Acontis Stack PowerPMAC setups

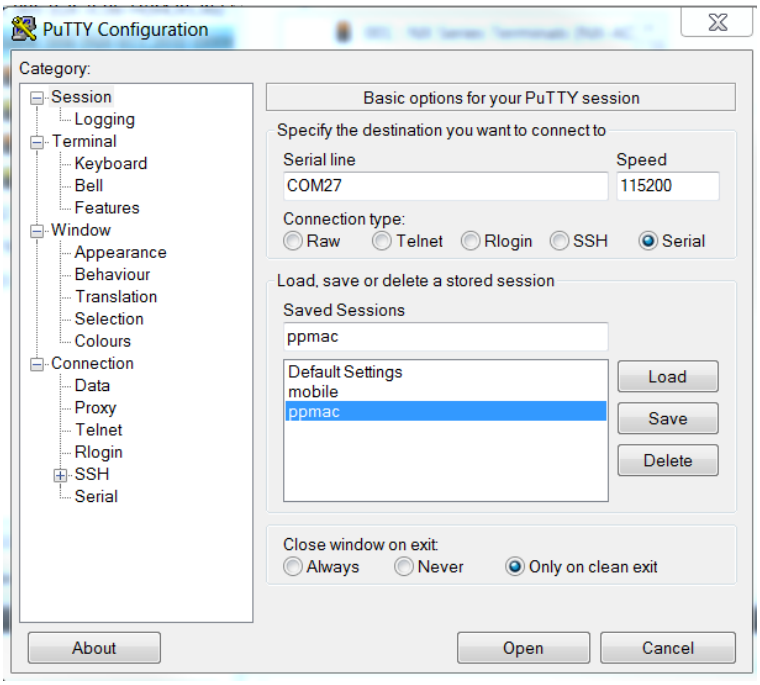
V1.8

November 24, 2017

Davis Gentry

This document is intended to allow an engineer who has basic familiarity with the PowerPMAC and with Sysmac Studio to commission a PowerPMAC using an EtherCAT network using the Acontis EtherCAT stack. At the time of writing the Acontis stack is used in two Omron PowerPMAC products – the µPowerPMAC (CK3E series motion controllers) and the PowerPMAC IPC (NY512-A600 series IPC with motion control). In this document the term 'PowerPMAC' will be used generically to refer to any Omron Delta Tau motion controllers using the Acontis stack.

Note: one powerful method of troubleshooting is to connect a micro USB cable to the diagnostics port on the PowerPMAC (beside the power connector on the CK3E). The PC will detect it as a serial port - COM27 on my PC. Connect using the software tool PuTTy, serial type, 115200 baud. Default settings with the configuration shown will work:



Once it starts, press the PC enter key once, log in with ID: root, Password: deltatau

Step 1

Install the latest PowerPMAC IDE. As a part of that EC Engineer will be installed – this is the software needed to set up the Acontis stack.

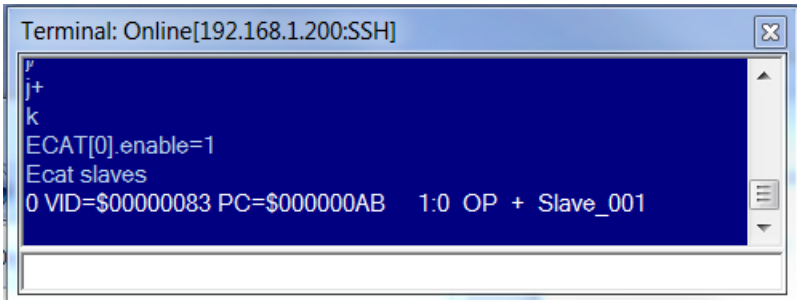
Open the PowerPMAC IDE and connect to the PowerPMAC from your computer. I suggest opening a Watch window and adding ECAT[0].Enable and ECAT[0].MasterState. Make sure that ECAT[0].Enable=0 before making any changes to the EtherCAT configuration (can be changed in the Terminal window if connected). Once the EtherCAT configuration is set and downloaded to the PowerPMAC, and a 'save' (write working memory to permanent memory) and '$$$' (reset)

has been done ECAT[0].MasterState should be greater than 0. 1, 2, or 8 are acceptable values. After setting ECAT[0].Enable = 1 from the terminal window (or code) there will be a brief delay, then you will see ECAT[0].Enable=1 in the Watch window. If there are any errors, or if the network is incorrectly set up, then ECAT[0].Enable will immediately reset to zero after you set it to one. This typically means that something about the EtherCAT configuration in the PowerPMAC does not match the configuration in the EtherCAT network. It could be the clocks, the network order, or anything that causes an EtherCAT error.

Step 2

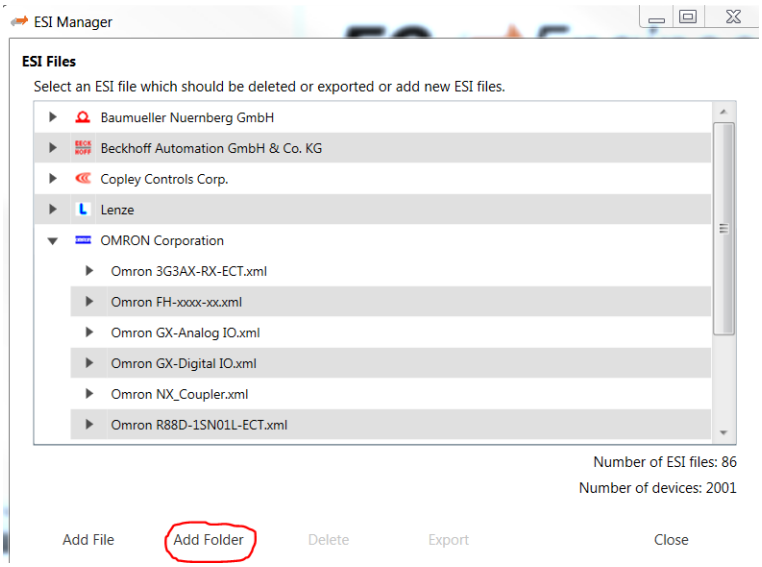
Connect the slave(s) to the EtherCAT network and power the slave(s) and the PowerPMAC.

At this point issuing the 'ecat slaves' (ECAT[0].enable can equal 0 or 1 and still see the same response) from the terminal window should show everything on the network. Vendor IDs and Product Codes are shown under the hardware sections.

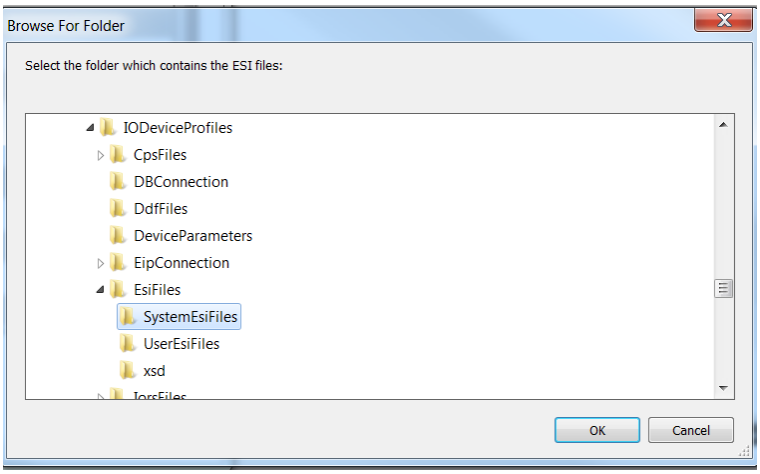


EC Engineer

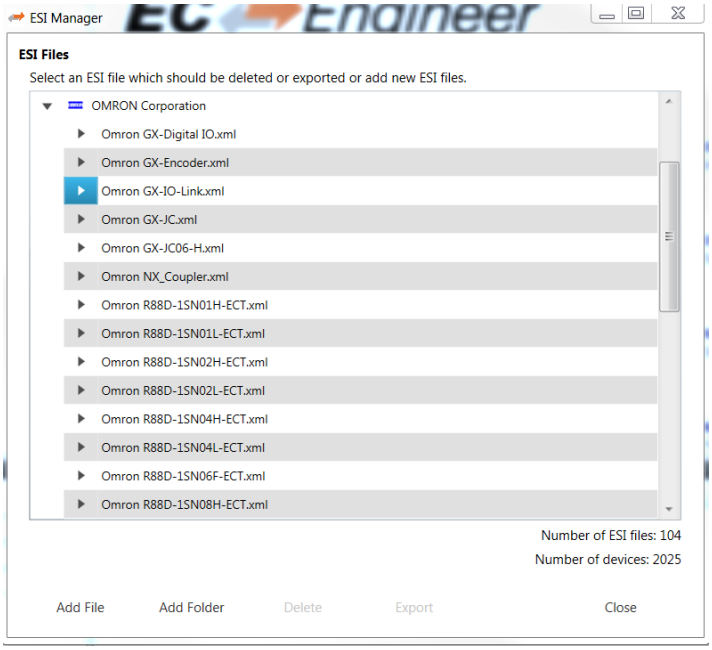
Make sure that you have added the Omron ESI files – the best way is to go to the ESI Manager from the File menu item in EC Engineer. Click the 'Add Folder' button at the bottom.



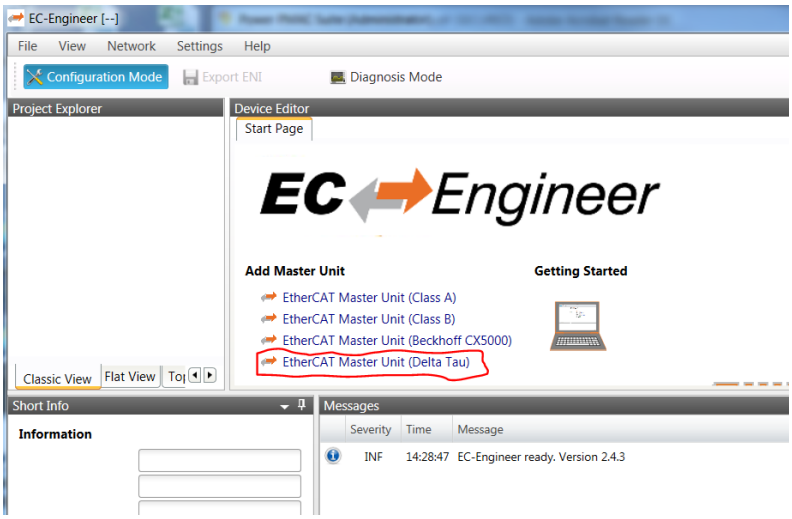
Navigate to C:\Program Files (x86)\OMRON\Sysmac Studio\IODeviceProfiles\EsiFiles and choose the SystemEsiFiles directory.



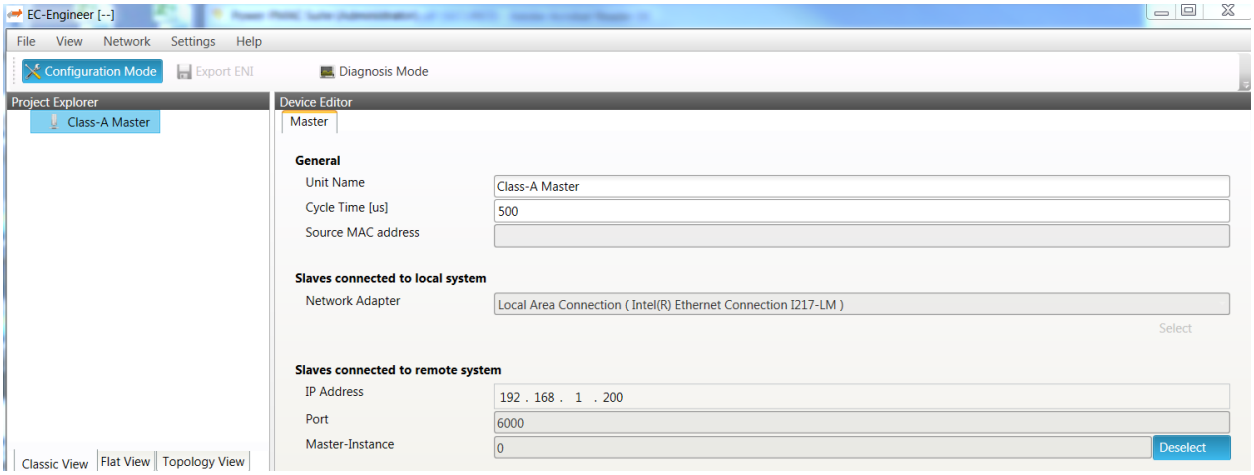
It may tell you that one or more ESI files already exist and ask if you want to add them. Say yes. It may have errors in some of the ESI files – for example the Omron E3NW-ECT.xml file cannot be read at the date of writing. If you have added several items by hand it may be easier to delete all and start from scratch. When finished you should have all of the Omron devices:



With the PowerPMAC and the EtherCAT devices connected and powered open EC Engineer and choose EtherCAT Master Unit (Delta Tau).



Click on the Class-A-Master in the Project Explorer (upper left pane) and set your network cycle time in microseconds – note that the example below is set to 500 usec, or 2000 Hz. This should match the value in Sys.ServoPeriod in the Power PMAC – if Sys.ServoPeriod = 0.5, then cycle time should by 500 usec. Also set the IP address of your PowerPMAC – note that 192.168.1.200 is the default address for the PowerPMAC. Then click the 'Select' button to the right of the 'Master-Instance' block in the 'Slaves connected to remote system section'.

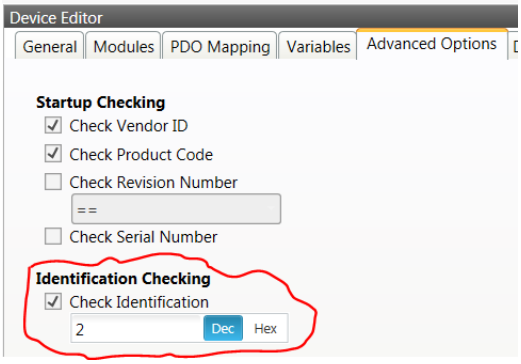


Add a line to the Global Definitions.pmh file to set the servo (and EtherCAT) frequency. This must match the value used in EC Engineer. This will look like:

```
Sys.ServoPeriod=0.5  //2 kHz   1.0  //1kHz
```

Remember to add a startup PLC (see appendix D) when done.

On ALL devices it is best to set the Check Identification on the Advanced Options tab. This forces it to check to make sure that the node is in the correct position on the network.



Some gotchas:

1) Make sure that the node address switches are set. For best results be sure that the nodes number from 1 up, with node 1 providing the clock and being the first node in the physical network.
2) Node 1 should always be set up as a Potential Reference Clock.

Step 3

Set up your various hardware as in the next sections.

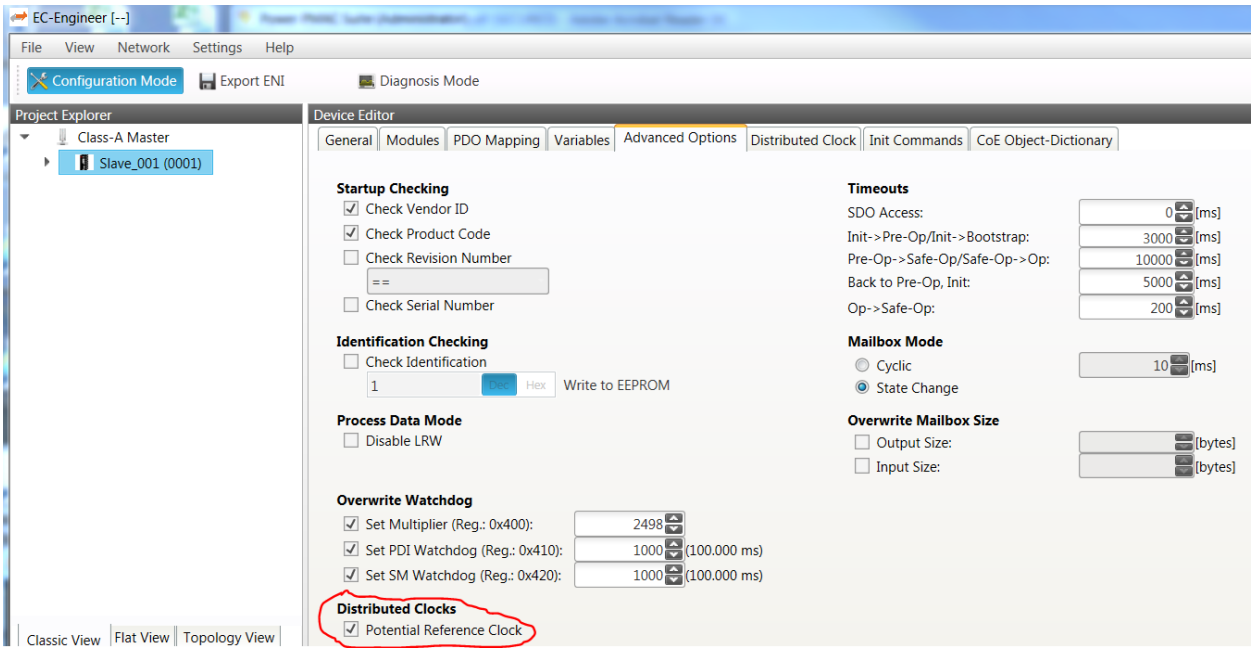
1S and G5 Drives

If the drive has previously been connected to a controller do an initialization (see appendix A). You will also need to do this before changing EtherCAT update frequencies. Using Sysmac Studio, set any desired parameters in the drive (Appendix B). You will probably need to change the Communications error counts (either 0 or f). Set the desired nodes using the rotary switches on the drive. Note that the motor setup utility in the PowerPMAC IDE gives unreliable setup parameters in some cases – easiest to set everything manually. Example settings are given in this document.

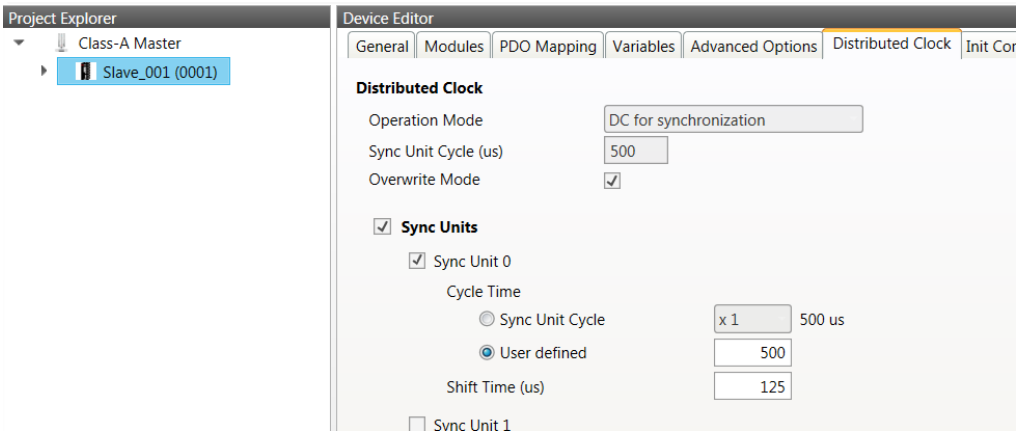
The 'ecat slaves' command with a 100 W 100V 1S drive will show Omron VID = $83 (0x83), and PC = $AB (0xAB).

The Cyclic Position mode will get your drives running well, with the only caveat being that they must be tuned in the drive rather than the PowerPMAC. If the application requires very tight coordination of multiple axes during the moves you may want to increase the servo frequency and go to Cyclic Torque control. This will be covered after Cyclic Position setup.

Open EC Engineer and scan the network (Appendix C). Click on the drive (Slave_001 in this example), then click on the Advanced Options tab in the Device Editor pane. Make sure that Potential Reference Clock is checked in the Distributed Clocks settings.

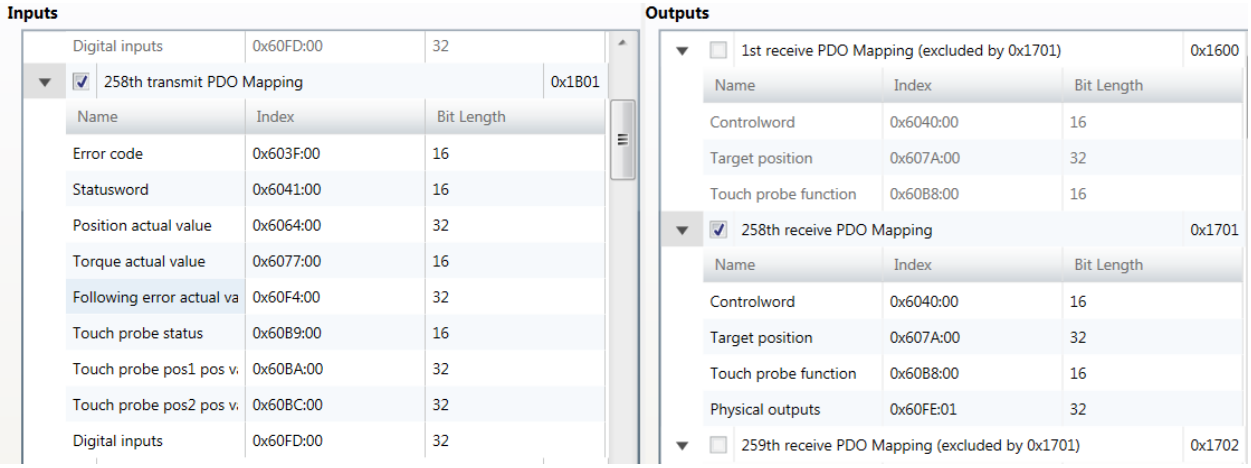


Click on the Distributed Clock tab in the Device Editor pane. You should see that the Sync Unit Cycle (us) is the same as the Master cycle time set prior to the scan. Check the Overwrite Mode box. Under Sync Unit 0 set the Cycle Time to User Defined and make sure that it matches the Sync Unit Cycle (us), then set the Shift Time (us) to 125.00.

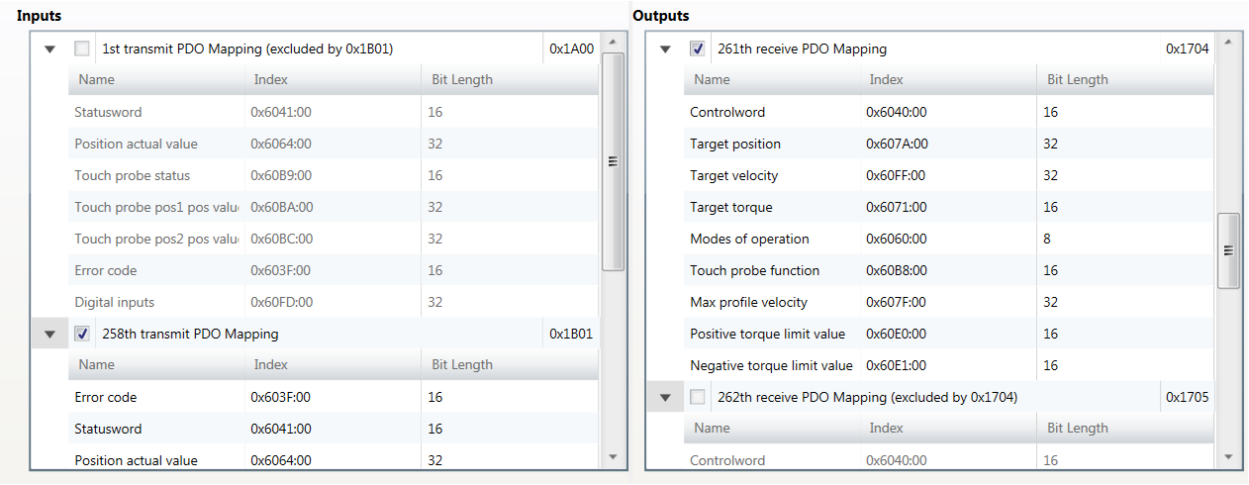


Click on the PDO Mapping in the Device Editor pane. See note 1.

If you are using Cyclic Position mode, no EtherCAT safety, and do not need any data not included in the 258 PDOs, make no changes here. If you plan to use EtherCAT Safety and already have the safety controller set up, add the 273 PDO mappings. The 258 PDO mappings include:

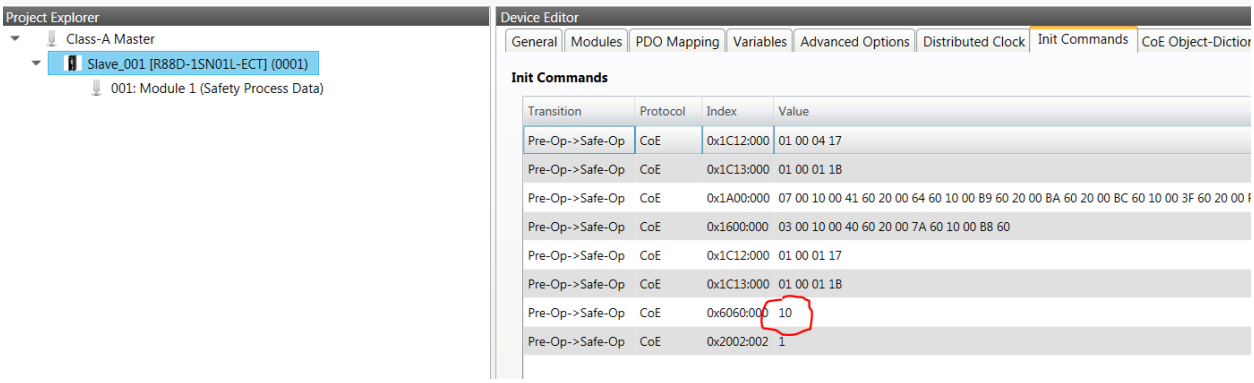


If planning to use Cyclic Torque or Cyclic Velocity control I suggest leaving the 259 transmit PDO mapping, deselecting the 258th receive PDO mapping, and selecting the 261th receive PDO Mapping.



The default is synchronous position mode. If not using synchronous position mode, go to the Init Commands tab, and set the x6060 parameter to according to:

```
8 = Cyclic synchronous position mode (csp)
9 = Cyclic synchronous velocity mode (csv)
10 = Cyclic synchronous torque mode (cst)
```



Use System Setup tool (after enabling Ecat[0].Enable=1) to set the motor up. Choose Amplifier 131xxxx, ID $AB. If cannot get enable to stay on once set up, set Motor[x].EcatAmpFaultLimit to 500 or above. If system setup is giving

problems try closing the IDE and restarting it. If you have used this drive before, especially with NX Safety, you may need to hook up THROUGH an NJ using EtherCAT and Sysmac Studio and initialize, build new project, and enable the drive once before you can get the 87 (ESTO) error to clear and / or the motor to enable.

This assumes that the EtherCAT header file created from the EC Engineer xml file looks something like (sample is 261 receive PDO mapping):

```
#define Slave_0_6040_0_Controlword  ECAT[0].IO[0].Data
#define Slave_0_607A_0_Targetposition  ECAT[0].IO[1].Data
#define Slave_0_60FF_0_Targetvelocity  ECAT[0].IO[2].Data
#define Slave_0_6071_0_Targettorque  ECAT[0].IO[3].Data
#define Slave_0_6060_0_Modesofoperation  ECAT[0].IO[4].Data
#define Slave_0_60B8_0_Touchprobefunctio  ECAT[0].IO[5].Data
#define Slave_0_607F_0_Maxprofilevelocit  ECAT[0].IO[6].Data
#define Slave_0_603F_0_Errorcode  ECAT[0].IO[4096].Data
#define Slave_0_6041_0_Statusword  ECAT[0].IO[4097].Data
#define Slave_0_6064_0_Positionactualval  ECAT[0].IO[4098].Data
#define Slave_0_6077_0_Torqueactualvalue  ECAT[0].IO[4099].Data
#define Slave_0_6061_0_Modesofoperationd  ECAT[0].IO[4100].Data
#define Slave_0_60B9_0_Touchprobestatus  ECAT[0].IO[4101].Data
#define Slave_0_60BA_0_Touchprobepos1pos  ECAT[0].IO[4102].Data
#define Slave_0_60BC_0_Touchprobepos2pos  ECAT[0].IO[4103].Data
#define Slave_0_60FD_0_Digitalinputs  ECAT[0].IO[4104].Data
```

When using the 261 receive PDO mapping you will need to set a few variables in order to run your motors. Assuming the mapping above, add the following to your startup PLC (or in operational PLC for dynamic changes):

```
//Set up for torque control
Slave_0_607F_0_Maxprofilevelocit=838860000;  //6000 rpm  units are cts/ms 419430000;  //3000 rpm units are cts/ms max velocity
Slave_0_60E0_0_Positivetorquelim=3000; //300% positive torque limit
Slave_0_60E1_0_Negativetorquelim=3000; //300% negative torque limit
Slave_0_6060_0_Modesofoperation=10;  //set for torque mode

/*Mode of Operation value Description
0 Not specified
1 Profile position mode (pp)
3 Profile velocity mode (pv)
6 Homing mode (hm)
8 Cyclic synchronous position mode (csp)
9 Cyclic synchronous velocity mode (csv)
10 Cyclic synchronous torque mode (cst)
*/
```

Note that once you have these basics done, you can use the Setup tool to set the motor up, and the Tuning tool to tune it once the setup is done. Note that everything shown here is using motor units of counts.

Some variables to set after the initial motor setup is done are:

```
Motor[1].JogTa=- -0.002 //0.002 ms^2/ct
Motor[1].JogTs= -0.0005  //0.0005 ms^3/ct
Motor[1].AbortTs=25
Motor[1].MaxSpeed=400000
Motor[1].JogSpeed=280000
```

One potential gotcha – there is a one cycle delay after enabling the motor – so if the motor is not enabled before telling it to move you may get a fatal following error. Best way to fix this is to enable your motors at the start of operation, wait a couple of milliseconds, then start your moves. If the motor is ever killed, repeat this.

Some PowerPMAC variables for a 1S (with a demo motor) sitting at the first node and setup on Motor[1] are:

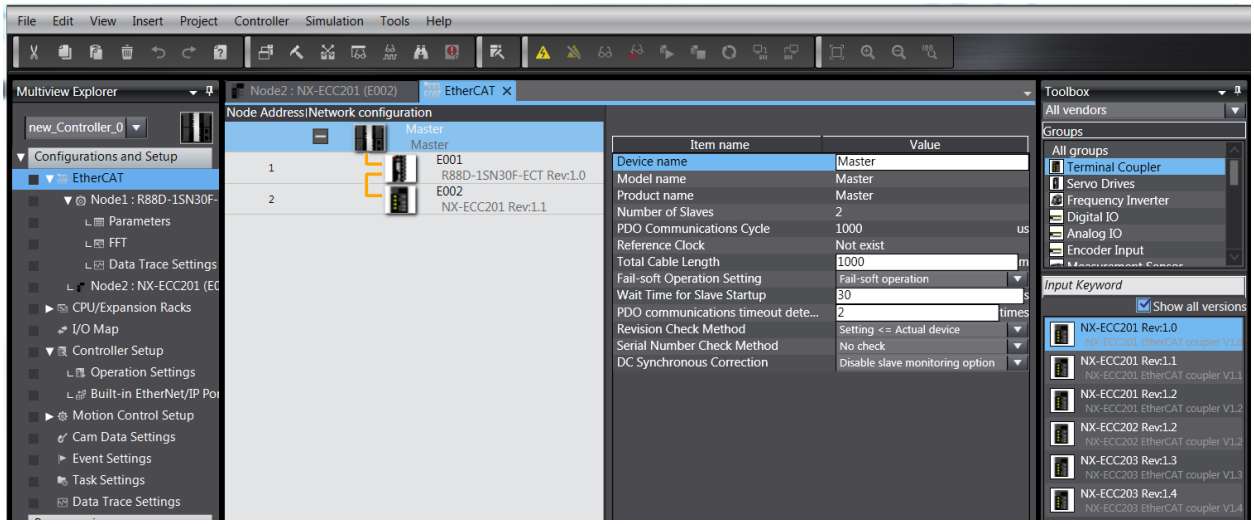
```
//Motor 1 – note this setting is for PDO map 259 target position
Motor[1].Ctrl=Sys.PosCtrl  // Sys.ServoCtrl for torque control
Motor[1].pDac=ECAT[0].IO[1].Data.a  //ECAT[0].IO[3].Data for torque control
Motor[1].MaxDac=3000                    //300%
Motor[1].JogTa=- -0.002
Motor[1].JogTs= -0.0005
Motor[1].AbortTs=25
Motor[1].MaxSpeed=400000
Motor[1].JogSpeed=280000
Motor[1].pEnc2=EncTable[32].a
Motor[1].AmpEnableBit=0
Motor[1].LimitBits=64
Motor[1].pAmpEnable=ECAT[0].IO[0].Data.a
Motor[1].pAmpFault=ECAT[0].IO[4097].Data.a
Motor[1].pLimits=ECAT[0].IO[4104].Data.a
Motor[1].AmpFaultBit=3
Motor[1].pEnc=EncTable[32].a
motor[1].FatalFeLimit=8000000
Motor[1].Control[0]=$10000f00
Motor[1].Control[1]=$0
EncTable[32].type=1
EncTable[32].index1=0
EncTable[32].index2=0
EncTable[32].index3=0
EncTable[32].index4=0
EncTable[32].index5=0
EncTable[32].pEnc1=Sys.pushm
EncTable[32].pEnc=ECAT[0].IO[4098].Data.a
EncTable[32].MaxDelta=0
EncTable[32].ScaleFactor=1
EncTable[32].index6=0
```

NX I/O

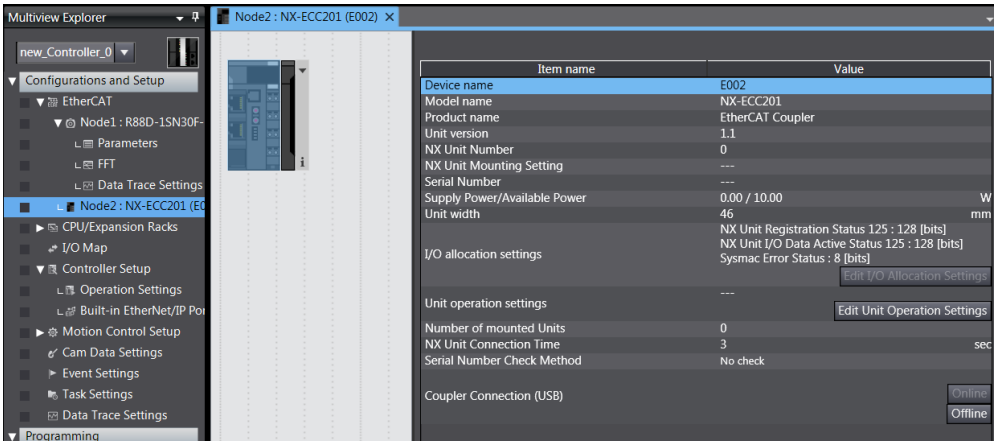
NOTE: Major caveat – using the revision 1.5 NX-ECC203 EtherCAT coupler:

This version of the NX-ECC203 card can only have ONE (total) of any of the following cards in the rack, and that one card must be in slot 1: position interface units, safety CPUs. When using these cards (and also any safety i/o cards) It must also be put in distributed clock more – NOT free run.

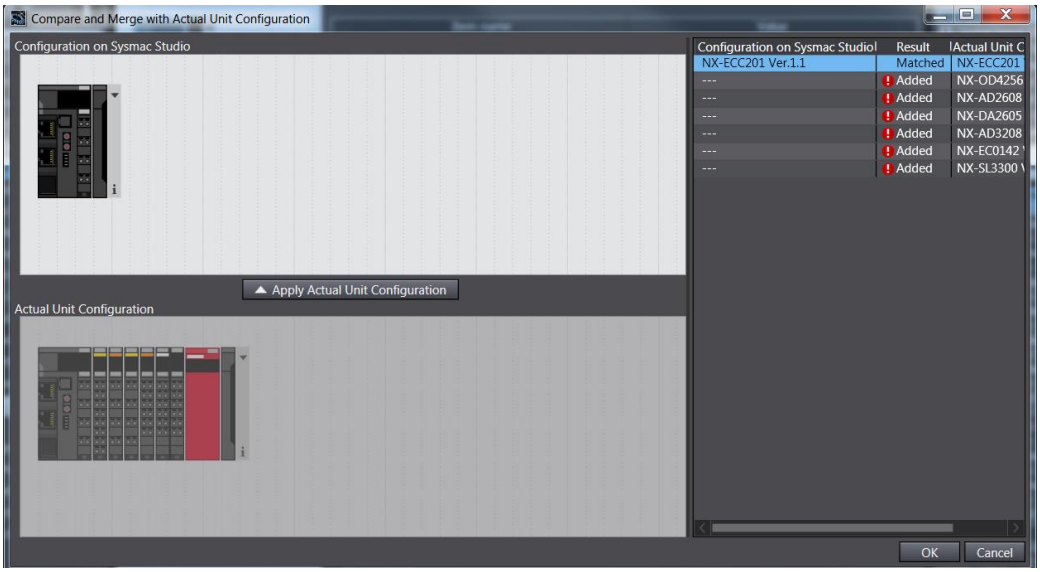
Note: ECC201 and ECC202 support only Free Run mode. To use with synchronized clocks MUST have an ECC203. Must use with synchronized clocks in order to use NX i/o to input motor feedback or to do motor control. Apply power and open Sysmac Studio. Going to the Configurations and Setup section double click on the EtherCAT sub section. If not already present, drag the NX-ECC20x which you are using into the EtherCAT network and drop it. In the example below I am using an older revision coupler, so I checked the 'Show all versions' box in the toolbox to see the Rev 1.1.



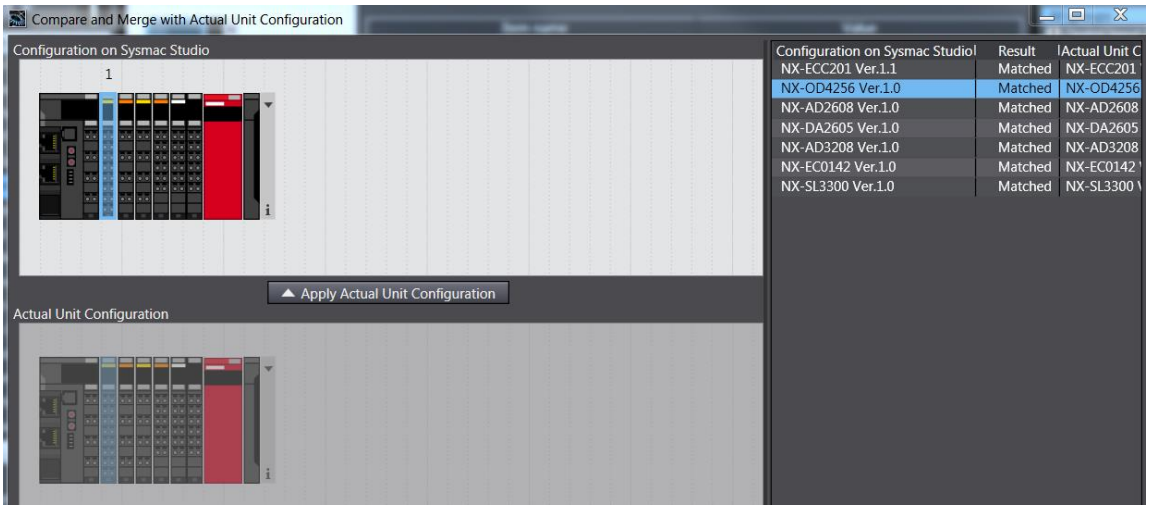
Connect to the NX coupler using a USB cable, and disconnect the EtherCAT cable. Right click on the NX node and select Coupler Connection (USB)-> Online. Double click on the NX node to open the NX configuration page.



Right click on the coupler in the node tab and choose ‘clear all memory’. Accept clearing coupler and all modules. Depending on modules attached this may take a couple of minutes to complete. Right click on the NX node and select Coupler Connection (USB)-> Offline. Cycle power on the unit. When this completes, right click on the NX node and select Coupler Connection (USB)-> Online. Right click on the coupler in the node tab and choose ‘Compare and Merge with Actual Unit Configuration’. A window with the existing modules should pop up.

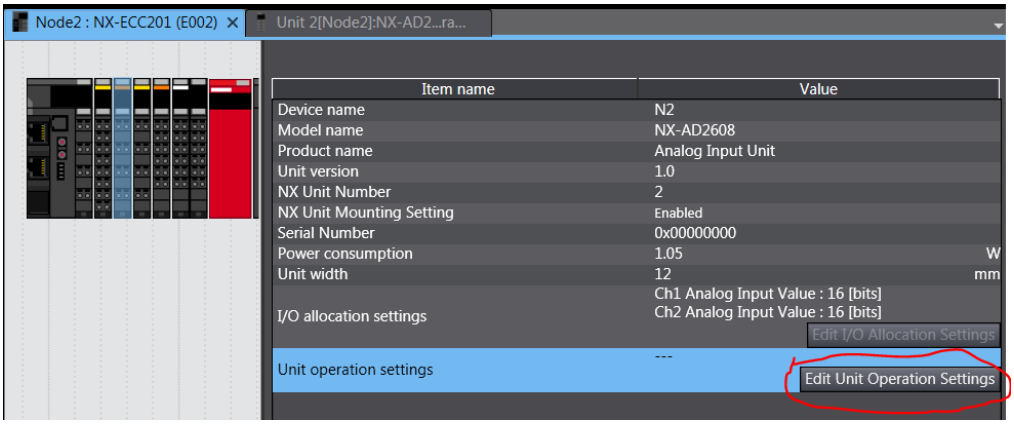


Click the ‘Apply Actual Unit Configuration’ button in the middle. Everything should match now.

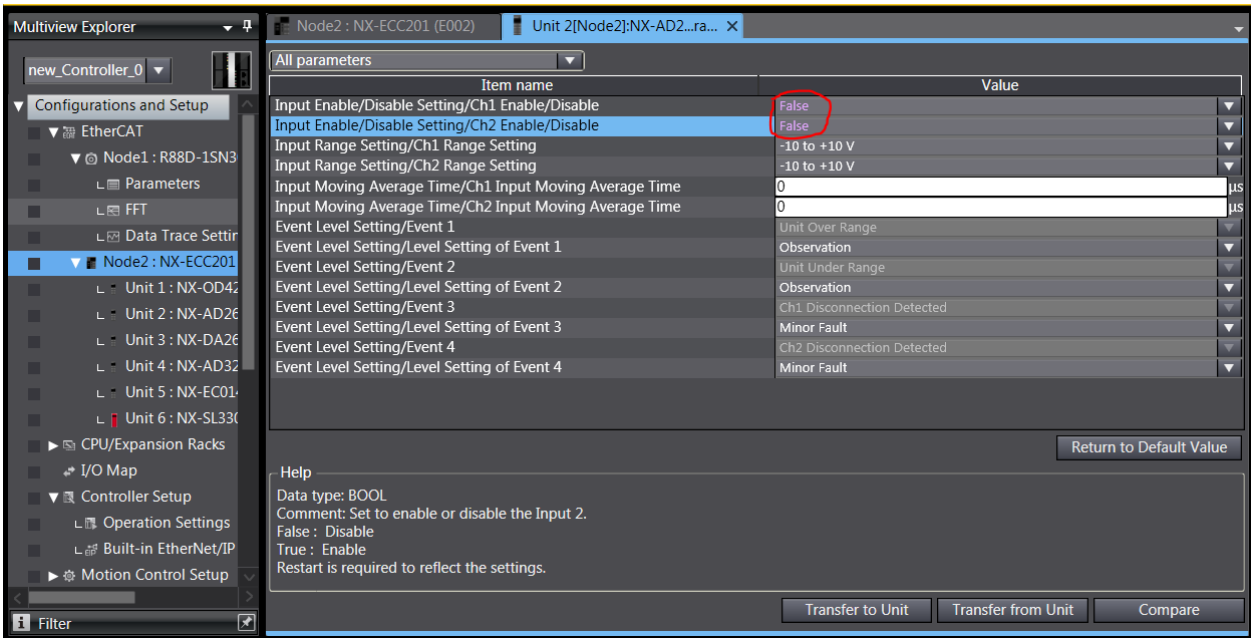


Click the ‘OK’ button.

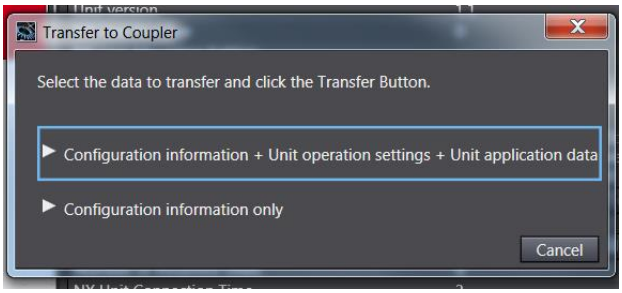
Now you can click on the various modules in the Node tab and select ‘Edit Unit Operation Settings’ for modules as needed. When finished editing for each module be sure to press the ‘Transfer to Unit’ button.



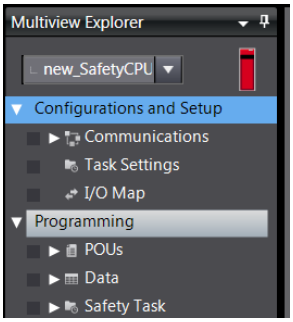
One important Analog Input module note – if you do not have inputs connected you must disable the input to avoid faults.



When all modules are configured, right click on the NX node and select Coupler Connection (USB)->Transfer to Coupler. Select the 'Configuration information + Unit operations settings + Unit application data' in the popup.



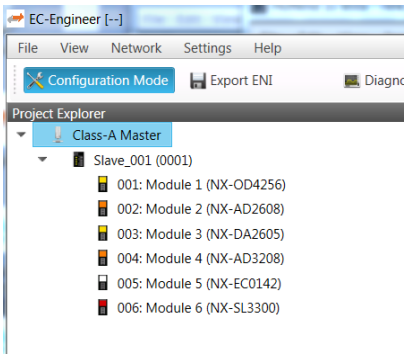
If you have a safety controller you may now select the new_SafetyCPU in the Multiview Explorer pane and configure and program it.



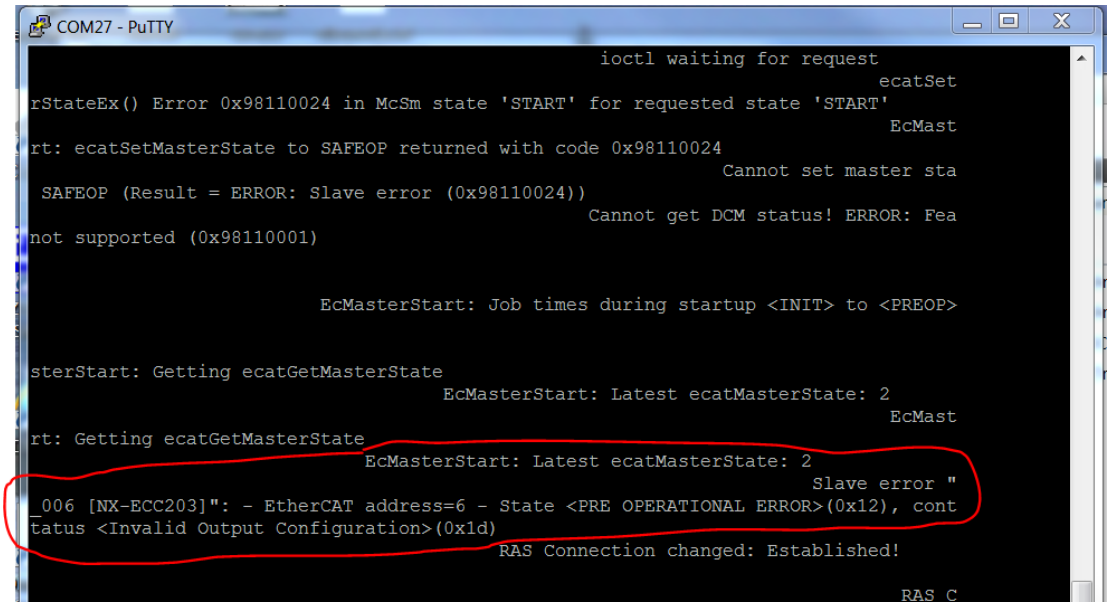
When done with the SafetyCPU (if any) go back to the controller, right click on the NX node and select Coupler Connection (USB)-> Offline. Power down, disconnect the USB cable, connect the EtherCAT cable(s), and power up again.

The 'ecat slaves' command with a NX-ECC201 EtherCAT coupler will show Omron VID as $83 (0x83) and PC as $83 (0x83).

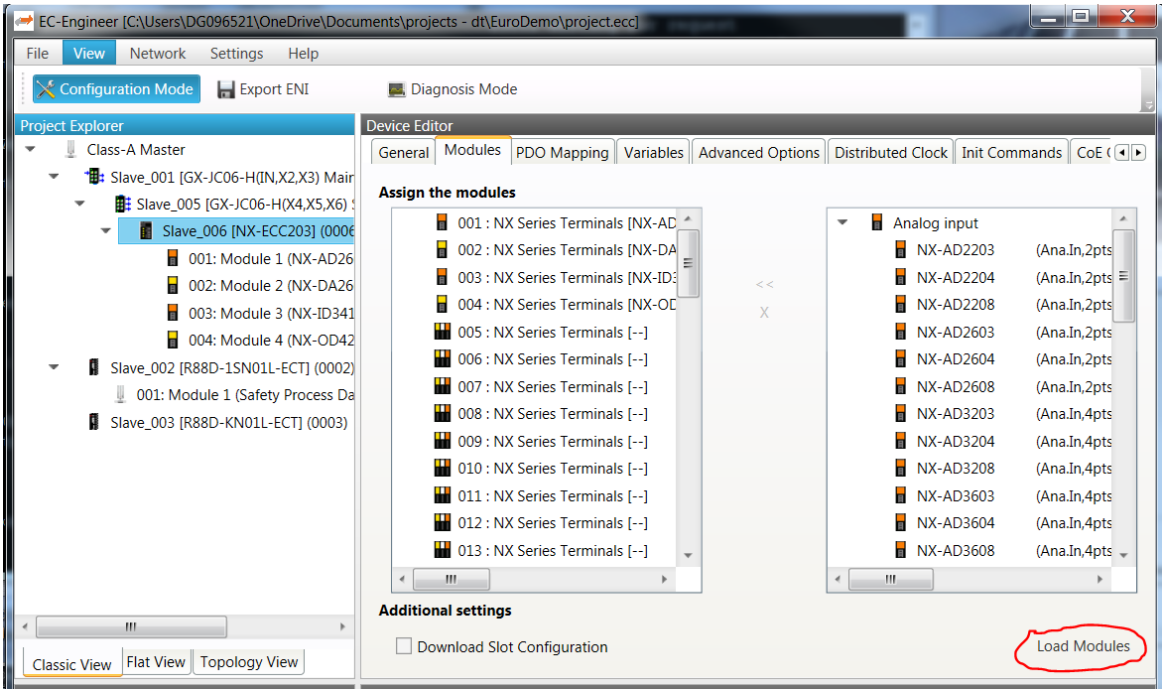
Open EC Engineer and scan the network (Appendix C). Click on the NX coupler (Slave_001 in this example) and open the subsections to see the modules.



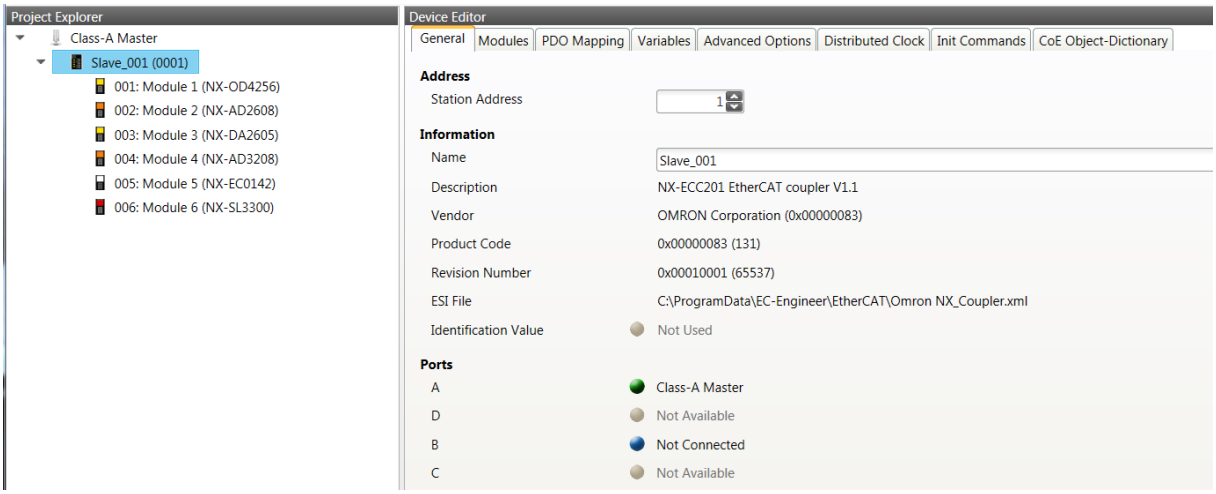
You should now see all the modules on the NX coupler. If the modules are not there and you try to start up using the configuration PuTTy will show the following error (on node 6) when you try to set Ecat[0].Enable=1:



If this happens go back to EC-Engineer and press the Load Modules button in the Device Editor pane:



Click on the NX slave in the Project Explorer pane.

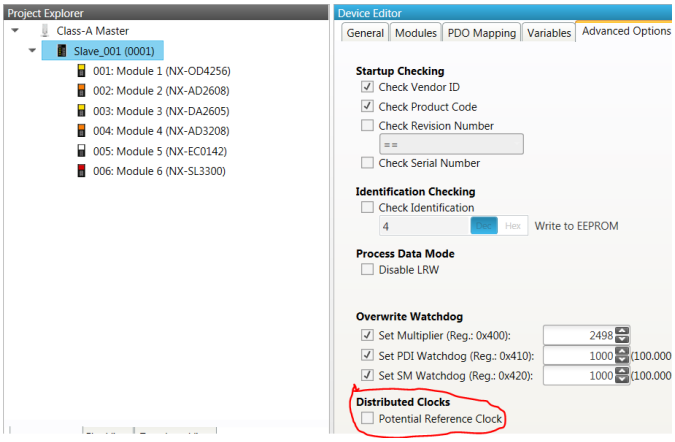


Click on the 'Modules' tab in the Device Editor pane. You should see all of the correct modules.

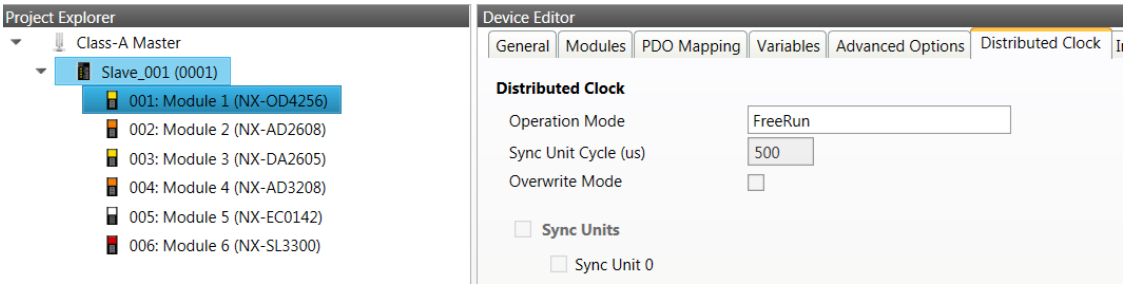
Click on the 'PDO Mapping' tab in the Device Editor pane. Select the appropriate Input and Output PDOs for your modules. By default all of the correct PDOs are already selected. See note 1.

Click on the 'Modules' tab in the Device Editor pane. You should see all of the variables transmitted for each PDO selected. If any data here is wrong you will need to check your ESI file

Click on the 'Advanced Options' tab in the Device Editor pane. If you have a drive in the network, or are using the ECC201 coupler, make sure that Potential Reference Clock is NOT checked in the Distributed Clocks settings.



Click on the 'Distributed Clock' tab in the Device Editor pane. Set the Operation Mode to 'Free Run'.



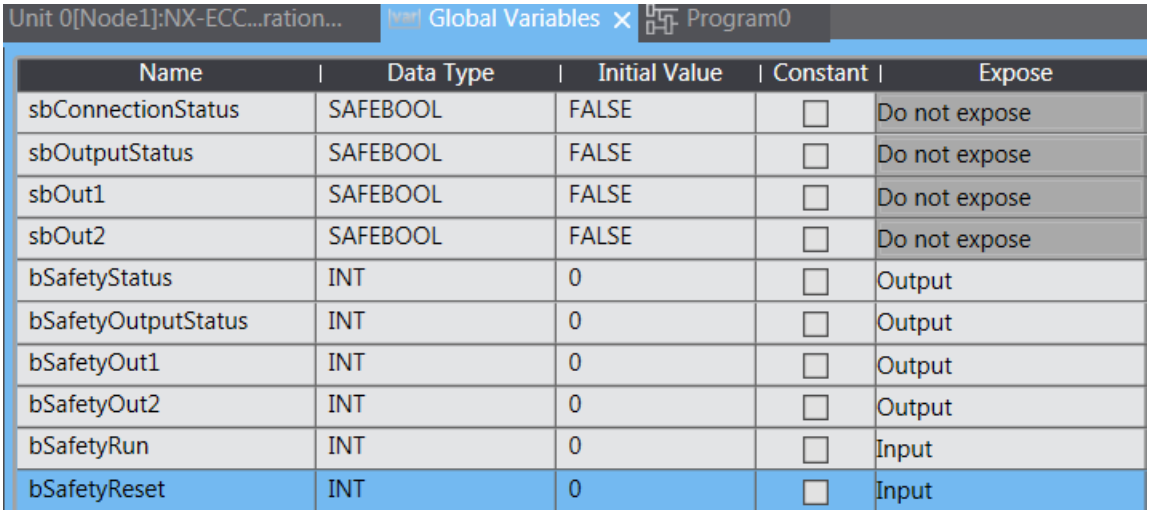
The EtherCAT header file created from the EC Engineer xml file looks like:

```
#define Slave_0_7001_1_OutputBit8bits  pshm->ECAT[0].IO[0].Data
```

```
#define Slave_0_7040_1_Ch1AnalogOutputVa  pshm->ECAT[0].IO[1].Data
#define Slave_0_7040_2_Ch2AnalogOutputVa  pshm->ECAT[0].IO[2].Data
#define Slave_0_7088_1_Ch1PulsePeriodMea  pshm->ECAT[0].IO[3].Data
#define Slave_0_7084_1_Ch1LatchFunction  pshm->ECAT[0].IO[4].Data
#define Slave_0_3003_4_NXUnitRegistratio(x) pshm->ECAT[0].IOBuffer[0 + x ]
#define Slave_0_3006_4_NXUnitIODataActiv(x) pshm->ECAT[0].IOBuffer[17 + x ]
#define Slave_0_2002_1_SysmacErrorStatus  pshm->ECAT[0].IO[4098].Data
#define Slave_0_6020_1_Ch1AnalogInputVal  pshm->ECAT[0].IO[4099].Data
#define Slave_0_6020_2_Ch2AnalogInputVal  pshm->ECAT[0].IO[4100].Data
#define Slave_0_6060_1_Ch1AnalogInputVal  pshm->ECAT[0].IO[4101].Data
#define Slave_0_6060_2_Ch2AnalogInputVal  pshm->ECAT[0].IO[4102].Data
#define Slave_0_6060_3_Ch3AnalogInputVal  pshm->ECAT[0].IO[4103].Data
#define Slave_0_6060_4_Ch4AnalogInputVal  pshm->ECAT[0].IO[4104].Data
#define Slave_0_6080_1_Ch1EncoderCounter  pshm->ECAT[0].IO[4105].Data
#define Slave_0_6081_1_Ch1ResetExternalI  pshm->ECAT[0].IO[4106].Data
#define Slave_0_6082_1_Ch1EncoderPresent  pshm->ECAT[0].IO[4107].Data
#define Slave_0_6088_1_Ch1PulsePeriodMea  pshm->ECAT[0].IO[4108].Data
#define Slave_0_6084_1_Ch1LatchStatus  pshm->ECAT[0].IO[4109].Data
#define Slave_0_6085_1_Ch1LatchInput1Dat  pshm->ECAT[0].IO[4110].Data
#define Slave_0_6086_1_Ch1LatchInput2Dat  pshm->ECAT[0].IO[4111].Data
```

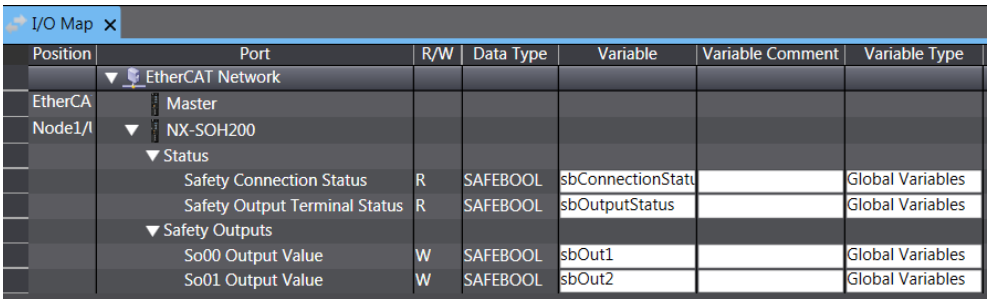
NX Safety

In Sysmac Studio go to the Safety CPU, then the Global Variables (Programming->Data->Global Variables). Define any desired variables to exchange between the NX Safety Controller and the PowerPMAC. Major note: At this time DO not use BOOLs – use INTs for data transferred between the NX Safety CPU and the PowerPMAC.

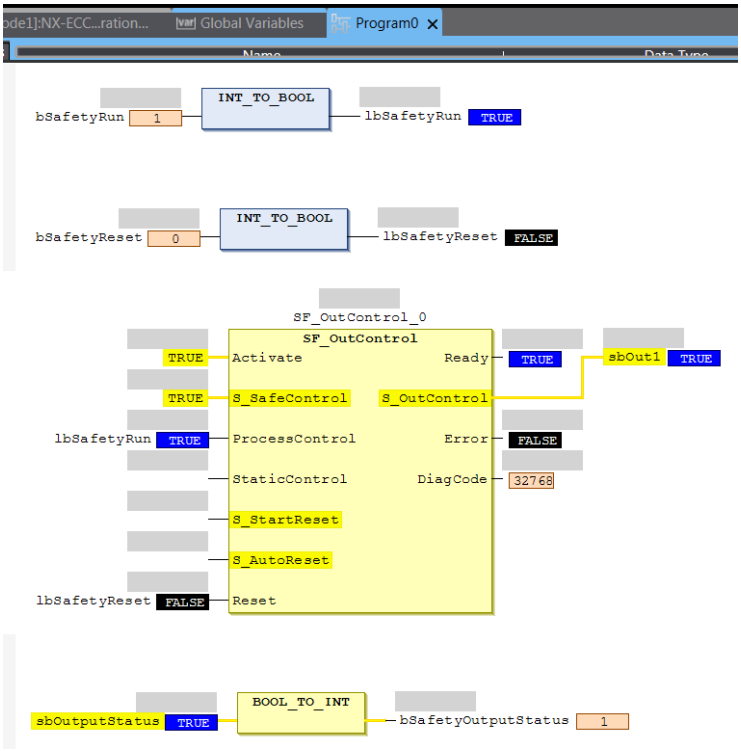


Name	Data Type	Initial Value	Constant	Expose
sbConnectionStatus	SAFEBOOL	FALSE	☐	Do not expose
sbOutputStatus	SAFEBOOL	FALSE	☐	Do not expose
sbOut1	SAFEBOOL	FALSE	☐	Do not expose
sbOut2	SAFEBOOL	FALSE	☐	Do not expose
bSafetyStatus	INT	0	☐	Output
bSafetyOutputStatus	INT	0	☐	Output
bSafetyOut1	INT	0	☐	Output
bSafetyOut2	INT	0	☐	Output
bSafetyRun	INT	0	☐	Input
bSafetyReset	INT	0	☐	Input

In the I/O Map section (Configurations and Setup->I/O Map) build any hardware variables necessary for the application.



Program the Safety Controller as desired. Note that this document will not go into safety programming or general use of the NX Safety Controller. Note that the program shown here is NOT a valid safety program – it will run and will operate the safety output based on input from the PowerPMAC, but is NOT meant to be used in a safety system.

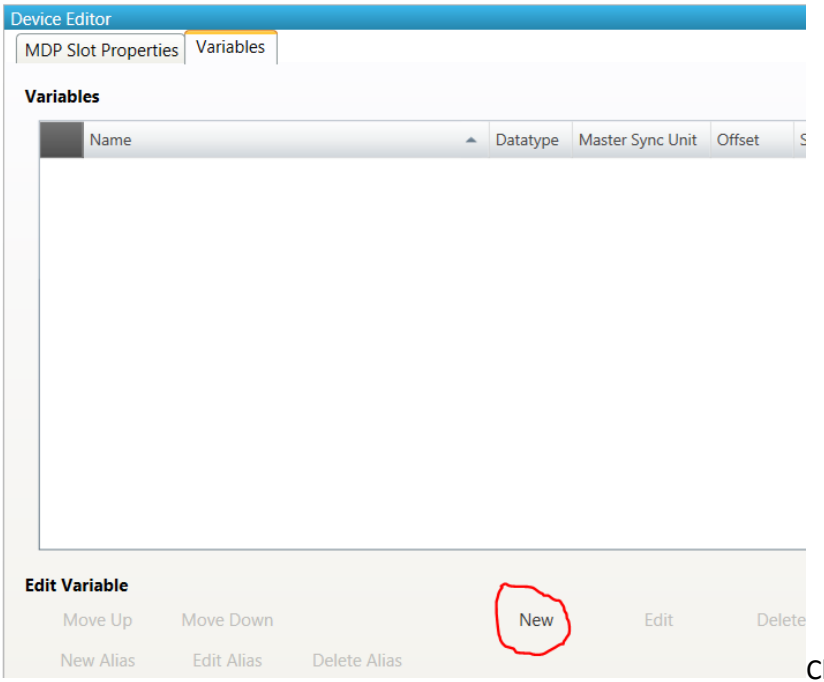


Validate the safety program, return to RUN mode, then go back to the NX controller side, save, build, and download to the controller. Afterwards cycle power on the entire system.

When all is done here, go to the Controller section in Sysmac Studio, double click on the EtherCAT section, then right click on the Master and select ‘Export All Couplers’ I/O Allocations’. Keep track of where you put the file created. Go to that directory in Windows Explorer and Extract All from the zip file. Open the CouplerMemoryMap.xml using Internet Explorer. It is very important that the Offsets seen here match those in later steps.

Device	PDO Mapping					PDO Entry				
Name	Name	Index	SM	Offset	Size	Name	Index	Data Type	Offset	Size
E001	TxPDO									
	Slot0(NX-ECC203)505th transmit PDO Mapping	#x1BF8	3	0.00	16.00	NX Unit Registration Status 63	#x2003:03	ARRAY [0..7] OF BYTE	0.00	8.00
						NX Unit I/O Data Active Status 63	#x2005:03	ARRAY [0..7] OF BYTE	8.00	8.00
	Slot0(NX-ECC203)512th transmit PDO Mapping	#x1BFF	3	16.00	1.00	Sysmac Error Status	#x2001:01	USINT	16.00	1.00
	Slot0(NX-ECC203)PaddingTxPdo	#x1BF4	3	17.00	1.00	---	---	---	---	1.00
	Slot1(NX-SL3300)Input Data Set 1	#x1A00	3	18.00	6.00	Node1/Unit2	#x6000:01	ARRAY [0..5] OF BYTE	18.00	6.00
	Slot1(NX-SL3300)Input Data Set 2	#x1A01	3	24.00	10.00	Safety CPU Status	#x6004:01	UINT	24.00	2.00
						bSafetyStatus	#x6001:02	INT	26.00	2.00
						bSafetyOutputStatus	#x6001:03	INT	28.00	2.00
						bSafetyOut1	#x6001:04	INT	30.00	2.00
						bSafetyOut2	#x6001:05	INT	32.00	2.00
	Slot2(NX-SOH200)Input Data Set 1	#x1A04	3	34.00	6.00	FSoE Slave CMD	#x6020:01	USINT	34.00	1.00
						Safety Input 1st Byte	#x6021:01	USINT	35.00	1.00
						FSoE Slave CRC_0	#x6020:03	UINT	36.00	2.00
						FSoE Slave Conn_ID	#x6020:02	UINT	38.00	2.00
	Slot2(NX-SOH200)Input Data Set 2	#x1A05	3	40.00	2.00	Standard Input 1st Byte	#x6022:01	USINT	40.00	1.00
						Standard Input 2nd Byte	#x6022:02	USINT	41.00	1.00
	RxPDO									
	Slot1(NX-SL3300)Output Data Set 1	#x1600	2	0.00	6.00	Node1/Unit2	#x7000:01	ARRAY [0..5] OF BYTE	0.00	6.00
	Slot1(NX-SL3300)Output Data Set 2	#x1601	2	6.00	4.00	bSafetyRun	#x7001:01	INT	6.00	2.00
						bSafetyReset	#x7001:02	INT	8.00	2.00
	Slot2(NX-SOH200)Output Data Set 1	#x1604	2	10.00	6.00	FSoE Master CMD	#x7020:01	USINT	10.00	1.00
						Safety Output 1st Byte	#x7021:01	USINT	11.00	1.00
						FSoE Master CRC_0	#x7020:03	UINT	12.00	2.00
						FSoE Master Conn_ID	#x7020:02	UINT	14.00	2.00
	Slot2(NX-SOH200)Output Data Set 2	#x1605	2	16.00	2.00	Standard Output 1st Byte	#x7022:01	USINT	16.00	1.00
						Standard Output 2nd Byte	#x7022:02	USINT	17.00	1.00

In EC Engineer select the safety controller, then click the Variables tab.



Click ‘New’. Add an ARRAY[0..1] of BYTE from SL300 Input Data Set 1, checking the ‘Combine’ box, Count=3. Add a UINT from SL3300 Input Data Set 2, using a Count of 1. Add an Array[0..1] of BYTE from SL3300 Input Data Set 2, using a count of 4. Add an ARRAY[0..1] of BYTE from SL300 Output Data Set 1, checking the ‘Combine’ box, Count=3. Add an Array[0..1] of BYTE from SL3300 Output Data Set 2, using a count of 2. The variables shown will initially be xxxxData Set y.Variable n. You can click on each variable, then click ‘Edit’ and rename the variable. Note: if have odd number in array size of BYTEs, can use type USINT, count = size of array, and check ‘combine’. Here is final result:

Device Editor

MDP Slot Properties | Variables

Variables

Name	Datatype	Master Sync Unit	Offset	Size
Slave_001 [NX-ECC203].Module 1 (NX-SL3300).Input Data Set 1.Variable 0	ARRAY [0..2] OF UINT	Id 0: Default 0	IN : 18.0	6.0
Slave_001 [NX-ECC203].Module 1 (NX-SL3300).Input Data Set 2.Safety CPU Status	UINT	Id 0: Default 0	IN : 24.0	2.0
Slave_001 [NX-ECC203].Module 1 (NX-SL3300).Input Data Set 2.bSafetyStatus	ARRAY [0..1] OF BYTE	Id 0: Default 0	IN : 26.0	2.0
Slave_001 [NX-ECC203].Module 1 (NX-SL3300).Input Data Set 2.bSafetyOutputStatus	ARRAY [0..1] OF BYTE	Id 0: Default 0	IN : 28.0	2.0
Slave_001 [NX-ECC203].Module 1 (NX-SL3300).Input Data Set 2.bSafetyOut1	ARRAY [0..1] OF BYTE	Id 0: Default 0	IN : 30.0	2.0
Slave_001 [NX-ECC203].Module 1 (NX-SL3300).Input Data Set 2.bSafetyOut2	ARRAY [0..1] OF BYTE	Id 0: Default 0	IN : 32.0	2.0
Slave_001 [NX-ECC203].Module 1 (NX-SL3300).Output Data Set 1.Variable 0	ARRAY [0..2] OF UINT	Id 0: Default 0	OUT : 0.0	6.0
Slave_001 [NX-ECC203].Module 1 (NX-SL3300).Output Data Set 2.bSafetyRun	ARRAY [0..1] OF BYTE	Id 0: Default 0	OUT : 6.0	2.0
Slave_001 [NX-ECC203].Module 1 (NX-SL3300).Output Data Set 2.bSafetyReset	ARRAY [0..1] OF BYTE	Id 0: Default 0	OUT : 8.0	2.0

Do the same exercise for any other safety I/O modules in the rack. My rack has only one SOH200, and the variable list for that is auto created, looking like:

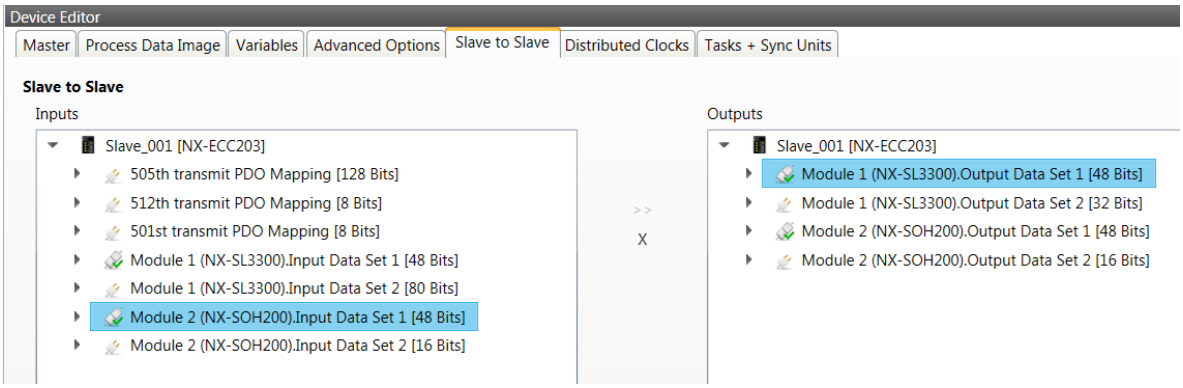
MDP Slot Properties | Variables

Variables

Name	Datatype	Master Sync Unit	Offset	Size
Slave_001 [NX-ECC203].Module 2 (NX-SOH200).Input Data Set 1.FSoE Slave CMD	USINT	Id 0: Default 0	IN : 34.0	1.0
Slave_001 [NX-ECC203].Module 2 (NX-SOH200).Input Data Set 1.Safety Input 1st Byte	USINT	Id 0: Default 0	IN : 35.0	1.0
Slave_001 [NX-ECC203].Module 2 (NX-SOH200).Input Data Set 1.FSoE Slave CRC_0	UINT	Id 0: Default 0	IN : 36.0	2.0
Slave_001 [NX-ECC203].Module 2 (NX-SOH200).Input Data Set 1.FSoE Slave Conn_ID	UINT	Id 0: Default 0	IN : 38.0	2.0
Slave_001 [NX-ECC203].Module 2 (NX-SOH200).Input Data Set 2.Standard Input 1st Byte	USINT	Id 0: Default 0	IN : 40.0	1.0
Slave_001 [NX-ECC203].Module 2 (NX-SOH200).Input Data Set 2.Standard Input 2nd Byte	USINT	Id 0: Default 0	IN : 41.0	1.0
Slave_001 [NX-ECC203].Module 2 (NX-SOH200).Output Data Set 1.FSoE Master CMD	USINT	Id 0: Default 0	OUT : 10.0	1.0
Slave_001 [NX-ECC203].Module 2 (NX-SOH200).Output Data Set 1.Safety Output 1st Byte	USINT	Id 0: Default 0	OUT : 11.0	1.0
Slave_001 [NX-ECC203].Module 2 (NX-SOH200).Output Data Set 1.FSoE Master CRC_0	UINT	Id 0: Default 0	OUT : 12.0	2.0
Slave_001 [NX-ECC203].Module 2 (NX-SOH200).Output Data Set 1.FSoE Master Conn_ID	UINT	Id 0: Default 0	OUT : 14.0	2.0
Slave_001 [NX-ECC203].Module 2 (NX-SOH200).Output Data Set 2.Standard Output 1st Byte	USINT	Id 0: Default 0	OUT : 16.0	1.0
Slave_001 [NX-ECC203].Module 2 (NX-SOH200).Output Data Set 2.Standard Output 2nd Byte	USINT	Id 0: Default 0	OUT : 17.0	1.0

NOTE: If you have added your own safety variables of type BOOL for the Safety Controller the autogenerated offsets may be wrong. Check the offsets to make sure they match those from the Sysmac Studio generated file. If not you will have to delete these variables and create new ones – you may have to change to an INT type.

Click on the Master in the Project Explorer, then the Slave to Slave tab. You will need to associate the Safety I/O slices with the Safety CPU. The Safety CPU Input Data Set 1 needs to be associated with EACH of the Safety I/O card's Output Data Set 1. The Safety CPU Output Data Set 1 needs to be associated with EACH of the Safety I/O card's Input Data Set 1. This is done by clicking on the Input Data Set in question in the left pane, then clicking on the Output Data Set in question in the right pane, then clicking on the '>>' between the panes. For one SOH200 it should look like this at the end:



NX Safety – Local Rack safety I/O only

See above for example.

NX Safety – Local Rack safety I/O plus one 1S drive

Use normal setup for the NX rack, including the safety above. Use normal setup for the 1S, but include the 273 transmit and receive PDOs in the PDO mapping (EC Engineer). When adding the variables for the Safety CPU you need to add variables for the 273 Transmit and Receive PDOs. The memory used for each drive will be divided by:

	INDEX	SIZE (BYTEs)	Description	ADDRESS	TYPE	Offset (from 0)	SIZE (BYTEs)	New Variable Type & Number
273th transmit PDO Mapping	#x1B10	7						
			FSoE Slave CMD	#xE600:01	USINT	0	1	USINT
			STO command	#x6640:00	BOOL	1	0.01	BOOL 8
			Padding - BOOL	---	---	---	0.06	
			error acknowledge	#x6632:00	BOOL	1.07	0.01	
			Padding - BOOL	---	---	---	0.07	BOOL 8
			Safety Connection	#xE601:01	BOOL	2.07	0.01	
			FSoE Slave CRC_0	#xE600:03	UINT	3	2	UINT
			FSoE Slave Conn_ID	#xE600:02	UINT	5	2	UINT
273th receive PDO Mapping	#x1710	7						
			FSoE Master CMD	#xE700:01	USINT	0	1	USINT
			STO command	#x6640:00	BOOL	1.00	0.01	BOOL 8
			Padding - BOOL	---	---	---	0.06	
			error acknowledge	#x6632:00	BOOL	1.07	0.01	
			Padding - BYTE	---	---	2	1	USINT
			FSoE Master CRC_0	#xE700:03	UINT	3	2	UINT
			FSoE Master	#xE700:02	UINT	5	2	UINT

The drive is Node 2, and the CouplerMemoryMap.xml for this particular section looks like - Input data set:

Slot1(NX-SL3300)Input Data Set 1	#x1A00	3	18.00	14.00	Node1/Unit2	#x6000:01	ARRAY [0..5] OF BYTE	18.00	6.00
					Node2	#x6000:02	ARRAY [0..6] OF BYTE	24.00	7.00
					Padding	#x6000:03	ARRAY [0..0] OF BYTE	31.00	1.00

Output data set:

Slot1(NX-SL3300)Output Data Set 1	#x1600	2	0.00	14.00	Node1/Unit2	#x7000:01	ARRAY [0..5] OF BYTE	0.00	6.00
					Node2	#x7000:02	ARRAY [0..6] OF BYTE	6.00	7.00
					Padding	#x7000:03	ARRAY [0..0] OF BYTE	13.00	1.00

Note that Dataset 2 in this example contains communications variables for use with the PowerPMAC, and that I have renamed the variables from default to give a logical description. So for a NX rack with one SL3300, one SOH200, and one 1S drive the Safety CPU input variables will look like:

Name	Datatype	Master Sync Unit	Offset		Size
Slave_001 [NX-ECC203].Module 1 (NX-SL3300).Input Data Set 1.Variable 0	ARRAY [0..2] OF UINT	Id 0: Default 0	IN :	18.0	6.0
Slave_001 [NX-ECC203].Module 1 (NX-SL3300).Input Data Set 1.DriveFSoECmd	USINT	Id 0: Default 0	IN :	24.0	1.0
Slave_001 [NX-ECC203].Module 1 (NX-SL3300).Input Data Set 1.DriveSTOCmd	BOOL	Id 0: Default 0	IN :	25.0	0.1
Slave_001 [NX-ECC203].Module 1 (NX-SL3300).Input Data Set 1.PadBool1	BOOL	Id 0: Default 0	IN :	25.1	0.1
Slave_001 [NX-ECC203].Module 1 (NX-SL3300).Input Data Set 1.PadBool2	BOOL	Id 0: Default 0	IN :	25.2	0.1
Slave_001 [NX-ECC203].Module 1 (NX-SL3300).Input Data Set 1.PadBool3	BOOL	Id 0: Default 0	IN :	25.3	0.1
Slave_001 [NX-ECC203].Module 1 (NX-SL3300).Input Data Set 1.PadBool4	BOOL	Id 0: Default 0	IN :	25.4	0.1
Slave_001 [NX-ECC203].Module 1 (NX-SL3300).Input Data Set 1.PadBool5	BOOL	Id 0: Default 0	IN :	25.5	0.1
Slave_001 [NX-ECC203].Module 1 (NX-SL3300).Input Data Set 1.PadBool6	BOOL	Id 0: Default 0	IN :	25.6	0.1
Slave_001 [NX-ECC203].Module 1 (NX-SL3300).Input Data Set 1.DriveFSoEErrorAck	BOOL	Id 0: Default 0	IN :	25.7	0.1
Slave_001 [NX-ECC203].Module 1 (NX-SL3300).Input Data Set 1.PadBool8	BOOL	Id 0: Default 0	IN :	26.0	0.1
Slave_001 [NX-ECC203].Module 1 (NX-SL3300).Input Data Set 1.PadBool9	BOOL	Id 0: Default 0	IN :	26.1	0.1
Slave_001 [NX-ECC203].Module 1 (NX-SL3300).Input Data Set 1.PadBool10	BOOL	Id 0: Default 0	IN :	26.2	0.1
Slave_001 [NX-ECC203].Module 1 (NX-SL3300).Input Data Set 1.PadBool11	BOOL	Id 0: Default 0	IN :	26.3	0.1
Slave_001 [NX-ECC203].Module 1 (NX-SL3300).Input Data Set 1.PadBool12	BOOL	Id 0: Default 0	IN :	26.4	0.1
Slave_001 [NX-ECC203].Module 1 (NX-SL3300).Input Data Set 1.PadBool13	BOOL	Id 0: Default 0	IN :	26.5	0.1
Slave_001 [NX-ECC203].Module 1 (NX-SL3300).Input Data Set 1.PadBool14	BOOL	Id 0: Default 0	IN :	26.6	0.1
Slave_001 [NX-ECC203].Module 1 (NX-SL3300).Input Data Set 1.SafetyConnStatus	BOOL	Id 0: Default 0	IN :	26.7	0.1
Slave_001 [NX-ECC203].Module 1 (NX-SL3300).Input Data Set 1.unDriveCRC_0	UINT	Id 0: Default 0	IN :	27.0	2.0
Slave_001 [NX-ECC203].Module 1 (NX-SL3300).Input Data Set 1.unDriveConnId	UINT	Id 0: Default 0	IN :	29.0	2.0
Slave_001 [NX-ECC203].Module 1 (NX-SL3300).Input Data Set 1.PaddingByte	ARRAY [0..0] OF BYTE	Id 0: Default 0	IN :	31.0	1.0
Slave_001 [NX-ECC203].Module 1 (NX-SL3300).Input Data Set 2.SafetyCPUStatus	UINT	Id 0: Default 0	IN :	32.0	2.0
Slave_001 [NX-ECC203].Module 1 (NX-SL3300).Input Data Set 2.bSafetyStatus	ARRAY [0..1] OF BYTE	Id 0: Default 0	IN :	34.0	2.0
Slave_001 [NX-ECC203].Module 1 (NX-SL3300).Input Data Set 2.bSafetyOutputStatus	ARRAY [0..1] OF BYTE	Id 0: Default 0	IN :	36.0	2.0
Slave_001 [NX-ECC203].Module 1 (NX-SL3300).Input Data Set 2.bSafetyOut1	ARRAY [0..1] OF BYTE	Id 0: Default 0	IN :	38.0	2.0
Slave_001 [NX-ECC203].Module 1 (NX-SL3300).Input Data Set 2.bSafetyOut2	ARRAY [0..1] OF BYTE	Id 0: Default 0	IN :	40.0	2.0

And output variables will look like:

Name	Datatype	Master Sync Unit	Offset	Size
Slave_001 [NX-ECC203].Module 1 (NX-SL3300).Output Data Set 1.Variable 0	ARRAY [0..2] OF UINT	Id 0: Default 0	OUT : 0.0	6.0
Slave_001 [NX-ECC203].Module 1 (NX-SL3300).Output Data Set 1.DriveFSoEMasterCmd	USINT	Id 0: Default 0	OUT : 6.0	1.0
Slave_001 [NX-ECC203].Module 1 (NX-SL3300).Output Data Set 1.DriveSTOCmd	BOOL	Id 0: Default 0	OUT : 7.0	0.1
Slave_001 [NX-ECC203].Module 1 (NX-SL3300).Output Data Set 1.Pad1	BOOL	Id 0: Default 0	OUT : 7.1	0.1
Slave_001 [NX-ECC203].Module 1 (NX-SL3300).Output Data Set 1.Pad2	BOOL	Id 0: Default 0	OUT : 7.2	0.1
Slave_001 [NX-ECC203].Module 1 (NX-SL3300).Output Data Set 1.Pad3	BOOL	Id 0: Default 0	OUT : 7.3	0.1
Slave_001 [NX-ECC203].Module 1 (NX-SL3300).Output Data Set 1.Pad4	BOOL	Id 0: Default 0	OUT : 7.4	0.1
Slave_001 [NX-ECC203].Module 1 (NX-SL3300).Output Data Set 1.Pad6	BOOL	Id 0: Default 0	OUT : 7.5	0.1
Slave_001 [NX-ECC203].Module 1 (NX-SL3300).Output Data Set 1.Pad7	BOOL	Id 0: Default 0	OUT : 7.6	0.1
Slave_001 [NX-ECC203].Module 1 (NX-SL3300).Output Data Set 1.DriveErrorAck	BOOL	Id 0: Default 0	OUT : 7.7	0.1
Slave_001 [NX-ECC203].Module 1 (NX-SL3300).Output Data Set 1.BytePad1	USINT	Id 0: Default 0	OUT : 8.0	1.0
Slave_001 [NX-ECC203].Module 1 (NX-SL3300).Output Data Set 1.DriveFSoEMasterCRC_0	UINT	Id 0: Default 0	OUT : 9.0	2.0
Slave_001 [NX-ECC203].Module 1 (NX-SL3300).Output Data Set 1.DriveFSoEMasterConnId	UINT	Id 0: Default 0	OUT : 11.0	2.0
Slave_001 [NX-ECC203].Module 1 (NX-SL3300).Output Data Set 1.BytePad2	USINT	Id 0: Default 0	OUT : 13.0	1.0
Slave_001 [NX-ECC203].Module 1 (NX-SL3300).Output Data Set 2.bSafetyRun	ARRAY [0..1] OF BYTE	Id 0: Default 0	OUT : 14.0	2.0
Slave_001 [NX-ECC203].Module 1 (NX-SL3300).Output Data Set 2.bSafetyReset	ARRAY [0..1] OF BYTE	Id 0: Default 0	OUT : 16.0	2.0

When setting the Slave to Slave mapping in EC engineer under the Class-A Master, set the NX mappings as above, then include the mapping of the NX Safety CPU to the 1S drive(s).

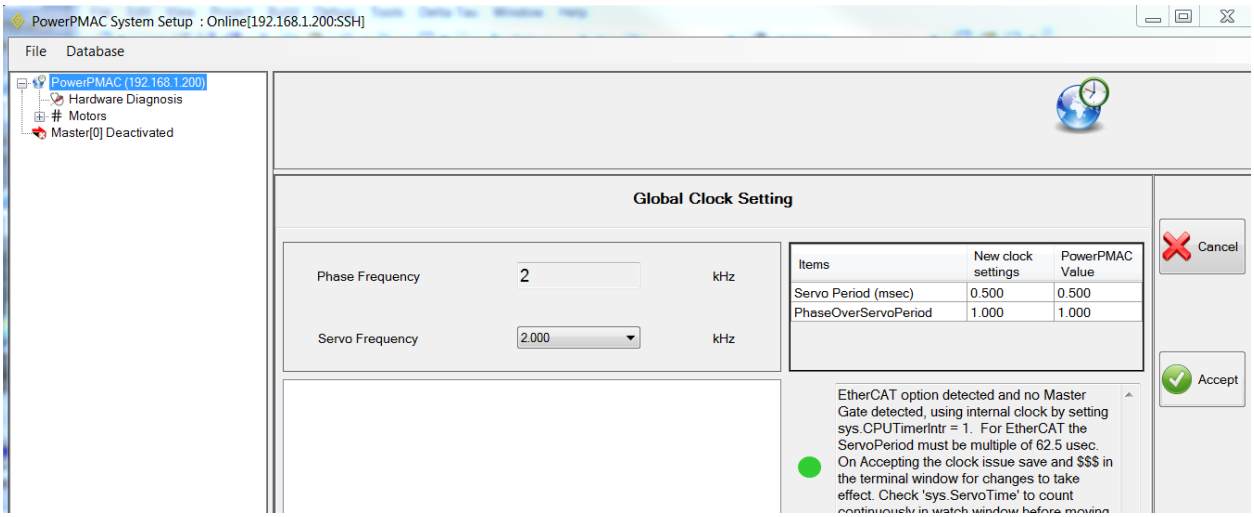
***Note 1:** You can check the PDO mappings for the various pieces of EtherCAT hardware by going into Sysmac Studio and comparing them. The PDO mapping of the same number should have the same variables with the same type.

Step 4

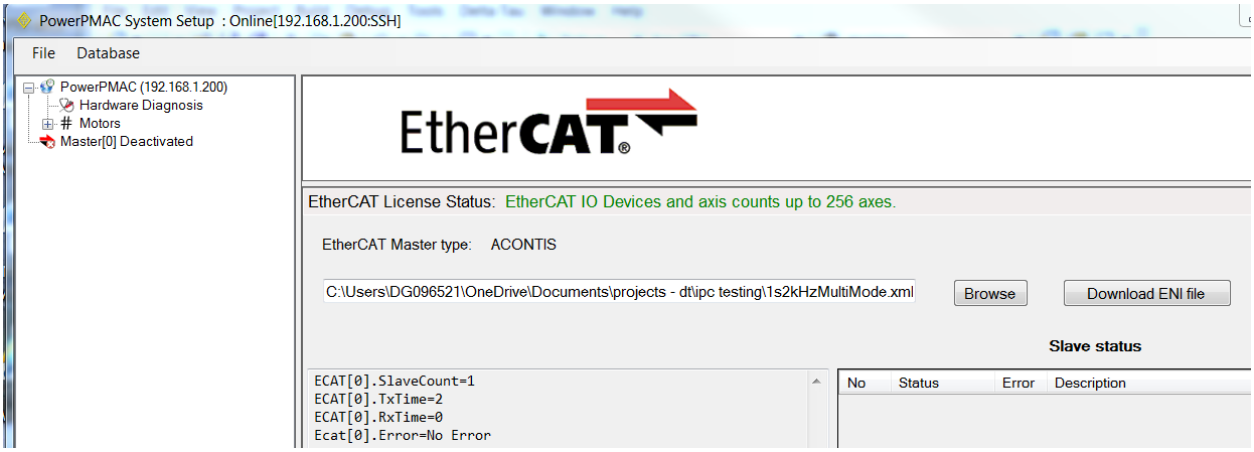
Once all the Slaves are set up (including at a minimum PDOs and clocks), choose the 'Save' option from the File menu item at the upper left of the EC Engineer window. Save the file where desired. Then click the 'Export ENI' button in the upper section of the EC Engineer window. Note where this file is being saved and give it a name with no spaces in the name. From the PowerPMAC IDE **make certain that ECAT[0].Enable=0** – otherwise any attempt to complete the next steps will fail, but may not show an error.

Step 5

From the PowerPMAC IDE choose 'System Setup' under the 'Tools' menu item. Click on the PowerPMAC object in the upper left corner of the System Setup window. Select the same servo frequency that was used in EC Engineer, and click 'Accept'. Here we are using 2 kHz:



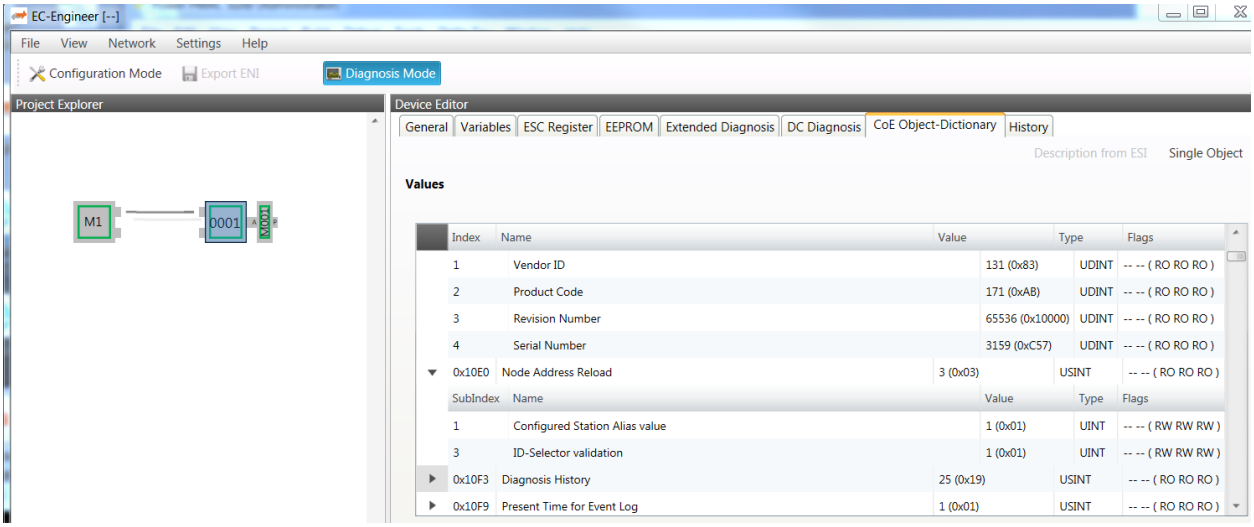
Click on the ‘Master[0] Deactivated’ object in the left pane of the System Setup window. Now click the ‘Browse’ button in the center pane. Navigate to the XML file created by EC Engineer, click on it, and click ‘OK’.



Now click the ‘Download ENI File’ button in the center pane. Assuming that ECAT[0].Enable=0 in the PowerPMAC, there should be no errors in the Setup Messages pane at the lower side of the System Setup window. Right click on the ‘Master[0] Deactivated’ object in the left pane of the System Setup window, and select ‘Export EtherCAT Variables’. Choose a header file name with NO spaces, and click the ‘Save’ button. If you are overwriting a file name tell it that that is ok when the popup windows appear. The Setup Messages pane should show no errors. Close the PowerPMAC System Setup window. Press the ‘OK’ button when the popup appears.

Go to the terminal window in the IDE. Type ‘save’ and enter. Type ‘$$$’ and enter. This saves everything to permanent memory and resets the PowerPMAC. At this point you should be able to type ‘ECAT[0].Enable=1’ and see that it remains at that value. Depending on how extensive changes are you may need to power cycle the entire system, or even initialize one or more of your slaves. If after doing all of this ECAT[0].Enable refuses to remain =1, there is something wrong with your EtherCAT configuration. Check for errors on your slaves to get an idea where the problem is. Also you can try typing ‘ecat slaves’ in the terminal window of the IDE – you should see all of your hardware listed. NOTE: I have seen a stubborn error where one drive had the same node address switch setting as the one beside it.

If you have slave errors, but are able to start EtherCAT communications, you may be able to get more information by going into Diagnosis Mode. Here I have chosen the 1S drive object and can see that the Node ID Sector matches the configured Alias value (which it should):



Some PowerPMAC variables for a 1S sitting at the first node and setup on Motor[1] are:

```
//Motor 1 – note this setting is for PDO map 259 target position
Motor[1].ServoCtrl=1  //activate processing for motor 1
Motor[1].Ctrl=Sys.PosCtrl  //Sys.ServoCtrl for torque control
Motor[1].pDac=ECAT[0].IO[1].Data.a  //This address needs to match the address of the drive
                    //Target Position if controlling in Position mode.
Motor[1].MotorTa=-0.000239999993937090039
Motor[1].MotorTs=10
Motor[1].JogTa=-0.00023999999
Motor[1].JogTs=10
Motor[1].AbortTs=25
Motor[1].MaxSpeed=40000
Motor[1].JogSpeed=280000
Motor[1].pEnc2=EncTable[32].a
Motor[1].AmpEnableBit=0
Motor[1].LimitBits=64                //bit 0 is MLIM, bit 1 is PLIM
Motor[1].pAmpEnable=ECAT[0].IO[0].Data.a
Motor[1].pAmpFault=ECAT[0].IO[4097].Data.a
Motor[1].pLimits=ECAT[0].IO[4104].Data.a
Motor[1].AmpFaultBit=3
Motor[1].pEnc=EncTable[32].a
motor[1].FatalFeLimit=8000000
Motor[1].Control[0]=$10000f00
Motor[1].Control[1]=$0
EncTable[32].type=1
EncTable[32].index1=0
EncTable[32].index2=0
EncTable[32].index3=0
EncTable[32].index4=0
EncTable[32].index5=0
EncTable[32].pEnc1=Sys.pushm
EncTable[32].pEnc=ECAT[0].IO[4098].Data.a
EncTable[32].MaxDelta=0
EncTable[32].ScaleFactor=1
EncTable[32].index6=0
```

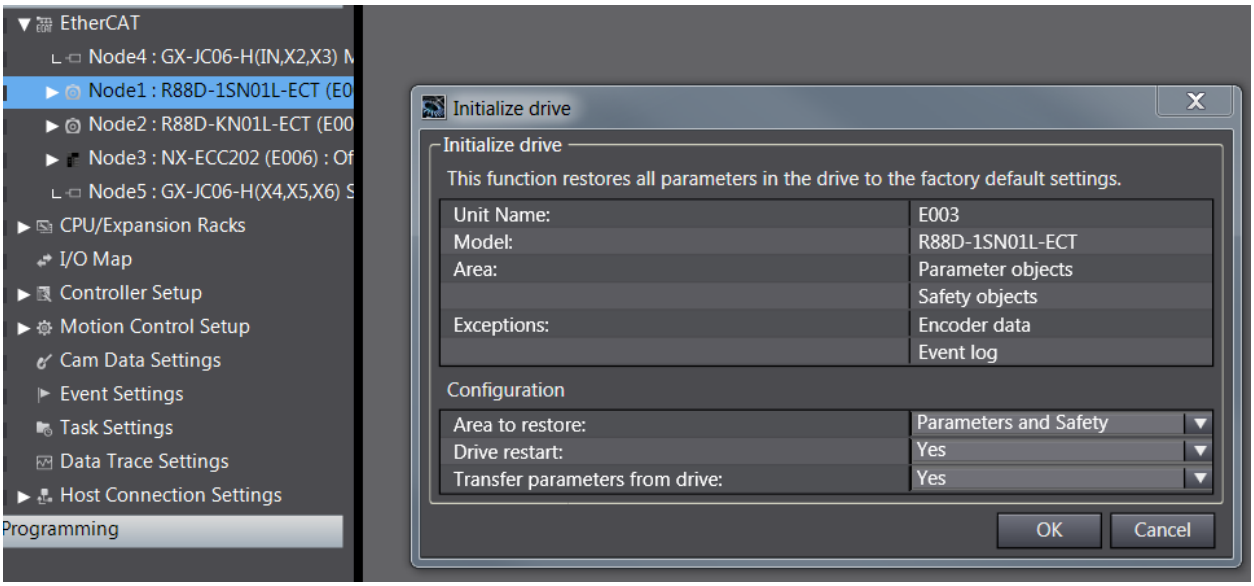
This assumes that the EtherCAT header file created from the EC Engineer xml file looks something like:

```
#define Slave_0_6040_0_Controlword  ECAT[0].IO[0].Data
#define Slave_0_607A_0_Targetposition  ECAT[0].IO[1].Data
#define Slave_0_60FF_0_Targetvelocity  ECAT[0].IO[2].Data
#define Slave_0_6071_0_Targettorque  ECAT[0].IO[3].Data
#define Slave_0_6060_0_Modesofoperation  ECAT[0].IO[4].Data
#define Slave_0_60B8_0_Touchprobefunctio  ECAT[0].IO[5].Data
#define Slave_0_607F_0_Maxprofilevelocit  ECAT[0].IO[6].Data
#define Slave_0_603F_0_Errorcode  ECAT[0].IO[4096].Data
#define Slave_0_6041_0_Statusword  ECAT[0].IO[4097].Data
#define Slave_0_6064_0_Positionactualval  ECAT[0].IO[4098].Data
#define Slave_0_6077_0_Torqueactualvalue  ECAT[0].IO[4099].Data
#define Slave_0_6061_0_Modesofoperationd  ECAT[0].IO[4100].Data
#define Slave_0_60B9_0_Touchprobestatus  ECAT[0].IO[4101].Data
#define Slave_0_60BA_0_Touchprobepos1pos  ECAT[0].IO[4102].Data
#define Slave_0_60BC_0_Touchprobepos2pos  ECAT[0].IO[4103].Data
#define Slave_0_60FD_0_Digitalinputs  ECAT[0].IO[4104].Data
```

Appendix A

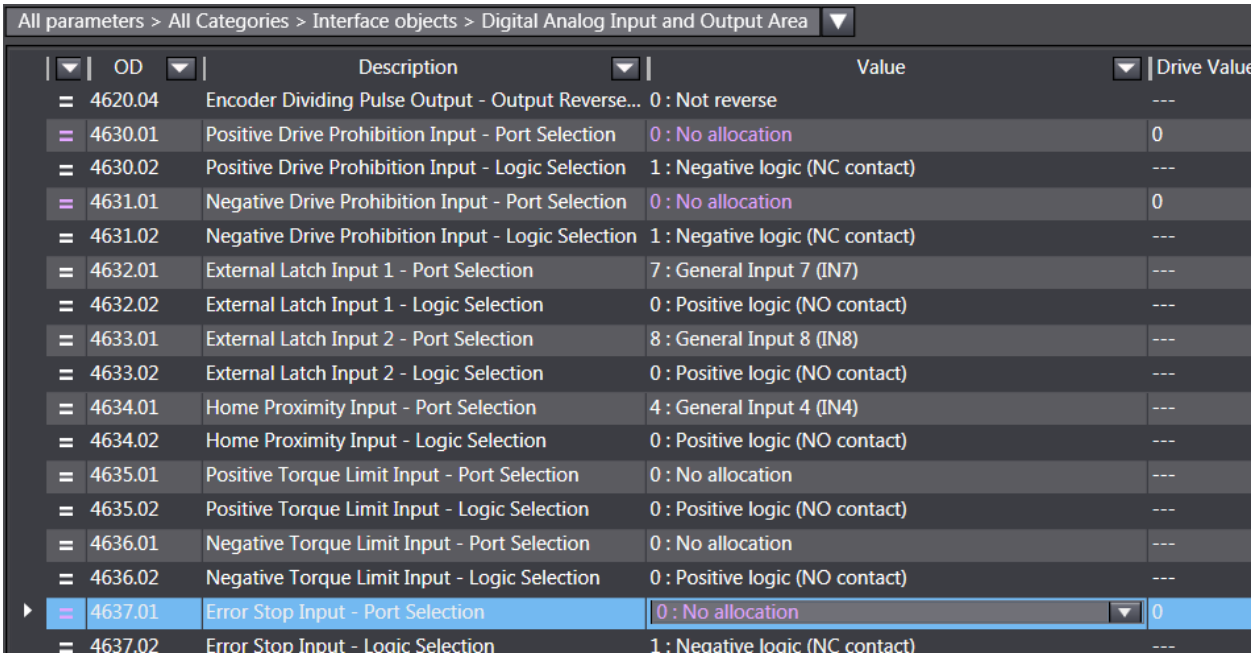
1S Drive Intialization

Connect up with USB, start Sysmac Studio, add the correct drive to the EtherCAT network. Right click on the node in the Multiview explorer tab, click Direct Connection (USB) -> Online. Then right click on the node and pick Initialize. Choose Parameters and Safety.



You will need to set the safety variables now according to how you have the drive wired. If using no safety or if using EtherCAT safety, in the drive parameters set 4637.01 (Error Stop Input) = 0:No allocation.

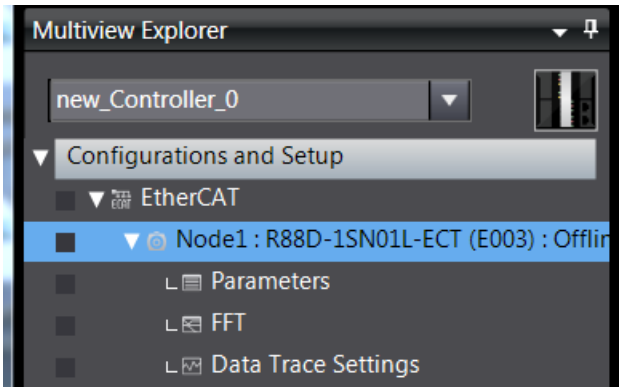
If using no hard limits in the drive parameters set 4630.01 (Positive Drive Prohibition) = 0:No allocation and 4631.01 (Negative Drive Prohibition) = 0:No allocation.



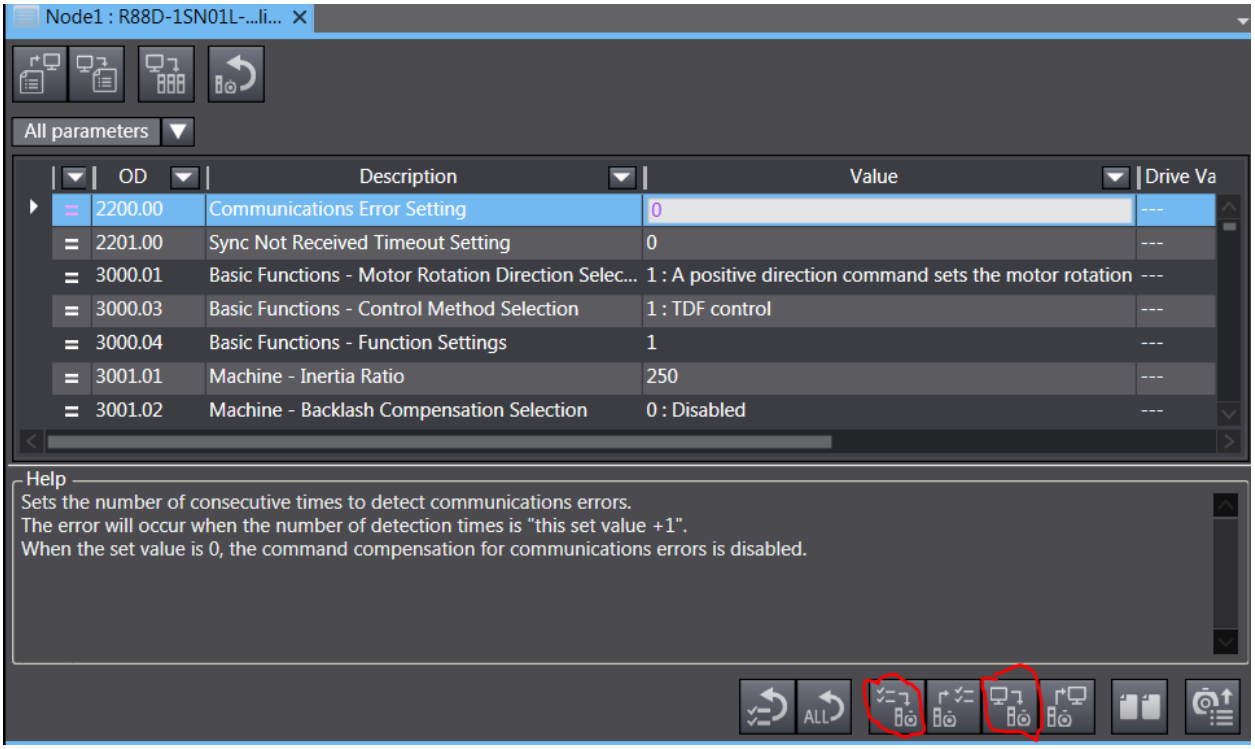
Appendix B

1S Drive parameter setting

Connect up with USB, start Sysmac Studio, add the correct drive to the EtherCAT network. Right click on the node in the Multiview explorer tab, click Direct Connection (USB) -> Online. Clicking on the right arrow beside the node will open the following tree:



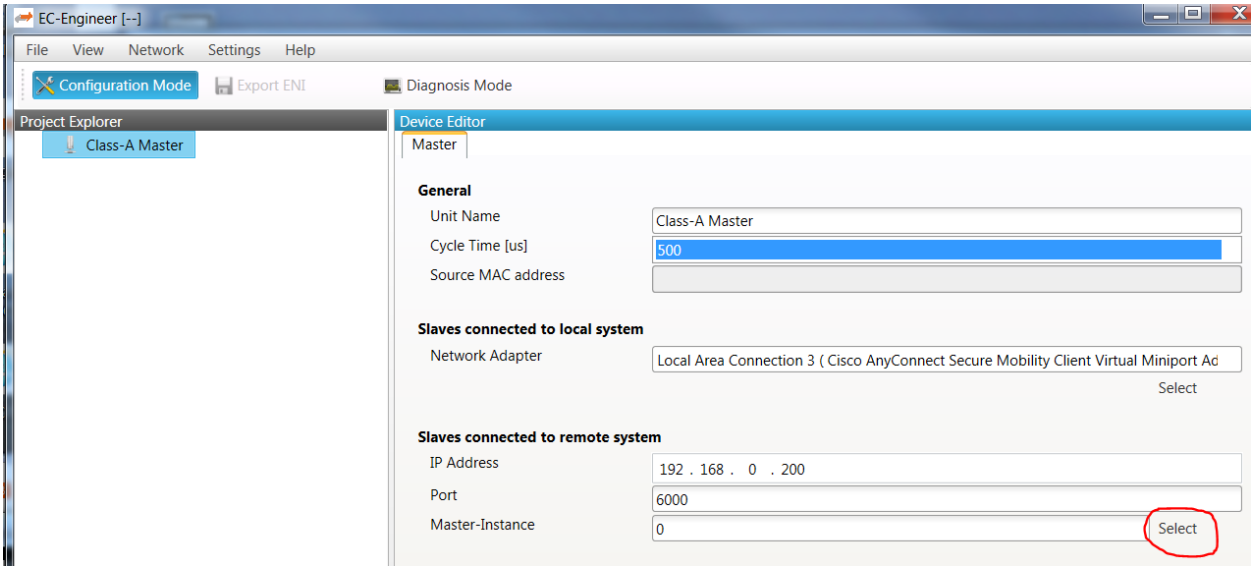
Double click on the Parameters. This will open a tab with the drive parameters. The values can be changed. Then either choose the individual parameters, right click, and choose Selection to Drive, or click the Transfer Selection to Drive or the Transfer All to Drive buttons to the lower right:



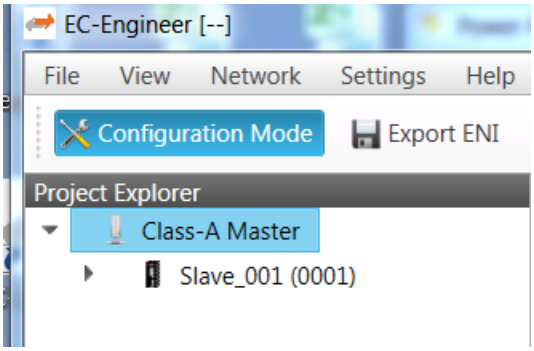
Appendix C

Scan network with EC engineer.

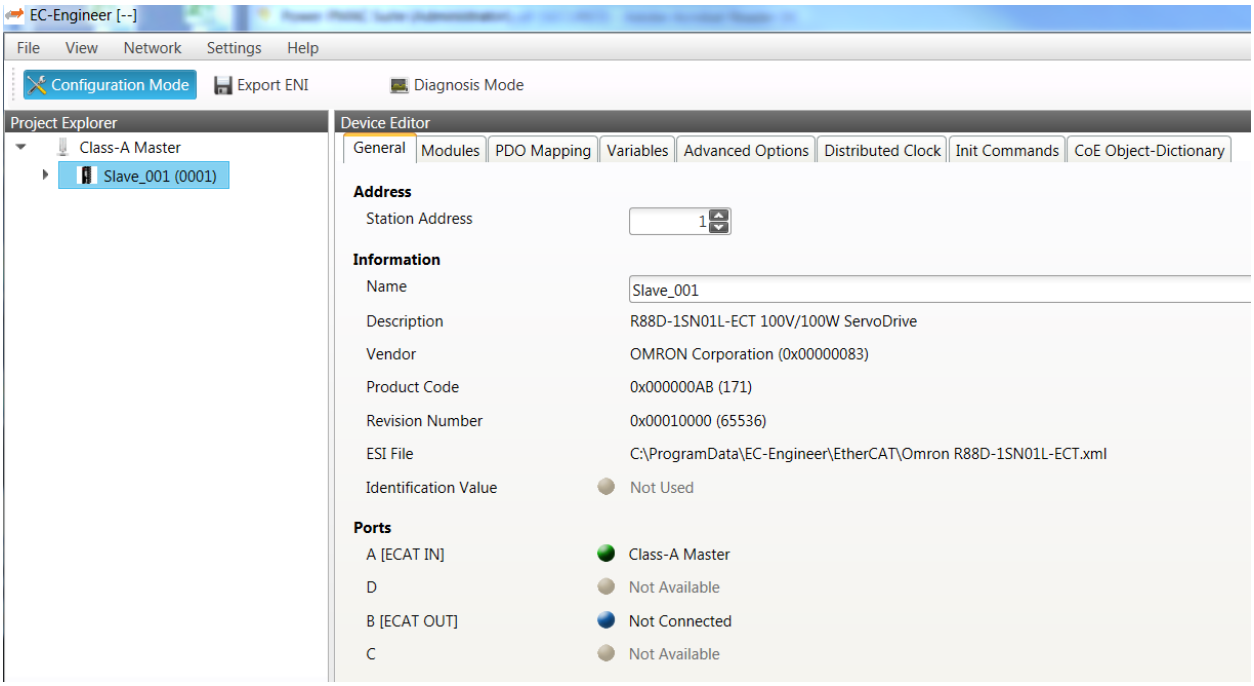
Open EC Engineer. Select EtherCAT Master Unit (Delta Tau) on the Start Page tab. Set the Cycle time in uSeconds to match the Sys.ServoPeriod set in the PowerPMAC. For a 2 kHz servo period Sys.ServoPeriod=0.5, and Cycle Time will be 500. Set the IP Address of the PowerPMAC, keep Port=6000, and probably Master-Instance=0. Click the 'Select' button to the right of the Mast-Instance.



Right click on the Class-A Master in the Project Explorer pane and select 'Scan EtherCAT Network'.



The slave devices in your network should now appear under the Class-A Master in the Project Explorer pane. If you get errors but are sure that your ESI files are correct, go into the ESI manager, delete the device you are trying to connect, then add the file again. This will clear up errors if a new version of a device was not in the ESI file when you first added it. Errors are shown in the Messages pane of EC engineer. A typical error might be that a device was not found for a particular VendorID (Omron is 0x83), ProductCode (a 100W/120V 1S drive is 0xAB), or Revision (at time of writing 0x10000 for the 1S drives).



Clicking on the slaves should show the station address and information, as well as the associated ESI file in the General tab of the Device Editor pane.

Appendix D

Startup PLC

In Solution Explorer right click on PMAC Script Language->PLC Programs->Add->New Item… Pick a name for the PLC and click Add button.

```
Open plc plcStartup
        local rStartupTimer;
        while(sys.ecatMasterReady==0){};
        ECAT[0].Enable=1;
        rStartupTimer =Sys.Time+2;              //time in seconds
        while(rStartupTimer >Sys.Time){};
        cmd"&1enable";
        rStartupTimer =Sys.Time+5;
        while(rStartupTimer >Sys.Time){};
        //cmd"&1b1r";
        disable plc 1;
close
```

Open pp_startup.txt and add this line:

Enable plc plcStartup