1.General

1.1 THIS DOCUMENT

The purpose of this document is to provide practical assistance to help set up a Eurotherm controller on a DeviceNet network. It is not a treatise on DeviceNet.

In order to make it practical specific hardware has to be used and in these examples the Rockwell Allen Bradley SLC500/03 processor has been used with a 1747-SDN Scanner module and a 1770-KFD RS232 Interface together with Rockwell RSLinx, RSNetWorx and RSLogic500. A familiarity with these software tools is assumed.

For other hardware the interfaces will be different but the basic process required will the same.

1.2 INTRODUCTION

DeviceNet was designed as a low-level network for communications between Programmable Logic Controllers and relatively simple devices such as limit switches and I/O clusters.

Although the Eurotherm Advanced DeviceNet products allow large I/O messages (2700 has 120 words of input and output data), practical systems are constrained by the total I/O space available in the scanner being used – for example 300 words of combined I/O data in the Allen Bradley 1747-SDN/B Scanner Module - and by the amount of traffic permissible on the network. To put this in perspective, a single Eurotherm device can generate and consume as much information as 15 DeviceNet 64 bit digital I/O modules! Not only is this a great deal of data, but it also takes 15 times as long to transmit as a simple I/O device.

Eurotherm controllers offer DeviceNet communications on three different levels. The first two only require configuration of the network, the third requires plc programming as well.

1. The simplest uses standard pre-configured parameters which allow a network to be set up with little configuration work, the most common parameters then being available in the plc.

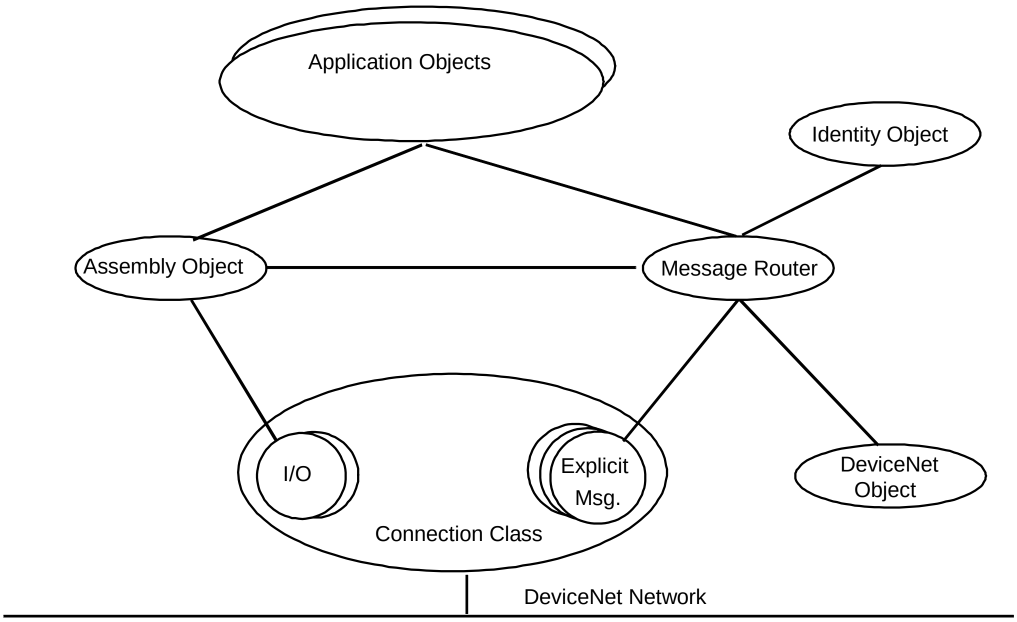
2. The next level uses ‘indirection’ techniques which gives access to any parameter in the controller via its Tag address. Again in this application the parameters are available in the plc.

3. The third level of communications uses ‘explicit message program control’ on DeviceNet. This allows the plc to control the network messages directly by issuing a Request and waiting for a Response. This can reduce network traffic.

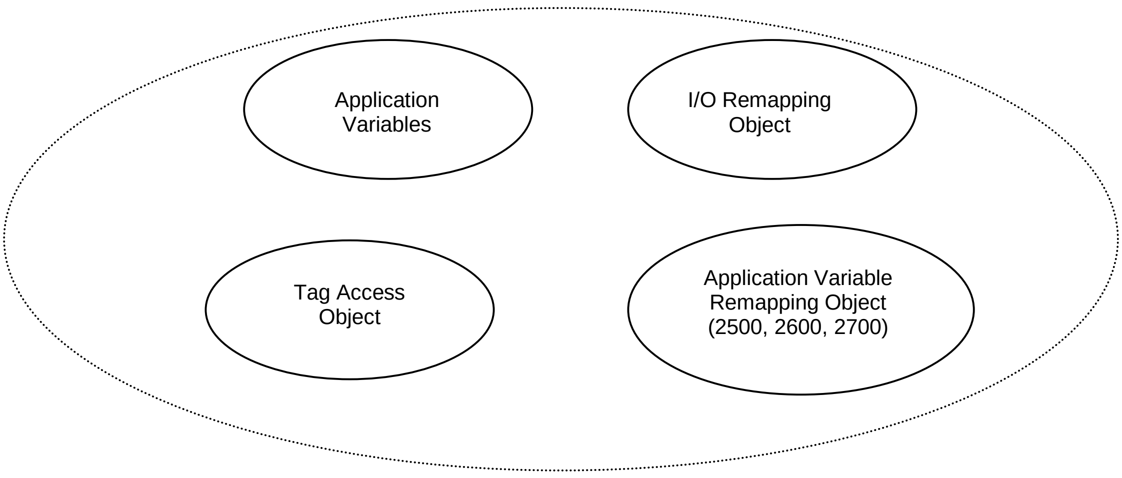
On the 2400, 2500, 2600 & 2700 explicit messaging can be used to read or write blocks of 16 parameters, such as may be used to up / download a recipe program.

1.3 THE DEVICENET OBJECT MODEL

The mix of objects within a DeviceNet device is usually depicted graphically. The diagram below shows the overall distribution of DeviceNet objects within the interface.



This is very much a standard object mapping, containing the usual mandatory DeviceNet objects, as described in the ODVA Specification.



The I/O Remapping Objects are the parameters being read/written on the DeviceNet network
The Application Variable Object is a list of predefined parameters available to be selected as IO Remapped objects.
The Application Variable Remapping Object uses Tag Addresses to add further User defined parameters to the Application Variables Object (and from there to the I/O Remapping Object)
The tag access object requires the use of explicit messaging to read and write blocks of data

1.4 .THE DEVICENET IMPLEMENTATION

This implementation provides a common DeviceNet interface firmware for 2400/2500/2600/2700. This takes the form of an internally fitted DeviceNet gateway. The physical implementation is identical for 2400/2600/2700, but is a separate board for 2500. The 2200 and the PC3000 are implemented in a different way but are included in the examples.

The implementation provides a Group 2 only Server, supporting all optional DeviceNet features, i.e.:

Polled I/O
COS/Cyclic
Bit Strobe
Heartbeat Messaging
Device Shutdown Messaging

For further information on the definitions of these functions, refer to the ODVA DeviceNet specification.
DeviceNet masters exchange data with DeviceNet slave devices by using I/O messages; there are four different types of I/O messages: strobe, poll, change-of-state (COS), and cyclic.

The strobe I/O message, which is 8 bytes (64 bits) in length, is broadcast by the master to all devices on the network. Each of the bits corresponds to one of the node addresses (0 to 63). Each device on the network that supports strobe I/O messages, responds by placing its input data, which can be a maximum of 8 bytes per device, on the network. This is supported by some Eurotherm controllers but is only likely to be useful if one or two parameters are required from a large number of devices.

The poll I/O message, which can be a maximum of 255 bytes in length, is sent by the master to a specific slave device on the network. This is effectively point-to-point communication. The Input data from the slave device is read continuously and the output data to the device is written continuously

The change-of-state (COS) I/O message is sent by a slave device to the master whenever the state of the input data changes or, in some cases, at a user-configurable rate (heartbeat). A COS I/O message does not solicit a response from the scanner. This is a point-to-point communication.

The cyclic I/O message is sent by a slave device to the master at a user-configurable rate.. A cyclic I/O message does not solicit a response from the master. This is a point-to-point communication.

I/O Message	Strobe	Poll	COS	Cyclic
2600 / 2700	√	√	√	√
2500	√	√	√	√
2400	√	√	√	√
2200		√		
PC3000		√		

The granularity of data transmitted on a DeviceNet network is one byte. This means that even if just a single bit of data needs to be transmitted a whole byte has to be used.

1.5 HARDWARE

The 2400, 2600,2700 use a plug in DeviceNet module. The 2200 and 2500 have to be ordered as DeviceNet ready products. The PC3000 requires a DeviceNet Communications Module.

1.6 TAG ADDRESSES

The default communications in all Eurotherm controllers is Modbus. Whilst Modbus is not available in controllers set up for DeviceNet the Tag Address map used to identify parameters within the controllers is based on the same tables.

To read and write the very common parameters it is not really necessary to refer to these Tag Address maps. However when other parameters are required to be used it may be necessary to find the particular parameter addresses.

The best way to find a parameter address is on iTools, otherwise it will be necessary to refer to the instrument documentation. Some tag details are included in the tables at the end of each instrument section. On the 2700 it is possible to determine a particular parameters address via the operator interface.

Documents:

Files: 2600_2700tag.xls, 2600_2700programmer.xls

2.Procedure for Setting up a Network

There are 5 stages to setting up a network.

2.1 PHYSICAL WIRING

The DeviceNet linear bus trunk line – drop line topology may be constructed from either DeviceNet Thick Cable or DeviceNet Thin Cable or a combination of both. DeviceNet Thick Cable enables long trunk line distances and sturdier trunk or drop lines while the use of DeviceNet Thin Cable provides easier routing and termination of either trunk lines or drop lines. Table 1 lists recommended DeviceNet Thick Cable and DeviceNet Thin Cable from Belden Wire & Company.

Belden DeviceNet Thick Cable & Thin Cable

Belden #	Conductors	Type
3082A	2 – 15 AWG/ 2 – 18 AWG	Trunk (Thick)
3083A	2 – 15 AWG/ 2 – 18 AWG	Trunk (Thick)
3084A	2 – 22 AWG/ 2 – 24 AWG	Drop (Thin)
3085A	2 – 22 AWG/ 2 – 24 AWG	Drop (Thin)

The total amount of trunk line allowable on the network depends upon the data rate and the type of cable used – thick or thin. The cable distance between any 2 points in the cable system must not exceed the *Maximum Cable Distance* allowed for the baud rate. For trunk lines constructed of only 1 type of cable refer to Table 2 to determine the *Maximum Cable Distance* based on the data rate and the type of cable used. Cable distance between two points includes both trunk line cable length and drop line cable length that exists between the two points.

Terminating resistors are required on each end of the trunk line. Specification for the resistors: 121ohm, 1% metal film, ¼ watt. Drop lines up to 6m (20 feet) each are permitted, allowing 1 or more nodes to be attached. DeviceNet allows branching structures only on a drop line. Termination resistors should never be installed at the end of a drop line, only at the ends of the trunk line. Up to 64 nodes are supported.

Network Length	Varies w/speed, up to 4000m possible w/repeaters		
Baud Rate	**125**	**250**	**500**
Thick trunk	500m (1,640ft)	250m ((820ft)	100m (328ft)
Thin trunk	100m (328ft)	100m (328ft)	100m (328ft)
Max drop	6m (20ft)	6m (20ft)	6m (20ft)
Cumulative drop	156m (512ft)	78m (256ft)	39m (128ft)

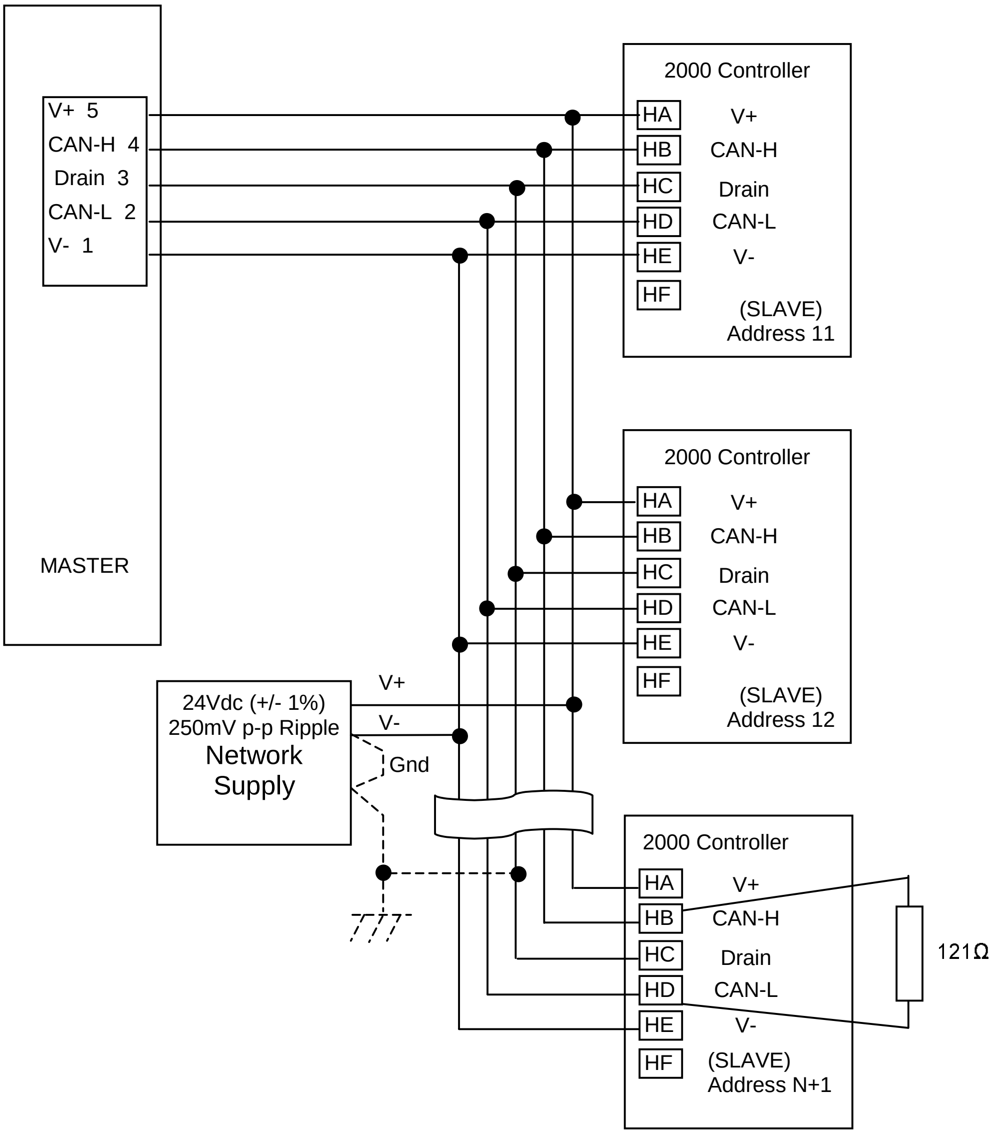
2.1.1 Terminal Description

2200 2400 2600 2700 Terminal	Std Label 2500	Colour	Description
HA	V+	Red	DeviceNet network power positive terminal. Connect the red wire of the DeviceNet cable here. If the DeviceNet network does not supply the power, connect the positive terminal of an external 11-25 Vdc power supply.
HB	CAN_H	White	DeviceNet CAN_H data bus terminal. Connect the white wire of the DeviceNet cable here.
HC	SHIELD	None	Shield/Drain wire connection. Connect the DeviceNet cable shield here. To prevent ground loops, the DeviceNet network should be grounded in only one location.
HD	CAN_L	Blue	DeviceNet CAN_L data bus terminal. Connect the blue wire of the DeviceNet cable here.
HE	V-	Black	DeviceNet network power negative terminal. Connect the black wire of the DeviceNet cable here. If the DeviceNet network does not supply the power, connect the negative terminal of an external 11-25 Vdc power supply.

For PC3000 see supplied installation documents.

Note that the DeviceNet network is powered by an external independent 24v supply which is separate from the internal powering of the individual controllers themselves.

2.1.2 Typical Wiring Diagram



2.1.3 Setting up the Controller

The configuration of controller for DeviceNet is slightly different for each type of controller but, having selected DeviceNet, there are only 2 parameters to set up – Baud rate and Address.

Valid Baud rates are 125k, 250k and 500k, and addresses may be from 0 to 63. Generally use 500k unless the network is longer than 100m. There is no priority in the addressing – all addresses are treated equally.

The 26/2700 and 2400 use a DeviceNet communications module in slot H. This may be added by the User. The 2200 and 2500 have to be purchased from the factory as DeviceNet ready instruments. A DeviceNet Comms module has to be purchased for PC3000.

In the 26/2700, 2400 and 2200 the configuration is set on the HA comms. The instrument must be in configuration mode to select DeviceNet communications and to set the baud rate. The address may be set in operating mode at access level 3.

The 2500 uses a DIL switch.

PC3000 is set using the DeviceNet slave function block as described in Section 7.

2.2 SETTING UP THE SCANNER

Use RSLinx and the Tools/Node Commissioning on RSNetWorx to set up the Scanner address and Baud Rate at which the network is to run. Baud rate cannot be changed ‘on-line’ it is only changed by closing down and re-starting the network.

Scanner is set up as the master.

Register all the required Eurotherm Electronic Data Sheets using the EDS Wizard in the Tools menu of RSNetWorx.

EDS Files are available from Eurotherm.

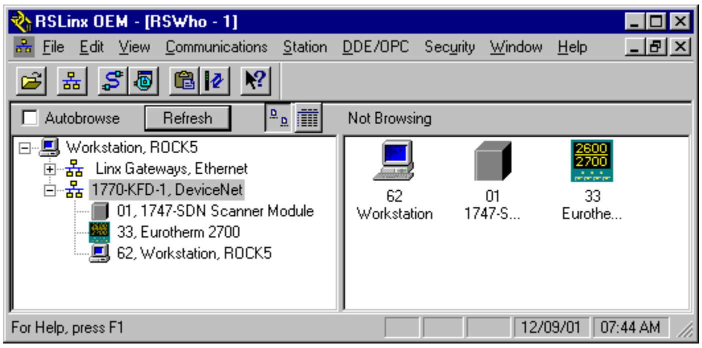
Files: 2200.eds 2400.eds 2500.eds 2600.eds 2700.eds PC3kDNS.eds
www.?????????????

Hint for Profibus Users: Note the EDS file is unique and applies to the specific device. The device itself, ***not*** *the .eds file, is configured for the DeviceNet application.*

2.3 ESTABLISHING COMMUNICATIONS

With the DeviceNet network correctly wired up and powered, and the scanner and controllers configured with valid unique addresses and the same baud rate, communications will commence. If there is no communications check the common baud rate, unique addresses, 24v supply, the wiring, the termination resistors and finally the devices themselves.

In these examples the Scanner in slot 6 has been set to address 1, the RS232 interface to 62 and the 2700 to address 33. All baud rates are 500k.



RSLinx will show the active items on the network.

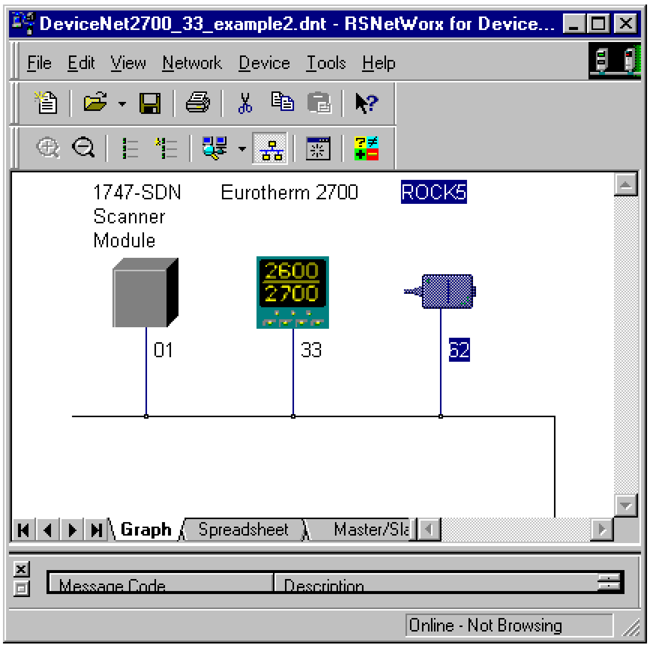
RS Linx showing the computer via the RS232 interface the scanner and the 2700 at address 33.

At this stage communications is active. Looking at Comms/H Module/Activity (Level 3 access) on the 2700 will show that it is active. At this stage though it is only 'Hardware' communications and there is no transfer of data.

Data transfer has to be set up as a separate operation which involves both setting up the 2700 so that it knows what parameters it has to handle and setting up the scanner to make use of these parameters.

Parameters are either INPUT parameters read by the Scanner or OUTPUT parameters written by the scanner.

One way to configure the 2700 INPUT parameters and OUTPUT parameters is via the DeviceNet network. To do this the following examples use RSNetWorx.



The network as seen on the graph view of RSNetWorx.

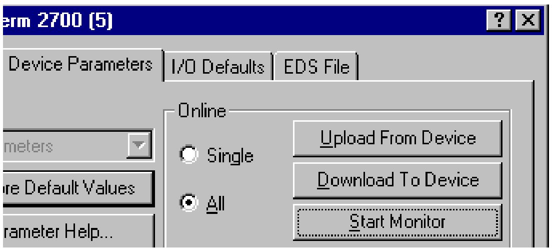
Looking on the master/slave view the 2700 will be under the list 'Slaves w/out master.'

The 1747-SDN scanner module and the host computer (via the 1770 RS232 unit) will be in the list as 'Devices which are not slaves'.

Note that going ON-LINE with RSNetWorx (Menu Network – On Line) or F10 will first give a warning message:

To Upload or download first right click on the device and select upload. In the case of Eurotherm Instruments do not download unless you are absolutely certain that the RSNetWorx file (*.DNT) is what you want. It is possible, for example, to overwrite the address of a slave.

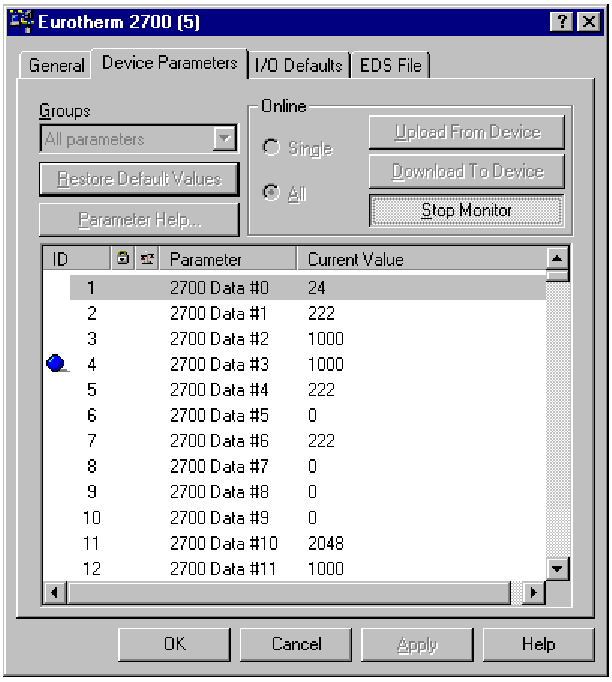
Having uploaded from the instrument and going to the Properties – Device Parameters tab you can see the on-line controls.



The radio buttons give the choice of single parameter upload/download or all parameters.

It is suggested that with Eurotherm devices Upload using 'All' but download each parameter one by one using 'Single'

The 'Start Monitor' button will continuously scan 'All' or a 'Single' parameter(s) giving the current values. This is a good way to double check parameters that you have modified as part of the network configuration.



The 'Monitor' mode showing live values of the 'predefined' parameters in the 2700.

Output parameters can clicked on and be written to.

The DeviceNet files from RSNetWorx yhat are included are only designed to allow them to be opened and browsed. There is NO purpose in downloading them to your network as the DeviceNet configuration also requires the actual physical instruments identities to match the original in the saved files.

For example, replacing one 2700 at address 33 with another at address 33 will cause the network to assume there is NO instrument at address 33. First delete the original from the scanlist and then add the new one.

3.Transferring Data – 2600 / 2700

This has been set up with a 2700 controller but the 2600 communications is identical.

The 2700 is a "Generic device type, Group 2 only server." As a DeviceNet slave or server the 2700 offers up to 60 analogue INPUT parameters TO the master or client and may receive up to 60 analogue OUTPUT parameters FROM the master or client.

The design of DeviceNet in the 2700 has been to make it easy to read the most common parameters but retain the possibility of a user selecting any other parameter within the controller. This information is all in the 2700 electronic data sheet registered in RSNetWorx, 2700.eds and is the way the parameters are viewed through the 2700 device properties in RSNetWorx.

The 2700 DeviceNet parameters are divided into 8 sections
A list of instrument parameters pre-defined and immediately available for selection on the INPUT or OUTPUT tables
a list of additional user defined parameters to add to the OUTPUT table
a list of additional user defined parameters to add to the INPUT table
the actual INPUT table of parameters to be READ by the DeviceNet client
the actual OUTPUT table of parameters to be WRITTEN by the client
the Tag address of additional parameters to be READ by the DeviceNet client
the Tag address of additional parameters to be WRITTEN by the client
a group that can be used for explicit messaging

EDS List	Quantity	Description
0 to 160	161	Predefined parameters 2700 Data #0 to #160
168 to 183	16	User defined OUTPUT parameters 2700 Data #168 to #183
184 to 199	16	User defined INPUT parameters 2700 Data #184 to #199
194 to 253	**60**	**Enter #<number> of required INPUT parameters**
254 to 313	**60**	**Enter #<number> of required OUTPUT parameters**
314 to 330	16	Enter Tag Address of user defined INPUT parameters
330 to 346	16	Enter Tag Address of user defined OUTPUT parameters
346 to 351	Specialist Parameters – explicit messaging	

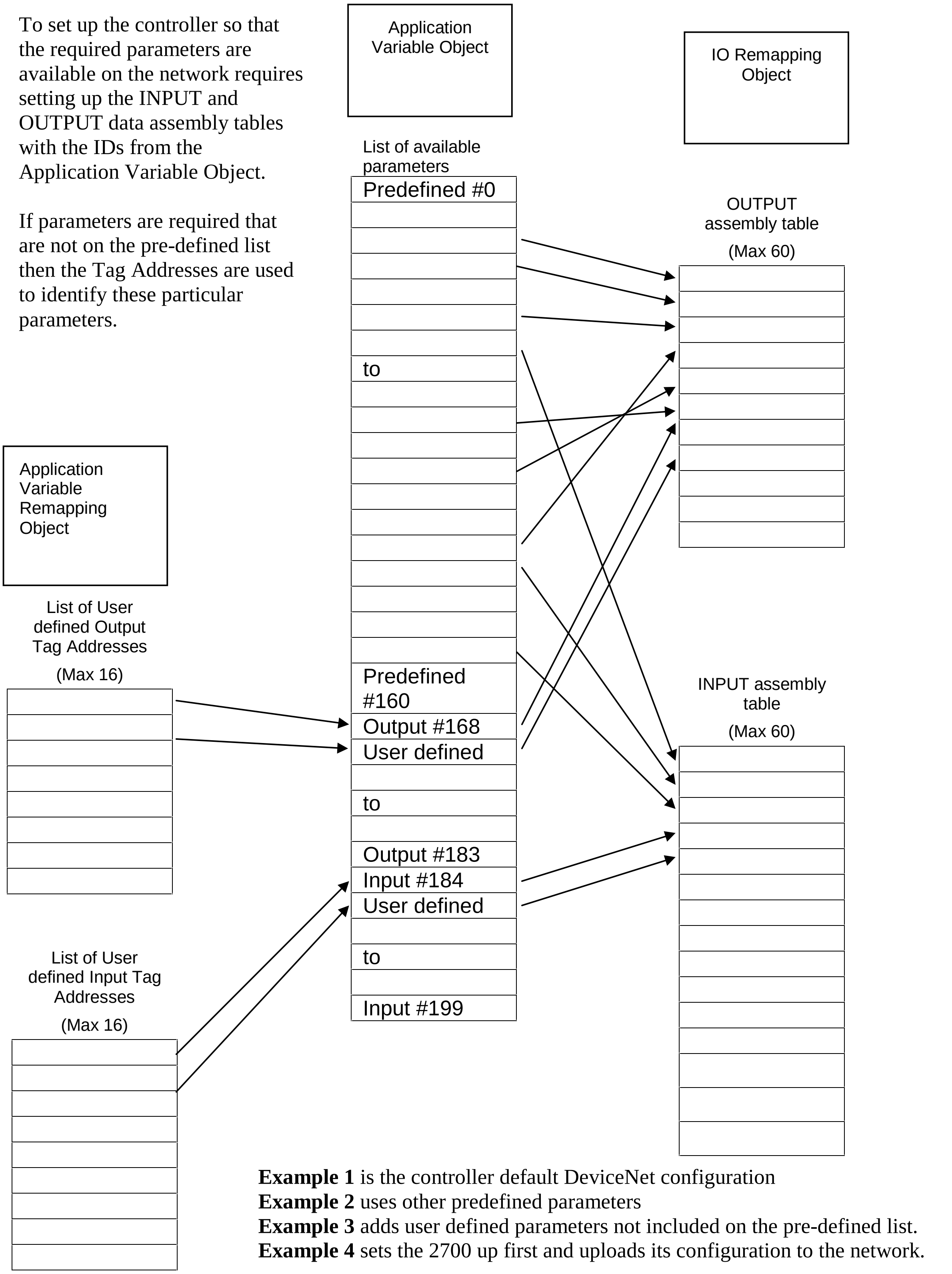
This information can be seen by inspecting the 2700.EDS file in a text editor.

To set up the controller so that the desired parameters can be read and written involves setting up the INPUT and OUTPUT data assembly tables (highlighted in the table above).

This is illustrated by the following diagram and examples.

To set up the controller so that the required parameters are available on the network requires setting up the INPUT and OUTPUT data assembly tables with the IDs from the Application Variable Object.

If parameters are required that are not on the pre-defined list then the Tag Addresses are used to identify these particular parameters.



Example 1 is the controller default DeviceNet configuration
Example 2 uses other predefined parameters
Example 3 adds user defined parameters not included on the pre-defined list.
Example 4 sets the 2700 up first and uploads its configuration to the network.

3.1 DEFAULT EXAMPLE 1

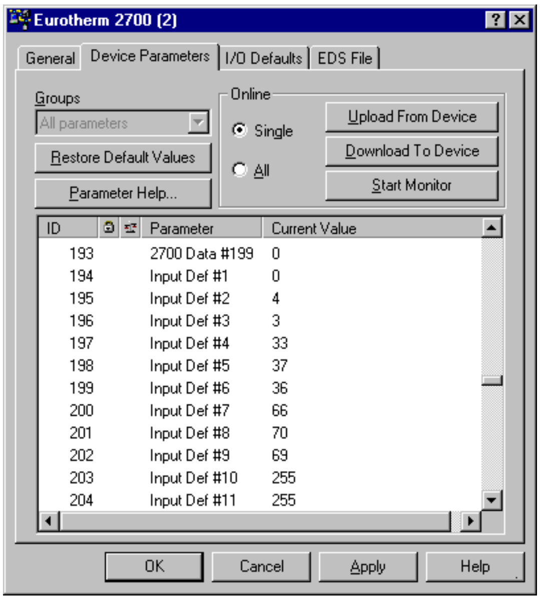
This is called the default example because this is the way the controller is delivered, as new, from Eurotherm. It is also the way the controller would be set up if a new DeviceNet module was added to an existing controller. There is an exception to this statement which is highlighted in Example 4.

The requirement is to be able to read and write the following parameters. Firstly the input parameters. The #numbers in the table below come from the Attribute ID Table in Section 3.5.

Input Parameter	#<number>
Process Variable (Loop 1)	0
Working Setpoint (Loop 1)	4
Working Output Power (Loop 1)	3
Process Variable (Loop 2)	33
Working Setpoint (Loop 2)	37
Working Output (Loop 2)	36
Process Variable (Loop 3)	66
Working Setpoint (Loop 3)	70
Working Output (Loop 3)	69
TOTAL LENGTH	18 bytes

Example 1 – Required INPUT Data assembly table

Right click on the 2700 icon in RSNetWorx and select Properties –to Device Parameters tab. The #numbers in the table above have to be entered into the first nine INPUT table parameters 'Input Def #1' to 'Input Def #9' as per the figure below.



This is the Device Parameters tab of the properties of the 2700 DeviceNet slave.

Add the required Attributes (#numbers) for the input table.

Download single parameters as they are entered.

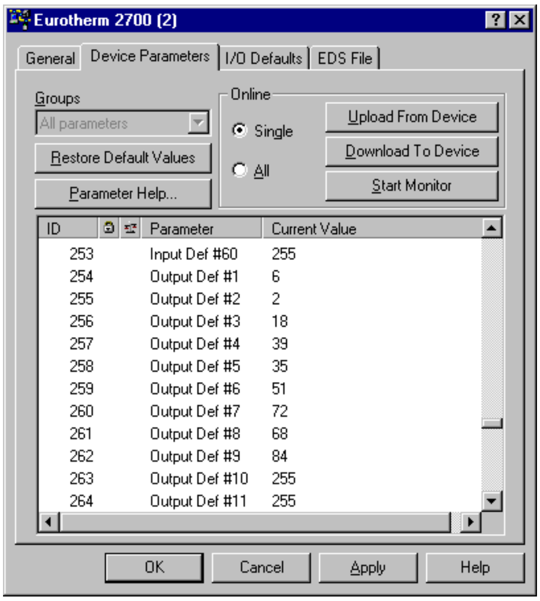
All the remaining parameters of the input table MUST be set to 255 to indicate that they are not being used – in this case Input Def#10 to #60.

Notice that there are 9 parameters giving a data table size of 18 bytes. This information will be required later to set up the scanner.

The same now has to be done for the OUTPUT parameters. The # numbers in this table have to be entered into the first nine OUTPUT table parameters 'Output Def#1' to 'Output Def#9' as per the figure below taken from the Device Parameters tab of the properties of the DeviceNet slave.

Output Parameter	#<number>
Target Setpoint (Loop 1)	6
Target Output Power (Loop 1)	2
Auto/Manual Select (Loop 1)	18
Target Setpoint (Loop 2)	39
Target Output Power (Loop 2)	35
Auto/Manual Select (Loop 2)	51
Target Setpoint (Loop 3)	72
Target Output Power (Loop 3)	68
Auto/Manual Select (Loop 3)	84
TOTAL LENGTH	18 bytes

Example 1 – Required Output Data Assembly Table



Adding the IDs for the output table.

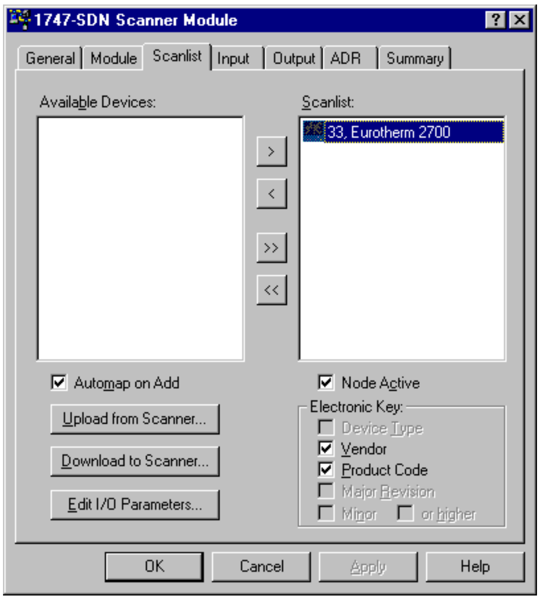
Download single parameters as they are entered.

All the remaining parameters of the output table MUST be set to 255 to indicate that they are not being used – in this case Output Def#10 to #60.

Note that the output data table length is 9 parameters or 18 bytes. This will be needed later to set up the scanner.

With all these parameters set in the controller the DeviceNet set up of the controller is completed. It now remains to set the scanner up to poll these parameters.

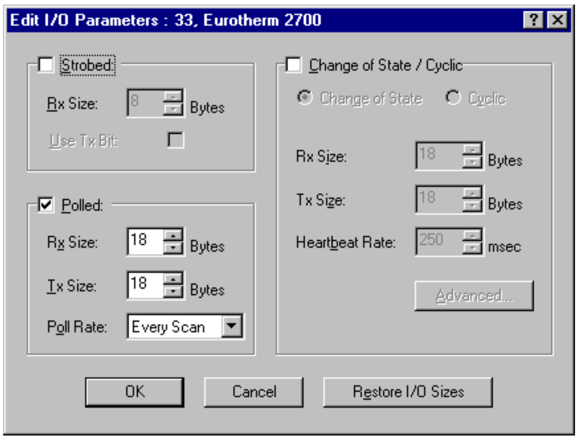
Setting up the scanner to read/write these parameters, right click on the Scanner Icon and go to the Scanner properties. Ensure under the 'Module' Tab that the Scanner is allocated to the correct slot.



Select the Scanlist tab.

Firstly, if the 2700 is not already in the scanlist highlight it and click the right arrow to add it.

Select the 2700 on the scanlist and then click on 'Edit I/O Parameters'.



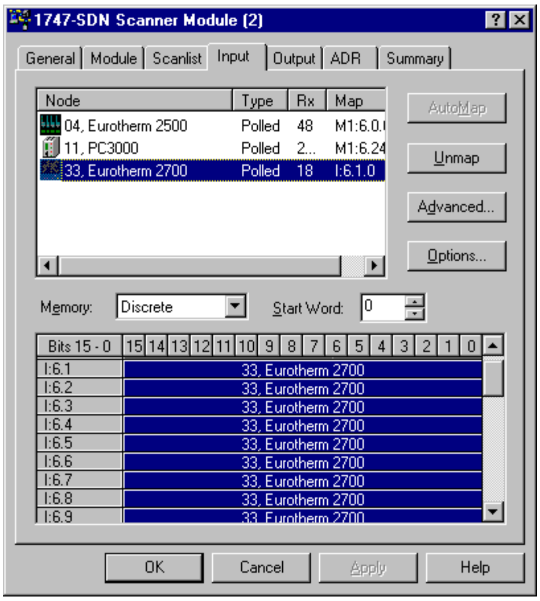
Select 'Polled' mode
Set the Rx (or INPUT) byte size to 18
Set the Tx (or OUTPUT) byte size to 18.

THIS MUST MATCH THE LENGTH OF THE PARAMETERS SET UP IN THE 2700.

In this example:
9 parameters = 9 words = 18 bytes.

This is now sufficient to establish communications between the scanner and the controller.

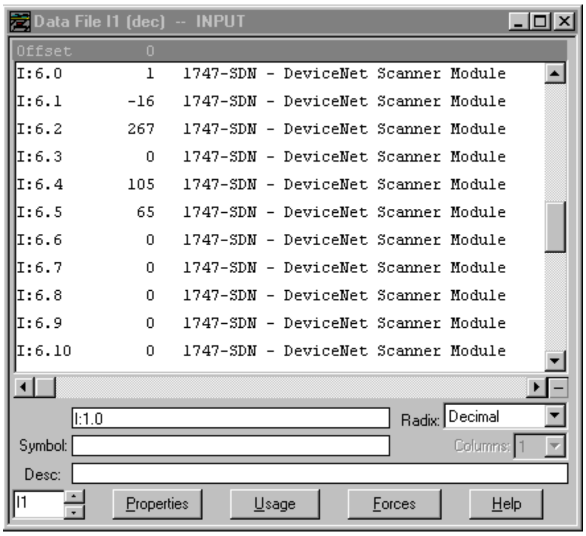
To make use of the parameters they have been mapped into discrete I/O. The scanner is in slot 6 and the INPUT values are in I:6.1 to I:6.9 and the OUTPUT values in O:6.1 to O:6.9



Click OK and now Download this into the scanner module.

The SLC500 has to be in PROGRAM mode. Once the data is downloaded power cycle the plc or reboot the Scanner.

In the plc using RSLogix500 the actual controller parameter values are



The parameters will be in the same order as they were defined in the controller INPUT table, i.e.
I:6.1 is Loop 1 PV = -16
I:6.4 = Loop 2 PV = 105
I:6.7 = Loop 3 PV = 0.

Note: these are integers.
If PV1 is displayed with 1 decimal point then the –16 above represents –1.6.
Similarly a setpoint of 10.0 would be sent as 100.

The parameters can now be used just like any other I/O in the plc.

Note that the setpoints are outputs from the plc. If an operator were to change a setpoint on the instrument itself, it would immediately be overwritten from the plc by the next scan of the DeviceNet network.

Note further that if the value written by the plc is ‘out of range’ as far as the controller is concerned it is silently ignored. This feature is required if, for example, the PID terms are in the output list. In this situation they would be being permanently written to putting the PID block into perpetual debump.

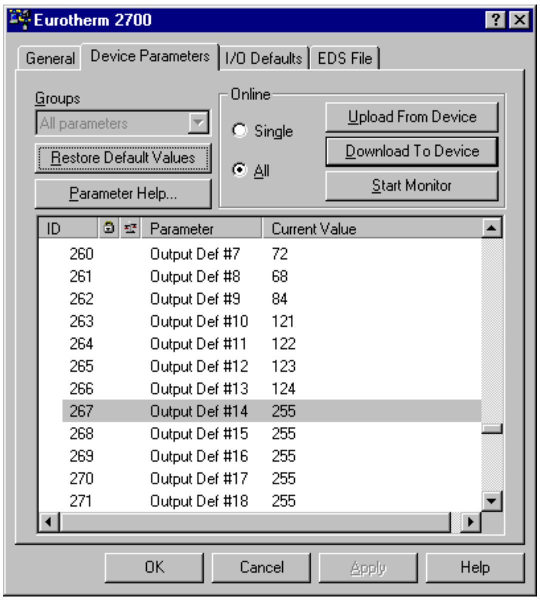
File: Default2700.dnt (RSNetWorx DeviceNet file) (Browse only – do not try to download).

3.2 EXAMPLE 2.

In this example the 2700 User Variables 1 to 4 will be added to the default OUTPUT table and the RealCV 1 to 8 (the results of analogue calculation blocks 1 to 8) will be added to the default INPUT table.
The IDs of the User Values 1 to 4 (ID 121 to 124) are taken off the Attribute ID Table of pre-defined parameters for the 2700 in section 3.5.

Output Parameter	#<number>
Target Setpoint (Loop 1)	6
Target Output Power (Loop 1)	2
Auto/Manual Select (Loop 1)	18
Target Setpoint (Loop 2)	39
Target Output Power (Loop 2)	35
Auto/Manual Select (Loop 2)	51
Target Setpoint (Loop 3)	72
Target Output Power (Loop 3)	68
Auto/Manual Select (Loop 3)	84
User Value 1	**121**
User Value 2	**122**
User Value 3	**123**
User Value 4	**124**
TOTAL LENGTH	26 bytes

Example 2 – Required Output Data Assembly Table



Add the extra IDs for the output table Output Def #10 to #13.

Download single parameters as they are entered.

Unused Output def parameters must be set to 255

Note that the OUTPUT table has 13 parameters and is therefore 26 bytes in length.

Input Parameter	#<number>
Process Variable (Loop 1)	0
Working Setpoint (Loop 1)	4
Working Output Power (Loop 1)	3
Process Variable (Loop 2)	33
Working Setpoint (Loop 2)	37
Working Output (Loop 2)	36
Process Variable (Loop 3)	66
Working Setpoint (Loop 3)	70
Working Output (Loop 3)	69
Analogue Block 1 O/P	
Analogue Block 2 O/P	
Analogue Block 3 O/P	
Analogue Block 4 O/P	
Analogue Block 5 O/P	
Analogue Block 6 O/P	
Analogue Block 7 O/P	
Analogue Block 8 O/P	
TOTAL LENGTH	34 bytes

Example 2 – Required Input Data Assembly Table

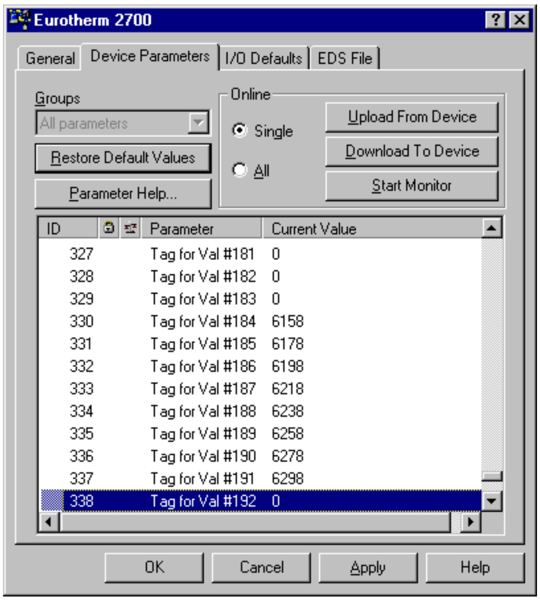
Studying the reference table in of pre-defined parameters (Section 3.5) it is not possible to find these new Analogue block parameters.

In this situation the user has first to define these parameters using their Tag address in the table using the device parameters 314 to 322 as highlighted below.

EDS list	Quantity	Description
0 to 160	161	Predefined parameters #0 to #160
168 to 183	16	User defined OUTPUT parameters #168 to #183
184 to 199	16	User defined INPUT parameters #184 to #199
194 to 253	60	Enter #<number> of required INPUT parameters
254 to 313	60	Enter #<number> of required OUTPUT parameters
314 to 330	16	Enter Tag Address of user defined OUTPUT parameters
330 to 346	**16**	Enter Tag Address of user defined INPUT parameters
346 to 351	Specialist Parameters – block read or write	

The Tag Addresses for these parameters are in the table below.

Parameter	Tag address
Analogue Block 1.RealCV	6158
Analogue Block 1.RealCV	6178
Analogue Block 1.RealCV	6198
Analogue Block 1.RealCV	6218
Analogue Block 1.RealCV	6238
Analogue Block 1.RealCV	6258
Analogue Block 1.RealCV	6278
Analogue Block 1.RealCV	6298



Add the Tag address for each new user defined parameter required in the input table ‘Tag for Val #184 to #191.

Download single parameters as they are entered.

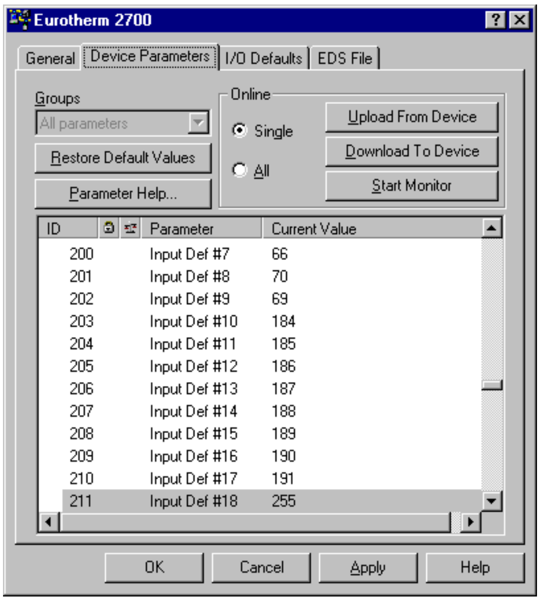
These 8 parameters are Tag for **#184** to **#191** so returning to the INPUT table, add these IDs, values 184 to 191 for the new parameters.

Input Parameter	Attribute ID #<number>
Process Variable (Loop 1)	0
Working Setpoint (Loop 1)	4
Working Output Power (Loop 1)	3
Process Variable (Loop 2)	33
Working Setpoint (Loop 2)	37
Working Output (Loop 2)	36
Process Variable (Loop 3)	66
Working Setpoint (Loop 3)	70
Working Output (Loop 3)	69
Analogue Block 1 O/P	184
Analogue Block 2 O/P	**185**
Analogue Block 3 O/P	**186**
Analogue Block 4 O/P	**187**
Analogue Block 5 O/P	**188**
Analogue Block 6 O/P	**189**
Analogue Block 7 O/P	**190**
Analogue Block 8 O/P	**191**
TOTAL LENGTH	34 bytes

Example 2 – Required INPUT data assembly with the 8 user defined parameters.

Note that the input data table is 19 parameters or 34 bytes long.

Giving the final set up of the controller



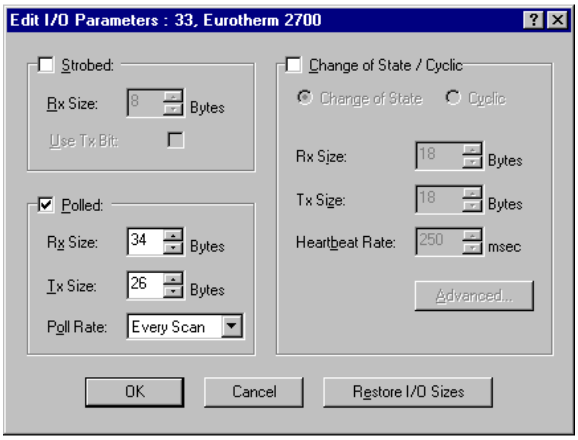
Add the IDs for the new user defined parameters into the input table.

Download single parameters as they are entered.

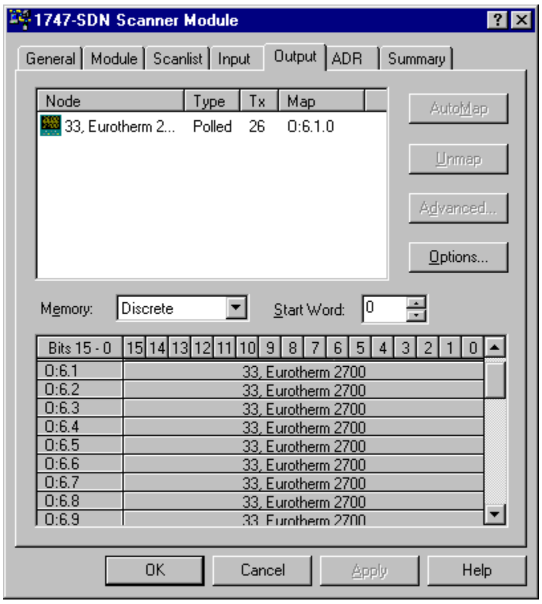
Using this technique of indirection any parameter within the controller may be accessed using its Tag address. There is a maximum of 16 INPUT and 16 OUTPUT parameters that can be defined in this way.

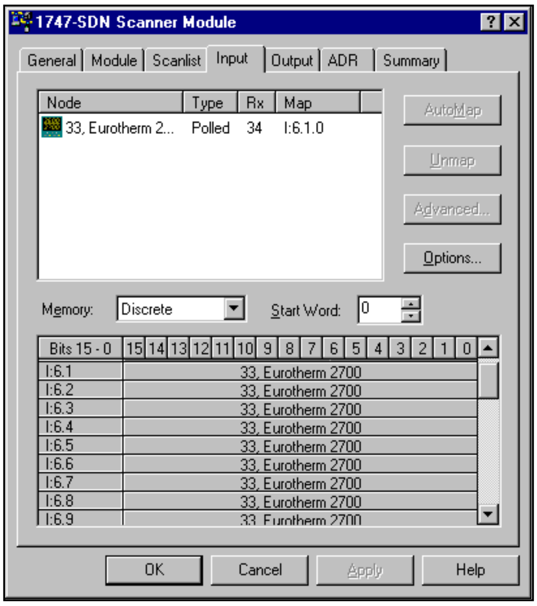
The 2700 is set up as required to match the defined data assembly tables of this example.

The scanner module now has to be updated to include these extra parameters which make the INPUT data assembly 34 bytes and the OUTPUT data assembly 26 bytes.



These may now be mapped into the plc I/O tables or memory.





Again the instrument parameters are available to the plc as any other I/O.

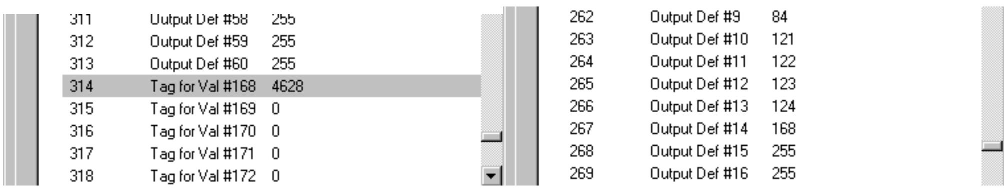
File: 2700example2.dnt (Browse only – do not try to download).

3.3 EXAMPLE 3

Example 2 used User defined parameters in the INPUT table.

To add User defined parameters to the OUTPUT table the Tag addresses would be entered starting in item 314 as 'Tag for Val #168'. This #168 is entered in the OUTPUT table in the first available slot starting at item 267 'Output Def #14' .

For example an analogue out put Module in slot 4 has a PV with the Tag address 4628. So 'Tag for Val#168' takes the value the Tag address 4628



This is 'Tag for Val #168' so the output assembly table Output Def #14 takes the value 168.

The Scanner has now also to be configured to have two more bytes in the output table making it 28.

File: 2700example3.dnt (Browse only – do not try to download).

3.4 EXAMPLE 4

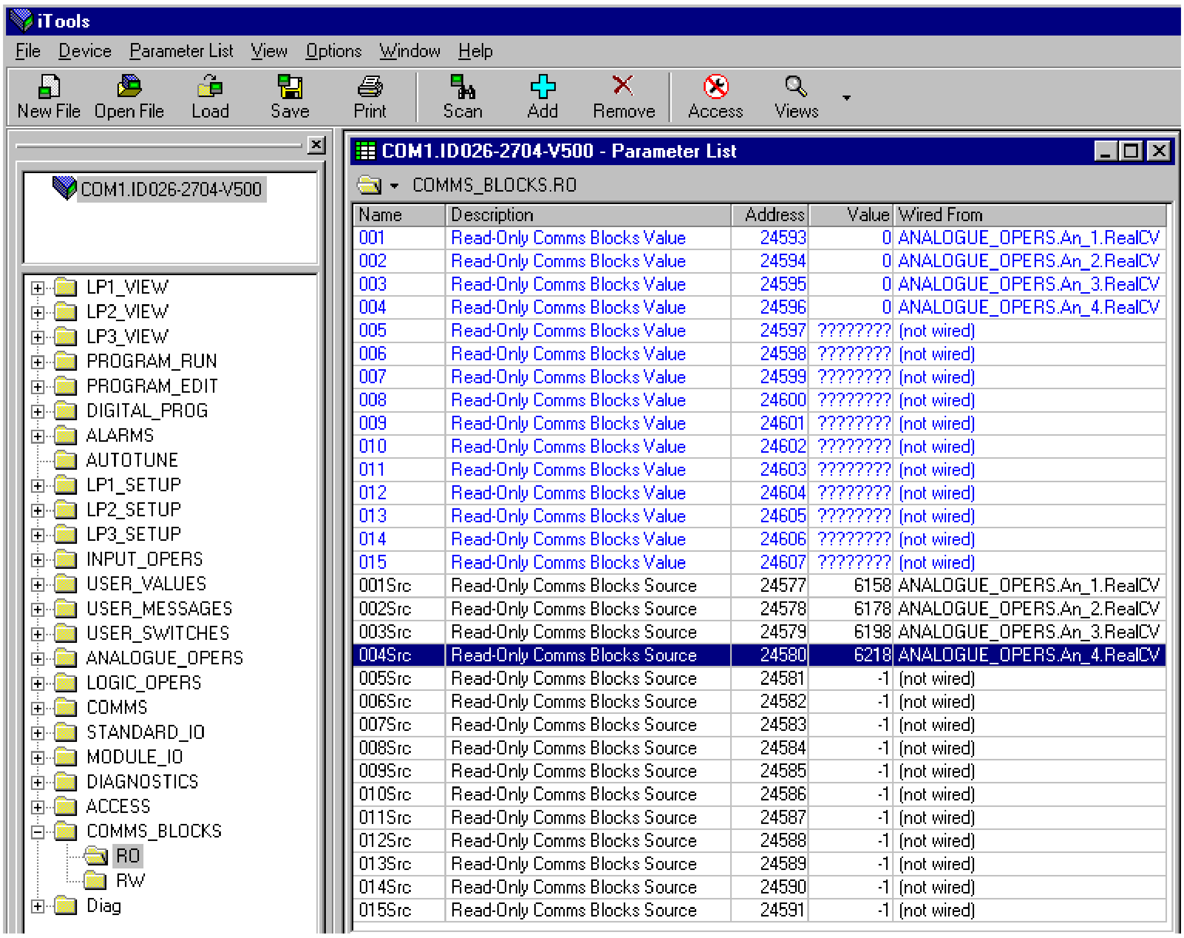
DeviceNet Configuration via the 2700 controller.

The indirection table for the 2600/2700 user defined parameters is available in the controller itself. The normal way to access this is via iTools which does not work on DeviceNet. To use iTools either use the J port or the H communications module in he instrument would have to be changed to a standard RS232 or 485 module and the instrument configuration changed to Modbus.

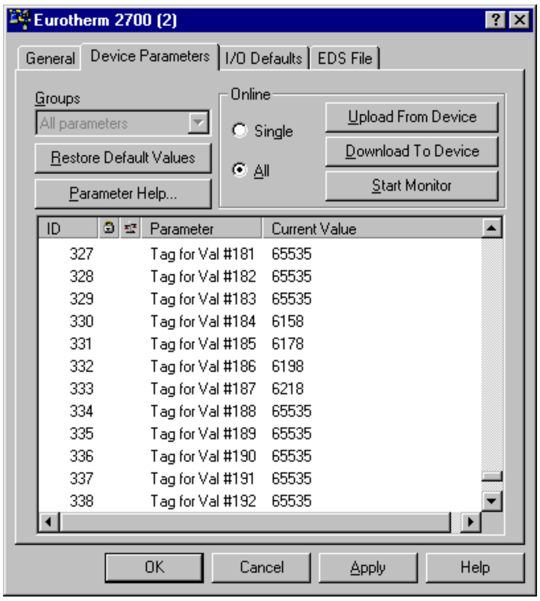
If the table has been set up in an instrument and then it is converted to DeviceNet the indirection table in the instrument will be copied automatically to the DeviceNet Module.

However if there is no indirection table configured, the DeviceNet module will automatically take the default configuration shown in Example 1.

In the example below a Modbus instrument was set up on iTools using the READ ONLY comms blocks. As can be seen the outputs from the first 4 analogue calculation blocks have been added.



Name	Description	Address	Value	Wired From
001	Read-Only Comms Blocks Value	24593	0	ANALOGUE_OPERS.An_1.RealCV
002	Read-Only Comms Blocks Value	24594	0	ANALOGUE_OPERS.An_2.RealCV
003	Read-Only Comms Blocks Value	24595	0	ANALOGUE_OPERS.An_3.RealCV
004	Read-Only Comms Blocks Value	24596	0	ANALOGUE_OPERS.An_4.RealCV
005	Read-Only Comms Blocks Value	24597	????????	(not wired)
006	Read-Only Comms Blocks Value	24598	????????	(not wired)
007	Read-Only Comms Blocks Value	24599	????????	(not wired)
008	Read-Only Comms Blocks Value	24600	????????	(not wired)
009	Read-Only Comms Blocks Value	24601	????????	(not wired)
010	Read-Only Comms Blocks Value	24602	????????	(not wired)
011	Read-Only Comms Blocks Value	24603	????????	(not wired)
012	Read-Only Comms Blocks Value	24604	????????	(not wired)
013	Read-Only Comms Blocks Value	24605	????????	(not wired)
014	Read-Only Comms Blocks Value	24606	????????	(not wired)
015	Read-Only Comms Blocks Value	24607	????????	(not wired)
001Src	Read-Only Comms Blocks Source	24577	6158	ANALOGUE_OPERS.An_1.RealCV
002Src	Read-Only Comms Blocks Source	24578	6178	ANALOGUE_OPERS.An_2.RealCV
003Src	Read-Only Comms Blocks Source	24579	6198	ANALOGUE_OPERS.An_3.RealCV
004Src	Read-Only Comms Blocks Source	24580	6218	ANALOGUE_OPERS.An_4.RealCV
005Src	Read-Only Comms Blocks Source	24581	-1	(not wired)
006Src	Read-Only Comms Blocks Source	24582	-1	(not wired)
007Src	Read-Only Comms Blocks Source	24583	-1	(not wired)
008Src	Read-Only Comms Blocks Source	24584	-1	(not wired)
009Src	Read-Only Comms Blocks Source	24585	-1	(not wired)
010Src	Read-Only Comms Blocks Source	24586	-1	(not wired)
011Src	Read-Only Comms Blocks Source	24587	-1	(not wired)
012Src	Read-Only Comms Blocks Source	24588	-1	(not wired)
013Src	Read-Only Comms Blocks Source	24589	-1	(not wired)
014Src	Read-Only Comms Blocks Source	24590	-1	(not wired)
015Src	Read-Only Comms Blocks Source	24591	-1	(not wired)



Looking at this back on DeviceNet the properties of the Read only parameters show the same parameters, addresses 6158, 6178 etc.
It is still necessary to add these (#184 to #188) to the Input assembly table via the network.

Similarly output parameters can be added to the indirection table.

3.5 2600 & 2700 CLASS, INSTANCE, ATTRIBUTE ID TABLES

EDS List	Quantity	Description
0 to 160	161	Predefined parameters 2700 Data #0 to #160 Class 100 Instance 1 Attributes 0 to 160
168 to 183	16	User defined OUTPUT parameters 2700 Data #168 to #183 **Class 100 Instance 1 Attributes 168 to 183**
184 to 199	16	User defined INPUT parameters 2700 Data #184 to #199 **Class 100 Instance 1 Attributes 184 to 199**
194 to 253	60	Enter #<number> of required INPUT parameters **Class 102 Instance 1 Attributes 1 to 60**
254 to 313	60	Enter #<number> of required OUTPUT parameters **Class 102 Instance 2 Attributes 1 to 60**
314 to 330	16	Enter Tag Address of user defined INPUT parameters **Class 103 Instance 2 Attributes 1 to 16**
330 to 346	16	Enter Tag Address of user defined OUTPUT parameters **Class 103 Instance 1 Attributes 1 to 16**
346 to 351	Specialist Parameters – explicit messaging **Class 101 Instance 1 Attributes 1 to 6**	

Attribute ID #<number>	2600/2700 Variable	Tag Address
0	Loop1 PV	1
1	Loop1TargetSP	2
2	Loop1 Manual Output	3
3	Loop1 Working OP	4
4	Loop1 Working SP	5
5	Loop1 SP select	15
6	Loop1 SP1	24
7	Loop1 SP2	25
8	Loop1 Valve Position	53
9	Loop1 Active Set	72
10	Loop1 Loop Status	76
11	Loop1 Feedforward Trim Limit	99
12	Loop1 Cascade Disable	131
13	Loop1 Ratio Enable	151
14	Loop1 Lead PV	155
15	Loop1 Ratio SP	156
16	Loop1 Ratio Trim	157
17	Loop1 Override Disable	160
18	Loop1 Manual Mode	273
19	Loop1 Alarm Status Word	336
20	Loop1 PropBand1	351
21	Loop1 IntegralTime1	352
22	Loop1 DerivativeTime1	353
23	Loop1 RelCoolGain1	354
24	Loop1 ManReset1	355
25	Loop1 CutbackHigh1	356
26	Loop1 CutbackLow1	357
27	Loop1 Remote SP	485

Attribute ID #<number>	2600/2700 Variable	Tag Address
28	Loop1 Remote SP enable	633
29	Loop1 Aux PV	769
30	Loop1 Aux WSP (override sp)	773
31	Loop1 Aux LSP (cascade slave lsp)	792
32	Loop1 Override SP	831

Attribute ID #<number >	2600/2700 Variable	Tag Address
33	Loop2 PV	1025
34	Loop2 Target SP	1026
35	Loop2 Man Output	1027
36	Loop2 Working OP	1028
37	Loop2 Working SP	1029
38	Loop2 SP select	1039
39	Loop2 SP1	1048
40	Loop2 SP2	1049
41	Loop2 Valve Position	1077
42	Loop2 Active Set	1096
43	Loop2 Loop Status	1100
44	Loop2 Feedforward Trim Limit	1123
45	Loop2 Cascade Disable	1155
46	Loop2 Ratio Enable	1175
47	Loop2 Lead PV	1179
48	Loop2 Ratio SP	1180
49	Loop2 Ratio Trim	1181
50	Loop2 Override Disable	1184
51	Loop2 Manual Mode	1297

Attribute ID #<number>	**2600/2700 Variable**	**Tag Address**
52	Loop2 Alarm Status Word	1360
53	Loop2 PropBand1	1375
54	Loop2 IntegralTime1	1376
55	Loop2 DerivativeTime1	1377
56	Loop2 RelCoolGain1	1378
57	Loop2 ManReset1	1379
58	Loop2 CutbackHigh1	1380
59	Loop2 CutbackLow1	1381
60	Loop2 Remote SP	1509
61	Loop2 Remote SP enable	1657
62	Loop2 Aux PV	1793
63	Loop2 Aux WSP (override sp)	1797
64	Loop2 Aux LSP (cascade slave lsp)	1816
65	Loop2 Override SP	1855

Attribute ID #<number>	2600/2700 Variable	Tag Address
66	Loop3 PV	2049
67	Loop3 Target SP	2050
68	Loop3 Manual Output	2051
69	Loop3 Working OP	2052
70	Loop3 Working SP	2053
71	Loop3 SP select	2063
72	Loop3 SP1	2072
73	Loop3 SP2	2073
74	Loop3 Valve Position	2101
75	Loop3 Active Set	2120
76	Loop3 L Status	2124
77	Loop3 Feedforward Trim Limit	2147
78	Loop3 Cascade Disable	2179
79	Loop3 Ratio Enable	2199
80	Loop3 Lead PV	2203
81	Loop3 Ratio SP	2204
82	Loop3 Ratio Trim	2205
83	Loop3 Override Disable	2208
84	Loop3 Manual Mode	2321
85	Loop3 Alarm Status Word	2384
86	Loop3 PropBand1	2399
87	Loop3 IntegralTime1	2400
88	Loop3 DerivativeTime1	2401
89	Loop3 RelCoolGain1	2402
90	Loop3 ManReset1	2403
91	Loop3 CutbackHigh1	2404
92	Loop3 CutbackLow1	2405
93	Loop3 Remote SP	2533
94	Loop3 Remote SP enable	2681
95	Loop3 Aux PV	2817
96	Loop3 Aux WSP (override sp)	2821
97	Loop3 Aux LSP (cascade slave lsp)	2840
98	Loop3 Override SP	2879

Attribute ID #<number>	**2600/2700 Variable**	**Tag Address**
99	Tune Loop	3072
100	Tune PID	3073
101	TUNE Tune State	3074
102	Autotune	3075

Attribute ID #<number>	2600/2700 Variable	Tag Address
103	Programmer WSP1	5800
104	Programmer WSP2	5801
105	Programmer WSP3	5802
106	Prog. Segm. Time Remaining	5813
107	Prog. Run Program Logic	5817
108	Programmer Run Program No	5820
109	Programmer Run Segment No	5822
110	Programmer PSP1 Run Target	5829
111	Programmer PSP2 Run Target	5830
112	Programmer PSP3 Run Target	5831
113	Programmer PSP1 Run Rate	5832
114	Programmer PSP2 Run Rate	5833
115	Programmer PSP3 Run Rate	5834
116	Programmer Dwell Time1	5841
117	Programmer Dwell Time2	5842
118	Programmer Dwell Time3	5843
119	Programmer Prog Run	5893
120	Programmer Prog Hold	5894

Attribute ID #<number>	**2600/2700 Variable**	**Tag Address**
121	User VAL 1	9220
122	User VAL2	9225
123	User VAL3	9230
124	User VAL4	9235
125	User VAL5	9240
126	User VAL6	9245
127	User VAL7	9250
128	User VAL8	9255
129	User VAL9	9260
130	User VAL10	9265
131	User VAL11	9270
132	User VAL12	9275

Attribute ID #<number>	**2600/2700 Variable**	**Tag Address**
133	Loop1 Alarm 1 Setpoint	11586
134	Loop1 Alarm 1 Ack	11594
135	Loop1 Alarm 2 Setpoint	11596
136	Loop1 Alarm 2 Ack	11604
137	Loop2 Alarm 1 Setpoint	11634
138	Loop2 Alarm 1 Ack	11642
139	Loop2 Alarm 2 Setpoint	11644
140	Loop2 Alarm 2 Ack	11652
141	Loop3 Alarm 1 Setpoint	11682
142	Loop3 Alarm 1 Ack	11690
143	Loop3 Alarm 2 Setpoint	11692
144	Loop3 Alarm 2 Ack	11700

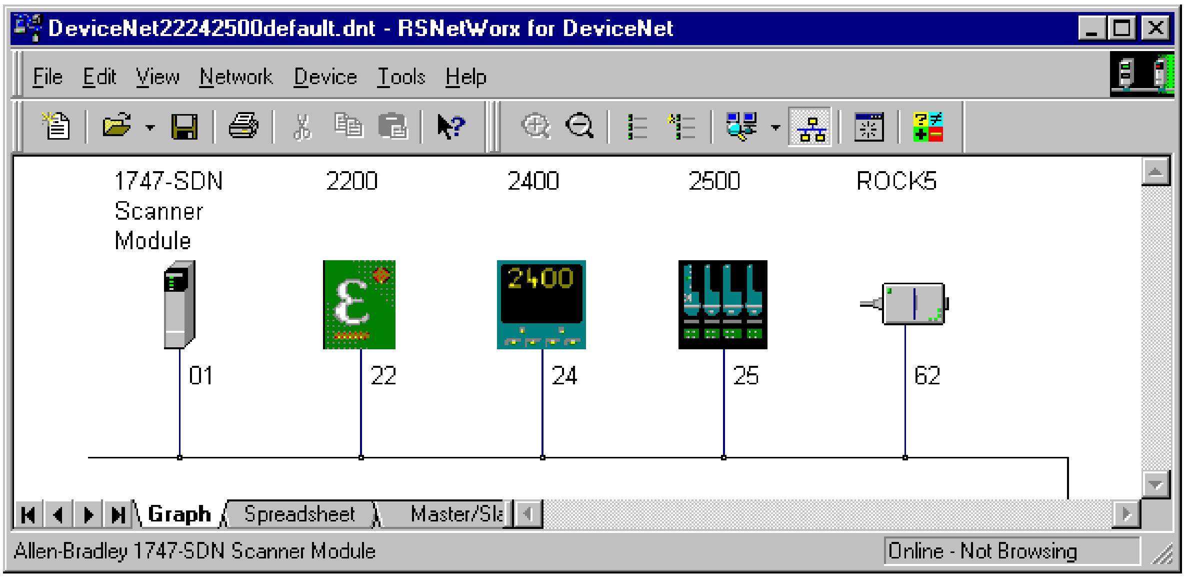
Attribute ID #<number>	**2600/2700 Variable**	**Tag Address**
145	User Analog Alarm 1 SP	11730
146	User Analog Alarm 1 Ack	11738
147	User Analog Alarm 2 SP	11746
148	User Analog Alarm 2 Ack	11754
149	User Analog Alarm 3 SP	11762
150	User Analog Alarm 3 Ack	11770

Attribute ID #<number>	2600/2700 Variable	Tag Address
151	User Analog Alarm 4 SP	11778
152	User Analog Alarm 4 Ack	11786
153	User Analog Alarm 5 SP	11794
154	User Analog Alarm 5 Ack	11802
155	User Analog Alarm 6 SP	11810
156	User Analog Alarm 6 Ack	11818
157	User Analog Alarm 7 SP	11826
158	User Analog Alarm 7 Ack	11834
159	User Analog Alarm 8 SP	11842
160	User Analog Alarm 8 Ack	11850
161-167	Reserved (not used)	

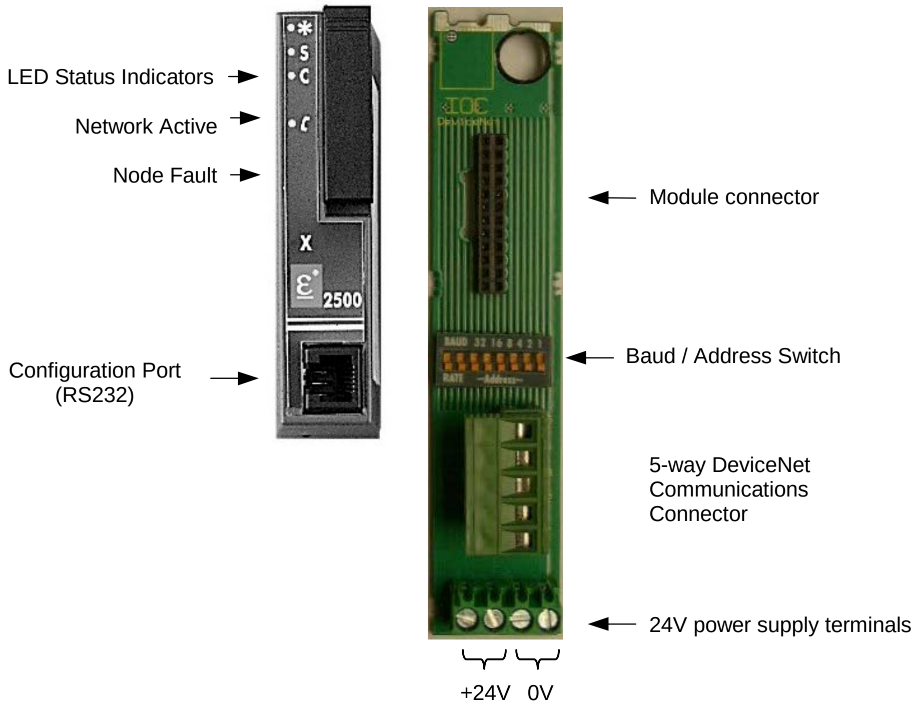
Attribute ID #<number>	2600/2700 Variable	Tag Address
168-183	Indirect OUTPUT parameters (configurable)	User
184-199	Indirect INPUT parameters (configurable)	defined

4.Transferring Data – 2500

A network has been set up with a 2500 base at address 25, a 2400 at address 24 and a 2200 at address 22. The baud rate has been set at 500k.



The 2500 has to be set up using the DIL switches on the IOC terminal base
The DeviceNet IOC is identified by the front label and the order code printed on the side label. This IOC must be used with the DeviceNet Terminal Unit.



Address

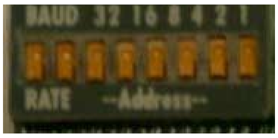
Baud 32 16 8 4 2 1

Baud

125k ↓↓

250k ↓↑

500k ↑↓



↑ = On

The 2500 uses the standard DeviceNet 5 way connector marked on the terminal unit.

2500 CAN Label	Colour
V+	Red
CAN_H	White
SHIELD	None
CAN_L	Blue
V-	Black

Once correctly connected to the network the yellow 'Network Active' LED will flash. This shows that the module is being scanned but does not have a master. The red Node Fault LED will be on.

The DeviceNet parameter mapping in the 2500 is done solely by means of user defined parameters. There is no fixed list of pre-defined parameters.

The 2500 device parameters are divided into 7 sections

the values of the user defined parameters in the OUTPUT table
the values of the user defined parameters in the INPUT table
the actual INPUT table of parameters to be READ by the DeviceNet client
the actual OUTPUT table of parameters to be WRITTEN by the client
the Tag addresses of the user defined OUTPUT parameters
the Tag addresses of the user defined INPUT parameters
a group that can be used to control block parameter read / write

EDS list	Quantity	Description
1 to 100	100	User defined OUTPUT parameter values 2500 Data #0 to #99
101 to 200	100	User defined INPUT parameter values 2500 Data #100 to #199
201 to 260	60	Enter #number of required INPUT parameters
261 to 320	60	Enter #number of required OUTPUT parameters
321 to 420	100	Enter Tag Address of user defined OUTPUT parameters
421 to 520	100	Enter Tag Address of user defined INPUT parameters
521 to 526	Specialist Parameters – block read or write by explicit messaging	

Output Parameters

The procedure to be followed is

enter Tag address of required parameters in list from 'Tag for Val#0'.
Unused slots should be set to 65535.
enter the #number of these parameters (values 1 to 100) starting at 'Output Def #1'.
Unused slots must be set to 255.
Note the number of input bytes that have been used.

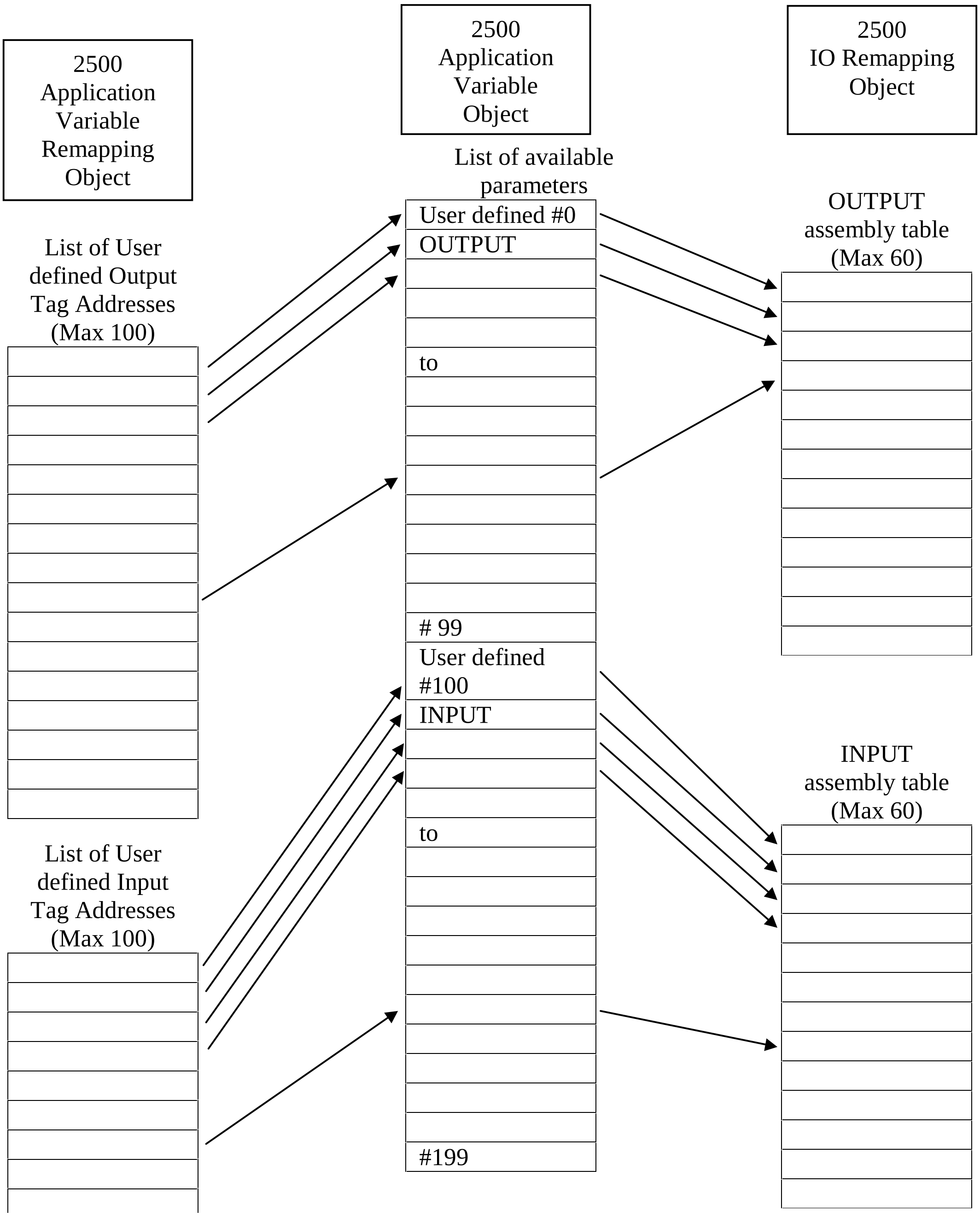
The values of these parameters will be found on-line in 2500 Data#0 to 99.

Input Parameters

The procedure to be followed is

enter Tag address of required parameters in list from 'Tag for Val#100'.
Unused slots should be set to 65535.
enter the #number of these parameters (values 100 to 199) starting at 'Input Def #1'.
Unused slots must be set to 255.
Note the number of input bytes that have been used.

The values of these parameters will be found on-line in 2500 Data#100 to 199.



This is further illustrated by 3 examples.

Example 1 is as the controller is delivered new.
Example 2 uses other user defined parameters.
Example 3 shows how the 2500 can be configured beforehand using Eurotherm's iTools and how this configuration can be uploaded to be used by the Scanner. However the Scanner still has to be set up to match.

4.1 DEFAULT EXAMPLE 1

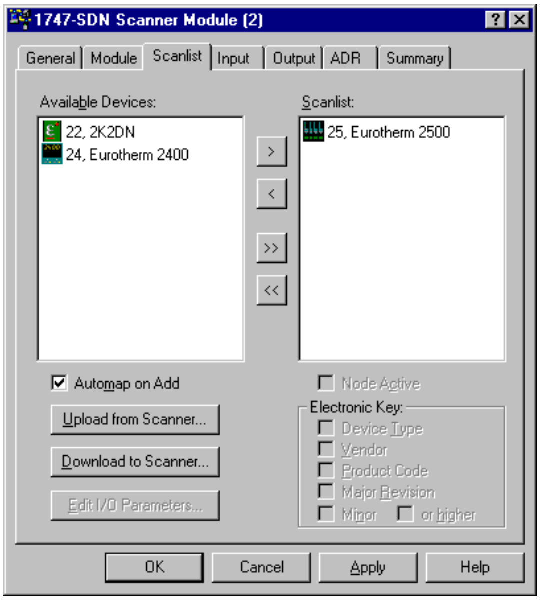
A new 2500 DeviceNet controller comes with a default parameter set as laid out below. This provides the most commonly required parameters for the 8 loops amounting to 24 input parameters and 24 output parameters.

	Input Parameter	Tag Address
1	Process Variable (Loop 1)	1
2	Working Setpoint (Loop 1)	5
3	Alarm Status (Loop 1)	16085
4	Process Variable (Loop 2)	513
5	Working Setpoint (Loop 2)	517
6	Alarm Status (Loop 2)	16086
7	Process Variable (Loop 3)	1025
8	Working Setpoint (Loop 3)	1029
9	Alarm Status (Loop 3)	16087
10	Process Variable (Loop 4)	1537
11	Working Setpoint (Loop 4)	1541
12	Alarm Status (Loop 4)	16088
13	Process Variable (Loop 5)	2049
14	Working Setpoint (Loop 5)	2053
15	Alarm Status (Loop 5)	16089
16	Process Variable (Loop 6)	2561
17	Working Setpoint (Loop 6)	2565
18	Alarm Status (Loop 6)	16090
19	Process Variable (Loop 7)	3073
20	Working Setpoint (Loop 7)	3077
21	Alarm Status (Loop 7)	16091
22	Process Variable (Loop 8)	3585
23	Working Setpoint (Loop 8)	3589
24	Alarm Status (Loop 8)	16092
TOTAL LENGTH = 24 words = 48 bytes		

	Output Parameter	Tag Address
1	Target Setpoint (Loop 1)	2
2	Auto/Manual Select (Loop 1)	152
3	Alarm Group Ack (Loop 1)	13344
4	Target Setpoint (Loop 2)	514
5	Auto/Manual Select (Loop 2)	664
6	Alarm Group Ack (Loop 2)	13384
7	Target Setpoint (Loop 3)	1026
8	Auto/Manual Select (Loop 3)	1176
9	Alarm Group Ack (Loop 3)	13424
10	Target Setpoint (Loop 4)	1538
11	Auto/Manual Select (Loop 4)	1688
12	Alarm Group Ack (Loop 4)	13464
13	Target Setpoint (Loop 5)	2050
14	Auto/Manual Select (Loop 5)	2200
15	Alarm Group Ack (Loop 5)	16160
16	Target Setpoint (Loop 6)	2562
17	Auto/Manual Select (Loop 6)	2712
18	Alarm Group Ack (Loop 6)	16200

Using RSNetWorx , Right-Clicking on the 2500 and select 'Properties' and the Device I/O Parameter List tab. The information reflects the tables above.
Tag for Val #0 to #23 will be the output Tag addresses (remainder 65535)
Output Def 1 to 24 will have the value 1 to 23 (remainder 255)
Tag for Val #100 to #123 will be the input Tag addresses (remainder 65535)
Input Def 1 to 24 will have the value 100 to 123 (remainder 255)

The default table is 24 INPUT parameters and 24 OUTPUT parameters which is 48 bytes each. For the default example nothing has to be changed.



Now we need to set up the Scanner to read and write these parameters.

'General' Tab – information only
'Module' Tab – set the Scanner module slot correctly (6 in this example)

'Scanlist' Tab – add the Eurotherm 2500 to the scan list (shown)
Edit I/O parameters – leave set to Polled 48 Input and 48 output

'Input' Tab – Map the 24 input parameters to the M file M1:6.0 to M1:6.23

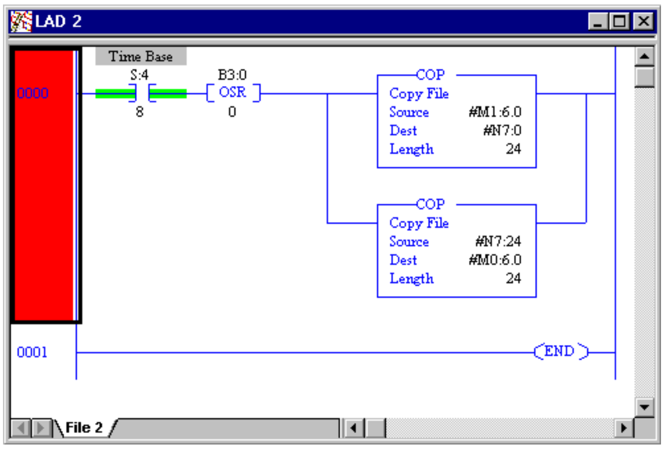
'Output' Tab – Map the 24 output parameters to the M file M0:6.0 to M1:6.23

Click apply and download this to the Scanner (PLC must be in prog mode).

Once the network is restarted the 2500 yellow 'Network Active' LED will change from flashing to steady and the red 'Node Fault' LED will go off.

Similarly the Scanner error indicator will show no error on node 25.

The 24 INPUT and 24 OUTPUT parameters are now being transferred back and forth on the network and using COP the data can be transferred periodically between the plc and the M files.



This simple ladder uses COP to transfer the data to and from the M files.

Note that in the interests of minimising resources it would be advantageous to co-ordinate the 2500 refresh rate – typically 220 or 330 mS, the DeviceNet background poll rate and this file transfer interval.

RSNetWorx file is DeviceNet22242500default.dnt. (Browse only – do not try to download).

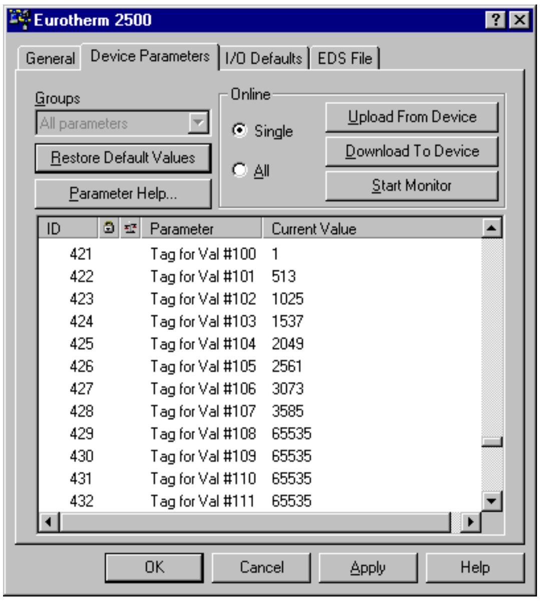
4.2 EXAMPLE 2

The default table as supplied by Eurotherm in a new module will not suit every application. In this example we will reduce the PLC INPUT parameters to just the 8 loop process variables and the PLC OUTPUT parameters to the 8 loop target setpoints and the 4 setpoints for the User Analogue Alarms blocks.

Number	Input Parameter	Tag Address
1	Process Variable (Loop 1)	1
2	Process Variable (Loop 2)	513
3	Process Variable (Loop 3)	1025
4	Process Variable (Loop 4)	1537
5	Process Variable (Loop 5)	2049
6	Process Variable (Loop 6)	2561
7	Process Variable (Loop 7)	3073
8	Process Variable (Loop 8)	3585
TOTAL LENGTH = 8 words = 16 bytes		

To set this up we go to the 2500 properties and enter the above Tag Address values. Any unused parameters may be set to 65535.

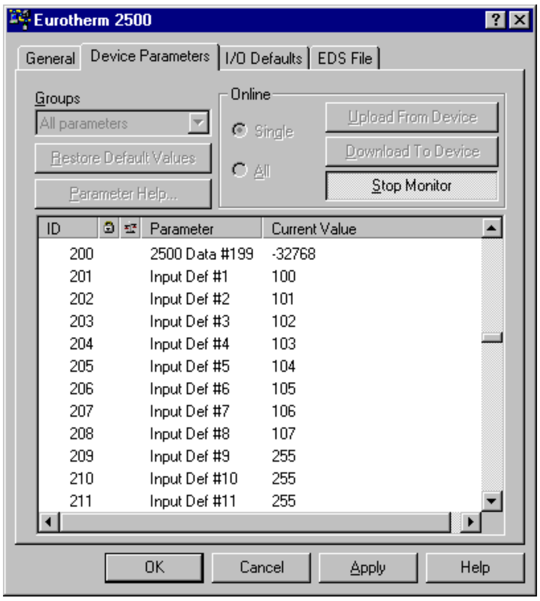
EDS list	Quantity	Description
1 to 100	100	User defined OUTPUT parameter values #0 to #99
101 to 200	100	User defined INPUT parameter values #100 to #199
201 to 260	60	Enter #<number> of required INPUT parameters
261 to 320	60	Enter #<number> of required OUTPUT parameters
321 to 420	100	Enter Tag Address of user defined OUTPUT parameters
421 to 520	**100**	**Enter Tag Address of user defined INPUT parameters**
521 to 526	Specialist Parameters – block read or write	



The Tag addresses of the required parameters have been entered against the Tags for Val #100 to 107.

Unused parameters may be set to the value 65535.

EDS list	Quantity	Description
1 to 100	100	User defined OUTPUT parameter values #0 to #99
101 to 200	100	User defined INPUT parameter values #100 to #199
201 to 260	60	Enter #<number> of required INPUT parameters
261 to 320	60	Enter #<number> of required OUTPUT parameters
321 to 420	100	Enter Tag Address of user defined OUTPUT parameters
421 to 520	100	Enter Tag Address of user defined INPUT parameters
521 to 526	Specialist Parameters – block read or write	



The value 100 to 107 are now entered into Input Def #1 to #8

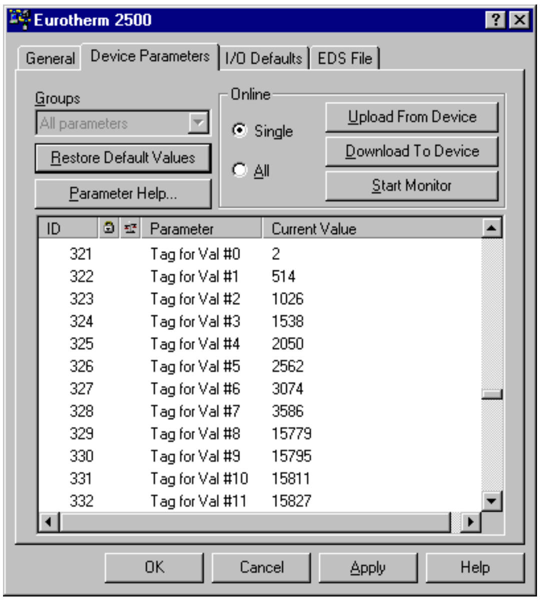
Unused parameters **must** be set to the value 255.

Now for the OUTPUT parameters.

Number	Output Parameter	Tag address
1	Target Setpoint (Loop 1)	2
2	Target Setpoint (Loop 2)	514
3	Target Setpoint (Loop 3)	1026
4	Target Setpoint (Loop 4)	1538
5	Target Setpoint (Loop 5)	2050
6	Target Setpoint (Loop 6)	2562
7	Target Setpoint (Loop 7)	3074
8	Target Setpoint (Loop 8)	3586
9	User Analogue Alarm SP1	15779
10	User Analogue Alarm SP2	15795
11	User Analogue Alarm SP3	15811
12	User Analogue Alarm SP4	15827
TOTAL LENGTH = 12 words = 24 bytes		

To set this up we go to the 2500 properties and enter the above Tag values. Any unused parameters may be set to 65535.

EDS List	Quantity	Description
1 to 100	100	User defined OUTPUT parameter values #0 to #99
101 to 200	100	User defined INPUT parameter values #100 to #199
201 to 260	60	Enter #<number> of required INPUT parameters
261 to 320	60	Enter #<number> of required OUTPUT parameters
321 to 420	100	Enter Tag Address of user defined OUTPUT parameters
421 to 520	100	Enter Tag Address of user defined INPUT parameters
521 to 526	Specialist Parameters – block read or write	

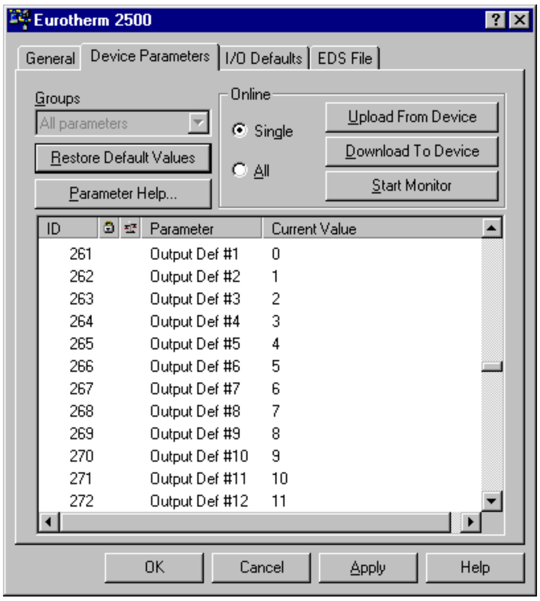


The Tag addresses of the required parameters have been entered against the Tags for Val #0 to #11.

Unused parameters may be set to the value 65535.

EDS List	Quantity	Description
1 to 100	100	User defined OUTPUT parameter values #0 to #99
101 to 200	100	User defined INPUT parameter values #100 to #199
201 to 260	60	Enter #<number> of required INPUT parameters
261 to 320	**60**	**Enter #<number> of required OUTPUT parameters**
321 to 420	100	Enter Tag Address of user defined OUTPUT parameters
421 to 520	100	Enter Tag Address of user defined INPUT parameters
521 to 526	Specialist Parameters – block read or write	

So these values #0 to #11 must be entered into the OUTPUT data list from Def #1 to #12.



Now download all these new settings to the 2500.

Each value can be downloaded after entry using ‘Single’ or download ‘All’.

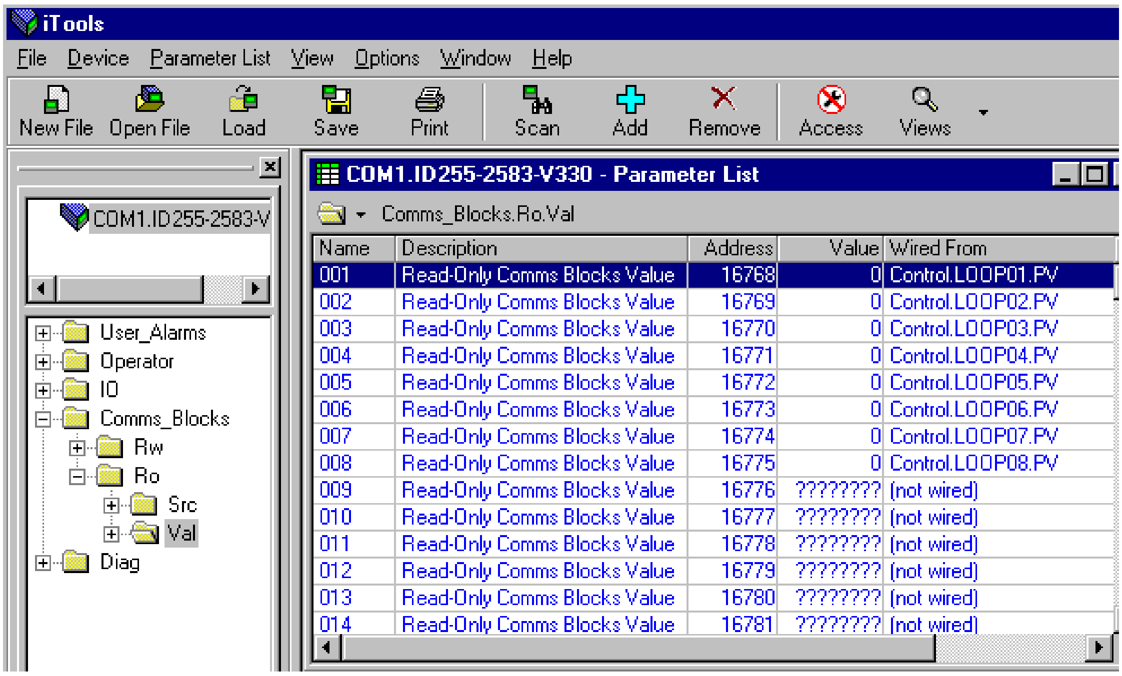
Finally return to the properties of the Scanner Module. On the scanlist select the 2500 and use ‘Edit I/O’ to modify the input byte count to 16 and the output byte count to 24. Map these parameters as required. As there are less parameters than were used in the default example it is possible to map them directly onto discrete I/O, in these example the input data is on M1:6.9 to 6.16 and the output data on M0:6.7 to 6.18

Download this new setting to the scanner and restart the network.

RSNetWorx file is DeviceNet*2500EX2default.dnt*. (Browse only – do not try to download).

4.3 EXAMPLE 3 – ITOOLS

In Example 2 the required 2500 parameters were selected by configuring the 2500 DeviceNet tables using RSNetWorx. It is also possible to first configure the 2500 directly using iTools and then UPLOAD the configuration from the 2500 into RSNetWorx.



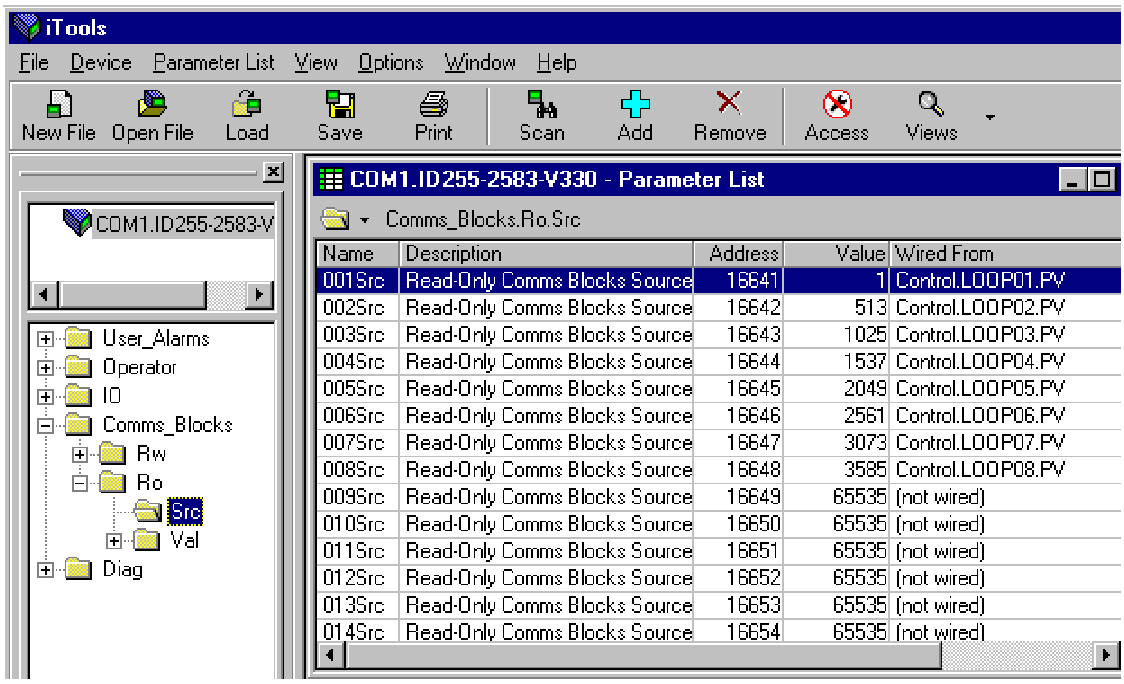
The data in the above screen shows the configuration from Example 2.

The INPUT parameters are in **Comms_Blocks.Ro**. The Val directory shown above gives the current value of the selected parameters and where the data comes from. The Tag addresses are in the **Src** directory.

The OUTPUT parameters are in **Comms_Blocks.Rw**. The Val directory gives the current value of the selected parameters and where the data goes to. The Tag addresses are in the **Src** directory.

To configure the 2500 it must be connected to iTools using the configuration cable into the RJ11 configuration port on the IOC itself. Plugging in this cable will disconnect the 2500 slave from the DeviceNet network.

Now enter the required Tag Addresses into the Comms_Blocks.Ro.Src and Comms_Blocks.Rw.Src directories.



In the **Src** directory the tag Value is the Tag address of the required user defined parameter.

Now remove the configuration cable. Using RSNetWorx UPLOAD the 2500 Device Parameters and, if necessary (i.e. the input or output byte count has changed, the mapping needs to be changed etc) modify the Scanner 2500 scanlist I/O parameters to suit.

Restart the DeviceNet network.

4.4 2500 CLASS, INSTANCE, ATTRIBUTE ID TABLE

EDS list	Quantity	Description
1 to 100	100	User defined OUTPUT parameter values 2500 Data #0 to #99 **Class 100 Instance 1 Attributes 0 to 99**
101 to 200	100	User defined INPUT parameter values 2500 Data #100 to #199 **Class 100 Instance 1 Attributes 100 to 199**
201 to 260	60	Enter #number of required INPUT parameters **Class 102 Instance 1 Attributes 1 to 60**
261 to 320	60	Enter #number of required OUTPUT parameters **Class 102 Instance 2 Attributes 1 to 60**
321 to 420	100	Enter Tag Address of user defined OUTPUT parameters **Class 103 Instance 1 Attributes 1 to 100**
421 to 520	100	Enter Tag Address of user defined INPUT parameters **Class 103 Instance 2 Attributes 1 to 100**
521 to 526	Specialist Parameters – block read or write **Class 101 Instance 1 Attributes 1 to 6**	

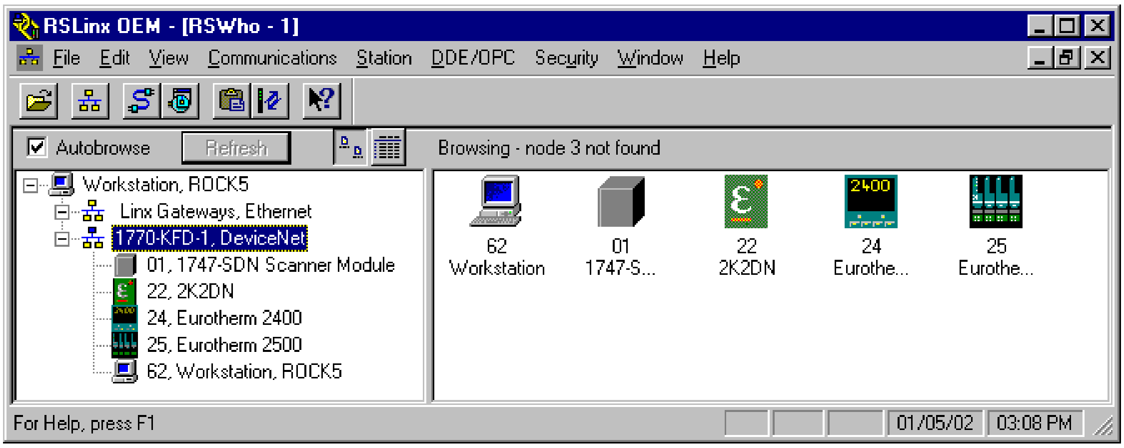
Note – all variables of type INT unless configured using indirection table to be 32 bit (described elsewhere in this document). Data formats are therefore as Tag, i.e. Scaled Integers.

Attribute ID #<number>	Variable	Tag Address
0-99	Indirect R/W parameters (configurable)	16512
100-199	Indirect RO parameters (configurable)	16768

Note that all these parameters are User defined as described in the examples above. To define the parameters their Tag Addresses are required. With the 2500 the best source of Tag Addresses is iTools.

5.Transferring Data – 2400

A network has been set up with a 2500 base at address 25, a 2400 at address 24 and a 2200 at address 22. The baud rate has been set at 500k.



The 2400 has to be set up via the user interface. The controller must have a DeviceNet communications module fitted.

In Configuration mode ***HA*** must be set as shown below

id = CmS
Func = dnEt
bAud = 500
res = FuLL

In Operating Mode, with Full Access, in the ***cmS LiSt*** the address must be set as shown below

Addr = 24

Hardware wiring is as follows:

Series 2400 Terminal	CAN Label	Colour
HA	V+	Red
HB	CAN_H	White
HC	SHIELD	None
HD	CAN_L	Blue
HE	V-	Black

Once correctly wired to the network and with the correct DeviceNet baud rate and a unique address, communications at the hardware level will be established. This is indicated at the controller by the flashing **REM** beacon.

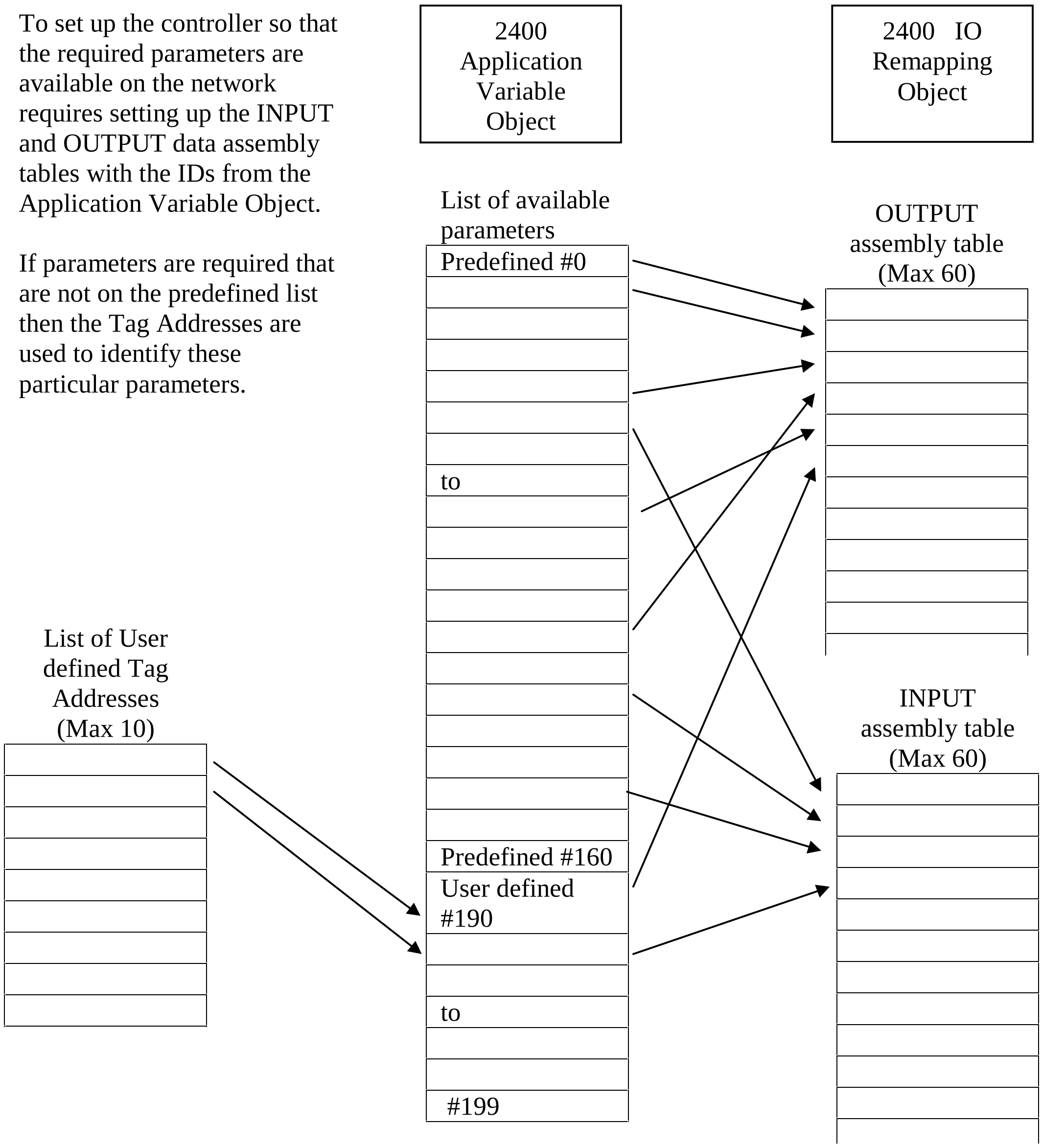
Now the 2400 and the Scanner have to be configured to transfer parameter data.

The 2400 device parameters (Full table in Section 5.5) are divided into 6 'groups'

A list of instrument parameters pre-defined and immediately available for selection on the INPUT or OUTPUT tables
a list of additional user defined parameters to add to the INPUT or OUTPUT table
the actual INPUT table of parameters to be READ by the DeviceNet client
the actual OUTPUT table of parameters to be WRITTEN by the client
the Tag address of parameters to be READ by or WRITTEN to the DeviceNet client
a group that can be used to control block parameter read / write

To set up the controller so that the required parameters are available on the network requires setting up the INPUT and OUTPUT data assembly tables with the IDs from the Application Variable Object.

If parameters are required that are not on the predefined list then the Tag Addresses are used to identify these particular parameters.



EDS list	Quantity	Description
1 to 161	161	Predefined parameters #0 to #160
162 to 171	10	User defined INPUT or OUTPUT parameters #190 to #199
172 to 231	**60**	**Enter #<number> of required INPUT parameters**
232 to 291	**60**	**Enter #<number> of required OUTPUT parameters**
292 to 301	10	Enter Tag Address of user defined INPUT or OUTPUT parameters
302 to 307	Specialist Parameters – block read or write	

To set up the controller so that the desired parameters can be read and written involves setting up the INPUT and OUTPUT tables (highlighted in the table above).

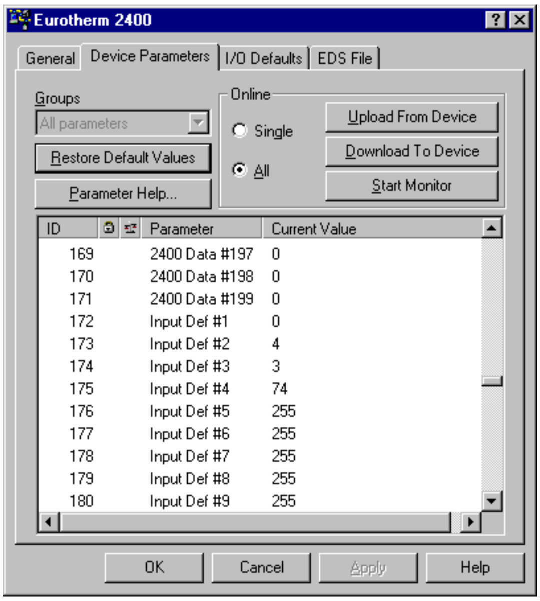
This information can be seen by inspecting the 2400.EDS file in a text editor and is the way in which the data is displayed in RSNetWorx Device Parameters.

This is best illustrated by three examples.
Example 1 is the default 2400 DeviceNet configuration.
Example 2 uses other pre-defined parameters and includes a simple plc application.
Example 3 adds user defined parameters not included on the pre-defined list.

5.1 DEFAULT EXAMPLE 1

As supplied new the 2400 comes with the following DeviceNet parameter setup. The Attribute for the parameters on the table below come from the full listing in Section 5.4.

Item	Input Parameter	Attribute
1	Process Value	0
2	Working Setpoint	4
3	Working Output Power	3
4	Summary Status	74
Total length = 4 words = 8 bytes		

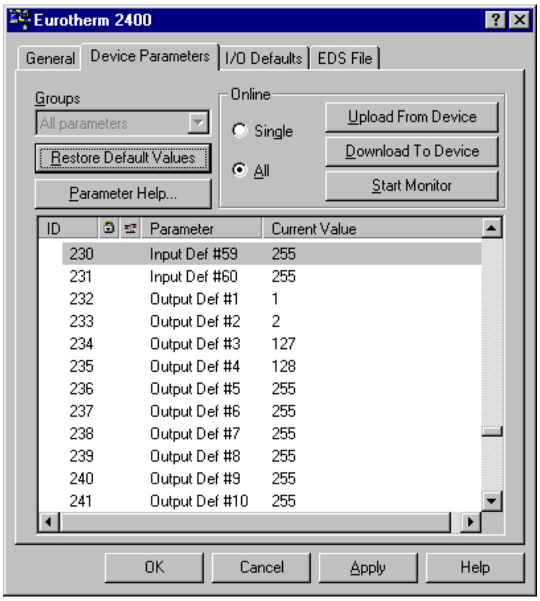


Looking at the Device Parameters via the network we will see these 4 input parameters defined with the above IDs.

Remaining input parameters must be set to 255.

Item	Output Parameter	Attribute
1	Target Setpoint	1
2	Target Output Power	2
3	Auto / Manual	127
4	Alarm Acknowledge	128
Total length = 4 words = 8 bytes		

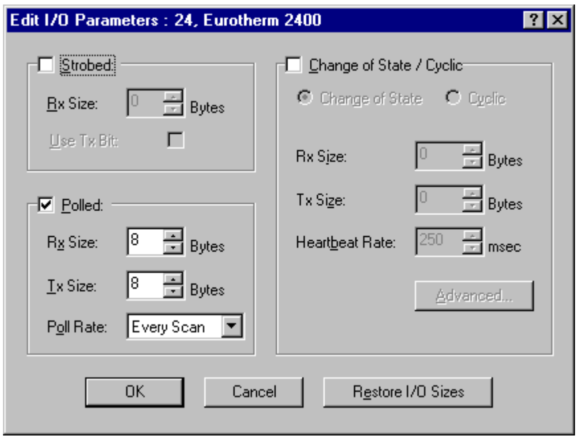
Looking at the Device Parameters via the network we will see these 4 output parameters defined with the above IDs.
Remaining input parameters must be set to 255.



Looking at the Device Parameters via the network we will see these 4 output parameters defined with the above IDs.

Remaining input parameters must be set to 255.

All that has to be configured now is the Scanner to transfer these variables.



On the Scanner properties:
'General' Tab – information only
'Module' Tab – set the Scanner module slot correctly (6 in this example)
'Scanlist' tab – add the Eurotherm 2400 to the scan list
Edit I/O parameters – leave the default settings - Polled 8 input and 8 output (shown)
'Input' Tab – Map the 4 input parameters to the M file M1:6.0 to M1:6.3
'Output' Tab – Map the 4 output parameters to the M file M0:6.0 to M1:6.3

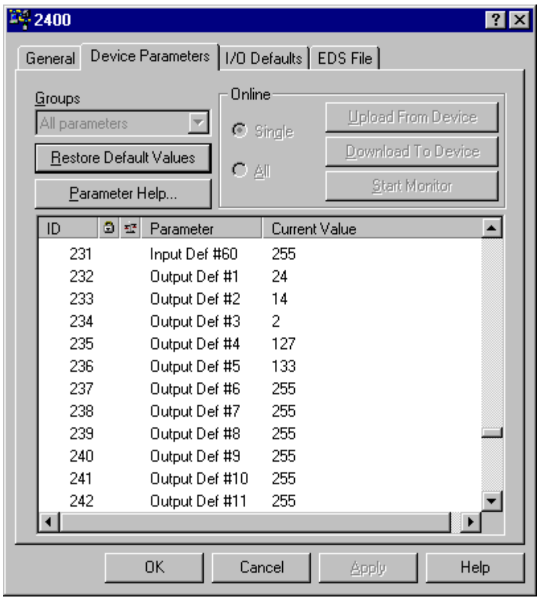
Click apply to download this to the Scanner (PLC must be in program mode). Once the network is restarted the 2400 REM will change from flashing to steady. Similarly the Scanner error indicator will show no error on node 24. The 4 INPUT and 4 OUTPUT parameters are now being transferred back and forth on the network and will be available in the plc in the I/O files.

RSNetWorx file is *DeviceNet22242500default.dnt*. (Browse only – do not try to download).

5.2 EXAMPLE 2

In this example the INPUT table stays the same as Example 1 but the output table modified and extended to include the 2 setpoints and the setpoint selector and the disable keys function.

Item	Output Parameter	Attribute
1	Setpoint 2	24
2	Setpoint Select	14
3	Target Output Power	2
4	Auto / Manual	127
5	Disable Keys	133
Total length = 5 words = 10 bytes		



On the 2400 Device Parameters change the Output parameters to the new IDs as listed in the table above.

Download them 'Singly' as entered. Do NOT download all.

All that has to be configured now is the Scanner to transfer these variables.
On the Scanner properties
'Scanlist' tab –
Edit I/O parameters – leave set to Polled 8 Input and change the output length to 12.
'Input' Tab – Map the 4 input parameters to the I file I:6.1 to I:6.4
'Output' Tab – Map the 5 output parameters to the O file O:6.1 to O:6.5

Click apply and download this to the Scanner (PLC must be in program mode). Restart the network.

The 4 INPUT and 5 OUTPUT parameters are now being transferred back and forth on the network, and will be available in the plc in the I/O files.

RSNetWorx file is *DeviceNet2400Ex2default.dnt*. (Browse only – do not try to download).

5.2.1 Simple plc application

This uses the 2400 Example 2 configuration:

Item	Output Parameter	PLC Map
1	Setpoint 2	O:6.1
2	Setpoint Select	O:6.2
3	Target Output Power	O:6.3
4	Auto / Manual	O:6.4
5	Disable Keys	O:6.5

Item	Input Parameter	PLC Map
1	Process Value	I:6.1
2	Working Setpoint	I:6.2
3	Working Output Power	I:6.3
4	Summary Status	I:6.4

The 2400 controls a furnace which is used during the day by the operators who use and change the main setpoint, SP 1. The plant is turned down to a standby setpoint SP2 overnight. Setpoint 2 During the day the operators can change the controller settings but overnight the keys are locked.

File 2400Example2.rss ladder Do not leave the keys locked when closing application!

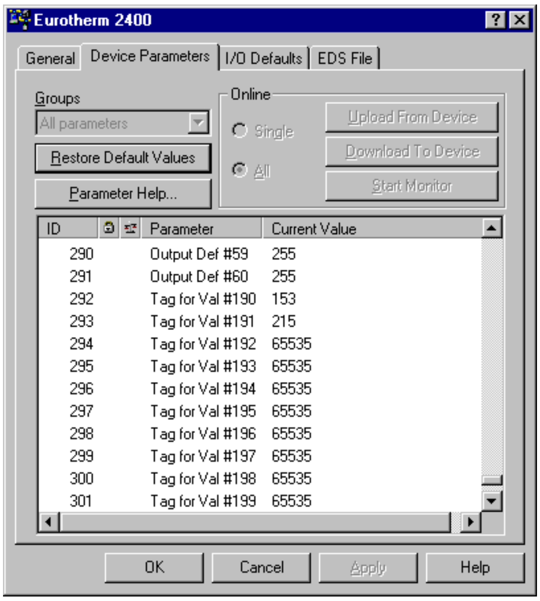
5.3 EXAMPLE 3

The list of pre-defined parameters in the 2400 is extensive but there is a means of addressing parameters which have not been included. This is done by using the Tag address of the required parameter in the 2400.

For example we shall enable the plc to write to the gain scheduler setpoint at Tag address **153** and to read the CJC temperature of the 2400 t/c input (a useful way to monitor the temperature of a remote cabinet) at Tag Address **215**. See Section 5.4.

These two tags will be added onto the end of an existing configuration.

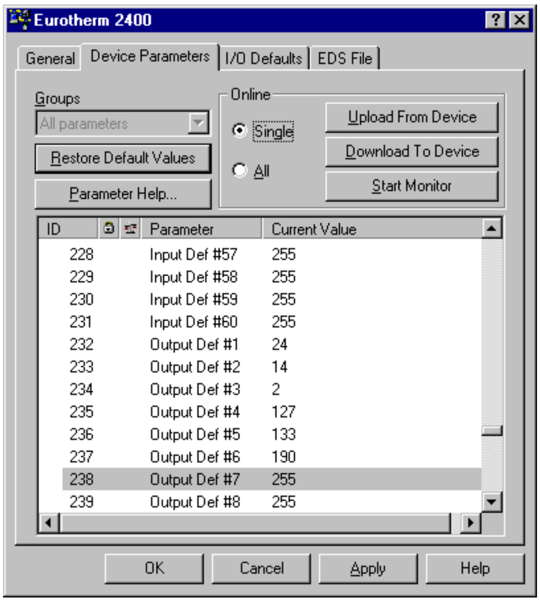
EDS list	Quantity	Description
1 to 161	161	Predefined parameters #0 to #160
162 to 171	10	User defined INPUT or OUTPUT parameters #190 to #199
172 to 231	60	Enter #<number> of required INPUT parameters
232 to 291	60	Enter #<number> of required OUTPUT parameters
292 to 301	**10**	**Enter Tag Address of user defined INPUT or OUTPUT parameters**
302 to 307	Specialist Parameters – block read or write	

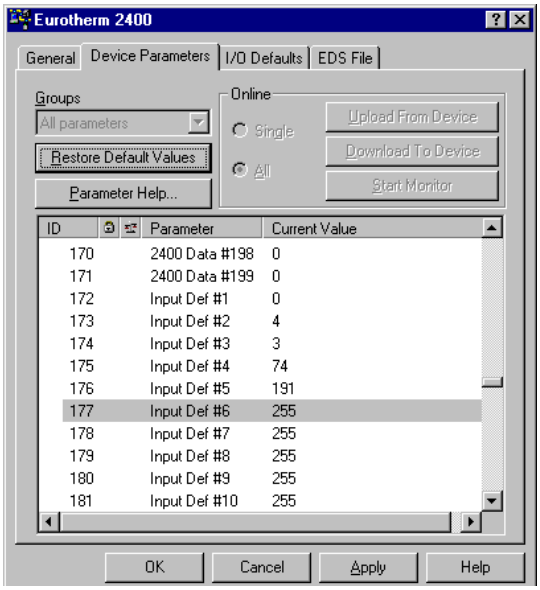


On the properties list for the 2400 we have to go to the list no 292 to enter these Tag addresses in 'Tag for Val #190' and #191.

Gain Schedule SP has Tag address 153, is an OUTPUT and now has ID #190

Input CJC has Tag address 215, is an INPUT and now has ID#191.





So on the output definition list we modify the next available parameter Output Def #7 from 255 to 190

and on the input definition list we modify the next available parameter Input Def #5 from 255 to 191

Finally we have to update the scanlist on the Scanner as the 2400 now has 10 input bytes and 14 Output bytes. These extra parameters will have to be mapped and will then be available to the plc.

5.4 2400 CLASS, INSTANCE AND ATTRIBUTE ID TABLE

EDS list	Quantity	Description
1 to 161	161	Predefined parameters #0 to #160 Class 100 Instance 1 Attributes 0 to 160
162 to 171	10	User defined INPUT or OUTPUT parameters #190 to #199 **Class 100 Instance 1 Attributes 190 to 199**
172 to 231	60	Enter #<number> of required INPUT parameters **Class 102 Instance 1 Attributes 0 to 60**
232 to 291	60	Enter #<number> of required OUTPUT parameters **Class 102 Instance 1 Attributes 0 to 60**
292 to 301	10	Enter Tag Address of user defined INPUT or OUTPUT parameters
302 to 307	Specialist Parameters – block read or write **Class 101 Instance 1 Attributes 1 to 6**	

Note – all variables of type INT - 32 bit format is not supported in this instrument type. Data formats are therefore as Tag, i.e. Scaled Integers. The scaling is based on the number of decimal point places used on the instrument display.

Tag addresses are used to identify parameters in the controller and are identical to the Modbus addresses which are also listed in the Series 2000 Communications Manual, Eurotherm Part No. HA 026230.

After the description is the Tag Address, followed where it is available, by the Attribute ID used in DeviceNet.
There are 161 parameters pre-defined for use on DeviceNet and a further 10 that can be pointed to using parameters with Attributes 190 to 199.

Attribute ID	Description
190-199	Indirect R/W parameters (configurable) – these are not managed via an indirection table, but rather within the interface itself

The tables that follow include all the main instrument parameters in the 'pages' as they appear on the instrument display.

.Controller Display	Home Tab Parameter Description	Tag	ID
	Process Variable	1	**0**
SP	Target setpoint	2	**1**
OP	% Output power For ON/OFF controllers the following power levels must be written: Cool -100% OFF 0% Heat 100%	3	**2**
w.SP	Working set point. Read only: use Target set point or currently selected set point (1 to 16) to change the value	5	**4**
m-A	Auto-man select 0: Auto 1: Manual	273	**127**
	Pot Position	317	
-	Valve Posn (computed by VP algorithm)	53	**52**
-	VP Manual Output (alterable in Man only)	60	**59**
AmPS	Heater current (With PDSIO mode 2)	80	**79**
C.id	Customer defined identification number	629	
	Setpoint Span	552	
	Error (PV-SP)	39	**38**
	Remote Input Value	26	**25**

Status Tab		Tag	ID
Summary Output Status Word		75	74
BIT	DESCRIPTION		
0	Alarm 1 State (0 = Safe 1 = Alarm)		
1	Alarm 2 State (0 = Safe 1 = Alarm)		
2	Alarm 3 State (0 = Safe 1 = Alarm)		
3	Alarm 4 State (0 = Safe 1 = Alarm)		
4	Manual Mode (0 = Auto 1 = Manual)		
5	Sensor Break (0 = Good PV 1 = Sensor Broken)		
6	Loop Break (0 = Good closed loop 1 = Open Loop)		
7	Heater Fail (0 = No Fault 1 = Load fault detected)		
8	Tune Active (0 = Auto Tune disabled 1 = Auto Tune active)		
9	Ramp/Program Complete (0 = Running/Reset 1 = Complete)		
10	PV out of range (0 = PV within table range 1 = PV out of table range)		
11	DC control module fault (0= Good. 1= BAD)		
12	Programmer Segment Synchronise (0 = Waiting, 1 = Running)		
13	Remote input sensor break (0 = Good, 1 = Bad)		
14	IP1 Fault		
15	Reserved		

Status Tab		Tag	ID
Fast Status Byte		74	**73**
BIT	DESCRIPTION		
Bit 0	Alarm 1 State (0 = Safe 1 = Alarm)		
Bit 1	Alarm 2 State (0 = Safe 1 = Alarm)		
Bit 2	Alarm 3 State (0 = Safe 1 = Alarm)		
Bit 3	Alarm 4 State (0 = Safe 1 = Alarm)		
Bit 4	Manual Mode (0 = Auto 1 = Manual)		
Bit 5	Sensor Break (0 = Good PV 1 = Sensor Broken)		
Bit 6	Loop Break (0 = Good closed loop 1 = Open Loop)		
Bit 7	Heater Fail (0 = No Fault 1 = Load fault detected)		
Control Status Word		76	**75**
BIT	DESCRIPTION		
0	Control algorithm Freeze		
1	PV input sensor broken		
2	PV out of sensor range		
3	Self Tune failed		
4	PID servo signal		
5	PID debump signal		
6	Fault detected in closed loop behaviour (loop break)		
7	Freezes the integral accumulator		
8	Indicates that a tune has completed successfully		
9	Direct/reverse acting control		
10	Algorithm Initialisation flag		
11	PID demand has been limited.		
12	Autotune enabled		
13	Adaptive tune enabled		
14	Automatic Droop compensation enabled		
15	Manual / Auto mode switch		
Instrument Status Word		77	**76**
BIT	DESCRIPTION		
0	Config/Oper mode switch		
1	Disables limit checking		
2	SRL ramp running (Read Only)		
3	Remote setpoint active		
4	Alarm acknowledge switch.		
5	Reserved		
6	Reserved		
7	Reserved		
8	Reserved		
9	Reserved		
10	Reserved		
11	Reserved		
12	Reserved		
13	Reserved		
14	Reserved		
15	Reserved		
Program Logic Status		162	
BIT	DESCRIPTION		
0	Program Output 1 (0 = OFF 1 = ON)		
1	Program Output 2 (0 = OFF 1 = ON)		
2	Program Output 3 (0 = OFF 1 = ON)		
3	Program Output 4 (0 = OFF 1 = ON)		
4	Program Output 5 (0 = OFF 1 = ON)		
5	Program Output 6 (0 = OFF 1 = ON)		
6	Program Output 7 (0 = OFF 1 = ON)		
7	Program Output 8 (0 = OFF 1 = ON)		
8	Reserved		
9	Reserved		
10	Reserved		
11	Reserved		
12	Reserved		
13	Reserved		
14	Reserved		
15	Reserved		
Digital Output Status Word		551	
BIT	DESCRIPTION		
0	H Interface module telemetry (0 = Off, 1 = On)		
1	J Interface module telemetry (0 = Off, 1 = On)		
2	1A module telemetry (0 = Off, 1 = On)		
3	LB logic telemetry (0 = Off, 1 = On)		
4	LA logic telemetry (0 = Off, 1 = On)		
5	1B module telemetry (0 = Off, 1 = On)		
6	1C module telemetry (0 = Off, 1 = On)		
7	2A module telemetry (0 = Off, 1 = On)		
8	2B module telemetry (0 = Off, 1 = On)		
9	2C module telemetry (0 = Off, 1 = On)		

10	3A module telemetry (0 = Off, 1 = On)		
11	3B module telemetry (0 = Off, 1 = On)		
12	3C module telemetry (0 = Off, 1 = On)		
13	AA relay telemetry (0 = Off, 1 = On)		
14	Reserved		
15	Reserved		
	Digital Input Status Word	87	**86**
BIT	**DESCRIPTION**		
0	H Interface module (0 = Off, 1 = On)		
1	J Interface module (0 = Off, 1 = On)		
2	1A module (0 = Off, 1 = On)		
3	LB logic input (0 = Off, 1 = On)		
4	LA logic input (0 = Off, 1 = On)		
5	1B module telemetry (0 = Off, 1 = On)		
6	1C module (0 = Off, 1 = On)		
7	2A module (0 = Off, 1 = On)		
8	2B module (0 = Off, 1 = On)		
9	2C module (0 = Off, 1 = On)		
10	3A module (0 = Off, 1 = On)		
11	3B module (0 = Off, 1 = On)		
12	3C module (0 = Off, 1 = On)		
13	Reserved		
14	Reserved		
15	Reserved		

Parameter Description	**Tag**	**ID**
SP Rate Limit Holdback Status 0: Inactive 1: Active	41	**40**
Pot Break	350	
Freeze Control Flag 0: Controlling 1: Hold	257	
SP Rate Limit Active Status 0: No setpoint rate limit 1: Setpoint rate limit active	275	**129**
Sensor Break Status Flag 0: Good 1: Sensor break	258	
Power Failed flag 0: Good 1: Power fail detected	259	
New Alarm Flag	260	
Loop Break Status Flag 0: Good 1: Loop break	263	
Integral Hold Status Flag 0: Good 1: Integral hold	264	
SRL Complete Status 0: Setpoint rate limit incomplete 1: Setpoint rate limit complete	277	**131**
Remote Input Status Flag 0: Good 1: Fault	280	
Sync Continue Flag 0: Continue 1: Awaiting synch	281	

Controller	**Run Tab**	**Tag**	**ID**
Display	**Parameter Description**		
PrG	Current program running (active prog no.)	22	**21**
StAt	Program Status 1: Reset 2: Run 4: Hold 8: Holdback 16: Complete	23	**22**
PSP	Programmer setpoint	163	
CYC	Program cycles remaining	59	**58**
SEG	Current segment number	56	**55**
StYP	Current segment type 0: End 1: Ramp (Rate) 2: Ramp (Time to target) 3: Dwell 4: Step 5: Call	29	**28**
SEGt	Segment time remaining in secs	36	**35**
	Segment time remaining in mins	63	**62**
tGt	Target setpoint (current segment)	160	
rAtE	Ramp rate	161	
PrGt	Program time remaining	58	**57**
FASt	Fast run 0: No 1: Yes	57	**56**
out.1	Logic 1 output (current program) 0: Off (applies to all 8 logic outputs) 1: On (applies to all 8 logic o/ps)	464	
out.2	Logic 2 output (current program)	465	
out.3	Logic 3 output (current program)	466	

out.4	Logic 4 output (current program)	467	
out.5	Logic 5 output (current program)	468	
out.6	Logic 6 output (current program)	469	
out.7	Logic 7 output (current program)	470	
out.8	Logic 8 output (current program)	471	
SYnc	Segment synchronisation 0: No 1: Yes	488	
SEG.d	Flash active segment in lower display	284	
	Advance Segment Flag	149	
	Skip Segment Flag	154	
	Program Logic Status	162	

Controller Display	Alarm Tab Parameter Description	Tag	ID
1---	Alarm 1setpoint value	13	**12**
2---	Alarm 2setpoint value	14	**13**
3---	Alarm 3setpoint value	81	**80**
4---	Alarm 4setpoint value	82	**81**
HY 1	Alarm 1 hysteresis	47	**46**
HY2	Alarm 2 hysteresis	68	**67**
HY3	Alarm 3 hysteresis	69	**68**
HY4	Alarm 4 hysteresis	71	**70**
Lbt	Loop break time 0: Off	83	**82**
di AG	Enable diagnostic messages 0: No Diagnostics 1: Diagnostics	282	
	Acknowledge All Alarms	274	**128**

Controller Display	Autotune Tab Parameter Description	Tag	ID
tunE	Autotune enable 0: No Tune 1: Tune	270	**124**
drA	Adaptive tune enable 0: No Adaotive Tune 1: Tune	271	**125**

drA.t	Adaptive tune trigger level	100	**99**
Adc	Automatic droop compensation (manual reset) 0: Manual reset 1: Calculated	272	**126**

Controller Display	PID Tab Parameter Description	Tag	ID
G.SP	Gain scheduler setpoint	153	
SEt	Current PID set (read only if gain scheduling is selected) 0: Set 1 1: Set 2	72	**71**
Pb	Proportional band PID1	6	**5**
ti	Integral time PID1 0: Off	8	**7**
td	Derivative time PID1 0: Off	9	**8**
rES	Manual reset PID1	28	**27**
Hcb	Cutback high PID1 0: Auto	18	**17**
Lcb	Cutback low PID1 0: Auto	17	**16**
rEL.c	Relative cool gain PID1	19	**18**
Pb2	Proportional band PID2	48	**47**
ti 2	Integral time PID2 0: Off	49	**48**
td2	Derivative time PID2 0: Off	51	**50**
rES2	Manual reset PID2	50	**49**
Hcb2	Cutback high PID2 0: Auto	118	
Lcb2	Cutback low PID2 0: Auto	117	
rEL2	Relative cool gain PID2	52	**51**
FF.Pb	Feedforward proportional band	97	**96**
FF.tr	Feedforward trim	98	**97**
FF.du	Feedforward trim limit	99	**98**

Controller Display	Motor Tab Parameter Description	Tag	ID
tm	Valve travel time	21	**20**
Int	Valve inertia time	123	**100**
bAct	Valve backlash time	124	**101**
mPt	Minimum pulse time	54	
u.br	Bounded sensor break strategy	128	**105**
Sb.oP	VP Bounded sensor break	62	**61**

Controller Display	I/O Tab Parameter Description	Tag	ID
	DC Output 1A Telemetry	12694	
	DC Output 2A Telemetry	12758	
	DC Output 3A Telemetry	12822	
	BCD Input Value	96	**95**

Controller Display	Setpoint Tab Parameter Description	Tag	ID
SSEL	Select setpoint 0: SP1 1: SP2	15	**14**
	2: SP 3 3: SP 4 4: SP 5 5: SP 6 6: SP 7 7: SP 8 8: SP 9 9: SP 10 10: SP 11 11: SP 12 12: SP13 13: SP14 14: SP15 15: SP16		
L-r	Local or remote setpoint select 0: Local 1: Remote	276	**130**
SP 1	Setpoint 1	24	**23**
SP 2	Setpoint 2	25	**24**
SP 3	Setpoint 3	164	
SP 4	Setpoint 4	165	
SP 5	Setpoint 5	166	
SP 6	Setpoint 6	167	
SP 7	Setpoint 7	168	
SP 8	Setpoint 8	169	
SP 9	Setpoint 9	170	
SP 10	Setpoint 10	171	
SP 11	Setpoint 11	172	
SP 12	Setpoint 12	173	
SP 13	Setpoint 13	174	
SP 14	Setpoint 14	175	
SP 15	Setpoint 15	176	
SP 16	Setpoint 16	177	
rm.SP	Remote setpoint	485	
rmt.t	Remote setpoint trim	486	
rAt	Ratio setpoint	61	**60**
Loc.t	Local setpoint trim	27	**26**
SP L	Setpoint 1 low limit	112	
SP H	Setpoint 1 high limit	111	
SP2L	Setpoint 2 low limit	114	
SP2H	Setpoint 2 high limit	113	
Loc.L	Local setpoint trim low limit	67	**66**
Loc.H	Local setpoint trim high limit	66	**65**
SPrr	Setpoint rate limit 0: Off	35	**34**
Hb.ty	Holdback type for sp rate limit 0: Off 1: Low 2: High 3: Band	70	**69**
Hb	Holdback value for srtpoint rate limit	65	**64**
	Programmer State Write	57	
	Programmer state Read	23	**22**

Controller Display	Input Tab Parameter Description	Tag	ID
FiLt	Input 1 filter time constant 0: Off	101	
FLt2	Input 2 filter time constant 0: Off	103	
PV.iP	Select input 1 or input 2	288	**136**
F.1	Derived input function factor	292	**14**

	1		**0**
F2	Derived input function factor 2	293	**141**
Hi.IP	Switchover transition region high	286	**134**
Lo.IP	Switchover transition region low	287	**135**
	Potentiometer Calibration Enable	310	
	Potentiometer Input Calibration Node	311	
	Potentiometer Calibration Go	312	
EmIS	Emmisivity	38	**37**
EmI.S2	Emmisivity input 2	104	
CAL	User calibration enable 0: Factory 1: User	110	
CAL.S	Selected calibration point 0: None 1: Input 1 low 2: Input 1 high 3: Input 2 low 4: Input 2 high	102	
AdJ	User calibration adjust input 1	146	
AdJ	User calibration adjust input 2	148	
OFS.1	Input 1 calibration offset	141	**118**
OFS.2	Input 2 calibration offset	142	**119**
mV.1	Input 1 measured value	202	
mV.2	Input 2 measured value	208	
CJC.1	Input 1 cold junction temp. reading	215	
CJC.2	Input 2 cold junction temp. reading	216	
LI.1	Input 1 linearised value	289	**137**
LI.2	Input 2 linearised value	290	**138**
PV.SL	Currently selected setpoint	291	**139**

Controller Display	Output Tab Parameter Description	Tag	ID
OP.Lo	Low power limit	31	**30**
OP.Hi	High power limit	30	**29**

rOP.L	Remote low power limit	33	**32**
rOP.H	Remote high power limit	32	**31**
OPrr	Output rate limit 0: Off	37	**36**
FOP	Forced output level	84	**83**
CYC.H	Heat cycle time	10	**9**
hYS.H	Heat Hysteresis (on/off output)	86	**85**
ont.H	Heat output minimum on time 0: Auto	45	**44**
CYC.C	Cool cycle time	20	**19**
hYS.C	Cool Hysteresis (on/off output)	88	**87**
ont.C	Cool output minimum on time 0: Auto	89	**88**
HC.db	Heat/cool dead band (on/off op)	16	**15**
End.P	Power in end segment	64	**63**
Sb.OP	Sensor break output power	34	**33**
Sb.OP	On/Off Sensor Brk OP Power 0: -100% 1: 0% 2: 100%	40	**39**

Controller Display	Information Tab Parameter Description	Tag	ID
dI SP	Lower readout display 0: Standard 1: Load current 2: Output power 3: Status 4: Program time 5: None 6: Valve position 7: Process value 2 8: Ratio setpoint 9: Selected prog. number 10: Remote setpoint	106	
LoG.L	PV minimum	134	**111**
LoG.H	PV maximum	133	**110**
LoG.A	PV mean value	135	**112**
LoG.t	Time PV above threshold	139	**116**

	level		
LoG.u	PV threshold for timer log	138	**115**
rESL	Logging reset 0: Not reset 1: Reset	140	**117**
mCt	Maximum Control Task Time	201	
w.OP	Working output	4	**3**
SSr	PDSIO SSR status 0: Good 1: Load fail 2: Open 3: Heater fail 4: SSR fail 5: Sn fail	79	**78**
FF.OP	Feedforward component of output	209	
P OP	Proportional component of output	214	
I OP	Integral component of output	55	**54**
d OP	Derivative component of output	116	
UP S	VP motor calibration state 0: Start 1: Waiting 2: Open valve 3: BLUp/InDn 4: Ttup 5: Overshoot 6: InUp/BLDn 7: TT down 8: Open 9: Low lim 10: Stopping 11: Raise 12: Inert up 13: Lower 14: Low lim 15: Stopping 16: Lower 17: InDn/BL 99: Abort	210	

Controller	**Miscellaneous Tab**	Tag	ID
Display	**Parameter Description**		
	Instrument Mode	199	**120**
	Instrument Version Number	107	
	Instrument Ident	122	
	Slave Instrument Target Setpoint	92	**91**
	Slave Instrument Ramp Rate	93	**92**
	Slave Instrument Sync	94	**93**
	Remote SRL Hold	95	**94**
	CNOMO Manufacturers ID	121	
	Remote Parameter	151	
	Error Logged Flag	73	**72**
	Ramp Rate Disable	78	**77**
	Maximum Input Value	548	
	Minimum Input Value	549	
	Holdback Disable	278	**132**
	All User Interface Keys Disable	279	**133**

5.4.1 Ramp/Dwell Programmer Data

Program Data Organisation

There are no pre-defined DeviceNet Tags in this area. To read and write programs will require Explicit Messaging. See Section 8 for examples.

A 2400 series controller can contain multiple "programs", each consisting of up to 16 segments. The data for each program starts at the base tag address given by the following table:

Program	Base Address (Decimal)	Base Address (Hex)
Program 0 (Currently Running Program - changes permitted only in hold, and are not permanently stored)	8192	2000
Program 1	8328	2088
Program 2	8464	2110
Program 3	8600	2198
Program 4	8736	2220

The parameters used to describe a program are organised into 17 blocks, each of 8 words in length, starting at the base address for the program. There is one block for general program data, such as the units to be used for ramp and dwell times, and 16 further blocks for the segment data itself. To obtain the tag address of the data block for a given program, add the block offset given in the next table to the program

Contents	Offset (Decimal)	Offset (Hex)
Program General Data	0	0
Segment 1	8	8
Segment 2	16	10
Segment 3	24	18
Segment 4	32	20
Segment 5	40	28
Segment 6	48	30
Segment 7	56	38
Segment 8	64	40
Segment 9	72	48
Segment 10	80	50
Segment 11	88	58
Segment 12	96	60
Segment 13	104	68
Segment 14	112	70
Segment 15	120	78
Segment 16	128	80

Program General Data

The offsets of each parameter within the program general data block is given by the next table:

Address Offset	Parameter
0	HoldbackType 0: None 1: Low 2: High 3: Band
1	HoldbackValue
2	Ramp Units 0: Secs 1: Mins 2: Hours
3	Dwell Units

	0: Secs 1: Mins 2: Hours
4	Program Cycles
5	Reserved
6	Reserved
7	Reserved

Program Segment Data

Program segment data is specified using 8 tag addresses, with the contents varying depending on the type of the segment. The format per segment is detailed in the following table, which gives the offset from the start of a segment data block for each item.

Address Offset	Segment Types					
	STEP	DWELL	RAMP RATE	RAMP TIME TO TARGET	CALL	END
0	Segment Type	Segment Type	Segment Type	Segment Type	Segment Type	Segment Type
1	Target Setpoint		Target Setpoint	Target Setpoint		
2		Duration	Rate	Duration		
3					Program Number	End Type
4	Logic O/P's	Logic O/P's	Logic O/P's	Logic O/P's	Call Cycles	Logic O/P's
5						
6						
7						

Example Address calculations

Program 1, Segment 4, Segment Type = 8328 + 32 + 0 = 8360 (20A8 Hex)
Program 2, Holdback Value = 8464 + 0 + 1 = 8465 (2111 Hex)
Program 4 Segment 16, End Type = 8872 + 128 + 3 = 9003 (232B Hex)

Power Level in End Segment

This has the tag address 64 in 2400 controllers.

Summary of Programmer Enumerators

Controller Display	Parameter Description
tYPE	Current Segment Type 0: End 1: Ramp (Rate) 2: Ramp (Time to target) 3: Dwell 4: Step 5: Call
Endt	End Segment Type 0: Reset 1: Indefinate Dwell 2: Set Output

Controller Display	Parameter Description
Hb	Holdback Type 0: None 1: Low 2: High 3: Band
dwL U	Dwell Units 0: Seconds 1: Minutes 2: Hours
rmP U	Ramp Units 0: Seconds 1: Minutes 2: Hours

6.Transferring Data - 2200

A network has been set up with a 2500 base at address 25, a 2400 at address 24 and a 2200 at address 22. The baud rate has been set at 500k.

The table below lists the factory default assignment for the Polled I/O message described in the [IO_Info] section of the EDS file.

Default 2200 INPUT Parameters

Input Words	Parameter
1	Measured Value (PV)
2	Target Setpoint (TS)
3	Output Power (OP)
4	Summary Output Status

where the Summary Output Status bits are set according to the table below.

Summary Output Status	
Bit	**Description**
0	Alarm 1 State
1	Alarm 2 State
2	Alarm 3 State
3	Alarm 4 State
4	Manual Mode
5	Sensor Break
6	Loop Break
7	Heater Fail
8	Load Fail
9	Ramp/Program
10	PV out of range
11	SSR Fail
12	New Alarm
13	Remote input sensor break
14	Reserved
15	Reserved

Default 2200 OUTPUT Parameters

A single user configurable output parameter is available.

Output Words	Parameter
1	Parameter Value
2	‘Address’

To write a parameter using the Polled I/O message, first enter the new Parameter Value, then the Address. To calculate the value required for the ‘Address’, add 256 to the Attribute ID value shown in the Parameter Address Maps shown at the end of this document.

For example, to write to the Target Setpoint – Attribute value 5 – the Address value would be 261.

6.1 DEFAULT EXAMPLE 1

The 2200 parameter tables are not directly configurable so the only task is to transfer the data required. There are 4 input parameters (8 bytes) and 2 output parameters (4 bytes). The example ***2200Examples.rss*** has been set up as follows and the data may be used as required.

2200 parameter	I/O file
Measured Value (PV)	I6.6
Target Setpoint (TS)	I6.7
Output Power (OP)	I6.8
Summary Output Status	I6.9

To read any other parameters within the instrument the only possibility is to use explicit messaging which is described in the section below.
There is only one write parameter, but by selecting the appropriate ID from the tables in Section 6.3 any parameter in the instrument can be written to. Alternatively see the next section on explicit messaging.

2200 parameter	I/O file
Set value	O6.7
Parameter ID + 256	O6.8

The normal requirement is to set the address to 256 + 5 to write to the setpoint (ID = 5). The set value is then written to the controller. There is a Custom Data Monitor showing these values in the example.
To write to a different parameter set the new value as required and its address in the same copy sequence.
e.g. set value 25 and address 272 will set the proportional band to 25.

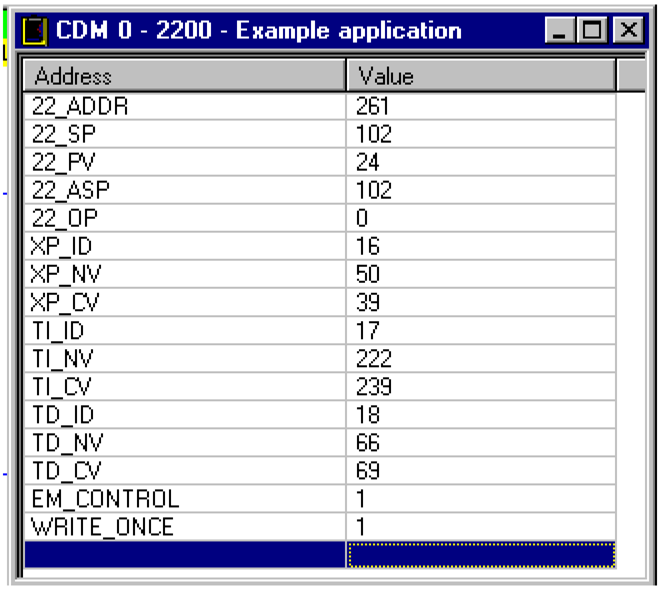
6.2 EXPLICIT MESSAGING

An example is included in the ***2200Examples.rss*** application. This will either read 3 parameters continually or, with new values entered into the table a 'write-once' sequence may be executed. The 3 parameters selected are the PID terms.
The ladder logic sequence has been done as a straight sequence, avoiding indirect addressing, error detection etc., for simplicity and only as an example to show communication. Section 6.2.4 gives key data on Explicit Messaging in the 1747SDN Scanner Module.

6.2.1 User Parameters

N7:60 to 69 are the parameters used to send and read values from the 2200.

N7 address	Value	Description
60	16	Prop band ID
61		New value
62		Current Value
63	17	Integral ID
64		New value
65		Current Value
66	18	Derivative ID
67		New value
68		Current Value
69	0 1 2	Do nothing Read continuous Write once



CDM 0 - 2200 - Example application

Address	Value
22_ADDR	261
22_SP	102
22_PV	24
22_ASP	102
22_OP	0
XP_ID	16
XP_NV	50
XP_CV	39
TI_ID	17
TI_NV	222
TI_CV	239
TD_ID	18
TD_NV	66
TD_CV	69
EM_CONTROL	1
WRITE_ONCE	1

The first 5 parameters are read and written using the normal DeviceNet polling.

XP_*, TI_* and TD_* are the parameters taken from the previous table.

Explicit Messaging will be used to read the current value (CV) or to write the new value (NV). The parameters are selected by choosing the correct Attribute ID from the tables in 6.3. The PID parameters have IDs 16, 17, and 18.

Use EM_CONTROL to read (=1) write once (= 2) or do nothing (=0).

Reset the WRITE_ONCE bit to write continuously.

6.2.2 Explicit Read Message

Transaction Read File – request sent to scanner **M0:224**. The required format is shown below. The transaction is initiated by sending this to the M0 file and the waiting for the response in the M1 file.

N7 address	Description	Upper byte	Lower Byte	Description	Word value
70	Transaction ID	00000001	00000001	Command: Execute	257
71	Port	00000000	00000110	Byte Size	6
72	Service (14)	00001110	00010110	MacID (22)	3606
73	Class	00000000	01100100	Class=100	100
74	Instance	00000000	00000001	Instance =1	1
75	Attribute	Set to required parameter (from N60, 63 or 66)			

Note: byte size is class, instance, and attribute.

When transaction has been executed then Scanner Input file I:6/15 is set

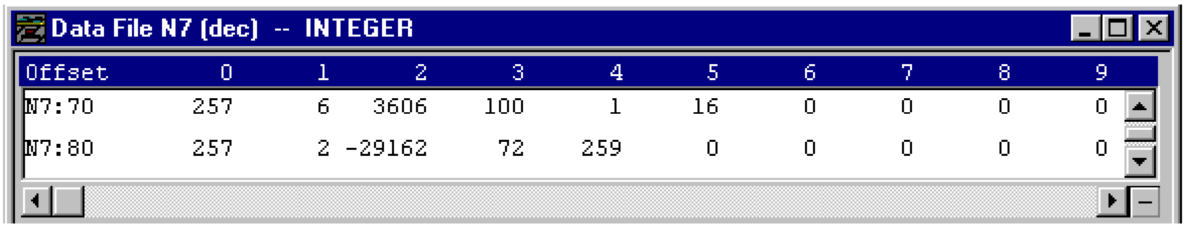
Transaction Read File – response received from scanner **M1:224**

N7 address	Description	Upper byte	Lower Byte	Description	Word value
80	Transaction ID	00000001	00000001	Status: OK	257
81	Port	00000000	00000110	Byte Size	2
82	Service	10001111	00010110	MacID (22)	
83	Value	Transfer this to N62, N65, N68			

Clear Transaction – sent to scanner **M0:224**

N7 address	Description	Upper byte	Lower Byte	Description	Word value
84	Transaction ID	00000001	00000003	Command: Execute	259

When transaction has been cleared then Scanner Input file I:6/15 is reset.



Data File N7 (dec) -- INTEGER

Offset	0	1	2	3	4	5	6	7	8	9
N7:70	257	6	3606	100	1	16	0	0	0	0
N7:80	257	2	-29162	72	259	0	0	0	0	0

6.2.3 Explicit Write

Transaction Write File – request sent to scanner **M0:224**

N7 address	Description	Upper byte	Lower Byte	Description	Word value
90	Transaction ID	00000001	00000001	Command: Execute	257
91	Port	00000000	00001000	Byte Size	8
92	Service (16)	00010000	00010110	MacID (22)	4118
93	Class	00000000	01100100	Class=100	100
94	Instance	00000000	00000001	Instance =1	1
95	Attribute	Set to required parameter (from N60, 63 or 66)			
96	New value	Get from N61, N64, N67			

Note: byte size is class, instance, attribute, and value

When transaction has been executed then Scanner Input file I:6/15 is set

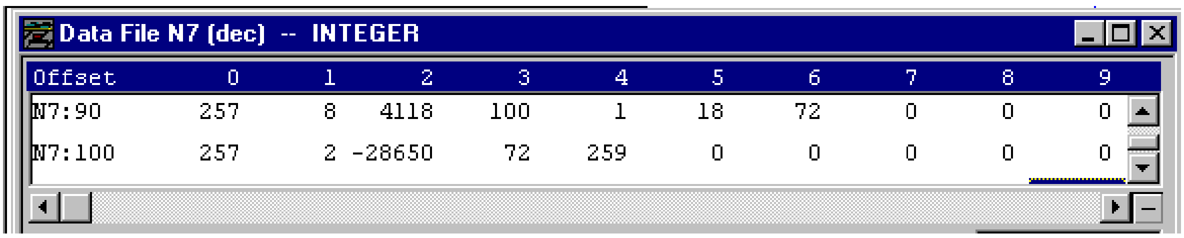
Transaction Write File – response received from scanner M1:224

N7 address	Description	Upper byte	Lower Byte	Description	Word value
100	Transaction ID	00000001	00000001	Status: OK	257
101	Port	00000000	00000010	Byte Size	2
102	Service	10010000	00010110	MacID (22)	
103	Value	Transfer this to N62, N65, N68			

Clear Transaction – sent to scanner M0:224

N7 address	Description	Upper byte	Lower Byte	Description	Word value
104	Transaction ID	00000001	00000003	Reset	259

When transaction has been cleared then Scanner Input file I:6/15 is reset.



Data File N7 (dec) -- INTEGER

Offset	0	1	2	3	4	5	6	7	8	9
N7:90	257	8	4118	100	1	18	72	0	0	0
N7:100	257	2	-28650	72	259	0	0	0	0	0

6.2.4 Explicit Message Descriptions

Transaction ID – a user given number to identify the transaction. Up to 10 may be used but the 2200 example only ever uses one.

MacID is the slave address.

Commands – used in transaction request
This example uses 1 and 3.

Command Code	Description
0	Ignore
1	Execute
2	Get status of TXID
3	Reset all TXID
4	Delete transaction TXID
5-255	Reserved

Status – in transaction response.
The scanner errors are not used in this example.

Status Code	Description
0	Ignore
1	OK
2	In progress
3 - 15	Errors (See scanner documentation)
16-255	Reserved

Services.
This example only uses the 'Single' services.

Name	Code	Description
Get Attribute Single	14	Upload single value
Set Attribute Single	16	Download single value
Get Attribute All	1	Upload all value
Set Attribute All	2	Download all value

6.3 2200 CLASS INSTANCE &ATTRIBUTE ID TABLE

Parameters are Read/Write (RW) which may be used as OUTPUT or INPUT parameters or Read Only (RO) which may only be used as INPUT parameters.

Class = 100 Instance = 1 Attribute in Table.

Home List

Mn	Parameter Name	Attribute		
	Process Variable	0	0h	RO
VPoS	Valve Position	6	6h	
OP	% Output Level	1	1h	RO RW
w.SP	Working Setpoint	2	2h	RO
SP	Target Setpoint (SP)	5	5h	RW
m-A	Manual Mode	8	8h	RW
AmPS	Load Current Requires PDSIO Mode 2	7	7h	RO
DiSP	Lower Display	139	8Bh	R/W
C.id	Customer Defined Identification Number	140	8Ch	R/W

Alarm List

Mn	Parameter Name	Attribute		
1---	Alarm 1 Setpoint	9	09h	RW
2---	Alarm 2 Setpoint	10	0Ah	RW
3---	Alarm 3 Setpoint	11	0Bh	RW
4---	Alarm 4 Setpoint	12	0Ch	RW
HY	Alarm 1-4 Hysteresis	141	8Dh	R/W
Lbt	Loop Break Time	13	0Dh	RW

Autotune List

Mn	Parameter Name	Attribute		
tunE	Autotune Enable	14	0Eh	RW
Adc	Automatic Droop Compensation (PD only control)	15	0Fh	RW

PID List

Mn	Parameter Name	Attribute		
PB	Proportional Band	16	10h	RW
Ti	Integral Time	17	11h	RW
Td	Derivative Time	18	12h	RW
Res	Manual Reset %	19	13h	RW
Lcb	Cutback Low	20	14h	RW
Hcb	Cutback High	21	15h	RW
reL.c	Relative Cool Gain	22	16h	RW

Setpoint List

Mn	Parameter Name	Attribute		
SSEL	Setpoint Select	23	17h	RW
L-r	Local or Remote Setpoint Select	24	18h	RW
SP 1	Setpoint 1	25	19h	RW
SP 2	Setpoint 2	26	1Ah	RW
rm.SP	Remote Setpoint	27	1Bh	RW
Loc.t	Local Setpoint Trim	28	1Ch	RW
SP1L	Setpoint 1 Low Limit	29	1Dh	RW
SP1H	Setpoint 1 High Limit	30	1Eh	RW
SP2L	Setpoint 2 Low Limit	31	1Fh	RW
SP2H	Setpoint 2 High Limit	32	20h	RW
LocL	Local Setpoint Trim Low Limit	33	21h	RW
LocH	Local Setpoint Trim High Limit	34	22h	RW
SPrr	Setpoint Rate Limit	35	23h	RW
dwEL	Dwell Time	36	24h	RW
End.t	Go To State At End of Program	37	25h	RW
ProG	Program Control	38	26h	RW
StAt	Program Status	39	27h	RO

Input List

Mn	Parameter Name	Attribute		
FiLt	Input filter time	40	28h	RW
OFSt	Process Value Offset	41	29h	RW
CAL	Calibration Type	42	2Ah	RW
CAL.S	Calibration Select	43	2Bh	RW
AdJ	User Calibration Adjust	44	2Ch	RO
CJC	Cold Junction Compensation Temperature	45	2Dh	RO
mU	Input Millivolt Value	46	2Eh	RO

Output List

Mn	Parameter Name	Attribute		
OP.Lo	Low Power Limit	47	2Fh	RW
OP.Hi	High Power Limit	48	30h	RW
Sb.OP	Sensor Break Output	49	31h	RW

CYC.H	Heat Cycle Time	50	32h	RW
CYC.C	Cool Cycle Time	51	33h	RW
ont.H	Heat Output Minimum ON Time	52	34h	RW
ont.C	Cool Output Minimum ON Time	53	35h	RW
mtr	Motor Travel Time	54	36h	RW

On/Off List

Mn	Parameter Name	Attribute		
hYS.H	Heat Hysteresis	55	37h	RW
hYS.C	Cool Hysteresis	56	38h	RW
HC.db	Heat/Cool Dead Band	57	39h	RW

Comms List

	Parameter Name	Attribute		
Addr	Comms Address	58	3Ah	RW

Misc. Status & Comms-Only Parameters

Mn	Parameter Name	Attribute		
	Process Error	143	8Fh	RO
	Controller Version Number	61	3Dh	RO
	CNOMO Manufactures Identifier	62	3Eh	RO
	Controller Identifier	59	3Bh	RO
	Instrument Mode	60	3Ch	RW
	PV Millivolts From Comms	63	3Fh	RW
	Input Test Point Enable	64	40h	
	Sensor Break Sourced From Test	99	63h	
	Filter Initialization Flag	66	42h	
	Sensor Break Status Flag	67	43h	RO
	Acknowledge All Alarms	68	44h	RW
	Disable Keys	142	8Eh	RW

Control Status

Bit	Description	Attribute		
0	Control algorithm freeze	4	04h	RW
1	PV input sensor broken			
2	PV out of sensor range			
3	Self-tune fail			
4	PID servo signal			
5	PID debump signal			
6	Fault detected in closed loop behaviour (loop break)			
7	Freezes the integral accumulator			
8	Indicates that a tune has completed successfully			
9	Direct/reverse acting control			
10	Algorithm initialization flag			
11	PID demand has been lifted			
12	Reserved			
13	Auto/Adaptive tune enabled			
14	Automatic droop compensation enabled			
15				

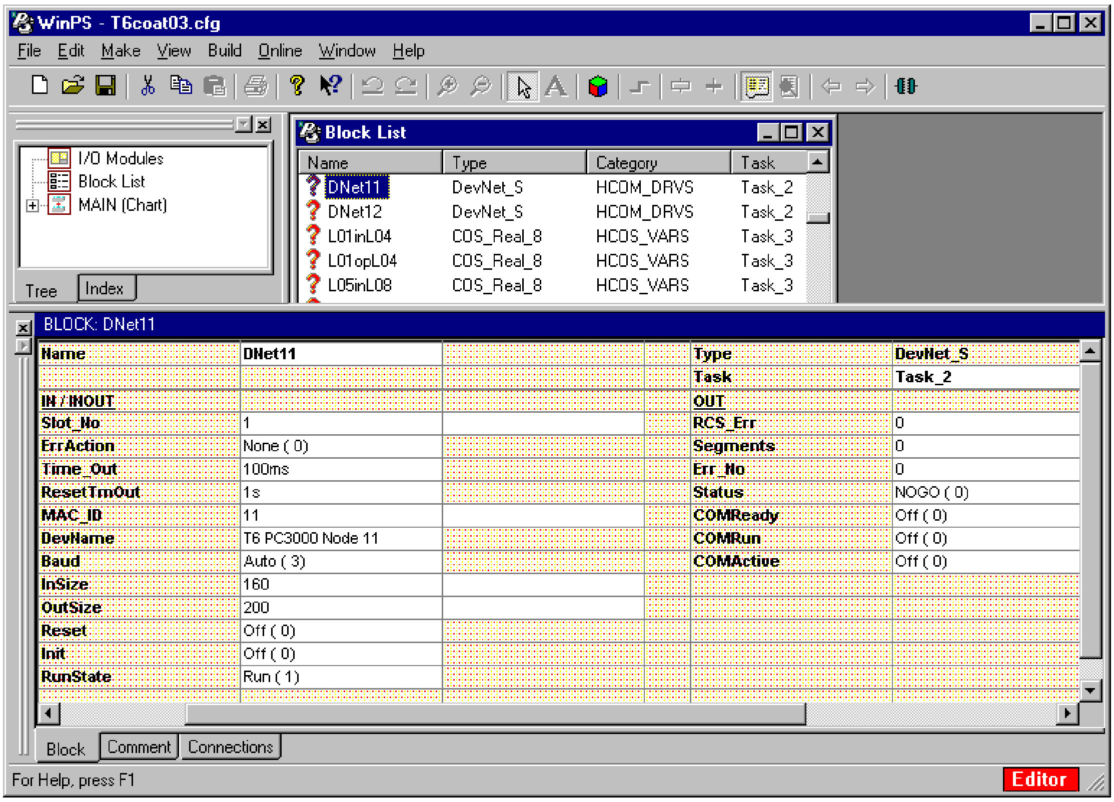
7.PC3000

To place a PC3000 as a slave on a DeviceNet network requires a DeviceNet Communications card placed in the first available slot in the main rack – slot 1 in this example. The card uses a Hilscher sub-board which may be configured using Hilscher's own configuration tools. However it can also be configured by the PC3000 application program itself.

The PC3000 LCM+ requires the extra RAM3 board to take the Fieldbus downloadable function block library and the PS or WinPS tools require the library templates.

This is a brief summary of the requirements of the PC3000 application program. Firstly an instance of the HCOS_DRVS.Com_Table has to be created to allocate the necessary memory etc.

Then an instance of a DeviceNet slave HCOS_DRVS.DevNet_S has to be created.

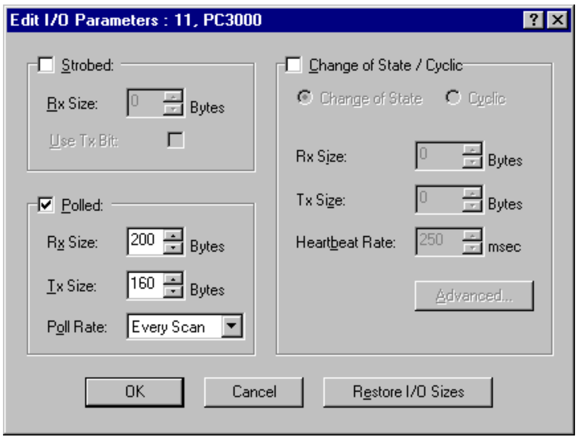


In this example the slave address (Mac_ID) has been set to 11 and the Baud rate on Auto – it will set itself.

It is also necessary to calculate the exact INPUT (160 bytes) and OUTPUT (200 bytes) sizes and enter them. These sizes MUST be multiples of 8 bytes. For this reason the maximum number of INPUT bytes is 248 and OUTPUT bytes is 248.

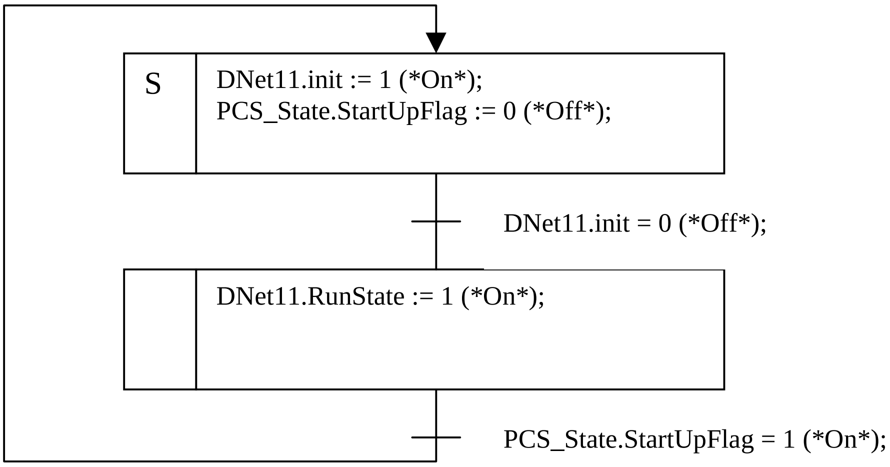
There must also be the corresponding instances of HCOS_VARS variables to match these byte sizes. COS_Real_8 have been used and there are 20 set with 'Mode' to INPUT with addresses from 1:0:2~, 1:16:2~, 1:32:2~ up to 1:144:2~ giving 160 bytes or 80 variables.

For the OUTPUTS there are 25 with 'Mode' set to Output with addresses from 1:0:2~ to 1:192:2~ with this last block limited to 4 variables (NoOfVars = 4). This gives 200 bytes or 100 variables.



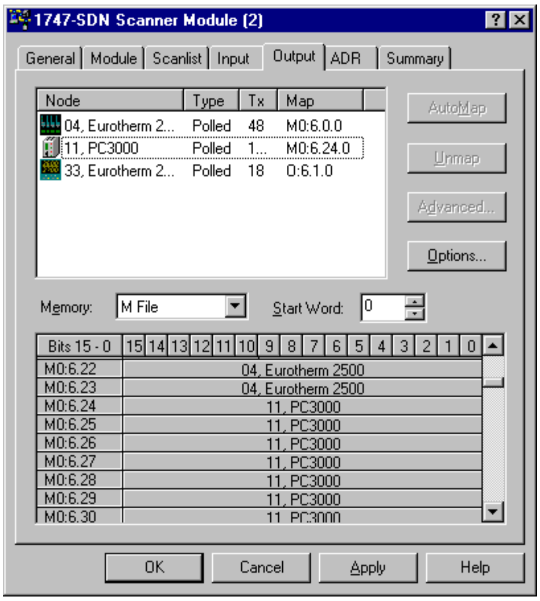
The INPUT bytes to PC3000 are OUTPUT bytes (Tx) from the plc and vice versa as shown in the scanner I/O configuration shown.

The final task is a short sequence to auto configure the DeviceNet card on power up.



The first step tells the slave to configure itself with the data in the driver block. When this is complete it resets the init flag and step 2 puts the slave in run mode. On power up the startup flag ensures the initialisation is repeated. Because the slave board is separate from the PC3000 LCM itself it does not affect the DeviceNet network if the PC3000 is not in Run mode. This means you can build and download without disturbing the network.

Finally with this number of parameters the data would be transferred using the M files.



Note that the PLC output parameters are written all the time – this means that any values in PC3000 INPUT variables will be over written as soon as communications is established.

8.Explicit Message Block Read/Write

First see Section 6.2 on the use of Explicit Messaging to read and write individual parameters on the 2200 controller. Explicit messaging will work on any slave given the CLASS, INSTANCE and ID of the required parameter. It can therefore also be used with the 2400, 2500, 2600 or 2700.

Explicit messaging involves sending a precisely formatted file to the Scanner file M1:224 and the response is picked up from M0:224. See the scanner's supporting documentation for details and Section 6.2 for a basic example with the 2200.

This block mode uses a Tag Access Application Variable to read/write 2400, 2500, 2600 or 2700 controller variables in blocks of up to 32 bytes starting at a 16 bit Tag address. This is usually 16 analogue values, but can be 8 32 bit values including single precision floating point data. Most often used to download recipe values or ramp/soak programs to controllers. It is not available in 2200 or PC3000.

The examples will be a read block and a write block where the start address and block size are integers which can be set to suit.

A further example will read and write a specific program to a 2700.

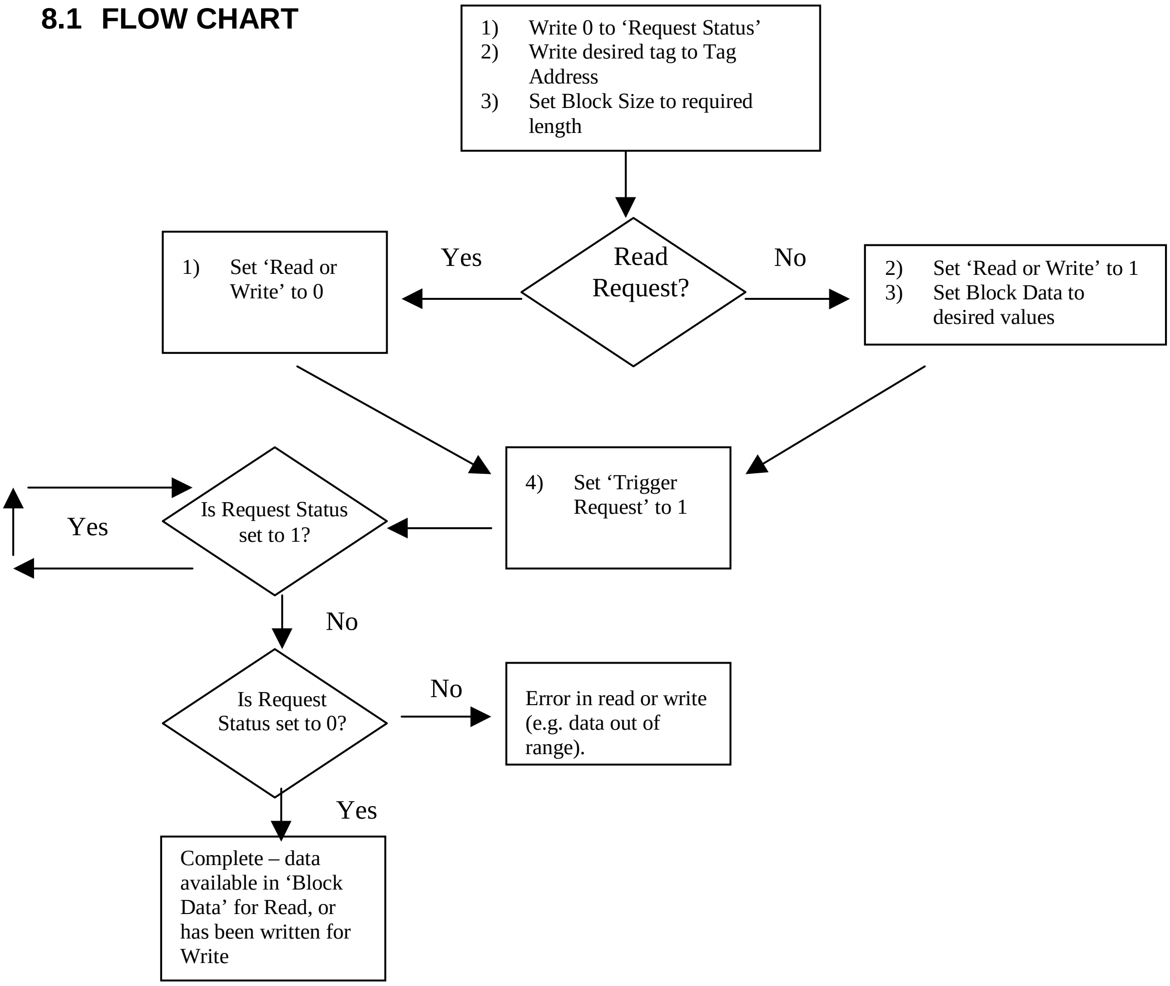
This uses the Tag Access Application Variable Object in the 2400, 2500 and 2600/2700 device parameter list.

Class is 101, Instance is 1

Attribute ID	Variable	Description
1	Tag Address	Base Tag for the Block access
2	Block Size	Size of Block in 16 bit words
3	Block Data	Actual data values
4	Read or Write	Selects read or write operation
5	Trigger Request	Triggers request
6	Request Status	Write value of 0 to clear. Read values: 0: Complete 1: Request Pending 3: Error in read/write request (for example write data is out of range).

The flowchart shows how these variables are used.

8.1 FLOW CHART



8.2 IMPLEMENTATION – 2700 READ BLOCK

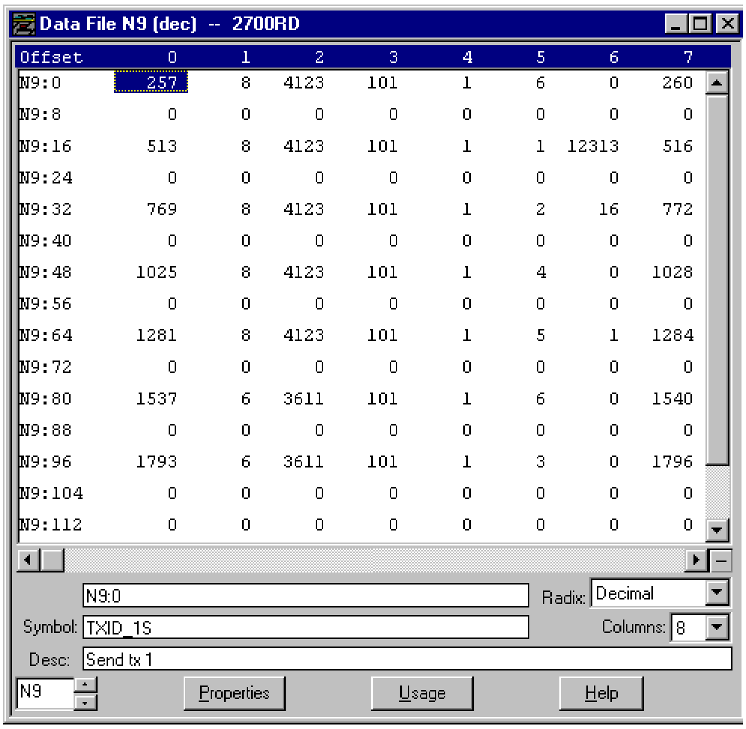
The flowchart is implemented in the follow order in the plc. 2700 address is 27.

Order	Action.	Value	Transaction Table	
0	Clear all transactions in Scanner	Write Reset Command to Scanner	Request	Response
1	Clear sequence in 2700	Write 0 to ID 6	N9:0	N9:8
2	Write Base Tag address	Write <addr> to ID 1	N9:16	N9:24
3	Write Block Size	Write <size> to ID 2	N9:32	N9:40
4	Set mode to Read	Write 0 to ID 4	N9:48	N9:56
5	Execute instruction	Write 1 to ID 5	N9:64	N9:72
6	Check result	Read ID 6	N9:80	N9:88
7	If result is OK transfer data	Read ID 3	N9:96	N9:104
Back to 6	If result is error repeat Execute			
Back to 1	Back to the start after time delay			

This is the N9 data file as set up. With 8 columns each line represents an EM request followed by an EM response.

The N9 table addresses are the start of each transaction block. These expand to the following data table.

The tables below show how the transaction requests are built up and the expected responses.



1a. Request to clear sequence in instrument

N9 address	Description	Upper byte	Lower Byte	Description	Word value
0	Transaction ID=1	00000001	00000001	Command: Execute	257
1	Port	00000000	00001000	Byte Size	8
2	Service (16) = write	00010000	00011011	MacID (27)	4123
3				Class=101	101
4				Instance =1	1
5				ID = 6	6
6				0 = clear	0

Byte size is Class, Instance, Attribute and Value

1b. Response to clear request

N9 address	Description	Upper byte	Lower Byte	Description	Word value
8	Transaction ID=1	00000001	00000001	Status: OK	257

1c. Delete transaction

N9 address	Description	Upper byte	Lower Byte	Description	Word value
7	Transaction ID=1	00000001	00000004	Command: Delete transaction	260

2a. Request to write block base address (=12313 the first half of segment 1 in program)

N9 address	Description	Upper byte	Lower Byte	Description	Word value
16	Transaction ID=2	00000010	00000001	Command: Execute	513
17	Port	00000000	00001000	Byte Size	8
18	Service (16) = write	00010000	00011011	MacID (27)	4123
19				Class=101	101
20				Instance =1	1
21				ID = 2	2
22				Value	0

Byte size is Class, Instance, Attribute and Value

2b. Response to write block request

N9 address	Description	Upper byte	Lower Byte	Description	Word value
24	Transaction ID=2	00000010	00000001	Status: OK	513

2c. Delete transaction

N9 address	Description	Upper byte	Lower Byte	Description	Word value
23	Transaction ID=2	00000010	00000004	Command: Delete transaction	516

3a. Request to write block size (=16)

N9 address	Description	Upper byte	Lower Byte	Description	Word value
32	Transaction ID=3	00000011	00000001	Command: Execute	769
33	Port	00000000	00001000	Byte Size	8
34	Service (16) = write	00010000	00011011	MacID (27)	4123
35				Class=101	101
36				Instance =1	1
37				ID = 2	2
38				Value	16

Byte size is Class, Instance, Attribute and Value

3b. Response to write block size request

N9 address	Description	Upper byte	Lower Byte	Description	Word value
40	Transaction ID=3	00000011	00000001	Status: OK	769

3c. Delete transaction

N9 address	Description	Upper byte	Lower Byte	Description	Word value
39	Transaction ID=3	00000011	00000004	Command: Delete transaction	772

4a. Request to set mode to READ (=0)

N9 address	Description	Upper byte	Lower Byte	Description	Word value
48	Transaction ID=4	00000100	00000001	Command: Execute	1025
49	Port	00000000	00001000	Byte Size	8
50	Service (16) = write	00010000	00011011	MacID (27)	4123
51				Class=101	101
52				Instance =1	1
53				ID = 4	4
54				Value	0

Byte size is Class, Instance, Attribute and Value

4b. Response to mode READ request

N9 address	Description	Upper byte	Lower Byte	Description	Word value
56	Transaction ID=4	00000100	00000001	Status: OK	1025

4c. Delete transaction

N9 address	Description	Upper byte	Lower Byte	Description	Word value
55	Transaction ID=4	00000100	00000004	Command: Delete transaction	1028

5a. Request to write Execute instruction

N9 address	Description	Upper byte	Lower Byte	Description	Word value
64	Transaction ID=5	00000101	00000001	Command: Execute	1281
65	Port	00000000	00001000	Byte Size	8
66	Service (16) = write	00010000	00011011	MacID (27)	4123
67				Class=101	101
68				Instance =1	1
69				ID = 5	5
70				Value	1

Byte size is Class, Instance, Attribute and Value

5b. Response to write Execute instruction

N9 address	Description	Upper byte	Lower Byte	Description	Word value
72	Transaction ID=5	00000101	00000001	Status: OK	1281

5c. Delete transaction

N9 address	Description	Upper byte	Lower Byte	Description	Word value
71	Transaction ID=5	00000101	00000004	Command: Delete transaction	1284

6a. Request to Check Status (0 = Block read OK, 3 = Error)

N9 address	Description	Upper byte	Lower Byte	Description	Word value
80	Transaction ID=6	00000110	00000001	Command: Execute	1537
81	Port	00000000	00000110	Byte Size	6
82	Service (14) = read	00001110	00011011	MacID (27)	3611
83				Class=101	101
84				Instance =1	1
85				ID = 6	6

Byte size is Class, Instance, Attribute and Value

6b. Response to Check status request

N9 address	Description	Upper byte	Lower Byte	Description	Word value
88	Transaction ID=6	00000110	00000001	Status: OK	1537
89				Byte Size	2
90					
91	Value returned			0 = action completed OK	0

6c. Delete transaction

N9 address	Description	Upper byte	Lower Byte	Description	Word value
87	Transaction ID=6	00000110	00000004	Command: Delete transaction	1540

7a. Request to read all block values

N9 address	Description	Upper byte	Lower Byte	Description	Word value
96	Transaction ID=7	00000111	00000001	Command: Execute	1793
97	Port	00000000	00000110	Byte Size	6
98	Service (14) = read	00001110	00011011	MacID (27)	4123
99				Class=101	101
100				Instance =1	1
101				ID = 3	5

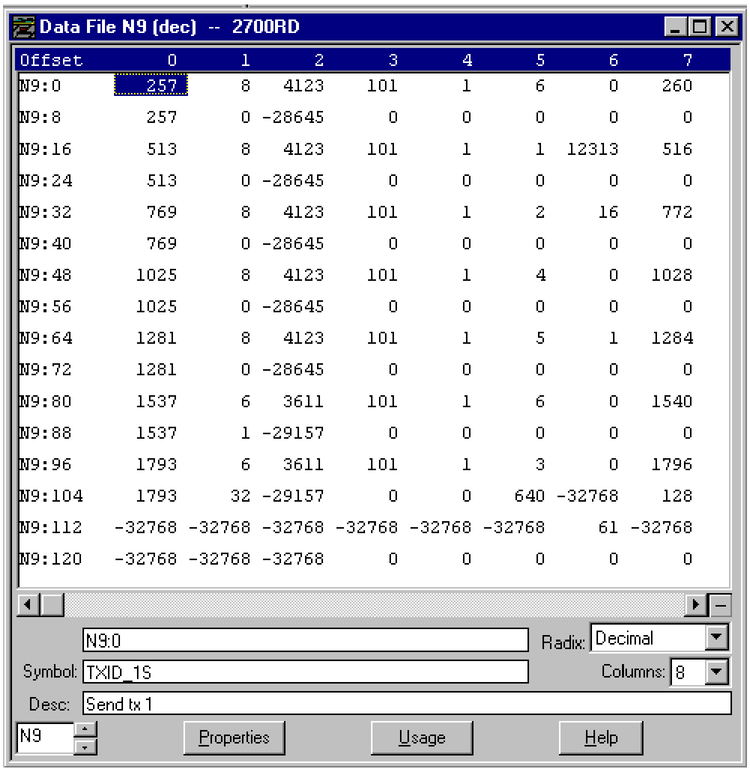
Byte size is Class, Instance, Attribute and Value

7b. Response to write Execute instruction

N9 address	Description	Upper byte	Lower Byte	Description	Word value
104	Transaction ID=7	00000111	00000001	Status: OK	1793
105				Byte Size	32
106					
107				Value_0	
To 122				Value_15	

7c. Delete transaction

N9 address	Description	Upper byte	Lower Byte	Description	Word value
103	Transaction ID=7	00000111	00000004	Command: Delete transaction	1796



Offset	0	1	2	3	4	5	6	7
N9:0	257	8	4123	101	1	6	0	260
N9:8	257	0	-28645	0	0	0	0	0
N9:16	513	8	4123	101	1	1	12313	516
N9:24	513	0	-28645	0	0	0	0	0
N9:32	769	8	4123	101	1	2	16	772
N9:40	769	0	-28645	0	0	0	0	0
N9:48	1025	8	4123	101	1	4	0	1028
N9:56	1025	0	-28645	0	0	0	0	0
N9:64	1281	8	4123	101	1	5	1	1284
N9:72	1281	0	-28645	0	0	0	0	0
N9:80	1537	6	3611	101	1	6	0	1540
N9:88	1537	1	-29157	0	0	0	0	0
N9:96	1793	6	3611	101	1	3	0	1796
N9:104	1793	32	-29157	0	0	640	-32768	128
N9:112	-32768	-32768	-32768	-32768	-32768	-32768	61	-32768
N9:120	-32768	-32768	-32768	0	0	0	0	0

The N9 data file after the sequence has been run.

Each transaction has been successful as the first value is the same as the sent value (257, 5123, 769 etc.)

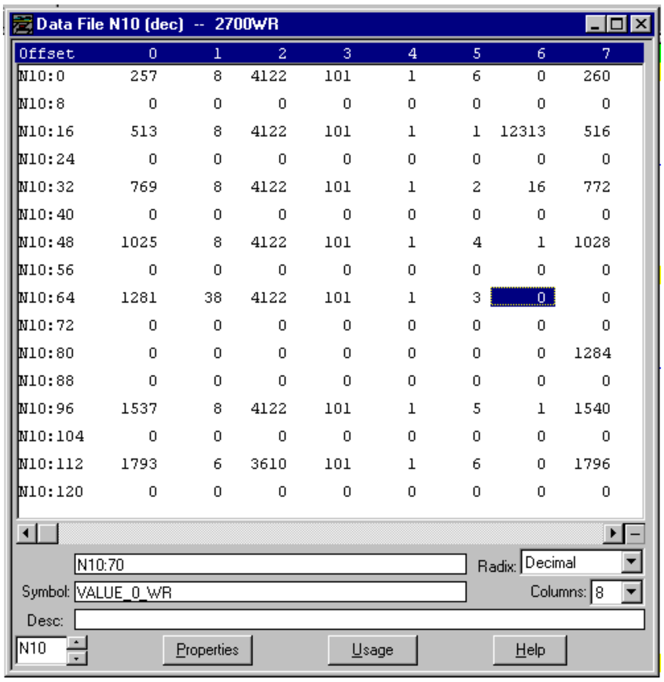
The 16 values are 0, 0, 640, -32768, 128 etc.

The application is in the ladder 2700RD and the bits B3:2/0 to 15 are used to control it. B3:2/15 starts the sequence. It repeats every 2 seconds based on timer T4:0. It was found necessary to add timer T4:1 to provide a short 50mS delay between each transaction otherwise the Scanner was liable to get overloaded and fault.

8.3 IMPLEMENTATION – 2700 WRITE BLOCK

The flowchart is implemented in the follow order in the plc.

Order	Action.	Value	Transaction Table	
0	Clear all transactions in Scanner	Write Reset Command to Scanner	Request	Response
1	Clear sequence in 2700	Write 0 to ID 6	N10:0	N10:8
2	Write Base Tag address	Write <addr> to ID 1	N10:16	N10:24
3	Write Block Size	Write <size> to ID 2	N10:32	N10:40
4	Set mode to Write	Write 1 to ID 4	N10:48	N10:56
5	Trnsfer Data	Write <values> to ID3	N10:64	N10:88
6	Execute instruction	Write 1 to ID 5	N10:96	N10:104
7	Check result	Read ID 6	N10:112	N10:120
	If result is error set flag		B3:1/14	
Back to 0	Back to the start after time delay			


Data File N10 (dec) -- 2700WR

Offset	0	1	2	3	4	5	6	7
N10:0	257	8	4122	101	1	6	0	260
N10:8	0	0	0	0	0	0	0	0
N10:16	513	8	4122	101	1	1	12313	516
N10:24	0	0	0	0	0	0	0	0
N10:32	769	8	4122	101	1	2	16	772
N10:40	0	0	0	0	0	0	0	0
N10:48	1025	8	4122	101	1	4	1	1028
N10:56	0	0	0	0	0	0	0	0
N10:64	1281	38	4122	101	1	3	0	0
N10:72	0	0	0	0	0	0	0	0
N10:80	0	0	0	0	0	0	0	1284
N10:88	0	0	0	0	0	0	0	0
N10:96	1537	8	4122	101	1	5	1	1540
N10:104	0	0	0	0	0	0	0	0
N10:112	1793	6	3610	101	1	6	0	1796
N10:120	0	0	0	0	0	0	0	0

N10:70 Radix: Decimal
Symbol: VALUE_0_WR Columns: 8
Desc:
N10 Properties Usage Help

This N10 data file is made up in exactly the same way as the read table in the previous example.

With 8 columns each line represents an EM request followed by an EM response.

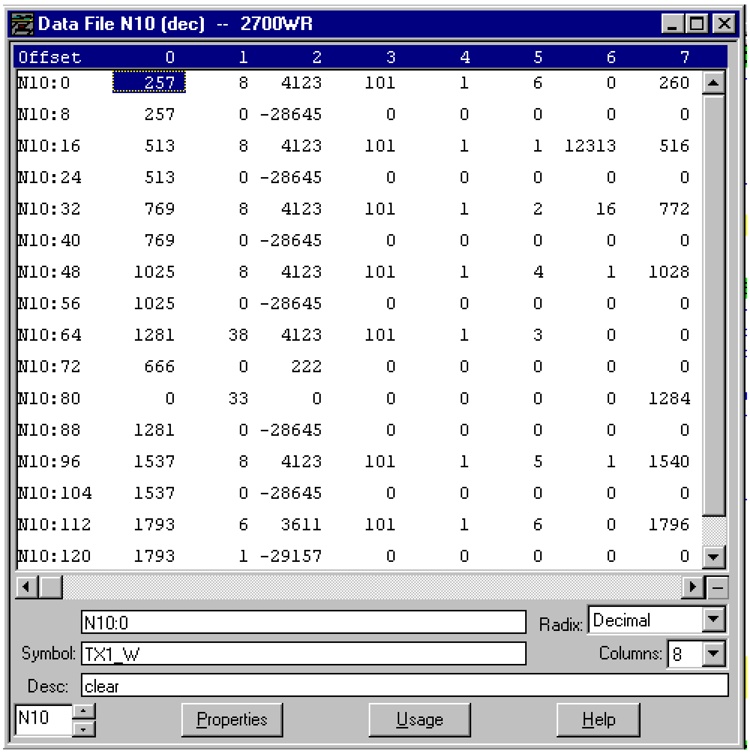
In this case the 16 values to be written are in N10:70 to 85

After the ladder is run the results are as shown.

Each transaction was successful as the response was the same as the request (257, 513, 769 etc).

The values written were the 3 loop setpoints for segment 1 of the program and are Loop 1 = 666 (N10:72) Loop 2 = 222 (N10: 74) Loop 3 = 33 (N10:81)

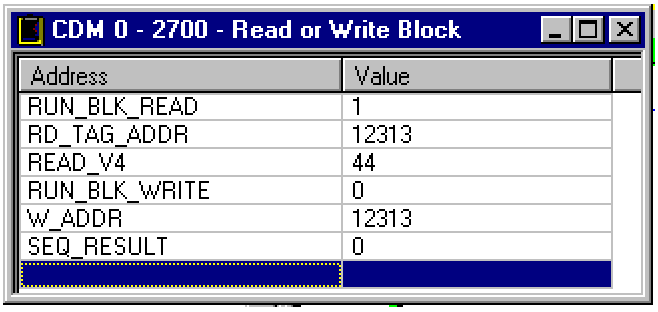
N10:123 shows that the write was successful.


Data File N10 (dec) -- 2700WR

Offset	0	1	2	3	4	5	6	7
N10:0	257	8	4123	101	1	6	0	260
N10:8	257	0	-28645	0	0	0	0	0
N10:16	513	8	4123	101	1	1	12313	516
N10:24	513	0	-28645	0	0	0	0	0
N10:32	769	8	4123	101	1	2	16	772
N10:40	769	0	-28645	0	0	0	0	0
N10:48	1025	8	4123	101	1	4	1	1028
N10:56	1025	0	-28645	0	0	0	0	0
N10:64	1281	38	4123	101	1	3	0	0
N10:72	666	0	222	0	0	0	0	0
N10:80	0	33	0	0	0	0	0	1284
N10:88	1281	0	-28645	0	0	0	0	0
N10:96	1537	8	4123	101	1	5	1	1540
N10:104	1537	0	-28645	0	0	0	0	0
N10:112	1793	6	3611	101	1	6	0	1796
N10:120	1793	1	-29157	0	0	0	0	0

The application is in the ladder 2700WR and the bits B3:3/0 to 15 are used to control it. B3:3/15 starts the sequence. It repeats every 2 seconds based on timer T4:2. It was found necessary to add timer T4:3 to provide a short 50mS delay between each transaction otherwise the Scanner was liable to get overloaded and fault.

A custom data monitor shows the key variables for this and the previous READ example. The RD_TAG_ADDR and W_ADDR are set to the start of the first segment of the programmer.


CDM 0 - 2700 - Read or Write Block

Address	Value
RUN_BLK_READ	1
RD_TAG_ADDR	12313
READ_V4	44
RUN_BLK_WRITE	0
W_ADDR	12313
SEQ_RESULT	0

8.4 IMPLEMENTATION – 2600 PROGRAMMER UPLOAD/DOWNLOAD

This follows on from the previous example. It requires a 2600 (or 2700) at address 26 to be mapped into the scan list.
Sequence for Upload (Block READ)

Order	Action.	Value
0	Clear all transactions in Scanner	Write Reset Command to Scanner
1	Clear sequence in 2600	Write 0 to ID 6
2	Set mode to Read	Write 1 to ID 4
3	Write Block Size	Write <size> to ID 2
4	Write 1st Base Tag address	Write <addr> to ID 1
5	Execute instruction	Write 1 to ID 5
7	Check result	Read ID 6
8	If result is error set flag	
9	Get Data 1	Read <values> on ID3
10	Transfer data	
Back to 4	Write 2nd Base Tag address	Write <addr> to ID 1
	Execute instruction	Write 1 to ID 5
	Check result	Read ID 6
	If result is error set flag	
	Get Data 2	Read <values> on ID3
	Transfer data	
Back to 4	Write 3nd Base Tag address	Write <addr> to ID 1
	Etc	

Sequence for Download (Block WRITE)

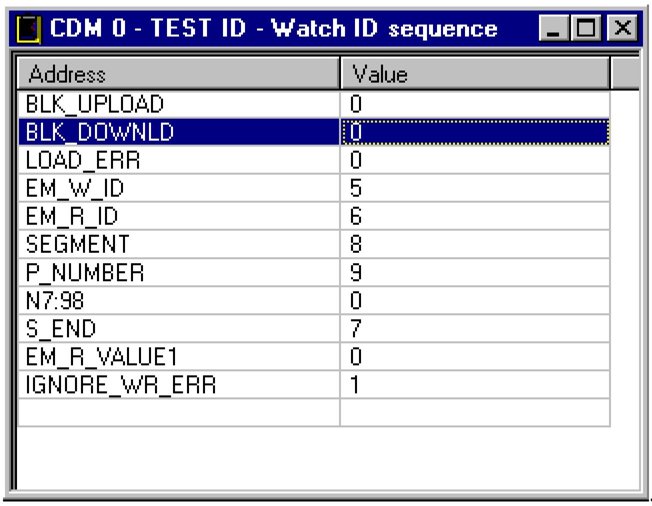
Order	Action.	Value
0	Clear all transactions in Scanner	Write Reset Command to Scanner
1	Clear sequence in 2600	Write 0 to ID 6
2	Set mode to Write	Write 1 to ID 4
3	Write Block Size	Write <size> to ID 2
4	Write 1st Base Tag address	Write <addr> to ID 1
5	Transfer Data 1	Write <values> to ID3
6	Execute instruction	Write 1 to ID 5
7	Check result	Read ID 6
	If result is error set flag	
Back to 4		
	Write 2nd Base Tag address	Write <addr> to ID 1
	Transfer Data 2	Write <values> to ID3
	Execute instruction	Write 1 to ID 5
	Check result	Read ID 6
	If result is error set flag	
	Write 3nd Base Tag address	Write <addr> to ID 1
	Etc	

The 2600 programmer Tag address map is laid out as outlined in the table below.

Start Tag Address	Length	Description
12288	24	Program details
12312	32	Segment 1 details
12328	32	Segment 2 details
12344	32	Segment 3 details
… etc up to a max of 100 segments		

See 2700programAddresses.xls for details

The application only downloads blocks of 16 parameters so the program details are written 16 from the base address and then 16 from the base address + 8. The segments are then each written with 2 blocks of 16.
In the application all the data is mapped directly onto file N9. With this set at a max of 256 elements it allows all the data of a 7 segment program to be uploaded. (Program is 24 words + 7 x 32 segments = 248 words).
The data may then be modified and downloaded. The application is controlled through a Custom Data Monitor.



Set P_NUMBER to the program number that is to be up or downloaded.
(Wait until the new number has been downloaded).

Set S_END to the last segment number required (max 7 in this example)

Set BLK_UPLOAD to upload a program.
(It resets when complete)

Set BLK_DOWNLD to download the program but be sure N9 has valid data.
(It resets when complete).

LOAD_ERR is set if there is a response from the 2600 says that the last sequence was in error. This never occurred in uploads but did occasionally in downloads although the data was written correctly. There are two actions. Firstly set the parameter at Tag address 12753to 1 (instrument must be in configuration mode). This only ever has to be done once. Setting IGNORE_WR_ERROR allows the sequence to continue regardless of the reported error.
The plc program has 3 ladder files – a main ladder (2600) and two subroutines one for explicit message **write** and one for explicit message **read**.
The Program Number can be changed and is immediately written. Then Bit B3:2/0 initiates an upload and B3:2/1 a download. Note that the new data must be ready before invoking the download. Bit words B3:0 and 3:1 are used to control the sequences.
File N7 is used to set up the explicit message requests and receive the responses. Writes start at N7:0 and reads at N7:40.
Timers have been included to introduce a delay between transactions. These may not be necessary but it was found that the scanner occasionally faulted without them.

9.References

http://www.eurotherm.co.uk/
Select: Document Library and Downloads
Use Category 'Devicenet Support'

TAG ADDRESSES

Files: 2600_2700tag.xls, 2600_2700programmer.xls.
Books: Series 2000 Communications Handbook HA 026230

ELECTRONIC DATA SHEETS

Files: 2200.eds 2400.eds 2500.eds 2600.eds 2700.eds PC3kDNS.eds
www.Eurotherm.com

RSNETWORX FILES.

File: Default2700.dnt 2700example2.dnt 2700example3.dnt
DeviceNet22242500default.dnt, DeviceNet2400EX2default.dnt, DeviceNet2500EX2default.dnt.
NOTE: These files are for browsing only – do not try to download.

SLC500 LADDER EXAMPLES.

File 2400Example2.RSS, 2200Examples.RSS, 2700ReadWrite.RSS, 2600PROGRAMMERv2.RSS
NOTE: These files are to show typical applications and as such come with no guarantees. If any part is used in any way in real applications the application will require full functional testing in the situation where it is used.

Versions

CPU SLC500/03	1747-L531 OS302
Scanner : 1747SDN/B	Rev 4.026
Interface: 1770KFD	Rev 1.004
RSLinx	2.30.01
RSNetWorx	2.22.18.0
RSLogix500	4.50.00

The above products are all from Allen-Bradley, a Rockwell Automation Business.
Allen-Bradley Headquarters
1201 South Second Street
Milwaukee
WI 53204 USA

2700	v5.00
2600	v3.00
2500	v3.30
2400	U4.09
2200	U3.03

The products above are from Eurotherm, an Invensys Business
Eurotherm Ltd
Faraday Close
Durrington
BN16 1SJ UK