WAGO-I/O-*PRO* library

WagoLib_IO_Link.lib

The WagoLib_IO_Link.lib library provides function blocks which allow to configure the IO Link module 750-657 as well as to read the IO Link sensors ISDUs.

Content

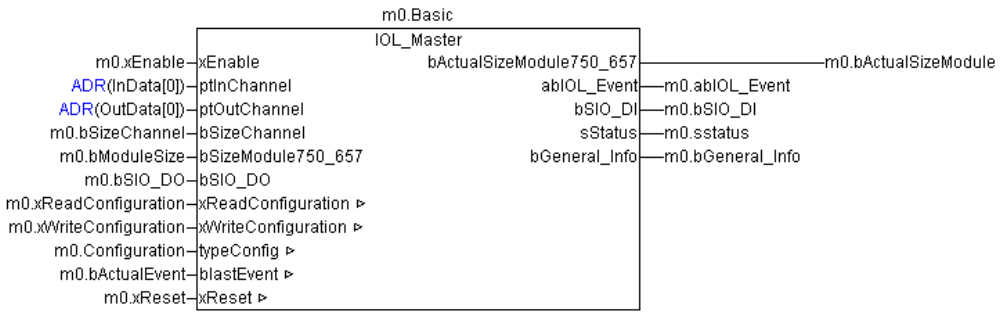
WagoLib_IO_Link.lib

IOL

IOL_Master

WAGO-I/O-*PRO* 32 Library elements			
Category:	IO Link Master		
Name:	IOL_Master		
Type:	Function	Function block **X**	Program
Library name:	WagoLib_IO_Link.lib		
Requiered libraries			
Applicable to:	**750-657**		
Input parameter:	**Data type:**	**Comment:**	
xEnable	BOOL	Enable configuration channel	
ptInChannel	POINTER TO ARRAY[0..255] OF BYTE	Pointer to input array of communication channel	
ptOutChannel	POINTER TO ARRAY[0..255] OF BYTE	Pointer to output array of communication channel	
bSizeChannel	BYTE	Size of communication channel (in Byte)	
bSizeModule750_657	BYTE	Size of the whole module (4,12,16,24)	
bSIO_DO	BYTE	Output digital signals (SIO)	
Input/Output parameter:	**Data type:**	**Comment:**	
xReadConfiguration	BOOL	Read parameter	
xWriteConfiguration	BOOL	Write parameter	
typeConfig	typeConfig	Parameter (app. 1,2,3,6)	
bLastEvent	BYTE	Actual event counter	
xReset	BOOL	Reset	
Output parameter:	**Data type:**	**Comment:**	
bActualSizeModule750_657	BYTE	Actual module size	
abIOL_Event	ARRAY[0..gcIOL_EventBuffer] OF IOL_typEvent	Array with diagnostic messages	
bSIO_DI	BYTE	Digital sensors input (SIO)	
sStatus	STRING	Status	
bGeneral_Info	BYTE	Status Byte 0 of module	
Graphical description:			

WAGO-I/O-*PRO* 32 Library elements



WAGO-I/O-*PRO* 32 Library elements

Functional description:

This function block handles the access to the IO Link module. The variables ptInChannel and ptOutChannel should be mapped to the input and output addresses of the modules acyclic channel.

Each module 750-657 needs exactly one function block instance.

This function block provides diagnosis information. The variable bLastEvent indicates the actual event counter. The event details are stored in the array abIOL_Event. The size of the array is defind by the global constant gcIOL_EventBuffer. This buffer is handled as a ring buffer.

The actual event counter will be increased with each received event. The appropriate event is listed in the array abIOL_Event[abIOL_Event-1, therefore the variable bLastEvent always points to the next free entry in the buffer.

A subset of configuration values may be read on written by this function block.

Parameter 1: Channel fragmentation enable, Channel diagnostic enable(only RAM)

Parameter 2: Channel input size

Parameter 3: Channel output size

Parameter 6: Channel mode (deactivated, input, output, IO Link)

Some of the values are written only to the RAM like diagnostic enable (for details see manual).

bSIO:

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	0	0	0	PORT4	PORT3	PORT2	PORT1

bGeneral_Info:

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
REG	ERR	PORT4	PORT3	PORT2	PORT1	IOL_M	INTERN

WAGO-I/O-*PRO* 32 Library elements
TYPE IOL_typSimpleConfig : STRUCT bSizeChannel_1_Input:BYTE; bSizeChannel_1_Output:BYTE; xFragmentedChannel_1:BOOL;(*1=fragmented data transmission*) bModeChannel_1:BYTE;(*0:deactivated,1=DI,2=DO,3=IOL data*) xEnableDiagnoseChannel_1:BOOL;(*1=diagnosis enabled*) bSizeChannel_2_Input:BYTE; bSizeChannel_2_Output:BYTE; xFragmentedChannel_2:BOOL;(*1=fragmented data transmission*) bModeChannel_2:BYTE;(*0:deactivated,1=DI,2=DO,3=IOL data*) xEnableDiagnoseChannel_2:BOOL;(*1=diagnosis enabled*) bSizeChannel_3_Input:BYTE; bSizeChannel_3_Output:BYTE; xFragmentedChannel_3:BOOL;(*1=fragmented data transmission*) bModeChannel_3:BYTE;(*0:deactivated,1=DI,2=DO,3=IOL data*) xEnableDiagnoseChannel_3:BOOL;(*1=diagnosis enabled*) bSizeChannel_4_Input:BYTE; bSizeChannel_4_Output:BYTE; xFragmentedChannel_4:BOOL;(*1=fragmented data transmission*) bModeChannel_4:BYTE;(*0:deactivated,1=DI,2=DO,3=IOL data*) xEnableDiagnoseChannel_4:BOOL;(*1=diagnosis enabled*) END_STRUCT END_TYPE

IOL_Call

WAGO-I/O-*PRO* 32 Library elements			
Category:	IO Link Master		
Name:	IOL_Call		
Type:	Function	Function block **X**	Program
Library name:	WagoLib_IO_Link.lib		
Requiered libraries			
Applicable to:	**750-657**		

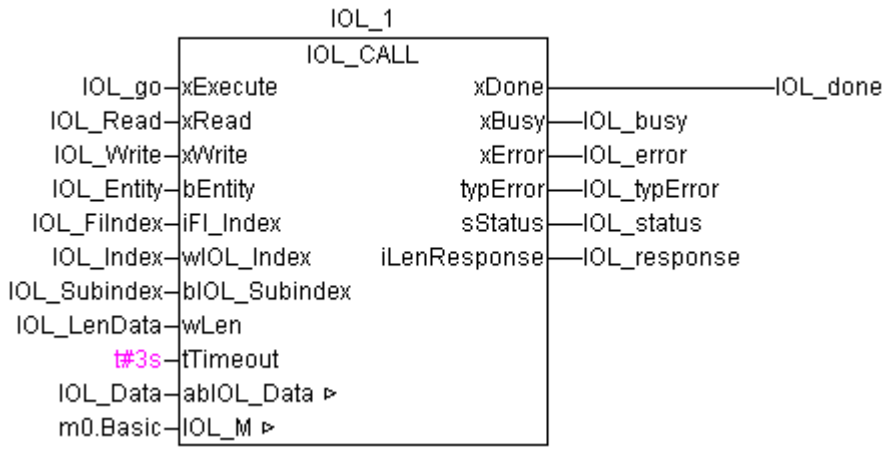
Input parameter:	**Data type:**	**Comment:**
xExecute	BOOL	Execute action
xRead	BOOL	Read data
xWrite	BOOL	Write data
bEntity	BYTE	0:Master,1..4 IO Link device
iFI_Index	INT	Index according to data area
wIOL_Index	WORD	Index if ISDU Call
bIOL_Subindex	Byte	Subindex if ISDU Call
wLen	WORD	Size of data to be written
tTimeout	TIME	Time after which a response is expected

Input/Output parameter:	**Data type:**	**Comment:**
abIOL_Data	ARRAY[0..255] OF BYTE	Reading: data will be filled by functionblock Writing: fill in data to be written
IOL_M	IOL_Master	Instance of the communication function block IOL_Master

Output parameter:	**Data type:**	**Comment:**
xDone	BOOL	Job done
xBusy	BOOL	Job busy
xError	BOOL	Error occurred, detail see typError
typError	IOL_typPDU_Error	Error details according to the manual
sStatus	STRING	Status details
iLenResponse	INT	Length of response

Graphical description:

WAGO-I/O-*PRO* 32 Library elements



WAGO-I/O-*PRO* 32 Library elements

Functional description:

- This functionblock allows to perform a ISDU Call to the IO Link sensor
  bEntity=1..4, number of port where the IO Link sensor is connected
  iFI_Index=98
  wIOL_Index= according to IO Link specification
  bIOL_Subindex= according to IO Link specification
  wLen=length of the data in case of writing
  abIOL_Data=data
  Example:
  wIOL_Index=16 Name of IO Link device
  bIOL_Subindex=0
- This functionblock allows to acces the master configuration
  bEntity=0
  iFI_Index=98
  wIOL_Index=0x….
  bIOL_Subindex=0
  wLen=length of the data in case of writing
  Example:
  wIOL_Index=0x0200 Port configuration Port 1
  bIOL_Subindex=0
- This functionblock allows to acces the I&M records
  bEntity=0
  iFI_Index=0..4
  wLen=length of the data in case of writing
  abIOL_Data=data

  wIOL_Index=input not used
  bIOL_Subindex= input not used
- TYPE IOL_typPDU_Error:(manual chapter 11.1.2.1.4)

```
STRUCT
        M_Error_Code        :  WORD;(manual table 62,63)
        D_Error_Code        :  BYTE;(manual table 64)
        D_Add_Error_Code    :  BYTE;(manual table 64)
END_STRUCT
END_TYPE
```

IOL_M_Configuration

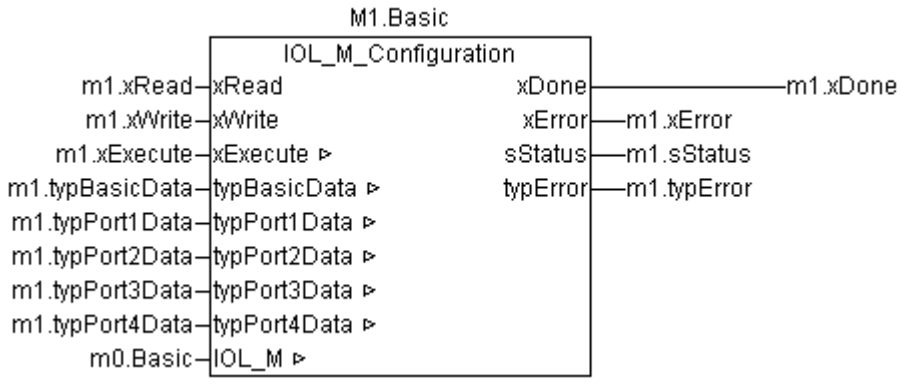
WAGO-I/O-*PRO* 32 Library elements			
Category:	IO Link Master		
Name:	IOL_M_Configuration		
Type:	Function	Function block **X**	Program
Library name:	WagoLib_IO_Link.lib		
Requiered libraries			
Applicable to:	**750-657**		

Input parameter:	**Data type:**	**Comment:**
xRead	BOOL	Read master and port 1..4 configuration
xWrite	BOOL	Write master and port 1..4 configuration

Input/Output parameter:	**Data type:**	**Comment:**
xExecute	BOOL	Start configuration process
typBasicData	IOL_typBasicConfiguration	Data according to IOL_M_BasicConfiguration
typPort1Data	IOL_typPortConfiguration	Data according to IOL_M_PortConfiguration
typPort2Data	IOL_typPortConfiguration	Data according to IOL_M_PortConfiguration
typPort3Data	IOL_typPortConfiguration	Data according to IOL_M_PortConfiguration
typPort4Data	IOL_typPortConfiguration	Data according to M_Port_Configuration
IOL_M	IOL_Master	Instance of the communication function block IOL_Master

Output parameter:	**Data type:**	**Comment:**
xDone	BOOL	Configuration process finished
xError	BOOL	Error while configurating
sStatus	STRING	Status information
typError	IOL_typPDU_Error	Error details according to the manual

Graphical description:



WAGO-I/O-*PRO* 32 Library elements

Functional description:

- This function block allows to configure the module. The values will be stored as user settings in the flash memory. Therefor make sure the write command is not perfomed permanently since the flash allows app. 100.000 write cycles.
- Details of the different configuration values will be described in the manual (chapter 11.3.1 and 11.3.2).
- The visualisation template IOL_Expert allows to configure the values graphically.
- TYPE IOL_typBasicConfiguration :(manual chapter 11.3.1 table 75)

```
STRUCT
        No_Channel          :   BYTE; (*read only; always 4*)
        Com_Mode            :   BYTE ;(*read only; 3
        Master_Max_Cycle    :   BYTE ;
        IOL_Rev             :   BYTE ;
        IOLM_Feature1       :   WORD ;
        IOLM_Feature2       :   WORD ;
        IOLM_PAB_STRUCT     :   ARRAY[0..36] OF BYTE;
END_STRUCT
END_TYPE
```

- TYPE IOL_typPortConfiguration :(manual chapter 11.3.2 table 93)

```
        STRUCT
        Port_Configuration      :   WORD ;
        COMP_Data               :   ARRAY[0..23] OF BYTE ;
        Master_CMD              :   BYTE ;
        PortInspectionLevel     :   BYTE ;
        Param_Serv_Modi         :   BYTE ;
        IOLM_Feature1           :   WORD ;
        IOLM_Feature2           :   WORD ;
        END_STRUCT
END_TYPE
```

- TYPE IOL_typPDU_Error :(manual chapter 11.1.2.1.4)

```
        STRUCT
                M_Error_Code    :   WORD;(manual table 62,63)
                D_Error_Code    :   BYTE;(manual table 64)
                D_Add_Error_Code:   BYTE;(manual table 64)
        END_STRUCT
END_TYPE
```

IOL_Data_IO_Fragmented

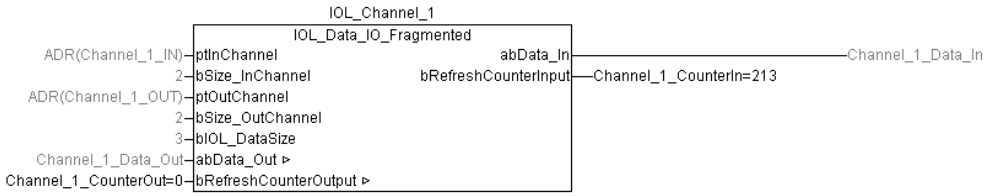
WAGO-I/O-*PRO* 32 Library elements			
Category:	IO Link Master		
Name:	IOL_Data_IO_Fragmented		
Type:	Function	Function block **X**	Program
Library name:	WagoLib_IO_Link.lib		
Requiered libraries			
Applicable to:	**750-657**		

Input parameter:	**Data type:**	**Comment:**
ptInChannel	POINTER TO ARRAY[0..255] OF BYTE	Pointer to input array of port channel
bSize_InChannel	BYTE	Size of input channel (in Byte)
ptOutChannel	POINTER TO ARRAY[0..255] OF BYTE	Pointer to output array of port channel
bSize_OutChannel	BYTE	Size of output channel (in Byte)
bIOL_DataSize	BYTE	Size of the IO Link data (in Byte)

Input/Output parameter:	**Data type:**	**Comment:**
abData_Out	ARRAY[0..31] OF BYTE	Process data output
bRefreshCounterOutput	BYTE	Each time a process value has been written this counter will be increased

Output parameter:	**Data type:**	**Comment:**
abData_In	ARRAY[0..31] OF BYTE	Process data input
bRefreshCounterInput	BYTE	Each time a process value has been received this counter will be increased

Graphical description:



Functional description:

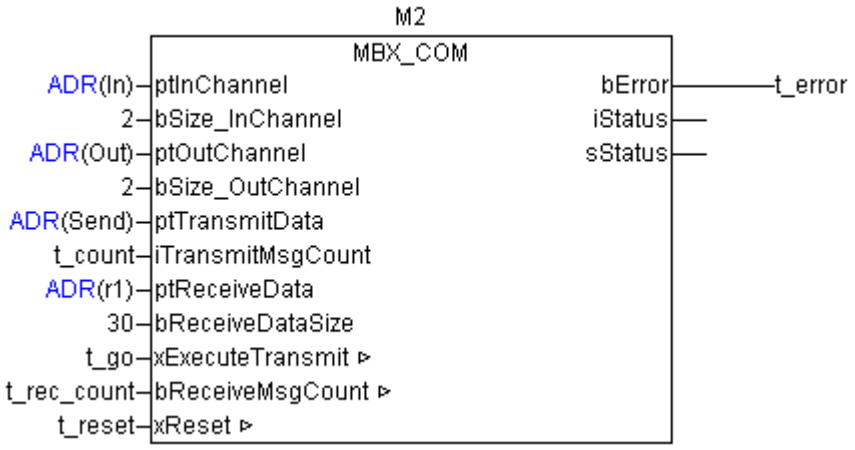
- Allows to receive process data from a fragmented port
- Allows to transmit process data to an IO Link device using a fragmented port
- Example: An IO Link sensor with 3 Byte should only use a 2 Byte port. In this case the settings are: bSize_InChannel=2, bSize_OutChannel=2 and the bIOL_DataSize=3

Mailbox

MBX

WAGO-I/O-*PRO* 32 Library elements			
Category:	Mailbox 2.0		
Name:	MBX_Com		
Type:	Function	Function block **X**	Program
Library name:	WagoLib_IO_Link.lib		
Requiered libraries			
Applicable to:	**750-657**		
Input parameter:	**Data type:**	**Comment:**	
ptInChannel	POINTER TO ARRAY[0..255] OF BYTE	Pointer to input array of communication channel	
bSize_InChannel	BYTE	Size of input channel [in Byte]	
ptOutChannel	POINTER TO ARRAY[0..255] OF BYTE	Pointer to output array of communication channel	
bSize_OutChannel	BYTE	Size of output channel [in Byte]	
ptTransmitData	POINTER TO ARRAY[0..gcIOL_M0_TxMaxMsg] OF MBX_typMsg	Pointer to the messages which shall be transmitted	
iTransmitMsgCount	INT	Number of messages to be transmitted	
ptReceiveData	POINTER TO ARRAY[0..gcIOL_M0_RxMaxMsg] OF MBX_typMsg	Pointer to the area where the received messages will be stored	
bReceiveDataSize	BYTE	Size of the receive message area (gcIOL_M0_RxMaxMsg+1)	
Input/Output parameter:	**Data type:**	**Comment:**	
xExecuteTransmit	BOOL	Transmit message	
bReceiveMsgCount	BYTE	Reset communication channel	
xReset	BOOL	Reset communication channel	
Output parameter:	**Data type:**	**Comment:**	
bError	BYTE	Error	
iStatus	INT	Status as numerical value	
sStatus	STRING	Status	
Graphical description:			

WAGO-I/O-*PRO* 32 Library elements



Functional description:

This function block handles the basic communication with the module. The communication is based on the mailbox2.0 light. This block is used by the IOL_Master and IOL_Data_IO_Fragmented function blocks.

- Internal use
- Handles all messages(ID 0, ID1, ID 2, ID x)

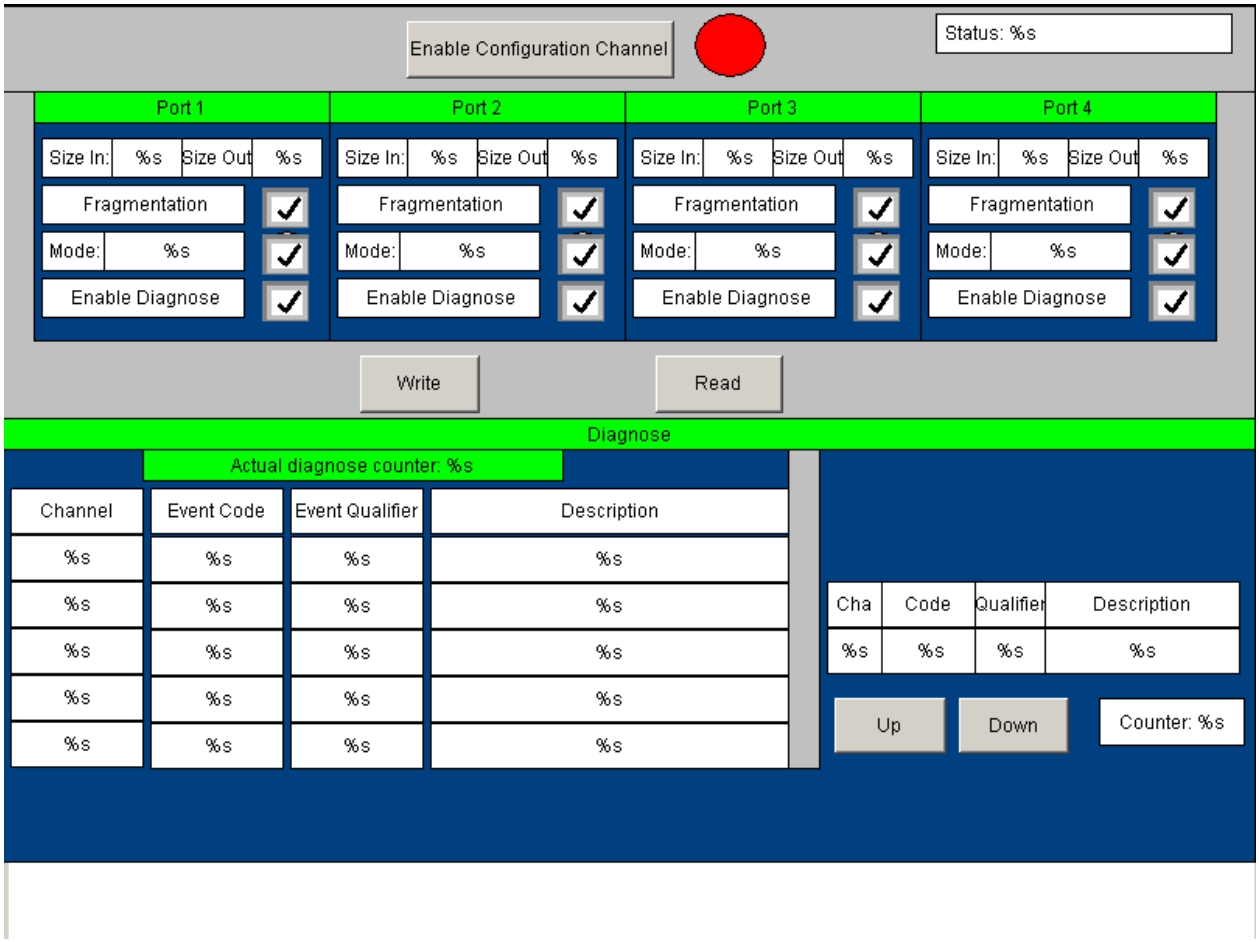
Global Constants

```
gcMBX_MaxMsgLength                : UINT:=255; (*depending on the module ->750-657 do not change*)
gcMBX_RxMaxMsg                    : BYTE:=100;
gcIOL_M0_RxMaxMsg                 : BYTE:=10;(*size of receive message buffer*)
gcIOL_M0_TxMaxMsg                 : BYTE:=10;(*size of transmit message buffer*)
gcIOL_EventBuffer                 : BYTE:=10;(*size of event buffer*)
gcIOL_RegisterCommunikationWatchdog : TIME:=t#500ms;
gcIOL_MsgID_Blocking              : BYTE:=199;(*please do not change*)
gcIOL_DebugSize                   : INT:=20;(*only for debugging purpose*)
```

These constants allow to configure some of the internal buffers.

IOL Visualisation Templates

IOL_Main



This template provides a placeholder “IOL1” which needs to be configured with the instance of the IOL_Master function block.

The button “Enable Configuration Channel” enables the IOL_Master block. If the actual module size matches the configured size, the colour of the indicator changes from red to green.

After this, the program will perform a read command automatically to update the port information.

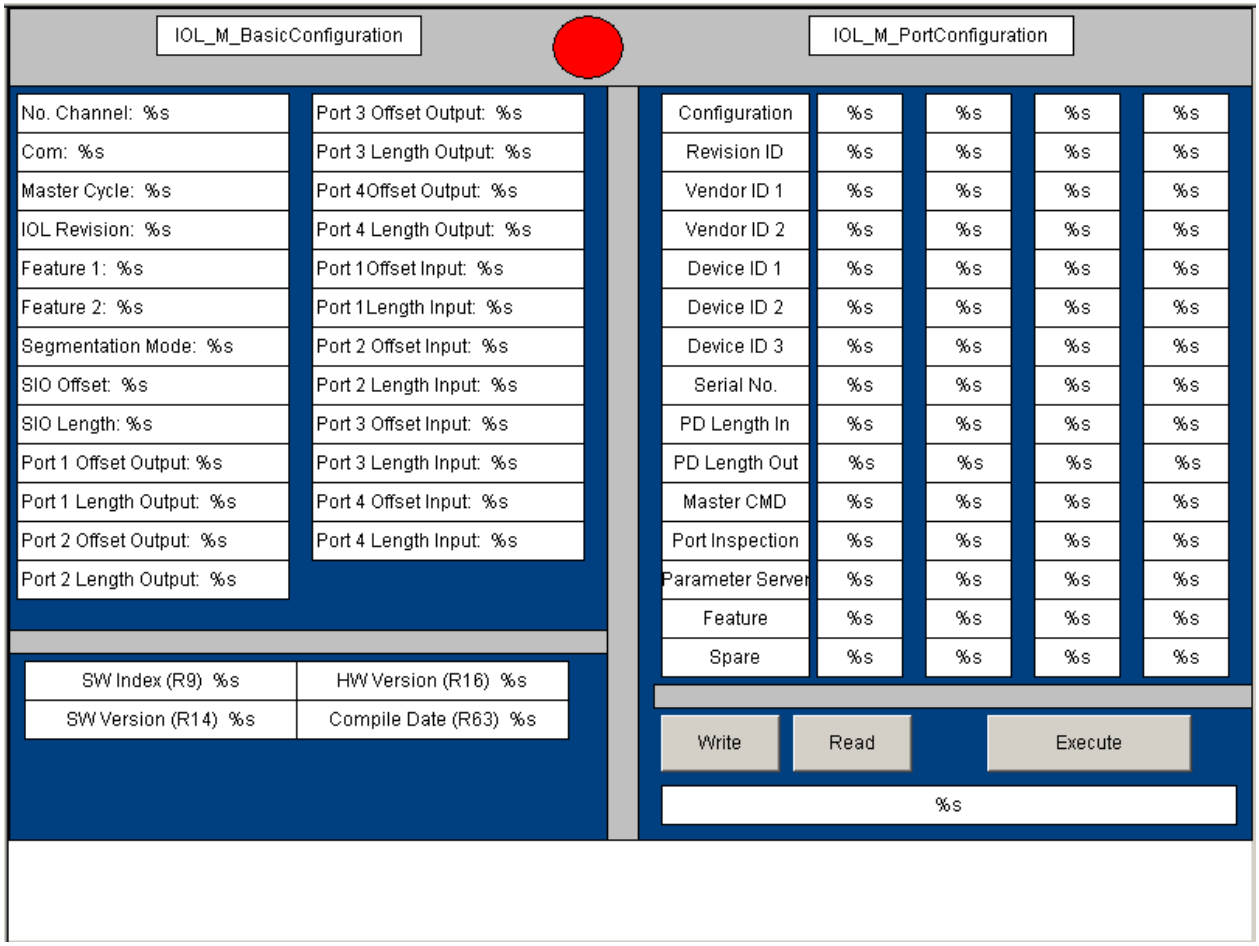
Each port maybe configured in size, fragmentation, mode and diagnosis. Not all changes will be stored permanently.

The maximum size for the ports is 15.

The lower part shows diagnosis details. The lastest five events are displayed. The “Actual diagnose counter” displayes the total number of events (modulo buffer size).

The buttons “Up” and “Down” allow to scroll through the whole event buffer wheras the “Counter” shows the position in the buffer.

IOL_Expert



This template provides a placeholder “IOL1” which needs to be configured with the instance of the IOL_M_Configuration function block.

If the indicator is green the IOL_Master is ready and the configuration maybe processed.

Reading of the configuration can be performed by activating “Read” and “Execute”. The progress is displayed in the textbox below.

Writing of the configuration can be performed by activating “Write” and “Execute”. These values will be written to the flash.

The different parameters a described in detail in the manual (refere to chapter 11.3.1 and 11.3.2).