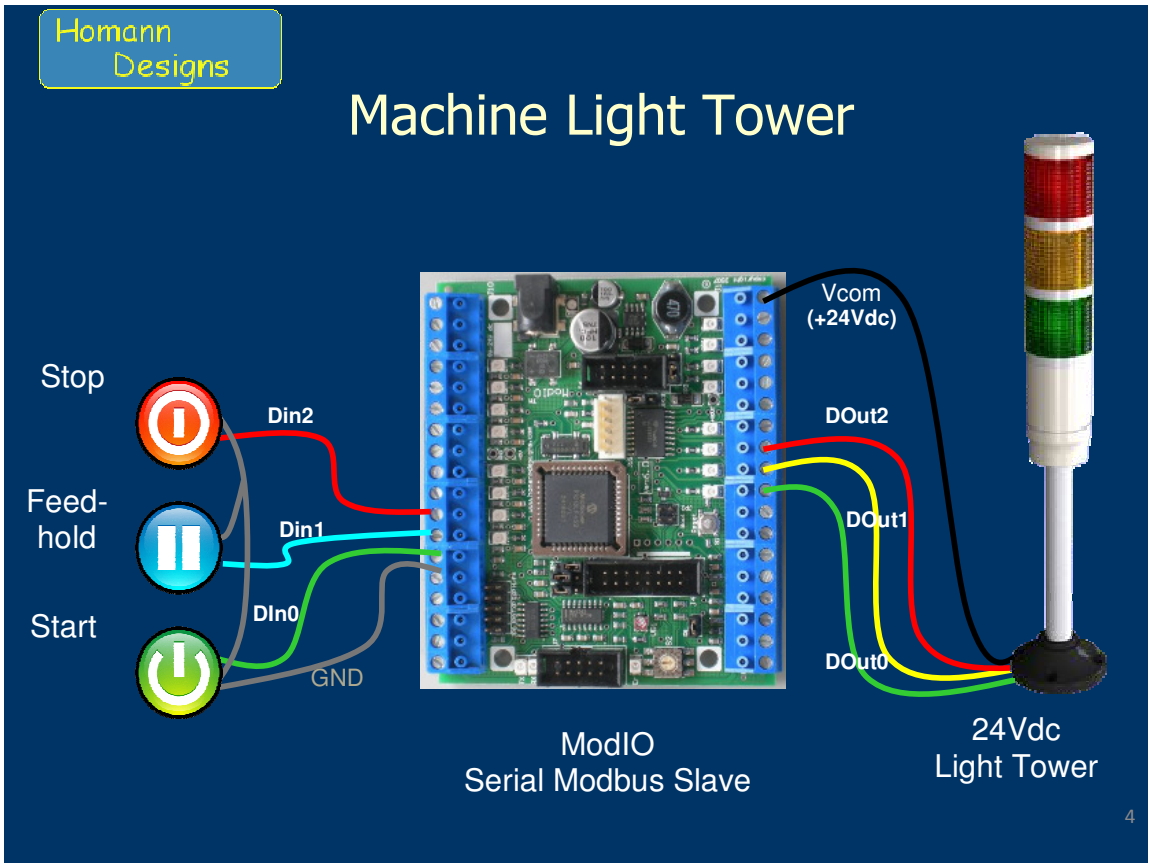


Figure 2 Machine Light Tower Example

Mach3 Modbus Configuration

In order to use Modbus in Mach3, a number of configurations need to be set up. Mach has the ability to support Modbus RTU over a serial interface, a TCP/IP Ethernet interface, or both.

Additionally, for Modbus over a serial line, Mach3 has two methods.

1. Legacy Modbus Interface
2. Plugin Supported Interface

The legacy interface was the initial Modbus interface initially targeting the ModIO™ device. It is no longer the preferred interface, and will eventually be removed from Mach3. As such, this article will focus on the Plugin Supported Interface.

The first thing to do in getting Modbus to work is to enable it in the Ports and Pins Configuration page. As can be seen in Figure 3 below, the checkboxes for Modbus InputOutput Support and Modbus Plugin Supported are both ticked. You may also notice that the TCP Modbus Support checkbox is also ticked. You only need to tick this one if you are also going to use Modbus devices that communicate over Ethernet.

You should now exit Mach3 and restart for this part of the configuration to take effect.

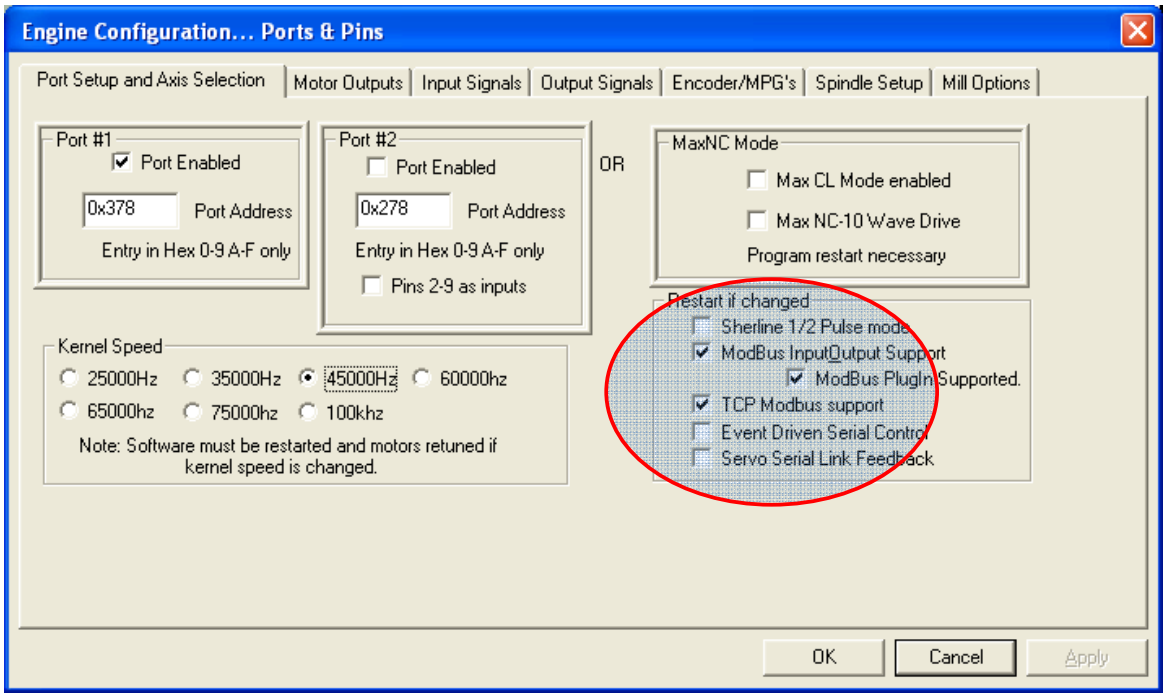


Figure 3 Modbus Module Enable

Serial Modbus Configuration

For Mach3 to talk to the ModIO™ shown in Figure 2 above, the serial port needs to be configured, as does the transfer of data (Modbus messages) between Mach3 and the ModIO™.

The transfer of Modbus data is best thought of as shared memory between the master and slave. Figure 4 below depicts the Modbus Query/Response cycle that occurs between Mach3. Mach3 transfers output data to the ModIO™, and reads back Input data from the ModIO™. These transfers are done continuously at a rate set by the user, to a maximum of 40 times per second.

The result is that Mach3 continuously gets a copy of the ModIO™'s inputs (switches), and the ModIO™ gets a copy of the status outputs from Mach3

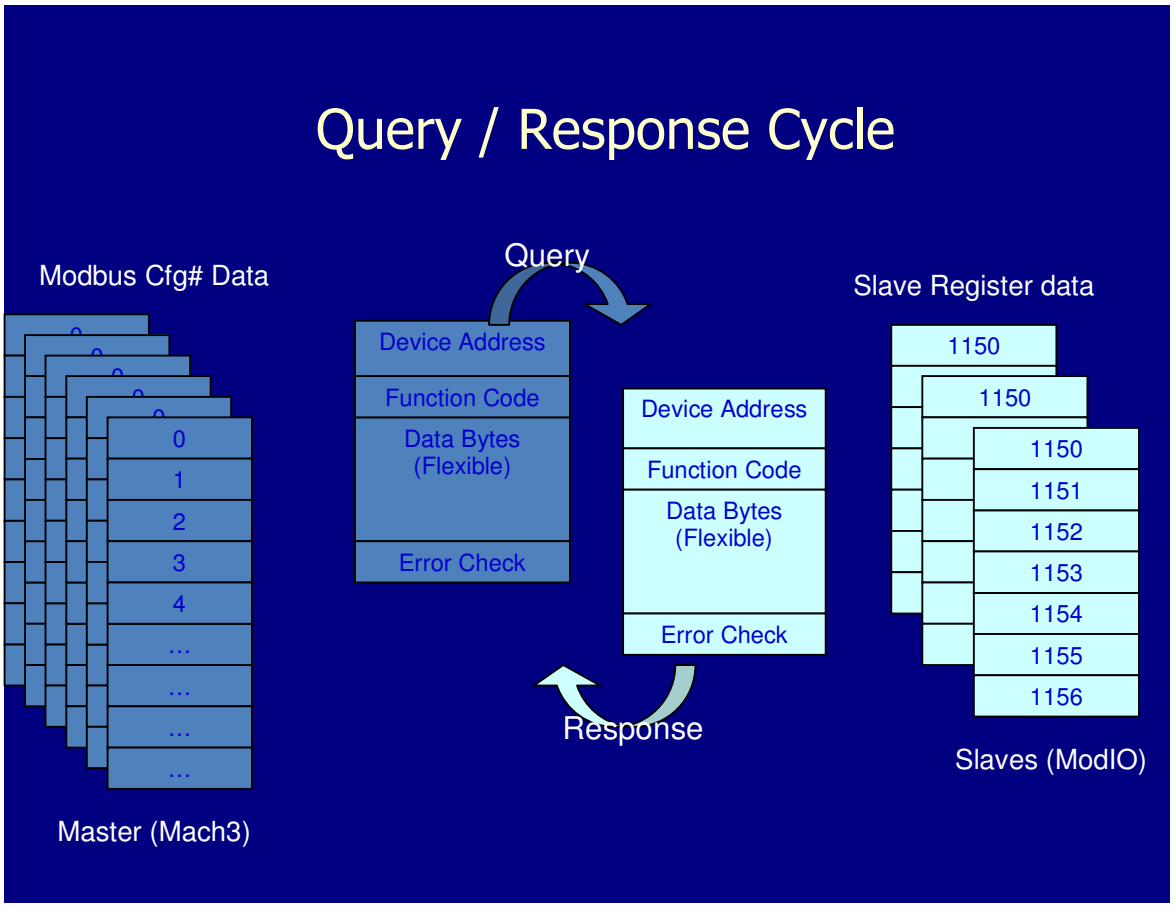
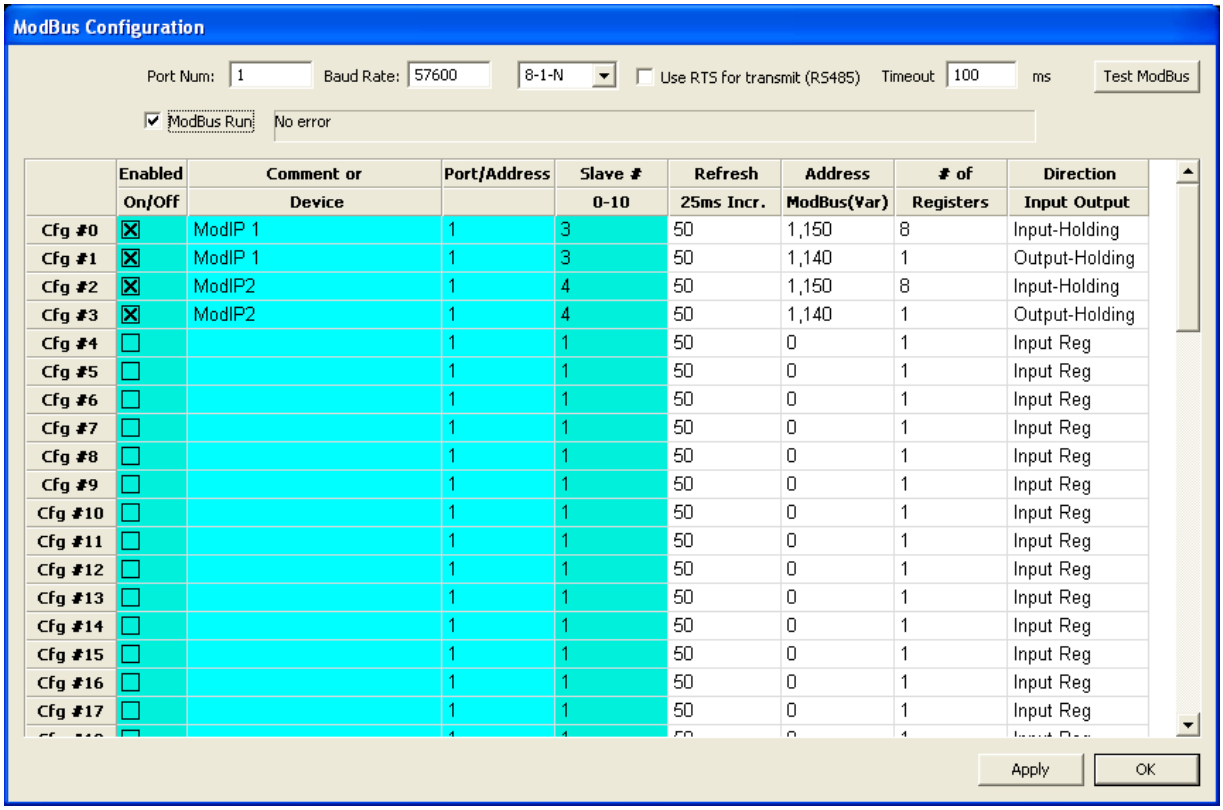


Figure 4 Query / Response Cycle

By continuously transferring the data between Mach3 and the ModIO™, any spurious data errors will be corrected in the next transfer, and will be transparent to the user.



	Enabled On/Off	Comment or Device	Port/Address	Slave # 0-10	Refresh 25ms Incr.	Address ModBus(Var)	# of Registers	Direction Input Output
Cfg #0	☒	ModIP 1	1	3	50	1,150	8	Input-Holding
Cfg #1	☒	ModIP 1	1	3	50	1,140	1	Output-Holding
Cfg #2	☒	ModIP2	1	4	50	1,150	8	Input-Holding
Cfg #3	☒	ModIP2	1	4	50	1,140	1	Output-Holding
Cfg #4	☐		1	1	50	0	1	Input Reg
Cfg #5	☐		1	1	50	0	1	Input Reg
Cfg #6	☐		1	1	50	0	1	Input Reg
Cfg #7	☐		1	1	50	0	1	Input Reg
Cfg #8	☐		1	1	50	0	1	Input Reg
Cfg #9	☐		1	1	50	0	1	Input Reg
Cfg #10	☐		1	1	50	0	1	Input Reg
Cfg #11	☐		1	1	50	0	1	Input Reg
Cfg #12	☐		1	1	50	0	1	Input Reg
Cfg #13	☐		1	1	50	0	1	Input Reg
Cfg #14	☐		1	1	50	0	1	Input Reg
Cfg #15	☐		1	1	50	0	1	Input Reg
Cfg #16	☐		1	1	50	0	1	Input Reg
Cfg #17	☐		1	1	50	0	1	Input Reg

Figure 5 Modbus Message Configuration

Figure 5 above shows the setup of the serial port and of the message transfers. Along the top you can see that serial port 1 is being used, along with the settings of 57600 baud, 8 data bits, 1 stop bit and no parity. Also notice that a timeout of 100ms is configured. This means that if the ModIO™ does not respond to the Modbus request within 100ms, Mach3 will assume that it is not going to get a reply, and moves on to the next query.

There is also a checkbox to use RTS for transmit. When using a RS232 to RS485 converter, it is necessary to tell the converter when Mach3 is transmitting. This allows the converter to switch from receive to transmit (A bit like pressing the talk button on a walkie-talkie).

One other very important checkbox is the one labelled “Modbus Run”. This needs to be checked for Modbus to work.

Figure 5 above also shows the Message transfers that have been set up. We will only be looking at the first two configuration entries namely, Cfg #0 and Cfg #1. The other two entries are for a 2nd ModIO™ that we are not concerned with.

The first configuration Cfg #0 is reading 8 holding registers from a ModIO™ with a Modbus sub-address of 3. The 8 registers in the ModIO™ start at address 1150. Also they are being read every 50ms, a rate of 20 times per second.

The 2nd configuration Cfg #1 is writing a single holding registers to the same ModIO™ with a Modbus sub-address of 3. The register is written to address 1040. It also is being written every 50ms, a rate of 20 times per second.

To understand better the contents of the ModIO™ registers being transferred, it is recommended that you read the user manual for the ModIO™ as it gives a detailed description of each register.

In Brief, register 1040 of the ModIO™ uses the lower eight bits of the register to represent the state of the 8 ModIO™ outputs. For each of these 8 bits, if it is set, then the corresponding output will turn on.

Also, register 1150, contains the status of the ModIO™'s 8 inputs. If the input is active the corresponding bit in this register will be 1.

The end result of the above configuration is that there is now data being continuously being transferred between Mach3 and the ModIO™. The status of the 3 switches connected to the ModIO™ will now reside in the data buffer for Cfg #0. Similarly, the contents of the data buffer for Cfg #1 will now be written to the output resister in the ModIO™.

The next step is for Mach3 to process the switch data in the buffer for Cfg #0, and to write the status to the buffer for Cfg #1 so that it can be reflected by the indicators in the Light Tower. The processing of this data performed via Mach3 Brains.

Mach3 Brains

The Mach3 brains are a graphical programming environment that allows the user to create programs (Brains) so as to customise the functionality of Mach3.

For our example we need to create a brain to perform the following functionality.

1. When Mach3 is in RESET, flash the Red Indicator
2. When Mach3 is in FEEDHOLD, turn on the Yellow Indicator
3. When Waiting for a tool-change, flash the Yellow Indicator
4. When running G-code, or in FEEDHOLD, or waiting for a tool-change turn on the Green Indicator

The Brains environment consists of 3 modules;

- Brains Control Form
- Brains Editor
- Brains Real Time Viewer

A full description on the use of Brains in Mach3 could fill an article in its own right. For more detailed description of Brains have a look at the video tutorials on the Mach3 support sight at;
http://www.machsupport.com/videos.php

Also have a look through the relevant forum topics on the Mach3 support forum at;
http://www.machsupport.com/forum/index.php

Brain Control Form

This Form provides the control to load the brains, turn them on and off and to view them in the Brains Real Time Viewer

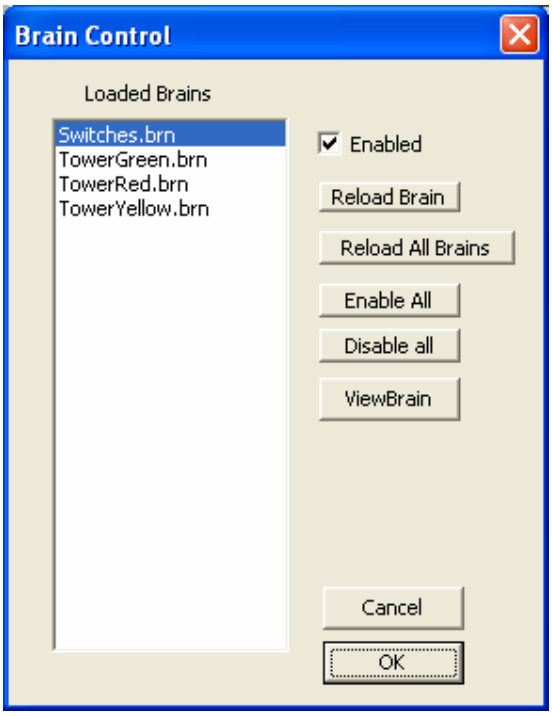


Figure 6 Brain Control Form

For out demonstration, you can see four brains, one for the switches, and one each for the three light tower indicators. Breaking up your brain tasks into a number of small brains is a good idea as is cuts down the complexity when working in the Brains editor.

Brain Editor

The editor is where you can create and edit your Brains. Depicted below in Figure 7 is the switches brain. It is a very simple brain. On the left hand side are the inputs, followed by any processing done to the inputs with the result being set into the output on the right hand side.

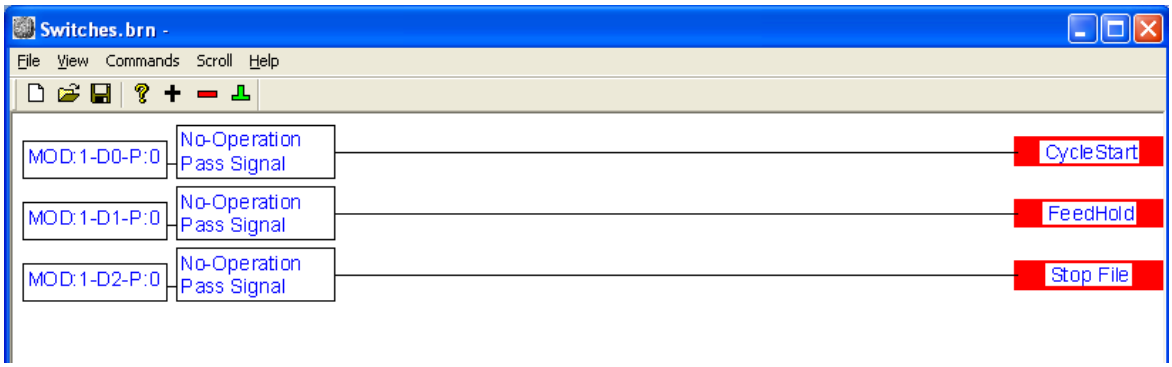


Figure 7 Switches Brain

In the brain above if we look at the third line, the input box "***MOD: 1-D2-P:0***" specifies;

- ***MOD:*** The input is a Modbus input
- ***1*** The 1st register in the Modbus buffer
- ***D2*** Data bit 2 is being used (Red Switch is connect to this input)
- ***P:0*** Cfg #0 data buffer is being used.

The processing or function box which is next shows that the input is just passed through without modification. The output Box shows that the output item being written to is the Mach3 **Stop File** button.

The other two lines in this brain work in a similar way and the result of all of this;

- When the Green switch is pressed **Cycle Start** is activated
- When the Blue switch is pressed **Feed Hold** is activated
- When the Red switch is pressed **Stop** is activated

A more complex brain is depicted below in Figure 8. This is for the yellow indicator that is used to indicate when a tool-change is occurring and when feed hold is activated.

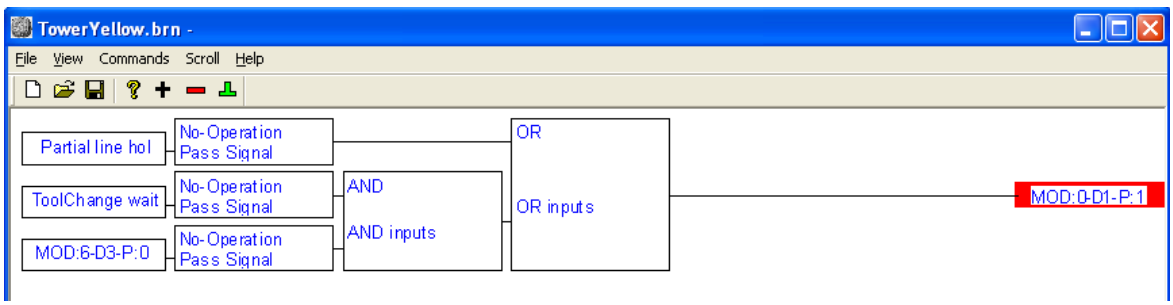


Figure 8 Tower Yellow Indicator

Looking at the inputs we can note the following;

- ***Partial Line Hold*** is used to determine when the G-Code is waiting due to Feed Hold being active
- ***ToolChange Wait*** is used to indicate that the tool change is waiting for the user to press Cycle Start again
- **MOD:6-D3-P:0** This Modbus input from the ModIO™ is used to generate a flashing signal from the indicator. The register identified by the 6 is a continuously incrementing counter generated by the ModIO™. Data bit 3 of this register is toggling at a rate of about 2 per second, so we can used this as a flashing indicator.

The processing is a bit more complex that the previous brain. The flashing indicator from the ModIO™ and the Tool change wait signal are ANDed together with the result that we now have a Tool change Wait signal that now is flashing.

The next function to the right takes the flashing tool change Wait signal and ORs it with the Partial Line Hold signal.

The output of this block is then set to Bit 1 of the first register in the buffer for Cfg#1 for the Modbus Serial Plugin module. This then gets sent to the ModIO™ and since the yellow indicator is connected to output D1, it will be controlled by this brain

So now the Yellow indicator will turn on if the Feedhold is active. Or, if Mach3 is waiting for a Tool change, the indicator will flash, otherwise it will be off.

Similarly, the Green indicator Brain as been written so that the indicator will turn on when G-code is running or when waiting for a tool change or feed hold is active. In other words the green indicator will be on whenever Mach3 is not in Reset or Stopped.

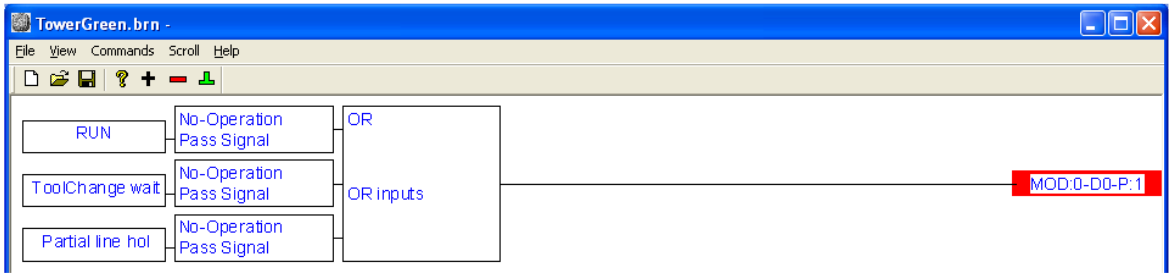


Figure 9 Green Indicator Brain

The Red indicator is used to indicate that RESET is active and will flash the Red indicator when active.

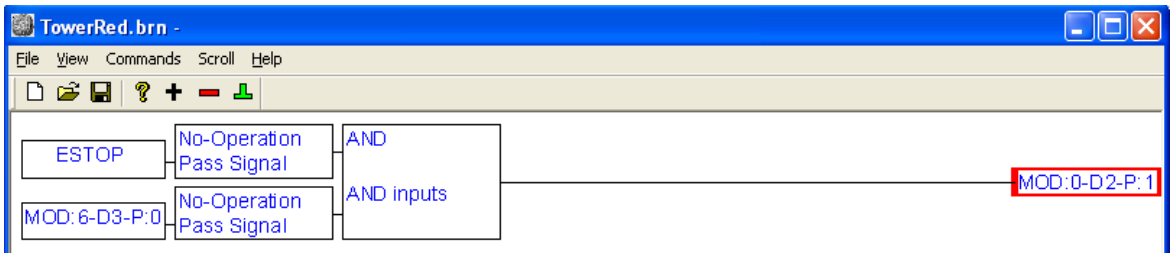


Figure 10 Red Indicator Brain

Brain Real Time Viewer

The Viewer is accessed from the Brains Control Form and can be used to debug a brain as it shows the state of the inputs, outputs and processing blocks

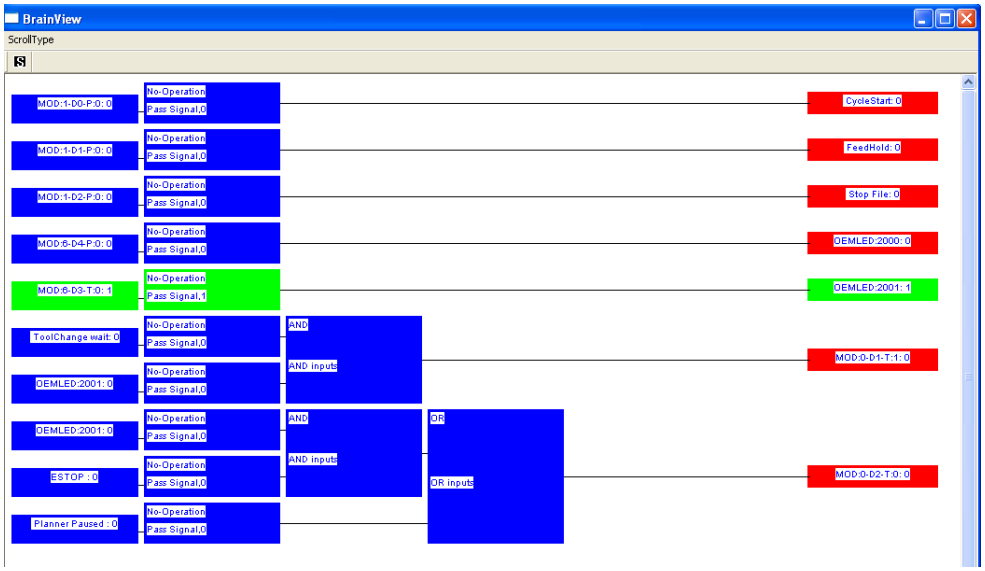


Figure 11 Brains Real-time Viewer

The Viewer depicted above in Figure 11shows the state of each of the blocks with the brain. As can been seen, one of the inputs is active and is shown is Green. The corresponding functions block that are also active are shown in green as well. Outputs are coloured green or red depending whether they as active or not.

The status is shown in real time so if a switch is pressed, that input will change to green to indicate this. The viewer is able to help with the debugging of brains when the system doesn't appear to behave as expected

Conclusion

Mach3's integration of Modbus provides a very powerful, industrial strength interface, which will allow you to control many off the shelf I/O devices such as Programmable Logic Controllers (PLCs) and Variable Frequency Drives (VFDs). You can use it with devices such as the ModIO™ to create complex custom I/O devices such as Automatic Tool changers and remote Pendants.

For further information and resources on Modbus, applications and hardware a number of resources are listed below.

Modbus Specification
http://www.modbus.org/specs.php

ModIO™ Manual
http://homanndesigns.com/store/index.php?main_page=product_info&products_id=4&zenid=fbbc5a68049baf9e2a51ee4192ca79c0

Modbus Customisation examples
http://www.machsupport.com/MachCustomizeWiki/index.php?title=Customization_case_studies

Mach3 Modbus and Brain Tutorial Videos
http://www.machsupport.com/videos.php

Mach3 Support Forum
http://www.machsupport.com/forum/index.php