

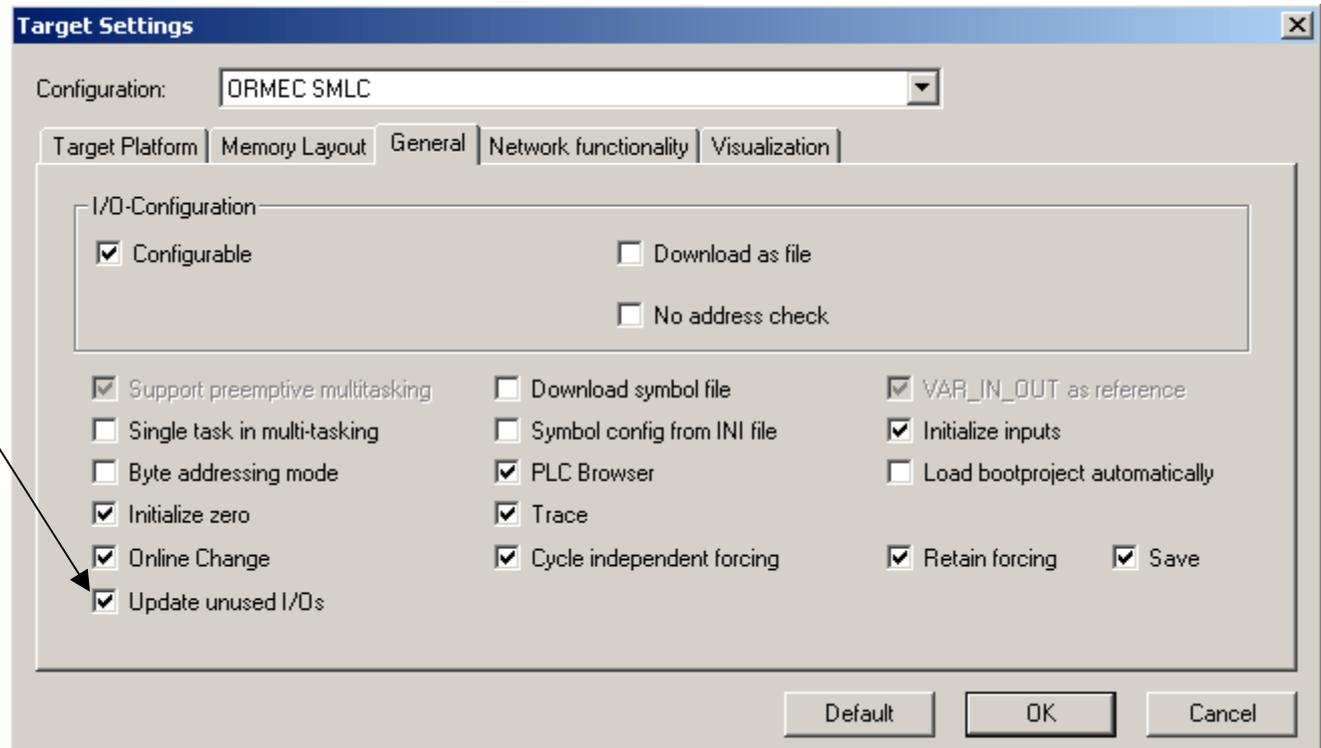
# SMLC PROFIBUS DP Master Tutorial

- PROFIBUS is the world's most popular fieldbus with over 13,000,000 nodes in use world wide.
- There are over 2500 products available from over 400 vendors
- PROFIBUS DP is designed for fast data exchange with devices at the field level
- PROFIBUS DP-V0 provides the basic functionality of DP, including cyclic data exchange, station, module and channel-specific diagnostics.
- PROFIBUS DP-V1 contains enhancements geared toward process automation, in particular acyclic communication for parameter assignment, operation and visualization.
- PROFIBUS supports up to 126 nodes
- Network length can be between 100m and 1200m depending on data rate
- Data rates from 9.6kBits/sec to 12 Mbits/sec.
- PROFIBUS is a Master/Slave network
- A PM option in the SMLC part number indicate that the SMLC is capable of being a PROFIBUS DP Master :
- e.g. SMLC-30-PM-0
- e.g. SMLC-80-PM-0
- e.g. SMLC-160-PM-0



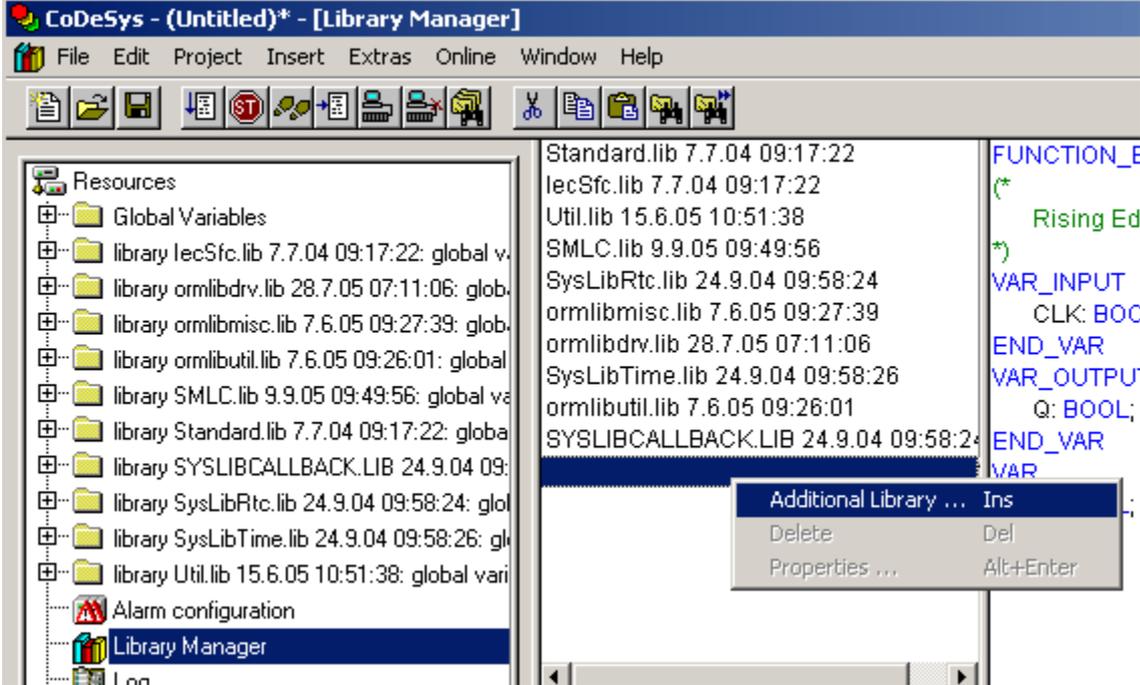
## “Update unused I/Os” target setting

- Create a new project for an SMLC target
- **TIP:** Go to the General page of the target settings and check the “Update unused I/Os” checkbox. This will enable you to view/change the state of I/O even if they are not used in your program.
- **Note:** You must have SMLC firmware v2.0.2 or later installed to use the PROFIBUS capabilities.



# Adding the optional libraries

- Go to the Library Manager and right click to add additional libraries.
- Select BusDiag.lib to access PROFIBUS diagnostic bytes
- Select either SysLibDPV1.lib or SysLibDPV1ex.lib if you want to use DPV1 Read/write function blocks.
- SysLibDPV1ex adds the capability of returning error status from DPV1 reads and writes.
- Adding the BusDiag library is completely optional but may be useful in diagnosing problems.
- The DPV1 libraries are only required if you intend to do DPV1 communications to a slave device.



# BusDiag library

- The BusDiag library allows you to read the diagnostic status bytes of the master as well as from each of the slave devices.
- DiagGetBusState is for reading the Master's diagnostic status bytes
- DiagGetState is for reading the individual slave's diagnostic status bytes.

```

Standard.lib 7.7.04 09:17:22
lecSfc.lib 7.7.04 09:17:22
Util.lib 15.6.05 10:51:38
SMLC.lib 9.9.05 09:49:56
SysLibRtc.lib 24.9.04 09:58:24
ormlibmisc.lib 7.6.05 09:27:39
ormlibdrv.lib 28.7.05 07:11:06
busdiag.lib 7.7.04 09:17:20
SysLibTime.lib 24.9.04 09:58:26
ormlibutil.lib 7.6.05 09:26:01
SYSLIBCALLBACK.LIB 24.9.04 09:58:24
SysLibDPV1ex.lib 10.6.05 10:41:42
  
```

```

(* FB can be used to get the bus state of a master.
   Every bus member has its own diagnostic byte in the extended info *)
FUNCTION_BLOCK DiagGetBusState
VAR_INPUT
  ENABLE : BOOL ; (* Start the diagnosis *)
  DRIVERNAME : POINTER TO STRING ; (* Driver name the get info from *)
  DEVICENUMBER : INT ; (* Device number of same driver *)
END_VAR
VAR_OUTPUT
  READY : BOOL ; (* Ready flag; is set when diagnostic info is available *)
  STATE : INT ; (* Global Bus state:
  BUSSTATE_BUSOK
  BUSSTATE_BUSFAULT
  BUSSTATE_BUSNOTCOMMUNICATING
  BUSSTATE_BUSTOPPED *)
  EXTENDEDINFO : ARRAY[0..129] OF BYTE ; (* Diagnostic Info per Bus member (Slave). Only the first 80 bytes are used
  Bit 0: Bus member is configured
  Bit 1: Bus member is alive
  Bit 2: Bus member reports an error, detailed info can be found in the extended info *)
  
```

DIAGGETBUSSTATE

— ENABLE : BOOL	— READY : BOOL
— DRIVERNAME : POINTER TO STRING(80)	— STATE : INT
— DEVICENUMBER : INT	— EXTENDEDINFO : ARRAY [0..129] OF BYTE

# DPV1 Libraries

- There are two libraries for DPV1 communications: SysLibDPV1.lib and SysLibDPV1ex.lib.
- SysLibDPV1 contains only the DPV1\_Read and DPV1\_Write function blocks
- SysLibDPV1ex adds the DPV1\_ReadEx and DPV1\_WriteEx function blocks
- The “Ex” function blocks add the capability of reading the error status of a DPV1 transaction.
- Only add one of these libraries to prevent compilation errors for duplicate function blocks
- **TIP:** use the SysLibDPV1ex.lib even if you don't intend to read the error status codes.

Standard.lib 7.7.04 09:17:22  
 lecSfc.lib 7.7.04 09:17:22  
 Util.lib 15.6.05 10:51:38  
 SMLC.lib 9.9.05 09:49:56  
 SysLibRtc.lib 24.9.04 09:58:24  
 ormlibmisc.lib 7.6.05 09:27:39  
 ormlibdrv.lib 28.7.05 07:11:06  
 busdiag.lib 7.7.04 09:17:20  
 SysLibTime.lib 24.9.04 09:58:26  
 ormlibutil.lib 7.6.05 09:26:01  
 SYSLIBCALLBACK.LIB 24.9.04 09:58:24  
**SysLibDPV1ex.lib 10.6.05 10:41:42**

```

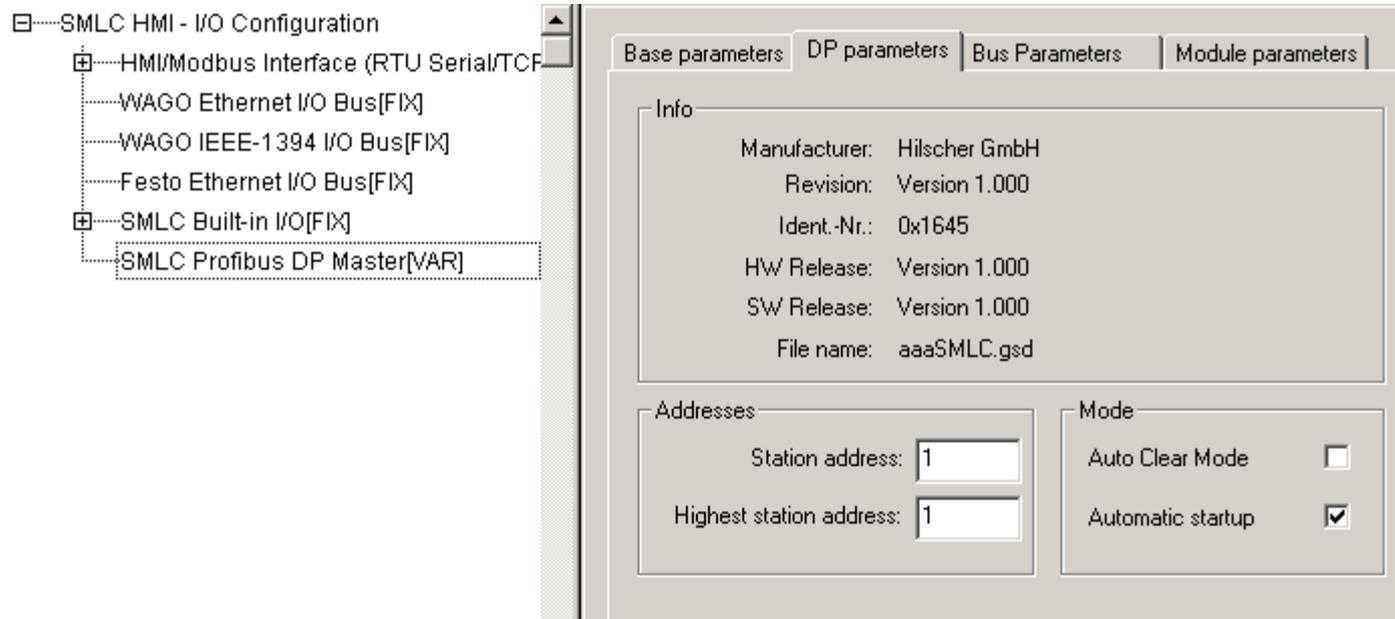
FUNCTION_BLOCK DPV1_Read
VAR_INPUT
    ENABLE:BOOL;
    Device:INT;
    StationAddr:INT;
    Slot:INT;
    Index:INT;
    Len:INT;
    buffer:DWORD;
END_VAR
VAR_OUTPUT
    Ready:BOOL;
    State:V1State;
    Size:INT;
END_VAR
VAR
    ENABLEOLD:BOOL;
    JobId:DWORD;
END_VAR
        
```

POUs

- DPV1\_Read (FB)
- DPV1\_ReadEx (FB)
- DPV1\_Write (FB)
- DPV1\_WriteEx (FB)

# PLC Configuration - DP Parameters

- Go to the PLC Configuration on the Resources tab
- Select SMLC PROFIBUS DP Master.
- Select the DP Parameters tab
- Set the station address for the master.
- Masters must be station 0, 1 or 2
- The default master station address is 1



- The SMLC uses a Hilscher PROFIBUS DP Master card. The .gsd file is completely identical to the Hilscher CIF50-PB and CIF104-PB with the exception of the network node name (SMLC PROFIBUS DP Master)
- Its not necessary to set the Highest station address at this point. CoDeSys will automatically adjust it as you add your slave devices.

## PLC Configuration - Bus Parameters

- Go to the Bus Parameters tab and set the Baud rate
- The most typical baud rate selections are 1500 or 12000
- **TIP:** leave the Use defaults option checked for optimal bus parameter values.

Base parameters | DP parameters | **Bus Parameters** | Module parameters

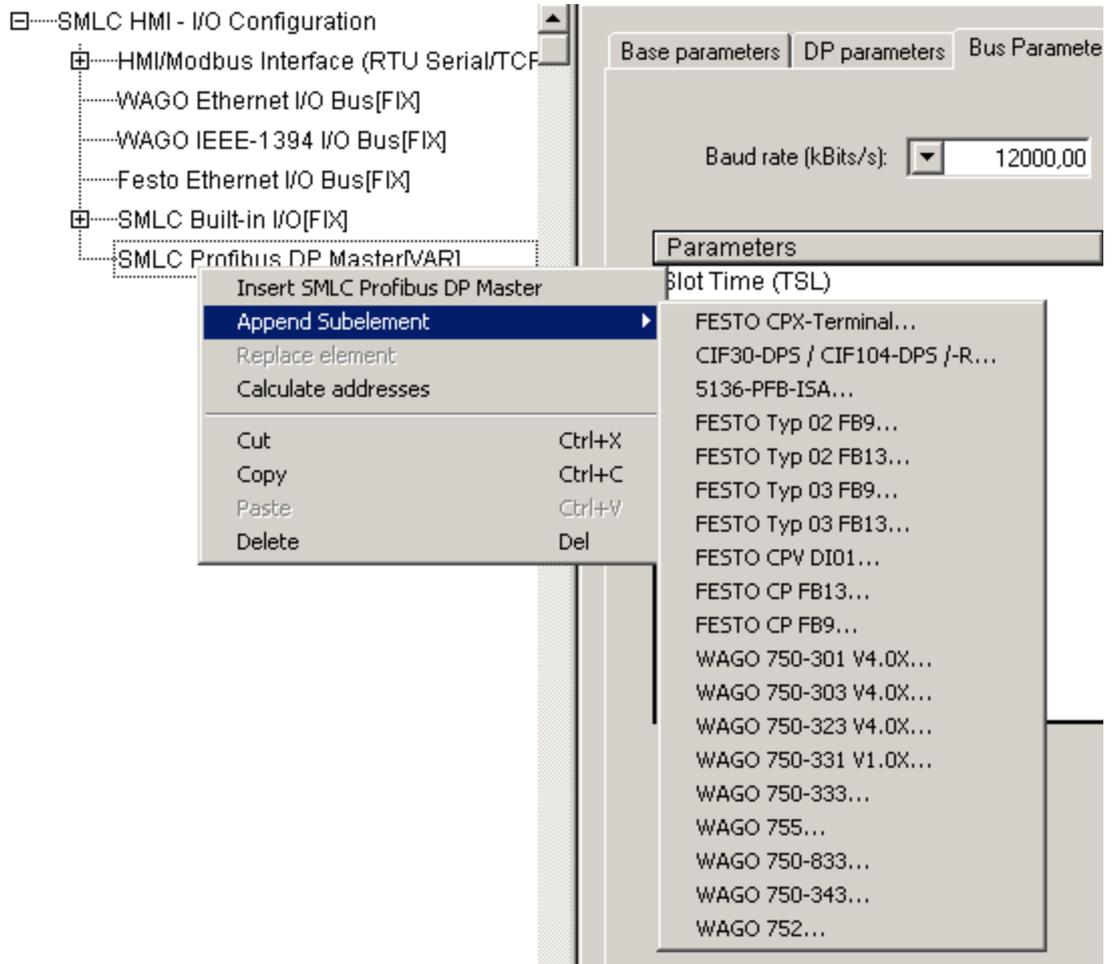
Baud rate (kBits/s):   Use defaults

1500,00 ▲  
3000,00  
6000,00  
12000,00 ▼

Parameters	Value	Unit
Slot Time (TSL)	1000	tBit
Min.Station Delay (min TSDR)	11	tBit
Max.Station Delay (max TSDR)	800	tBit
Quiet Time (TQUI)	9	tBit
Setup Time (TSET)	16	tBit
Target Rotation Time (TTR)	6647	tBit
Gap Update Factor	10	
Max. Retry Limit	4	
Min. Slave Interval	10	100 µs
Poll Timeout	10	10 ms
Data Control Time	1200	ms

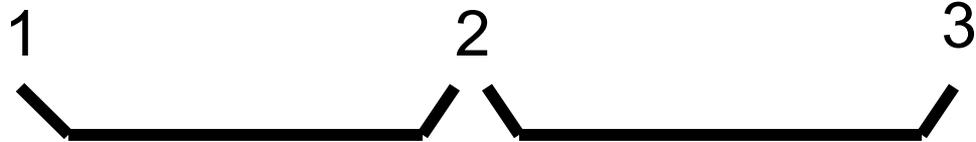
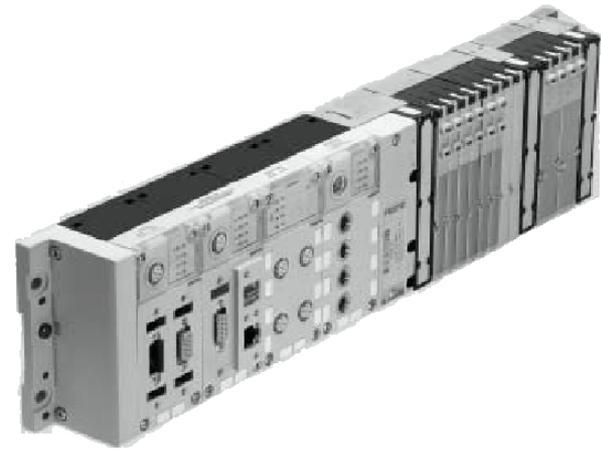
## PLC Configuration - Adding a slave device

- Right click on the SMLC PROFIBUS DP Master and select Append Subelement
- Pick your slave device from the list.
- If you don't see the device you wish to use need to copy the device's .gsd file to the SMLC target directory.
- A .gsd file is the file provided by the I/O manufacturer that contains all of the data regarding the capabilities of the I/O module.
- The SMLC target directory is located in the ORMEC\_SMLC subdirectory of the CoDeSys v2.3\Targets folder. (Typically c:\program files\3s Software\CodeSys v2.3\Targets\ORMEC\_SMLC)
- If you add a new .gsd file you need to quit and restart CoDeSys for it to show up in the list.



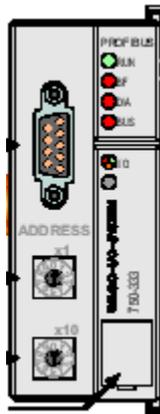
# Our tutorial network

- For the purposes of this tutorial we will create the following network
- SMLC PROFIBUS DP Master, station address 1
- Wago 750-333 bus coupler, station address 2
- FESTO CPX-Terminal, station address 3



## PLC Configuration - Configuring the Wago 750-333

- Select a Wago 750-333 and go to the DP parameters tab
- Set the Station address to 2
- Our Wago bus coupler has the following I/O modules installed
- 750-430 8 DC inputs
- 750-530 8 DC outputs
- The rotary switches on the Wago bus coupler are set for a station address of 2



Base parameters | DP parameters | Input/Output | User parameters | Groups | Module parameters

Info

Manufacturer: WAGO Kontakttechnik GmbH  
Revision: 2.00  
HW Release: HW 03  
SW Release: SW 03  
File name: Wagob754.gsd  
Slave type: 3@WAGO-IO-SYSTEM 750

GSD file...

Identification:

Station address: 2  
Station name: WAGO 750-333

Standard parameter

Identnumber: 0xB754  
TSDR (TBit): 11  
Lock/Unlock: 2

Activation

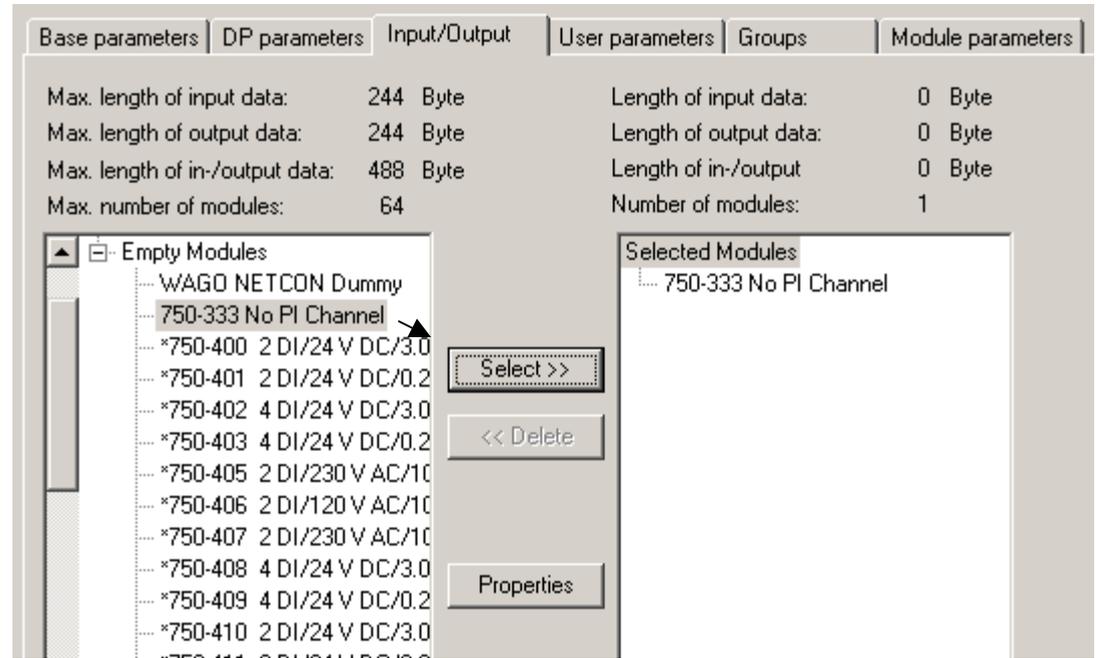
Slave active in current configuration:

Watchdog

Watchdog Control:   
Time (ms): 1000

## PLC Configuration - Configuring the Wago 750-333

- Select the Input/Output tab
- Here is where we select the modules that are installed on our bus coupler.
- Now is a good time to read the documentation that came with your slave device. If the configuration doesn't match exactly the slave will not go online.
- The Wago documentation tells us that we need to insert an Empty Module - "No PI channel" as the first module on our slave. If we don't do this we will get a configuration error!



# PLC Configuration - Adding the input module

- Go to the input modules section of the tree
- Click on the 750-430 and hit the Select button
- If you then click on the 750-430 module in the right hand pane and click on the Properties button you can set the individual properties for the module. For this module the only option is whether it is physically plugged (present) or not.

Base parameters | DP parameters | **Input/Output** | User parameters | Groups | Module parameters

Max. length of input data:	244 Byte	Length of input data:	1 Byte
Max. length of output data:	244 Byte	Length of output data:	0 Byte
Max. length of in-/output data:	488 Byte	Length of in-/output:	1 Byte
Max. number of modules:	64	Number of modules:	2

Selected Modules

- 750-333 No PI Channel
- 750-430 8 DI/24 V DC/3.0 ms**

Buttons: Select >>, << Delete, Properties

**Module Properties**

Name: 750-430 8 DI/24 V DC/3.0 ms  
 Config: 0x10  
 Length input (Byte): 1  
 Length output (Byte): 0  
 Symbolic names:

Parameters	Value	Allowed Values
"Terminal is physically"	plugged	Bit(5) 1 0-1

# PLC Configuration - Adding the output module

- Go to the Output Modules section of the tree and select a 750-530 module.
- For this module we can see that there are many properties that can be set, including the state for each output should the PROFIBUS network fail. Refer to the Wago documentation for more information.

Base parameters | DP parameters | **Input/Output** | User parameters | Groups | Module parameters

Max. length of input data:	244 Byte	Length of input data:	1 Byte
Max. length of output data:	244 Byte	Length of output data:	1 Byte
Max. length of in-/output data:	488 Byte	Length of in-/output:	2 Byte
Max. number of modules:	64	Number of modules:	3

Selected Modules

- 750-333 No PI Channel
- 750-430 8 DI/24 V DC/3.0 ms
- 750-530 8 DO/24 V DC/0.5 A**

Buttons: Select >>, << Delete, Properties

**Module Properties**

Name: 750-530 8 DO/24 V DC/0.5 A  
 Config: 0x20  
 Length input (Byte): 0  
 Length output (Byte): 1  
 Symbolic names:

Parameters	Value	Allowed Values
"Terminal is physically"	plugged	Bit(5) 1 0-1
"Substitute Value Channel 1"	0	Bit(0) 0 0-1
"Substitute Value Channel 2"	0	Bit(1) 0 0-1
"Substitute Value Channel 3"	0	Bit(2) 0 0-1
"Substitute Value Channel 4"	0	Bit(3) 0 0-1
"Substitute Value Channel 5"	0	Bit(4) 0 0-1
"Substitute Value Channel 6"	0	Bit(5) 0 0-1
"Substitute Value Channel 7"	0	Bit(6) 0 0-1
"Substitute Value Channel 8"	0	Bit(7) 0 0-1

## PLC Configuration - Verify Wago configuration

- At this point your configuration should look like this.
- If the order of the modules does not match exactly the physical configuration then you need to fix it before proceeding. If necessary delete and re-select modules to get them into the proper position.
- **TIP** - If you want to append a module click on “Selected Modules” before clicking the Select button. If you want to insert a module click on the module you want to insert before and then click on Select.

The screenshot shows the 'Input/Output' configuration tab in the Wago software. It is divided into two columns of parameters:

Parameter	Value	Parameter	Value
Max. length of input data:	244 Byte	Length of input data:	1 Byte
Max. length of output data:	244 Byte	Length of output data:	1 Byte
Max. length of in-/output data:	488 Byte	Length of in-/output:	2 Byte
Max. number of modules:	64	Number of modules:	3

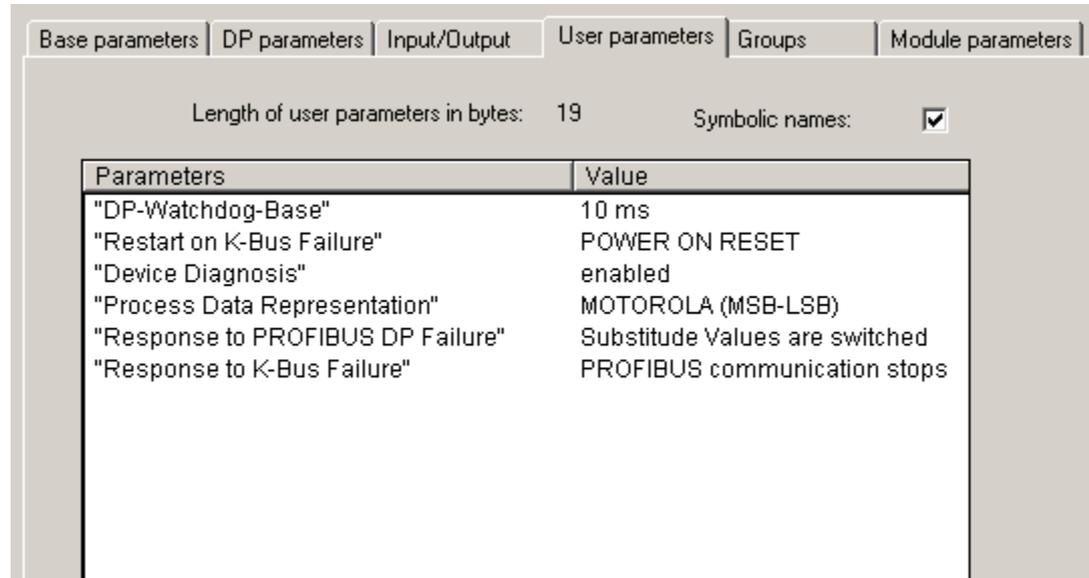
Below the parameters, there are two lists of modules:

- Output Modules:**
  - 750-501 2 DO/24 V DC/0.5
  - 750-502 2 DO/24 V DC/2.0
  - 750-504 4 DO/24 V DC/0.5
  - 750-506 2 DO/24 V DC/0.5
  - 750-507 2 DO/24 V DC/2.0
  - 750-509 2 DO/230 V AC/0.
  - 750-512 2 DO Relay/250 V
  - 750-513 2 DO Relay/250 V
  - 750-514 2 DO Relay/125 V
  - 750-516 4 DO/24 V DC/0.5
  - 750-517 2 DO Relay/230 V
  - 750-519 4 DO/5 V DC/20 n
- Selected Modules:**
  - 750-333 No PI Channel
  - 750-430 8 DI/24 V DC/3.0 ms
  - 750-530 8 DO/24 V DC/0.5 A

Buttons for 'Select >>', '<< Delete', and 'Properties' are located between the two module lists.

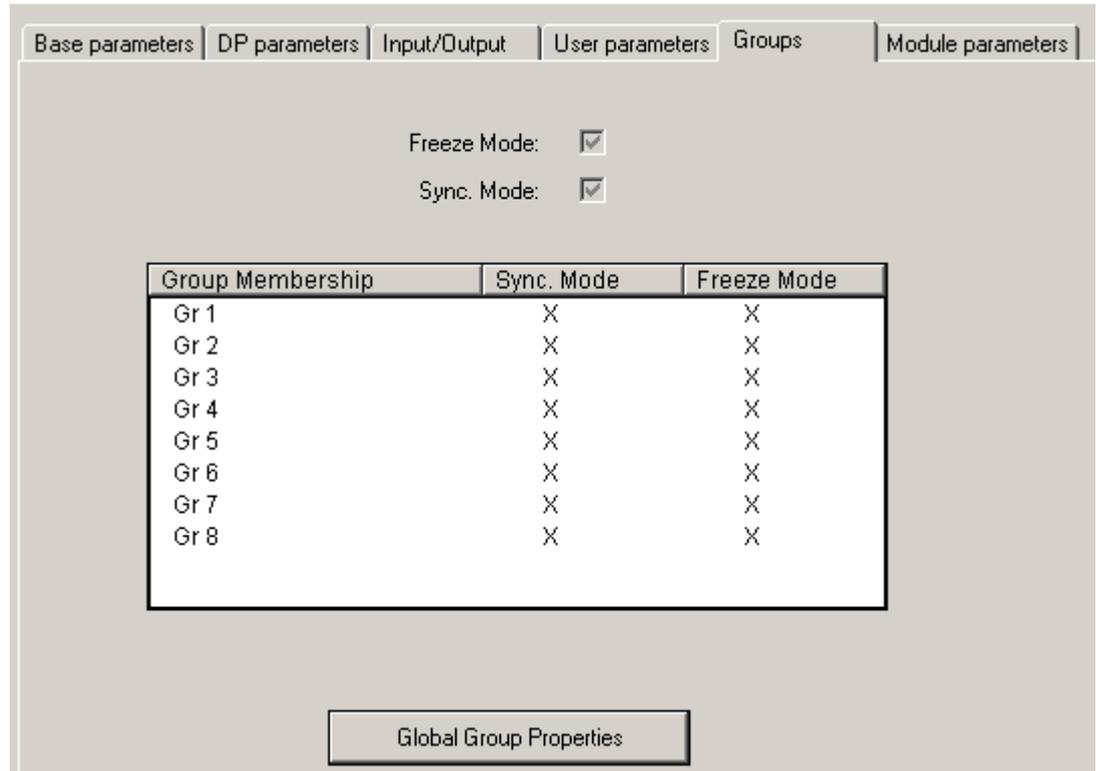
## PLC Configuration - User parameters for the Wago node

- Go to the User parameters tab
- Here we see the settings for the bus coupler itself.
- To change a value double click on it. The values will toggle through all of their possible settings.



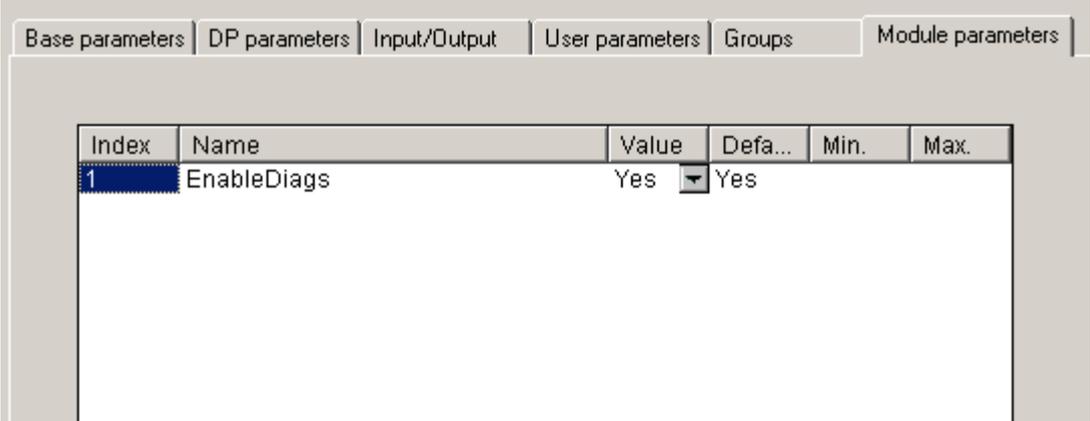
## PLC Configuration - Group parameters for the Wago node

- Go to the Groups tab
- Here you can set the group membership and Sync/Freeze modes for this node.
- This topic is beyond the scope of this tutorial.



## PLC Configuration - Module parameters for the Wago node

- Go to the Module parameters tab
- Here we can set the Module parameters for this node.
- **TIP:** Leave the default setting EnableDiags = Yes. This allows you to retrieve diagnostic information from this module.



The screenshot shows a software interface with several tabs: 'Base parameters', 'DP parameters', 'Input/Output', 'User parameters', 'Groups', and 'Module parameters'. The 'Module parameters' tab is active, displaying a table with the following data:

Index	Name	Value	Defa...	Min.	Max.
1	EnableDiags	Yes	Yes		

## PLC Configuration - Configuring the FESTO CPX-Terminal

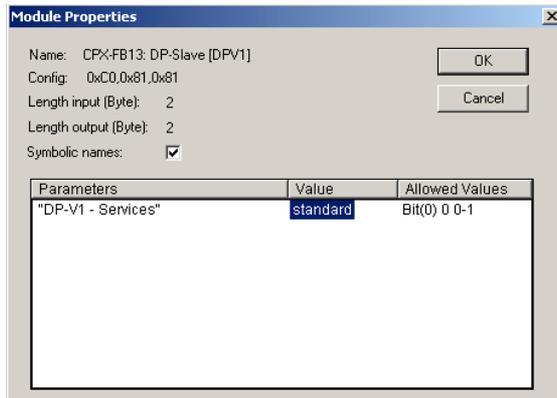
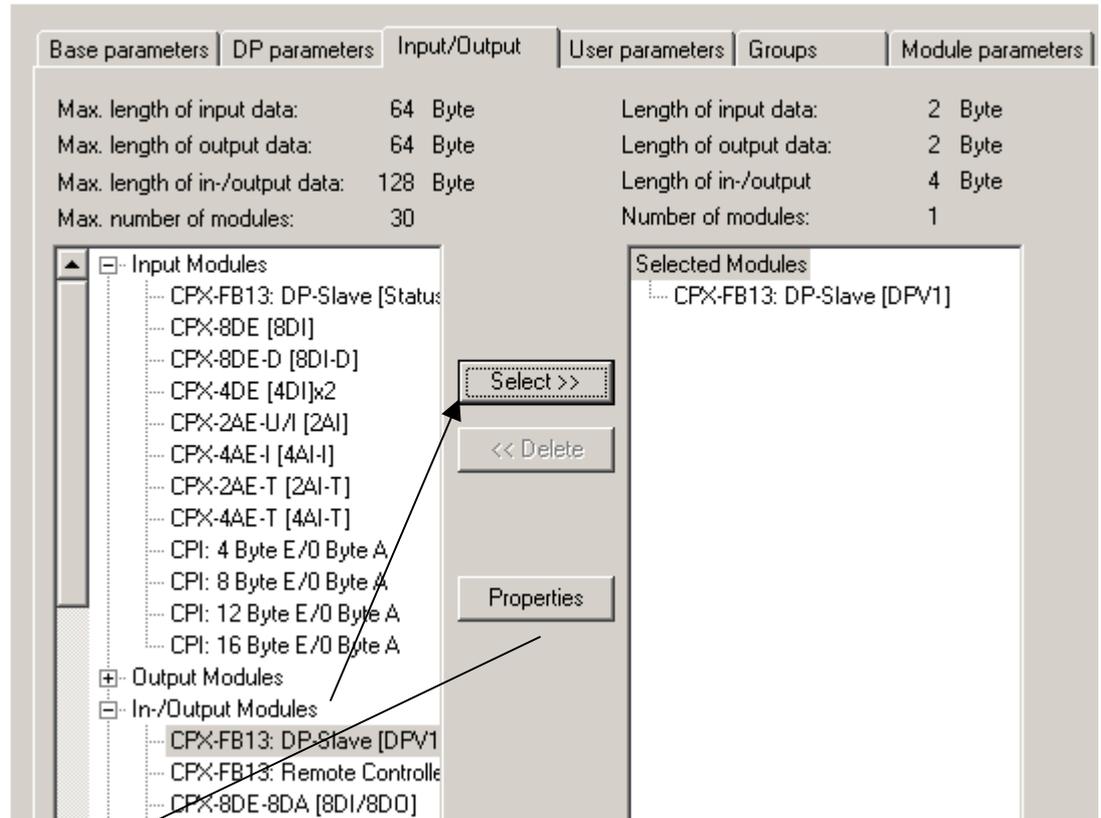
- Select a FESTO CPX Terminal and go to the DP parameters tab.
- Set the Station address to 3
- In our setup the CPX has the following modules:
- CPX-8DE-8DA (8DI/8DO)
- CPX-2AE-U/I (2 analog inputs)
- CPX-2AA-U/I (2 analog outputs)
- MPA1S (valve bank w/ 4 valves)

The screenshot displays the configuration interface for a FESTO CPX-Terminal, specifically the 'DP parameters' tab. The interface is organized into several sections:

- Base parameters:** Includes tabs for 'Base parameters', 'DP parameters', 'Input/Output', 'User parameters', 'Groups', and 'Module parameters'. The 'DP parameters' tab is currently selected.
- Info:** A box containing device details: Manufacturer: FESTO AG Co., Revision: 09, HW Release: 02.02, SW Release: V2.9, File name: CPX\_059E.GSE, and Slave type: 4. A 'GSD file...' button is located to the right.
- Identification:** Contains fields for 'Station address' (set to 3) and 'Station name' (set to 'FESTO CPX-Terminal').
- Activation:** A checkbox labeled 'Slave active in current configuration' is checked.
- Standard parameter:** Contains fields for 'Identnumber' (0x059E), 'TSDR (TBit)' (11), and 'Lock/Unlock' (2).
- Watchdog:** A checkbox labeled 'Watchdog Control' is checked, and a 'Time (ms)' field is set to 1000.

# PLC Configuration - Configuring the FESTO CPX-Terminal

- Go to the Input/Output tab
- From reading the documentation for the FESTO CPX we know that we can operation this device in one of three ways:
- DP Slave (no status), DP Slave (status), DP Slave (DPV1 enabled)
- For this example we want to use the DPV1 functions so under the In/Output Modules tree select CPX-FB13: DP-Slave [DPV1]
- Select the Properties and verify the value Standard.



# PLC Configuration - Adding the 8DI/8DO module

- Next add the 8DI/8DO module
- Click on the module in right pane after insertion and then click on the Properties button to see all of the options for this module
- **TIP** - after viewing the properties click on Selected Modules before inserting the next module or the new module may be inserted rather than appended.

Base parameters	DP parameters	Input/Output	User parameters	Groups	Module parameters
Max. length of input data:	64 Byte		Length of input data:		3 Byte
Max. length of output data:	64 Byte		Length of output data:		3 Byte
Max. length of in-/output data:	128 Byte		Length of in-/output		6 Byte
Max. number of modules:	30		Number of modules:		2

Parameters	Value
"Monitor SCS"	enabled
"Monitor SCO"	enabled
"Monitor VoutWval"	enabled
"Behaviour after SCS"	Vsen switch on again
"Behaviour after SCO"	Vout switch on again
"Debounce time"	3.0 ms
"Signal extension time"	15 ms
"Channel 0: Signal extension"	disabled
"Channel 1: Signal extension"	disabled

# PLC Configuration - Adding the Analog input module

- Next add the 2AI module
- Click on the module in right pane after insertion and then click on the Properties button to see all of the options for this module

**Module Properties**

Name: CPX-2AE-U/I [2AI] OK

Config: 0x51 Cancel

Length input (Byte): 4

Length output (Byte): 0

Symbolic names:

Parameters	Value	Allow
"Monitor SCS"	enabled	Bit(0)
"Monitor parameters"	enabled	Bit(7)
"Behaviour after SCS"	Vsen switch on again	Bit(0)
"Input format"	Sign+12bit right side	BitAre
"Channel 0: Monitor lower limit"	disabled	Bit(0)
"Channel 0: Monitor upper limit"	disabled	Bit(1)
"Channel 0: Monitor wire fracture"	disabled	Bit(2)
"Channel 0: Monitor parameters"	enabled	Bit(7)
"Channel 1: Monitor lower limit"	disabled	Bit(0)

Base parameters | DP parameters | **Input/Output** | User parameters | Groups | Module parameters

Max. length of input data: 64 Byte      Length of input data: 7 Byte

Max. length of output data: 64 Byte      Length of output data: 3 Byte

Max. length of in-/output data: 128 Byte      Length of in-/output: 10 Byte

Max. number of modules: 30      Number of modules: 3

**Input Modules**

- CPX-FB13: DP-Slave [Status]
- CPX-8DE [8DI]
- CPX-8DE-D [8DI-D]
- CPX-4DE [4DI]x2
- CPX-2AE-U/I [2AI]**
- CPX-4AE-I [4AI-I]
- CPX-2AE-T [2AI-T]
- CPX-4AE-T [4AI-T]
- CPI: 4 Byte E/0 Byte A
- CPI: 8 Byte E/0 Byte A
- CPI: 12 Byte E/0 Byte A

Select >>

<< Delete

Properties

**Selected Modules**

- CPX-FB13: DP-Slave [DPV1]
- CPX-8DE-8DA [8DI/8DO]
- CPX-2AE-U/I [2AI]

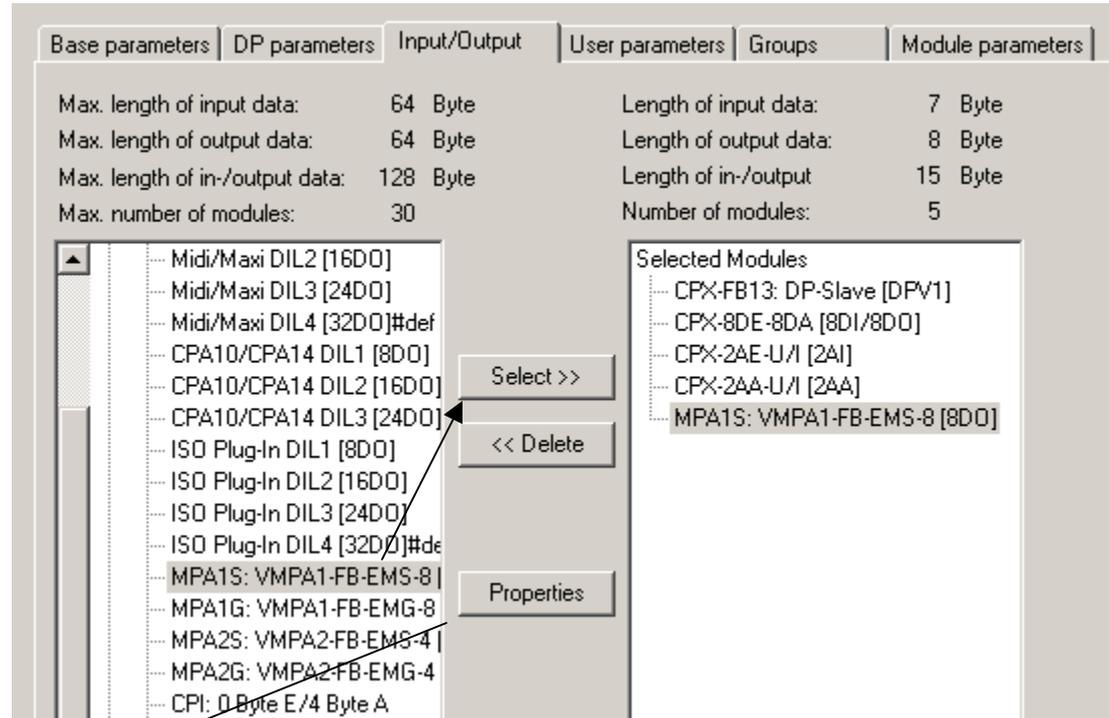
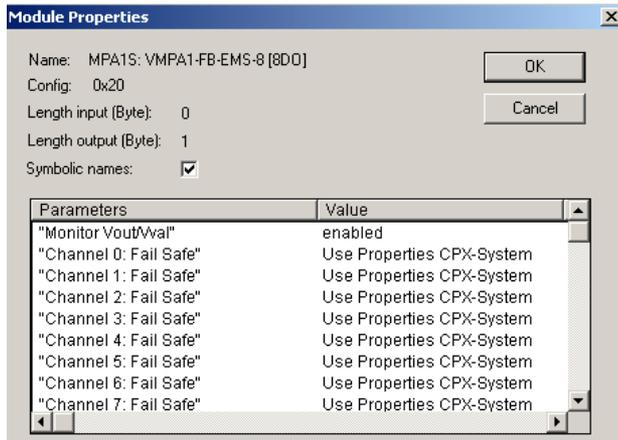
# PLC Configuration - Adding the Analog output module

- Next add the 2AO module
- Click on the module in right pane after insertion and then click on the Properties button to see all of the options for this module

Parameters	Value
"Monitor SCO"	enabled
"Monitor parameters"	enabled
"Behaviour after SCO signal"	Signal remains switched off
"Behaviour after SCO"	Vout switch on again
"Output format"	Sign+12bit right side
"Channel 0: Lower limit active"	enabled
"Channel 0: Upper limit active"	enabled
"Channel 0: Overload/short circuit"	enabled
"Channel 0: Monitor wire fracture"	disabled

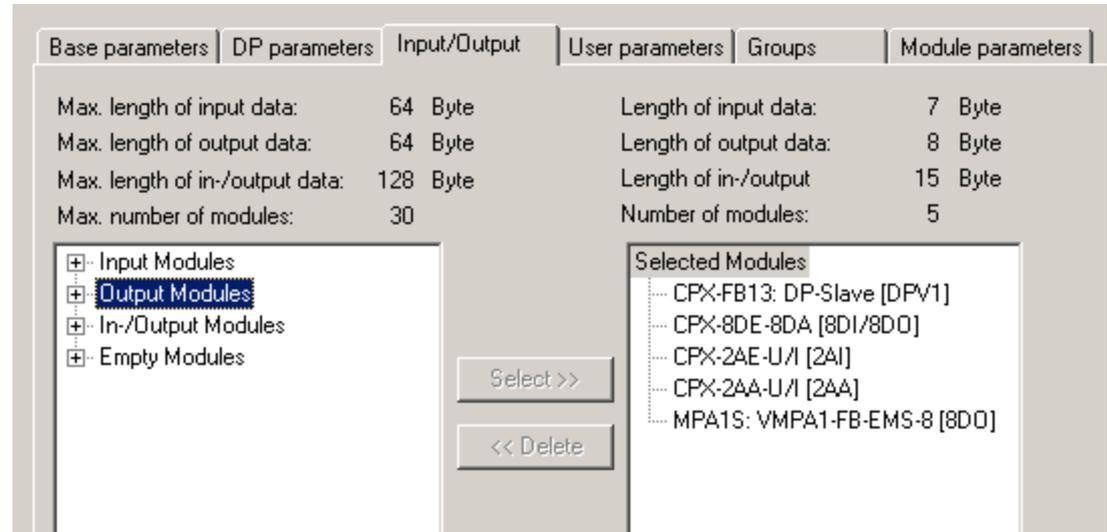
# PLC Configuration - Adding the valve bank

- Next add the MPA1S module
- Click on the module in right pane after insertion and then click on the Properties button to see all of the options for this module



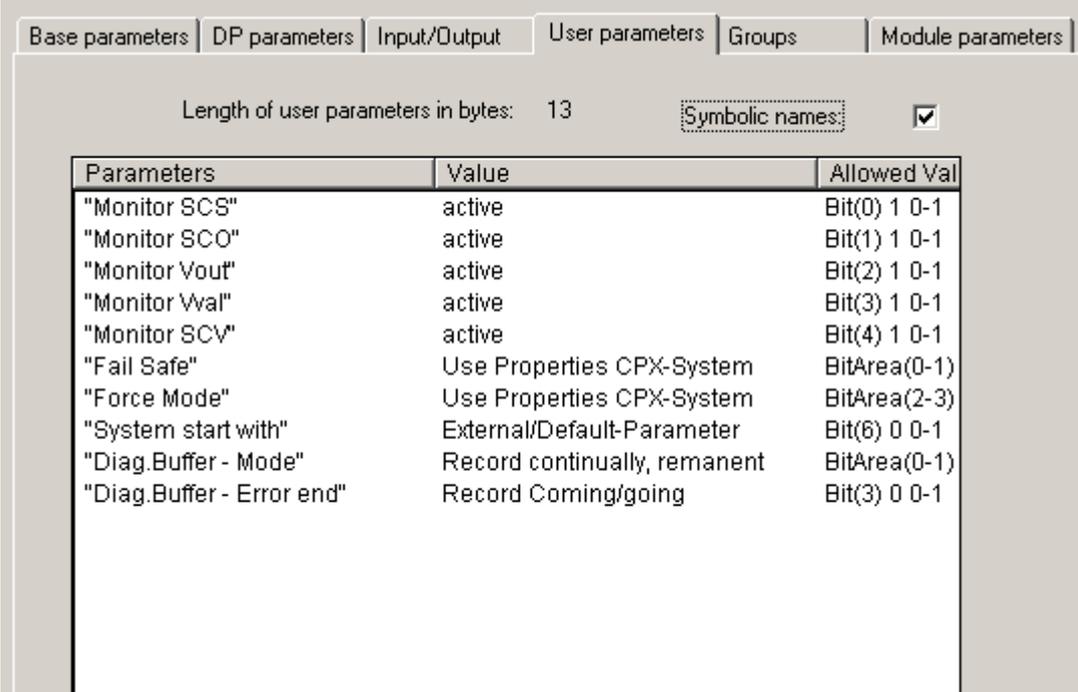
## PLC Configuration - Verify the FESTO configuration

- At this point your configuration should look like this.
- If the order of the modules does not match exactly the physical configuration then you need to fix it before proceeding. If necessary delete and re-select modules to get them into the proper position.
- **TIP** - If you want to append a module click on “Selected Modules” before clicking the Select button. If you want to insert a module click on the module you want to insert before and then click on Select.



## PLC Configuration - User parameters for the FESTO node

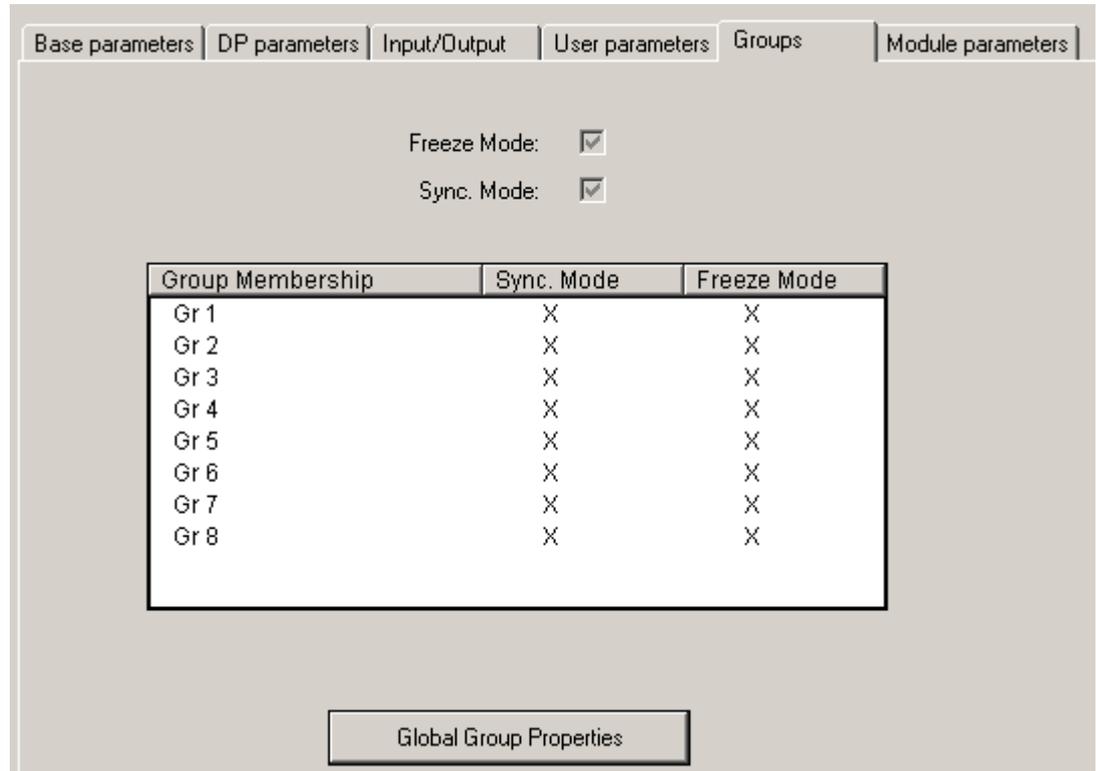
- Go to the User parameters tab
- Here we see the settings for the bus coupler itself.
- To change a value double click on it. The values will toggle through all of their possible settings.



Parameters	Value	Allowed Val
"Monitor SCS"	active	Bit(0) 1 0-1
"Monitor SCO"	active	Bit(1) 1 0-1
"Monitor Vout"	active	Bit(2) 1 0-1
"Monitor Vval"	active	Bit(3) 1 0-1
"Monitor SCV"	active	Bit(4) 1 0-1
"Fail Safe"	Use Properties CPX-System	BitArea(0-1)
"Force Mode"	Use Properties CPX-System	BitArea(2-3)
"System start with"	External/Default-Parameter	Bit(6) 0 0-1
"Diag.Buffer - Mode"	Record continually, remanent	BitArea(0-1)
"Diag.Buffer - Error end"	Record Coming/going	Bit(3) 0 0-1

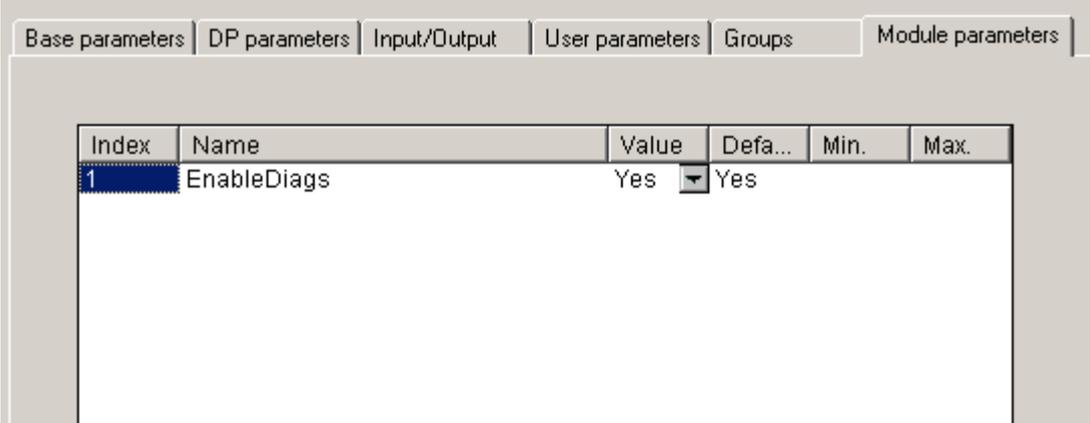
## PLC Configuration - Group parameters for the FESTO node

- Go to the Groups tab
- Here you can set the group membership and Sync/Freeze modes for this node.
- This topic is beyond the scope of this tutorial.



## PLC Configuration - Module parameters for the FESTO node

- Go to the Module parameters tab
- Here we can set the Module parameters for this node.
- **TIP:** Leave the default setting EnableDiags = Yes. This allows you to retrieve diagnostic information from this module.

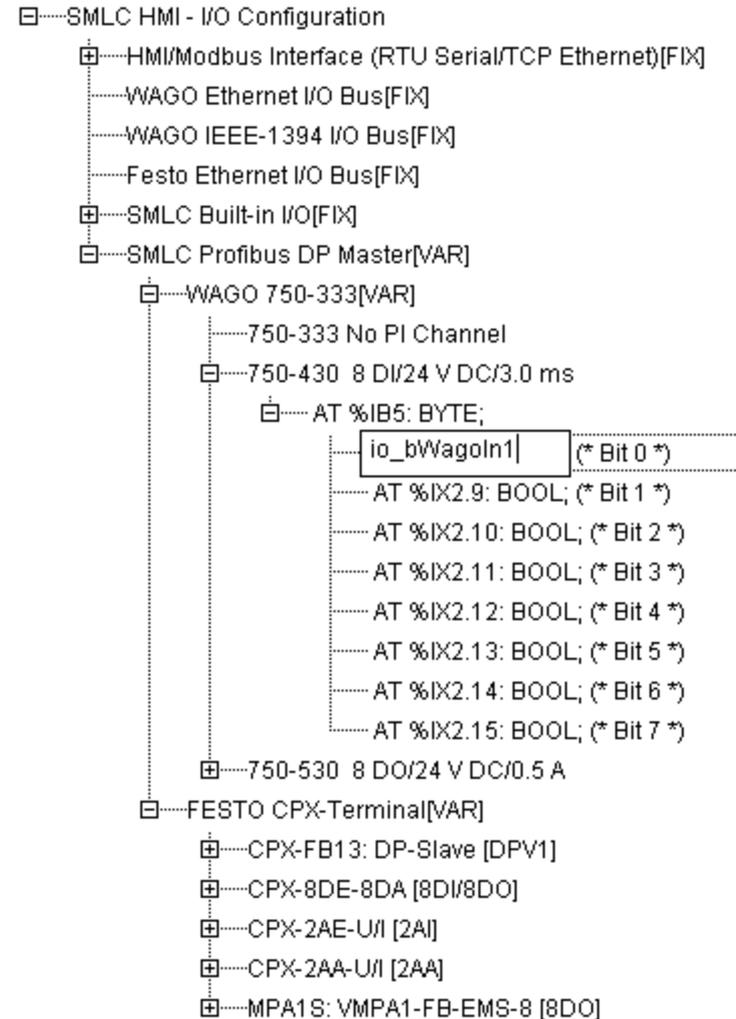


The screenshot shows a software interface with several tabs: 'Base parameters', 'DP parameters', 'Input/Output', 'User parameters', 'Groups', and 'Module parameters'. The 'Module parameters' tab is active, displaying a table with the following data:

Index	Name	Value	Defa...	Min.	Max.
1	EnableDiags	Yes	Yes		

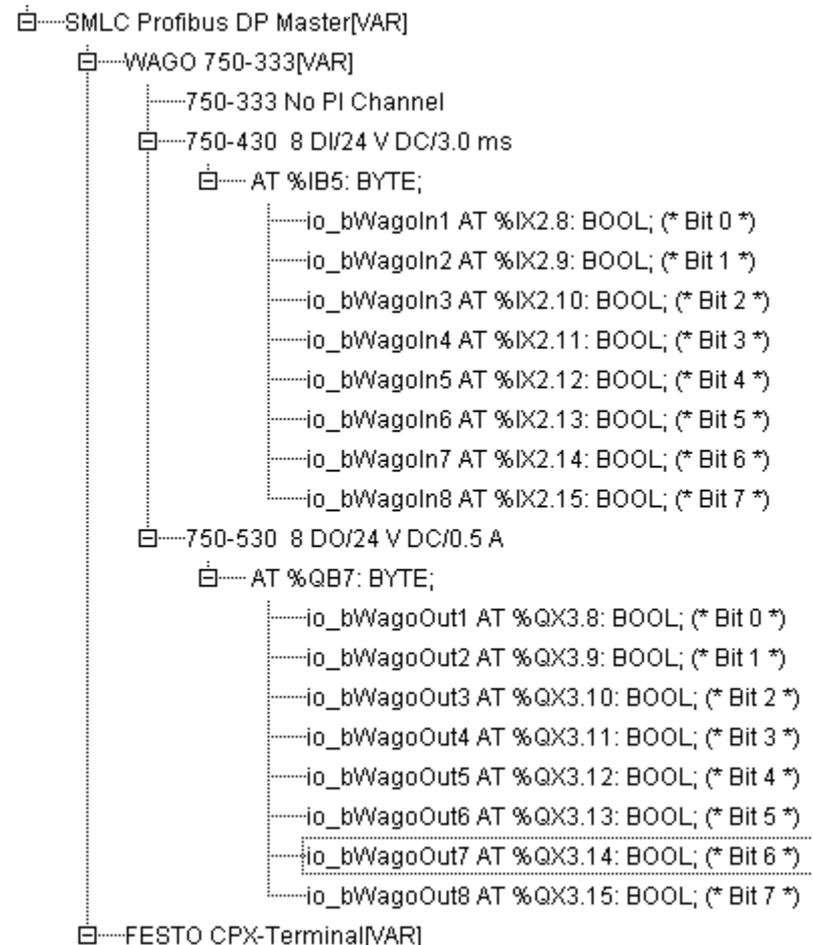
# PLC Configuration - Assigning the I/O variable names

- In the PLC Configuration tree expand the Wago 750-430 module. We see that there is one byte of data. Expand the byte and we can see the individual I/O bits on this module.
- Double click on that AT for Bit 0 and enter the I/O point's name.
- This automatically creates a global variable in your program, allowing you to access this input.
- **TIP** - use a naming convention to help you identify physical I/O point amongst your program variables. We like to use the io\_ prefix to indicate that it is a physical I/O point. The b indicates that it is a boolean (or bit) value. The rest of the name should provide a meaningful description of the device connected to this point. E.g. io\_bJogFwdPB.



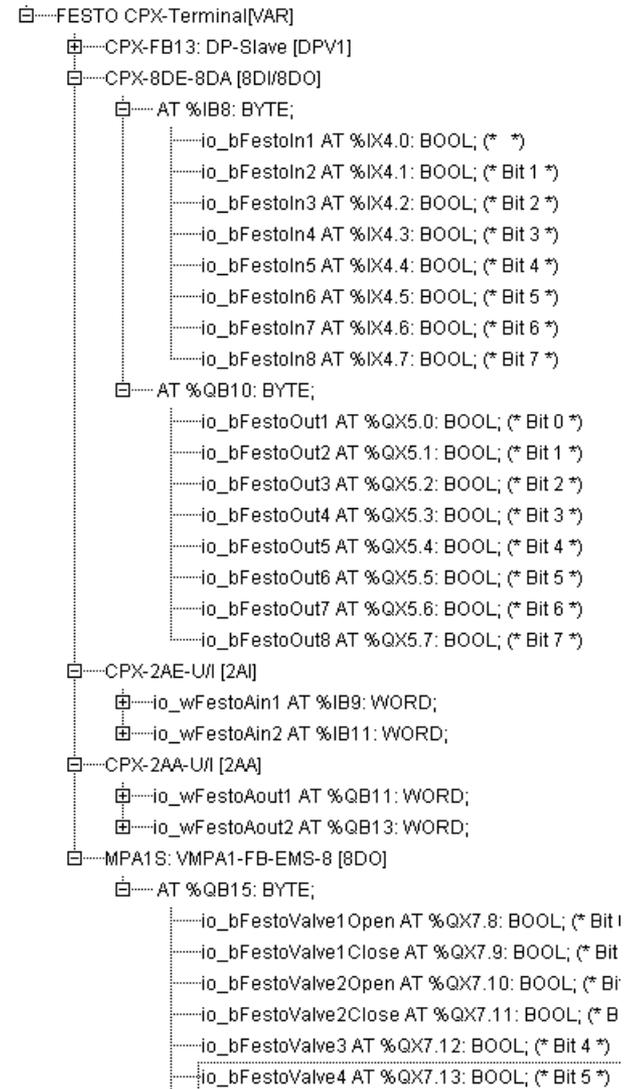
# PLC Configuration - Assigning the I/O variable names

- Fill in the rest of the Wago I/O point names.



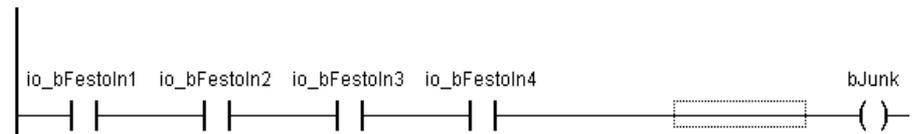
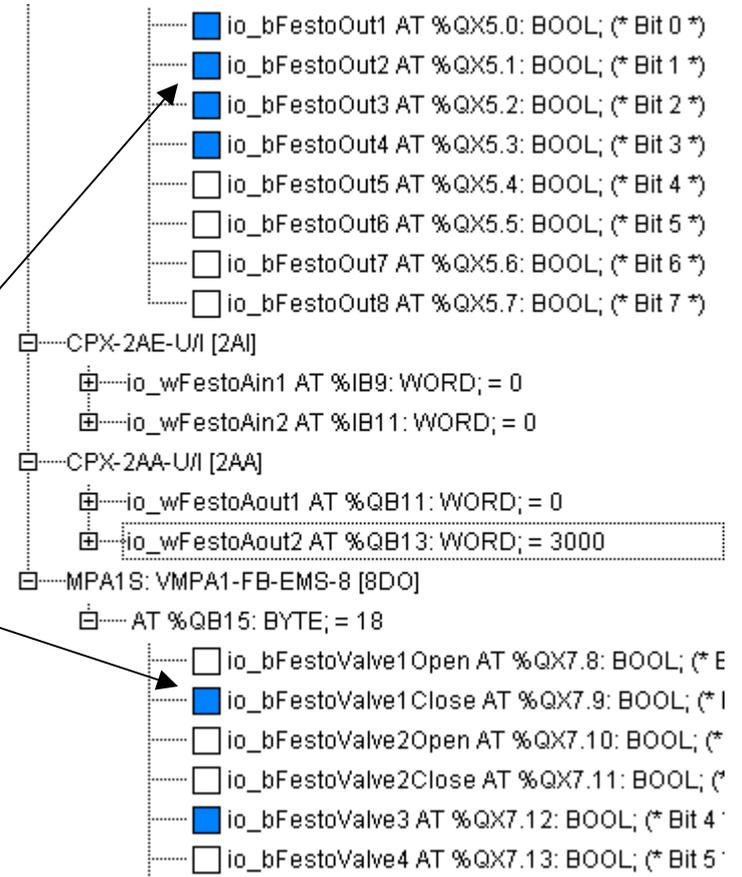
# PLC Configuration - Assigning the I/O variable names

- Fill in the Festo I/O point names.
- Note that the analog inputs and outputs are Words
- Note that our valve bank has two two-way valves and two single way valves. The two way valves use one output to open the valve and one output to close the valve. The single way valves are open when on, closed when off.
- Always refer to the vendor's documentation to understand how the bits, bytes and words are mapped to a particular device.



# PLC Configuration - Testing the Configuration

- At this point you should be able to log in to your SMLC and download this I/O configuration.
- If the bus couplers have been powered up you should see Bus Fault indications before you have downloaded the program. If you have configured everything correctly the Bus Fault indicators should go out after the program has been downloaded. If the Bus Fault indicator do not go out then proceed to the troubleshooting section of this tutorial.
- If the Bus Fault indicators did go out then you can proceed to test your I/O. Go to the PLC Configuration tree and toggle the Output points by clicking in the square before each point (or click to enter an analog value).
- Note:** the inputs will not be updated unless you are actually using them in your program or you have checked the "Update unused I/Os" option on the General tab of the Target Settings.
- Note:** in version 2.0.2 of the SMLC firmware the unused inputs may not update correctly even if you have checked the box on the target settings if you have more than one PROFIBUS node. If this is the case, uncheck the check box and "use" the inputs in your program, even in "junk" logic. This will allow the inputs to update correctly on the PLC Configuration screen.

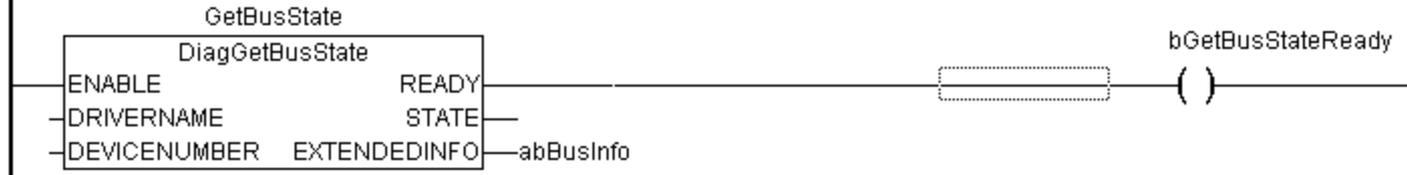


# Troubleshooting the network

- The first steps in troubleshooting a PROFIBUS network are to check the obvious things
- Are all nodes powered up? Plugged into the network?
- Is the network properly terminated (terminator turned on only at the ends of the network)?
- Are the station IDs properly assigned to match the nodes?
- Have you created the configuration properly? Do all of the modules match the configuration perfectly? Did you include the correct modules for the node itself (the No PI Channel for Wago, the Status/No Status/DPV1 for the FESTO CPX).
- Try powering down all of the nodes and doing a clean all | rebuild all in CoDeSys. Download this and power all of the nodes back up.
- The I/O modules may provide status information that can direct you to the problem. For example, Wago bus couplers will flash an error code (blink code) that can pinpoint specific problems. Refer to the Diagnosis section of the Wago Profibus Bus Coupler manual (provided on the CDS-SDK CD in the Wago documentation folder).

## Troubleshooting the network using the BusDiag library

This is an example of using the DiagGetBusState function from BusDiag.lib. The ExtendedInfo output returns an array of bytes that indicate the status of the bus



- If none of the previous trouble shooting tips worked its time to use the BusDiag function blocks.
- Insert the DiagGetBusState function block.
- It is not necessary to fill in the DRIVERNAME or DEVICENUMBER inputs
- The EXTENDEDINFO output is an array of 130 bytes declared like this:  
`abBusInfo: ARRAY [0..129] OF BYTE;`
- Run the program and look at the contents of the abBusInfo array.
- abBusInfo is a map of the network showing the status of each slave device. The slave's network ID is the offset into the array. In this example device 2 is the Wago 750-333 and its status is 7. The FESTO CPX is node 3 and its status is 3.

```

E-abBusInfo
-----
abBusInfo[0] = 0
abBusInfo[1] = 0
abBusInfo[2] = 7
abBusInfo[3] = 3
abBusInfo[4] = 0
abBusInfo[5] = 0
abBusInfo[6] = 0
  
```

# Troubleshooting the network using the BusDiag library

This is an example of using the DiagGetState function from BusDiag.lib. The ExtendedInfo output returns an array of bytes that indicate the status of the node. You don't need to fill in the DriverName or DeviceNumber inputs.



- Insert the DiagGetState function block.
- It is not necessary to fill in the DRIVERNAME or DEVICENUMBER inputs
- The BUSMEMBERID is the station address of the slave whose status we wish to read.
- The EXTENDEDINFO output is an array of 100 bytes declared like this:  
abBusInfo: ARRAY [0..99] OF BYTE;
- Run the program and look at the contents of the abExtendedInfo array.
- Refer to the vendors documentation to determine the meaning of the status bytes.

```

abExtendedInfo
-----
.....abExtendedInfo[0] = 0
.....abExtendedInfo[1] = 12
.....abExtendedInfo[2] = 0
.....abExtendedInfo[3] = 1
.....abExtendedInfo[4] = 5
.....abExtendedInfo[5] = 158
.....abExtendedInfo[6] = 0
.....abExtendedInfo[7] = 0
.....abExtendedInfo[8] = 0
.....abExtendedInfo[9] = 0
.....abExtendedInfo[10] = 0
.....abExtendedInfo[11] = 0
.....abExtendedInfo[12] = 0
    
```

# DPV1 Communications

- Recall that we inserted the SysLibDPV1ex.lib earlier (if you didn't do this then now is the time).
- This library provides the DPV1\_Read and DPV1\_Write function blocks
- The Ex versions add error status
- DPV1 provides asynchronous read/write capability between devices that support it.
- Not all PROFIBUS slaves support DPV1 communications. Refer to the vendors documentation be sure.
- DPV1 uses the concept of slots. These slots may correspond to physical modules/slots or virtual.
- Each slot contains data, accessible by an index

The screenshot shows a software interface with a file explorer on the left and a function block definition on the right. The file explorer lists several libraries, with 'SysLibDPV1ex.lib 10.6.05 10:4' selected. The function block definition for 'FUNCTION\_BLOCK DPV1\_Read' is shown as follows:

```

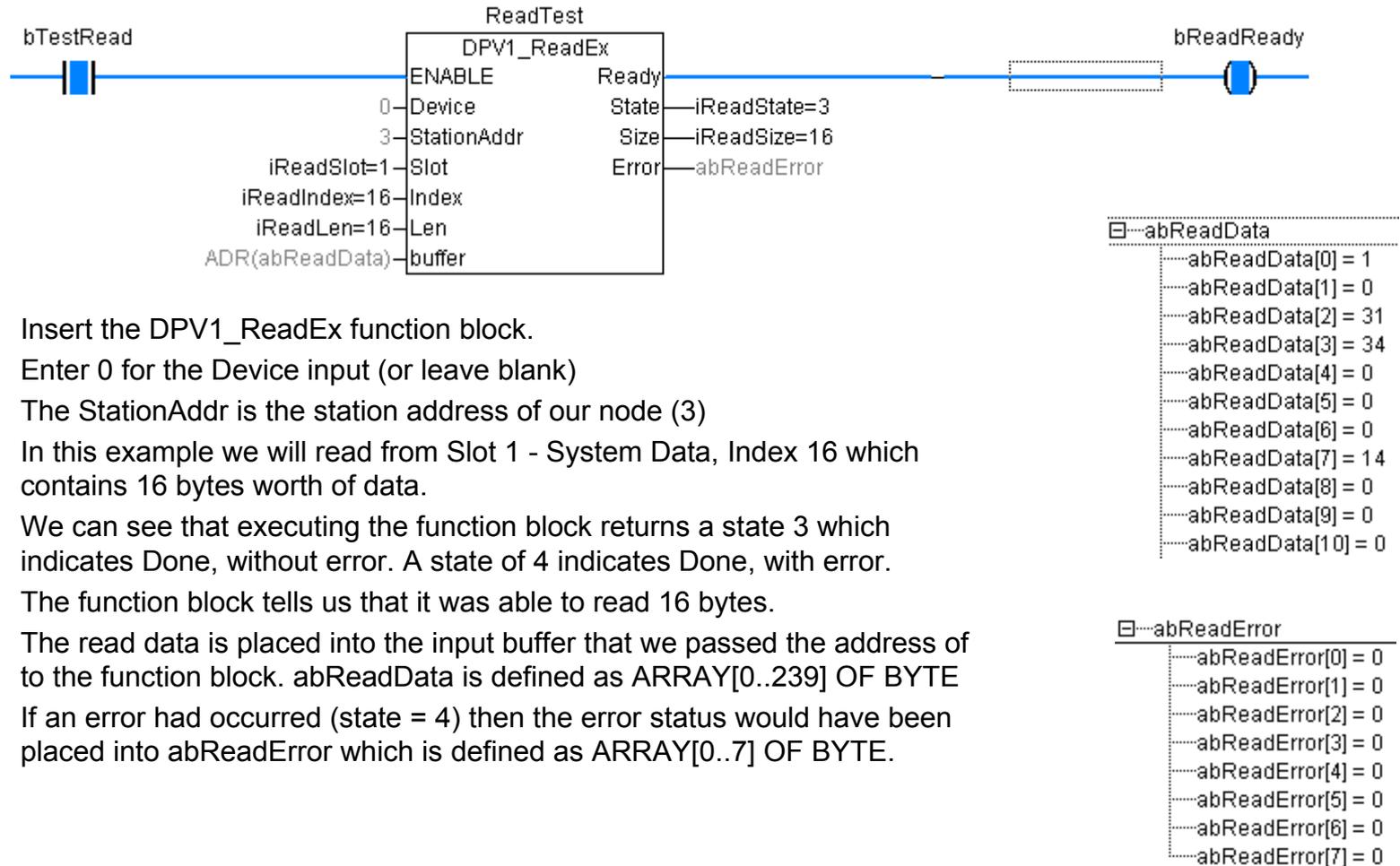
FUNCTION_BLOCK DPV1_Read
VAR_INPUT
  ENABLE:BOOL;
  Device:INT;
  StationAddr:INT;
  Slot:INT;
  Index:INT;
  Len:INT;

```

Below the definition is a diagram of the 'DPV1\_READ' function block. It is a rectangular box with the title 'DPV1\_READ' at the top. On the left side, there are six input lines labeled: 'ENABLE : BOOL', 'Device : INT', 'StationAddr : INT', 'Slot : INT', 'Index : INT', and 'Len : INT'. On the right side, there are three output lines labeled: 'Ready : BOOL', 'State : V1 State', and 'Size : INT'. At the bottom of the box, there is a line labeled 'buffer : DWORD'.

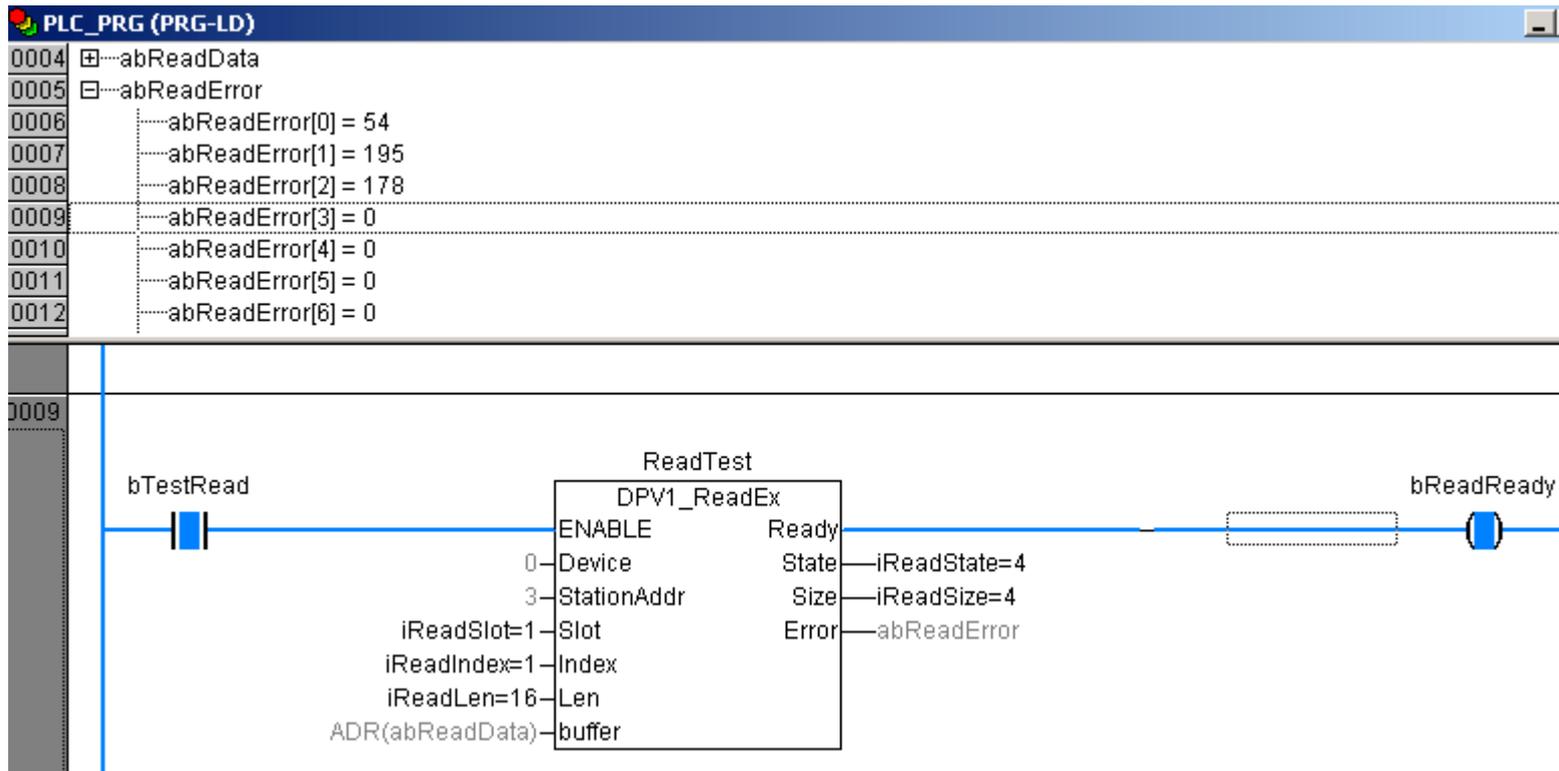
- DPV1 supports 0-255 slots per device with 0-255 indices per slot. Each index may contain up to 240 bytes of data. Refer to the vendor documentation for the device to determine what it supports.
- In our example we will communicate with a FESTO CPX-FB13 which supports slots 1,2,3 and 100-147. Slots 1 contains system parameters, Slot 2 contains Channel-specific module parameters and Slot 3 contains indexed addressing of objects. Slots 100-147 contain Module data and module parameters.

# DPV1 Read



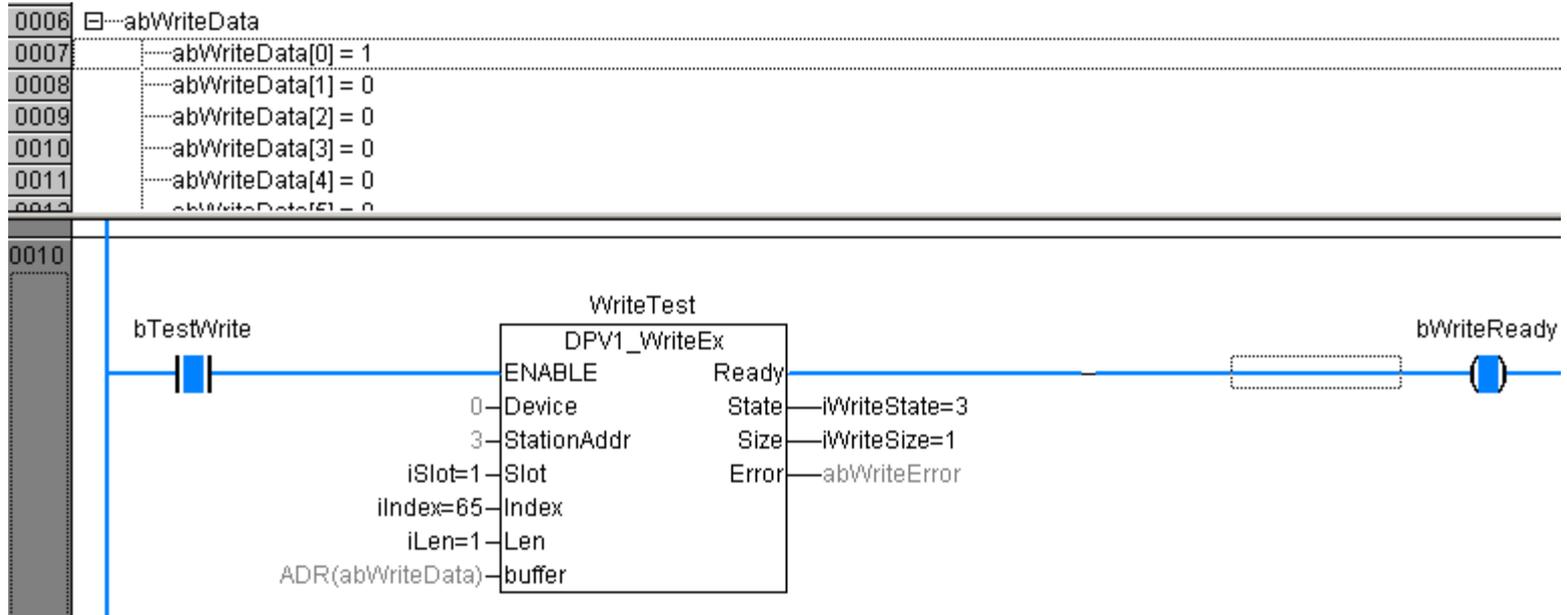
- Insert the DPV1\_ReadEx function block.
- Enter 0 for the Device input (or leave blank)
- The StationAddr is the station address of our node (3)
- In this example we will read from Slot 1 - System Data, Index 16 which contains 16 bytes worth of data.
- We can see that executing the function block returns a state 3 which indicates Done, without error. A state of 4 indicates Done, with error.
- The function block tells us that it was able to read 16 bytes.
- The read data is placed into the input buffer that we passed the address of to the function block. abReadData is defined as ARRAY[0..239] OF BYTE
- If an error had occurred (state = 4) then the error status would have been placed into abReadError which is defined as ARRAY[0..7] OF BYTE.

# DPV1 Read w/Error



- In this example we try to read an invalid Index (1) for Slot 1. The function block returns State 4 (Done, with Error) and we can see the error codes in the abReadError byte array.
- Refer to the vendor documentation for the meaning of the error codes.

# DPV1 Write



- For our DPV1\_Write example we will clear the diagnostic memory of the CPX-FB13 by writing to Slot 1, Index 65.
- Insert the DPV1\_WriteEx function block. The Device input is 0 (or blank), the station address is 3, the slot number is 1, the Index is 65 and the length is 1
- Enter the value you wish to write into the abWriteData array which is defined as ARRAY[0..239] OF BYTE
- Send the message. In this example we can see that the State is 3 which is Done, No error
- Always refer to the vendor documentation for the details on which slots and indexes are available to write to.

## Conclusion

- This concludes the SMLC/PROFIBUS DP Master tutorial
- PROFIBUS and PROFIBUS DP are registered trademarks of PROFIBUS International.

