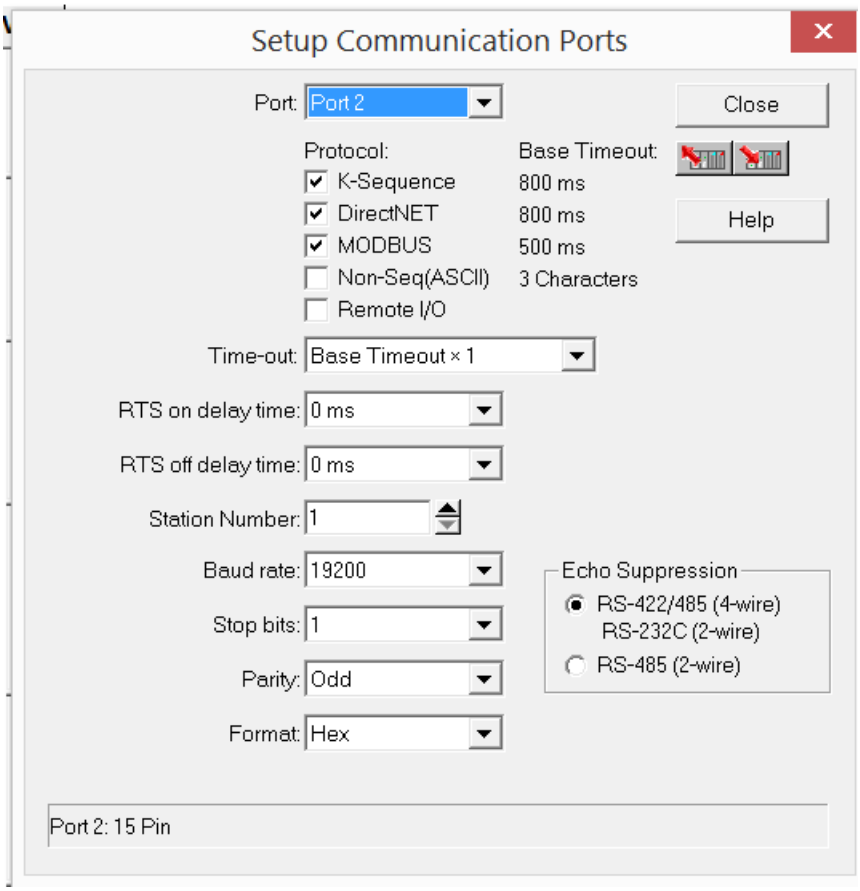
S S Systems, LLC
PO Box 235
Briceville, TN 37710
Ph: (865) 426-9728
https://sites.google.com/site/sssystemsautomation

16July2014

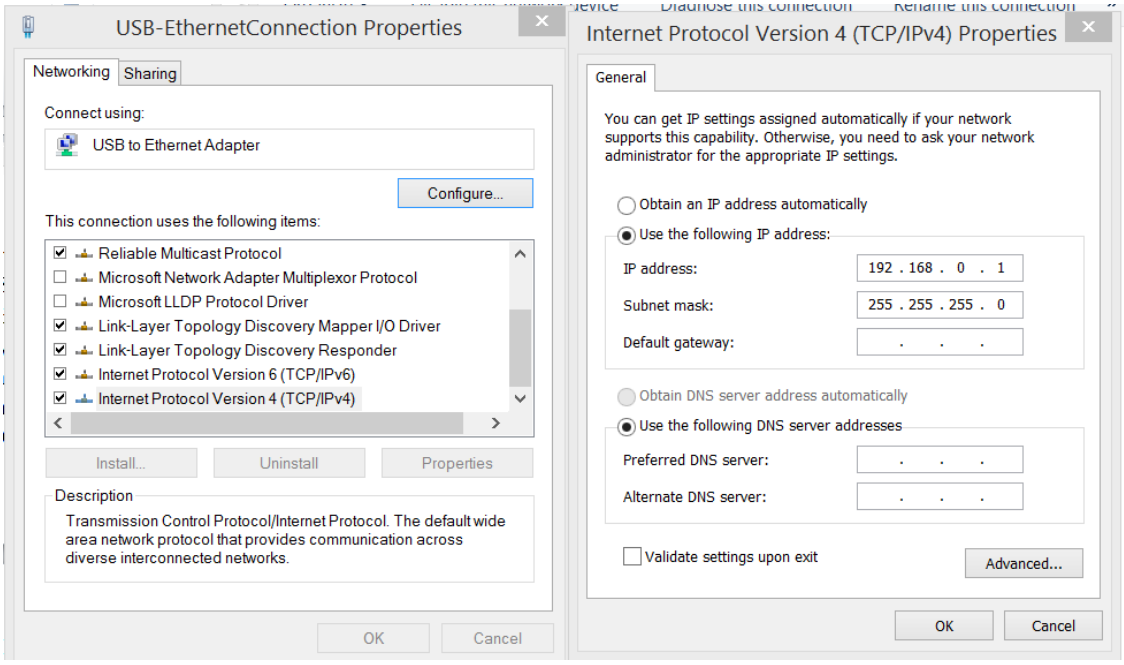
How to set up Mach4 Modbus; Serial and TCP/IP to talk to a DL06 PLC:

This article will show you how to set up Serial RTU and TCP/IP Modbus to talk to a PLC from/to Mach4. The PLC we will use in this example is an Automation Direct DL06 PLC with an ECOM100 (Ethernet TCP/IP module). If you don't have an Ethernet module then you will only be able to set up the Serial example part. Since the DL06 has two serial com ports, com1 is the Programming port and we will be hooked to that one with DirectSoft5 PLC programming software with "Link View" enabled so we can watch bits change states, power flow and the data monitoring update. Com 2 on the DL06 will be set to: 19200, 8, 1, Odd, device 1 and "Modbus" tick box will be checked. (Screen shot of Second Port (Com2), in the DS 5 software)



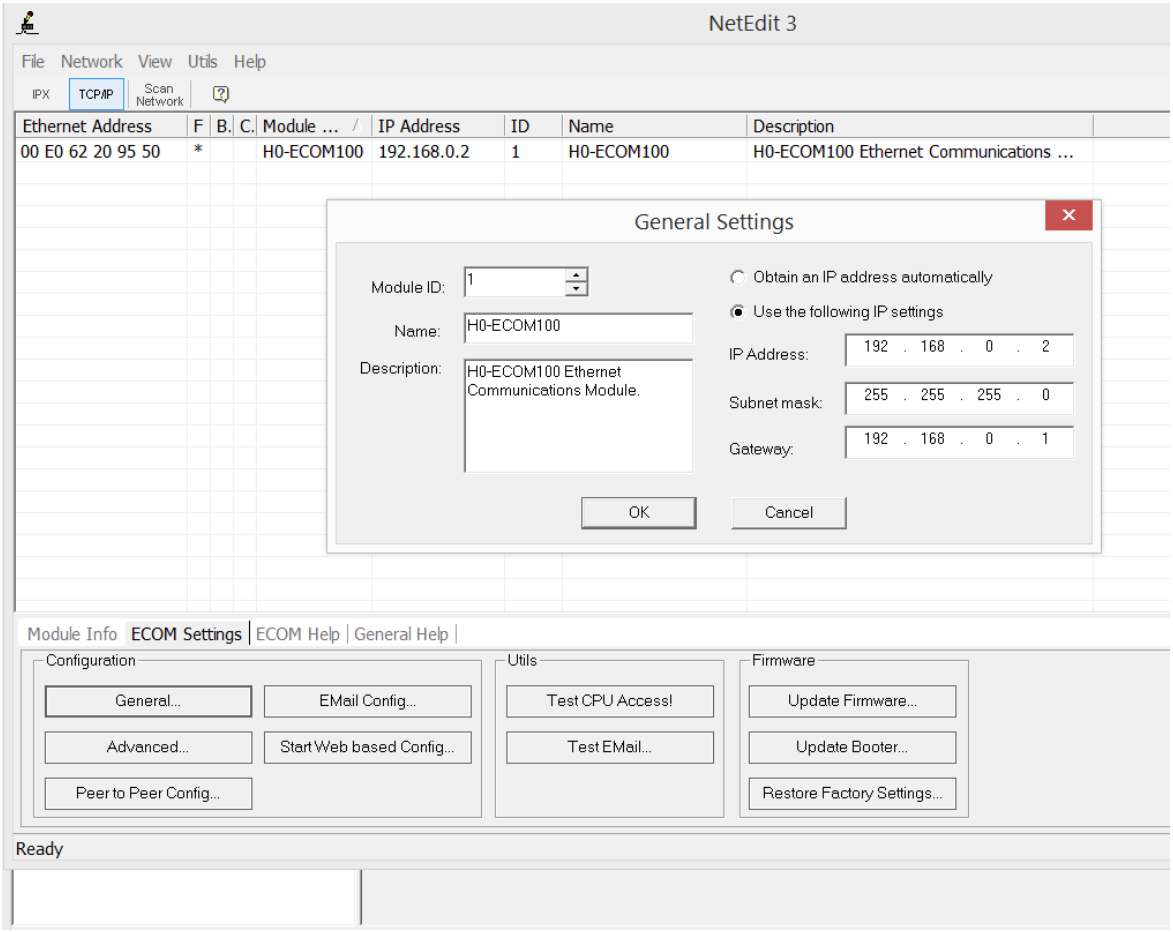
(NOTE: If you use another PLC brand then you will have to adapt this to what you're used to working with.)

The PC will have its Ethernet port set to: 192.168.0.1, SNM 255.255.255.0, under Advanced -> Wins set NetBIOS over TCP/IP. (Screen Shot of the setup of the PC Ethernet Card).



In the ECOM module use NetEdit3 to set its address to: 192.168.0.2, SNM: 255.255.255.0 and put in the Gateway: 192.168.0.1 (the PC). (Screen Shot of NetEdit3, in the DS5 Software suite).

NOTE: There may be LARGE gaps of White Space between pages that have pictures due to picture size.



So you will need 2 (TWO) programming cables for the PLC, a “Port 1 Programming Cable” to hook to Com1, and a “Port 2 Programming Cable” (both from ADC) to hook to Com2 of the PLC. You will also need a “Cross-Over” Cat5 cable if direct hook up to the ECOM100 Module, or if using patch you will have to go through a hub.

In the attached Zip file that has the Mach4 ini file for this project, it is run from the stock Mach Mill profile, but if you want you can create a new profile, and drop this ini in its profile folder. So the Zip File contains:

1. Mach4 ini (profile) file. (Use this, since it has all the Modbus settings already done for you in M4).
2. “wxMachATC.set” (screen set that works with this; it has an ATC tab to do the testing on).
3. Project Files for the “mach4testing” PLC program.
4. This Article, with screen shots.

You will need to work this example:

1. Direct Logic 06 PLC from Automation Direct (ADC).
2. Direct Soft 5 PLC Programming software from ADC.
3. ECOM100 (TCP/IP, Ethernet) module IF you plan on doing the Ethernet portion of this project.
4. Cat5 Patch cable if connecting directly to the ECOM100, or a hub and 2 Cat5's if using patch cable.
5. A Port 1 Programming cable for the DL06 from ADC (you have to have this to program the PLC).
6. A Port 2 Programming cable for the DL06 from ADC this allows you to talk to Port 2 via RTU Serial Modbus. You will need this to talk serial Modbus with Mach4.

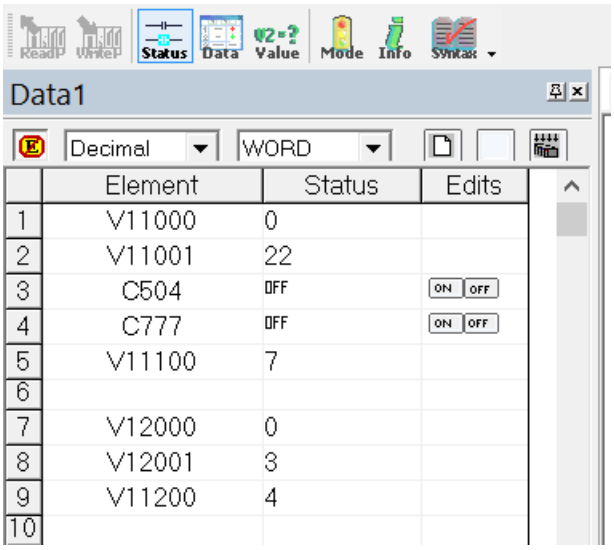
Project goals:

A simple PLC program for both Serial RTU and TCP/IP Modbus; that will read the status 3 physical switches (PLC Inputs: X4, X6 and X7) and report that to Mach4, also Mach4 will send 3 outputs (OUTPUT1, 2, and 3) from Buttons on the ATC screen to the PLC and the PLC will turn on 3 lights at: Y4, Y5 and Y6. The Inputs and Outputs will be controlled by using Bit-Of-Word (otherwise known as Bit Packing), for reading/writing discrete bit statuses, both in the PLC and in M4 Serial RTU Modbus.

We will be doing the same as above for the Ethernet TCP/IP Modbus, but the inputs from the PLC will be from a counter, counting up from 1-3, in 1 second intervals and that will turn on the M4 ATC bits in sequence. The TCP/IP on the ATC page will also have 3 buttons, which send Mach4 OUTPUT4, 5, and 6 to the PLC to turn on another set of 3 lights at: Y7, Y10, and Y11.

On the ATC Screen, there will be a top DRO and a bottom DRO for both the Serial and TCP respectively, the top DRO shows the "Sum" of the bit-packed register, i.e. when the PLC fills a word that Mach4 is reading and it is a bit packed word, then the holding register would show "7" if bits 0, 1, and 2 from the PLC where filled…. BUT!!!! If you're looking at that SAME register and bringing it in as "Bit-Packed" then that fills by "Big Indian" (or Most Significant Bit first)!!! So, if Bit 0 of the word that Mach4 is reading is "1", then in the Bit Packed register bit 15 will be on. Bit 1 in PLC will be bit 14 under bit pack ect.
YOU NEED TO BE AWARE OF THIS BEHAVIOR!!

Now the second DRO in each area, is for you to send an Integer value (max 16 bit) to the PLC, where you can read it in the "Data View" area, as register values: V11001 (for RTU serial), and V12001 (for TCP/IP), the data view is set to read those values as decimal.
(Data View in the DS5 software for the PLC).

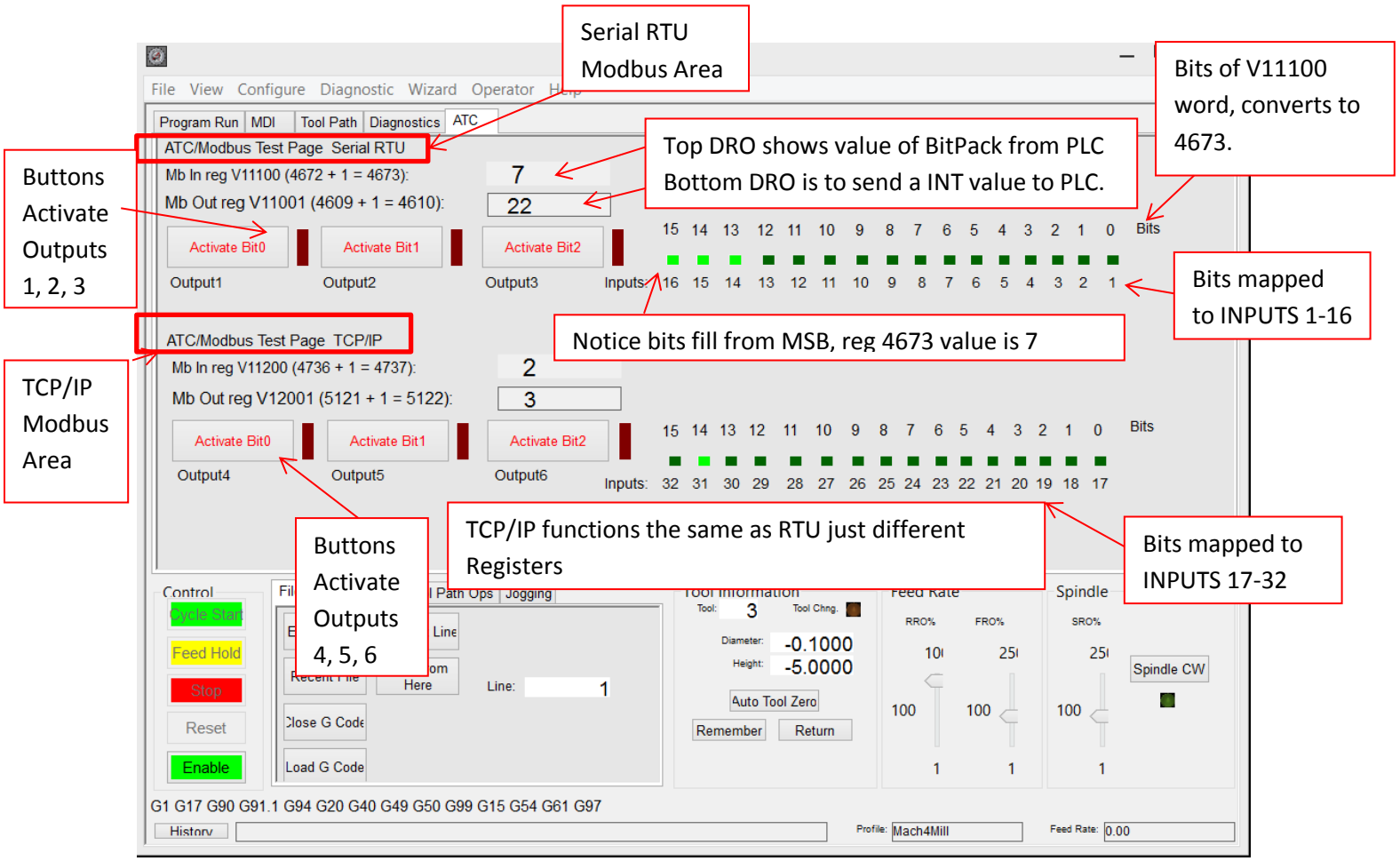


V11000 is the Serial RTU register that shows the "Sum" value for the bit-packed register in the PLC: B11000.0-15 this is where M4 is setting Outputs1-3

V11001 is the Int value sent from M4
V11100 is the "Sum" of the bit-packed value that the PLC is sending to M4 (the little LEDs on the ATC page).

V12000, V12001, and V11200 is the same as above but for the TCP/IP area on the ATC page.

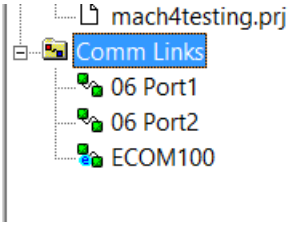
(Screen Shot the ATC page in Mach4)



OK, what's going on with all those crazy numbers!!! The PLC uses "Octal" addressing, and that has to be converted to "Decimal" for the Modbus plugin to use it, **FURTHER, you have to ADD 1 to the converted decimal address** (due to the Modicon standard that Steve is using). So to the left of each of the two DROs in both areas you will see the labels that show the conversion, from PLC address to M4 MB decimal reg with additional 1 added.

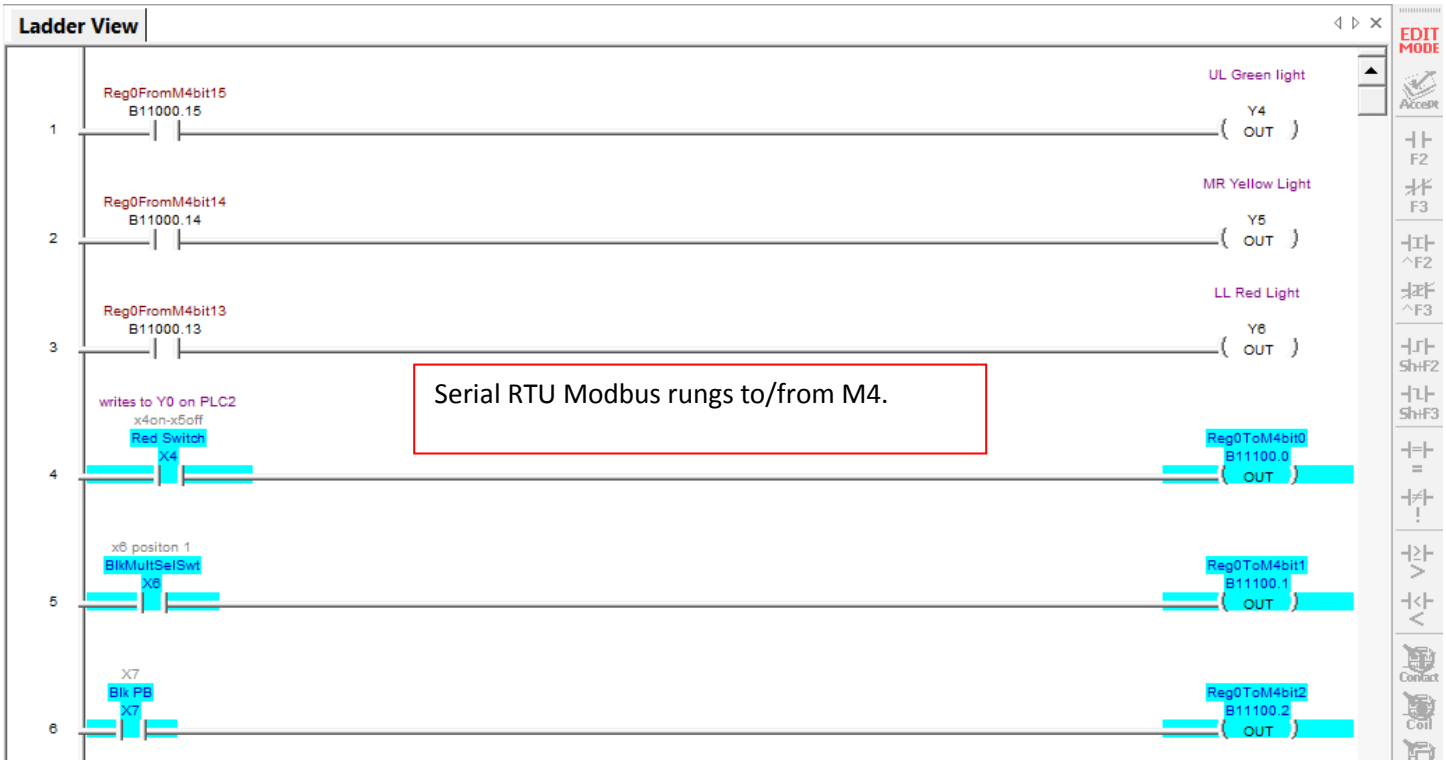
Another Note on Bit-Packing for OUTPUT registers, it also will fill MSB first before it goes out. So, if you have OUTPUT1 mapped to ModbusOutputReg Bit 0, then in the PLC, Bit 15 will turn on.

So now that the screen is explained, let's move on now to the PLC program, we will make the M4 modbus config after we get the PLC program designed. Make sure you have all three com links set up in DS5 software: (Screen Shot of the three com links we are using)

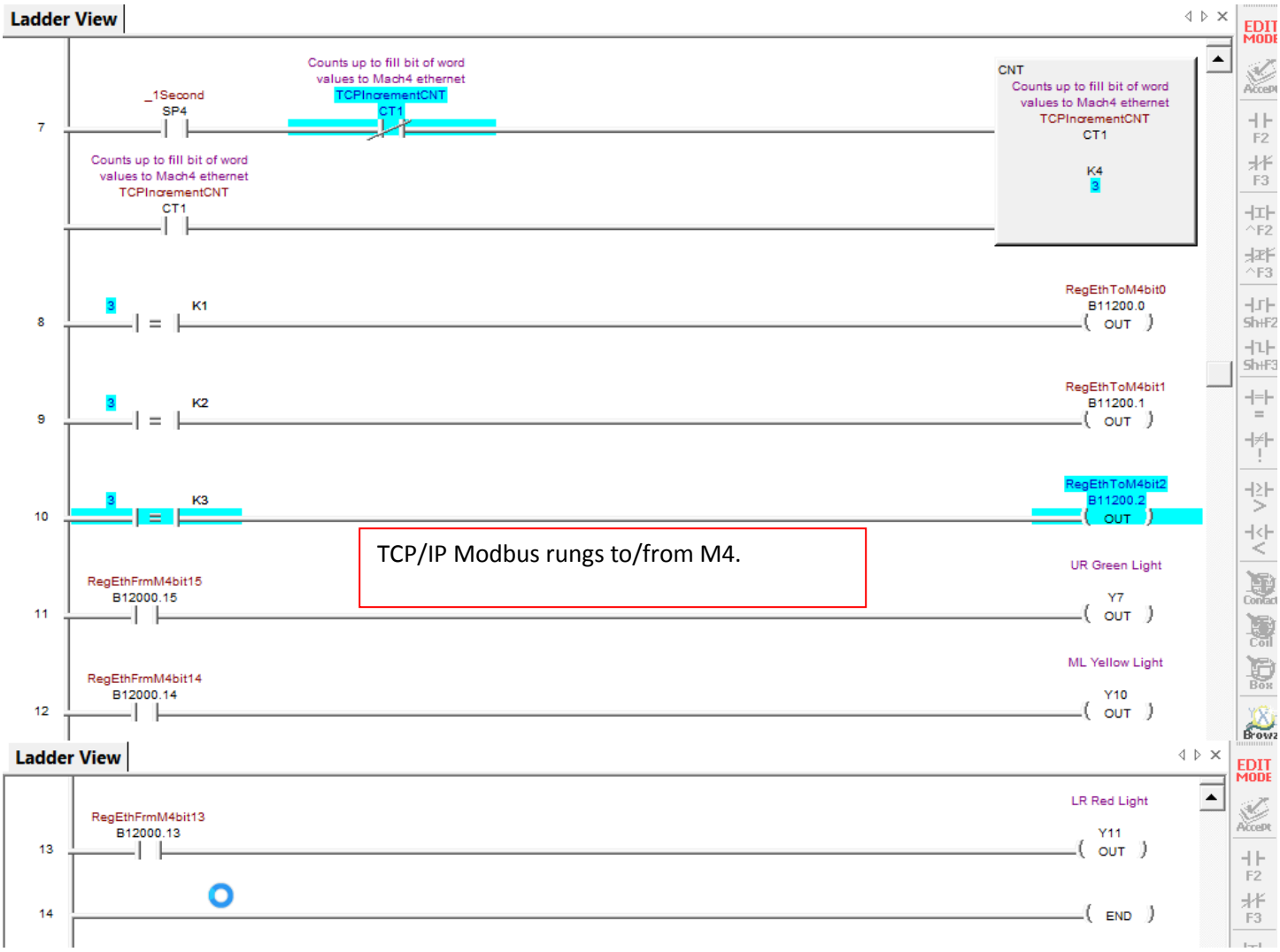


In the Zip file you will have the PLC Project files, you will need to copy their contents to your DS5 "Projects" folder and paste them in. Once you have done that, when you open the DS5 software, you will see the "mach4testing" project, if not pick Open project from the menu and pick it. The following are the screen shots of the PLC program itself.

(screen shot of PLC Ladder, To the end of this page is the Serial Modbus Section)



(screen shot of PLC Ladder, To the end of this page is the TCP/IP Modbus Section)



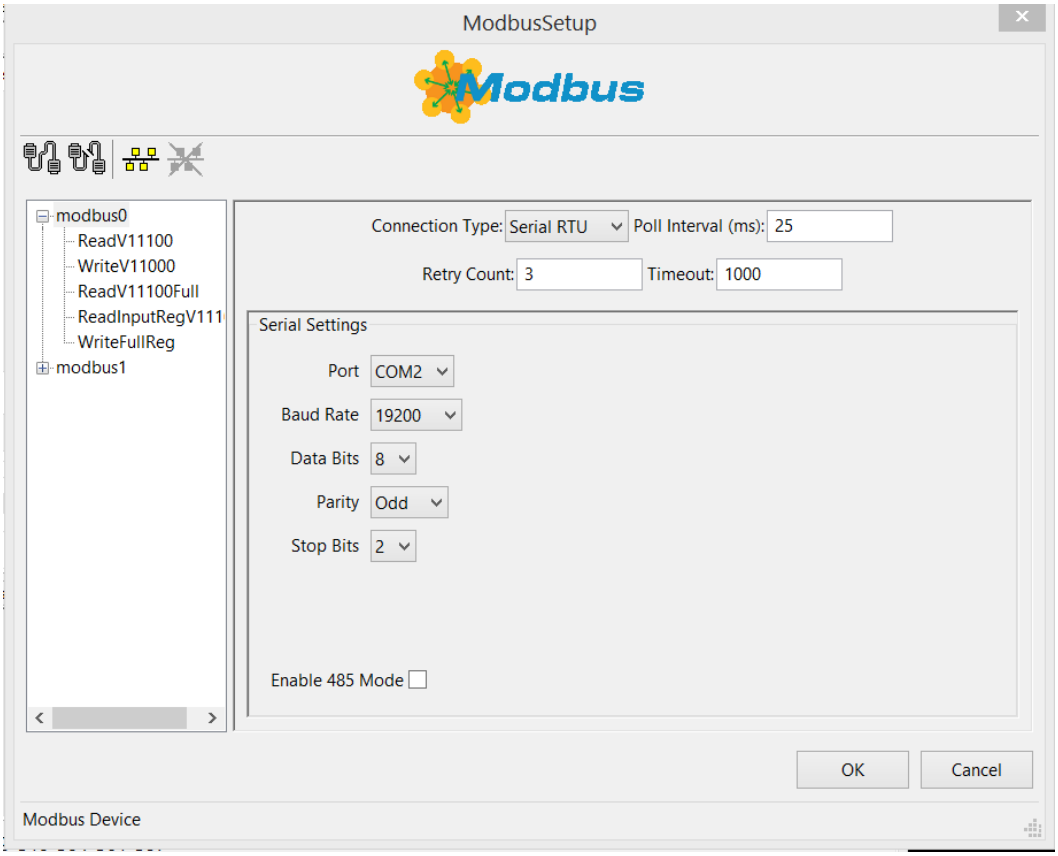
You can see in the above Ladder, that “Bit Of Word” addressing is used for both control bits coming from M4, and going to M4. Also note that we have to use Big Indian (or MSB), both to and from.

Next we will look at the Mach4 Config for Modbus both Serial and TCP/IP.

First we will set up the Serial Com Port for Serial RTU Modbus, in M4 go to Configure->Plugins (you should have M4 in a disabled state). Scroll down to Modbus, and hit the configure button.

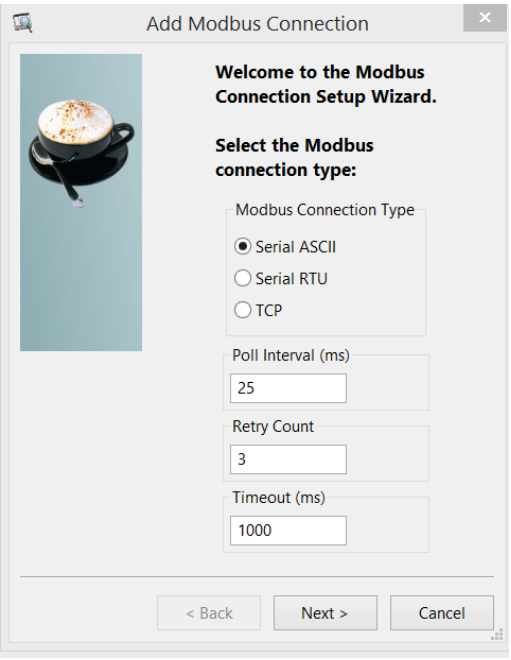
Note: To add a Modbus Communication channel, press the button that looks like a snake with a head on each end, to delete a channel click on the channel and hit the broken snake.
To add a Modbus Function, press the button that looks like a little computer network, to delete a function click on it and hit the button with the big read X on it.

To edit an already existing communication channel, highlight the Mod0 (we will use this for our Serial).
(see screen shot next page).



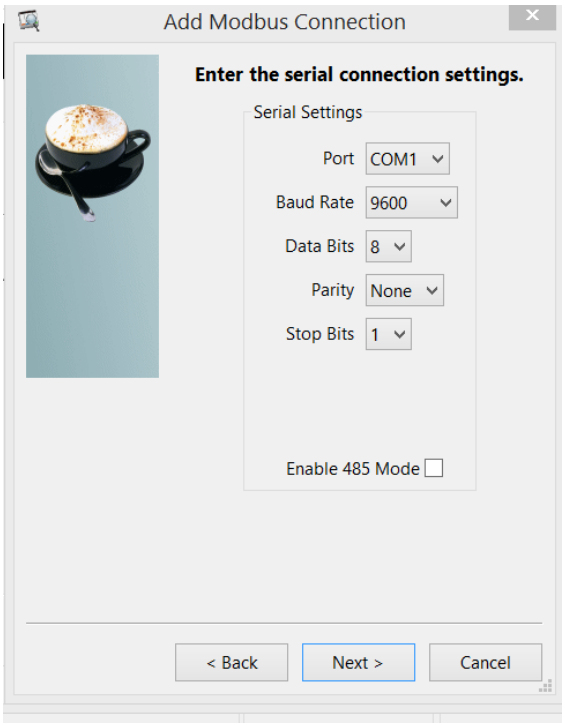
The Config here is the same as in “Port 2” of the DL06.

To add a communication channel push the button that looks like a Snake with a head on each end.

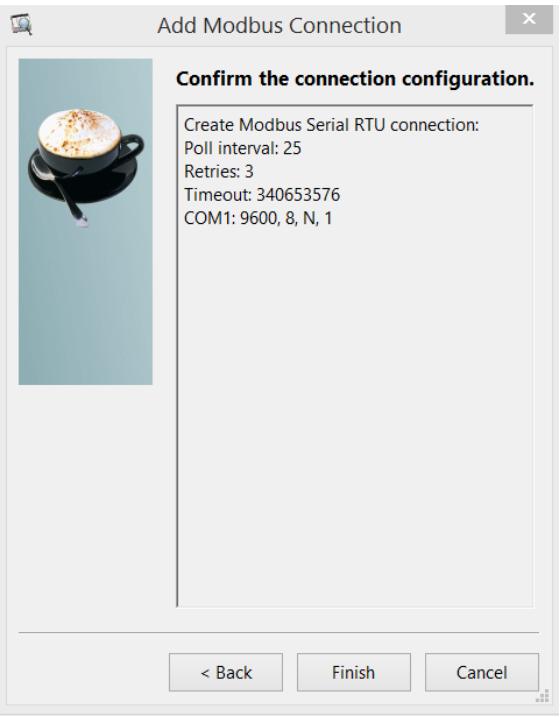


Here you pick what kind of Modbus you want to add, Since the Mod0 is serial already, we would use this to add Mod1 TCP. But you can add ASCII if you sending “ASCII” text to a device.

For RTU Serial it would be:

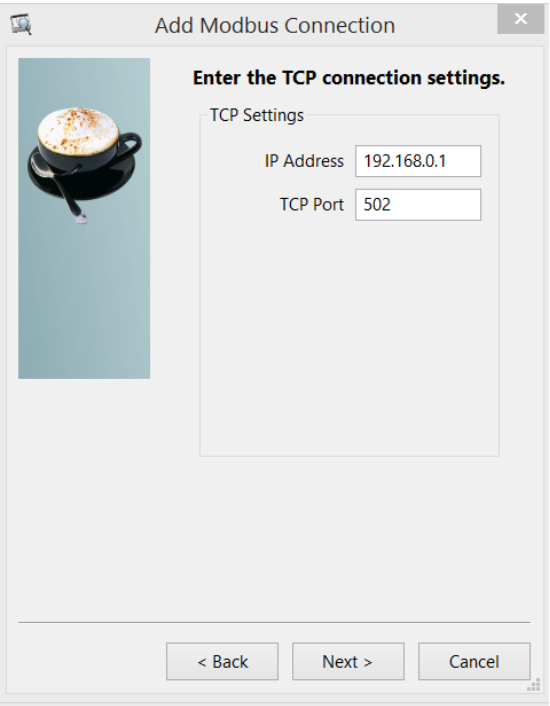


Then:

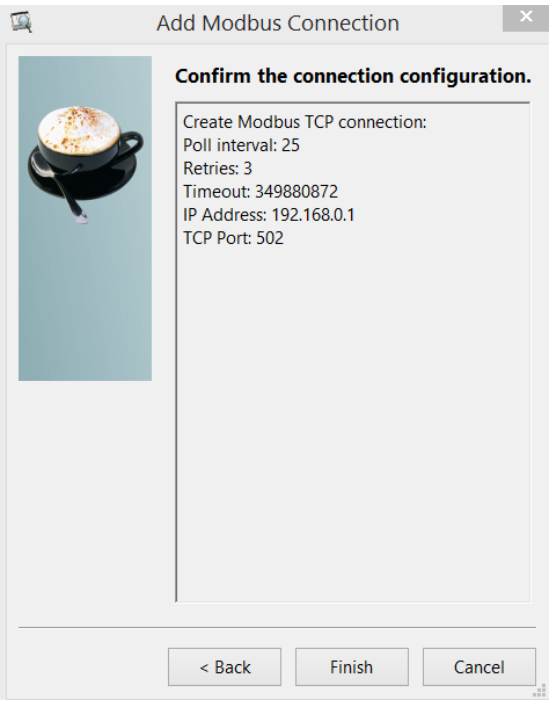


Hit finish and your Serial MB com port is added.

For TCP/IP

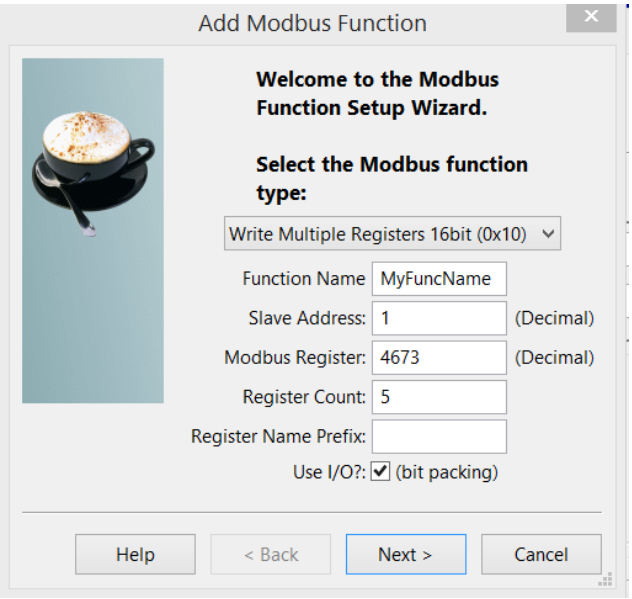


Then:

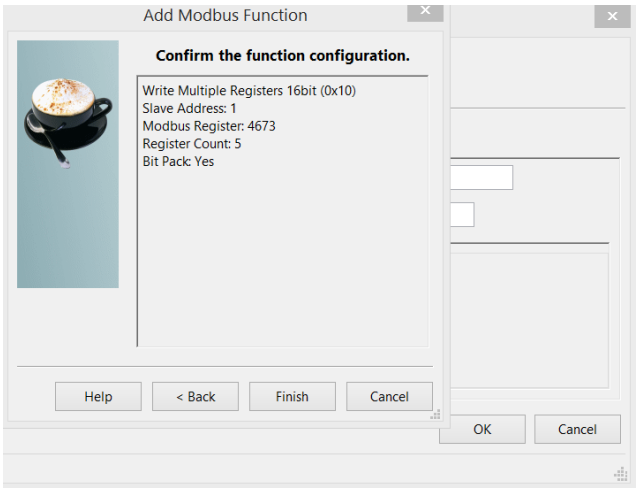


Hit finish and your TCP/IP MB com port is added.

Next is how to add a Modbus function, click on the Modbus channel you want to add a function to, i.e. mod0, or mod1, then hit the button that has computers networked together pic.

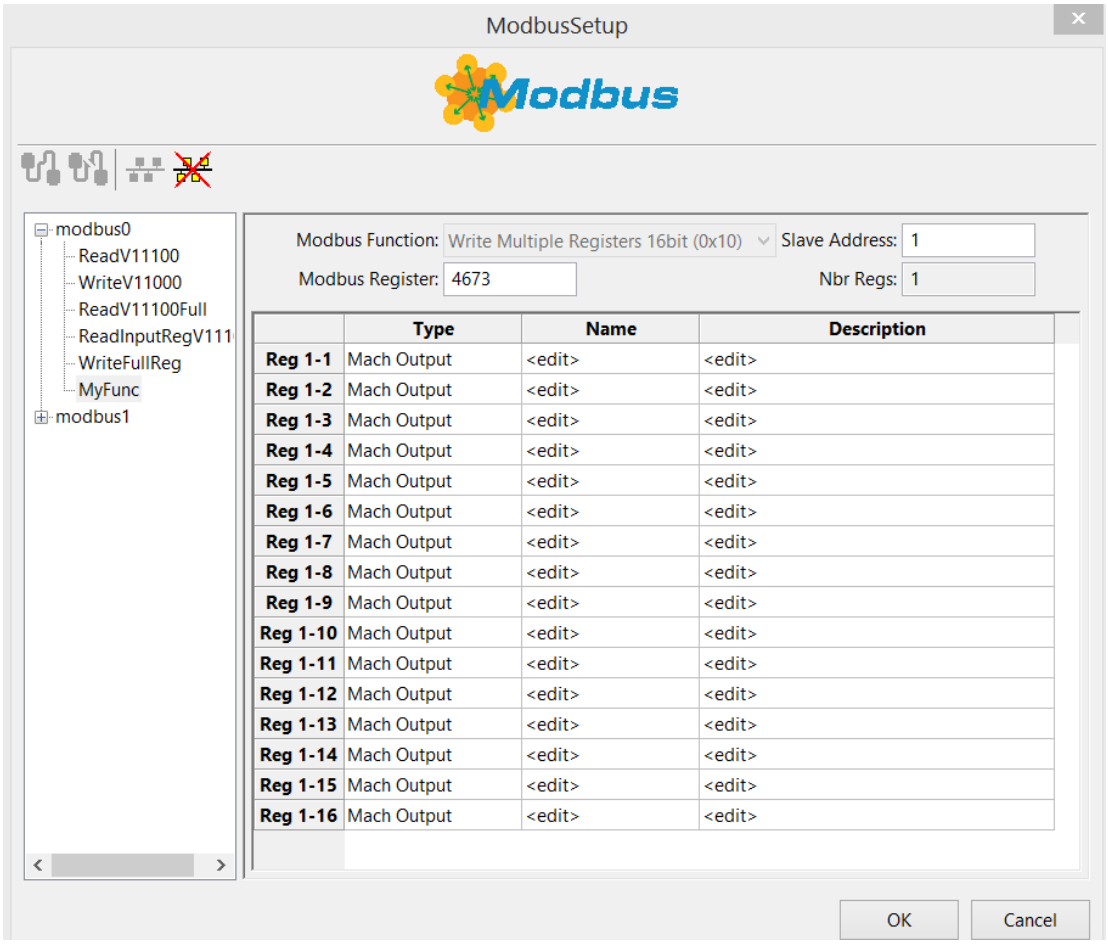


Select the Modbus Function type, then name your function, put in the slave address of your device, put in the Devices address converted to its equivalent decimal value with “1” added to it. If using multiple registers, pick how many to use, the Modbus Register address becomes the “Start” Address, Tick the Box “Use I/O?” (Bit Packing). Hit next.

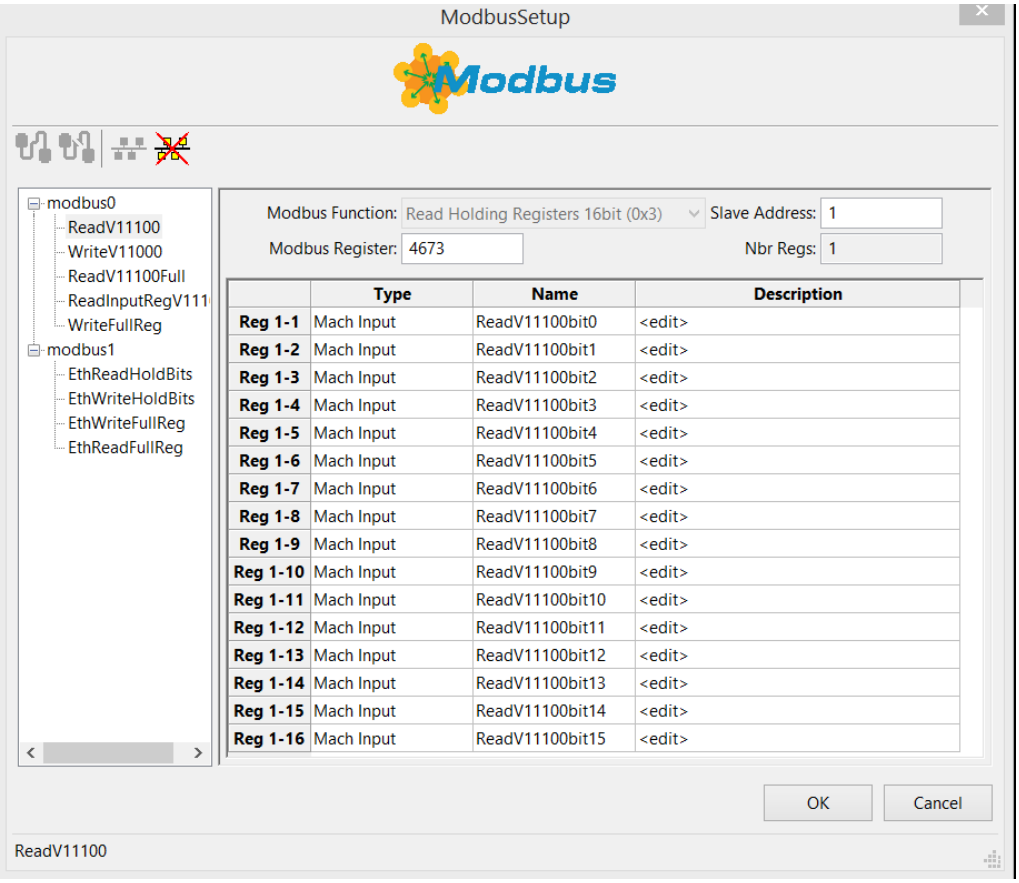


Confirm, by hitting Finish, if wrong hit the back button, get out by cancel.

Next if you would have picked Bit Packing on a Write Multiple registers for instance, it would bring up this page. Here the “NAME” section is the most important, this is where you name the I/O, and that is what you will map into or out from M4. The description is optional but at least put a “na” in each slot so you can move from func to func without have to close and reopen the plugin config.

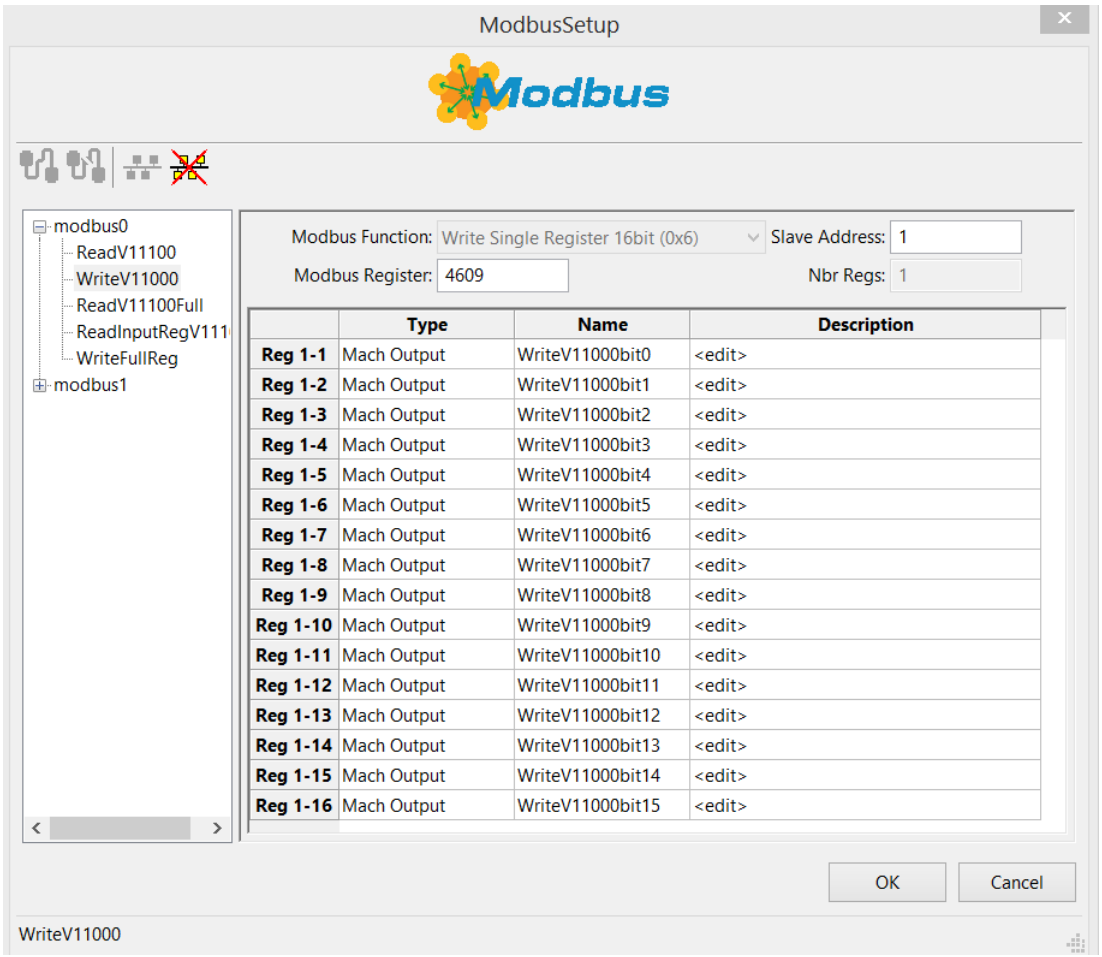


This Serial RTU Read Register is reading the value in the PLC at its address of V11100 and that value coming in is Bit-Packed, its address converts to 4673 (using the rules stated above). Under Name, the "ReadV11100bit0- ReadV11100bit15" is mapped to Mach4 input Reg 1-1 to 1-16 respectively, we will map this and the following I/O to Mach4 in the config, after this.
This is our first Read register we are bringing this in as "Bit Packed".
(screen shot, bit packed read reg).

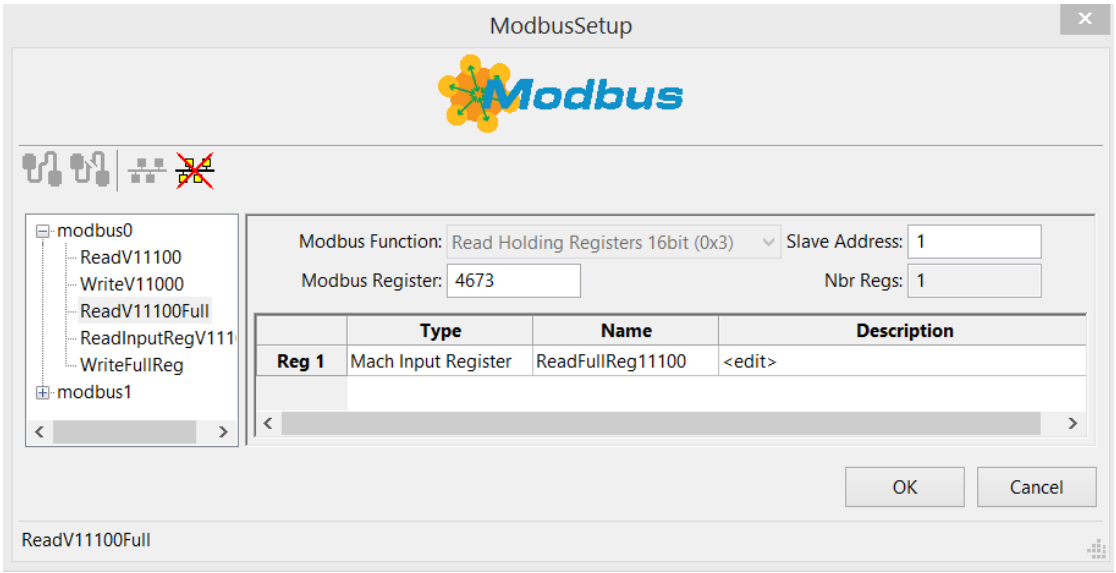


Next page is the Write serial

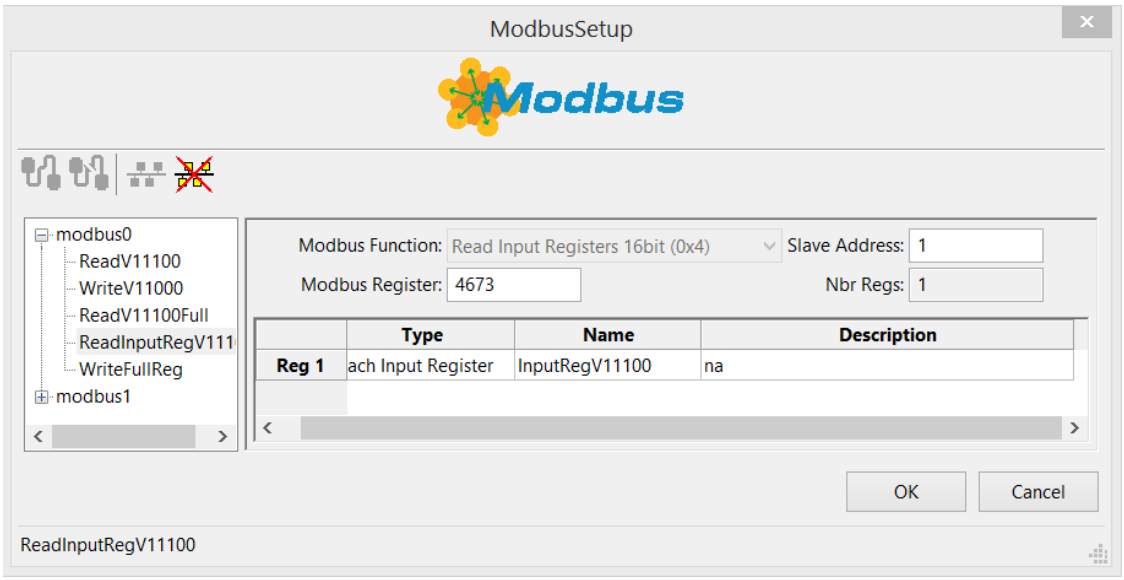
This Serial RTU Write Register is writing the value in the PLC at its address of V11000 and that value going out is Bit-Packed, its address converts to 4609 (using the rules stated above).



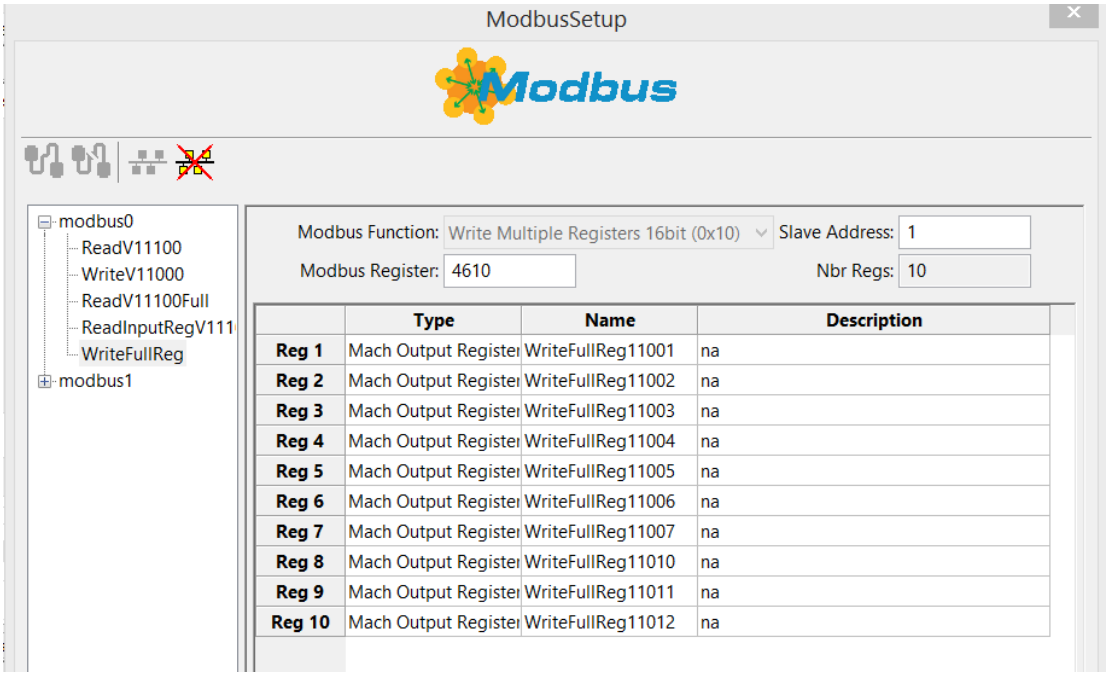
Next is Read Full (Holding) register



Next is read input register (for fun).

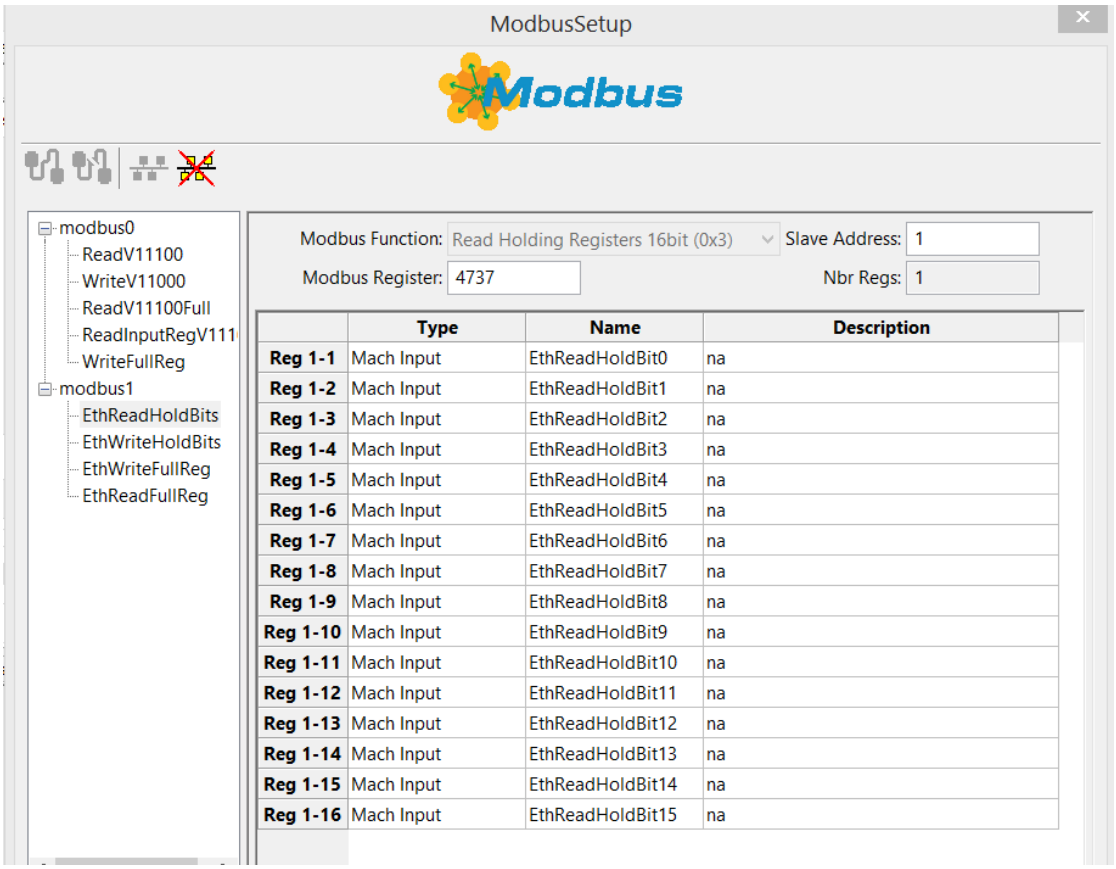


Next is Write full (multiple) reg, Notice under name the numbers, since they represent the actual PLC address range I am talking to, it progress in an Octal sequence.



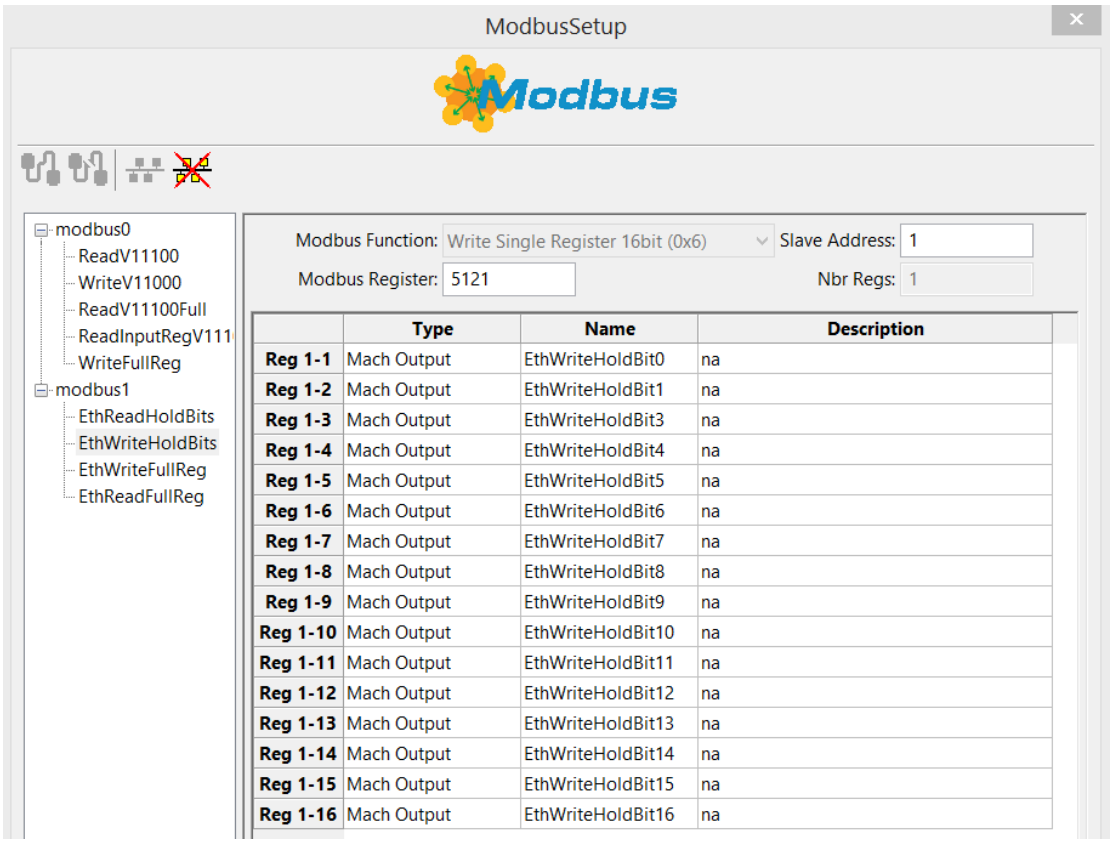
Up above we have already added, the TCP Modbus communication channel, mod1, so now we are going to add functions to it.

First Read Bit Packed TCP.



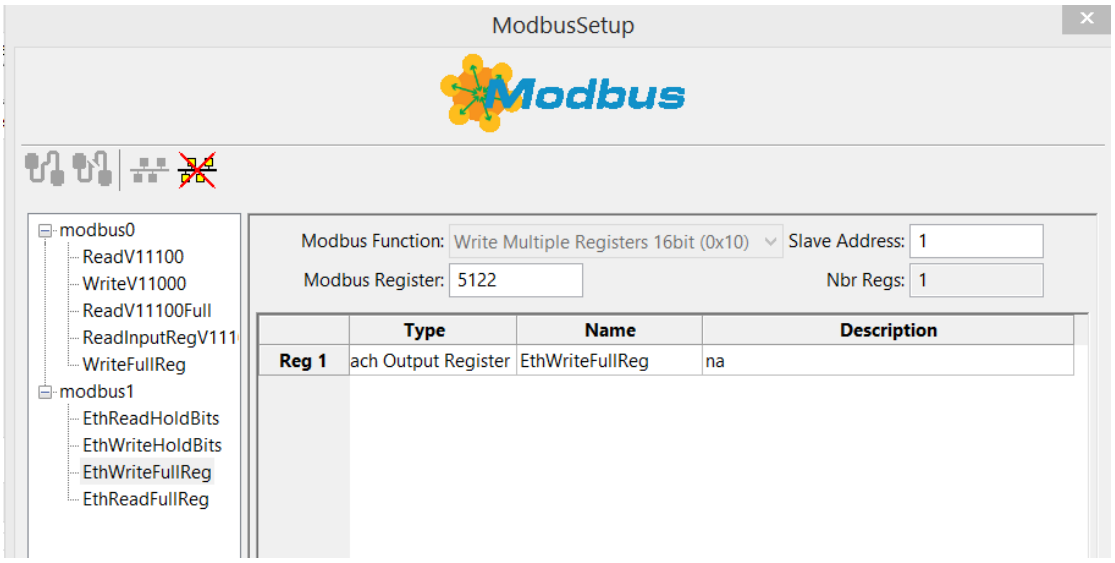
	Type	Name	Description
Reg 1-1	Mach Input	EthReadHoldBit0	na
Reg 1-2	Mach Input	EthReadHoldBit1	na
Reg 1-3	Mach Input	EthReadHoldBit2	na
Reg 1-4	Mach Input	EthReadHoldBit3	na
Reg 1-5	Mach Input	EthReadHoldBit4	na
Reg 1-6	Mach Input	EthReadHoldBit5	na
Reg 1-7	Mach Input	EthReadHoldBit6	na
Reg 1-8	Mach Input	EthReadHoldBit7	na
Reg 1-9	Mach Input	EthReadHoldBit8	na
Reg 1-10	Mach Input	EthReadHoldBit9	na
Reg 1-11	Mach Input	EthReadHoldBit10	na
Reg 1-12	Mach Input	EthReadHoldBit11	na
Reg 1-13	Mach Input	EthReadHoldBit12	na
Reg 1-14	Mach Input	EthReadHoldBit13	na
Reg 1-15	Mach Input	EthReadHoldBit14	na
Reg 1-16	Mach Input	EthReadHoldBit15	na

Write Bit Packed TCP

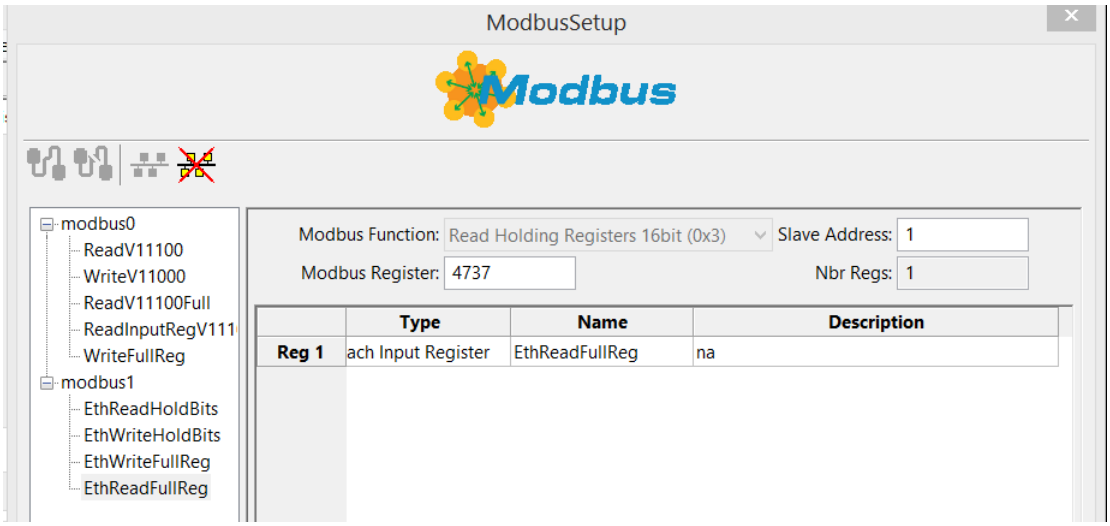


	Type	Name	Description
Reg 1-1	Mach Output	EthWriteHoldBit0	na
Reg 1-2	Mach Output	EthWriteHoldBit1	na
Reg 1-3	Mach Output	EthWriteHoldBit3	na
Reg 1-4	Mach Output	EthWriteHoldBit4	na
Reg 1-5	Mach Output	EthWriteHoldBit5	na
Reg 1-6	Mach Output	EthWriteHoldBit6	na
Reg 1-7	Mach Output	EthWriteHoldBit7	na
Reg 1-8	Mach Output	EthWriteHoldBit8	na
Reg 1-9	Mach Output	EthWriteHoldBit9	na
Reg 1-10	Mach Output	EthWriteHoldBit10	na
Reg 1-11	Mach Output	EthWriteHoldBit11	na
Reg 1-12	Mach Output	EthWriteHoldBit12	na
Reg 1-13	Mach Output	EthWriteHoldBit13	na
Reg 1-14	Mach Output	EthWriteHoldBit14	na
Reg 1-15	Mach Output	EthWriteHoldBit15	na
Reg 1-16	Mach Output	EthWriteHoldBit16	na

Write Multiple Full Reg TCP

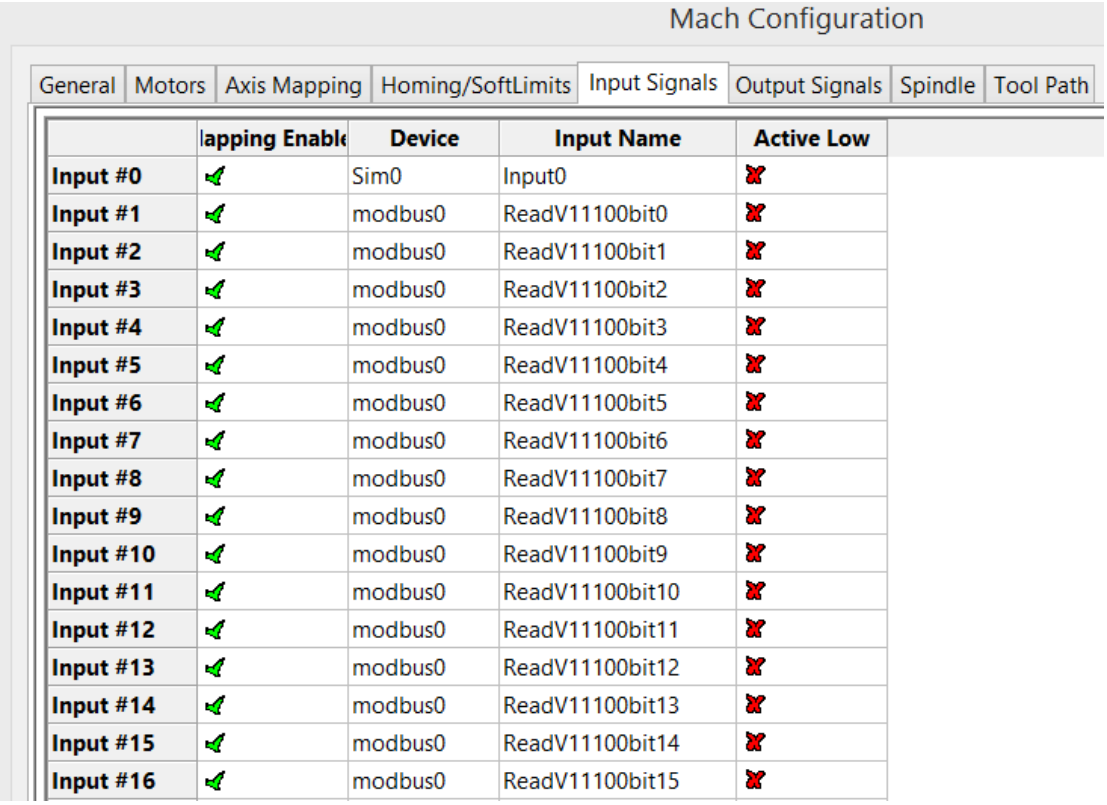


Next Read Holding Reg TCP



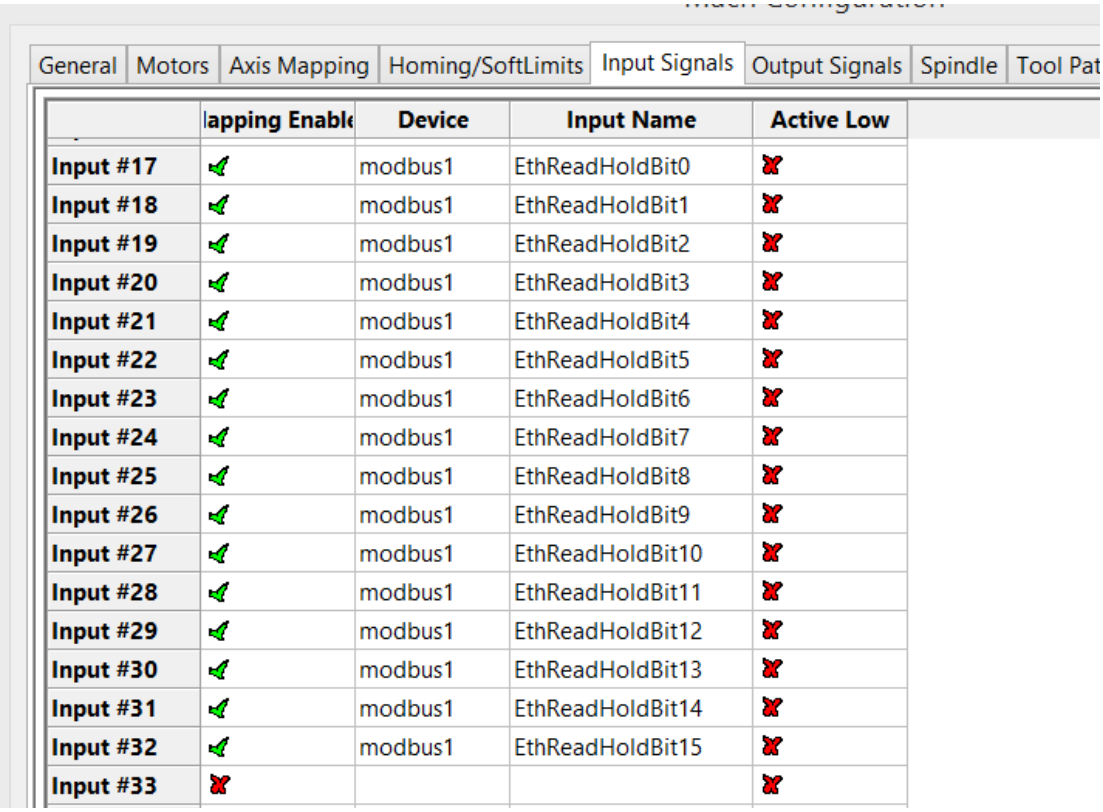
Now we will “MAP” all this I/O into Mach4 (to include screen set). NOTE any time you add or change I/O in the Modbus Config Plugin, you will need to CLOSE and REOPEN Mach4!!! Otherwise that I/O will NOT show up as available in the Screen Designer, or the Mach4 Input/Output config.

So, in M4 menu goto configure->Mach then the Input Signals Tab.
Since Input0 is used by the “Sim” plugin, we will use INPUTS1-16 for the serial bit packed read from the PLC.



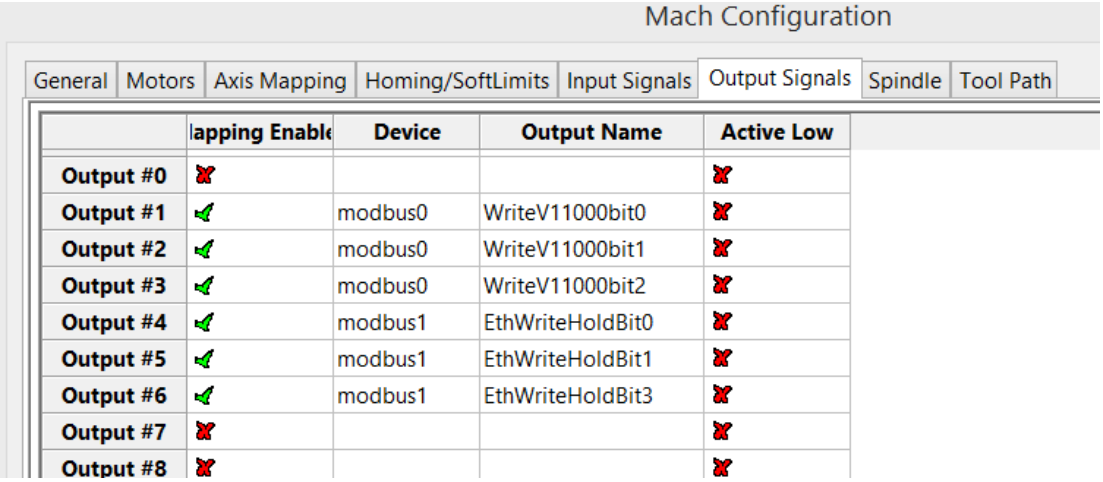
	lapping Enable	Device	Input Name	Active Low
Input #0	✓	Sim0	Input0	✗
Input #1	✓	modbus0	ReadV11100bit0	✗
Input #2	✓	modbus0	ReadV11100bit1	✗
Input #3	✓	modbus0	ReadV11100bit2	✗
Input #4	✓	modbus0	ReadV11100bit3	✗
Input #5	✓	modbus0	ReadV11100bit4	✗
Input #6	✓	modbus0	ReadV11100bit5	✗
Input #7	✓	modbus0	ReadV11100bit6	✗
Input #8	✓	modbus0	ReadV11100bit7	✗
Input #9	✓	modbus0	ReadV11100bit8	✗
Input #10	✓	modbus0	ReadV11100bit9	✗
Input #11	✓	modbus0	ReadV11100bit10	✗
Input #12	✓	modbus0	ReadV11100bit11	✗
Input #13	✓	modbus0	ReadV11100bit12	✗
Input #14	✓	modbus0	ReadV11100bit13	✗
Input #15	✓	modbus0	ReadV11100bit14	✗
Input #16	✓	modbus0	ReadV11100bit15	✗

TCP read reg, bit packed will map to INPUTS17-32



	lapping Enable	Device	Input Name	Active Low
Input #17	✓	modbus1	EthReadHoldBit0	✗
Input #18	✓	modbus1	EthReadHoldBit1	✗
Input #19	✓	modbus1	EthReadHoldBit2	✗
Input #20	✓	modbus1	EthReadHoldBit3	✗
Input #21	✓	modbus1	EthReadHoldBit4	✗
Input #22	✓	modbus1	EthReadHoldBit5	✗
Input #23	✓	modbus1	EthReadHoldBit6	✗
Input #24	✓	modbus1	EthReadHoldBit7	✗
Input #25	✓	modbus1	EthReadHoldBit8	✗
Input #26	✓	modbus1	EthReadHoldBit9	✗
Input #27	✓	modbus1	EthReadHoldBit10	✗
Input #28	✓	modbus1	EthReadHoldBit11	✗
Input #29	✓	modbus1	EthReadHoldBit12	✗
Input #30	✓	modbus1	EthReadHoldBit13	✗
Input #31	✓	modbus1	EthReadHoldBit14	✗
Input #32	✓	modbus1	EthReadHoldBit15	✗
Input #33	✗			✗

Next is outputs that Mach4 will map to a bit packed register going to the PLC, first set of 3 is for Serial MB Outputs1-3, the next set of 3 Outputs4-6 is for TCP. These OUTPUTS are controlled by buttons on the ATC screen page.

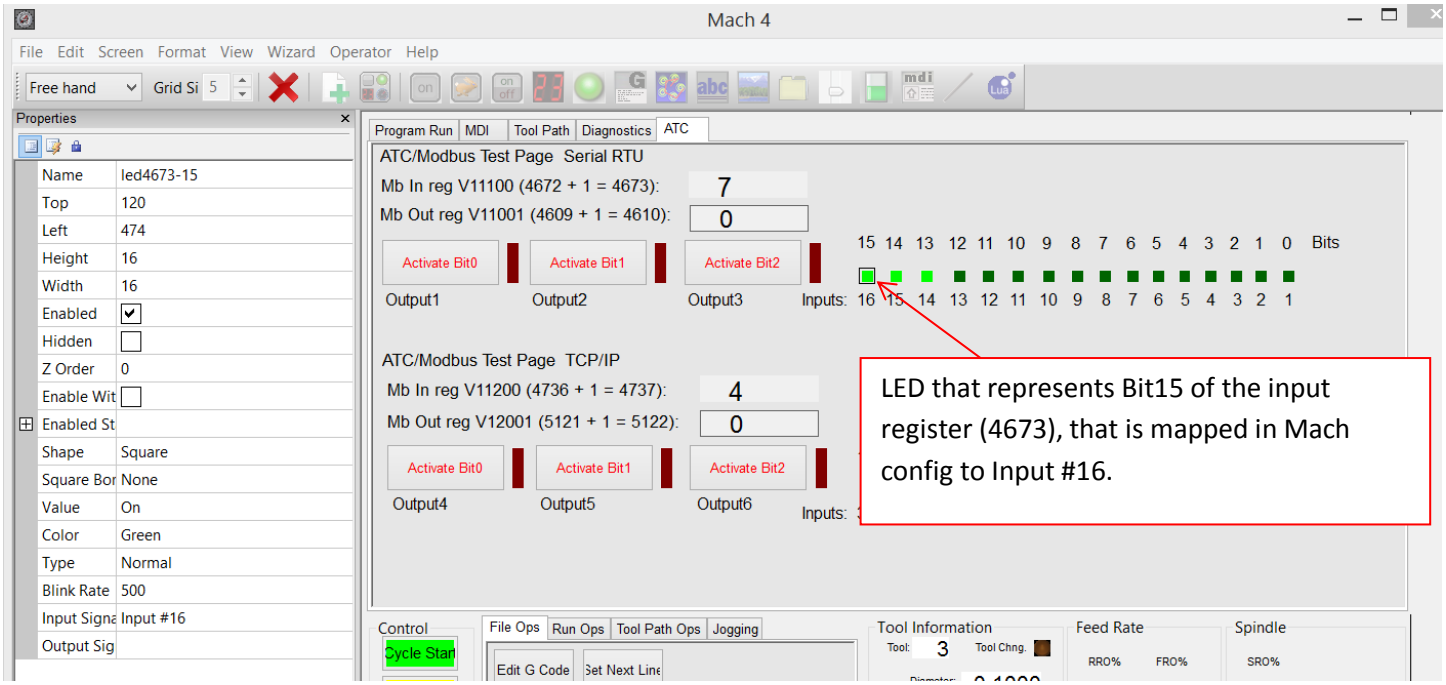


	lapping Enable	Device	Output Name	Active Low
Output #0	✗			✗
Output #1	✓	modbus0	WriteV11000bit0	✗
Output #2	✓	modbus0	WriteV11000bit1	✗
Output #3	✓	modbus0	WriteV11000bit2	✗
Output #4	✓	modbus1	EthWriteHoldBit0	✗
Output #5	✓	modbus1	EthWriteHoldBit1	✗
Output #6	✓	modbus1	EthWriteHoldBit3	✗
Output #7	✗			✗
Output #8	✗			✗

NOTE: Serial MB outputs are modbus0, WriteV11000bit 0-2, for TCP MB outputs are modbus1, EthWriteHoldBit0-3.

Tying all this into a screen set for control, I am not going to go into how to make a screen set. I am just going to show how to tie I/O to a screen set you can look at the screen set in the screen designer that is in the Zip to see how it is done.

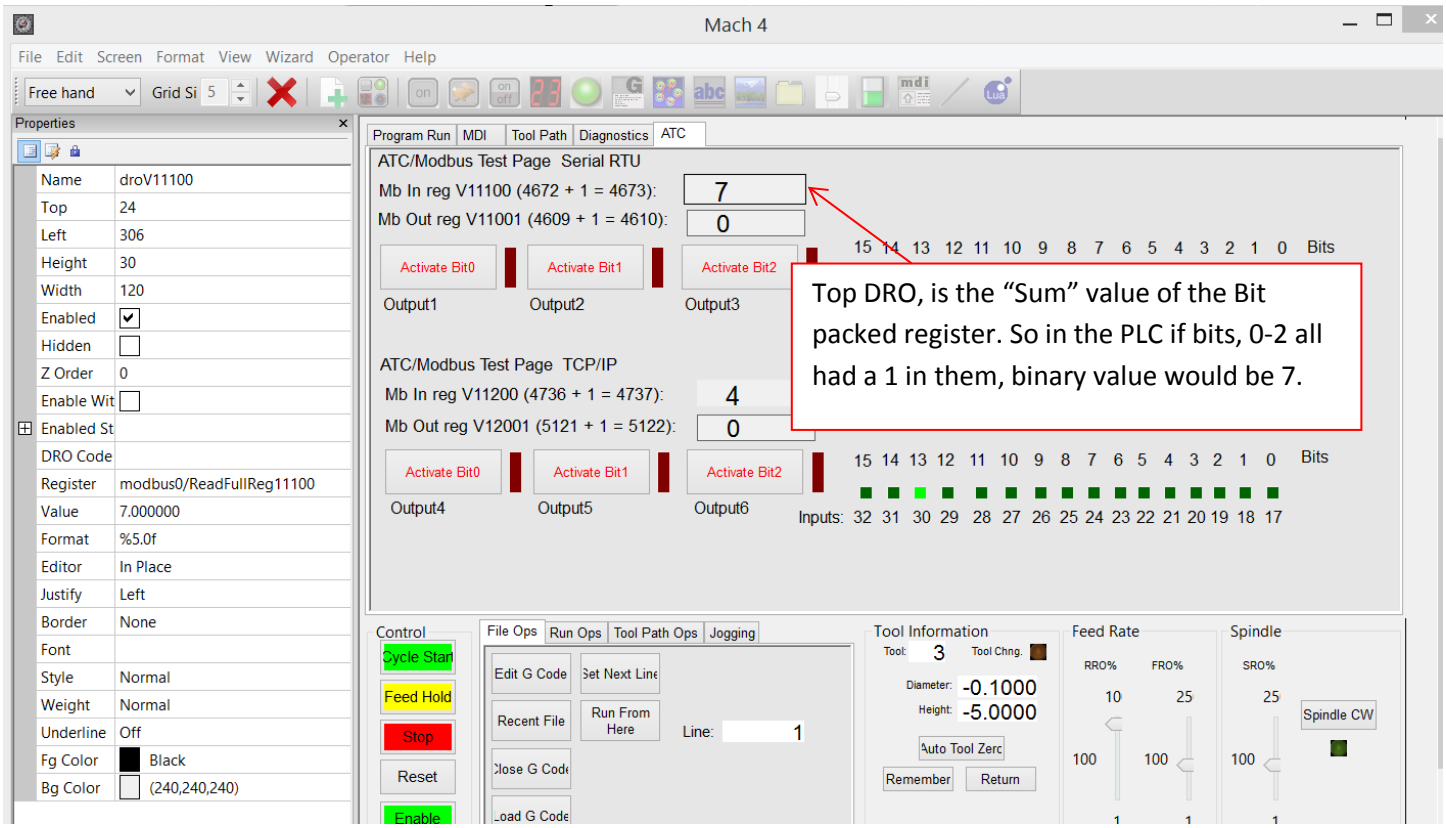
I will show an example of each element type on the ATC screen page. We will be picking from the "Upper" section of the "Serial RTU Modbus".
The first element will be the "Bit 15" LED, (that is also mapped to INPUT16), when in the PLC Bit0 of PLC word V11100 (or Bit of word, B11100.0) is set to one, since M4 brings this in MSB, it will light, Bit15 on the ATC page in the Serial section.



See Properties Column to the left above: Under Name the LED is named "led4673-15" this is so I know what register number and bit it is tied to, but you can call it Popcorn if you want, it doesn't matter, only that it is unique, and you can address these screen objects by name using the screen property calls… (not covering that). Toward the bottom you see the property "Input Signal", and in that value is "Input#16" so, from the Mach config where we mapped the MB input to this input, when it goes hot the LED turns on.

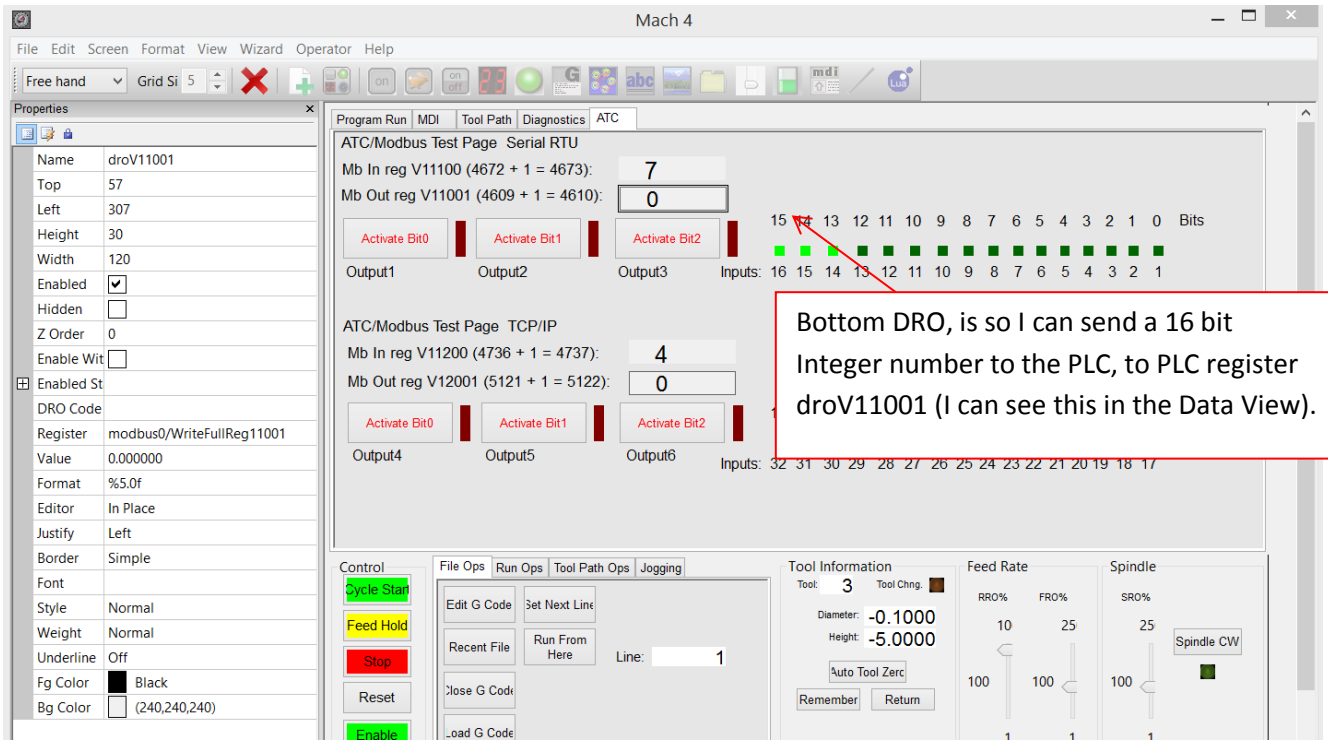
Next page is "Top DRO"

Top DRO:



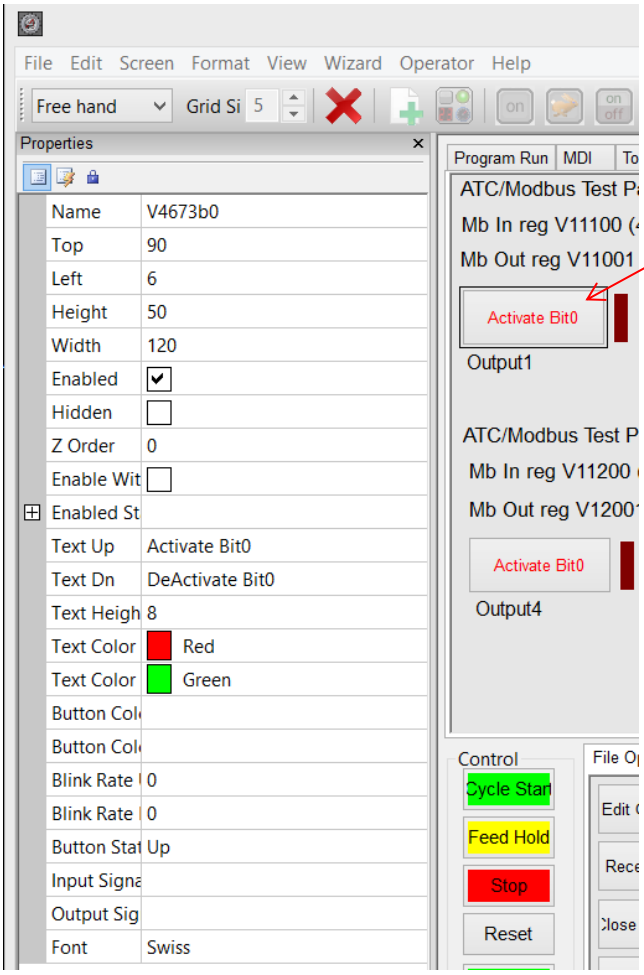
Properties column above, Name is "droV11100" This is so I know what Register I am reading in the PLC. The "Register" property shows what is feeding this DRO: "modbus0/ReadFullReg11100".

The next is the "Bottom DRO"



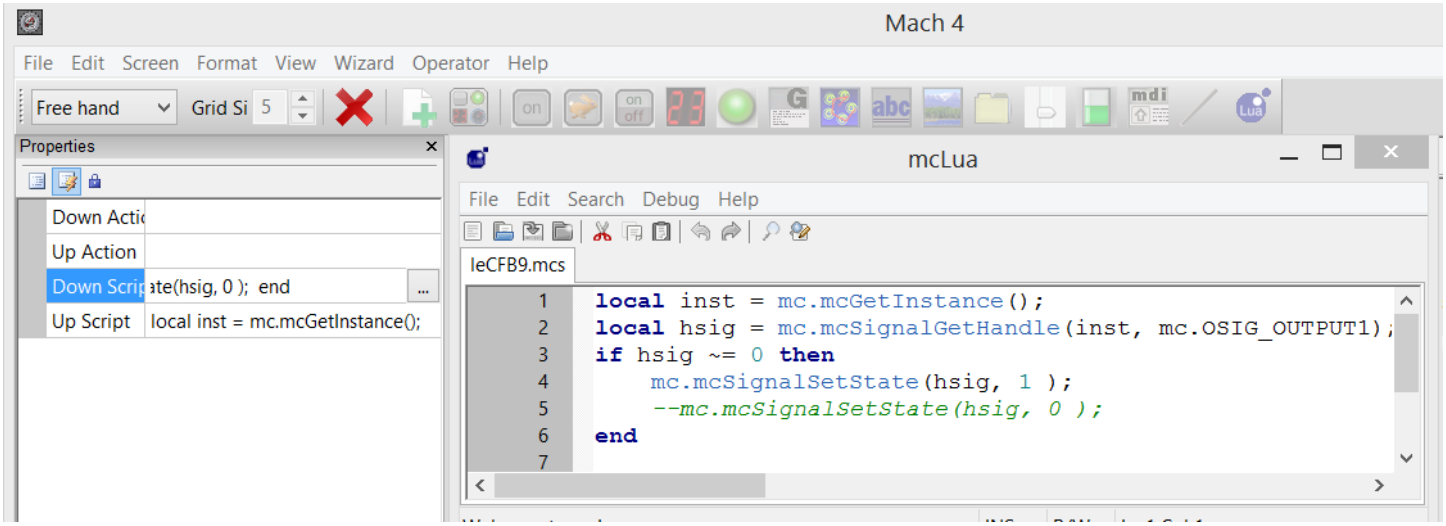
Name is: "droV11001", Register that it writes to and is connected to is: "modbus0/WriteFullReg11001".

Next is how to use a Button in this case a “Toggle Button” to Activate and De-Activate an Output signal from mach4 to the PLC so it can turn on a light. First here is the button property.



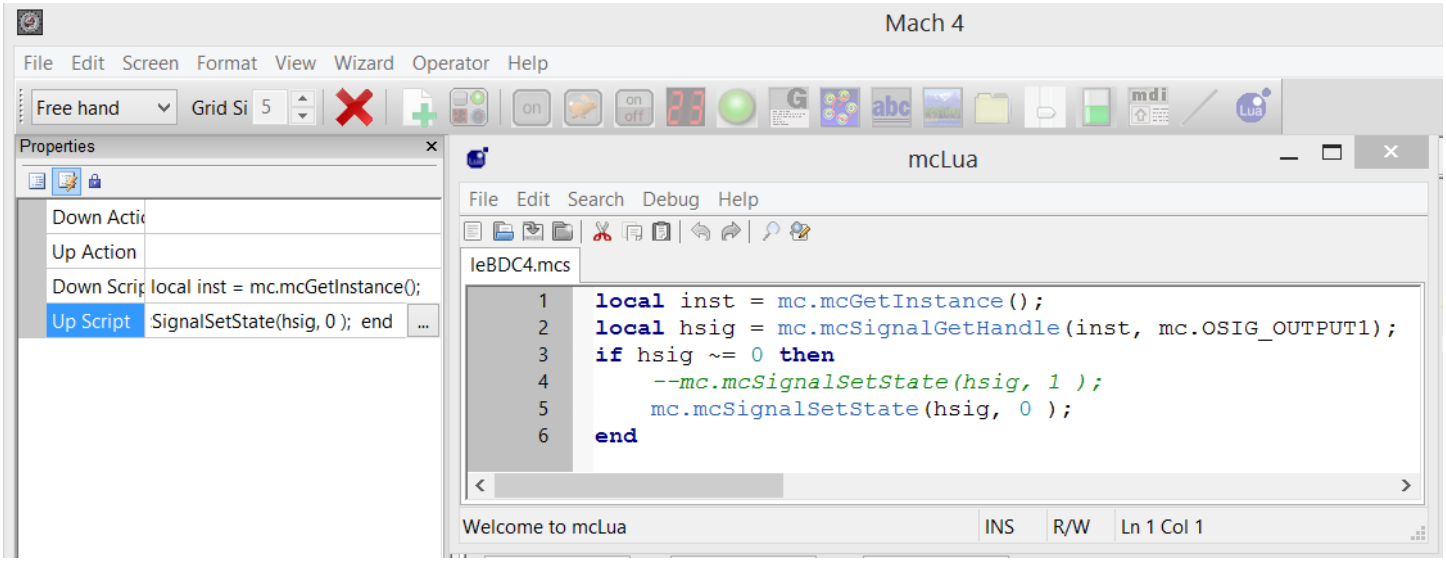
The Activate Bit0 button when pushed down, activates Output1, when Pushed again it De-Activates Output1. In the properties window I name it the bit it controls, also the text changes wording and color depending on its state, pushed or not pushed.

Next is the Code for the “Down” or Pushed state, in the properties window, you have to push the little lightning bolt button to get to the events properties where you can put in code.

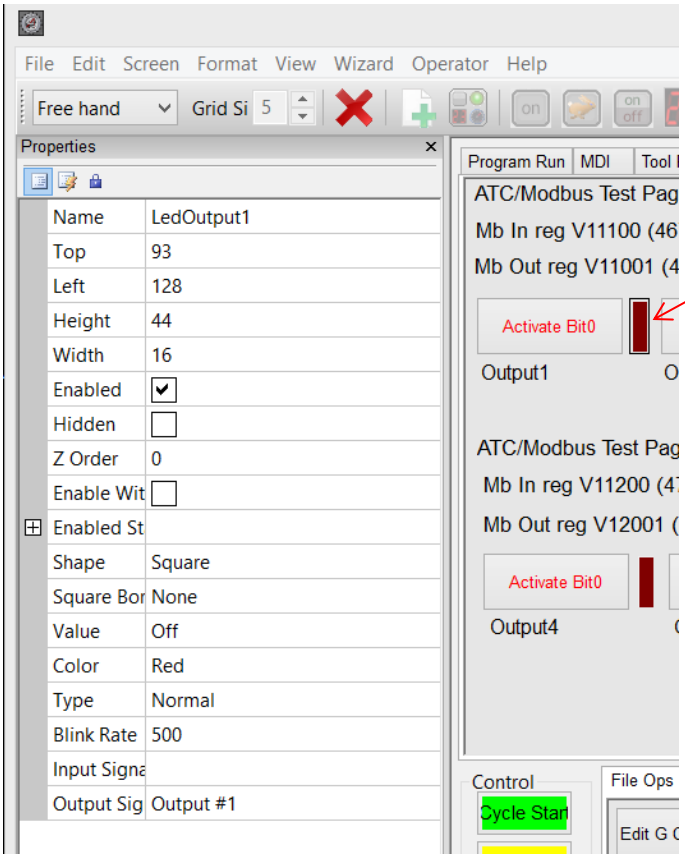


This code will fire once when the button is pushed “down”, the button will stay “down” until pushed again. This code will Turn ON, OUTPUT1 in Mach4, that is mapped to the bit packed write register of the serial MB. Remember it goes out MSB so OUTPUT1 bit0 in Mach4 will go out “backwards” and turn on Bit15 in the PLC, so B11000.15 in the PLC will turn on, which will turn on Y4.

Next code is the Toggle Button UP code that will fire when the currently down button is pushed again, and that will De-Activate OUTPUT1



Last is the Output1 LED, this LED is mapped via the properties to OUTPUT1.



This is the OUTPUT1 LED, it lights up if OUTPUT1 is ON. In the property window I named it “LedOutput1” (kind of catchy huh?) Also in the properties window, “Output Signal” the value is: Output #1

The same pattern is followed for all the other Objects on the ATC page, the TCP section is tied into the TCP MB. Hopefully this has made it all clear as mud!!

Thanks for reading!

Scott “Poppa Bear” Shafer
S S Systems, LLC