sigmadue CU-02 & MP-01

IEC 61131-3 Function Block Library

IEC 61131-3 Function Block Library
M.U. CU02-IEC-FB 06/11.05
Cod. J50 - 828 – SWS2 - E

Ascon Tecnologic srl

Headquarters:	via Indipendenza 56, 27029 Vigevano (PV)	***Milan office:***	Via Falzarego 9/11, 20021 Baranzate (MI)
Phone	+39 02 333 371	***Fax***	+39 02 350 4243
`www.ascontecnologic.com`		`sales@ascontecnologic.com`	

INDEX

Chapter 4

Chapter 5

Introduction

The products described in this manual should be installed, operated and maintained only by qualified application programmers and software engineers who are almost familiar with EN 61131-3 concepts of PLC programming, automation safety topics and applicable national standards.

Using this manual

Specifications within the text of this manual are given in the International System of Units (SI), with non SI equivalents in parentheses.

Fully Capitalized words within the text indicate markings found on the equipment.

Words **in bold** style within the text indicate markings found in the Configuration Tools.

Warnings, Cautions and Notes are used to emphasize critical instructions:

DANGER!

Indicates an imminently hazardous situation which, if not avoided, will result in death or serious injury.

WARNING

Indicates a potentially hazardous situation which, if not avoided, could result in death or serious injury.

Caution

Indicates a potentially hazardous situation which, if not avoided, may result in minor or moderate injury, or property damage.

Note: Highlights important information about an operating procedure or the equipment.

Purpose

The purpose of this document is to describe the structure and use of Ascon Spa's proprietary *IEC61131-3* libraries. These libraries cover different aspects of the features of the CU-02 and MP-01 units: from I/O modules interface to Modbus access from the control to the logic management.

Caution

The function blocks of the various libraries are valid for both CU-02 and MP-01 units except when it is expressly indicated.

Description

Chapter 1 *AsconACLib* is a Function Block library that contains a set of generic functionalities that come from the Ascon AC Station Device useful for the IEC 61131 programming.

Chapter 2 *AsconBasicIOLib* (valid only for CU-02 unit) is a Function Block library enabling the access of Ascon SpA sigma**due** line devices from *OpenPCS* programming environment. Thanks to the FBs, the user does not take care of communications details related to CANopen[1] protocol, but he simply has to access to the available functions of the individual devices. The library gives the access to the basic functions of the I/O modules.

Chapter 3 The *AsconControlLib* is a Function Block library dedicated to the process control. It uses the basic functionalities dedicated to the PID implementation present in the firmware of the control unit (CU-02 and MP-01) device in order to provide solution ready to use. In fact in the library there is the implementation of a complete standard regulator in both version: single action and double action for heat and cool application. Please note that are present also different function blocks dedicated to the tuning algorithms.

Chapter 4 *AsconCPULib* (valid only for CU-02 unit) is a function block library that allows the access to the control unit (CU-02) device of the Ascon sigma**due** line, from the OpenPCS programming tool. These FBs allow the user to set and manage the CANopen network activities: diagnostic, failure management of the connected devices, synchronization,…

Chapter 5 *AsconMBCommLib* is a Function Block library which simplifies the access to the MODBUS communication ports available in Ascon SpA sigma**due** line devices.

Current Documentation on the Internet

Make sure you are always working with the latest version of this document.

ASCON spa reserves the right to make changes to its products in the name of technological advancement. New manual revisions, when published, and can be found online at:

http://www.ascon.it

1. A complete series of firmware function blocks implementing the CANopen communications primitives are available for the user: these FBs grant the access to the module directly through the CANopen protocol messages. For further details, see "*CANopen extension for IEC61131-3*" **[6]** available on the installation CD.

Chapter 1
AsconACLib

1-1 Purpose

The purpose of this document is to provide a complete description of the library *AsconACLib*.

1-2 Descritpion

The AsconACLib is a function block library that contains a set of generic functionalities that come from the Ascon AC Station Device useful for the IEC 61131 programming.

The table here reported gives the complete list of the function blocks of the library.

ALARM_ABS	Absolute Alarm
ALARM_ADVANCED	Advanced Alarm
ALARM_BND	Band Alarm
ALARM_DEV	Deviation Alarm
ALARM_RATE	Rate Alarm
AVG_ADV_8REAL	Advanced Instantaneous Average calculation
AVG_MOVING	Moving Average calculation
AVG_RUNNING	Running Average calculation
CHARACTERIZER_8	8 points linearization
CHARACTERIZER_16	16 points linearization
COMPARATOR	Comparator with hysteresis
CONV_AD8	Conversion from BYTE to 8 bits
CONV_AD16	Conversion from WORD to 16 bits
CONV_AD32	Conversion from DWORD to 32 bits
CONV_DA8	Conversion from 8 bits to BYTE
CONV_DA16	Conversion from 16 bits to WORD
CONV_DA32	Conversion from 32 bits to DWORD
COUNTER	Rising Edge Counter
DECODER_8	8 bit Decoder
DEW_POINT	Dew Point calculation
F0_CALCULATION	Sterilization time for bacterial load reduction calculation
FLIPFLOP_D	Flip - Flop D Type

FLIPFLOP_JK	Flip - Flop JK Type
HOLD_VALUE	Analogue Holding value
HR_DRY_WET_BULB	Relative humidity calculation method with dry/wet bulb
INBETWEEN	Analogue Middle Selector
LIMITER_VALUE	Analogue value Limiter
MASS FLOW	Compensate Flow calculation
MIN_MAX_SELECTOR	Minimum/Maximum analogue Selector
MONOSTABLE_DS	Monostable with Delay
MONOSTABLE_NED	Monostable with Delay on the Negative Edge
MONOSTABLE_PED	Monostable with Delay on the Positive Edge
MONOSTABLE_PUL	Monostable Pulse Generator
MP_RTD_LIN	RTD Linear rescaling (for microPAC ONLY)
MUX_A8	Analogue Multiplexer 8 Input selection
MUX_A16	Analogue Multiplexer 16 Input selection
MUX_D8	Digital Multiplexer 8 Input selection
MUX_D16	Digital Multiplexer 16 Input selection
RESCALE	Analogue Rescaling
SAMPLING_TIME	Application Sampling Time Statistics
SLOPE_LIMIT	Analogue step variations (Slopes) Limiter
TIMER_ADV	Advanced Timer
TOTALIZER	Analogue Totalizer
TOTALIZER_ADV	Advanced Analogue Totalizer with Time Base selection

1-3 Description of the individual Function Blocks

1-3-1 ALARM_ABSOLUTE MODULE

FB Prototype

ALARM_ABS

Input parameters

Label	Type	Description
VALUE	REAL	Input value
ABS_HIGH	REAL	Absolute Alarm Max
HY_AHIGH	REAL	Hysteresis Absolute Alarm Max
ABS_LOW	REAL	Absolute Alarm Min
HY_ALOW	REAL	Hysteresis Absolute Alarm Min

Output parameters

Label	Type	Description
ABS_H	BOOL	Absolute High Alarm
ABS_L	BOOL	Absolute Low Alarm

Description This function block generates only independent ABSOLUTE high and low alarms.

Alarm description

Alarm Type	Description
Absolute High	The output ABS_H is active if it is TRUE the condition: VALUE > (ABS_HIGH + HY_AHIGH)
Absolute Low	The output ABS_L is active if it is TRUE the condition VALUE < (ABS_LOW - HY_ALOW)

Default Values Table

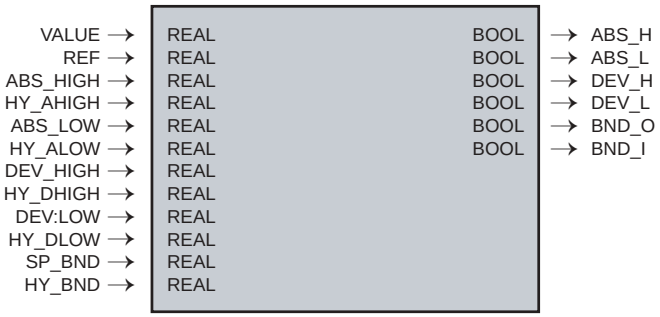
Input	Default value
VALUE	0.0
ABS_HIGH	0.0
HY_AHIGH	1.0
ABS_LOW	0.0
HY_ALOW	1.0

Output	Description
ABS_H	FALSE
ABS_L	FALSE

1-3-2 ALARM_ADVANCED

FB Prototype

ALARM_ADVANCED



Input parameters

Label	Type	Description
VALUE	REAL	Input value
REF	REAL	Reference value
ABS_HIGH	REAL	Absolute Alarm Max
HY_AHIGH	REAL	Hysteresis Absolute Alarm Max
ABS_LOW	REAL	Absolute Alarm Min
HY_ALOW	REAL	Hysteresis Absolute Alarm Min

Output parameters

Label	Type	Description
ABS_H	BOOL	Absolute High Alarm
ABS_L	BOOL	Absolute Low Alarm
DEV_H	BOOL	Deviation High Alarm
DEV_L	BOOL	Deviation Low Alarm
BND_O	BOOL	Band Out Alarm
BND_I	BOOL	Band In Alarm

Description

This function block generates different types of alarms. It can be used also as a comparator. Please note that all the alarm conditions are always evaluated.

Alarm description

Alarm Type	Description
Absolute High	The output ABS_H is active if it is TRUE the condition: VALUE > (ABS_HIGH + HY_AHIGH)
Absolute Low	The output ABS_L is active if it is TRUE the condition VALUE < (ABS_LOW - HY_ALOW)
Deviation High	The output DEV_H is active if it is TRUE the condition VALUE > (REF +(DEV_HIGH + HY_DHIGH))
Deviation Low	The output DEV_L is active if it is TRUE the condition VALUE < (REF + (DEV_LOW - HY_DLOW))
Band Out	The output BND_O is active if it is TRUE the conditions: VALUE > (REF + SP_BND / 2 + HY_BND) or VALUE < (REF - SP_BND / 2 + HY_BND)
Band In	The output BND_I is active if it is TRUE the conditions: VALUE > (REF - SP_BND / 2 - HY_BND) and VALUE < (REF + SP_BND / 2 - HY_BND)

Default values

Input	Default values
VALUE	0.0
REF	0.0
ABS_HIGH	0.0
HY_AHIGH	1.0
ABS_LOW	0.0
HY_ALOW	1.0

Output	Default values
ABS_H	FALSE
ABS_L	FALSE
DEV_H	FALSE
DEV_L	FALSE
BND_O	FALSE
BND_I	FALSE

1-3-3 ALARM_BAND MODULE

FB Prototype

ALARM_BND

Input parameters

Label	Type	Description
VALUE	REAL	Input value
REF	REAL	Reference value
SP_BND	REAL	Band Alarm
HY_BND	REAL	Hysteresis Band Alarm

Output parameters

Label	Type	Description
BND_O	BOOL	Band Out Alarm
BND_I	BOOL	Band In Alarm

Description

This function block generates only independent BAND inside or outside alarms.

Alarm description

Alarm Type	Description
Band Out	The output BND_O is active if it is TRUE the conditions: VALUE > (REF + SP_BND / 2 + HY_BND) or VALUE < (REF - SP_BND / 2 + HY_BND)
Band In	The output BND_I is active if it is TRUE the conditions: VALUE > (REF - SP_BND / 2 - HY_BND) and VALUE < (REF + SP_BND / 2 - HY_BND)

Default values

Input	Default value
VALUE	0.0
REF	0.0
SP_BND	0.0
HY_BND	1.0

Output	Default value
BND_O	FALSE
BND_I	FALSE

1-3-4 ALARM_DEVIATION MODULE

FB Prototype

ALARM_DEV

Input parameters

Label	Type	Description
VALUE	REAL	Input value
REF	REAL	Reference value
DEV_HIGH	REAL	Deviation Alarm Max
HY_DHIGH	REAL	Hysteresis Deviation Alarm Max
DEV_LOW	REAL	Deviation Alarm Min
HY_DLOW	REAL	Hysteresis Deviation Alarm Min

Output parameters

Label	Type	Description
DEV_H	BOOL	Deviation High Alarm
DEV_L	BOOL	Deviation Low Alarm

Description This function block generates only independent DEVIATION high and low alarms.

Alarm description

Alarm Type	Description
Deviation High	The output DEV_H is active if it is TRUE the condition VALUE > (REF +(DEV_HIGH + HY_DHIGH))
Deviation Low	The output DEV_L is active if it is TRUE the condition VALUE < (REF + (DEV_LOW - HY_DLOW))

Default values

Input	Default values
VALUE	0.0
REF	0.0
ABS_HIGH	0.0
HY_AHIGH	1.0
ABS_LOW	0.0
HY_ALOW	1.0

Output	Default values
DEV_H	FALSE
DEV_L	FALSE

1-3-5 ALARM RATE

FB Prototype

ALARM_RATE

Input parameters

Label	Type	Description
ENABLE	BOOL	Function block enable input
VALUE	REAL	Measure value input
SLOPE_UP	REAL	UP rate input variation per second
SLOPE_DOWN	REAL	DOWN rate input variation per second
HYST	REAL	Hysteresis

Output parameters

Label	Type	Description
CONFIRM	BOOL	Function block confirm
DOUT	BOOL	Active if input slope rate exceeds max./min. limits

Description

This function block generate an alarm if the input value changes with a slope rate grater than the set limits.

Default values

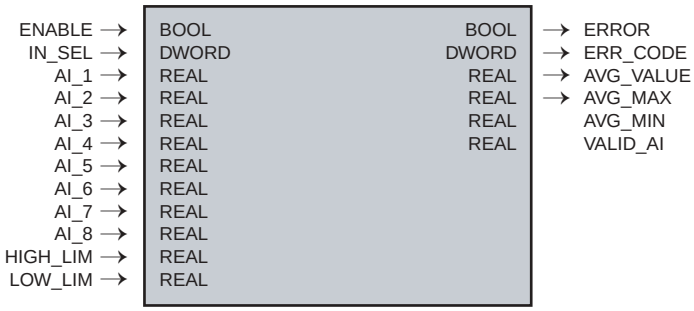
Input	Default value
ENABLE	FALSE
VALUE	0.0
SLOPE_UP	0.0
SLOPE_DOWN	0.0
HYST	0.0

Output	Default value
CONFIRM	FALSE
DOUT	FALSE

1-3-6 AVG_ADV_8REAL

FB Prototype

AVG_ADV_8REAL



Input parameters

Label	Type	Description	Range
ENABLE	BOOL	Command to enable average calculation	
IN_SEL	REAL	Bit mask to enable/disable the AI channels to be evaluated [bit mask]	2#00000000... 2#11111111
AI_1... AI_8	REAL	n^{th} analogue input value [e.u.]	LOW_LIM... HIGH_LIM
HIGH_LIM	REAL	High limit for the AI_x channel values [e.u.]	$-3.4E^{-38}$... $3.4E^{+38}$
LOW_LIM	REAL	Low limit for the AI_x channel values [e.u.]	$-3.4E^{-38}$... $3.4E^{+38}$

Output parameters

Label	Type	Description	Range
ERROR	BOOL	Error status	
ERR_CODE	DWORD	Error code [bit mask]	16#00 00 00 00… 16#FF FF FF FF
AVG_VALUE	REAL	Averaged calculated value [e.u.]	LOW_LIM... HIGH_LIM
AVG_MAX	REAL	Highest average calculated value [e.u.]	LOW_LIM... HIGH_LIM
AVG_MIN	REAL	Lowest average calculated value [e.u.]	LOW_LIM... HIGH_LIM
VALID_AI	REAL	Amount of active valid input value [num]	0... 8

Description

This function block performs an instantaneous math average calculation of the selected AI_x input valid values. As “valid values” are intended all those AI_x inputs which the value is inside the range defined by LOW_LIM and HIGH_LIM parameters: other channels are automatically removed from the calculation. The function block returns also some statistical information and, in particular the highest and lowest ever average value calculated since the last activation by the ENABLE command and the instantaneous amount of AI channels on which the average is performed.

Default Values

Input	Default Value
ENABLE	FALSE
IN_SEL	0
AI_1	0.0
AI_2	0.0
AI_3	0.0
AI_4	0.0

Input	Default Value
AI_5	0.0
AI_6	0.0
AI_7	0.0
AI_8	0.0
HIGH_LIM	999.0
LOW_LIM	-99.0

Reference Table

Output	Description
ERR_CODE.0	AI_1 value out of admitted range
ERR_CODE.1	AI_2 value out of admitted range
ERR_CODE.2	AI_3 value out of admitted range
ERR_CODE.3	AI_4 value out of admitted range
ERR_CODE.4	AI_5 value out of admitted range
ERR_CODE.5	AI_6 value out of admitted range
ERR_CODE.6	AI_7 value out of admitted range
ERR_CODE.7	AI_8 value out of admitted range
ERR_CODE.8	No valid AI_x input values

1-3-7 AVG_MOVING

FB Prototype

AVG_MOVING

Input parameters

Label	Type	Description	Range
ENABLE	BOOL	Command to enable average calculation	
AI	REAL	Analogue input value [e.u.]	LOW_LIM... HIGH_LIM
T_SAMP	REAL	Analogue input sampling time value [ss]	0.1... 3600.0
NR_DATA	REAL	No. of samples to be computed [num]	1... 24
HIGH_LIM	REAL	High limit for the AI channel value [e.u.]	$-3.4E^{-38}$... $3.4E^{+38}$
LOW_LIM	REAL	Low limit for the AI channel value [e.u.]	$-3.4E^{-38}$... $3.4E^{+38}$

Output parameters

Label	Type	Description	Range
ERROR	BOOL	Error status	
ERR_CODE	DWORD	Error status	
BUFF_FULL	BOOL	Error code [bit mask]	16#00 00 00 00… 16#FF FF FF FF
AVG_VALUE	REAL	Buffer fulfil status indication	1 = NR_DATA value reached

Description

This function block performs a moving average calculation of the AI input valid value. A “valid value” means the AI input value is inside the range defined by LOW_LIM and HIGH_LIM parameters, otherwise it will be not computed. The function block returns a continuous averaged value of the last NR_DATA amount of samples, accordingly to the sampling time defined by the T_SAMP parameter (basically it uses a FIFO technique), since the last activation by the ENABLE command. The BUFF_FULL status indicates when the average calculation is performed really on the desired amount of samplings (NR_DATA value).

Default Values

Input	Default Value
ENABLE	FALSE
AI	0.0
T_SAMP	0.5
NR_DATA	12
HIGH_LIM	999.0
LOW_LIM	-99.0

Reference Table

Output	Description
ERR_CODE.0	Sampling time lower than the admitted value
ERR_CODE.1	Sampling time higher than the admitted value
ERR_CODE.2	Number of sampling lower than the admitted value
ERR_CODE.3	Number of sampling higher than the admitted value
ERR_CODE.4	AI value lower than the admitted value

Output	Description
ERR_CODE.5	AI value higher than the admitted value
ERR_CODE.6	Cyclic time exceeds the desired sampling time

1-3-8 AVG_RUNNING

FB Prototype

AVG_RUNNING

Input parameters

Label	Type	Description	Range
ENABLE	BOOL	Command to enable average calculation	
AI	REAL	Analogue input value [e.u.]	LOW_LIM... HIGH_LIM
T_SAMP	REAL	Analogue input sampling time value [ss]	0.1... 3600.0
NR_DATA	REAL	Number of samples to be computed [num]	1... 255
HIGH_LIM	REAL	High limit for the AI channel value	$-3.4E^{-38}$... $3.4E^{+38}$
LOW_LIM	REAL	Low limit for the AI channel value	$-3.4E^{-38}$... $3.4E^{+38}$

Output parameters

Label	Type	Description	Range
ERROR	BOOL	Error status	
ERR_CODE	DWORD	Error code [bit mask]	16#00 00 00 00 … 16#FF FF FF FF
BUFF_FULL	BOOL	Buffer fulfil status indication	1 = NR_DATA value reached
AVG_VALUE	REAL	Averaged calculated value [e.u.]	LOW_LIM... HIGH_LIM

Description

This function block performs a running average calculation of the AI input valid value. A “valid value” means the AI input value is inside the range defined by LOW_LIM and HIGH_LIM parameters, otherwise it will be not computed. The function block returns a continuous averaged value up to the NR_DATA amount of samples, accordingly to the sampling time defined by the T_SAMP parameter and then stops. The BUFF_FULL status indicates when the average calculation reaches the desired amount of samplings (NR_DATA value).

Default Values Table

Input	Default Value
ENABLE	FALSE
AI	0.0
T_SAMP	0.5
NR_DATA	120
HIGH_LIM	999.0
LOW_LIM	-99.0

Reference Table

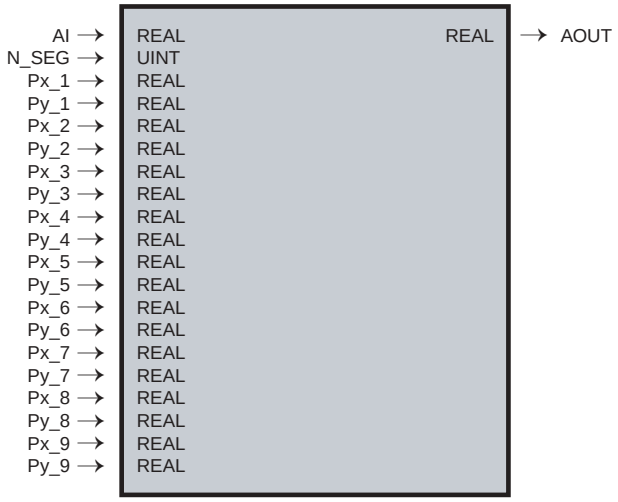
Output	Description
ERR_CODE.0	Sampling time lower than the admitted value
ERR_CODE.1	Sampling time higher than the admitted value
ERR_CODE.2	Number of sampling lower than the admitted value
ERR_CODE.3	Number of sampling higher than the admitted value
ERR_CODE.4	AI value lower than the admitted value

Output	Description
ERR_CODE.5	AI value higher than the admitted value
ERR_CODE.6	Cyclic time exceeds the desired sampling time

1-3-9 CHARACTERIZER 8 SEGMENTS

FB Prototype

CHARACTERIZER_8



Input parameters

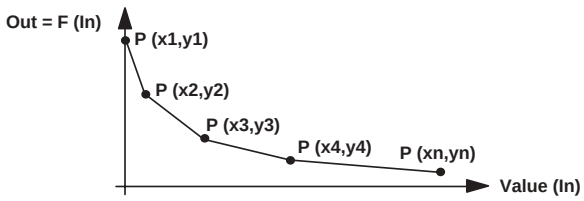
Label	Type	Description
AI	REAL	Input value
N_SEG	UINT	Number of segments
Px_1 ÷ Px_9	REAL	X Coordinates of the segments
Py_1 ÷ Py_9	REAL	Y Coordinates of the segments

Output parameters

Label	Type	Description
AOUT	REAL	Output linearized value

Description

This function block provides a linear interpolation of the input signal, using the table passed as input (*Px_1 ÷ Px_9* and *Py_1 ÷ Py_9*). The parameter *N_SEG* specifies the number of the segments in which is divided the input range: if this value is 0 or grater then 8, the output *AOUT* is always 0.0. Furthermore the FB checks if the *AI* input is out of range, and in this case it forces the minimum or the maximum values provided for the output linearization.



Default values

Input	Default values
AI	0.0
N_SEG	0

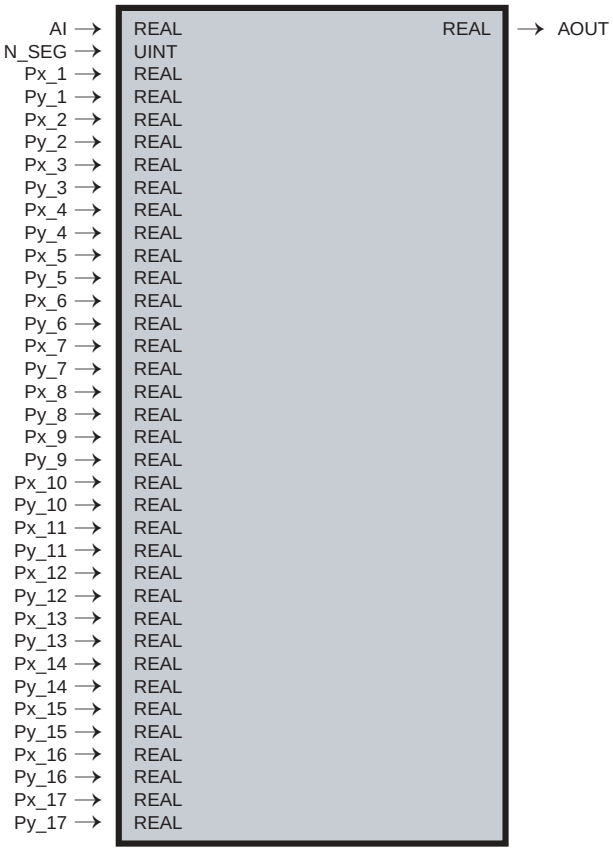
Output parameters

Output	Default values
AOUT	0.0

1-3-10 CHARACTERIZER 16 SEGMENTS

FB Prototype

CHARACTERIZER_16



Input parameters

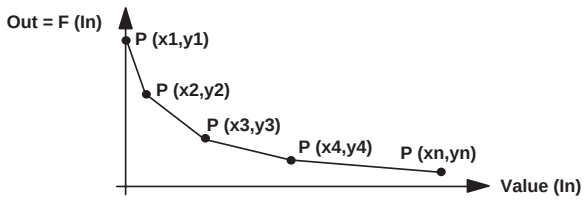
Label	Type	Description
AI	REAL	Input value
N_SEG	UINT	Number of segments
Px_1 ÷ Px_17	REAL	X Coordinates of the segments
Py_1 ÷ Py_17	REAL	Y Coordinates of the segments

Output parameters

Label	Type	Description
AOUT	REAL	Output linearized value

Description

This function block provides a linear interpolation of the input signal, using the table passed as input (*Px_1 ÷ Px_17* and *Py_1 ÷ Py_17*). The parameter *N_SEG* specifies the number of the segments in which is divided the input range: if this value is 0 or grater then 16, the output *AOUT* is always 0.0. Furthermore the FB checks if the *AI* input is out of range, and in this case it forces the minimum or the maximum values provided for the output linearization.



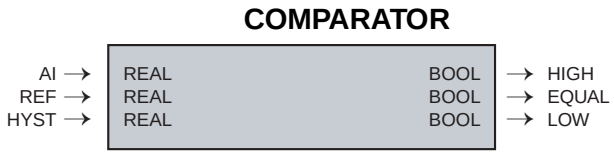
Default values

Inputs	Default values
AI	0.0
N_SEG	0

Outputs	Default values
AOUT	0.0

1-3-11 COMPARATOR

FB Prototype



Input parameters

Label	Type	Description
AI	REAL	Input value
REF	REAL	Reference value
HYST	REAL	Hysteresis (symmetrical respect to reference)

Output parameters

Label	Type	Description
HIGH	BOOL	Digital output for *AI > REF + HYST*
EQUAL	BOOL	Digital output for *REF – HYST* $\leqq$ *AI* $\leqq$ *REF + HYST*
LOW	BOOL	Digital output for *AI < REF - HYST*

Description

This function block compares the input value with a reference value, and provides a digital output indication.

Default values

Input	Default values
AI	0.0
REF	0.0
HYST	0.1

Output	Default values
HIGH	FALSE
EQUAL	FALSE
LOW	FALSE

1-3-12 BYTE A/D CONVERTER

FB Prototype

CONV_AD8

Input parameters

Label	Type	Description
INPUT	BYTE	Analog input

Output parameters

Label	Type	Description
DO_1... DO_8	BOOL	8 bit representation of the analog input

Description This function block converts the analog input BYTE into an 8 single bits format.

Default values

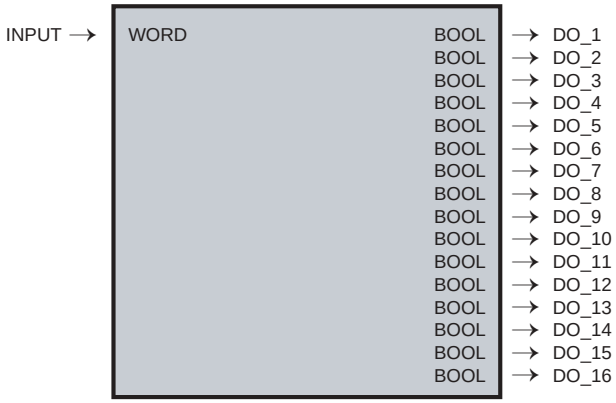
Label	Default values
INPUT	0

Output	Default values
DO_1... DO_8	FALSE

1-3-13 WORD A/D CONVERTER

FB Prototype

CONV_AD16



Input parameters

Label	Type	Description
INPUT	WORD	Analog input

Output parameters

Label	Type	Description
DO_1... DO_16	BOOL	16 bit representation of the analog input

Description This function block converts the analog input WORD into an 16 single bits format.

Default values

Label	Default values
INPUT	0

Output	Default values
DO_1... DO_16	FALSE

1-3-14 DWORD A/D CONVERTER

FB Prototype

CONV_AD32

Input parameters

Label	Type	Description
INPUT	DWORD	Analog input

Output parameters

Label	Type	Description
DO_1... DO_32	BOOL	32 bit representation of the analog input

Description This function block converts the analog input DWORD into an 32 single bits format.

Default values

Label	Default values
INPUT	0

Output	Default values
DO_1... DO_32	FALSE

1-3-15 BYTE D/A CONVERTER

FB Prototype

CONV_DA8

Input parameters

Label	Type	Description
DI_1... DI_8	BOOL	8 bits digital input

Output parameters

Label	Type	Description
OUTPUT	BYTE	byte representation of converted digital input

Description

This function block converts 8 input bits to an output value.

Default values

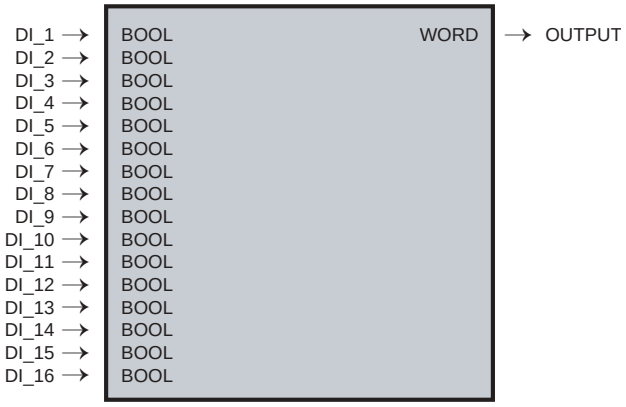
Label	Default values
DI_1... DI_8	FALSE

Output	Default values
OUTPUT	0

1-3-16 WORD D/A CONVERTER

FB Prototype

CONV_DA16



Input parameters

Label	Type	Description
DI_1... DI_16	BOOL	16 bits digital input

Output parameters

Label	Type	Description
OUTPUT	WORD	word representation of converted digital input

Description

This function block converts 16 input bits to an output value.

Default values

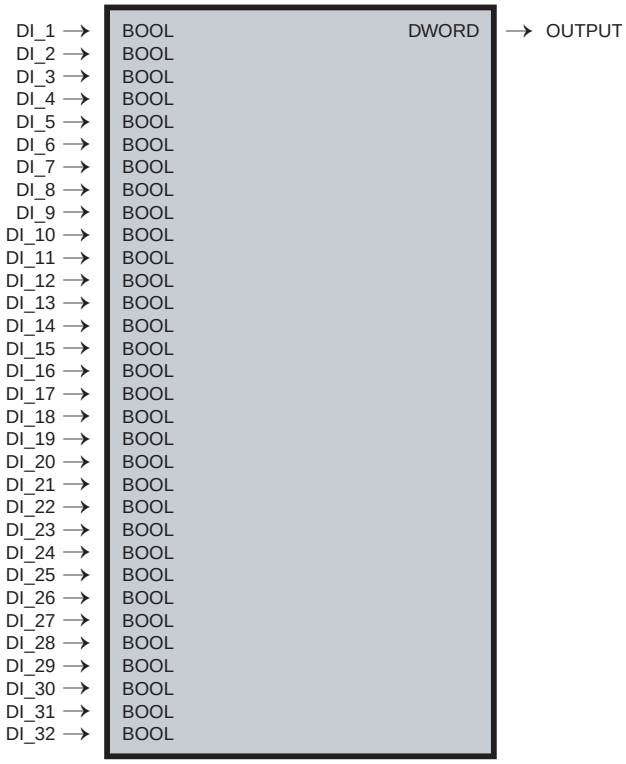
iNPUT	Default values
DI_1... DI_16	FALSE

Output	Default values
OUTPUT	0

1-3-17 DWORD D/A CONVERTER

FB Prototype

CONV_DA32



Input parameters

Label	Type	Description
DI_1... DI_32	BOOL	32 bits digital input

Output parameters

Label	Type	Description
OUTPUT	DWORD	dword representation of converted digital input

Description

This function block converts 32 input bits to an output value.

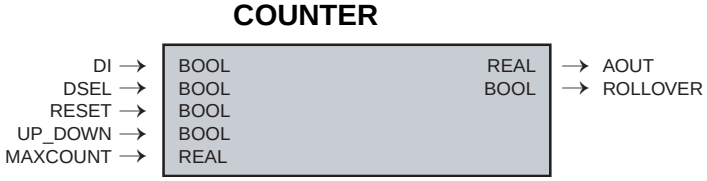
Default values

Label	Default values
DI_1... DI_32	FALSE

Output	Default values
OUTPUT	0

1-3-18 COUNTER

FB Prototype



Input parameters

Label	Type	Description	Range
DI	BOOL	Counter input	
DSEL	BOOL	Counter enable	
RESET	BOOL	Counter reset	
UP_DOWN	BOOL	Up/down count	FALSE = up, TRUE = down
MAXCOUNT	REAL	Maximum limit for the output	max. value = 999999.0

Output parameters

Label	Type	Description
AOUT	REAL	Number of transitions of DI since last reset
ROLLOVER	BOOL	TRUE if *AOUT = MAXCOUNT*

Description

This function block counts the number of the input transitions (FALSE to TRUE transitions). The counter works if DSEL is active and holds the value when DSEL is inactive. If RES is active the counter is reset. When the AOUT reaches the MAX-COUNT value the counter is reset, and the output ROLLOVER is TRUE for a cycle.

Default values

Input	Default values
DI	FALSE
DSEL	FALSE
RESET	FALSE
UP_DOWN	FALSE
MAXCOUNT	999999

Output	Default values
AOUT	0.0
ROLLOVER	FALSE

1-3-19 DIGITAL DECODER

FB Prototype

DECODER_8

Input parameters

Label	Type	Description
DS_1	BOOL	Digital selection input
DS_2	BOOL	Digital selection input
DS_3	BOOL	Digital selection input
D_SEL	BOOL	Decoder enable input
RESET	BOOL	Decoder reset input

Output parameters

Label	Type	Description
DO_1...DO_8	BOOL	Digital output 1... 8

Description

This function block selects one of 8 digital outputs, according to binary selection applied on inputs *DS_1*,*DS_2* and *DS_3*. *D_SEL* enables the decoder. Disabling *D_SEL*, the output holds the previous value. *RESET* will reset all the conditions.

Digital selection

When *D_SEL* is active, the input binary value is decoded as follows.

DS_3	DS_2	DS_1	OUT
FALSE	FALSE	FALSE	DO_1
FALSE	FALSE	TRUE	DO_2
FALSE	TRUE	FALSE	DO_3
FALSE	TRUE	TRUE	DO_4
TRUE	FALSE	FALSE	DO_5
TRUE	FALSE	TRUE	DO_6
TRUE	TRUE	FALSE	DO_7
TRUE	TRUE	TRUE	DO_8

Default values

Input	Default values
DS_1... DS3	FALSE
D_SEL	FALSE
RESET	FALSE

Output	Default values
DO_1... DO8	FALSE

1-3-20 DEW_POINT

FB Prototype

DEW_POINT

Input parameters

Label	Type	Description	Range
ENABLE	BOOL	Command to enable average calculation	
SEL_C_F	BOOL	Temperature engineering unit [°C or °F]	
TEMP	REAL	Temperature input value [in °C or °F]	LOW_LIM... HIGH_LIM
HR	REAL	Relative Humidity input value [%]	0.0... 100.0
HIGH_LIM	REAL	High limit for the AI channel value [e.u.]	$-3.4E^{-38}$... $3.4E^{+38}$
LOW_LIM	REAL	Low limit for the AI channel value [e.u.]	$-3.4E^{-38}$... $3.4E^{+38}$

Output parameters

Label	Type	Description	Description
ERROR	BOOL	Error status	
ERR_CODE	DWORD	Error code [bit mask]	6#00 00 00 00... 16#FF FF FF FF
DP_VALUE	REAL	Dew Point calculated value [°C or F]	LOW_LIM... HIGH_LIM

Description

This function block performs the Dew Point calculation by using temperature (TEMP) and relative humidity (HR) valid values. A “valid value” means TEMP input value is inside the range defined by LOW_LIM HIGH_LIM parameters and HR value in between 0.0 100.0 %, otherwise they will be not computed. The function block returns a temperature value accordingly to the SEL_C_F selection.

Default Values Table

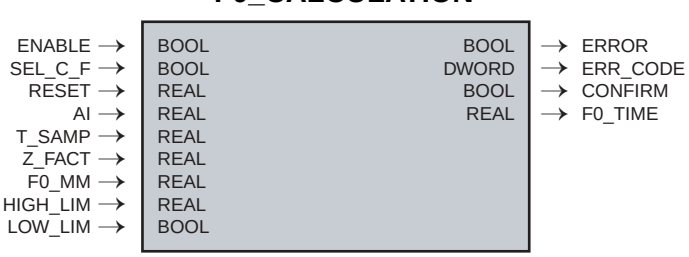
Input	Default Value
ENABLE	FALSE
SEL_C_F	FALSE = °C Degrees
TEMP	25.0
HR	50.0
HIGH_LIM	999.0
LOW_LIM	-99.0

Reference Table

Output	Description
ERR_CODE.0	Temperature value lower than the admitted value
ERR_CODE.1	Temperature value higher than the admitted value
ERR_CODE.2	Relative Humidity value lower than the admitted value
ERR_CODE.3	Relative Humidity value higher than the admitted value

1-3-21 F0_CALCULATION

FB Prototype



Input parameters

Label	Type	Description	Range
ENABLE	BOOL	Command to enable calculation	
SEL_C_F	BOOL	Temperature engineering unit [°C or °F]	
RESET	BOOL	Function 0 time calculation reset	
AI	REAL	Sterilization temperature input value [°C or °F]	LOW_LIM... HIGH_LIM
T_SAMP	REAL	AI temperature sampling time [ss]	0.1... 600.0
Z_FACT	REAL	Z temperature coefficient [°C or °F]	°C: 0.1... 25.0, °F: 32.18... 77.0
F0_MM	REAL	Target F0 duration time value [mm]	0.1... 300.0
HIGH_LIM	REAL	High limit for the AI channel value [e.u.]	$-3.4E^{-38}$... $3.4E^{+38}$
LOW_LIM	REAL	Low limit for the AI channel value [e.u.]	$-3.4E^{-38}$... $3.4E^{+38}$

Output parameters

Output	Type	Description	Range
ERROR	BOOL	Error status	
ERR_CODE	DWORD	Error code [bit mask]	16#00 00 00 00… 16#FF FF FF FF)
CONFIRM	BOOL	F0 target value duration confirmation	
F0_TIME	REAL	F0 time online calculation [mm]	0.0... $3.4E^{+38}$

Description

This function block performs the calculation of the “Function of 0 (zero)” needed during sterilization processes in order to reduce the bacterial load of a products. The calculation is a function of temperature variations and time. The temperature value is validated which means it has to be inside the range defined by LOW_LIM HIGH_LIM parameters, otherwise they will be not computed. The parameters T_SAMP, Z_FACT and F0_MM are also validated by specific limits. The function block returns also a confirmation status when the desired F0 duration is reached. In case of error, the calculation is suspended. Please refer to more detailed specific documentation can be find on internet.

Default Values

Input	Default Value
ENABLE	FALSE
SEL_C_F	FALSE = Celsius degrees
RESET	FALSE
AI	25.0
T_SAMP	5.0
Z_FACT	10.0

Input	Default Value
F0_MM	10.0
HIGH_LIM	999.0
LOW_LIM	-99.0

Reference Table

Output	Description
ERR_CODE.0	Sampling time value lower than the admitted value
ERR_CODE.1	Sampling time value higher than the admitted value
ERR_CODE.2	Sterilization temperature value lower than the admitted value
ERR_CODE.3	Sterilization temperature value higher than the admitted value
ERR_CODE.4	Temperature coefficient value lower than the admitted value
ERR_CODE.5	Temperature coefficient value higher than the admitted value
ERR_CODE.6	F0 duration time value higher than the admitted value
ERR_CODE.7	F0 duration time value higher than the admitted value
ERR_CODE.8	Cycle time value higher than sampling time

1-3-22 FLIP FLOP D

FB Prototype

FLIPFLOP_D

Input parameters

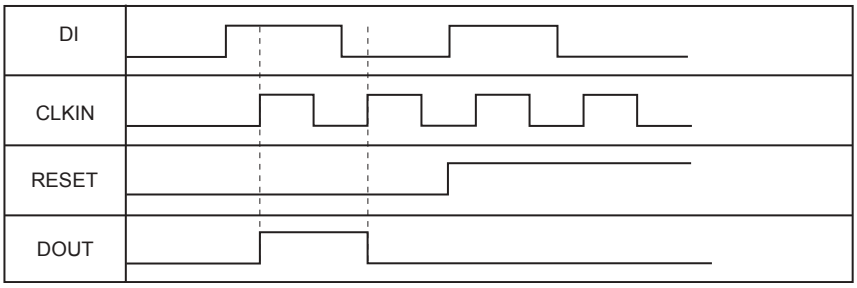
Label	Type	Description
DI	BOOL	Data input
CLKIN	BOOL	Clock input
RESET	BOOL	Reset input

Output parameters

Label	Type	Description
DOUT	BOOL	Digital output

Description

This function block performs the D-Latch flip flop function, as shown in following diagram.



Default values

Input	Default values
DI	FALSE
CLKIN	FALSE
RESET	FALSE

Output	Default values
DOUT	FALSE

1-3-23 FLIP FLOP JK

FB Prototype

FLIPFLOP_JK

Input parameters

Label	Type	Description
DI_J	BOOL	Data input J
DI_K	BOOL	Data input K

Output parameters

Label	Type	Description
DOUT	BOOL	Digital output

Description

This function block performs the JK-Latch flip flop function, as described in following table.

DI_J	DI_K	DOUT
FALSE	FALSE	No Toggle
FALSE	TRUE	FALSE
TRUE	FALSE	TRUE
TRUE	TRUE	Toggle

Default values

Input	Default values
DI_J	FALSE
DI_K	FALSE

Output	Default values
DOUT	FALSE

1-3-24 HOLD VALUE

FB Prototype

HOLD_VALUE

Input parameters

Label	Type	Description
AI	REAL	Input Value
D_HLD	BOOL	Hold Command

Output parameters

Label	Type	Description
AO UT	REAL	Function Block Output

Description

This FB has the functionality of a Sample and Hold circuit. It latches the value of the analog input *AI* on the output *AOUT*, in correspondence of the rising edge of the latch digital input *D_HLD*. When *D_HLD* is inactive, the value of the output *AOUT* is identical with the signal at the input AI, that is *AO = AI*. When *D_HLD* goes active, the *AOUT* output is latched and this value is hold until the *D_HLD* signal goes inactive again.

Default values

Input	Default values
AI	0.0
D_HLD	FALSE

Output	Default values
AO UT	0.0

1-3-25 HR_DRY_WET_BULB

FB Prototype

HR_DRY_WET_BULB

Input parameters

Input	Type	Description	Range
ENABLE	BOOL	Command to enable calculation	
SEL_C_F	BOOL	Temperature engineering unit [°C or °F]	
TEMP_DRY	REAL	Dry bulb temperature input value [°C or F]	LOW_LIM... HIGH_LIM
TEMP_WET	REAL	Wet bulb temperature input value [°C or F]	LOW_LIM... HIGH_LIM
P_ATM	REAL	Atmospheric pressure [kPa]	0.0... 1000.0
HIGH_LIM	REAL	High limit for the AI channel value [e.u.]	-3.4E-38... $3.4E^{+38}$
LOW_LIM	REAL	Low limit for the AI channel value [e.u.]	$-3.4E^{-38}$... $3.4E^{+38}$

Output parameters

Output	Type	Description	Range
ERROR	BOOL	Error status	
ERR_CODE	DWORD	Error code [bit mask]	16#00 00 00 00… 16#FF FF FF FF)
HR_VALUE	REAL	Relative Humidity calculated value [%]	0.0... 100.0
PSAT_DRY	REAL	Water vapour dry bulb Saturation Pressure calculated value [Pa]	$-3.4E^{-38}$... $3.4E^{+38}$
PSAT_WET	REAL	Water vapour wet bulb Saturation Pressure calculated value [Pa]	$-3.4E^{-38}$... $3.4E^{+38}$
ST_PART_P	REAL	Steam Partial Pressure calculated value [Pa]	$-3.4E^{-38}$... $3.4E^{+38}$

Description

This function block performs the Relative Humidity (HR) calculation by using the dry and wet bulb method. The calculation consider the difference of temperature between the TEMP_DRY and TEMP_WET valid input values which means they have to be inside the range defined by LOW_LIM HIGH_LIM parameters, otherwise they will be not computed. The calculation take care also about the atmospheric pressure in order to perform the right compensation. The function block returns also some other information used in the calculation like the saturation pressure for both the temperature values and the steam partial pressure.

Default Values Table

Input	Default Value
ENABLE	FALSE
SEL_C_F	FALSE = °C Degrees
TEMP_DRY	25.0

Input	Default Value
TEMP_WET	25.0
P_ATM	101.3 (value referred to the sea level)
HIGH_LIM	999.0
LOW_LIM	-99.0

Reference Table

Output	Description
ERR_CODE.0	Wet bulb temperature value lower than the admitted value
ERR_CODE.1	Wet bulb temperature value higher than the admitted value
ERR_CODE.2	Dry bulb temperature value lower than the admitted value
ERR_CODE.3	Dry bulb temperature value higher than the admitted value
ERR_CODE.4	Relative Humidity value lower than the admitted value
ERR_CODE.5	Relative Humidity value higher than the admitted value

1-3-26 IN BETWEEN

FB Prototype

INBETWEEN

Input parameters

Label	Type	Description
AI_1... AI_3	REAL	Analog input n. 1, 2 and 3

Output parameters

Label	Type	Description
AOUT	REAL	Middle value

Description

This function block transfers to the output *AOUT* the input value that is in between the other 2 input values.

Default values

Input	Default values
AI_1... AI_3	0.0

Output	Default values
AO UT	0.0

1-3-27 ANALOG LIMITER

FB Prototype

LIMITER_VALUE

Input parameters

Label	Type	Description
AI	REAL	Input value
HIGH	REAL	High limit
LOW	REAL	Low limit

Output parameters

Label	Type	Description
AOUT	REAL	Limited value

Description This function block limits the output value between high and low limits.

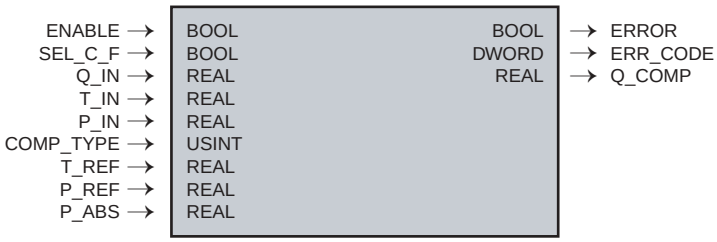
Default values

Input	Default values
AI	0.0
HIGH	100.0
LOW	0.0

Output	Default values
AOUT	0.0

1-3-28 MASS_FLOW

FB Prototype



Input parameters

Label	Type	Description	Range
ENABLE	BOOL	Command to ENABLE/DISABLE the FB execution	
SEL_C_F	BOOL	Temperature engineering unit [°C or °F]	
Q_IN	REAL	Actual Flow input value [e.u.]	$-3.4E^{-38}$... $3.4E^{+38}$
T_IN	REAL	Actual Temperature value [°C or °F]	$-3.4E^{-38}$... $3.4E^{+38}$
P_IN	REAL	Actual Pressure value [Bar]	$-3.4E^{-38}$... $3.4E^{+38}$
COMP_TYPE	USINT	Type of Compensation [num]	0... 3
T_REF	REAL	Reference Temperature value [°C or °F]	$-3.4E^{-38}$... $3.4E^{+38}$
P_REF	REAL	Reference Pressure value [Bar]	$-3.4E^{-38}$... $3.4E^{+38}$
P_ABS	REAL	Absolute Pressure value [Bar]	$-3.4E^{-38}$... $3.4E^{+38}$
Q_HIGH_LIM	REAL	Actual Flow High limit value [Bar]	$-3.4E^{-38}$... $3.4E^{+38}$
Q_LOW_LIM	REAL	Actual Flow Low limit value [Bar]	$-3.4E^{-38}$... $3.4E^{+38}$

Output parameters

Output	Type	Description	Range
ERROR	BOOL	Error status	
ERR_CODE	DWORD	Error code [bit mask]	16#00 00 00 00… 16#FF FFFF FF)
Q_COMP	REAL	Compensate Flow value [e.u.]	$-3.4E^{-38}$... $3.4E^{+38}$

Description

This function block can perform the compensation of the valid Fluid Flow input value by temperature or pressure or both. Fluid Flow valid input value means it has to be inside the range defined by Q_LOW_LIM... Q_HIGH_LIM parameters, otherwise it will be not computed.

Note: All the pressure values (P_IN, P_REF and P_ABS) **MUST BE SPECIFIED** as **Bar** engineering units so the value must be converted in advance (e.g. in case of kPa [kilo Pascal] 1 kPa = 0.01 Bar or in case of psi 1 psi = 0.068947573 Bar). The function block evaluate the temperature values accordingly to the SEL_C_F selection.

Default Values Table

Input	Default Value
ENABLE	FALSE
SEL_C_F	FALSE = Celsius degrees
Q_IN	10.0
T_IN	100.0

Input	Default Value
P_IN	15.0
COMP_TYPE	0 = no compensation, 1 = temperature compensation only, 2 = pressure compensation only, 3 = both temperature and pressure compensation
T_REF	25.0
P_REF	30.0
P_ABS	1.013
Q_HIGH_LIM	100.0
Q_LOW_LIM	0.0

Reference Table

Output	Description
ERR_CODE.0	Compensation type selection not valid
ERR_CODE.1	Actual Temperature value underrange (to avoid division by zero)
ERR_CODE.2	Actual flow value out of range
ERR_CODE.3	Reference and Absolute Pressure values equal to 0

1-3-29 MIN MAX SELECTOR

FB Prototype

MIN_MAX_SELECTOR

Input parameters

Label	Type	Description
AI_1 ... AI_4	REAL	Analog input value n. 1, 2, 3 and 4
N_AI	UINT	Number of inputs to be compared (2, 3, 4)

Output parameters

Label	Type	Description
AI_MIN	REAL	Minimum input value
AI_MAX	REAL	Maximum input value

Description

This function block returns the minimum and maximum values of the applied inputs.

Default values

Input	Default values
AI_1... AI_4	0.0
N_AI	0

Output	Default values
AI_MIN	0.0
AI_MAX	0.0

1-3-30 MONOSTABLE DELAY SHIFTED

FB Prototype

MONOSTABLE_DS

Input parameters

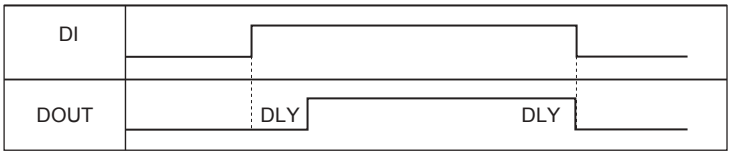
Label	Type	Description
DI	BOOL	Digital input
RESET	BOOL	Reset input
DELAY	DINT	Monostable delay (ms)

Output parameters

Label	Type	Description
DOUT	BOOL	Digital output

Description

This function block provides a time delay between input and output signals, as shown in the following diagram:



Default values

Input	Default values
DI	FALSE
RESET	FALSE
DELAY	0

Output	Default values
DOUT	FALSE

1-3-31 MONOSTABLE NEGATIVE EDGE DELAY

FB Prototype

MONOSTABLE_NED

Input parameters

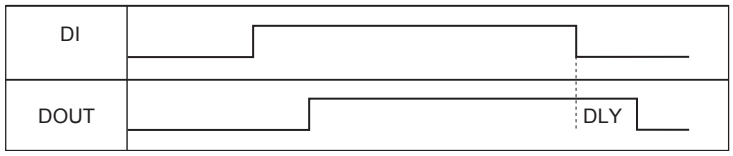
Label	Type	Description
DI	BOOL	Digital input
RESET	BOOL	Reset input
DELAY	DINT	Monostable delay (ms)

Output parameters

Label	Type	Description
DOUT	BOOL	Digital output

Description

This function block provides a time delay between input and output signals, as shown in the following diagram:



Default values

Input	Default values
DI	FALSE
RESET	FALSE
DELAY	0

Output	Default values
DOUT	FALSE

1-3-32 MONOSTABLE POSITIVE EDGE DELAY

FB Prototype

MONOSTABLE_PED

Input parameters

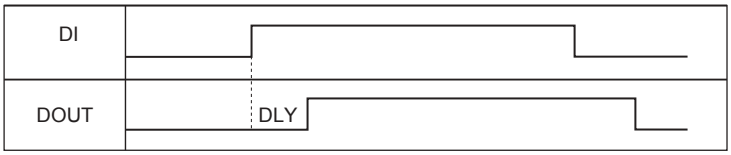
Label	Type	Description
DI	BOOL	Digital input
RESET	BOOL	Reset input
DELAY	DINT	Monostable delay (ms)

Output parameters

Label	Type	Description
DOUT	BOOL	Digital output

Description

This function block provides a time delay between input and output signals, as shown in the following diagram:



Default values

Input	Default values
DI	FALSE
RESET	FALSE
DELAY	0

Output	Default values
DOUT	FALSE

1-3-33 MONOSTABLE PULSE

FB Prototype

MONOSTABLE_PUL

Input parameters

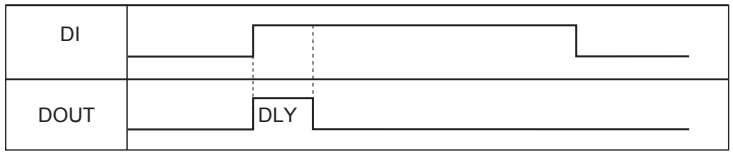
Label	Type	Description
DI	BOOL	Digital input
RESET	BOOL	Reset input
DELAY	DINT	Monostable delay (ms)

Output parameters

Label	Type	Description
DOUT	BOOL	Digital output

Description

This function block provides a time delay between input and output signals, as shown in the following diagram:



Default values

Input	Default values
DI	FALSE
RESET	FALSE
DELAY	0

Output	Default values
DOUT	FALSE

1-3-34 RTD LINEAR RESCALING (FOR MICROPAC ONLY)

FB Prototype

MP_RTD_LIN

Input parameters

Label	Type	Description	Range
AI	REAL	Analogue input value. [mA]	4.0... 20.0
AI_M	REAL	Gain coefficient: a specific value for each channel is specified on the documentation [num]	0.0... 90.0
AI_Q	REAL	Offset coefficient in micro Ampere: a specific value for each channel is specified on the documentation [µA]	-0.999999... 0.999999
SEL_C_F	BOOL	Degree conversion selectioN	°C/°F
TAU	REAL	Filter time constant: a value of 0.0 is intended to disable the filter action and the output will assume the input value [s]	Reasonable range 0.0... 60.0

Output parameters

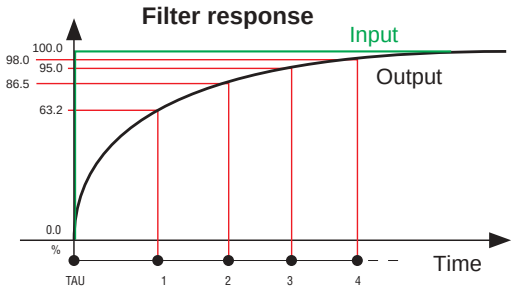
Label	Type	Description	Range
RTD_OUT	REAL	Converted RTD value	

Description

This function block returns a linearization of the input value as function of a gain and an offset value calculated as the following equation:

RTD_OUT = AI * AI_M + AI_Q

The FB then performs a first order filter on the converted value. Please note that the filter constant TAU has to be greater than the cycle time used for the task where the FB is used.



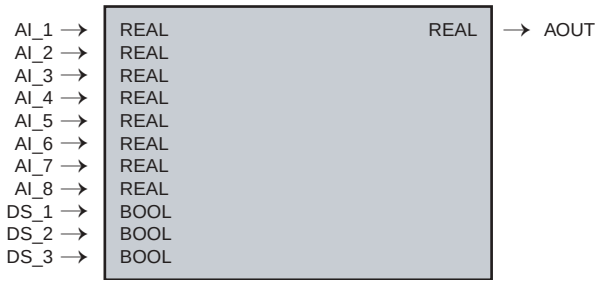
Default values

Label	Default Value
AI	4.0
AI_M	1.0
AI_Q	0.0
C_F_SEL	FALSE = °C degrees
TAU	0.1

1-3-35 ANALOG MULTIPLEXER 8 CHANNELS

FB Prototype

MUX_A8



Input parameters

Input	Type	Description
AI_1... AI_8	REAL	Analog Input 1... 8
DS_1	BOOL	Digital selection input
DS_2	BOOL	Digital selection input
DS_3	BOOL	Digital selection input

Output parameters

Label	Type	Description
AOUT	REAL	Analog Output

Description

This function block is an 8 to 1 multiplexer. One of the 8 inputs AIN_1... AIN_8, selected by mean of the 3 bit code on the *DS_1... DS_3* inputs, is retransmitted on the *AOUT* output.

Selection Table

DS_3	DS_2	DS_1	AOUT
FALSE	FALSE	FALSE	AI_1
FALSE	FALSE	TRUE	AI_2
FALSE	TRUE	FALSE	AI_3
FALSE	TRUE	TRUE	AI_4
TRUE	FALSE	FALSE	AI_5
TRUE	FALSE	TRUE	AI_6
TRUE	TRUE	FALSE	AI_7
TRUE	TRUE	TRUE	AI_8

Default values

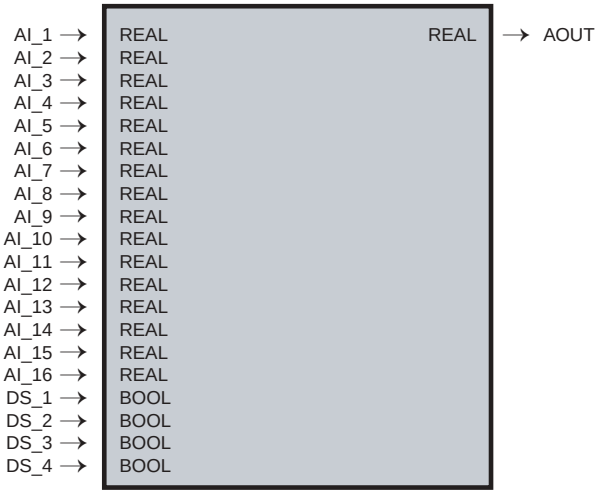
Input	Default values
AI_1... AI_8	0.0
DS_1... DS_3	FALSE

Output	Default values
AOUT	0.0

1-3-36 ANALOG MULTIPLEXER 16 CHANNELS

FB Prototype

MUX_A16



Input parameters

Label	Type	Description
AI_1... AI_16	REAL	Analog Input 1... 16
DS_1	BOOL	Digital selection input
DS_2	BOOL	Digital selection input
DS_3	BOOL	Digital selection input
DS_4	BOOL	Digital selection input

Output parameters

Label	Type	Description
AOUT	REAL	Analog Output

Description

This function block is an 16 to 1 multiplexer. One of the 16 inputs *AIN_1... AIN_16*, selected by mean of the 4 bit code on the *DS_1...DS_4* inputs, is retransmitted on the *AOUT* output.

Selection Table

DS_4	DS_3	DS_2	DS_1	AOUT
FALSE	FALSE	FALSE	FALSE	AI_1
FALSE	FALSE	FALSE	TRUE	AI_2
FALSE	FALSE	TRUE	FALSE	AI_3
FALSE	FALSE	TRUE	TRUE	AI_4
FALSE	TRUE	FALSE	FALSE	AI_5
FALSE	TRUE	FALSE	TRUE	AI_6
FALSE	TRUE	TRUE	FALSE	AI_7
FALSE	TRUE	TRUE	TRUE	AI_8
TRUE	FALSE	FALSE	FALSE	AI_9
TRUE	FALSE	FALSE	TRUE	AI_10
TRUE	FALSE	TRUE	FALSE	AI_11
TRUE	FALSE	TRUE	TRUE	AI_12
TRUE	TRUE	FALSE	FALSE	AI_13
TRUE	TRUE	FALSE	TRUE	AI_14

DS_4	DS_3	DS_2	DS_1	AOUT
TRUE	TRUE	TRUE	FALSE	AI_15
TRUE	TRUE	TRUE	TRUE	AI_16

Default values

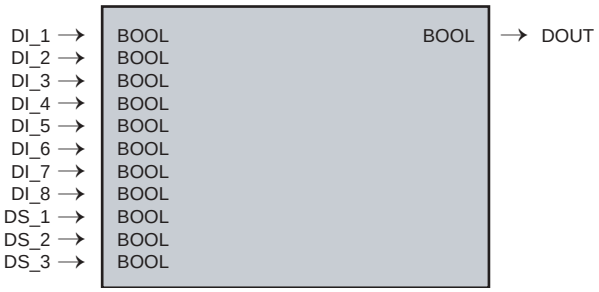
Input	Default values
AI_1... AI_16	0.0
DS_1... DS_4	FALSE

Output	Default values
AOUT	0.0

1-3-37 DIGITAL MULTIPLEXER 8 CHANNELS

FB Prototype

MUX_D8



Input parameters

Label	Type	Description
DI_1... DI_8	BOOL	Digital Input 1... 8
DS_1	BOOL	Digital selection input
DS_2	BOOL	Digital selection input
DS_3	BOOL	Digital selection input

Output parameters

Label	Type	Description
DOUT	BOOL	Digital OutputM

Description

This function block is an 8 to 1 multiplexer. One of the 8 inputs DIN_1... DIN_8, selected by mean of the 3 bit code on the *DS_1... DS_3* inputs, is retransmitted on the *DOUT* output.

Selection Table

DS_3	DS_2	DS_1	DOUT
FALSE	FALSE	FALSE	DI_1
FALSE	FALSE	TRUE	DI_2
FALSE	TRUE	FALSE	DI_3
FALSE	TRUE	TRUE	DI_4
TRUE	FALSE	FALSE	DI_5
TRUE	FALSE	TRUE	DI_6
TRUE	TRUE	FALSE	DI_7
TRUE	TRUE	TRUE	DI_8

Default values

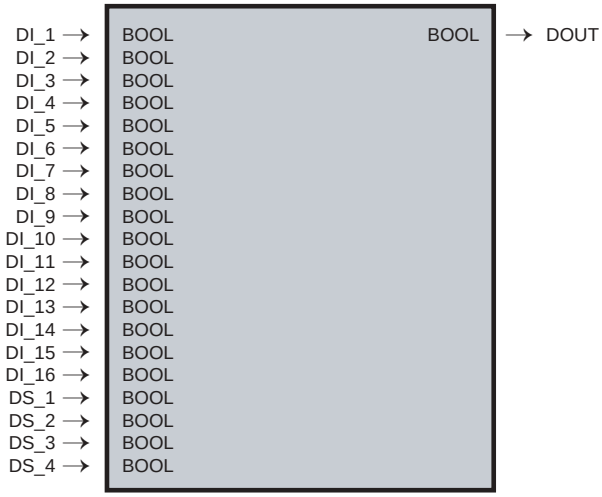
Input	Default values
DI_1... DI_8	FALSE
DS_1... DS_3	FALSE

Output	Default values
DOUT	FALSE

1-3-38 DIGITAL MULTIPLEXER 16 CHANNELS

FB Prototype

MUX_D16



Input parameters

Label	Type	Description
DI_1... DI_16	BOOL	Digital Input 1... 16
DS_1	BOOL	Digital selection input
DS_2	BOOL	Digital selection input
DS_3	BOOL	Digital selection input
DS_4	BOOL	Digital selection input

Output parameters

Label	Type	Description
DOUT	BOOL	Digital Output

Description

This function block is an 16 to 1 multiplexer. One of the 16 inputs *DIN_1... DIN_16*, selected by mean of the 4 bit code on the *DS_1... DS_4* inputs, is retransmitted on the *DOUT* output.

Selection Table

DS_4	DS_3	DS_2	DS_1	DOUT
FALSE	FALSE	FALSE	FALSE	DI_1
FALSE	FALSE	FALSE	TRUE	DI_2
FALSE	FALSE	TRUE	FALSE	DI_3
FALSE	FALSE	TRUE	TRUE	DI_4
FALSE	TRUE	FALSE	FALSE	DI_5
FALSE	TRUE	FALSE	TRUE	DI_6
FALSE	TRUE	TRUE	FALSE	DI_7
FALSE	TRUE	TRUE	TRUE	DI_8
TRUE	FALSE	FALSE	FALSE	DI_9
TRUE	FALSE	FALSE	TRUE	DI_10
TRUE	FALSE	TRUE	FALSE	DI_11
TRUE	FALSE	TRUE	TRUE	DI_12
TRUE	TRUE	FALSE	FALSE	DI_13
TRUE	TRUE	FALSE	TRUE	DI_14
TRUE	TRUE	TRUE	FALSE	DI_15

Default values

Input	Default values
DI_1...D_16	FALSE
DS_1... DS_4	FALSE

Output	Default values
DOUT	FALSE

1-3-39 RESCALE

FB Prototype

Input parameters

Label	Type	Description
AI	REAL	Analog input value
IN_HIGH	REAL	High Scale Analog input value
IN_LOW	REAL	Low Scale Analog input value
OUT_HIGH	REAL	High Scale Analog output value
OUT_LOW	REAL	Low Scale Analog output value

Output parameters

Label	Type	Description
AOUT	REAL	Analog output value
ERR	BOOL	Error : scaling not possible
OVER	BOOL	Over range: active if *AI > IN_HIGH*
UNDER	BOOL	Under range: active if *AI < IN_LOW*

Description This function block rescale the Analog input value in according to the specified input and output range.

Default values

Input	Default values
AI	0.0
IN_HIGH	100.0
IN_LOW	0.0
OUT_HIGH	100.0
OUT_LOW	0.0

Output	Default values
AOUT	0.0
ERR	FALSE
OVER	FALSE
UNDER	FALSE

1-3-40 SAMPLING_TIME

FB Prototype

SAMPLING_TIME

Input parameters

Label	Type	Description
ENABLE	BOOL	Command to ENABLE/DISABLE the FB execution

Output parameters

Label	Type	Description
TSAM_MAX	TIME	Max. cycle time reached by the application from the power ON
TSAM_ACT	TIME	Last actual cycle time of the application
TSAM_MIN	TIME	Min. cycle time reached by the application from the power ON

Description

This function block provides real – time and statistic information on the application actual cycle time, maximum and minimum cycle time reached during normal operations. Of course, it should not be used in a timer task otherwise you will get the task execution time.

Default Values Table

Input	Default Value
ENABLE	FALSE

1-3-41 SLOPE LIMIT

FB Prototype

SLOPE_LIMIT

Input parameters

Label	Type	Description
ENABLE	BOOL	Enable Slope Function
AI	REAL	Analog input value
SL_UP	REAL	Slope Up Analog Value
SL_DOWN	REAL	Slope Down Analog Value

Output parameters

Label	Type	Description
AOUT	REAL	Analog Output Value conditionated by the Slopes

Description

This function block limit the Output value rate of change in according to the specified *SL_UP* and *SL_DOWN* values.

Default values

Input	Default values
ENABLE	FALSE
AI	0.0
SL_UP	1.0
SL_DOWN	1.0

Output	Default values
AOUT	0.0

1-3-42 TIMER_ADV

FB Prototype

TIMER_ADV

Input parameters

Label	Type	Description
START	BOOL	Digital command to start the Timer
RUN_HOLD	BOOL	Digital command to Run or Hold the Timer count
T_RESET	BOOL	Digital command to Reset the Timer execution
TIME_HH	REAL	Preset Timer value in hours
TIME_MM	REAL	Preset Timer value in minutes
TIME_SS	REAL	Preset Timer value in seconds

Output parameters

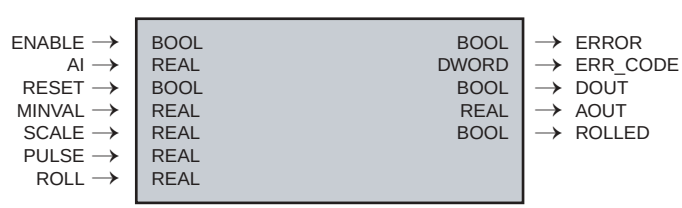
Label	Type	Description
START_STS	BOOL	Timer status
RUN_STS	BOOL	Timer Run status
HOLD_STS	BOOL	Timer Hold status
END_TIMER	BOOL	End Timer status
COUNT_HH	REAL	Timer countdown (hours)
COUNT_MM	REAL	Timer countdown (minutes)
COUNT_SS	REAL	Timer countdown (seconds)

Description

This FB performs a countdown Timer function with some advanced features. In detail, the START command (it is evaluated ONLY on the rising edge), basically, enable the execution ONLY of the FB but not yet the real count, it starts or stops only with the RUN_HOLD command. At any time, with the RESET input is possible to interrupt and reset the execution of the FB. Trough the digital output it is possible to know, at any time, the status and the amount of time at the end of the timer.

1-3-43 TOTALIZER

FB Prototype



Input parameters

Input	Type	Description	Range
ENABLE	BOOL	Command to enable the internal code execution	
AI	REAL	Analogue input value [e.u.]	Full range of the REAL data-type
RESET	BOOL	Digital reset	
MINVAL	REAL	Input dropout minimum value. If AI < MINVAL the output does not change [e.u.]	Full range of the REAL data-type
SCALE	REAL	Scale factor for input value [num]	0.01… 100.0
PULSE	REAL	When output reaches this value, DOUT generates a one cycle pulse [e.u.]	Range > MINVAL … < ROLL
ROLL	REAL	Rollover value. When output reaches roll, the output ROLLED became TRUE until a RESET command [e.u.]	Typical range > MINVAL and PULSE… 1000.0

Output parameters

Output	Type	Description	Range
ERROR	BOOL	Error status	
ERR_CODE	DWORD	Error code [bit mask]	16#00 00 00 00… 16#FF FF FF FF
DOUT	BOOL	Digital output which generates a pulse every time the calculation reaches the PULSE input value	
AOUT	REAL	Analogue output for totalized value [e.u.]	
ROLLED	BOOL	Digital output which stays TRUE until a RESET Command	

Description This function block integrates the input analog value in according to the sampling time. The equation for the FB is :

$$AOUT_{(t)} = AOUT_{(t-1)} + AI_{(t)} * SCALE * (T_{sample}/3600)$$

Note: In order to obtain a reasonable accuracy on the calculation, exceed a ratio of 10^6 of ROLL/IN_{SCL} where:

$$IN_{SCL} = AI * SCALE * (T_{sample}/3600$$

Default Values Table

Input	Default Value
ENABLE	FALSE
AI	0.0
RESET	FALSE
MINVAL	0.0

Input	Default Value
SCALE	1.0
PULSE	100.0
ROLL	1000.0

Reference Table

Output	Description
ERR_CODE.0	AI value lower than MINVAL
ERR_CODE.1	Invalid PULSE value (lower thanMINVAL or higher than ROLL)
ERR_CODE.2	Negative PULSE value

1-3-44 TOTALIZER_ADV

FB Prototype

TOTALIZER_ADV

Input parameters

Label	Type	Description	Range
ENABLE	BOOL	Command to enable the internal code execution	
AI	REAL	Analogue input value [e.u.]	Full range of the REAL data-type
RESET	BOOL	Digital reset	
MINVAL	REAL	Input dropout minimum value. If AI < MINVAL, the output does not change [e.u.] (full range of the REAL data-type)	Full range of the REAL data-type
SCALE	REAL	Scale factor for input value [num]	Typical range 0.01… 100.0
PULSE	REAL	When output reaches this value, DOUT generates a one cycle pulse [e.u.]	range > MINVAL… < ROLL
ROLL	REAL	Rollover value. When output reaches roll, the output ROLLED became TRUE until a RESET command [e.u.]	Typical range > MINVAL and PULSE… 1000.0
T_BASE	USINT	Integration time base. The value is evaluated ONLY while the ENABLE command is FALSE. It determines the time stamp which to refer the totalization value [num]	0 = e.u./Hours, 1 = e.u./Minutes and 2 = e.u./Seconds

Output parameters

Label	Type	Description	Range
ERROR	BOOL	Error status	
ERR_CODE	DWORD	Error code [bit mask]	16#00 00 00 00… 16#FF FF FF FF)
DOUT	BOOL	Digital output which generates a pulse every time the calcu-lation reaches the PULSE input value	
AOUT	REAL	Analogue output for totalized value [e.u.]	
ROLLED	BOOL	Digital output which stays TRUE until a RESET Command	

Description

This function block integrates the input analog value in according to the sampling time. The equation for the FB is:

$$AOUT_{(t)} = AOUT_{(t-1)} + AI_{(t)} * SCALE * (T_{sample}/T\_BASE)$$

Note: In order to obtain a reasonable accuracy on the calculation, exceed a ratio of 10^6 of ROLL/I_{NSCL} where IN_{SCL} = AI * SCALE * (T_{sample}/T_BASE)

Default Values Table

Label	Default Value
ENABLE	FALSE
AI	0.0
RESET	FALSE
MINVAL	0.0
SCALE	1.0
PULSE	100.0
ROLL	1000.0
T_BASE	0

Reference Table

Output	Description
ERR_CODE.0	AI value lower than MINVAL
ERR_CODE.1	Invalid PULSE value (lower than MINVAL or higher than ROLL)
ERR_CODE.2	Negative PULSE value

Chapter 2
AsconBasicIOLib

2-1 Purpose

The purpose of this document is to supply a complete description of **AsconBasicIOLib** Library project.

Caution
This library can be used **only** with the **CU-02** unit.

2-2 Description

AsconBasicIOLib is a Function Block library enabling the access of Ascon SpA sigma**due** line devices from *OpenPCS* programming environment. Thanks to the FBs, the user does not take care of communications details related to CANopen[1] protocol, but he simply has to access to the available functions of the individual devices. The library gives the access to the basic functions of the IO modules. For advanced functionalities of the sigma**due** modules see the *AsconAdvancedIOLib* documentation.

The following is the list of FBs present in the library.

aDM08TS	Advanced FB to interface the module DM-08TS
bDI16LV	Interface FB with DI-16LV module
bDI32LV	Interface FB with DI-32LV module
bDO04RL	Interface FB with DO-04RL module
bDO08RL	Interface FB with DO-08RL module
bDO04TX	Interface FB with DO-04TX module
bDO16TS	Interface FB with DO-16TS module
bDO16TP	Interface FB with DO-16TP module
bDO32TS	Interface FB with DO-32TS module
bDM08TS	Interface FB with DM-08TS module
bDM16TS	Interface FB with DM-16TS module
bDM32TS	Interface FB with DM-32TS module

1. A complete series of firmware function blocks implementing the CANopen communications primitives are available for the user: these FBs grant the access to the module directly through the CANopen protocol messages. For further details, see "*CANopen extension for IEC61131-3*" User Manual available on the installation CD.

bAI02UI	Interface FB with AI-02UI module
bAI04RT	Interface FB with AI-04RT module
bAI08TC	Interface FB with AI-08TC module
bAI08HL	Interface FB with AI-08HL module
bAI08DP	Interface FB with AI-08DP module
bAO08HL	Interface FB with AO-08HL module
bAO08DP	Interface FB with AO-08DP module
bERRORSTATEAN	FBs for error status management during communication for analogue output modules
bERRORSTATEDIG	FBs for error status management during communication for digital output modules

Each Function Block[1] has always the control signals:

Input description

Label	Description
NETWORK	It represents the physical bus connection
NODEID	Network identifier

Output description

Label	Description
CONFIRM	Boolean flag indicating whether the module is working properly

After specifying the node address through *NODEID* input, the block is enabled by forcing to 1 the bit in the position *NODEID* of the NETWORK input. Calling the Function Block with the corresponding bit to 1, a first configuration phase is initially completed and the real time module-CPU data exchange procedures are subsequently enabled. If the Function Block is called with enabling bit to 0, the internal status is reset: new calls with bit to 1 result in a new module configuration. CONFIRM output is set to TRUE when module initialisation/configuration is terminated and the communication messages including the input values and/or the output status are exchanged properly. For convenience and to reduce the number of used data types, the DWORD and REAL format are standardized for respectively digital and analogue modules.

2-2-1 Output module value setting in case of communication error

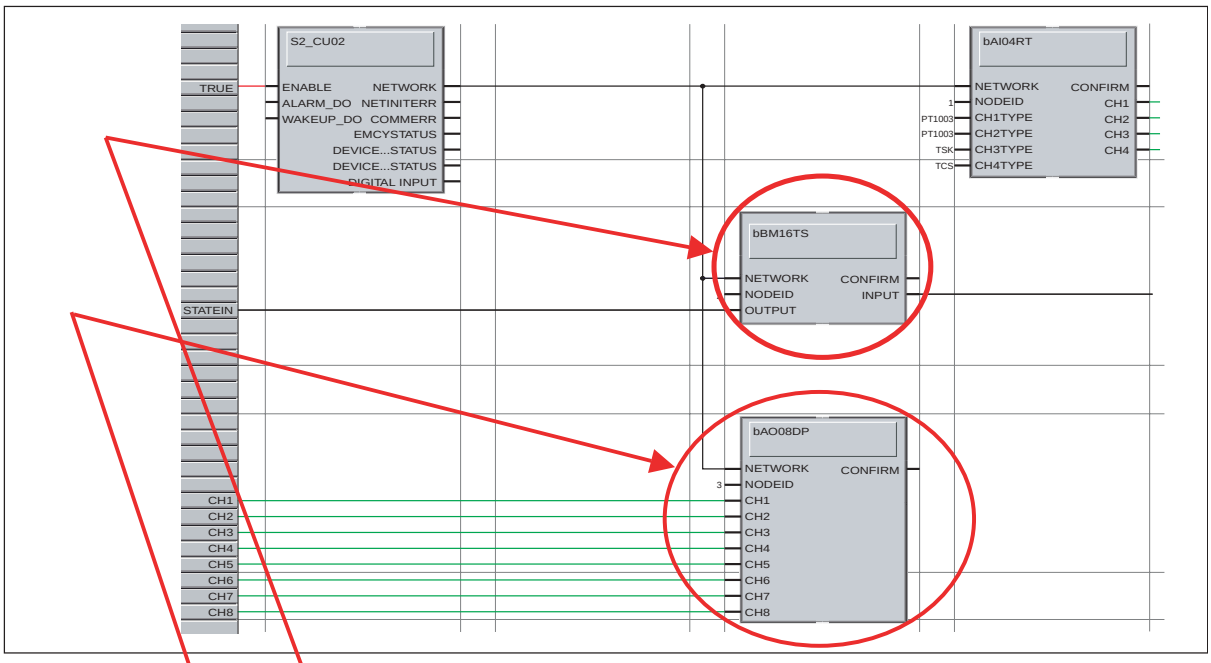
Each CAN output node of Ascon SpA sigma**due** series is capable of activating a particular value for each module output. These settings are not included in standard interface Function Block version and can be activated by using *bERROR-STATEAN* and *bERRORSTATEDIG* FBs: while *bERRORSTATEAN* configures the analogue module (*bAO08HL* and *bAO08DP*) behaviour, the second one configures the behaviour for all those digital modules with outputs (*aDM08TS, bDM08TS, bDM16TS, bDM32TS, bDO04RL, bDO08RL*, *bDO04TX, bDO16TS, bDO16TP* and *bDO32TS*). *For an effective control of communication-related error status, Ascon SpA recommends to use these configuration modules with Node-Guarding diagnostic system*. For further details on different enabable diagnostic types, see *CU-02* unit documentation.

With *NodeGuarding* diagnostic protocol activated, the default settings of Ascon SpA sigma**due** modules enable to force to zero the module outputs (digital and analogue) during communications error detection.This behaviour refers to Ascon SpA sigma**due** series module in *default* or *factory setting* status (for further details, see the User Manuals of each I/O module): *no guarantee is made for the modules with changed configuration*.

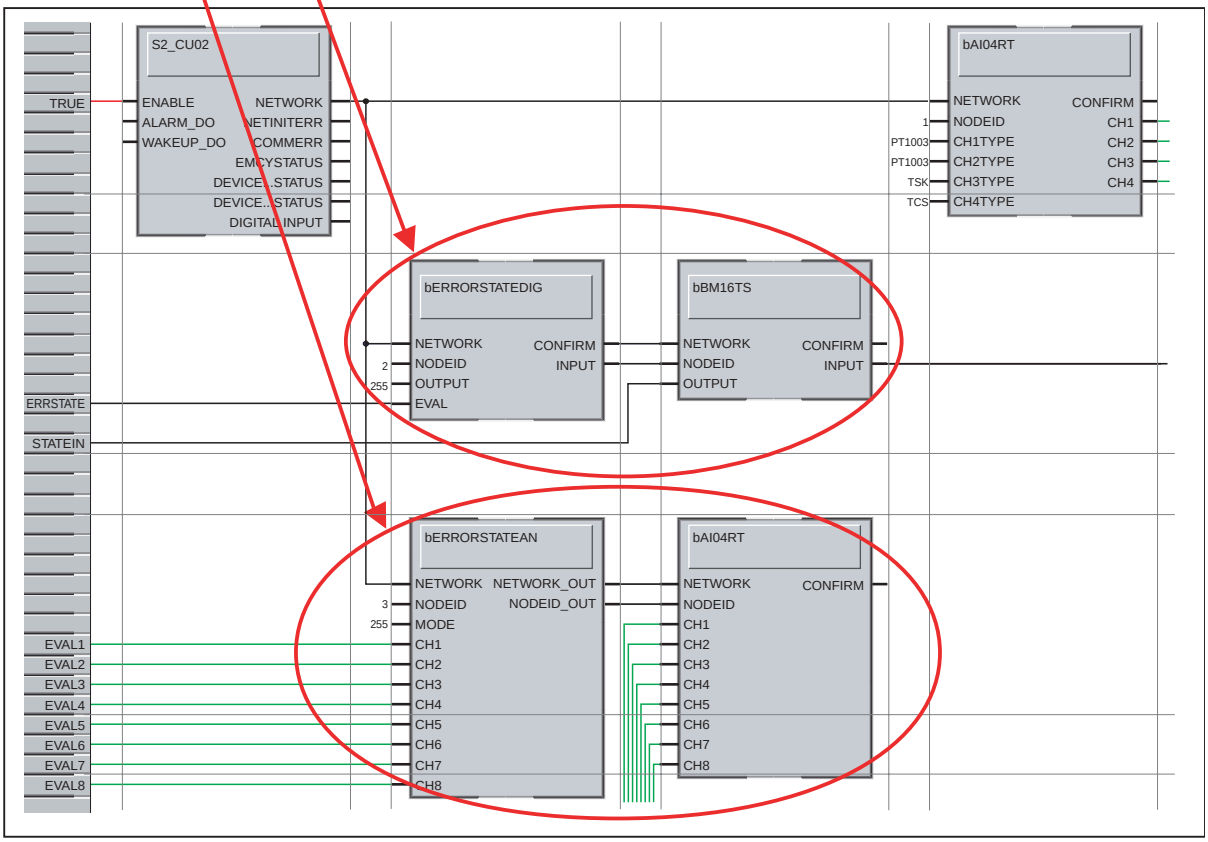
1. With the exception of bERRORSTATEAN and bERRORSTATEDIG modules.

The following is a network (CU-02 plus three modules) architecture definition, emphasizing the integration mode of the modules dedicated to the definition the output status in error conditions. The first figure shows the scheme with no use of these FBs; the second one shows also the output modules.

The figures show the entering mode of output status configuration FBs in case of communication error. Note that these FBs only set the necessary parameters to the error status definition: according to this configuration they achieve their tasks, leaving the complete control of the module to the interface FB.



Network standard architecture definition without status error definition



Network standard architecture definition with status error definition

2-3 Description of the individual Function Blocks

2-3-1 bDI16LV

FB Prototype

bDI16LV

Input parameters

Label	Type	Description
Network	DWORD	CAN Network Connection
NodeID	USINT	CANopen Device Identifier

Output parameters

Label	Type	Description
Confirm	BOOL	Function Block Confirm
Input	DWORD	Input State

Description

This Function Block acquires the input data from the module, and returns the value using the "Input" signal. As inputs the FB wants the network connection and the node identifier.

2-3-2 bDI32LV

FB Prototype

bDI32LV

Input parameters

Label	Type	Description
Network	DWORD	CAN Network Connection
NodeID	USINT	CANopen Device Identifier

Output parameters

Label	Type	Description
Confirm	BOOL	Function Block Confirm
Input	DWORD	Input State

Description

This Function Block acquires the input data from the module, and returns the value using the "Input" signal. As inputs the FB wants the network connection and the node identifier.

2-3-3 bDO04RL

FB Prototype

bDO04RL

Input parameters

Label	Type	Description
Network	DWORD	CAN Network Connection
NodeID	USINT	CANopen Device Identifier
Output	DWORD	Output State

Output parameters

Label	Type	Description
Confirm	BOOL	Function Block Confirm

Description

This Function Block sets the output of the module using the provided “Output” value signal. Furthermore the FB wants as inputs also the network connection and the node identifier.

2-3-4 bDO08RL

FB Prototype

bDO08RL

Input parameters

Label	Type	Description
Network	DWORD	CAN Network Connection
NodeID	USINT	CANopen Device Identifier
Output	DWORD	Output State

Output parameters

Label	Type	Description
Confirm	BOOL	Function Block Confirm

Description

This Function Block sets the output of the module using the provided “Output” value signal. Furthermore the FB wants as inputs also the network connection and the node identifier.

2-3-5 bDO04TX

FB Prototype

bDO04TX

Input parameters

Label	Type	Description
Network	DWORD	CAN Network Connection
NodeID	USINT	CANopen Device Identifier
Output	DWORD	Output State

Output parameters

Label	Type	Description
Confirm	BOOL	Function Block Confirm

Description

This Function Block sets the output of the module using the provided “Output” value signal. Furthermore the FB wants as inputs also the network connection and the node identifier.

2-3-6 bDO16TS

FB Prototype

bDO16TS

Input parameters

Label	Type	Description
Network	DWORD	CAN Network Connection
NodeID	USINT	CANopen Device Identifier
Output	DWORD	Output State

Output parameters

Label	Type	Description
Confirm	BOOL	Function Block Confirm

Description

This Function Block sets the output of the module using the provided “Output” value signal. Furthermore the FB wants as inputs also the network connection and the node identifier.

2-3-7 bDO16TP

FB Prototype

bDO16TP

Input parameters

Label	Type	Description
Network	DWORD	CAN Network Connection
NodeID	USINT	CANopen Device Identifier
Output	DWORD	Output State

Output parameters

Label	Type	Description
Confirm	BOOL	Function Block Confirm

Description

This Function Block sets the output of the module using the provided “Output” value signal. Furthermore the FB wants as inputs also the network connection and the node identifier.

2-3-8 bDO32TS

FB Prototype

bDO32TS

Input parameters

Label	Type	Description
Network	DWORD	CAN Network Connection
NodeID	USINT	CANopen Device Identifier
Output	DWORD	Output State

Output parameters

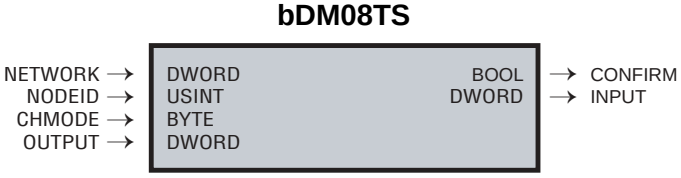
Label	Type	Description
Confirm	BOOL	Function Block Confirm

Description

This Function Block sets the output of the module using the provided “Output” value signal. Furthermore the FB wants as inputs also the network connection and the node identifier.

2-3-9 bDM08TS

FB Prototype



Input parameters

Label	Type	Description
Network	DWORD	CAN Network Connection
NodeID	USINT	CANopen Device Identifier
ChMode	BYTE	Channel Direction (default value = 0x00)
Output	DWORD	Output State

Output parameters

Label	Type	Description
Confirm	BOOL	Function Block Confirm
Input	DWORD	Input State

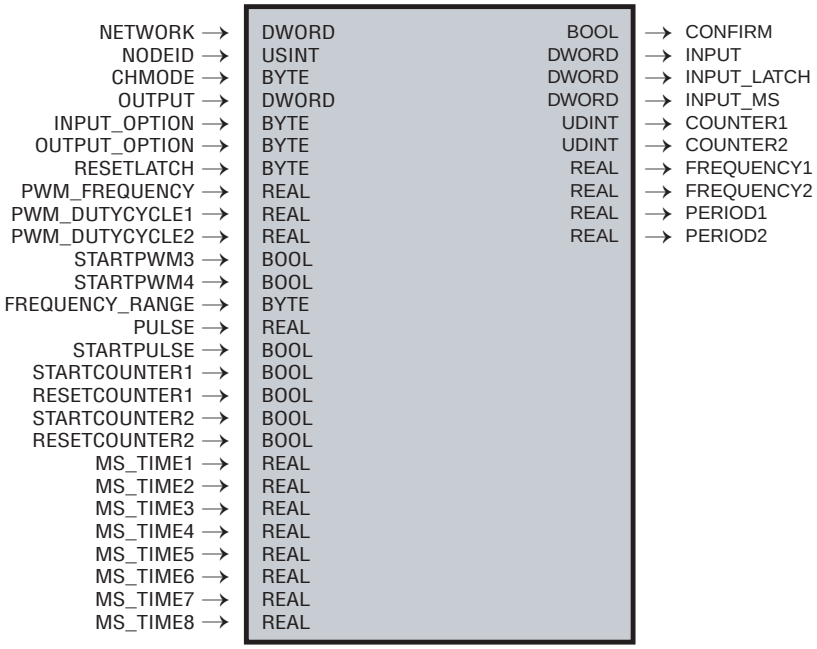
Description

This Function Block allows the user to set, for each, channel the direction (input or output), and then it sets the outputs using the provided “Output” value signal, and it returns the input values using the "Input" signal. For the ChMode input, the channel n is an input if the n-th bit is set to 0, the channel m is an output if the m-th bit is set to 1. Furthermore the FB wants as inputs also the network connection and the node identifier.

2-3-10 aDM08TS

FB Prototype

aDM08TS



Input parameters

Input	Type	Description
Network	DWORD	CAN Network Connection
NodeID	USINT	CANopen Device Identifier
ChMode	BYTE	Channel Direction (default value = 0 – all inputs)
Output	DWORD	Output State (default value = 0)
Input_Option	BYTE	Input Option (default value = 0)
Output_Option	BYTE	Output Option (default value = 0)
ResetLatch	BYTE	Reset Latched Input (default value = 0)
PWM_Frequency	REAL	PWM Frequency [Hz] (from 15mHz to 4KHz step by 1mHz -default value = 1.0)
PWM_DutyCycle1	REAL	PWM Duty Cycle Channel 1 [%] (default value = 0.0)
PWM_DutyCycle2	REAL	PWM Duty Cycle Channel 2 [%](default value = 0.0)
Frequency_Range	BYTE	Input Signal Frequency Selection (default value = 0)
StartPWM3	BOOL	Start PWM Signal Generation Channel 3 (default = FALSE)
StartPWM4	BOOL	Start PWM Signal Generation Channel 4 (default = FALSE)
Pulse	REAL	Pulse Length Channel 3 [s] (5ms... 65.0s step by 5ms - default value = 1.0)
StartPulse	BOOL	Start Pulse Command Channel 3 (default = FALSE)
StartCounter1	BOOL	Start Counter Command Channel 1 (default = FALSE)
ResetCounter1	BOOL	Reset Counter Command Channel 1 (default = FALSE)
StartCounter2	BOOL	Start Counter Command Channel 2 (default = FALSE)
ResetCounter2	BOOL	Reset Counter Command Channel 2 (default = FALSE)
MS_Time1	REAL	Input Monostable Time Channel 1 [s] (5ms... 65.0s step by 5ms - default value = 1.0)
MS_Time2	REAL	Input Monostable Time Channel 2 [s] (5ms... 65.0s step by 5ms - default value = 1.0)
MS_Time3	REAL	Input Monostable Time Channel 3 [s] (from 5ms... 65.0s step by 5ms - default value = 1.0)

Input	Type	Description
MS_Time4	REAL	Input Monostable Time Channel 4 [s] (from 5ms to 65.0s step by 5ms - default value = 1.0)
MS_Time5	REAL	Input Monostable Time Channel 5 [s] (from 5ms to 65.0s step by 5ms - default value = 1.0)
MS_Time6	REAL	Input Monostable Time Channel 6 [s] (from 5ms to 65.0s step by 5ms - default value = 1.0)
MS_Time7	REAL	Input Monostable Time Channel 7 [s] (from 5ms to 65.0s step by 5ms - default value = 1.0)
MS_Time8	REAL	Input Monostable Time Channel 8 [s] (from 5ms to 65.0s step by 5ms - default value = 1.0)

Output parameters

Output	Type	Description
Confirm	BOOL	Function Block Confirm
Input	DWORD	Input State (default value = 0)
Input_Latch	DWORD	Latched Input State (default value = 0)
Input_MS	DWORD	Monostable Input (default value = 0)
Counter1	UDINT	Counter Value Channel 1 (default value = 0)
Counter2	UDINT	Counter Value Channel 2 (default value = 0)
Frequency1	REAL	Frequency Value Channel 1 [Hz] (default value = 0.0)
Frequency2	REAL	Frequency Value Channel 2 [Hz] (default value = 0.0)
Period1	REAL	Period Value Channel 1 [s] (default value = 0.0)
Period2	REAL	Period Value Channel 2 [s] (default value = 0.0)

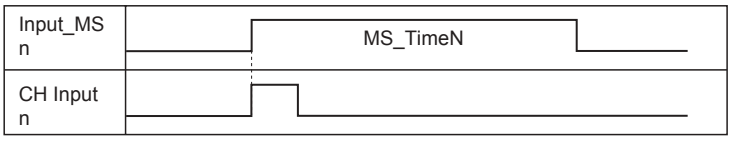
Description This function block is able to manage all the features present on the module IO-CB/DM-08TS. Each of the I/O channels of the module can be programmed as either input or output. In addition to that, two of the inputs (channel 1 and 2) can perform pulse counting, frequency and pulse width measurements, while two of the outputs can perform PWM (channel 3 and 4) and single pulse signal generation (channel 3).

Standard I/O Mode With the ChMode input the user is able to select which channel is configured as an input or output: the channel n is an input if the **n-th** bit is zero, while the channel m is an output if the **m-th** bit is 1. By the Output signal is possible to assign the output states of the module, and using the Input signal is possible to read the module's input status.

Input Latch For all the input channels of the modules is active the latch function. This value is available with the signal Input_Latch, while using the ResetLatch input it is possible for each channel to reset the latch condition (the n-th bit resets the n-th input channel).

Input Monostable For all the input channels of the modules is active the monostable function. With this function the **n-th** input is ON for a specified time starting from the rising edge of the **n-th** input channel. The signals involved are:

- Input_MS for the monostable input image
- MS_TimeX for the timing settings (where X is 1…8)



Input Special Function

Frequency Measurement

For the channels 1 and channel 2 it is possible to activate the function of frequency measurement. The module is able to provide the frequency of the digital input signals connected to the device. The outputs Frequency1 and Frequency2 contain the measured values while the user must specify the range of the input signal using the Frequency_Range parameter: 0 for the range 0.015Hz... 2KHz and 1 for the range 1Hz... 20KHz. Please note that the frequency range it is common for both the input channels.

To activate this function the user must:

- set for the channel used as input with ChMode
- set the proper value to the Input_Option parameter (see the table below)
- set the desired Frequency_Range value

Counter Input

For the channel 1 and channel 2 it is possible to activate the function of counter input. The module is able to count the rising edge of the digital input signal connected to the device. The outputs Counter1 and Counter2 contain the number of the edge detected.

To activate this function the user must:

- set for the channel used as input with ChMode
- set the proper value to the Input_Option parameter (see the table below)
- give the start command to the STARTCOUNTERx parameters

The value of the counter can be reset to 0 using the RESETCOUNTERx command.

Period Measurement

For the channels 1 and channel 2 it is possible to activate the function of period measurement. The module is able to provide the period time of the digital input signals connected to the device. The outputs Period1 and Period2 contain the measured values.

To activate this function the user must:

- set for the channel used as input with ChMode
- set the proper value to the Input_Option parameter (see the table below).

Input Option Table

Val	Channel 1	Channel 2
0	No Option Active – General purpose channel	No Option Active – General purpose channel
1	Frequency measurement	No Option Active – General purpose channel
2	Frequency measurement	Frequency measurement
3	Counter	No Option Active – General purpose channel
4	Counter	Counter
5	Period measurement	No Option Active – General purpose channel
6	Period measurement	Period measurement
7	Frequency measurement	Counter
8	Frequency measurement	Period measurement
9	Counter	Period measurement

Output Special Function

PWM Signal Generation

For the channel 3 and channel 4 it is possible to activate the function of PWM signal generation. The module is able to generate two different digital PWM signals with variable duty cyle (PWM_DutyCycleX) using the same frequency (PWM_Frequency).

To activate this function the user must:

- set for the channel used as output with ChMode
- set the proper value to the Output_Option parameter (see the table below)
- provide the value of the desired frequency with the PWM_Frequency parameter
- give the start command to the STARTPMWx parameters.

Pulse Generation

For the channel 3 it is possible to activate the function of pulse signal generation. The module is able to generate a digital pulse signal with a predefined length (PulseX).

To activate this function the user must:

- set for the channel used as output with ChMode
- set the proper value to the Output_Option parameter (see the table below)
- provide the value of the desired pulse length with the PulseX parameter
- give the start command to the STARTPULSEx parameters every time the pulse is needed

Output Option Table

Val	Channel 3	Channel 4
0	No Option Active – General purpose channel	No Option Active – General purpose channel
1	PWM	No Option Active – General purpose channel
2	PWM	PWM
3	PULSE	No Option Active – General purpose channel

2-3-11 bDM16TS

FB Prototype

bDM16TS

Input parameters

Label	Type	Description
Network	DWORD	CAN Network Connection
NodeID	USINT	CANopen Device Identifier
Output	DWORD	Output State

Output parameters

Label	Type	Description
Confirm	BOOL	Function Block Confirm
Input	DWORD	Input State

Description

This Function Block sets the outputs using the provided “Output” value signal, and it returns the input values using the "Input" signal. Furthermore the FB wants as inputs also the network connection and the node identifier.

2-3-12 bDM32TS

FB Prototype

bDM32TS

Input parameters

Label	Type	Description
Network	DWORD	CAN Network Connection
NodeID	USINT	CANopen Device Identifier
Output	DWORD	Output State

Output parameters

Label	Type	Description
Confirm	BOOL	Function Block Confirm
Input	DWORD	Input State

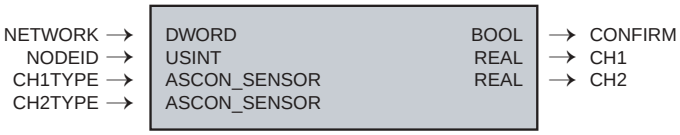
Description

This Function Block sets the outputs using the provided “Output” value signal, and it returns the input values using the "Input" signal. Furthermore the FB wants as inputs also the network connection and the node identifier.

2-3-13 bAI02UI

FB Prototype

bAI02UI



Input parameters

Label	Type	Description
Network	DWORD	CAN Network Connection
NodeID	USINT	CANopen Device Identifier
Ch1Type	ASCON_SENSOR	Input Type Channel 1 (default value = TCJ)
Ch2Type	ASCON_SENSOR	Input Type Channel 2 (default value = TCJ)

Output parameters

Label	Type	Description
Confirm	BOOL	Function Block Confirm
Ch1	REAL	Input1 in Engineering Units
Ch2	REAL	Input2 in Engineering Units

Description

This Function Block sets the chosen input type (see the table below), acquires the input data from the module and returns the value using the "Ch1" and "Ch2" signal in engineering units (°C,V, mV or mA). As inputs the FB wants the network connection, the node identifier and the type of the sensors connected.

Input tables

The configuration of the module inputs is done using the following strings. For example, if the value "V010" is assigned to the signal "Ch1Type", this means that the input channel 1 is configured as 0...10V input.

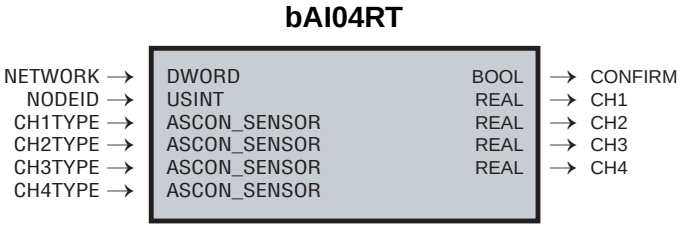
String	Sensor
TCJ (default value)	Thermocouple J
TCK	Thermocouple K
TCL	Thermocouple L
TCN	Thermocouple N
TCR	Thermocouple R
TCS	Thermocouple S
TCT	Thermocouple T
PT1002	PT100 – 2 wires
PT1003	PT100 – 3 wires
PT1004	PT100 – 4 wires
PT1000	PT1000 (2 wires)
V010	Linear Input 0... 10V
MV0150	Linear Input 0... 150mV
MA420	Linear Input 4... 20 mA
MA020	Linear Input 0... 20 mA
POT	Potentiometer

The following table gives in detail the Range for each input that may be connected to the module.

String	Range Lo	Range Hi
TCJ	-210°C (-346°F)	1200°C (2192°F)
TCK	-200°C (-328°F)	1372°C (2501.6°F)
TCL	-200°C (-328°F)	600°C (1112°F)
TCN	0°C (32°F)	1300°C (2372°F)
TCR	0°C (32°F)	1600°C (2912°F)
TCS	0°C (32°F)	1760°C (3200°F)
TCT	-200°C (-328°F)	400°C (752°F)
PT100	-200°C (-328°F)	600°C (1112°F)
PT1000	-200°C (-328°F)	600°C (1112°F)
V010	0V	10V
MV0150	0mV	150mV
MA420	4mA	20mA
MA020	0mA	20mA
POT	0%	100%

2-3-14 bAI04RT

FB Prototype



Input parameters

Label	Type	Description
Network	DWORD	CAN Network Connection
NodeID	USINT	CANopen Device Identifier
Ch1Type	ASCON_SENSOR	Input Type Channel 1 (default value = PT1003)
Ch2Type	ASCON_SENSOR	Input Type Channel 2 (default value = PT1003)
Ch3Type	ASCON_SENSOR	Input Type Channel 3 (default value = PT1003)
Ch4Type	ASCON_SENSOR	Input Type Channel 4 (default value = PT1003)

Output parameters

Label	Type	Description
Confirm	BOOL	Function Block Confirm
Ch1	REAL	Input1 in Engineering Units
Ch2	REAL	Input2 in Engineering Units
Ch3	REAL	Input3 in Engineering Units
Ch4	REAL	Input4 in Engineering Units

Description

This Function Block sets the chosen input type (see the table below), acquires the input data from the module and returns the value using the "ChX" signals in engineering units (°C or mV). As inputs the FB wants the network connection, the node identifier and the type of the sensors connected.

Input tables

The configuration of the module inputs is done using the following strings. For example, if the value “TCJ” is assigned to the signal "Ch1Type", this means that the input channel 1 is configured as a thermocouple J.

String	Sensor
TCJ	Thermocouple J
TCK	Thermocouple K
TCL	Thermocouple L
TCN	Thermocouple N
TCR	Thermocouple R
TCS	Thermocouple S
TCT	Thermocouple T
PT1002	PT100 – 2 wires
PT1003 (default value)	PT100 – 3 wires
PT1000	PT1000 (2 wires)
MVPM1000	Linear Input ±1000 mV
MVPM50	Linear Input ±50 mV
MVPM300	Linear Input ± 300 mV

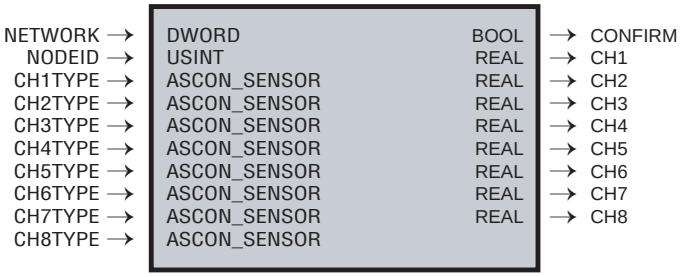
The following table gives in detail the Range for each input that may be connected to the module.

String	Range Lo	Range Hi
TCJ	-210°C (-346°F)	1200°C (2192°F)
TCK	-200°C (-328°F)	1372°C (2501.6°F)
TCL	-200°C (-328°F)	600°C (1112°F)
TCN	0°C (32°F)	1300°C (2372°F)
TCR	0°C (32°F)	1600°C (2912°F)
TCS	0°C (32°F)	1760°C (3200°F)
TCT	-200°C (-328°F)	400°C (752°F)
PT100	-200°C (-328°F)	600°C (1112°F)
PT1000	-200°C (-328°F)	600°C (1112°F)
MVPM1000	-1V	1V
MVPM50	-50mV	50mV
MVPM300	-300mV	300mV

2-3-15 bAI08TC

FB Prototype

bAI08TC



Input parameters

Label	Type	Description
Network	DWORD	CAN Network Connection
NodeID	USINT	CANopen Device Identifier
Ch1Type	ASCON_SENSOR	Input Type Channel 1 (default value = TCJ)
Ch2Type	ASCON_SENSOR	Input Type Channel 2 (default value = TCJ)
Ch3Type	ASCON_SENSOR	Input Type Channel 3 (default value = TCJ)
Ch4Type	ASCON_SENSOR	Input Type Channel 4 (default value = TCJ)
Ch5Type	ASCON_SENSOR	Input Type Channel 5 (default value = TCJ)
Ch6Type	ASCON_SENSOR	Input Type Channel 6 (default value = TCJ)
Ch7Type	ASCON_SENSOR	Input Type Channel 7 (default value = TCJ)
Ch8Type	ASCON_SENSOR	Input Type Channel 8 (default value = TCJ)

Output parameters

Label	Type	Description
Confirm	BOOL	Function Block Confirm
Ch1	REAL	Input1 in Engineering Units
Ch2	REAL	Input2 in Engineering Units
Ch3	REAL	Input3 in Engineering Units
Ch4	REAL	Input4 in Engineering Units
Ch5	REAL	Input5 in Engineering Units
Ch6	REAL	Input6 in Engineering Units
Ch7	REAL	Input7 in Engineering Units
Ch8	REAL	Input8 in Engineering Units

Description

This Function Block sets the chosen input type (see the table below), acquires the input data from the module and returns the value using the "ChX" signals in engineering units (°C or mV). As inputs the FB wants the network connection, the node identifier and the type of the sensors connected.

Input tables

The configuration of the module inputs is done using the following strings. For example, if the value “MV0100” is assigned to the signal "Ch1Type", this means that the input channel 1 is configured as 0...100mV input.

String	Sensor
TCJ (default value)	Thermocouple J
TCK	Thermocouple K
TCL	Thermocouple L
TCN	Thermocouple N
TCR	Thermocouple R
TCS	Thermocouple S
TCT	Thermocouple T
MV0100	Linear Input 0... 100 mV
MV01000	Linear Input 0... 1000 mV
MVPM100	Linear Input ±100 mV
MVPM1000	Linear Input ±1000 mV

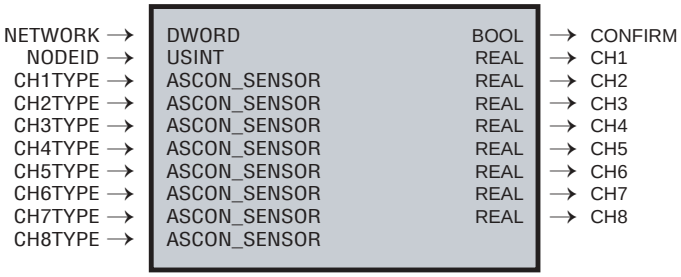
The following table gives in detail the Range for each input that may be connected to the module.

String	Range Lo	Range Hi
TCJ	-210°C (-346°F)	1200°C (2192°F)
TCK	-200°C (-328°F)	1372°C (2501.6°F)
TCL	-200°C (-328°F)	600°C (1112°F)
TCN	0°C (32°F)	1300°C (2372°F)
TCR	0°C (32°F)	1600°C (2912°F)
TCS	0°C (32°F)	1760°C (3200°F)
TCT	-200°C (-328°F)	400°C (752°F)
PT100	-200°C (-328°F)	600°C (1112°F)
PT1000	-200°C (-328°F)	600°C (1112°F)
MV0100	0V	100mV
MV01000	0V	1V
MVPM100	-100mV	100mV
MVPM1000	-1V	1V

2-3-16 bAI08HL

FB Prototype

bAI08HL



Input parameters

Label	Type	Description
Network	DWORD	CAN Network Connection
NodeID	USINT	CANopen Device Identifier
Ch1Type	ASCON_SENSOR	Input Type Channel 1 (default value = V010)
Ch2Type	ASCON_SENSOR	Input Type Channel 2 (default value = V010)
Ch3Type	ASCON_SENSOR	Input Type Channel 3 (default value = V010)
Ch4Type	ASCON_SENSOR	Input Type Channel 4 (default value = V010)
Ch5Type	ASCON_SENSOR	Input Type Channel 5 (default value = V010)
Ch6Type	ASCON_SENSOR	Input Type Channel 6 (default value = V010)
Ch7Type	ASCON_SENSOR	Input Type Channel 7 (default value = V010)
Ch8Type	ASCON_SENSOR	Input Type Channel 8 (default value = V010)

Output parameters

Label	Type	Description
Confirm	BOOL	Function Block Confirm
Ch1	REAL	Input1 in Engineering Units
Ch2	REAL	Input2 in Engineering Units
Ch3	REAL	Input3 in Engineering Units
Ch4	REAL	Input4 in Engineering Units
Ch5	REAL	Input5 in Engineering Units
Ch6	REAL	Input6 in Engineering Units
Ch7	REAL	Input7 in Engineering Units
Ch8	REAL	Input8 in Engineering Units

Description

This Function Block sets the chosen input type (see the table below), acquires the input data from the module and returns the value using the "ChX" signals in engineering units (V or mA). As inputs the FB wants the network connection, the node identifier and the type of the sensors connected.

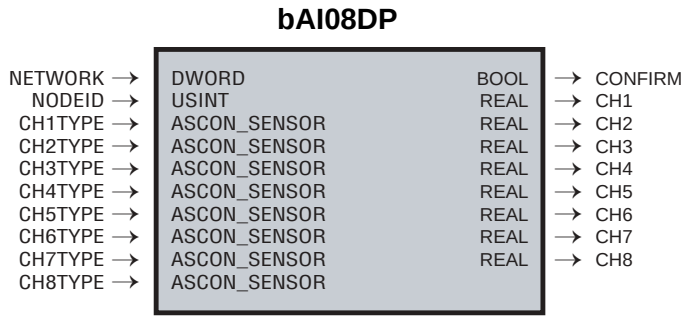
Input tables

The configuration of the module inputs is done using the following strings. For example, if the value “MV420” is assigned to the signal "Ch1Type", this means that the input channel 1 is configured as 4...20mA input.

String	Sensor
V010 (default value)	Linear Input 0... 10V
MA020	Linear Input 0... 20mA
MA420	Linear Input 4... 20mA

2-3-17 bAI08DP

FB Prototype



Input parameters

Label	Type	Description
Network	DWORD	CAN Network Connection
NodeID	USINT	CANopen Device Identifier
Ch1Type	ASCON_SENSOR	Input Type Channel 1 (default value = VPM10)
Ch2Type	ASCON_SENSOR	Input Type Channel 2 (default value = VPM10)
Ch3Type	ASCON_SENSOR	Input Type Channel 3 (default value = VPM10)
Ch4Type	ASCON_SENSOR	Input Type Channel 4 (default value = VPM10)
Ch5Type	ASCON_SENSOR	Input Type Channel 5 (default value = VPM10)
Ch6Type	ASCON_SENSOR	Input Type Channel 6 (default value = VPM10)
Ch7Type	ASCON_SENSOR	Input Type Channel 7 (default value = VPM10)
Ch8Type	ASCON_SENSOR	Input Type Channel 8 (default value = VPM10)

Output parameters

Label	Type	Description
Confirm	BOOL	Function Block Confirm
Ch1	REAL	Input1 in Engineering Units
Ch2	REAL	Input2 in Engineering Units
Ch3	REAL	Input3 in Engineering Units
Ch4	REAL	Input4 in Engineering Units
Ch5	REAL	Input5 in Engineering Units
Ch6	REAL	Input6 in Engineering Units
Ch7	REAL	Input7 in Engineering Units
Ch8	REAL	Input8 in Engineering Units

Description

This Function Block sets the chosen input type (see the table below), acquires the input data from the module and returns the value using the "ChX" signals in [V]. As inputs the FB wants the network connection, the node identifier and the type of the sensors connected.

Input tables

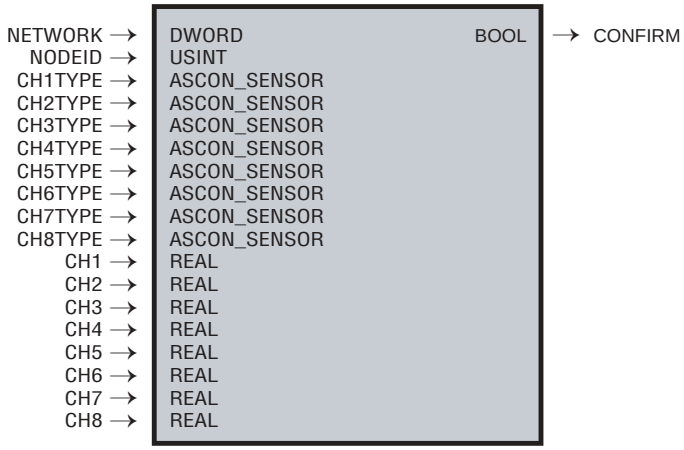
The configuration of the module inputs is done using the following strings. For example, if the value “VPM5” is assigned to the signal "Ch1Type", this means that the input channel 1 is configured as ±5V input.

String	Sensor
VPM10 (default value)	Linear Input ± 10V
VPM5	Linear Input ± 5V

2-3-18 bAO08HL

FB Prototype

bAO08HL



Input parameters

Label	Type	Description
Network	DWORD	CAN Network Connection
NodeID	USINT	CANopen Device Identifier
Ch1Type	ASCON_SENSOR	Input Type Channel 1 (default value = V010)
Ch2Type	ASCON_SENSOR	Input Type Channel 2 (default value = V010)
Ch3Type	ASCON_SENSOR	Input Type Channel 3 (default value = V010)
Ch4Type	ASCON_SENSOR	Input Type Channel 4 (default value = V010)
Ch5Type	ASCON_SENSOR	Input Type Channel 5 (default value = V010)
Ch6Type	ASCON_SENSOR	Input Type Channel 6 (default value = V010)
Ch7Type	ASCON_SENSOR	Input Type Channel 7 (default value = V010)
Ch8Type	ASCON_SENSOR	Input Type Channel 8 (default value = V010)
Ch1	REAL	Input1 in [0... 100%]
Ch2	REAL	Input2 in [0... 100%]
Ch3	REAL	Input3 in [0... 100%]
Ch4	REAL	Input4 in [0... 100%]
Ch5	REAL	Input5 in [0... 100%]
Ch6	REAL	Input6 in [0... 100%]
Ch7	REAL	Input7 in [0... 100%]
Ch8	REAL	Input8 in [0... 100%]

Output parameters

Label	Type	Description
Confirm	BOOL	Function Block Confirm

Description

This Function Block configures the chosen output type (see the table below), and sets the output value for all the channels of the module using 0... 100% range value. As inputs the FB wants the network connection, the node identifier, the type of the sensors connected, and the value of the output.

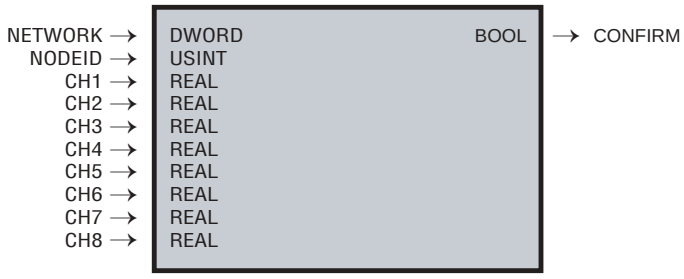
Input tables

The configuration of the module output is done using the following strings. For example, if value "MA420" is assigned to the signal "Ch1Type", this means that the output channel 1 is configured to generate current signals with the range 4... 20 mA. The numerical value expected from the Function Block is a number between 0.0 (for 4mA) and 100.0 (for 20mA) .

String	Sensor
V010 (default value)	Linear Output 0... 10V
MA020	Linear Output 0... 20mA
MA420	Linear Output 4... 20mA

2-3-19 bAO08DP

FB Prototype



Input parameters

Label	Type	Description
Network	DWORD	CAN Network Connection
NodeID	USINT	CANopen Device Identifier
Ch1	REAL	Input1 in [± 100%]
Ch2	REAL	Input2 in [± 100%]
Ch3	REAL	Input3 in [± 100%]
Ch4	REAL	Input4 in [± 100%]
Ch5	REAL	Input5 in [± 100%]
Ch6	REAL	Input6 in [± 100%]
Ch7	REAL	Input7 in [± 100%]
Ch8	REAL	Input8 in [± 100%]

Output parameters

Label	Type	Description
Confirm	BOOL	Function Block Confirm

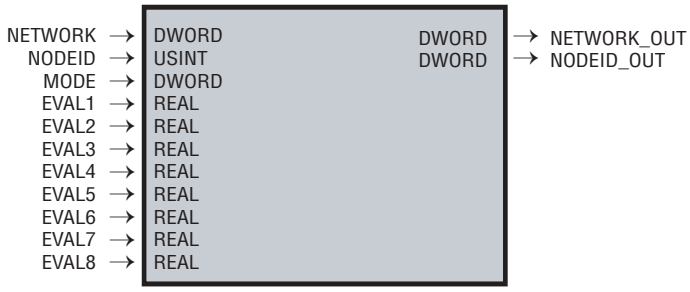
Description

This Function Block sets the output value for all the channels of the module using ±100% range value. As inputs the FB wants the network connection, the node identifier and the value of the output. Please note that this module is able to generate only signals in the range ±10V.

2-3-20 bERRORSTATEAN

FB Prototype

bERRORSTATEAN



Input parameters

Label	Type	Description
Network	DWORD	CAN Network Connection
NodeID	USINT	CANopen Device Identifier
Mode	DWORD	Error Mode Settingv (default value = 0xFF)
Eval1	REAL	Error Value Channel 1 in [0... 100%] (default value = 0.0)
Eval2	REAL	Error Value Channel 2 in [0... 100%] (default value = 0.0)
Eval3	REAL	Error Value Channel 3 in [0... 100%] (default value = 0.0)
Eval4	REAL	Error Value Channel 4 in [0... 100%] (default value = 0.0)
Eval5	REAL	Error Value Channel 5 in [0... 100%] (default value = 0.0)
Eval6	REAL	Error Value Channel 6 in [0... 100%] (default value = 0.0)
Eval7	REAL	Error Value Channel 7 in [0... 100%] (default value = 0.0)
Eval8	REAL	Error Value Channel 8 in [0... 100%] (default value = 0.0)

Output parameters

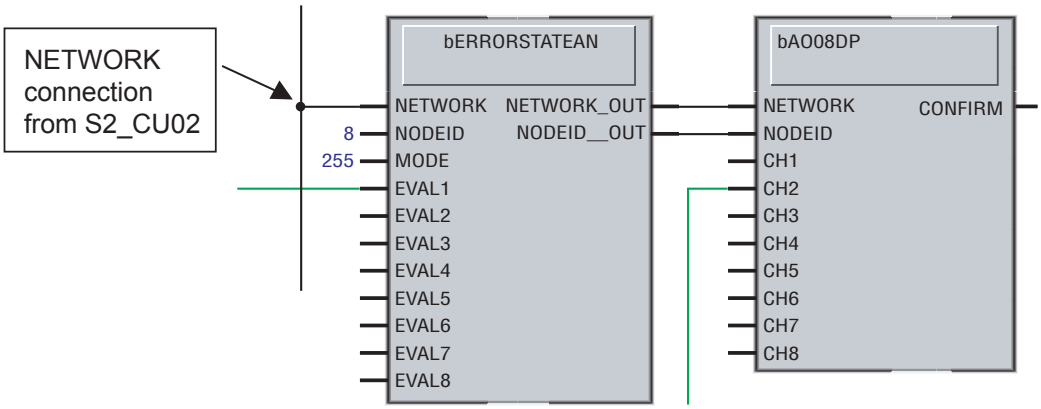
Label	Type	Description
Network_Out	DWORD	CAN Network Connection Output
NodeID_Out	USINT	CANopen Device Identifier Output

Description

This Function Block is used to configure the behaviour for the analog output modules of the sigma**due** *I/O* series of Ascon SpA in case of failure of the CAN communication. Using this function block the user is able to activate the capability of the CAN node to manage the error condition. Ascon SpA strictly recommends to use as a diagnostic system the *NodeGuarding* protocol in order to process correctly the error condition (for more details about the diagnostic protocols available on *sigmaPAC*, please refer to the *CU-02* User Manual).

When a communication error occurs, each module for each channel can set automatically the output to a predefined value. The *MODE* parameter defines the behaviour for each channel: if the **n-th** bit is **0** in case of error the output remains at the actual value; if the **n-th** bit is **1** the output assume the value passed with the input *EVALn* in case of error (where n is the channel). See the "*Mode Parameter*" section.

The following figure represents the way of use of this Function Block.



Note: The *bERRORSTATEAN* works with *bAO08DP* and *bAO08HL* function block of the *AsconBasicIOLib*.

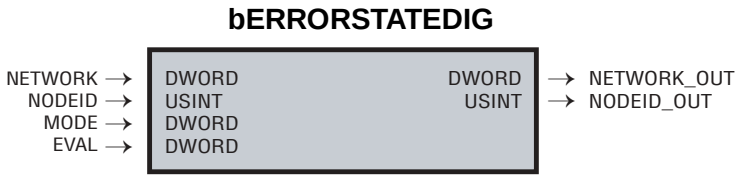
Mode parameter

MODE	Bit31-8	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
0	reserved	actual	actual	actual	actual	actual	actual	actual	actual
1		EVAL8	EVAL7	EVAL6	EVAL5	EVAL4	EVAL3	EVAL2	EVAL1

In case of error If the n-th bit of the mode parameter is 0 the output remains at the actual value; if the n-th bit is 1 the output assume the value passed with the input *EVALn* in case of error (where n is the channel).

2-3-21 bERRORSTATEDIG

FB Prototype



Input parameters

Label	Type	Description
Network	DWORD	CAN Network Connection
NodeID	USINT	CANopen Device Identifier
Mode	DWORD	Error Mode Setting (default value = 0xFF)
Eval	DWORD	Error Value (default value = 0x00)

oUTput parameters

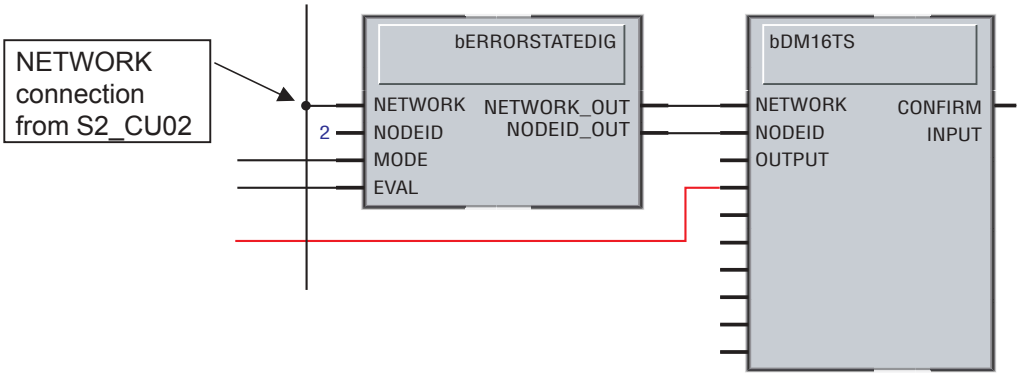
Label	Type	Description
Network_Out	DWORD	CAN Network Connection Output
NodeID_Out	USINT	CANopen Device Identifier Output

Description

This Function Block is used to configure the behaviour for the digital output modules of the sigma**due** *I/O* series of Ascon SpA in case of failure of the CAN communication. Using this function block the user is able to activate the capability of the CAN node to manage the error condition. Ascon SpA strictly recommends to use as a diagnostic system the *NodeGuarding* protocol in order to process correctly the error condition (for more details about the diagnostic protocols available on *sigmaPAC*, please refer to the *CU-02* User Manual).

When a communication error occurs, each module for each output can set automatically the output to a predefined value. The *MODE* parameter defines the behaviour for each channel: if the n-th bit is 0 in case of error the n-th output remains at the actual value; if the n-th bit is 1 the n-th output assume the value at the n-th position of EVAL input. See the "*Mode Parameter*" section.

The following figure represents the way of use of this Function Block.



Note: The *bERRORSTATEAN* works with *aDM08TS, bDM08TS, bDM16TS, bDM32TS, bDO04RL, bDO08RL*, *bDO04TX, bDO16TS, bDO16TP* and *bDO32TS* function block of the *AsconBasicIOLib*.

Mode parameter

MODE	Bit31-8	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
0	actual	actual	actual	actual	actual	actual	actual	actual	actual
1	EVAL 31-8 bit	EVAL bit7	EVAL bit6	EVAL bit5	EVAL bit4	EVAL bit3	EVAL bit2	EVAL bit1	EVAL bit0

In case of error If the n-th bit of the mode parameter is 0 the output remains at the actual value; if the n-th bit is 1 the n-th output assume the value at the n-th position of EVAL input.

Chapter 3
AsconControlLib

3-1 Purpose

The purpose of this document is to provide a complete description of the *AsconControlLib*.

3-2 Description

The *AsconControlLib* is a function block library dedicated to the process control. It uses the basic functionalities dedicated to the P.I.D. implementation present in the firmware of the control unit (CU-02 and MP-01) device in order to provide solution ready to use. In fact in the library there is the implementation of a complete standard regulator in both version: single action and double action for heat and cool application.

Please note that are present also different function blocks dedicated to the tuning algorithms. The table here reported gives the complete list of the function blocks of the library.

Function Block name	Description
S2_CONTROLLER	Single Action Controller
S2_HC_CONTROLLER	Heat and Cool Controller
S2_TNATFREQ	Tuning with Natural Frequency Algorithm for Single Action Loops
S2_TSTEPRESP	Tuning with Step Response Algorithm for Single Action Loops
S2_TFUZZY	Tuning with Fuzzy Logic for Single Action Loops
S2_HC_TNATFREQ	Tuning with Natural Frequency Algorithm for Heat and Cool Loops
S2_HC_TSTEPRESP	Tuning with Step Response Algorithm for Heat and Cool Loops
S2_HC_TFUZZY	Tuning with Fuzzy Logic for Heat and Cool Loops
S2_EZ_TUNE	Tuning with Modified Step Response Algorithm for Single Action Loops
S2_HC_EZ_TUNE	Tuning with Modified Step Response Algorithm for Heat and Cool Loops
S2_MV	AutoMan station for output manual value direct access for single action loop
S2_HCMV	AutoMan station for output manual value direct access for double action loop
S2_SPLITMV	AutoMan station for output manual value direct access for double action loop with SplitRange
S2_FILTER	First Order Filter

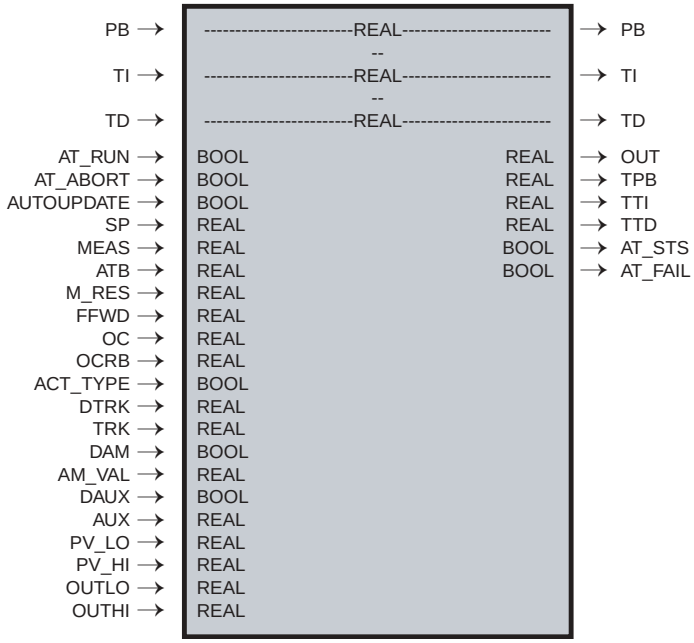
For more details please refer to the specific documentation of each function block.

3-3 Function Block Decription

3-3-1 S2_CONTROLLER

FB Prototype

S2_CONTROLLER



Input parameters

Label	Type	Description	Range
PB	REAL	Proportional Band [%]	0.0... 1000.0
TI	REAL	Integral Time [s]	1.0... 6000.0 0.0 disables the action
TD	REAL	Derivative Time [s]	0.5... 600.0 0.0 disables the action
AT_RUN	BOOL	Tuning RUN command. The command is received on the input rising edge	
A_ABORT	BOOL	Tuning ABORT command. The command is received on the input rising edge	
AUTOUPDATE	BOOL	Tuning Auto Update Parameters behavior	
SP	REAL	Setpoint value [e.u.]	PVLO... PVHI
MEAS	REAL	Measure value [e.u.]	PVLO... PVHI
ATB	REAL	Auto – Tune with Bias [% of range PV_LO and PV_HI]	0… 100.0
M_RES	REAL	Manual Reset [%](range 0…100.0)	0…100.0
FFWD	REAL	Value for Feed Forward Action [%]	-100.0…100.0
OC	REAL	Overshoot Control [num]	0.01…1.00
OCRB	REAL	Overshoot Control Relative Band [num]	0.2… 5.0
ACT_TYPE	BOOL	PID Action Type (direct/reverse)	FALSE = Rev. TRUE = Dir.
DTRK	BOOL	Output Digital Tracking command	
TRK	REAL	Output Tracking value [%]	OUTLO… OUTHI

Label	Type	Description	Range
DAM	BOOL	Output Digital Auto/Manual Command	
AM_VAL	REAL	Output Auto/Manual value [%]	0…100.0
DAUX	BOOL	Output Digital Auxiliary command	
AUX	REAL	Output Auxiliary Signal [%]	OUTLO… OUTHI
PV_LO	REAL	PV input Low range value [e.u.]	
PV_HI	REAL	PV input High range value [e.u.]	
OUTLO	REAL	Control Output Low limit value [%]	0…100.0
OUTHI	REAL	Control Output High limit value [%]	0…100.0

Output parameters

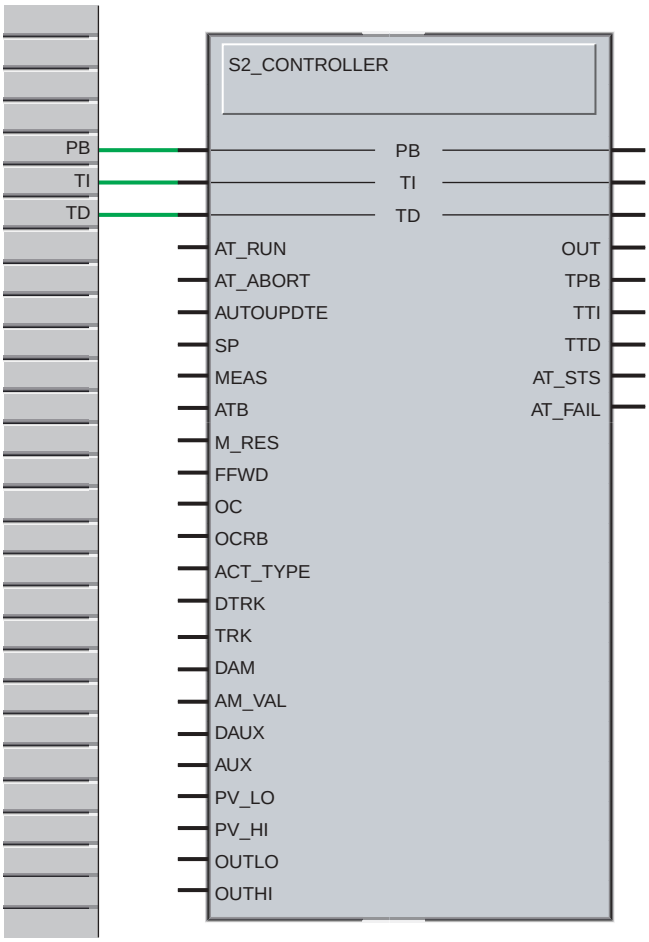
Label	Type	Description	Range
OUT	REAL	Control Output value [%]	OUTLO… OUTHI
TPB	REAL	Tuning Result - Calculated Proportional Band [%]	
TTI	REAL	Tuning Result - Calculated Integral Time [s]	
TTD	REAL	Tuning Result - Calculated Derivative Time [s]	
AT_STS	BOOL	This output is true during tuning activities	
AT_FAIL	BOOL	This output is set to false at the beginning of tuning process and true in case process fails	

Description

This function block implements the same functionalities of a complete single action regulator. It needs as input the process variables (*SP* and *MEAS*) in engineering format (the range is passed as input from the variables *PV_LO* and *PV_HI*), the Ascon PID parameters (*PB*, *TI*, *TD*, *M_RES*, *FFWD*, *OC*, *OCRB*, ACT_TYPE), the tuning commands (AT_RUN, AT_ABORT and AUTOUPDATE), and the signals for the manual access of the block (*TRK*, *AUX*, *DTRK*, *DAUX*, *DAM* and *AM_VAL*). As output provides the control value OUT and all the parameters calculated from the tuning algorithm (*TPB*, *TTI*, *TTD*, *AT_STS* and *AT_FAIL*).

The parameters PB, TI, TD, here indicated as inputs, in the FB code are declared as **IN_OUT** parameters: *THIS MEANS THAT YOU ALWAYS MUST PROVIDE AN INPUT CONNECTION WITH A PROCESS VARIABLE FOR EACH OF THEM*, and please note that when you create an instance, you will have the same parameters as input and as output.

The following picture shows an example using CFC languages..



Here is reported a detailed description of the parameters:

Parameter	Description
PB	Proportional band coefficient that multiplies the error
TI	Integral time value, that specifies the time required by the integral term to generate an output equivalent to the proportional term
TD	Is the time required by the proportional term P to repeat the output provided by the derivative term D
AT_RUN	Start of the tuning operation. The command is received on the rising edge of the input
AT_ABORT	Abort a running tuning process. The command is received on the rising edge of the input
AUTOUPDATE	If TRUE the tuning operation result will be updated automatically
SP	Setpoint Value in engineering units
MEAS	Measure Value in engineering units
ATB	Threshold value [0100%]
M_RES	Is the control output value when PV = SP in a PD only algorithm (lack of the integral term)
FFWD	Value for Feed Forward Action
OC	Specifies the span within the overshoot control operates. Low values (1.00... 0.01) emphasize the overshoot strength during set point variations. The overshoot control does not affect the PID action effectiveness. A value of 1.00 means OC disabled.

Parameter	Description
OCRB	Defines the zone across the setpoint where the PID algorithm is not affected by the overshoot control
ACT_TYPE	Selection of the control output direction: direct or reverse.
DTRK	Tracking Operating mode Command
TRK	Tracking signal value
DAM	Automatic/Manual Operating Mode Command
AM_VAL	Manual Value [0…100.0]
DAUX	Auxiliary Operating mode Command
AUX	Auxiliary signal value
PV_LO	Lo Span of the Process Value
PV_HI	Hi Span of the Process Value
OUTLO	Lo value for the PID output. This limit is active always except for the manual movements of the output (i.e. DAM active).
OUTHI	Hi value for the PID output. This limit is active always except for the manual movements of the output (i.e. DAM active).
OUT	Controller Output Value.
TPB	Pb calculated from the tuning algorithm
TTI	Ti calculated from the tuning algorithm
TTD	Td calculated from the tuning algorithm
AT_STS	This output is true during tuning activities
AT_FAIL	This output is set to false at the tuning process beginning and is set to true if tuning process fails

Parameters Default

Here are reported the default values for the input/output parameters:

Input	Default Value
PB	These parameters must be connected with an external variable. For this reason, the default value is the one specified as initial. Please note that, in any case, internally the PID actions are limited by these ranges: **PB**: 0.5... 1000.0 **TI**: 1.0... 6000.0 - the value 0.0 is allowed to exclude the integral action **TD**: 0.5... 600.0 - the value 0.0 is allowed to exclude the derivative action
TI	
TD	
SP	0.0
MEAS	0.0
M_RES	50.0
FFWD	0.0
OC	1.0
OCRB	0.5
ACT_TYPE	FALSE
AT_RUN	FALSE
AT_ABORT	FALSE
AUTOUPDATE	FALSE
DTRK	FALSE
TRK	0.0
DAM	FALSE

Input	Default Value
AM_VAL	0.0
DAUX	FALSE
AUX	0.0
PVLO	0.0
PVHI	100.0
OUTLO	0.0
OUTHI	100.0

Output	Default Value
OUT	0.0
TPB	0.0
TTI	0.0
TTD	0.0
AT_STS	FALSE
AT_FAIL	FALSE

Manual Access Parameters

The FB provides all the functionalities the operator needs to interact with the loop. The loop is controlled by one PID, connected internally to an AutoManual block. By a set of digital inputs the function block provides auto/manual/track/aux operating mode as listed above, ordered according to their priority level.

Trk *DTRK*
AutoMan *DAM*
Aux *DAUX*

Depending on these digital inputs, the functionality of the module changes as follows:

- If none of these digital signals are active, the module does not alter the control output signal (from the PID module).
- If *DTRK* (Tracking) is active, the output *OUT* assumes the value of the analog input *TRK*.
- If the *DAM* input is active (and *DTRK* is inactive) the output *OUT* assumes the value of the analog input AM_VAL.
- If *DAUX* (Aux) is active (and both *DAM* and *DTRK* are inactive), the output *OUT* assumes the value of the analog input *AUX*.

Note: Please note that the *OUT* limit inputs (*OUTLO*, *OUTHI*) do not affect the output values coming from the Manual Access (i.e. *DAM input TRUE*).

Inputs M_RES and FFWD are used to set a fixed offset to add to the PID calculation. The system does not add boot terms but select FFWD if it is different from zero, elsewhere the system add M_RES.

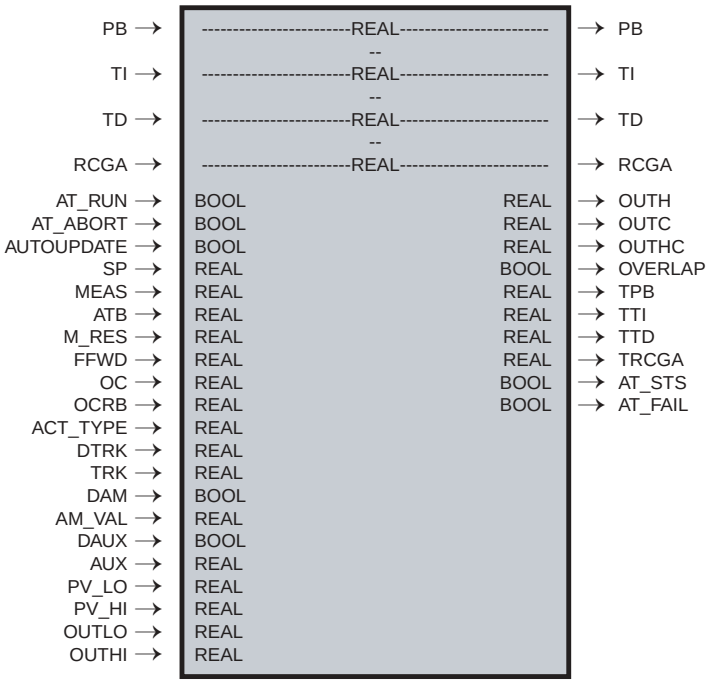
Automatic Tune

This function block is able to perform tuning calculation. The FB select the EZ tune method if ATB is greater than zero, elsewhere it use the FUZZY method. To understand how these methods work refers to the documentation of the function blocks S2_EZ_TUNE and S2_TFUZZY.

3-3-2 S2_HC_CONTROLLER

FB Prototype

S2_HC_CONTROLLER



Input parameters

Label	Type	Description	Range
PB	REAL	Proportional Band [%]	0.0... 1000.0
TI	REAL	Integral Time [s]	1.0... 6000.0 0.0 disables the action
TD	REAL	Derivative Time [s]	0.5... 600.0 0.0 disables the action
RCGA	REAL	Relative Cool GAin [num]	0.1... 10.0
AT_RUN	BOOL	Tuning RUN command. The command is received on the input rising edge	
A_ABORT	BOOL	Tuning ABORT command. The command is received on the input rising edge	
AUTOUPDATE	BOOL	Tuning Auto Update Parameters behavior	
SP	REAL	Setpoint value [e.u.]	PVLO... PVHI
MEAS	REAL	Measure value [e.u.]	PVLO... PVHI
ATB	REAL	Auto – Tune with Bias [% of range PV_LO and PV_HI]	0…100.0
M_RES	REAL	Manual Reset [%]	-100.0…100.0
FFWD	REAL	Value for Feed Forward Action [%]	-100.0…100.0
OC	REAL	Overshoot Control [num]	0.01…1.00
OCRB	REAL	Overshoot Control Relative Band [num]	0.2… 5.0
DBND	REAL	Heat and Cool Action Dead Band [%]	-10.0… 10.0
DTRK	BOOL	Output Digital Tracking command	
TRK	REAL	Output Tracking value [%]	-OUTC_HI … OUTH_HI
DAM	BOOL	Output Digital Auto/Manual Command	

Label	Type	Description	Range
AM_VAL	REAL	Output Auto/Manual value [%]	-100.0…100.0
DAUX	BOOL	Output Digital Auxiliary command	
AUX	REAL	Output Auxiliary Signal [%]	-OUTC_HI … OUTH_HI
PV_LO	REAL	PV input Low range value [e.u.]	
PV_HI	REAL	PV input High range value [e.u.]	
OUTH_HI	REAL	Heat Control Output High limit value [%]	0… 100.0
OUTC_HI	REAL	Cool Control Output High limit value [%]	0… 100.0

Output parameter

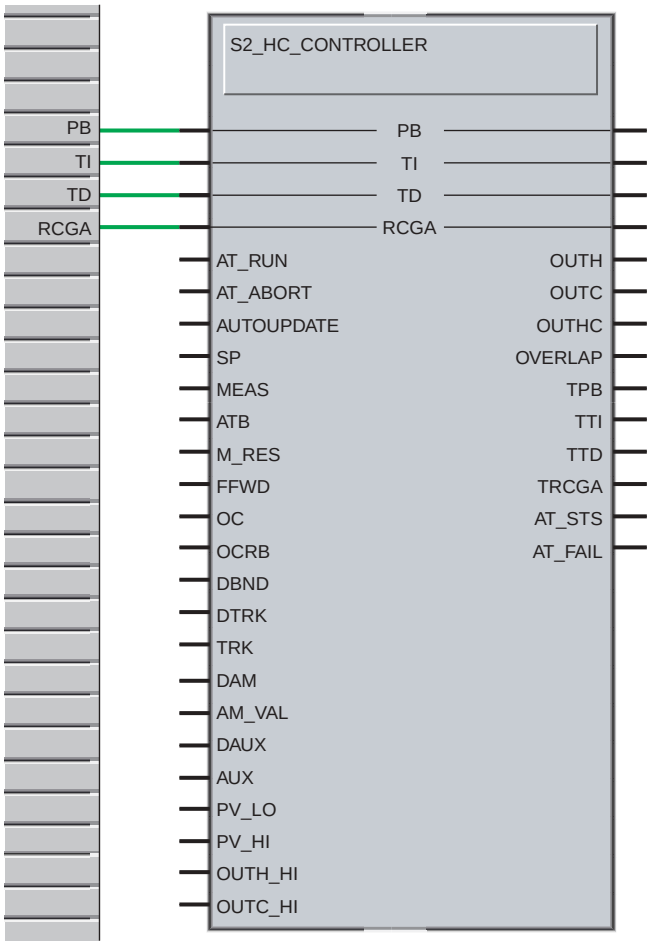
Label	Type	Description	Range
OUTH	REAL	Heat Control Output value [%]	0... OUTH_HI
OUTC	REAL	Cool Control Output value [%]	0... OUTC_HI
OUTHC	REAL	Full range Output value [%]	OUTC_HI... +OUTH_HI
OVERLAP	BOOL	Active if both OUTH and OUTC are active	
TPB	REAL	Tuning Result - Calculated Proportional Band [%]	
TTI	REAL	Tuning Result - Calculated Integral Time [s]	
TTD	REAL	Tuning Result - Calculated Derivative Time [s]	
TRCGA	REAL	Tuning Result - Calculated Relative Cool Gain Parameter [num]	
AT_STS	BOOL	This output is true during tuning activities	
AT_FAIL	BOOL	This output is set to false at the beginning of tuning processand true in case process fails	

Description

This function block provides the functionalities of a complete double action regulator. It needs as input the process variables (*SP* and *MEAS*) in engineering format (the range is passed as input from the variables P*V_LO* and *PV_HI*), the Ascon PID parameters (*PB, TI, TD, RCG, FFWD, M_RES, OC, OCRB*), the tuning commands (*TRUN* and *AUTOUPDATE*), and the signals for the manual access of the block (*TRK, AUX,DTRK,DAUX,DAM and AM_VAL*). As output it provides the control values for the two actions (*OUTH* and *OUTC*, limited using the values *OUTH_HI* and *OUTC_HI*), an output (*OUTHC* with range [-OUTC_HI…*+OUTH_HI*]) devoted to the visualization, and all the parameters calculated from the tuning parameters (*TPB, TTI, TTD, TRCGA, AT_STS, AT_FAIL*).

The parameters *PB*, *TI*, *TD*, *RCGA*, here indicated as inputs, in the FB code are declared as ***IN_OUT*** parameters: ***THIS MEANS THAT YOU ALWAYS MUST PROVIDE AN INPUT CONNECTION WITH A PROCESS VARIABLE FOR EACH OF THEM,*** and please note that when you create an instance, you will have the same parameters as input and as output.

The following picture shows an example using CFC languages.



Here is reported a detailed description of the parameters:

Input	Description
PB	Proportional band coefficient that multiplies the error
TI	Integral time value, that specifies the time required by the integral term to generate an output equivalent to the proportional term
TD	Is the time required by the proportional term P to repeat the output provided by the derivative term D
RCGA	Gain Factor for the cool action
AT_RUN	Start of the tuning operation. The command is received on the rising edge of the input
AT_ABORT	Abort a running tuning process. The command is received on the rising edge of the input
AUTOUPDATE	If TRUE the result of the tuning operation will be updated automatically
SP	Setpoint Value in engineering units
MEAS	Measure Value in engineering units
M_RES	Is the control output value when PV = SP in a PD only algorithm (lack of the integral term)
FFWD	Value for Feed Forward Action
OC	Specifies the span within the overshoot control operates. Low values (1.00... 0.01) emphasize the overshoot strength during set point variations. The overshoot control does not affect the PID action effectiveness. A value of 1.00 means OC disabled.

Input	Description
OCRB	Defines the zone across the setpoint where the PID algorithm is not affected by the overshoot control
DBND	Is the zone where it is possible to separate or overlap the heat and cool actions
DTRK	Tracking Operating mode Command
TRK	Tracking signal value
DAM	Automatic/Manual Operating Mode Command
AM_VAL	Manual Value desired as Output
DAUX	Auxiliary Operating mode Command
AUX	Auxiliary signal value
PV_LO	Lo Span of the Process Value
PV_HI	Hi Span of the Process Value
OUTH_HI	Hi value for the heat output. This limit is active always except for the manual movements of the output (i.e. DAM active).
OUTC_HI	Hi value for the cool output. This limit is active always except for the manual movements of the output (i.e. DAM active).

Output	Description
OUTH	Output for the Heat Action
OUTC	Output for the Cool Action
OUTHC	Global output value for the visualization
OVERLAP	Active if both OUTH and OUTC are active
TPB	Pb calculated from the tuning algorithm
TTI	Ti calculated from the tuning algorithm
TTD	Td calculated from the tuning algorithm
TRCGA	RCGA calculated from the tuning algorithm
AT_STS	This output is true during tuning activities
AT_FAIL	This output is set to false at the tuning process beginning and is set to true if tuning process fails

Parameters Default

Here are reported the default values for the input/output parameters.

Input Parameters Default

Input	Default Value
PB	These parameters need the connection with an external variable. The default value will be the default value for these external VAR. Please note that in any case internally the PID actions will be limited using these ranges: **PB**: 0.5... 1000.0 **TI**: 1.0... 6000.0 - the value 0.0 is allowed to exclude the integral action **TD**: 0.5... 600.0 - the value 0.0 is allowed to exclude the derivative action **RCGA**: 0.1... 10.0
TI	
TD	
RCGA	
SP	0.0
MEAS	0.0
M_RES	50.0
FFWD	-100
OC	1.0

Input	Default Value
OCRB	0.5
AT_RUN	FALSE
AT_ABORT	FALSE
DBND	0.0
AUTOUPDATE	FALSE
DTRK	FALSE
TRK	0.0
DAM	FALSE
AM_VAL	0.0
DAUX	FALSE
AUX	0.0
PV_LO	0.0
PV_HI	100.0
OUTH_HI	100.0
OUTC_HI	100.0

Output Parameters Default

Output	Default Value
OUTH	0.0
OUTC	0.0
OUTHC	0.0
OVERLAP	FALSE
TPB	0.0
TTI	0.0
TTD	0.0
TRCGA	0.0
AT_STS	FALSE
AT_FAIL	FALSE

Manual Access Parameters

The FB provides all the functionalities the operator needs to interact with the loop. The loop is controlled by one PID, connected internally to a Auto Manual block (the Split Range version) which provides auto/manual/track/aux operating mode. These modes, that are unique for both the Cool and the Heat output, are defined by a set of digital inputs, as listed above, ordered according to their priority level.

Trk DTRK

AutoMan DAM

Aux DAUX

Depending on these digital inputs, the functionality of the module changes as follows:

- If none of these digital signals are active, the FB does not alter the control output signals. The *DEADBAND* parameter defines how the Cool and the Heat channel interact each other. When it is positive, this parameter defines the dead band width, centered on the value 0 of the output. It means that for each output value included in the dead band both the *OUTH* and the *OUTC* output are 0. When it is negative, the opposite occurs; for each value included in the deadband (now named cross band), both the *OUTH* and the *OUTC* are different from 0.

- If *DTRK* (Track) is active, the outputs are set according to the value of the analog input *TRK*. This signal, with range [-100.0... 100.0], drives both *OUTH* and *OUTC* through a phase splitter. When its value is in the range [-100.0 … 0] it means that a Cool action must be taken; therefore, *OUTH* is set to 0, while *OUTC* is set to the value of *TRK*, with the sign inverted. Inversely, when it is in the range [0... 100.0] *OUTC* is set to 0, while *OUTH* is set to *TRK*. Please note that the tracking value is not affected by the *DEADBAND* parameter or by the output limits.
- If the *DAM* input is active (and *DTRK* is inactive) the output *OUT* assumes the value of the analog input *AM_VAL*.
- If *DAUX* (Aux) is active (and both *DAUX* and *DTRK* are inactive), the outputs are set according to the value of the analog input *AUX*. This signal, with range [-100.0... 100.0], drives both *OUTH* and *OUTC* through a phase splitter. When its value is in the range [-100.0... 0] it means that a Cool action must be taken; therefore, *OUTH* is set to 0, while *OUTC* is set to the value of *AUX*, with the sign inverted. Inversely, when it is in the range [0... 100.0] *OUTC* is set to 0, while *OUTH* is set to *AUX*. These values are valid if the *DEADBAND* of the phase splitter is set to 0, otherwise they slightly change. Please note that the *AUX* value is not affected by the output limits.

Inputs *M_RES* and *FFWD* are used to set a fixed offset to add to the PID calculation. The system does not add boot terms but select *FFWD* if it is different from -100, elsewhere the system add *M_RES*.

Automatic Tune

This function block is able to perform tuning calculation. The FB select the EZ tune method if ATB is greater than zero, elsewhere it use the FUZZY method. To understand how these methods work refers to the documentation of the function blocks *S2_EZ_HC_TUNE* and *S2_HC_TFUZZY*.

3-3-3 S2_TNATFREQ

FB Prototype

S2_TNATFREQ

Input parameters

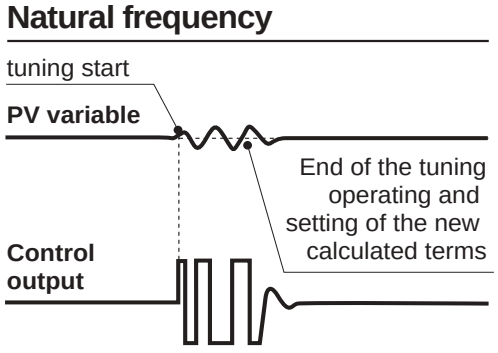
Input	Type	Description
ENABLE	BOOL	Function block enable
SP	REAL	Setpoint value [0…100.0]
MEAS	REAL	Measure value [0…100.0]
ACT_TYPE	BOOL	PID action type
OUTLO	REAL	Lo value for OUT output [0…100.0]
OUTHI	REAL	Hi value for OUT output [0…100.0]

Output parameters

Output	Type	Description
TOUT	REAL	Function block output [OUTLO…OUTHI]
TPB	REAL	Calculated proportional band [%]
TTI	REAL	Calculated integral time [s]
TTD	REAL	Calculated derivative time [s]
TDBMP	BOOL	Bumpless command for the PID
TRUNNING	BOOL	Running flag
TVALID	BOOL	Successful tuning calculation flag

Description

This function block is able to perform the tuning calculation using the Natural frequency algorithm. It works together a P.I.D. block with which shares the inputs and the outputs from and for the process. Please note that these process parameters are using the range [0… 100.0]. The method has the advantage of a better accuracy in term of calculation with a reasonable speed calculation.



3-3-4 S2_TSTEPRESP

FB Prototype

S2_TSTEPRESP

Input parameters

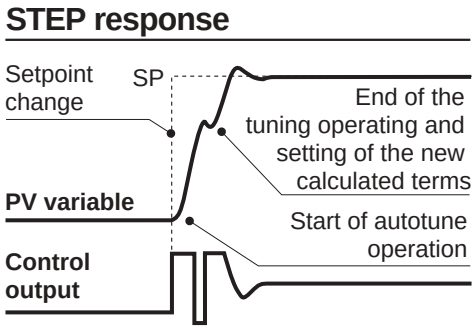
Input	Type	Description
ENABLE	BOOL	Function block enable
SP	REAL	Setpoint value [0…100.0]
MEAS	REAL	Measure value [0…100.0]
ACT_TYPE	BOOL	P.I.D. action type
OUTLO	REAL	Lo value for OUT output [0…100.0]
OUTHI	REAL	Hi value for OUT output [0…100.0]

Output parameters

Output	Type	Description
TOUT	REAL	Function block output [OUTLO…OUTHI]
TPB	REAL	Calculated proportional band [%]
TTI	REAL	Calculated integral time [s]
TTD	REAL	Calculated derivative time [s]
TDBMP	BOOL	Bumpless command for the P.I.D.
TRUNNING	BOOL	Running flag
TVALID	BOOL	Successful tuning calculation flag

Description

This function block is able to perform the tuning calculation using the step response algorithm. It works together a P.I.D. block with which shares the inputs and the outputs from and for the process. Please note that these process parameters are using the range [0… 100.0]. The method has the big advantage of fast calculation, with reasonable accuracy in term of calculation.



3-3-5 S2_TFUZZY

FB Prototype

S2_TFUZZY

Input parameters

Input	Type	Description
ENABLE	BOOL	Function block enable
SP	REAL	Setpoint value [0…100.0]
MEAS	REAL	Measure value [0…100.0]
ACT_TYPE	BOOL	P.I.D. action type
OUTLO	REAL	Lo value for OUT output [0…100.0]
OUTHI	REAL	Hi value for OUT output [0…100.0]

Output parameters

Output	Type	Description
TOUT	REAL	Function block output [OUTLO…OUTHI]
TPB	REAL	Calculated proportional band [%]
TTI	REAL	Calculated integral time [s]
TTD	REAL	Calculated derivative time [s]
TDBMP	BOOL	Bumpless command for the P.I.D.
TRUNNING	BOOL	Running flag
TVALID	BOOL	Successful tuning calculation flag

Description

This function block is able to perform a tuning calculation selecting the best method according to the process conditions. The possible algorithms are the "*Step Response*" or the "*Natural Frequency*" (please refer to the specific documentation), and the FB selects the first if the PV is far from the setpoint more then 5% (at the moment of the run command), or the second in the other cases. It works together a P.I.D. block with which shares the inputs and the outputs from and for the process.

3-3-6 S2_HC_TNATFREQ

FB Prototype

S2_HC_TNATFREQ

Input parameters

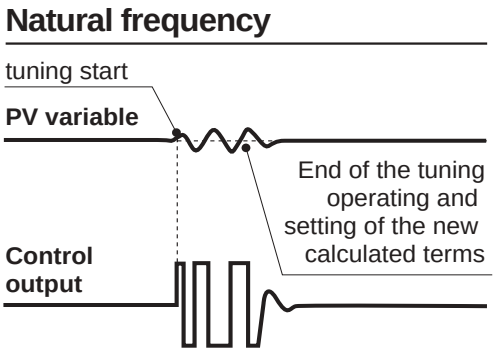
Input	Type	Description
ENABLE	BOOL	Function block enable
SP	REAL	Setpoint value [-100.0…100.0]
MEAS	REAL	Measure value [-100.0…100.0]
OUTH_HI	REAL	Limit value for heat action output [0... 100.0]
OUTC_HI	REAL	Limit value for cool action output [0... 100.0]

Output parameters

Output	Type	Description
TOUT	REAL	Function block output [OUTCHI... OUTH_HI]
TPB	REAL	Calculated proportional band [%]
TTI	REAL	Calculated integral time [s]
TTD	REAL	Calculated derivative time [s]
TRCGA	REAL	Calculated relative cool gain
TBMP	BOOL	Bumpless command for the P.I.D.
TRUNNING	BOOL	Running flag
TVALID	BOOL	Successful tuning calculation flag

Description

This function block is able to perform the tuning calculation using the Natural frequency algorithm *for a double action process ("heat and cool")*. It works together a P.I.D. block with which shares the inputs and the outputs from and for the process and an automatic/manual station with *Split Range capability*. Please note that these process parameters are using the range [0... 100.0]. The method has the advantage of a better accuracy in term of calculation with a reasonable speed calculation.



3-3-7 S2_HC_TSTEPRESP

FB Prototype

S2_HC_TSTEPRESP

Input parameters

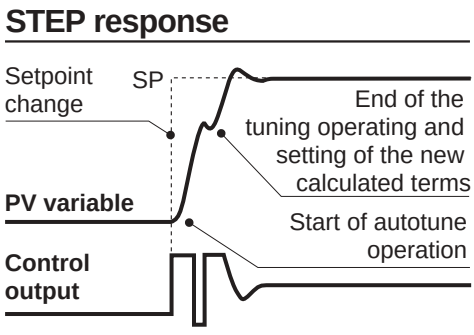
Input	Type	Description
ENABLE	BOOL	Function block enable
SP	REAL	Setpoint value [-100.0…+100.0]
MEAS	REAL	Measure value [-100.0…+100.0]
OUTH_HI	REAL	Limit value for heat action output [0...100.0]
OUTC_HI	REAL	Limit value for cool action output [0...100.0]

Output parameters

Output	Type	Description
TOUT	REAL	Function block output [OUTCHI...OUTH_HI]
TPB	REAL	Calculated proportional band
TTI	REAL	Calculated integral time [s]
TTD	REAL	Calculated derivative time [s]
TRCGA	REAL	Calculated relative cool gain
TBMP	BOOL	Bumpless command for the P.I.D.
TRUNNING	BOOL	Running flag
TVALID	BOOL	Successful tuning calculation flag

Description

This function block is able to perform the tuning calculation using the step response algorithm *for a double action process ("heat and cool")*. It works together a P.I.D. block with which shares the inputs and the outputs from and for the process and an automatic/manual station with *Split Range capability*. Please note that these process parameters are using the range [0...100.0]. The method has the advantage of a better accuracy in term of calculation with a reasonable speed calculation.



3-3-8 S2_HC_TFUZZY

FB Prototype

Input parameters

Input	Type	Description
ENABLE	BOOL	Function block enable
SP	REAL	Setpoint value [0…100.0]
MEAS	REAL	Measure value [0…100.0]
OUTH_HI	REAL	Limit value for heat action output [0...100.0]
OUTC_HI	REAL	Limit value for cool action output [0...100.0]

Output parameters

Output	Type	Description
TOUT	REAL	Function block output [OUTC_HI... OUTH_HI]
TPB	REAL	Calculated proportional band [%]
TTI	REAL	Calculated integral time [s]
TTD	REAL	Calculated derivative time [s]
TRCGA	REAL	Calculated relative cool gain
TBMP	BOOL	Bumpless command for the P.I.D.
TRUNNING	BOOL	Running flag
TVALID	BOOL	Successful tuning calculation flag

Description

This function block is able to perform a tuning calculation selecting the best method according to the process conditions *for a double action process ("heat and cool")*. The possible algorithms are the "*Step Response*" or the "*Natural Frequency*" (please refer to the specific documentation), and the FB selects the first if the PV is far from the setpoint more then 5% (at the moment of the run command), or the second in the other cases. It works together a PID block with which shares the inputs and the outputs from and for the process and an automatic/manual station with *Split Range capability*. Please note that these process parameters are using the range [0...100.0].

3-3-9 S2_EZ_TUNE

FB Prototype

S2_EZ_TUNE

Input parameters

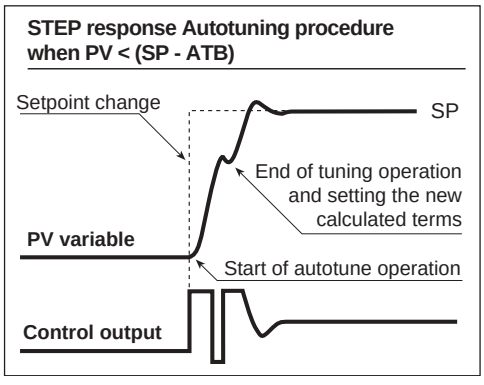
Input	Type	Description
ENABLE	BOOL	Function block enable
SP	REAL	Setpoint value [0…100.0]
MEAS	REAL	Measure value [0…100.0]
ACT_TYPE	BOOL	P.I.D. action type
ATB	REAL	Threshold value [0....100.0]
OUTLO	REAL	Lo value for OUT output [0…100.0]
OUTHI	REAL	Hi value for OUT output [0…100.0]

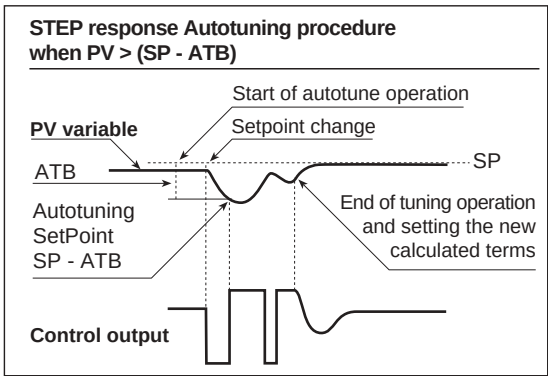
Output parameters

Output	Type	Description
TOUT	REAL	Function block output [OUTLO…OUTHI]
TPB	REAL	Calculated proportional band [%]
TTI	REAL	Calculated integral time [s]
TTD	REAL	Calculated derivative time [s]
TBMP	BOOL	Bumpless command for the P.I.D.
TRUNNING	BOOL	Running flag
TVALID	BOOL	Successful tuning calculation flag

Description

This function block is able to perform a tuning calculation using the "*Step Response*" method with a *ATB* parameter. The *ATB* value affects the quality of the results of the method: higher is its value, better are the results, but heavier is the perturbance caused to the control process; the solution is to choose a correct compromise. At the moment of the run command of the tune process, if *MEAS* is close to *SP*, the controller forces the output to the lower limit value in order to increase the error (SP - MEAS) to the value specified by ATB. Before running this procedure, please verify that the modified caused error can be reached by the process (e.g.: in an ambient temperature process with only heat control, the tune setpoint value *SP - ATB* is quite impossible to be reached).





S2_HC_EZ_TUNE

FB Prototype

S2_HC_EZ_TUNE

Input parameters

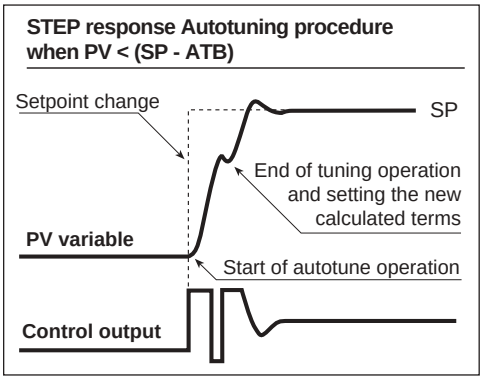
Input	Type	Description
ENABLE	BOOL	Function block enable
SP	REAL	Setpoint value [0...100.0]
MEAS	REAL	Measure value [0...100.0]
ATB	REAL	Threshold value [0...100.0]
OUTH_HI	REAL	Limit value for heat action output [0...100.0]
OUTC_HI	REAL	Limit value for cool action output [0...100.0]

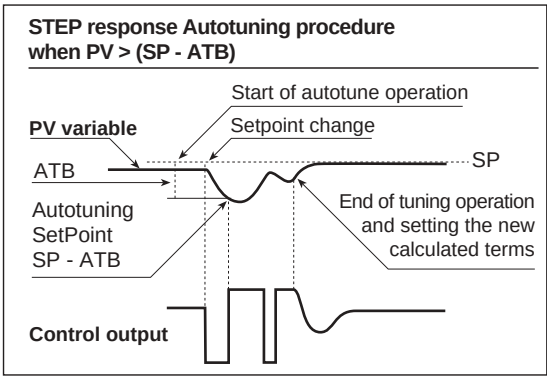
Output parameters

Output	Type	Description
TOUT	REAL	Function block output [OUTC_HI... OUTH_HI]
TPB	REAL	Calculated proportional band [%]
TTI	REAL	Calculated integral time [s]
TTD	REAL	Calculated derivative time [s]
TDBMP	BOOL	Bumpless command for the P.I.D.
TRUNNING	BOOL	Running flag
TVALID	BOOL	Successful tuning calculation flag

Description

This function block is able to perform a tuning calculation using the "*Step Response*" method with a ATB parameter *for a double action process ("heat and cool")*. It works together a PID block with which shares the inputs and the outputs from and for the process and an automatic/manual station with *Split Range* capability. The *ATB* value affects the quality of the results of the method: higher is its value, better are the results, but heavier is the perturbance caused to the control process; the solution is to choose a correct compromise. At the moment of the run command of the tune process, if *MEAS* is close to *SP*, the controller forces the output to the lower limit value (i.e. max value for the cool action) in order to increase the error (SP - MEAS) to the value specified by ATB. Before running this procedure, please verify that the modified caused error can be reached by the process (e.g.: in an ambient temperature process with only heat control, the tune setpoint value *SP - ATB* is quite impossible to be reached).





3-3-10 S2_MV

FB Prototype

S2_MV

Input parameters

Input	Type	Description
MVH	REAL	Analog input, corresponding to the control output (also named manipulated output) of the Heat P.I.D. module. Value in [0…100.0]
MVC	REAL	Analog input, corresponding to the control output (also named manipulated output) of the Cool P.I.D. module. Value in [0 … 100.0]
DBND	REAL	Deadband, with range from [-10.0...+10.0]
DTRK	BOOL	Digital tracking Command
TRK	REAL	Tracking signal [-100...+100.0]
DAM	BOOL	Digital AutoMan Command
AM_VAL	REAL	Manual Value [0...100.0]
DAUX	BOOL	Digital Auxiliary Command
AUX	REAL	Auxiliary Signal [0…100.0]

Output parameters

Output	Type	Description
OUTH	REAL	Control Output for HEAT [0...100.0]
OUTC	REAL	Control Output for COOL [0...100.0]
OUTHC	REAL	Control Output for visualization [-100.0...+100.0]
DBMP	BOOL	BUMPLESS command
OVERLAP	BOOL	Active if both OUTH and OUTC are active

Description

This module is an Output station for an Heat/Cool loop, providing the functionalities the operator needs to interact with the loop. The loop is controlled by two PIDs, connected downstream to this module; each one is dedicated to one channel.
By a set of digital inputs the S2_HCMV_VAL function block provides auto/manual/track/aux operating mode as listed above, ordered according to their priority level.

Trk *DTRK*
AutoMan *DAM*
Aux *DAUX*

Depending on these digital inputs, the functionality of the module changes as follows:

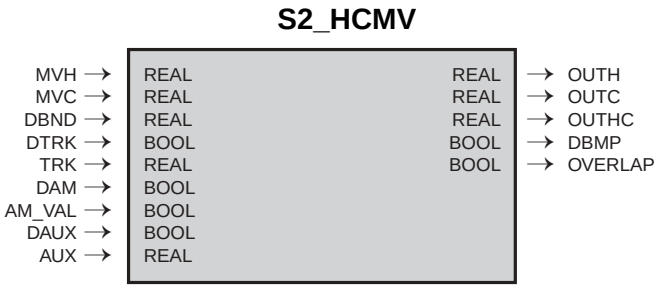
- If none of these digital signals are active, the module does not alter the control output signals, that are led, unchanged, directly from the input *MVC* and *MVH* to the output *OUTH* and *OUTC*. The *DBND* parameter defines how the Cool and the Heat channel interact each other. When it is positive, this parameter defines the dead band width, centered around the output value 0. It means that for each output value included in the deadband both the *OUTC* and the *OUTH* output are 0. When it is negative, the opposite

occurs; for each value included in the dead band (now named cross band), both the *OUTC* and the *OUTH* are different from 0.

- If *DTRK* (Track) is active, the outputs are set according to the value of the analog input *TRK*. This signal has a range from *[-100.0...+100.0]* and it drives both *OUTH* and *OUTC*. When its value is in the range -100.0...0, it means that a Cool action must be taken; therefore, *OUTH* is set to 0.0, while *OUTC* is set to the value of *TRK*, with the sign inverted. Inversely, when it is in the range 0…100.0, *OUTC* is set to 0.0, while *OUTH* is set to *TRK*. Please note that the tracking value is not affected by the *DBND* parameter.
- If the *DAM* input is active (and *DTRK* is inactive) the outputs are set according to the value of the analog input *AM_VAL*. This signal has a range from *[-100.0...+100.0]* and it drives both *OUTH* and *OUTC*. When its value is in the range -100.0...0, it means that a Cool action must be taken; therefore, *OUTH* is set to 0.0, while *OUTC* is set to the value of *TRK*, with the sign inverted. Inversely, when it is in the range 0...100.0, *OUTC* is set to 0.0, while *OUTH* is set to *TRK*. These values are valid if the *DBND* of the phase splitter is set to 0.0, otherwise they slightly change.
- If *DAUX* (Aux) is active(and both *DAUX* and *DTRK* are inactive), the outputs are set according to the value of the analog input *AUX*. This signal has a range from *[-100.0...+100.0]* and it drives both *OUTH* and *OUTC*. When its value is in the range -100.0...0, it means that a Cool action must be taken; therefore, *OUTH* is set to 0.0, while *OUTC* is set to the value of *AUX*, with the sign inverted. Inversely, when it is in the range 0...100.0, *OUTC* is set to 0.0, while *OUTH* is set to *AUX*. These values are valid if the *DBND* of the phase splitter is set to 0.0, otherwise they slightly change.

3-3-11 S2_HCMV

FB Prototype



Input parameters

Input	Type	Description
MVH	REAL	Analog input, corresponding to the control output (also named manipulated output) of the Heat PID module. Value in [0…100.0].
MVC	REAL	Analog input, corresponding to the control output (also named manipulated output) of the Cool PID module. Value in [0…100.0].
DBND	REAL	Deadband, with range from [-10.0…+10.0]
DTRK	BOOL	Digital Tracking Command
TRK	REAL	Tracking signal [-100.0…100.0]
DAM	BOOL	Digital AutoMan Command
AM_VAL	REAL	Manual Value [-100.0…100.0]
DAUX	BOOL	Digital Auxiliary Command
AUX	REAL	Auxiliary Signal [-100.0…100.0]

Output parameters

Output	Type	Description
OUTH	REAL	Control Output for HEAT [0 … 100.0]
OUTC	REAL	Control Output for COOL [0 … 100.0]
OUTHC	REAL	Control Output for visualization [-100.0 … 100.0]
DBMP	BOOL	BUMPLESS command
OVERLAP	BOOL	Active if both OUTH and OUTC are active

Description

This module is an Output station for an Heat/Cool loop, providing the functionalities the operator needs to interact with the loop. The loop is controlled by two PIDs, connected downstream to this module; each one is dedicated to one channel.
By a set of digital inputs the S2_HCMV_VAL function block provides auto/manual/track/aux operating mode as listed above, ordered according to their priority level.

Trk *DTRK*
AutoMan *DAM*
Aux *DAUX*

Depending on these digital inputs, the functionality of the module changes as follows:

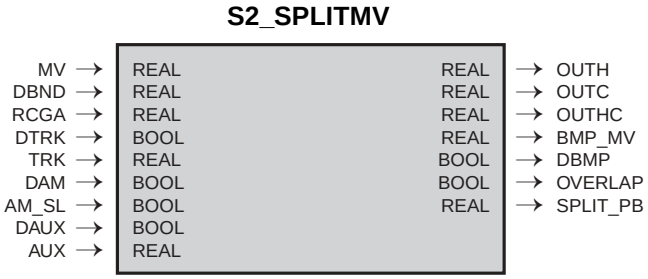
- If none of these digital signals are active, the module does not alter the control output signals, that are led, unchanged, directly from the input *MVC* and *MVH* to the output *OUTH* and *OUTC*. The *DBND* parameter defines how the Cool and the Heat channel interact each other. When it is positive, this parameter defines the dead band width, centered around the output value 0. It means that for each output value included in the deadband both

the *OUTC* and the *OUTH* output are 0. When it is negative, the opposite occurs; for each value included in the dead band (now named cross band), both the *OUTC* and the *OUTH* are different from 0.

- If *DTRK* (Track) is active, the outputs are set according to the value of the analog input *TRK*. This signal has a range from *[-100.0 … +100.0]* and it drives both *OUTH* and *OUTC*. When its value is in the range -100.0...0, it means that a Cool action must be taken; therefore, *OUTH* is set to 0.0, while *OUTC* is set to the value of *TRK*, with the sign inverted. Inversely, when it is in the range 0…100.0, *OUTC* is set to 0.0, while *OUTH* is set to *TRK*.
  Please note that the tracking value is not affected by the *DBND* parameter.
- If the *DAM* input is active (and *DTRK* is inactive) the outputs are set according to the value of the analog input *AM_VAL*. This signal has a range from *[-100.0...+100.0]* and it drives both *OUTH* and *OUTC*. When its value is in the range -100.0...0, it means that a Cool action must be taken; therefore, *OUTH* is set to 0.0, while *OUTC* is set to the value of TRK, with the sign inverted. Inversely, when it is in the range 0...100.0, *OUTC* is set to 0.0, while *OUTH* is set to *TRK*. These values are valid if the *DBND* of the phase splitter is set to 0.0, otherwise they slightly change.
- If *DAUX* (*Aux*) is active(and both *DAUX* and *DTRK* are inactive), the outputs are set according to the value of the analog input *AUX*. This signal has a range from *[-100.0...+100.0]* and it drives both *OUTH* and *OUTC*. When its value is in the range -100.0...0, it means that a Cool action must be taken; therefore, *OUTH* is set to 0.0, while *OUTC* is set to the value of *AUX*, with the sign inverted. Inversely, when it is in the range 0...100.0, *OUTC* is set to 0.0, while *OUTH* is set to *AUX*. These values are valid if the *DBND* of the phase splitter is set to 0.0, otherwise they slightly change.

3-3-12 S2_SPLITMV

FB Prototype



Input parameters

Input	Type	Description
MV	REAL	Controlled variable (also named manipulated variable). It is generated by the PID module with range [0 … 100.0]
DBND	REAL	Deadband, with range from [-10.0 … +10.0]
RCGA	REAL	Relative Cool Gain
DTRK	BOOL	Digital Tracking Command
TRK	REAL	Tracking signal [0…100.0]
DAM	BOOL	Digital AutoMan Command
AM_VAL	REAL	Manual Value [0 … 100.0]
DAUX	BOOL	Digital Auxiliary Command
AUX	REAL	Auxiliary Signal [0…100.0]

Output parameters

Output	Type	Description
OUTH	REAL	Control Output for HEAT [0 … 100.0]
OUTC	REAL	Control Output for COOL [0 … 100.0]
OUTHC	REAL	Control Output for visualization [-100.0 … 100.0]
BMP_MV	REAL	BUMPLESS output value
DBMP	BOOL	BUMPLESS command
OVERLAP	BOOL	Active if both OUTH and OUTC are active
SPLIT_PB	REAL	Correction factor for PB to obtain a PB value for the PID to apply cool action

Description

This module is an Output station for an Heat/Cool loop, providing the functionalities the operator needs to interact with the loop. The loop is controlled by one PID, connected downstream to this module; when the PID output value is between [50.0...0] the cool action is active, when the PID output value is between [50.0...100.0] the heat action is active. The function block supply, in the output SPLIT_PB, a value of 1.0 when heat action is actuated and the value of RCGA when cool action is actuated; to obtain the right process regulation to the PB input of the PID Block must be passed the PB calculated for the heat process regulation divided by the SPLIT_PB value.
By a set of digital inputs the S2_SPLITMV_VAL function block provides auto/manual/track/aux operating mode as listed above, ordered according to their priority level.

Trk *DTRK*
AutoMan *DAM*
Aux *DAUX*

Depending on these digital inputs, the functionality of the module changes as follows:

- If none of these digital signals are active, the module split the input *MV* into the two action *MVH* and *MVC*. The *DBND* parameter defines how the Cool and the Heat channel interact each other. When it is positive, this parameter defines the dead band width, centered around the value 50.0 of the input.
  It means that when the input is in the range [50.0 – DBND…50.0 + DBND] both the *OUTC* and the *OUTH* output are 0. When it is negative, the opposite occurs; for a value included in the dead band, both the *OUTC* and the *OUTH* are different from 0.
- If *DTRK* (Track) is active, the outputs are set according to the value of the analog input *TRK*. This signal has a range from [0...100.0] and it drives both *OUTH* and *OUTC*. When its value is in the range 0...50.0, it means that a Cool action must be taken; therefore, *OUTH* is set to 0.0, while *OUTC* is set to *[RCGA * 2.0 * (50.0 - TRK)]*. Inversely, when it is in the range 50.0...100, *OUTC* is set to 0, while *OUTH* is set to *[2.0 * (TRK – 50.0)]*.
  Please note that the tracking value is not affected by the *DBND* parameter.
- If the *DAM* input is active (and *DTRK* is inactive) the outputs are set according to the value of the analog input *AM_VAL*. This signal has a range from [0.0...100.0] and it drives both *OUTH* and *OUTC*. When its value is in the range 0.0 to 50.0, it means that a Cool action must be taken; therefore, *OUTH* is set to 0, while *OUTC* is set to *[RCGA * 2.0 * (50.0 – AM_VAL)]*. Inversely, when it is in the range 50.0 to 100, *OUTC* is set to 0, while *OUTH* is set to *[2.0 * (AM_VAL – 50.0)]*. These values are valid if the *DBND* of the phase splitter is set to 0, otherwise they slightly change.
- If *DAUX* (Aux) is active (and both *DAUX* and *DTRK* are inactive), the outputs are set according to the value of the analog input *AUX*. This signal has a range from [0.0...100.0] and it drives both *OUTH* and *OUTC*. When its value is in the range 0.0...50.0, it means that a Cool action must be taken; therefore, *OUTH* is set to 0, while *OUTC* is set to *[RCGA * 2.0 * (50.0 - AUX)]*. Inversely, when it is in the range 50.0...100, *OUTC* is set to 0, while *OUTH* is set to *[2.0 * (AUX – 50.0)]*. These values are valid if the *DBND* of the phase splitter is set to 0, otherwise they slightly change.

3-3-13 S2_FILTER

FB Prototype

S2_FILTER

Input parameters

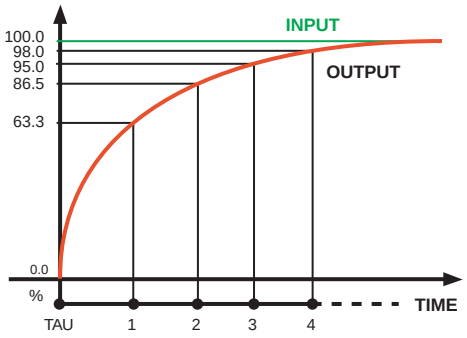
Input	Type	Description
INPUT	REAL	Filter input (default value = 0.0)
TAU	REAL	Filter time constant: a value of 0.0 is intended to disable the filter action and the output will assume the input value [s] (default value = 0.1)

Output parameters

Output	Type	Description
OUTPUT	REAL	Filter output (default value = 0.0)

Description

The FB is able to perform a first order filter on the signal connected as input. Please note that the filter constant TAU has to be greater then the cycle time used for the task where the instance of this FB is created.



Chapter 4
AsconCPULib

4-1 Purpose

The purpose of this document is to provide a complete description of the library *AsconCPULib*.

Caution
This library can be used **only** with the **CU-02** unit.

4-2 Description

The *AsconCPULib* is a function block library that allows the access to the control unit (CU-02) device of the Ascon sigma**due** line, from the OpenPCS programming tool. These FBs allow the user to set and manage the CANopen network activities: diagnostic, failure management of the connected devices, synchronization,…

The table here reported gives the complete list of the function blocks of the library. Please note that some of these must be considered as *system function blocks:* in fact these FBs implement particular functionalities dedicated to the global managment of the CANopen network.

Function Block Name	Description
S2_CU02	Function block interface for the Control Unit Module
SPLIT_ENABLE	Extract the enable signal from the Network connection of the modules
SET_TT (note)	Set the communication type of a CANopen node
SET_TT_MODULE (note)	Set the transmission type parameters of all PDOs of a device
RECOGNIZESIGMAIO (note)	Recognize the module name of a device coming from the sigma**due** I/O line

Note: System Function Block

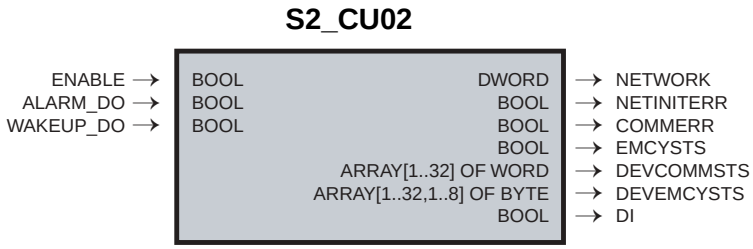
4-2-1 CANopen network definition

The use of the S2_CU02 joined with the library for the IO modules (please see the *AsconBasicIOLib* documentation), allows the user to define easly the architecture of his CANopen network. Using the *NETWORK* output signal of the S2_CU02 is possibile to connect all the NETWORK inputs of the IO function blocks of the *AsconBasicIOLib* library in order to grant an effective control of the communication status, and a correct management of the possible failures[1]. The picture on the right shows an example of a simple system with a CU-02 and three IO modules.

4-3 Function Block Description

4-3-1 S2_CU02

FB Prototype



Input parameters

Label	Type	Description
Enable	BOOL	Function Block Enable
Alarm_DO	BOOL	AlarmOut Output State
Wakeup_DO	BOOL	WakeOut Output State

Output parameters

Label	Type	Description
Network	DWORD	Network Line Connection
NetInitErr	BOOL	Network Initialization Error
CommErr	BOOL	Error Communication Flag
EmcySts	BOOL	Emergency Error Flag
DevCommSts	ARRAY[1..32] OF WORD	Communication Status Node Array
DevEmcySts	ARRAY[1..32,1..8] OF BYTE	Emergency Message Node Matrix
DI	BOOL	Digital Input State

Description

This Function Block performs a scan of the network in order to recognize the modules connected, and it is able to control the enable of each single IO FBs linked to the *NETWORK* output[2]. It is able to provide diagnostic activities in order to verify the proper CAN communication status, and it is also able to detect the emergency messages eventually generated by devices connected to the line. The output *NETINITERR* is used to signal an error occurred during the

1. The S2_CU02 function block needs some gloabal variables that must be linked to the active resource at compile time. In the Ascon Template project these variables are present in the file "*SystemDirectVar.POE*"
2. The S2_CU02 function block needs some gloabal variables that must be linked to the active resource at compile time. In the Ascon Template project these variables are present in the file "*SystemDirectVar.POE*"

initialization phase of the communication. The FB provide also a global indication of both communication error state and emergency error condition using the *COMMERR* and the *EMCYSTS* signals, and a punctual indication using the *DEVCOMMSTS* array and the *DEVEMCYSTS* matrix. Finally the FB is able to set the local digital output of the CPU and to read the state of the digital input.

CommErr Flag

This flag is TRUE if at least one of the devices connected to the CAN line has a communication problem. Please check the *DevCommSts* array to identify which is this node.

DevCommSts Array

Each element of this array contains the communication state of the relative node. The table here reported gives the numerical value of these states.

Element Value	Communication State
16#0000	INIT state
16#0001	RESET_COMM state
16#0002	RESET_APP state
16#0003	PREOPERATIONAL state
16#0004	STOPPED state
16#0005	OPERATIONAL state
16#0006	UNKNOWN state
16#0007	NOTAVAIL state

For more details about these values please refer to the "*CiA document DS301 – CANopen Application Layer and Communication Profile*" and to the Systec Manual "*CANopen Extension for IEC61131-3*". Please use the "*Diagnostic Application Note*" from Ascon for a complete description.

EmcySts Flag

This flag is TRUE if at least one of the devices connected to the CAN line sends an emergency object. Please check the *DevEmcySts* matrix to identify which is this node.

DevEmcySts Matrix

This matrix has a row for every device connected to the network, where is reported the content of the last error message. For the meaning of these messages please refer to the specific user manual of the device.

Alarm_DO, Wakeup_DO and DI

These signals are related with the physical IO of the CPU, where *ALARM_DO*, and *WAKEUP_DO* are the outputs, and *DI* is the input state.

4-3-2 SPLIT_ENABLE

FB Prototype

SPLIT_ENABLE

Input parameters

Label	Type	Description
Network	DWORD	CAN Network Connection
NodeID	USINT	CANopen Device Identifier

Output parameters

Label	Type	Description
ENABLE	BOOL	Function Block Output

Description This Function Block sets the boolean "*Enable*" output on the basis of the value of the n-th bit of the "*Network*" input. The position of the n-th bit is specified by the input "*NodeID".* Typically is used when it is needed to extract from the "*Network*" output of the FB "*ControlUnit*" the digital enable of a certain node connected to the network.

4-3-3 SET_TT (note)

FB Prototype

SET_TT

Note: System Function Block

Input parameters

Label	Type	Description
Enable	BOOL	Function Block Enable
NodeID	USINT	CANopen Device Identifier
Module	ASCON_MODULES	Device Name

Output parameters

Label	Type	Description
Confirm	BOOL	Function Block Confirm
Result	WORD	Error Code

Description This Function Block is used internally to set the communication of the module type specified by the input "*Module*" with the CANopen identifier specified by "*NodeID*". The Confirm "*Output*" is TRUE if the configuration is completed successfully, otherwise is FALSE and the output "*Result*" returns the error code. For the meaning of the error codes please refer to the Systec Manual "*CANopen Extension for IEC61131-3*".

4-3-4 SET_TT_MODULE (note)

FB Prototype

Note: System Function Block

Input parameters

Label	Type	Description
Node	USINT	CANopen Device Identifier
TT_Rx	ARRAY[1...4] of BOOL	Transmission Type for the RxPDOs
TT_Tx	ARRAY[1...4] of BOOL	Transmission Type for the TxPDOs

Output parameters

Label	Type	Description
Confirm	BOOL	Function Block Confirm
Result	WORD	Error Code

Description This Function Block is used internally to set the proper synchronous communication of a CANopen Device. If an element of an array is TRUE, the corresponding transmission type of the PDO is set to 1. If an element of an array is FALSE, the corresponding transmission type of the PDO is set to 0xFC. For more details about the meaning of the values used here please refer to the "*CiA document DS301 – CANopen Application Layer and Communication Profile*". The Confirm "*Output*" is TRUE if the configuration is completed successfully, otherwise is FALSE and the output "*Result*" returns the error code. For the meaning of the error codes please refer to the Systec Manual "*CANopen Extension for IEC61131-3*".

4-3-5 RECOGNIZESIGMAIO (note)

FB Prototype

Note: System Function Block

Input parameters

Label	Type	Description
DeviceType	DWORD	CANopen Device Identifier
DeviceName	DWORD	Transmission Type for the RxPDOs

Output parameters

Label	Type	Description
ModuleName	ASCON_MODULES	Function Block Confirm

Description This Function Block is used internally to recognize the IO module type starting from the parameters *DEVICETYPE* and *DEVICENAME*.

Chapter 5
AsconMBCommLib

5-1 Purpose

The purpose of this document is to supply a complete description of AsconMBCommLib Library project.

5-2 Description

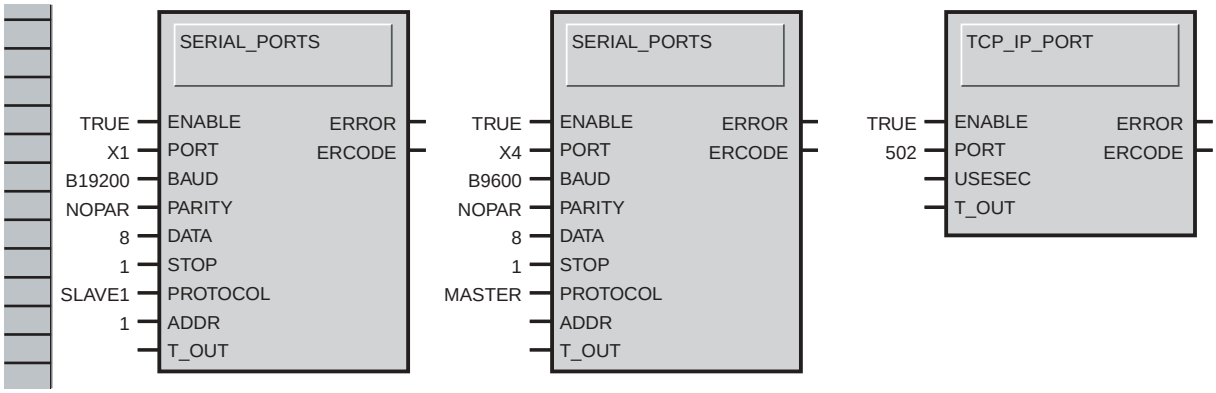
AsconMBCommLib is a Function Block library which simplifies the access to the MODBUS communication ports available in Ascon SpA sigma**due** line devices. The following is the list of FBs present in the library.

MB_MST_SYNC	Modbus Master: Synchronization of operations
MB_MST_RD_COIL	Modbus Master: Coil reading
MB_MST_WR_COIL	Modbus Master: Coil writing
MB_MST_RD_WORD	Modbus Master: Word reading
MB_MST_WR_WORD	Modbus Master: Word writing
MB_16WORD_TO_ARRAY	Modbus Master: packaging of 16 WORD in an array
MB_ARRAY_TO_16WORD	Modbus Master: un-packaging of an array into 16 WORD
MB_MST_RD8_DINT	Modbus Master: conversion and management of 8 DINT read values
MB_MST_RD8_DWORD	Modbus Master: conversion and management of 8 DWORD read values
MB_MST_RD8_REAL	Modbus Master: conversion and management of 8 REAL read values
MB_MST_RD8_UDINT	Modbus Master: conversion and management of 8 UDINT read values
MB_MST_WR8_DINT	Modbus Master: conversion and management of 8 DINT write values
MB_MST_WR8_DWORD	Modbus Master: conversion and management of 8 DWORD write values
MB_MST_WR8_REAL	Modbus Master: conversion and management of 8 REAL write values
MB_MST_WR8_UDINT	Modbus Master: conversion and management of 8 UDINT write values
MB_SLV_RD8_DWORD	Modbus Slave: reading of 8 DWORD values
MB_SLV_RD8_REAL	Modbus Slave: reading of 8 REAL values
MB_SLV_RD16_WORD	Modbus Slave: reading of 16 WORD values
MB_SLV_RD32_DIGITAL	Modbus Slave: reading of 32 digital values
MB_SLV_RD_DIGITAL	Modbus Slave: reading of 1 digital value
MB_SLV_RD_DWORD	Modbus Slave: reading of 1 DWORD value
MB_SLV_RD_REAL	Modbus Slave: reading of 1 REAL value
MB_SLV_RD_WORD	Modbus Slave: reading of 1 WORD value
MB_SLV_WR8_DWORD	Modbus Slave: writing of 8 DWORD values

MB_SLV_WR8_REAL	Modbus Slave: writing of 8 REAL values
MB_SLV_WR16_WORD	Modbus Slave: writing of 16 WORD values
MB_SLV_WR32_DIGITAL	Modbus Slave: writing of 32 digital values
MB_SLV_WR_DIGITAL	Modbus Slave: writing of 1 digital value
MB_SLV_WR_DWORD	Modbus Slave: writing of 1 DWORD value
MB_SLV_WR_REAL	Modbus Slave: writing of 1 REAL value
MB_SLV_WR_WORD	Modbus Slave: writing of 1 WORD value
MODEM_CHECK	Modem operational verification
MODEM_CONF	Modem configuration management
MODEM_SMS_SEND	Modem SMS (Short text Message Service) send management
MP_SERIAL_PORTS	Set the configuration for the Modbus RTU ports of the MP Unit
PROFIBUS_PORT	Set the configuration and data exchange for the Profibus DP port of the CU unit
SEND_EMAIL	Set the configuration for a client SMTP to send e-mail
SERIAL_PORTS	Set the configuration for the Modbus RTU ports of the CU unit
SYS_OPRS_MNGT	Set communication operational parameters on Modbus RTU and TCP agents
TCP_IP_PORT	Set the configuration for the Modbus TCP port

5-2-1 Communication port configuration

The *SERIAL_PORTS/MP_SERIAL_PORT* and *TCP_IP_PORT* FBs enable the communication port configuration. The communication parameters (baudrate, logic ports), the desired protocol type (MODBUS SLAVE, MODBUS MASTER), the desired communication port (RS232, RS485 or Ethernet connection) can be thus assigned. The figure below shows an example of port communication configuration.



For further details on the labels to be used for Function Blocks input parameter configuration, see individual module documentation.

5-2-2 Modbus Slave Protocol

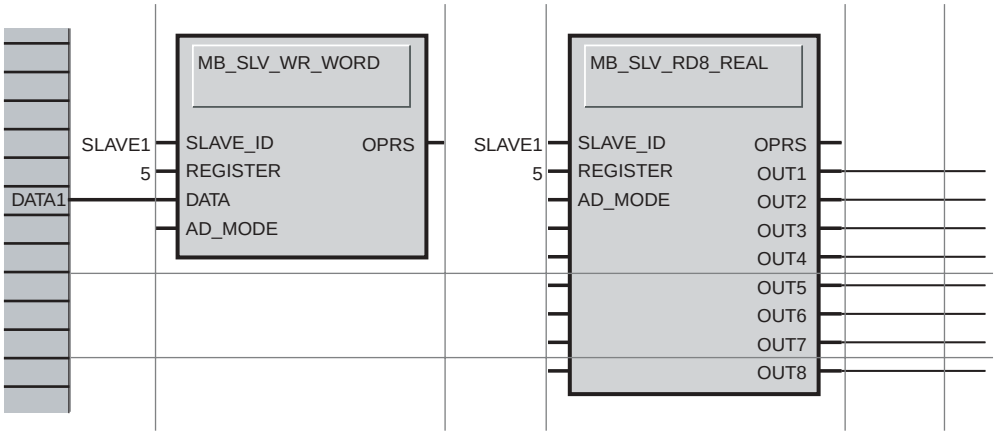
Once the communication port has been configured as slave (or server in case of MODBUS/TCP), the data exchange between the program written in the IEC61131 environment and the memory area, that can be seen from the master (or client in case of MODBUS/TCP), can occur in two different ways:

- by declaring and using the %M variables;
- by using the Function Blocks in AsconMBCommLib library.

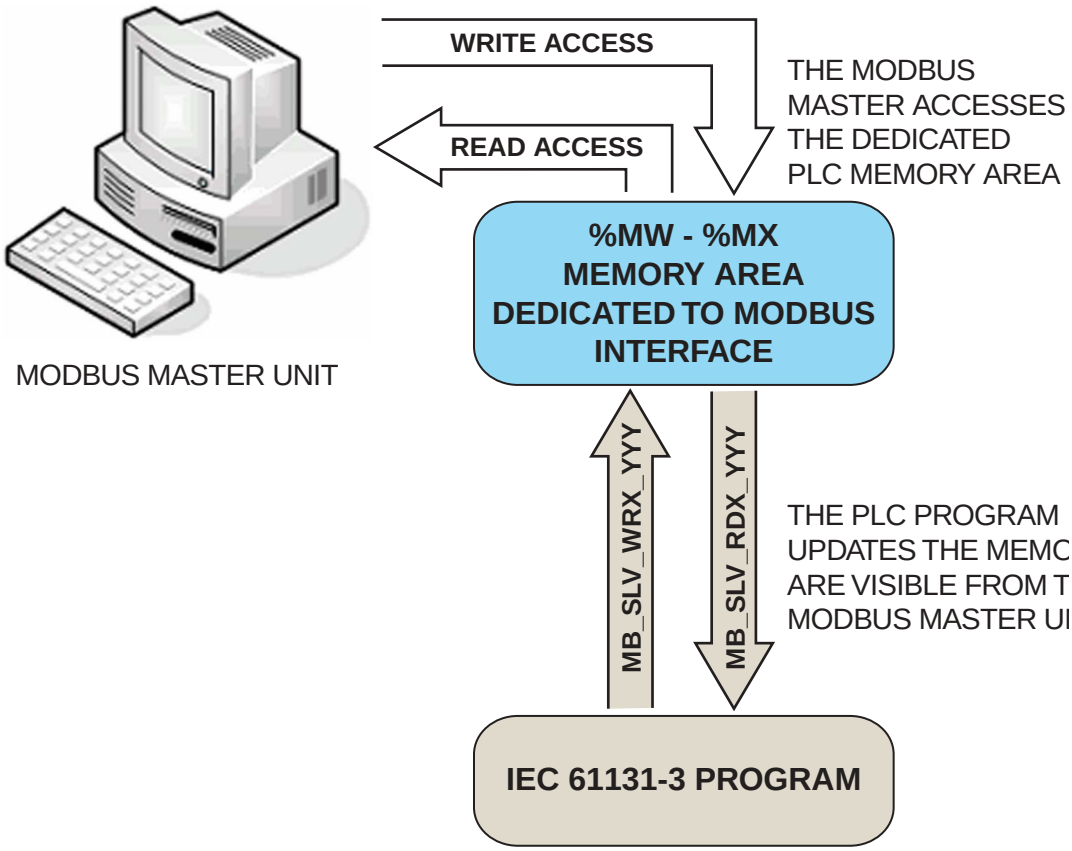
The first one is described in serial communications manual **[3]**, and it needs the declaration of variables in the %M area that will be assigned to the process variables.

The second mode provides the direct access to the central unit internal memory, that can be seen from the communications master (or client in case of MODBUS/TCP), without using %M variables.

The following is an example of memory access using the Function Blocks present in the library.



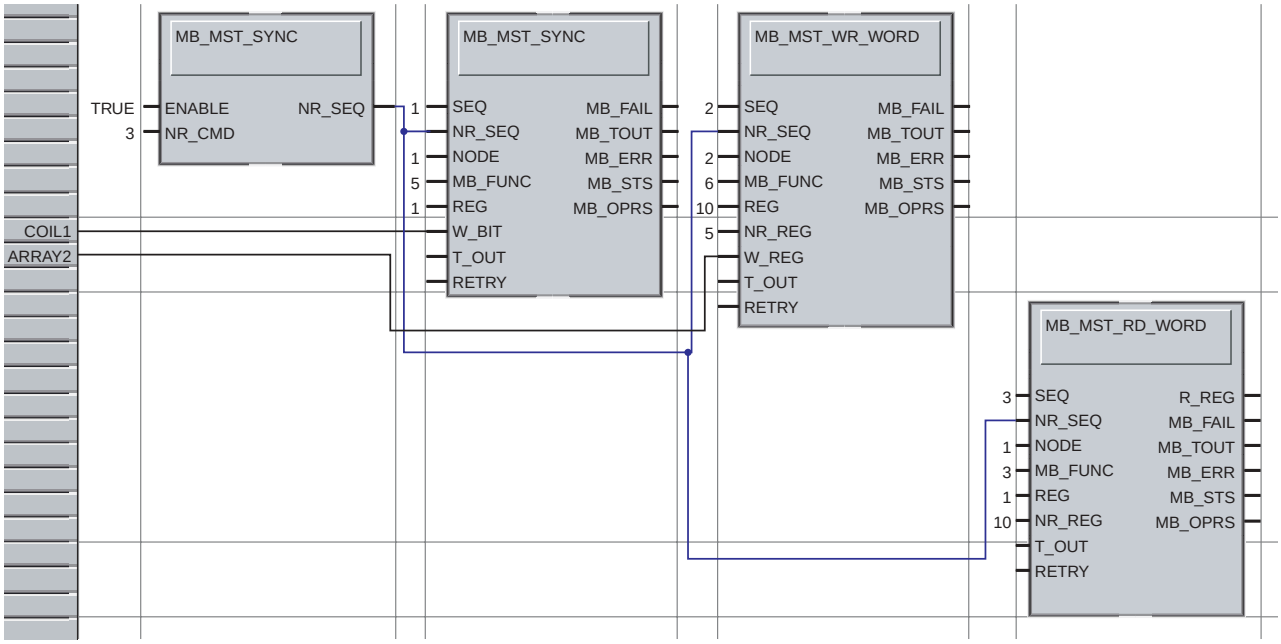
Note that in this case a word writing updates the memory area that the communications master (or client in case of MODBUS/TCP) will subsequently read. This means that *MB_SLV_XX_YYYY* Function Blocks are always to be used for SLAVE protocols and update the memory area that the communication smaster accesses. The following block diagram clarifies the access modes to the different memory areas.



For further details on these Function Blocks, see the specific documentation of the individual module.

5-2-3 Modbus Master protocol

The sigma**due** CPU device firmware includes the communications primitives for MODBUS MASTER RTU (see "*Serial Communication Function Block Library*"). The definition of Function Blocks simplifies its use and enables to define the sequence of messages to be generated by the MODBUS master, after correctly configurating the communications port. The following example shows how to use the library master Function Blocks[1].



In this case two different slave devices (address 1 and address 2) are accessed in sequence for three times. *MB_MST_SYNC* Function Block gives access to one of the three other present modules: each one will be enabled after the operation of the previous one is terminated, whereas the order is set by the *SEQ* input of each Function Block.

For further details on these Function Blocks, see the specific documentation of the individual module.

1. For their correct operation, *MB_MST_xx_yyy* Function Blocks need some global variables that have to be linked during compilation. In Ascon template these variables are included in "*SystemDirectVar.POE*" and "*SystemVar.POE*" files.

5-3 Description of the individual Function Blocks

5-3-1 MODBUS MASTER SYNCHRONIZER

FB Prototype

MB_MST_SYNC

Input parameters

Label	Type	Description	Range
ENABLE	BOOL	Enable/Disable the Modbus master synchronization	
NR_CMD	USINT	Number of inquires to be managed [num]	1... 255

Output parameters

Label	Type	Description
NR_SEQ	USINT	Number of the active control

Description

This function block perform a synchronization of the Modbus Master Inquires using the dedicated function blocks MB_MST_RD_WORD, MB_MST_WR_WORD, MB_MST_RD_COIL and MB_MST_WR_COIL.

Reference Table

Input	Options
NR_CMD	From 1 to 255. Default value 1

Output	Options
NR_SEQ	From 1 to 255

5-3-2 Modbus MASTER COIL READ

FB Prototype

MB_MST_RD_COIL

Input parameters

Label	Type	Description	Range
SEQ	USINT	Sequence identifiable number. This number **MUST BE UNIQUE** for each MB_MST series FB [num]	1... 255
NR_SEQ	USINT	Input that **MUST** be connected to the MB_MST_Sync output [num]	
NODE	USINT	Node of the Slave Address to be Inquired [num]	1... 247
MB_FUNC	USINT	Modbus Query Function 1 or 2	
ID_COIL	UINT	Modbus starting COIL address [num]	1... 65535
NR_COIL	UINT	Modbus number of COIL to manage [num]	1... 32
T_OUT	UINT	Communication Time Out factor of 100 ms (eg. 10 means 10 x 100 ms = 1 s) [num]	0... 65535
RETRY	USINT	Number of Retry on the same operation [num]	1... 255

Output parameters

Label	Type	Description
RD_BIT	DWORD	Dword containing the coils read masked in according to the NR_COIL value
MB_FAIL	BOOL	Modbus Operation Fail Status
MB_TOUT	BOOL	Modbus Timeout Error Status
MB_ERR	BOOL	Generic Modbus Error Event
MB_STS	USINT	Modbus Communication Status
MB_OPRS	USINT	Modbus function block Operations Status

Description

This function block performs a request of max. 32 continuous Modbus COILS in according to the parameters specified as Inputs.

Rerefence table

Input

Label	Description
SEQ	Value from 1 to what specified as NR_CMD for the MB_MST_Sync function block (max. 255)
NODE	1... 247
MB_FUNC	1 or 2, in according to the Modbus Specifications
ID_COIL	1... 65535
NR_COIL	1... 32
T_OUT	1... 65535. Deafult value 10 (1000 ms)
RETRY	1... 255. Deafult value 1

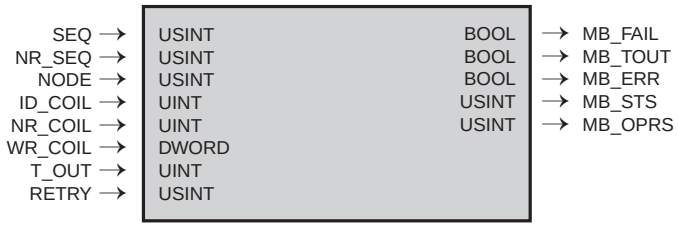
Output

Label	Description
MB_STS	0 = No operation/OK 1 = Transaction in progress 2 = Data available 3 = Timeout expired 4 = Agent event error
MB_OPRS	0 = No operation / OK 1 = Port not available 2 = Timeout expired 4 = CRC error 16 = ExCode error

5-3-3 Modbus MASTER COIL WRITE

FB Prototype

MB_MST_WR_COIL



Input parameters

Label	Type	Description	Range
SEQ	USINT	Sequence identifiable number. This number **MUST BE UNIQUE** for each MB_MST series FB [num]	1... 255
NR_SEQ	USINT	Input that **MUST** be connected to the MB_MST_Sync output [num]	
NODE	USINT	Node of the Slave Address to be Inquired [num]	0... 247
MB_FUNC	USINT	Modbus Query Function 5 (Single COIL) or 15 (Multiple COILS)	
ID_COIL	UINT	Modbus starting COIL address. [num]	1... 65535
NR_COIL	UINT	Modbus number of COIL to manage. [num]	1... 32
WR_COIL	DWORD	Dword containing the coils to be written masked in according to the NR_COIL value	
T_OUT	UINT	Communication Time Out factor of 100 ms (eg. 10 means 10 x 100 ms = 1 s) [num]	0... 65535
RETRY	USINT	Number of Retry on the same operation [num]	1... 255

Output parameters

Label	Type	Description
MB_FAIL	BOOL	Modbus operation fail status
MB_TOUT	BOOL	Modbus timeout error status
MB_ERR	BOOL	Generic Modbus error event
MB_STS	USINT	Modbus communication status
MB_OPRS	USINT	Modbus function block operations status

Description

This function block performs a writing of max 32 continuous Modbus COILS in according to the parameters specified as Inputs.

Rerefence table

Input

Label	Description
SEQ	Value from 1 to what specified as NR_CMD for the MB_MST_Sync function block (max. 255)
NODE	From 0... 247
MB_FUNC	5 or 15, in according to the Modbus specifications
ID_COIL	From 1... 65535
NR_COIL	From 1... 32
T_OUT	From 1... 65535. Deafult value 10 (1000 ms)
RETRY	From 1... 255. Deafult value 1

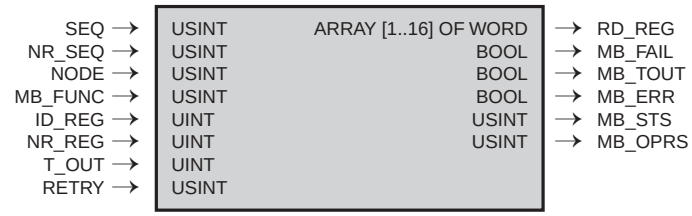
Output

Label	Description
MB_STS	0 = No operation/OK 1 = Transaction in progress 2 = Data available 3 = Timeout expired 4 = Agent error event
MB_OPRS	0 = No operation/OK 1 = Port not available 2 = Timeout expired 4 = CRC error 16 = ExCode error

5-3-4 Modbus MASTER WORD READ

FB Prototype

MB_MST_RD_WORD



Input parameters

Label	Type	Description	Range
SEQ	USINT	Sequence identifiable number. This number MUST BE UNIQUE for each MB_MST series FB [num]	1... 255
NR_SEQ	USINT	Input that MUST be connected to the MB_MST_Sync output [num]	
NODE	USINT	Node of the Slave Address to be Inquired. [num]	1... 247
MB_FUNC	USINT	Modbus query function 3 or 4	
ID_REG	UINT	Modbus starting REGISTERS address. [num]	1... 65535
NR_REG	UINT	Modbus number of REGISTERS to manage. [num] (admitted range 1... 16)	1... 16
T_OUT	UINT	Communication Time Out factor of 100 ms (eg. 10 means 10 x 100 ms = 1s) . [num]	0... 65535
RETRY	USINT	Number of Retry on the same operation. [num]	1... 255

Output parameters

Label	Type	Description
RD_REG	ARRAY [1..16] OF WORD	Array of 16 WORDS containing the read values
MB_FAIL	BOOL	Modbus operation fail status
MB_TOUT	BOOL	Modbus timeout error status
MB_ERR	BOOL	Generic Modbus error event
MB_STS	USINT	Modbus communication status
MB_OPRS	USINT	Modbus function block operations status

Description

This function block performs a request of max 16 continuous Modbus REGISTERS in according to the parameters specified as Inputs.

Rerefence table

Input

Input	Description
SEQ	Value from 1 to what specified as NR_CMD for the MB_MST_Sync function block (max. 255)
NODE	1... 247
MB_FUNC	3 or 4, in according to the Modbus Specifications
ID_REG	1... 65535
NR_REG	1... 16
T_OUT	1... 65535. Default value 10 (1000 ms)
RETRY	1... 255. Default value 1

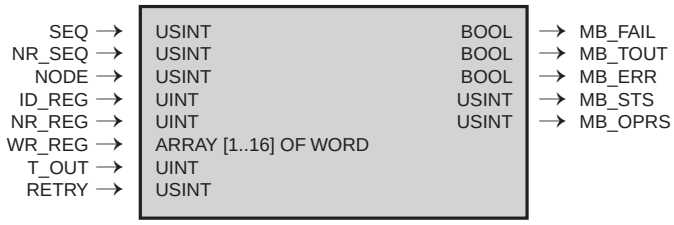
Output

Label	Description
MB_STS	0 = No operation / OK 1 = Transaction in progress 2 = Data available 3 = Timeout expired 4 = Agent error event
MB_OPRS	0 = No operation / OK 1 = Port not available 2 = Timeout expired 4 = CRC error 16 = ExCode error

5-3-5 Modbus MASTER WORD WRITE

FB Prototype

MB_MST_WR_WORD



Input parameters

Input	Type	Description	Range
SEQ	USINT	Sequence identifiable number. This number **MUST BE UNIQUE** for each MB_MST series FB [num]	1... 255
NR_SEQ	USINT	Input that **MUST** be connected to the MB_MST_Sync output. [num]	
NODE	USINT	Node of the Slave Address to be Inquired [num]	0... 247
MB_FUNC	USINT	Modbus Query Function 6 (Single REGISTERS) or 16 (Multiple REGISTERS)	
ID_REG	UINT	Modbus starting REGISTERS address [num]	1... 65535
NR_REG	UINT	Modbus number of REGISTERS to manage [num] (admitted range 1... 16)	1... 32
WR_REG	ARRAY [1..16] OF WORD	Array of 16 WORDS to be written	
T_OUT	UINT	Communication Time Out factor of 100 ms (eg. 10 means 10 x 100 ms = 1s) [num]	0... 65535
RETRY	USINT	Number of retry on the same operation	1... 255

Output parameters

Output	Type	Description
MB_FAIL	BOOL	Modbus operation fail status
MB_TOUT	BOOL	Modbus timeout error status
MB_ERR	BOOL	Generic Modbus error event
MB_STS	USINT	Modbus communication status
MB_OPRS	USINT	Modbus function block operations status

Description

This function block performs a writing of max 16 continuous Modbus REGISTERS in according to the parameters specified as Inputs[1].

1. The MB_MST_xx_yyy function blocks need a series of global variables that must be linked at compile time. In the Ascon Template project these are defined in the files “SystemDirectVar.POE” and “SystemVar.POE”

Rerefence table

Input

Input	Description
SEQ	Value from 1 to what specified as NR_CMD for the MB_MST_Sync function block (max. 255)
NODE	0... 247
MB_FUNC	6 or 16, in according to the Modbus Specifications
ID_REG	1... 65535
NR_REG	1... 16
T_OUT	1... 65535. Deafult value 10 (1000 ms)
RETRY	1... 255. Deafult value 1

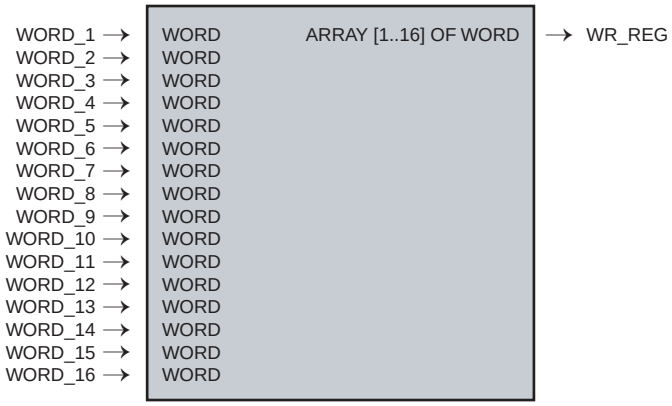
Output

Label	Description
MB_STS	0 = No operation / OK 1 = Transaction in progress 2 = Data available 3 = Timeout expired 4 = Agent error event
MB_OPRS	0 = No operation / OK 1 = Port not available 2 = Timeout expired 4 = CRC error 16 = ExCode error

5-3-6 MODBUS MASTER 16WORD TO ARRAY

FB Prototype

MB_16WORD_TO_ARRAY



Input parameters

Input	Type	Description	Range
WORD_1	WORD	1st word to be packed [num]	0... 65535
WORD_2	WORD	2nd word to be packed [num]	0... 65535
WORD_3	WORD	3rd word to be packed [num]	0... 65535
WORD_4	WORD	4th word to be packed [num]	0... 65535
WORD_5	WORD	5th word to be packed [num]	0... 65535
WORD_6	WORD	6th word to be packed [num]	0... 65535
WORD_7	WORD	7th word to be packed [num]	0... 65535
WORD_8	WORD	8th word to be packed [num]	0... 65535
WORD_9	WORD	9th word to be packed [num]	0... 65535
WORD_10	WORD	10th word to be packed [num]	0... 65535
WORD_11	WORD	11th word to be packed [num]	0... 65535
WORD_12	WORD	12th word to be packed [num]	0... 65535
WORD_13	WORD	13th word to be packed [num]	0... 65535
WORD_14	WORD	14th word to be packed [num]	0... 65535
WORD_15	WORD	15th word to be packed [num]	0... 65535
WORD_16	WORD	16th word to be packed [num]	0... 65535

Output parameters

Label	Type	Description
WR_REG	ARRAY [1..16] OF WORD	Array including each single input word

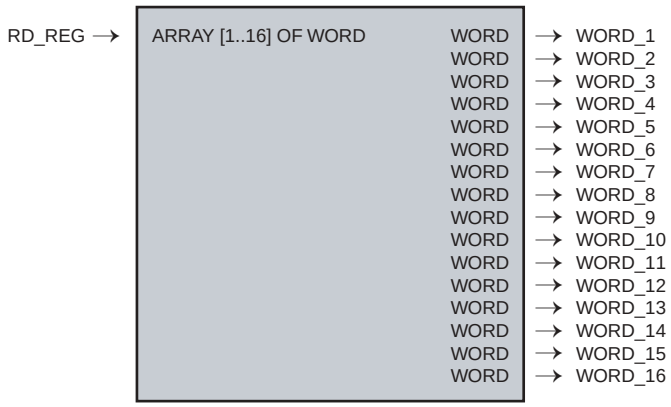
Description

The main goal of this function block is to pack 16 single word into an array of 16 word elements. The first input word will be copied to the first array element, the second input word to the second element and so on. This function block can be particularly powerful if used in conjunction with the function block MB_MST_WR_WORD because it allows to “prepare” the array required by its input WR_REG.

5-3-7 MODBUS MASTER ARRAY TO 16WORD

FB Prototype

MB_ARRAY_TO_16WORD



Input parameters

Label	Type	Description
RD_REG	ARRAY [1..16] OF WORD	Array including each single word to be unpacked

Output parameters

Label	Type	Description	Range
WORD_1	WORD	1st word to be unpacked [num]	0... 65535
WORD_2	WORD	2nd word to be unpacked [num]	0... 65535
WORD_3	WORD	3rd word to be unpacked [num]	0... 65535
WORD_4	WORD	4th word to be unpacked [num]	0... 65535
WORD_5	WORD	5th word to be unpacked [num]	0... 65535
WORD_6	WORD	6th word to be unpacked [num]	0... 65535
WORD_7	WORD	7th word to be unpacked [num]	0... 65535
WORD_8	WORD	8th word to be unpacked [num]	0... 65535
WORD_9	WORD	9th word to be unpacked [num]	0... 65535
WORD_10	WORD	10th word to be unpacked [num]	0... 65535
WORD_11	WORD	11th word to be unpacked [num]	0... 65535
WORD_12	WORD	12th word to be unpacked [num]	0... 65535
WORD_13	WORD	13th word to be unpacked [num]	0... 65535
WORD_14	WORD	14th word to be unpacked [num]	0... 65535
WORD_15	WORD	15th word to be unpacked [num]	0... 65535
WORD_16	WORD	16th word to be unpacked [num]	0... 65535

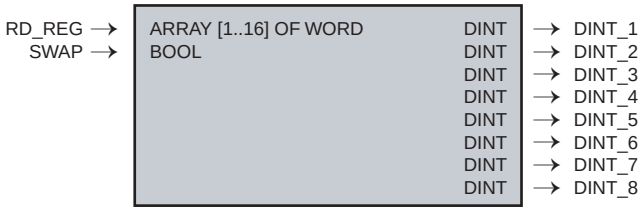
Description

The main goal of this function block is to unpack an array of 16 word elements into 16 single words. The first element of the array will be copied to the first output word, the second element to the second output word and so on. This function block can be particularly powerful if used in conjunction with the function block MB_MST_RD_WORD because it allows to split the array generated by its output RD_REG.

5-3-8 MODBUS MASTER CONVERSION TO 8 DINT VALUES

FB Prototype

MB_MST_RD8_DINT



Input parameters

Label	Type	Description	Range
RD_REG	ARRAY [1..16] OF WORD	Array including each single word to be unpacked	
SWAP	BOOL	Swap activation between Low and High word while converting data [bit]	FALSE = standard, TRUE = swapped

Output parameters

Label	Type	Description	Range
DINT_1	DINT	1st converted variable [num]	-2147483648... 2147483647
DINT_2	DINT	2nd converted variable [num]	-2147483648... 2147483647
DINT_3	DINT	3rd converted variable [num]	-2147483648... 2147483647
DINT_4	DINT	4th converted variable [num]	-2147483648... 2147483647
DINT_5	DINT	5th converted variable [num]	-2147483648... 2147483647
DINT_6	DINT	6th converted variable [num]	-2147483648... 2147483647
DINT_7	DINT	7th converted variable [num]	-2147483648... 2147483647
DINT_8	DINT	8th converted variable [num]	-2147483648... 2147483647

Description

The function block can be used to unpack an array of 16 word elements into 8 single variables as DINT data type. The first and second elements of the array will be converted, accordingly to the SWAP selection, into the first output DINT variable, the third and fourth elements into the second output variable and so on. This function block has been designed to work in conjunction with MB_MST_RD_WORD function block, when values coming from the slave device are formatted as DINT data type.

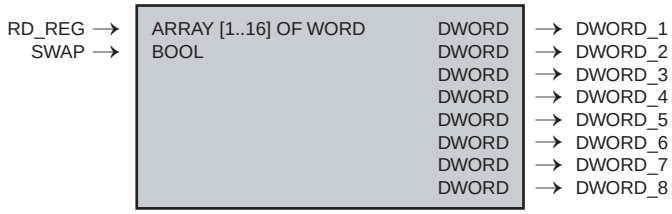
Default Values Table

Input	Default Value
SWAP	FALSE

5-3-9 MODBUS MASTER CONVERSION TO 8 DWORD VALUES

FB Prototype

MB_MST_RD8_DWORD



Input parameters

Label	Type	Description	Range
RD_REG	ARRAY [1..16] OF WORD	Array including each single word to be unpacked	
SWAP	BOOL	Swap activation between Low and High word while converting data [bit]	FALSE = standard, TRUE = swapped

Output parameters

Label	Type	Description	Range
DWORD_1	DWORD	1st variable to be converted [num]	0... 4294967295
DWORD_2	DWORD	2nd variable to be converted [num]	0... 4294967295
DWORD_3	DWORD	3rd variable to be converted [num]	0... 4294967295
DWORD_4	DWORD	4th variable to be converted [num]	0... 4294967295
DWORD_5	DWORD	5th variable to be converted [num]	0... 4294967295
DWORD_6	DWORD	6th variable to be converted [num]	0... 4294967295
DWORD_7	DWORD	7th variable to be converted [num]	0... 4294967295
DWORD_8	DWORD	8th variable to be converted [num]	0... 4294967295

Description

The function block can be used to unpack an array of 16 word elements into 8 single variables as DWORD data type. The first and second elements of the array will be converted, accordingly to the SWAP selection, into the first output DWORD variable, the third and fourth elements into the second output variable and so on. This function block has been designed to work in conjunction with MB_MST_RD_WORD function block, when values coming from the slave device are formatted as DWORD data type.

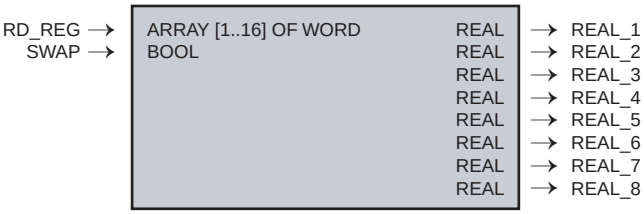
Default Values Table

Input	Default Value
SWAP	FALSE

5-3-10 MODBUS MASTER CONVERSION TO 8 REAL VALUES

FB Prototype

MB_MST_RD8_REAL



Input parameters

Label	Type	Description	Range
RD_REG	ARRAY [1..16] OF WORD	Array including each single word to be unpacked	
SWAP	BOOL	Swap activation between Low and High word while converting data [bit]	FALSE = standard, TRUE = swapped

Output parameters

Label	Type	Description	Range
REAL_1	REAL	1st converted variable [num]	$-3.4E^{-38}$... $3.4E^{+38}$
REAL_2	REAL	2nd converted variable [num]	$-3.4E^{-38}$... $3.4E^{+38}$
REAL_3	REAL	3rd converted variable [num]	$-3.4E^{-38}$... $3.4E^{+38}$
REAL_4	REAL	4th converted variable [num]	$-3.4E^{-38}$... $3.4E^{+38}$
REAL_5	REAL	5th converted variable [num]	$-3.4E^{-38}$... $3.4E^{+38}$
REAL_6	REAL	6th converted variable [num]	$-3.4E^{-38}$... $3.4E^{+38}$
REAL_7	REAL	7th converted variable [num]	$-3.4E^{-38}$... $3.4E^{+38}$
REAL_8	REAL	8th converted variable [num]	$-3.4E^{-38}$... $3.4E^{+38}$

Description

The function block can be used to unpack an array of 16 word elements into 8 single variables as DINT data type. The first and second elements of the array will be converted, accordingly to the SWAP selection, into the first output DINT variable, the third and fourth elements into the second output variable and so on. This function block has been designed to work in conjunction with MB_MST_RD_WORD function block, when values coming from the slave device are formatted as DINT data type.

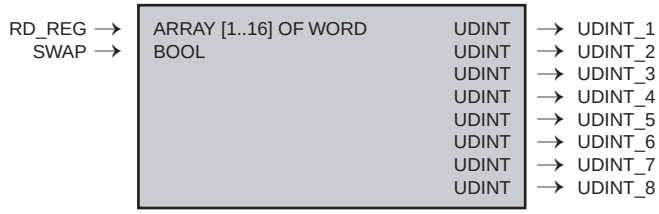
Default Values Table

Input	Default Value
SWAP	FALSE

5-3-11 MODBUS MASTER CONVERSION TO 8 UDINT VALUES

FB Prototype

MB_MST_RD8_UDINT



Input parameters

Label	Type	Description	Range
RD_REG	ARRAY [1..16] OF WORD	Array including each single word to be unpacked	
SWAP	BOOL	Swap activation between Low and High word while converting data [bit]	FALSE = standard, TRUE = swapped

Output parameters

Label	Type	Description	Range
UDINT_1	UDINT	1st converted variable [num]	0... 4294967295
UDINT_2	UDINT	2nd converted variable [num]	0... 4294967295
UDINT_3	UDINT	3rd converted variable [num]	0... 4294967295
UDINT_4	UDINT	4th converted variable [num]	0... 4294967295
UDINT_5	UDINT	5th converted variable [num]	0... 4294967295
UDINT_6	UDINT	6th converted variable [num]	0... 4294967295
UDINT_7	UDINT	7th converted variable [num]	0... 4294967295
UDINT_8	UDINT	8th converted variable [num]	0... 4294967295

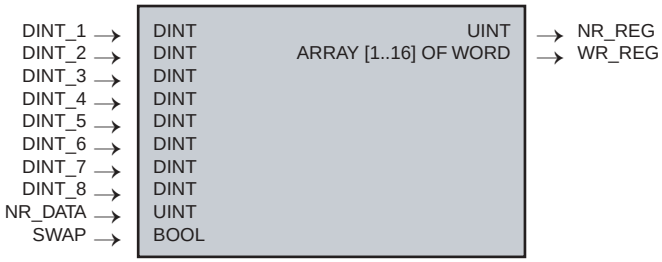
Description

The function block can be used to unpack an array of 16 word elements into 8 single variables as UDINT data type. The first and second elements of the array will be converted, accordingly to the SWAP selection, into the first output UDINT variable, the third and fourth elements into the second output variable and so on. This function block has been designed to work in conjunction with MB_MST_RD_WORD function block, when values coming from the slave device are formatted as UDINT data type.

5-3-12 MODBUS MASTER CONVERSION FROM 8 DINT VALUES

FB Prototype

MB_MST_WR8_DINT



Input parameters

Label	Type	Description	Range
DINT_1	DINT	1st variable to be converted [num]	-2147483648... 2147483647
DINT_2	DINT	2nd variable to be converted [num]	-2147483648... 2147483647
DINT_3	DINT	3rd variable to be converted [num]	-2147483648... 2147483647
DINT_4	DINT	4th variable to be converted [num]	-2147483648... 2147483647
DINT_5	DINT	5th variable to be converted [num]	-2147483648... 2147483647
DINT_6	DINT	6th variable to be converted [num]	-2147483648... 2147483647
DINT_7	DINT	7th variable to be converted [num]	-2147483648... 2147483647
DINT_8	DINT	8th variable to be converted [num]	-2147483648... 2147483647
NR_DATA	UINT	Number of input data to be converted [num]	0... 8
SWAP	BOOL	Swap activation between Low and High word while converting data [bit]	FALSE = standard, TRUE = swapped

Output parameters

Label	Type	Description
NR_REG	UINT	Number of WORD used by the converted variables
WR_REG	ARRAY [1..16] OF WORD	Array including the converted variables

Description

The function block can be used to convert and pack up to 8 DINT data type variables into an array of 16 word elements. The first DINT variable will be converted, accordingly to the SWAP selection, into the first and second element of the output array, the second DINT variable into third and fourth element and so on. This function block has been designed to better work in conjunction with MB_MST_WR_WORD function block, when the values to be written into the slave device are formatted as DINT data type.

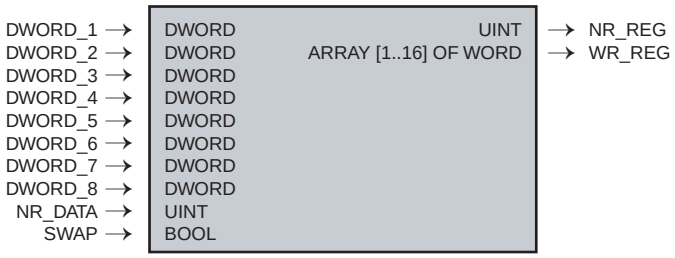
Default Values Table

Input	Default Value
NR_DATA	0
SWAP	FALSE

5-3-13 MODBUS MASTER CONVERSION FROM 8 DWORD VALUES

FB Prototype

MB_MST_WR8_DWORD



Input parameters

Label	Type	Description	Range
DWORD_1	DWORD	1st converted variable [num]	0... 4294967295
DWORD_2	DWORD	2nd converted variable [num]	0... 4294967295
DWORD_3	DWORD	3rd converted variable [num]	0... 4294967295
DWORD_4	DWORD	4th converted variable [num]	0... 4294967295
DWORD_5	DWORD	5th converted variable [num]	0... 4294967295
DWORD_6	DWORD	6th converted variable [num]	0... 4294967295
DWORD_7	DWORD	7th converted variable [num]	0... 4294967295
DWORD_8	DWORD	8th converted variable [num]	0... 4294967295
NR_DATA	DWORD	Number of input data to be converted [num]	0... 8
SWAP	BOOL	Swap activation between Low and High word while converting data [bit]	FALSE = standard, TRUE = swapped

Output parameters

Label	Type	Description
NR_REG	UINT	Number of WORD used by the converted variables
WR_REG	ARRAY [1..16] OF WORD	Array including the converted variables

Description

The function block can be used to convert and pack up to 8 DWORD data type variables into an array of 16 word elements. The first DWORD variable will be converted, accordingly to the SWAP selection, into the first and second element of the output array, the second DWORD variable into third and fourth element and so on. This function block has been designed to better work in conjunction with MB_MST_WR_WORD function block, when the values to be written into the slave device are formatted as DWORD data type.

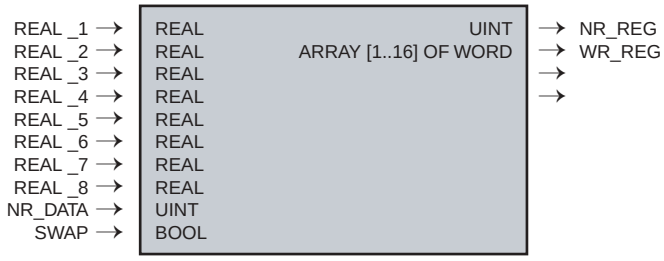
Default Values Table

Input	Default Value
NR_DATA	0
SWAP	FALSE

5-3-14 MODBUS MASTER CONVERSION FROM 8 REAL VALUES

FB Prototype

MB_MST_WR8_REAL



Input parameters

Label	Type	Description	Range
REAL _1	REAL	1st variable to be converted [num]	-3.4E-38... 3.4E+38
REAL _2	REAL	2nd variable to be converted [num]	-3.4E-38... 3.4E+38
REAL _3	REAL	1st variable to be converted [num]	-3.4E-38... 3.4E+38
REAL _4	REAL	2nd variable to be converted [num]	-3.4E-38... 3.4E+38
REAL _5	REAL	3rd variable to be converted [num]	-3.4E-38... 3.4E+38
REAL _6	REAL	4th variable to be converted [num]	-3.4E-38... 3.4E+38
REAL _7	REAL	5th variable to be converted [num]	-3.4E-38... 3.4E+38
REAL _8	REAL	6th variable to be converted [num]	-3.4E-38... 3.4E+38
NR_DATA	REAL	7th variable to be converted [num]	-3.4E-38... 3.4E+38
SWAP	BOOL	8th variable to be converted [num]	-3.4E-38... 3.4E+38

Output parameters

Label	Type	Description
NR_REG	UINT	Number of WORD used by the converted variables
WR_REG	ARRAY [1..16] OF WORD	Array including the converted variables

Description

The function block can be used to convert and pack up to 8 REAL data type variables into an array of 16 word elements. The first REAL variable will be converted, accordigly to the SWAP selection, into the first and second element of the output array, the second REAL variable into third and fourth element and so on. This function block has been designed to better work in conjunction with MB_MST_WR_WORD function block, when the values to be written into the slave device are formatted as REAL data type.

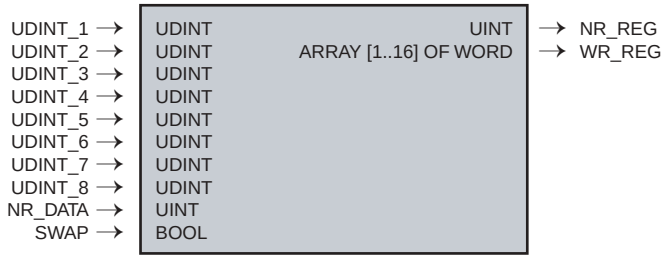
Default Values Table

Input	Default Value
NR_DATA	0
SWAP	FALSE

5-3-15 MODBUS MASTER CONVERSION FROM 8 UDINT VALUES

FB Prototype

MB_MST_WR8_UDINT



Input parameters

Label	Type	Description	Range
UDINT_1	UDINT	1st variable to be converted [num]	0... 4294967295
UDINT_2	UDINT	2nd variable to be converted [num]	0... 4294967295
UDINT_3	UDINT	3rd variable to be converted [num]	0... 4294967295
UDINT_4	UDINT	4th variable to be converted [num]	0... 4294967295
UDINT_5	UDINT	5th variable to be converted [num]	0... 4294967295
UDINT_6	UDINT	6th variable to be converted [num]	0... 4294967295
UDINT_7	UDINT	7th variable to be converted [num]	0... 4294967295
UDINT_8	UDINT	8th variable to be converted [num]	0... 4294967295
NR_DATA	UUINT	Number of input data to be converted [num]	0... 8
SWAP	BOOL	Swap activation between Low and High word while converting data [bit]	FALSE = standard, TRUE = swapped)

Output parameters

Output	Type	Description
NR_REG	UINT	Number of WORD used by the converted variables
WR_REG	ARRAY [1..16] OF WORD	Array including the converted variables

Description

The function block can be used to convert and pack up to 8 UDINT data type variables into an array of 16 word elements. The first UDINT variable will be converted, accordingly to the SWAP selection, into the first and second element of the output array, the second UDINT variable into third and fourth element and so on. This function block has been designed to better work in conjunction with MB_MST_WR_WORD function block, when the values to be written into the slave device are formatted as UDINT data type.

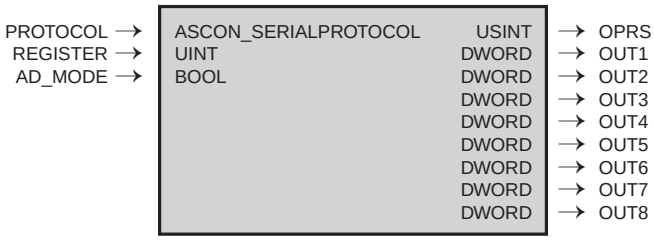
Default Values Table

Input	Default Value
NR_DATA	0
SWAP	FALSE

5-3-16 MB_SLV_RD8_DWORD

FB Prototype

MB_SLV_RD8_DWORD



Input parameters

Label	Type	Description
PROTOCOL	ASCON_SERIALPROTOCOL	Reference Protocol
REGISTER	UINT	Register
AD_MODE	BOOL	Addressing Mode Setting (default value = TRUE)

Output parameters

Label	Type	Description
OPRS	USINT	Function block result
OUT1 – OUT8	DWORD	Data

Description

This Function Block reads 8 DWORD values from the memory area starting from the passed register (*REGISTER*). The module returns the *OPRS* parameter that is used to detect if the operation is performed correctly, and the value of the registers (*OUT1 – OUT8*). The setting *AD_MODE* allows the user to select if the *REGISTER* input base is 0 (*AD_MODE = FALSE*) or 1 (*AD_MODE* = TRUE). The default value for *AD_MODE* is TRUE.

Reference tables

Here are reported the range of the parameters used by the function block.

Protocol

Protocol	Description
SLAVE1	Protocol Modbus Slave 1
SLAVE2	Protocol Modbus Slave 2

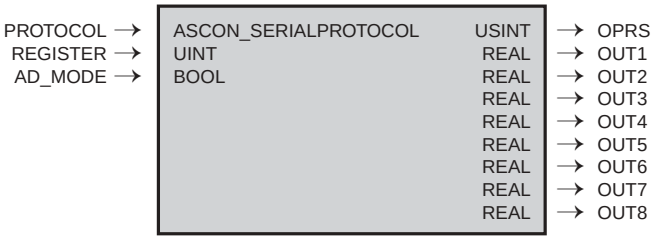
OPRS

OPRS	Description
0	Service executed correctly
1	Address not allowed
3	Protocol number not valid

5-3-17 MB_SLV_RD8_REAL

FB Prototype

MB_SLV_RD8_REAL



Input parameters

Label	Type	Description
PROTOCOL	ASCON_SERIALPROTOCOL	Reference Protocol
REGISTER	UINT	Register
AD_MODE	BOOL	Addressing Mode Setting (default value = TRUE)

Output parameters

Label	Type	Description
OPRS	USINT	Function block result
OUT1 – OUT8	REAL	Data

Description

This Function Block reads 8 REALvalues from the memory area starting from the passed register (*REGISTER*). The module returns the *OPRS* parameter that is used to detect if the operation is performed correctly, and the value of the registers (*OUT1 – OUT8*). The setting *AD_MODE* allows the user to select if the *REGISTER* input base is 0 (*AD_MODE = FALSE*) or 1 (*AD_MODE* = TRUE). The default value for *AD_MODE* is TRUE.

Reference tables

Here are reported the range of the parameters used by the function block.

Protocol

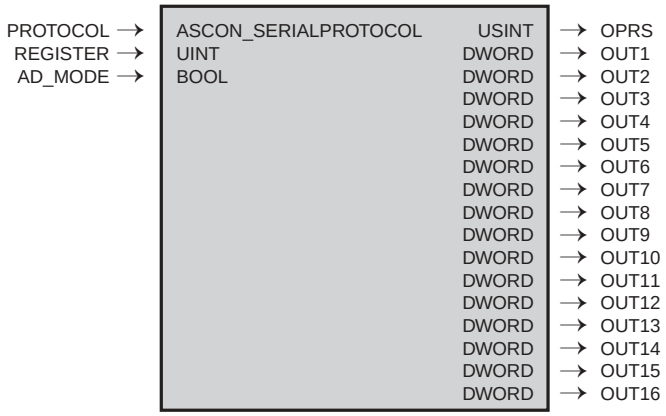
Protocol	Description
SLAVE1	Protocol Modbus Slave 1
SLAVE2	Protocol Modbus Slave 2

OPRS

OPRS	Description
0	Service executed correctly
1	Address not allowed
3	Protocol number not valid

5-3-18 MB_SLV_RD16_WORD

FB Prototype



Input parameters

Label	Type	Description
PROTOCOL	ASCON_SERIALPROTOCOL	Reference Protocol
REGISTER	UINT	Register
AD_MODE	BOOL	Addressing Mode Setting (default value = TRUE)

Output parameters

Label	Type	Description
OPRS	USINT	Function block result
OUT1 – OUT16	WORD	Data

Description

This Function Block reads 16 WORD values from the memory area starting from the passed register (*REGISTER*). The module returns the *OPRS* parameter that is used to detect if the operation is performed correctly, and the value of the registers (*OUT1 – OUT16*). The setting *AD_MODE* allows the user to select if the *REGISTER* input base is 0 (*AD_MODE = FALSE*) or 1 (*AD_MODE* = TRUE). The default value for *AD_MODE* is TRUE.

Reference tables

Here are reported the range of the parameters used by the function block.

Protocol

Protocol	Description
SLAVE1	Protocol Modbus Slave 1
SLAVE2	Protocol Modbus Slave 2

OPRS

OPRS	Description
0	Service executed correctly
1	Address not allowed
3	Protocol number not valid

5-3-19 MB_SLV_RD32_DIGITAL

FB Prototype

MB_SLV_RD32_DIGITAL

Input parameters

Input	Type	Description
PROTOCOL	ASCON_SERIALPROTOCOL	Reference Protocol
REGISTER	UINT	Register
AD_MODE	BOOL	Addressing Mode Setting (default value = TRUE)

Output parameters

Output	Type	Description
OPRS	BOOL	Function block result
DATA	DWORD	Data

Description

This Function Block reads 32 digital values from the memory area starting from the passed register (*REGISTER*). The module returns the *OPRS* parameter that is used to detect if the operation is performed correctly, and the value of the digitals (*DATA*) using the DWORD format. The setting *AD_MODE* allows the user to select if the *REGISTER* input base is 0 (*AD_MODE = FALSE*) or 1 (*AD_MODE* = TRUE). The default value for *AD_MODE* is TRUE.

Reference Tables

Here are reported the range of the parameters used by the function block.

Protocol

Protocol	Description
SLAVE1	Protocol Modbus Slave 1
SLAVE2	Protocol Modbus Slave 2

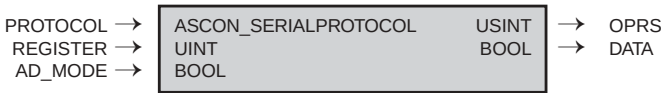
OPRS

OPRS	Description
0	Service executed correctly
1	Address not allowed
3	Protocol number not valid

5-3-20 MB_SLV_RD_DIGITAL

FB Prototype

MB_SLV_RD_DIGITAL



Input parameters

Label	Type	Description
PROTOCOL	ASCON_SERIALPROTOCOL	Reference Protocol
REGISTER	UINT	Register
AD_MODE	BOOL	Addressing Mode Setting (default value = TRUE)

Output parameters

Label	Type	Description
OPRS	BOOL	Function block result
DATA	BOOL	Data

Description

This Function Block reads a digital value from the memory area corresponding to the passed register (*REGISTER*). The module returns the *OPRS* parameter that is used to detect if the operation is performed correctly, and the value of the register (*DATA*). The setting *AD_MODE* allows the user to select if the *REGISTER* input base is 0 (*AD_MODE = FALSE*) or 1 (*AD_MODE* = TRUE). The default value for *AD_MODE* is TRUE.

Reference tables

Here are reported the range of the parameters used by the function block.

Protocol

Protocol	Description
SLAVE1	Protocol Modbus Slave 1
SLAVE2	Protocol Modbus Slave 2

OPRS

OPRS	Description
0	Service executed correctly
1	Address not allowed
3	Protocol number not valid

5-3-21 MB_SLV_RD_DWORD

FB Prototype

MB_SLV_RD_DWORD

Input parameters

Label	Type	Description
PROTOCOL	ASCON_SERIALPROTOCOL	Reference Protocol
REGISTER	UINT	Register
AD_MODE	BOOL	Addressing Mode Setting (default value = TRUE)

Output parameters

Label	Type	Description
OPRS	USINT	Function block result
DATA	DWORD	Data

Description

This Function Block reads a DWORD value from the memory area corresponding to the passed register (*REGISTER*). The module returns the *OPRS* parameter that is used to detect if the operation is performed correctly, and the value of the register (*DATA*). The setting *AD_MODE* allows the user to select if the *REGISTER* input base is 0 (*AD_MODE = FALSE*) or 1 (*AD_MODE* = TRUE). The default value for *AD_MODE* is TRUE.

Reference tables

Here are reported the range of the parameters used by the function block.

Protocol

Protocol	Description
SLAVE1	Protocol Modbus Slave 1
SLAVE2	Protocol Modbus Slave 2

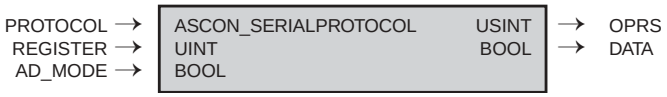
OPRS

OPRS	Description
0	Service executed correctly
1	Address not allowed
3	Protocol number not valid

5-3-22 MB_SLV_RD_REAL

FB Prototype

MB_SLV_RD_REAL



Input parameters

Label	Type	Description
PROTOCOL	ASCON_SERIALPROTOCOL	Reference Protocol
REGISTER	UINT	Register
AD_MODE	BOOL	Addressing Mode Setting (default value = TRUE)

Output parameters

Label	Type	Description
OPRS	BOOL	Function block result
DATA	REAL	Data

Description

This Function Block reads a REAL value from the memory area corresponding to the passed register (*REGISTER*). The module returns the *OPRS* parameter that is used to detect if the operation is performed correctly, and the value of the register (*DATA*). The setting *AD_MODE* allows the user to select if the *REGISTER* input base is 0 (*AD_MODE = FALSE*) or 1 (*AD_MODE* = TRUE). The default value for *AD_MODE* is TRUE.

Reference tables

Here are reported the range of the parameters used by the function block.

Protocol

Protocol	Description
SLAVE1	Protocol Modbus Slave 1
SLAVE2	Protocol Modbus Slave 2

OPRS

OPRS	Description
0	Service executed correctly
1	Address not allowed
3	Protocol number not valid

5-3-23 MB_SLV_RD_WORD

FB Prototype

MB_SLV_RD_WORD

Input parameters

Label	Type	Description
PROTOCOL	ASCON_SERIALPROTOCOL	Reference Protocol
REGISTER	UINT	Register
AD_MODE	BOOL	Addressing Mode Setting (default value = TRUE)

Output parameters

Label	Type	Description
OPRS	USINT	Function block result
DATA	WORD	Data

Description

This Function Block reads a WORD value from the memory area corresponding to the passed register (*REGISTER*). The module returns the *OPRS* parameter that is used to detect if the operation is performed correctly, and the value of the register (*DATA*). The setting *AD_MODE* allows the user to select if the *REGISTER* input base is 0 (*AD_MODE = FALSE*) or 1 (*AD_MODE* = TRUE). The default value for *AD_MODE* is TRUE.

Reference tables

Here are reported the range of the parameters used by the function block.

Protocol

Protocol	Description
SLAVE1	Protocol Modbus Slave 1
SLAVE2	Protocol Modbus Slave 2

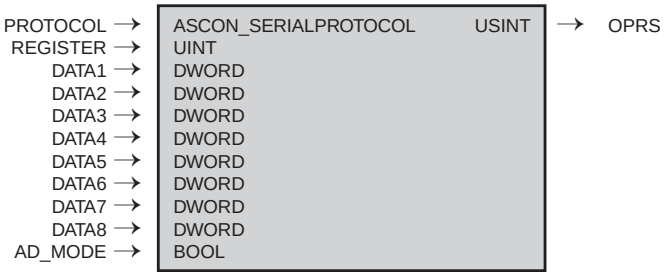
OPRS

OPRS	Description
0	Service executed correctly
1	Address not allowed
3	Protocol number not valid

5-3-24 MB_SLV_WR8_DWORD

FB Prototype

MB_SLV_WR8_DWORD



Input parameters

Label	Type	Description
PROTOCOL	ASCON_SERIALPROTOCOL	Reference Protocol
REGISTER	UINT	Register
DATA1 – DATA8	DWORD	Data
AD_MODE	BOOL	Addressing Mode Setting (default value = TRUE)

Output parameters

Label	Type	Description
OPRS	USINT	Function block result

Description

This Function Block writes 8 DWORD values passed with *DATA1 – DATA8* parameters to the memory area starting from the passed register (*REGISTER*). The output of the module (*OPRS*) is used to detect if the operation is performed correctly. The setting *AD_MODE* allows the user to select if the *REGISTER* input base is 0 (*AD_MODE = FALSE*) or 1 (*AD_MODE* = TRUE). The default value for *AD_MODE* is TRUE.

Reference tables

Here are reported the range of the parameters used by the function block.

Protocol

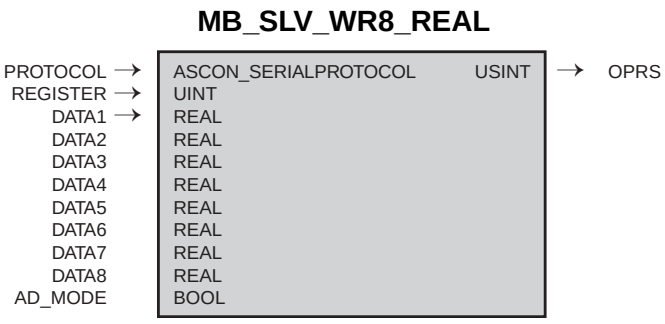
Protocol	Description
SLAVE1	Protocol Modbus Slave 1
SLAVE2	Protocol Modbus Slave 2

OPRS

OPRS	Description
0	Service executed correctly
1	Address not allowed
3	Protocol number not valid

5-3-25 MB_SLV_WR8_REAL

FB Prototype



Input parameters

Label	Type	Description
PROTOCOL	ASCON_SERIALPROTOCOL	Reference Protocol
REGISTER	UINT	Register
DATA1 – DATA8	REAL	Data
AD_MODE	BOOL	Addressing Mode Setting (default value = TRUE)

Output parameters

Label	Type	Description
OPRS	USINT	Function block result

Description

This Function Block writes 8 REAL values passed with *DATA1 – DATA8* parameters to the memory area starting from the passed register (*REGISTER*). The output of the module (*OPRS*) is used to detect if the operation is performed correctly.
The setting *AD_MODE* allows the user to select if the *REGISTER* input base is 0 (*AD_MODE = FALSE*) or 1 (*AD_MODE* = TRUE). The default value for *AD_MODE* is TRUE.

Reference tables

Here are reported the range of the parameters used by the function block.

Protocol

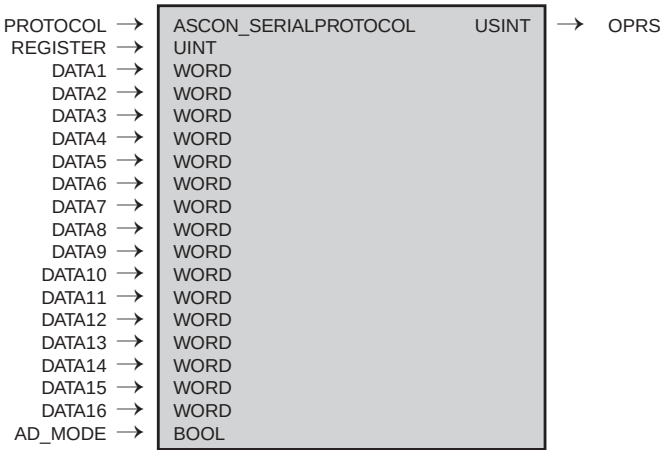
Protocol	Description
SLAVE1	Protocol Modbus Slave 1
SLAVE2	Protocol Modbus Slave 2

OPRS

OPRS	Description
0	Service executed correctly
1	Address not allowed
3	Protocol number not valid

5-3-26 MB_SLV_WR16_WORD

FB Prototype



Input parameters

Label	Type	Description
PROTOCOL	ASCON_SERIALPROTOCOL	Reference Protocol
REGISTER	UINT	Register
DATA1 – DATA16	WORD	Data
AD_MODE	BOOL	Addressing Mode Setting (default value = TRUE)

Output parameters

Label	Type	Description
OPRS	USINT	Function block result

Description

This Function Block writes 16 WORD values passed with *DATA1 – DATA16* parameters to the memory area starting from the passed register (*REGISTER*). The output of the module (*OPRS*) is used to detect if the operation is performed correctly.
The setting *AD_MODE* allows the user to select if the *REGISTER* input base is 0 (*AD_MODE = FALSE*) or 1 (*AD_MODE* = TRUE). The default value for *AD_MODE* is TRUE.

Reference tables

Here are reported the range of the parameters used by the function block.

Protocol

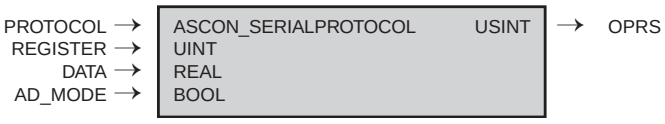
Protocol	Description
SLAVE1	Protocol Modbus Slave 1
SLAVE2	Protocol Modbus Slave 2

OPRS

OPRS	Description
0	Service executed correctly
1	Address not allowed
3	Protocol number not valid

5-3-27 MB_SLV_WR32_DIGITAL

FB Prototype



Input parameters

Label	Type	Description
PROTOCOL	ASCON_SERIALPROTOCOL	Reference Protocol
REGISTER	UINT	Register
DATA	DWORD	Data
AD_MODE	BOOL	Addressing Mode Setting (default value = TRUE)

Output parameters

Label	Type	Description
OPRS	BOOL	Function block result

Description

This Function Block writes 32 digital values passed with *DATA* parameter to the memory area starting from the passed register (*REGISTER*). The output of the module (*OPRS*) is used to detect if the operation is performed correctly.
The setting *AD_MODE* allows the user to select if the *REGISTER* input base is 0 (*AD_MODE = FALSE*) or 1 (*AD_MODE* = TRUE). The default value for *AD_MODE* is TRUE.

Reference tables

Here are reported the range of the parameters used by the function block.

Protocol

Protocol	Description
SLAVE1	Protocol Modbus Slave 1
SLAVE2	Protocol Modbus Slave 2

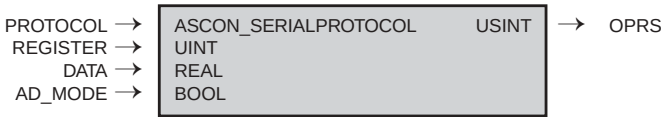
OPRS

OPRS	Description
0	Service executed correctly
1	Address not allowed
3	Protocol number not valid

5-3-28 MB_SLV_WR_DIGITAL

FB Prototype

MB_SLV_WR_DIGITAL



Input parameters

Label	Type	Description
PROTOCOL	ASCON_SERIALPROTOCOL	Reference Protocol
REGISTER	UINT	Register
DATA	BOOL	Data
AD_MODE	BOOL	Addressing Mode Setting default value = TRUE)

Output parameters

Label	Type	Description
OPRS	BOOL	Function block result

Description

This Function Block writes the digital value passed with DATA parameter to the memory area corresponding to the passed register (REGISTER). The output of the module (OPRS) is used to detect if the operation is performed correctly.
The setting AD_MODE allows the user to select if the REGISTER input base is 0 (AD_MODE = FALSE) or 1 (AD_MODE = TRUE). The default value for AD_MODE is TRUE.

Reference tables

Here are reported the range of the parameters used by the function block.

Protocol

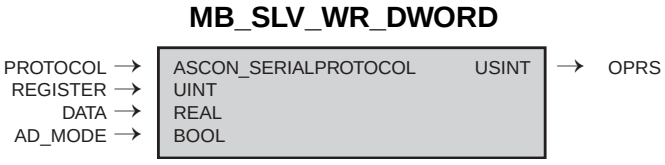
Protocol	Description
SLAVE1	Protocol Modbus Slave 1
SLAVE2	Protocol Modbus Slave 2

OPRS

OPRS	Description
0	Service executed correctly
1	Address not allowed
3	Protocol number not valid

5-3-29 MB_SLV_WR_DWORD

FB Prototype



Input parameters

Label	Type	Description
PROTOCOL	ASCON_SERIALPROTOCOL	Reference Protocol
REGISTER	UINT	Register
DATA	DWORD	Data
AD_MODE	BOOL	Addressing Mode Setting (default value = TRUE)

Output parameters

Label	Type	Description
OPRS	BOOL	Function block result

Description

This Function Block writes the DWORD value passed with *DATA* parameter to the memory area corresponding to the passed register (*REGISTER*). The output of the module (*OPRS*) is used to detect if the operation is performed correctly.
The setting *AD_MODE* allows the user to select if the *REGISTER* input base is 0 (*AD_MODE = FALSE*) or 1 (*AD_MODE* = TRUE). The default value for *AD_MODE* is TRUE.

Reference tables

Here are reported the range of the parameters used by the function block.

Protocol

Protocol	Description
SLAVE1	Protocol Modbus Slave 1
SLAVE2	Protocol Modbus Slave 2

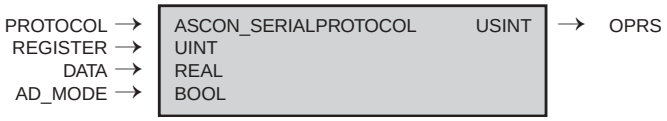
OPRS

OPRS	Description
0	Service executed correctly
1	Address not allowed
3	Protocol number not valid

5-3-30 MB_SLV_WR_REAL

FB Prototype

MB_SLV_WR_REAL



Input parameters

Label	Type	Description
PROTOCOL	ASCON_SERIALPROTOCOL	Reference Protocol
REGISTER	UINT	Register
DATA	BOOL	Data
AD_MODE	BOOL	Addressing Mode Setting (default value = TRUE)

Output parameters

Label	Type	Description
OPRS	BOOL	Function block result

Description

This Function Block writes the REAL value passed with DATA parameter to the memory area corresponding to the passed register (REGISTER). The output of the module (OPRS) is used to detect if the operation is performed correctly.
The setting AD_MODE allows the user to select if the REGISTER input base is 0 (AD_MODE = FALSE) or 1 (AD_MODE = TRUE). The default value for AD_MODE is TRUE.

Reference tables

Here are reported the range of the parameters used by the function block.

Protocol

Protocol	Description
SLAVE1	Protocol Modbus Slave 1
SLAVE2	Protocol Modbus Slave 2

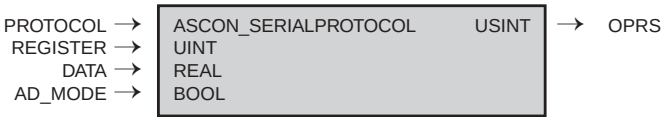
OPRS

OPRS	Description
0	Service executed correctly
1	Address not allowed
3	Protocol number not valid

5-3-31 MB_SLV_WR_WORD

FB Prototype

MB_SLV_WR_WORD



Input parameters

Label	Type	Description
PROTOCOL	ASCON_SERIALPROTOCOL	Reference Protocol
REGISTER	UINT	Register
DATA	WORD	Data
AD_MODE	BOOL	Addressing Mode Setting (default value = TRUE)

Output parameters

Label	Type	Description
OPRS	BOOL	Function block result

Description

This Function Block writes the WORD value passed with DATA parameter to the memory area corresponding to the passed register (REGISTER). The output of the module (OPRS) is used to detect if the operation is performed correctly.
The setting AD_MODE allows the user to select if the REGISTER input base is 0 (AD_MODE = FALSE) or 1 (AD_MODE = TRUE). The default value for AD_MODE is TRUE.

Reference tables

Here are reported the range of the parameters used by the function block.

Protocol

Protocol	Description
SLAVE1	Protocol Modbus Slave 1
SLAVE2	Protocol Modbus Slave 2

OPRS

OPRS	Description
0	Service executed correctly
1	Address not allowed
3	Protocol number not valid

5-3-32 MODEM_CHECK

FB Prototype

MODEM_CHECK

Input description

Label	Type	Description
CHK_MODEM	BOOL	Command to perform the modem operational status, **evaluated on the rising edge ONLY!!!**

Output description

Label	Type	Description	Range
CONFIRM	BOOL	Function block operational status	
ERROR	BOOL	Error status	
ERR_CODE	DWORD	Error code. [bit mask]	16#00 00 00 00… 16#FF FF FF FF
SIGN_PWR	INT	Modem received signal strength indication (rssi) [num]	-1... 99
SIGN_ERR	INT	Modem channel bit error (ber) [num]	-1… 99

Description

This function block performs an operation verification on the modem connected to the RS-232 ASCII configured port. It returns the signal strength of the GSM connection accordingly to the standard defined by the AT commands for GSM/ GPRS wireless modems. The function block has been successfully tested with the Ascon Tecnologic APS2MODEMEGxxx modem. It returns also detailed information on the possible errors. The function block should be used in conjunction with the MODEM_CONF in order to have a complete set of operations before to proceed with the normal modem data activities.

Note: For parameter description and meaning, please search for a good reference document about AT commands by searching the “AT commands for GSM/GPRS wireless modem” manual from MultiTech Systems.

Default Values Table

Input	Default Value
CHK_MODEM	FALSE

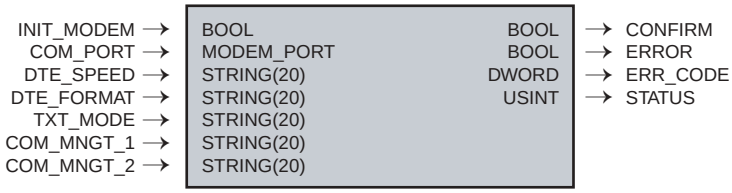
Reference Table

Output	Description
ERR_CODE.0	Modem not ready or not available
ERR_CODE.1	No bytes available into the buffer
ERR_CODE.2	Modem signal error
ERR_CODE.3	Port not configured for the ASCII serial protocol
ERR_CODE.4	Peripheral driver problems
ERR_CODE.5	Unavailable number of requested bytes in the input buffer
ERR_CODE.6	Number of bytes present in the input buffer less then requested

5-3-33 MODEM_CONF

FB Prototype

MODEM_CONF



Input description

Label	Type	Description	Range
INIT_MODEM	BOOL	Command to perform the modem initialisation, **evaluated on the rising edge ONLY!!!**	
DTE_SPEED	STRING(20)	DTE speed configuration string [chars]	up to 20 characters from all the supported AT commands for GSM/GPRS wireless modem standards
DTE_FORMAT	STRING(20)	Command to perform the modem initialisation, **evaluated on the rising edge ONLY!!!**	
TXT_MODE	STRING(20)	DTE speed configuration string [chars]	up to 20 characters from all the supported AT commands for GSM/GPRS wireless modem standards
COM_MNGT_1	STRING(20)	DTE data format configuration string [chars]	up to 20 characters from all the supported AT commands for GSM/GPRS wireless modem standards
COM_MNGT_2	STRING(20)	SMS text mode configuration string [chars]	up to 20 characters from all the supported AT commands for GSM/GPRS wireless modem standards

Output description

Output	Type	Description	Range
CONFIRM	BOOL	Function block operational status	
ERROR	BOOL	Error status	
ERR_CODE	DWORD	Error code [bit mask]	16#00 00 00 00… 16#FF FF FF FF
STATUS	USINT	Modem initialisation phase status value [num]	0… 8

Description

This function block performs the configuration and initialisation of the modem connected to the RS-232 ASCII configured port. It returns a TRUE value on the CONFIRM output if the procedure successfully reaches the end. The function block and default values of the parameters has been successfully tested with the

Ascon Tecnologic APS2MODEMEGxxx modem. It returns also detailed information on the possible errors.

Note: For parameter description and meaning, please search for a good reference document about AT commands by searching the "**AT commands for GSM/GPRS wireless modem**" manual from MultiTech Systems.

Default Values Table

Input	Default Value
INIT_MODEM	FALSE
DTE_SPEED	'AT+IPR=9600'
DTE_FORMAT	'AT+ICF=3'
TXT_MODE	'AT+CMGF=1'
COM_MNGT_1	'AT&C1'
COM_MNGT_2	'AT&D0'

Output	Default value
STATUS	0 = Waiting to start 1 = set DTE speed 2 = set DTE data format 3 = set SMS text mode 4 = set DCD managing mode 5 = set DTR managing mode 6 = ECHO disabling, enabling extended answers and save conf. 7 = waiting for modem confirm 8 = OK, done

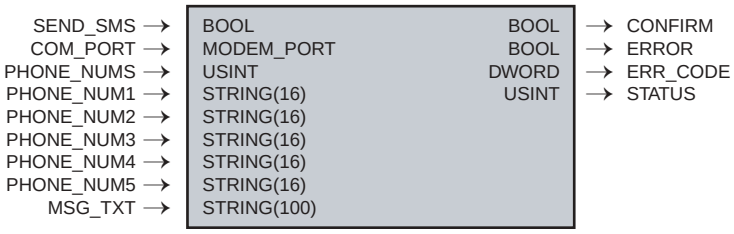
Reference Table

Output	Description
ERR_CODE.0	Port not configured for the ASCII serial protocol
ERR_CODE.1	Attempt to write a number of bytes that exceeds the allowed limit
ERR_CODE.2	Port not configured for the ASCII serial protocol
ERR_CODE.3	Peripheral driver problems
ERR_CODE.4	Unavailable number of requested bytes in the input buffer
ERR_CODE.5	Number of bytes present in the input buffer less then requested

5-3-34 MODEM_SMS_SEND

FB Prototype

MODEM_SMS_SEND



Input description

Label	Type	Description	
SEND_SMS	BOOL	Command to start the procedure to send the SMS message, **evaluated on the rising edge ONLY!!!**	
PHONE_NUMS	USINT	Amount of phone numbers which to send the SMS message [num]	1… 5
PHONE_NUM1	STRING(16)	1st phone number which to send the SMS message [chars]	+, 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 0
PHONE_NUM2	STRING(16)	2nd phone number which to send the SMS message [chars]	+, 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 0
PHONE_NUM3	STRING(16)	3rd phone number which to send the SMS message [chars]	+, 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 0
PHONE_NUM4	STRING(16)	4th phone number which to send the SMS message [chars]	+, 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 0
PHONE_NUM5	STRING(16)	5th phone number which to send the SMS message [chars]	+, 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 0
MSG_TXT	STRING(100)	Text message body to be sent [chars]	up to 100 characters from all the supported AT commands for GSM/GPRS wireless modem standards

Output description

Label	Type	Description	
CONFIRM	BOOL	Function block operational status	
ERROR	BOOL	Error status	
ERR_CODE	DWORD	Error code [bit mask]	16#00 00 00 00… 16#FF FF FF FF)
STATUS	USINT	Sequence of operation status value [num]	0… 6

Description

This function block performs the operations needed to deliver an SMS (Short Message Service) message up to 5 valid phone numbers trough the modem connected the RS-232 ASCII configured port. It returns a TRUE value on the CONFIRM output if the procedure successfully reaches the end. The function block

and default values of the parameters has been successfully tested with the Ascon Tecnologic APS2MODEMEGxxx modem. It returns also detailed information on the possible errors. The function block should be used in conjunction with the MODEM_CONF and MODEM_CHECK in order to have a complete set of operations before to proceed with the normal modem data activities.

Note: For parameter description and meaning, please search for a good reference document about AT commands by searching the "**AT commands for GSM/GPRS wireless modem**" manual from MultiTech Systems.

Default Values Table

Input	Default Value
SEND_SMS	FALSE
PHONE_NUMS	1
PHONE_NUM1	'+0123456789'
PHONE_NUM2	''
PHONE_NUM3	''
PHONE_NUM4	''
PHONE_NUM5	''
MSG_TXT	'HELLO WORLD!!!'

Output	Default Value
STATUS	0 = Waiting to start 1 = phone numbers handling 2 = configuring phone number 3 = formatting SMS message 4 = delivering SMS message 5 = waiting to complete SMS delivering 6 = done, waiting for next operation

Reference Table

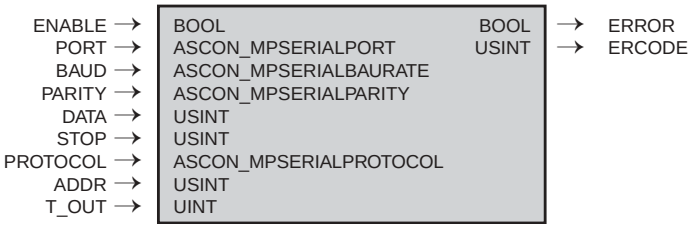
Output	Description
ERR_CODE.0	Port not configured for the ASCII serial protocol
ERR_CODE.1	Attempt to write a number of bytes that exceeds the allowed limit
ERR_CODE.2	Port not configured for the ASCII serial protocol
ERR_CODE.3	Peripheral driver problems
ERR_CODE.4	Unavailable number of requested bytes in the input buffer
ERR_CODE.5	Number of bytes present in the input buffer less then requested
ERR_CODE.6	Amount of phone numbers lower than admitted
ERR_CODE.7	Amount of phone numbers higher than admitted

5-3-35 MP SERIAL Ports

Caution
This Function Block can be used **only** with the **MP-01** unit.

FB Prototype

MP_SERIAL_PORTS



Input parameters

Label	Type	Description
ENABLE	BOOL	Command to enable the serial port operations
PORT	ASCON_MPSERIALPORT	Required serial port (default value = X1)
BAUD	ASCON_SERIALBAUDRATE	Communication baud rate (default value = B9600)
PARITY	ASCON_SERIALPARITY	Communication parity (default value = NOPAR)
DATA	USINT	Communication data length (default value = 8)
STOP	USINT	Communication stop bits (default value = 1)
PROTOCOL	ASCON_SERIALPROTOCOL	Required communication agent (default value = SLAVE1)
ADDR	USINT	Slave agent communication address (default value = 1)
T_OUT	UINT	Communication time out factor (this value is a multiple of 100ms - eg. 10 means 10 x 100ms = 1 s)

Output parameters

Label	Type	Description
ERROR	BOOL	Serial port communication error status
ERCODE	USINT	Serial port communication error code

Description

This function block configure the required Serial Port and the parameters to be used with the selected Modbus Agent.

Reference tables

Here are reported the range of the parameters used by the function block.

Input

Label	Options
ENABLE	FALSE = Serial Port DISABLE TRUE = Serial Port ENABLE
PORT	X0 = Communication Port X0 RS232/RS485 X1 = Communication Port X1 RS485 (default value)

Label	Options
PROTOCOL	ASCII = Serial ASCII interface SLAVE1 = Modbus SLAVE_1 Agent (default value) SLAVE2 = Modbus SLAVE_2 Agent MASTER = Modbus MASTER Agent
BAUD	B2400 = 2.4 kbps B4800 = 4.8 kbps B9600 = 9.6 kbps (default value) B19200 = 19.2 kbps B38400 = 38.4 kbps B57600 = 57.6 kbps B115200 = 115.2 kbps
PARITY	NOPAR = NONE (default value) EVEN = EVEN ODD = ODD
STOP	1 or 2 (default value = 1)
DATA	7 or 8 (default value = 8)
T_OUT	0... 65535. Deafult value = 10 (1000 ms)

Output

Label	Options
ERCODE	0 = Communication OK 1 = Port not available 2 = Protocol not available 3 = Invalid configuration

5-3-36 PROFIBUS PORT

FB Prototype

PROFIBUS_PORT

Input parameters

Label	Type	Description	Range
ENABLE	BOOL	Command to enable/disable the PROFIBUS DP port operations [bit]	FALSE = disable, TRUE = enable
ADDR	USINT	Profibus DP slave node address, evaluated ONLY on the ENABLE rising edge [num]	0... 255
IN_BYTES	USINT	Amount of bytes used by the Profibus DP Master INPUT telegram, evaluated ONLY on the ENABLE rising edge [bytes]	0... 244)
OUT_BYTES	USINT	Amount of bytes used by the Profibus DP Master OUTPUT telegram, evaluated ONLY on the ENABLE rising edge [bytes]	0... 244
PAGE_MODE	BOOL	Paging mode management, evaluated ONLY on the ENABLE rising edge [bit]	FALSE = disable, TRUE = enable
PAGE_OUT	USINT	Page to be sent to the Profibus DP Master [num]	1... 4

Output parameters

Output	Type	Description	Range
ERROR	BOOL	Error status	
ERR_CODE	USINT	Error code. [bit mask]	16#00 00 00 00 … 16#FF FF FF FF
PAGE_ACT	USINT	Actual page of data in use	

Description

This function block has been designed to setup the parameters needed to establish the communication with Profibus DP Master device. It includes also all the functionalities in order to verify the status of the communication and the data exchange.

Default Values Table

Input	Default Value
ENABLE	FALSE
ADDR	10
IN_BYTES	140
OUT_BYTES	210
PAGE_MODE	FALSE
PAGE_OUT	1

Reference Table

Output	Description
ERR_CODE.0	Initialization Error
ERR_CODE.1	Telegram Size Error
ERR_CODE.2	Hardware Error
ERR_CODE.3	Profibus agent not activated

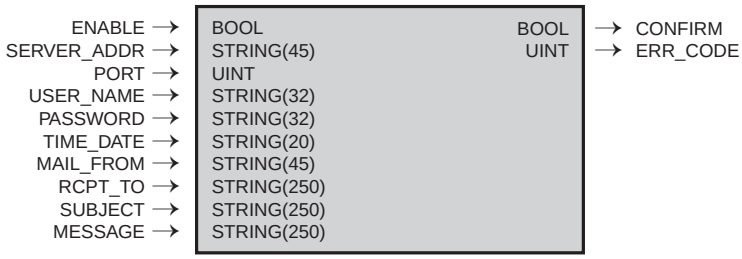
Output	Description
ERR_CODE.4	Protocol Error
ERR_CODE.5	Page Number Error

WARNING

The total amount of bytes exchanged between the CU-02 and the Profibus Master (IN_BYTES + OUT_BYTES) CANNOT EXCEEDS 350!!!

5-3-37 SEND_EMAIL

FB Prototype



Input parameters

Input	Type	Description
ENABLE	BOOL	Command to enable the operations
SERVER_ADDR	STRING	SMTP server address (45)
PORT	UINT	SMTP server connection port
USER_NAME	STRING	User name for the mail account (32)
PASSWORD	STRING	Password for the mail account (32)
TIME_DATE	STRING	Email sending date and time (20)
MAIL_FROM	STRING	Sender name (45)
RCPT_TO	STRING	Receiver addresses (250)
SUBJECT	STRING	Message subject (250)
MESSAGE	STRING	Message text (250)

Output parameters

Output	Type	Description
CONFIRM	BOOL	Confirm of the operation
ERR_CODE	UINT	Error code

Description

This function block implements the functions used by an e-mail client to establish a connection with a SMTP server in order to send an e-mail message.

ENABLE/CONFIRM Management

The communication between an SMTP Client and Server is established on a TCP/IP connection by exchanging some messages in a standardized sequence. The time needed to realize and complete the sequence depends from some factors as the connection speed, the server answering time, the functions supported by the server and the dimension of the inputs. To avoid delays in the PLC program execution cycle time, the function block is designed to execute the sequence of activities cyclically with the FB “calls”. For these reasons it is important to define the different synchronization mechanisms to control:

- when to start sending the message;
- when the sending process ended correctly;
- when a problem occurred during the process;
- when to abort the operations.

The parameters ENABLE and CONFIRM provide the correct functionalities in order to manage these situations. The FB does not execute any operations until the input ENABLE change to TRUE, then the process goes on until the output CONFIRM become TRUE. At this point, it is necessary to call at least one time the FB with the input ENABLE at FALSE in order to reset the sending sequence and to be ready to send a new message. At any time, it is possible to reset the message sending procedure by changing the ENABLE input value to FALSE.

Some practical indications on how to simplify the FB use:

- before to start a new e-mail sending procedure or in case of error, could be useful to execute the FB one time with the ENABLE input at FALSE to be sure the internal FB variables have been reset and ready to restart in the correct way
- while an e-mail sending procedure is active, it is important to verify the CONFIRM and ERR_CODE output. It is possible to assume the e-mail has been sent correctly only if the ERR_CODE output never changed from 0 (zero) during the entire sending process and until the CONFIRM output become TRUE. It happens sometimes the ERR_CODE will be generated at the same time the CONFIRM becomes TRUE

SERVER_ADDR

The SMTP address can be specified as numeric IP address (e.g. 192.168.5.1) or with the specific identification name (e.g. smtp.mail.yahoo.it); from the Server name, using the DNS, the system will find out the server IP address.

USER_NAME and PASSWORD

The FB supports both the 2 authentication modes indicated as “AUTH LOGIN” and “AUTH PLAIN”. Both are based on the string user name and password. The FB deals with the server the supported method and, in case no one of the method is supported, it will try to send anyway the message supposing the server does not require the user authentication.

MAIL_FROM

Sender e-mail address. The string used to specify the e-mail address MUST ACCOMPLISH to the standard e-mail format. Anyway, it is important to remember that the FB is designed only to send e-mail and not to receive so this field is just an information and does not have any functional implication.

RCPT_TO

With this field it is possible to define several receiver addresses separated by the “,” or “;”. The character “space” it is not considered a separator and **CANNOT** be use before or after a “,” or “;”.

ERR_CODE

The sequence to send a message can fail for a lot of reasons, when this happens, the ERR_CODE output change value from 0 (zero) to a different number in according with the following table:

Value	Description
1	Error with the server address
2	CPU error during the socket connection creation
3	Socket configuration error
4	Server connection error
5	Server communication timeout error
6	Internal socket error
7	Username or password encryption error
8	Username or password exceed allowed length
9	Username and password exceed allowed length
10	Address formatting not valid
11	Sender e-mail address exceed allowed length
12	Receiver e-mail address exceed allowed length

In addition to the above described errors, it is possible to get the ones complaining the RFC5321 standard:

221 <domain> Service closing transmission channel

251 User not local; will forward to <forward-path>

252 Cannot VRFY user

421 <domain> Service not available, closing transmission channel (This may be a reply to any command if the service knows it must shut down)

432 A password transition is needed

450 Requested mail action not taken: mailbox unavailable (e.g. mailbox busy or temporarily blocked for policy reasons)

451 Requested action aborted: local error in processing

452 Requested action not taken: insufficient system storage

454 Temporary authentication failure

455 Server unable to accommodate parameters

500 Syntax error, command unrecognized (this may include errors such as command line too long)

501 Syntax error in parameters or arguments

502 Command not implemented

503 Bad sequence of commands

504 Command parameter not implemented

530 Authentication required

534 Authentication mechanism is too weak. This response to the AUTH command indicates that the selected authentication mechanism is weaker than server policy permits for that user

535 server rejects the authentication data

538 Encryption required for requested authentication mechanism

550 Requested action not taken: mailbox unavailable (e.g. mailbox not found, no access, or command rejected for policy reasons)

551 User not local; please try <forward-path>

552 Requested mail action aborted: exceeded storage allocation

553 Requested action not taken: mailbox name not allowed (e.g. mailbox syntax incorrect)

554 Transaction failed (Or, in the case of a connection-opening response, "No SMTP service here")

555 MAIL FROM/RCPT TO parameters not recognized or not implemented

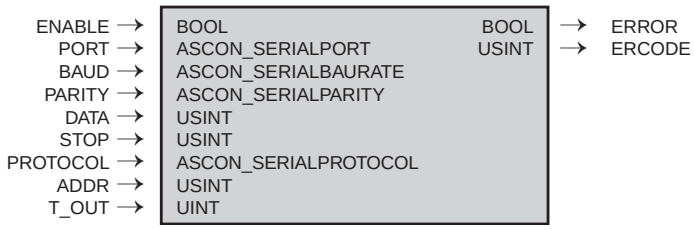
Note: In some situations, the e-mail message cannot be delivered to the receiver due to destination server problems. Usually, at this point, the server sends a warining message to advise of the problems. Because the Sigma CPU handles only outgoing messages, it is impossible to know the result of the delivery.

5-3-38 SERIAL Ports

Caution
This Function Block can be used **only** with the **CU-02** unit.

FB Prototype

SERIAL_PORTS



Input parameters

Label	Type	Description
ENABLE	BOOL	Command to enable the serial port operations
PORT	ASCON_SERIALPORT	Required serial port (default value = X4)
BAUD	ASCON_SERIALBAUDRATE	Communication baud rate (default value = B9600)
PARITY	ASCON_SERIALPARITY	Communication parity (default value = NOPAR)
DATA	USINT	Communication data length (default value = 8)
STOP	USINT	Communication stop bits (default value = 1)
PROTOCOL	ASCON_SERIALPROTOCOL	Required communication agent (default value = SLAVE1)
ADDR	USINT	Slave agent communication address (default value = 1)
T_OUT	UINT	Communication time out factor (this value is a multiple of 100ms - eg. 10 means 10 x 100ms = 1 s)

Output parameters

Label	Type	Description
ERROR	BOOL	Serial port communication error status
ERCODE	USINT	Serial port communication error code

Description

This function block configure the required Serial Port and the parameters to be used with the selected Modbus Agent.

Reference tables

Here are reported the range of the parameters used by the function block.

Input

Label	Options
ENABLE	FALSE = Serial Port DISABLE TRUE = Serial Port ENABLE

Label	Options
PORT	X1 = Communication Port X1 RS232 X3 = Communication Port X3 RS232 X4 = Communication Port X4 RS485 (default value)
PROTOCOL	ASCII = Serial ASCII interface SLAVE1 = Modbus SLAVE_1 Agent (default value) SLAVE2 = Modbus SLAVE_2 Agent MASTER = Modbus MASTER Agent
BAUD	B2400 = 2.4 kbps B4800 = 4.8 kbps B9600 = 9.6 kbps (default value) B19200 = 19.2 kbps B38400 = 38.4 kbps B57600 = 57.6 kbps B115200 = 115.2 kbps
PARITY	NOPAR = NONE (default value) EVEN = EVEN ODD = ODD
STOP	1 or 2 (default value = 1)
DATA	7 or 8 (default value = 8)
T_OUT	0... 65535. Deafult value 10 (1000 ms)

Output

Label	Options
ERCODE	0 = Communication OK 1 = Port not available 2 = Protocol not available 3 = Invalid configuration

5-3-39 SYS_OPRS_MNGT

FB Prototype

SYS_OPRS_MNGT

Input parameters

Label	Type	Description	Range
EN_MB_RTU	BOOL	Modbus RTU delay activation command **evaluated on the rising edge ONLY!!!**	
EN_MB_TCP	BOOL	Modbus TCP delay activation command **evaluated on the rising edge ONLY!!!**	
DELAY_MB_RTU	USINT	Modbus RTU activation cycle time delay [ms]	40… 255
DELAY_MB_TCP	USINT	Modbus TCP activation cycle time delay [ms]	40… 255

Output parameters

Label	Type	Description
STS_MB_RTU	BOOL	Modbus RTU delay activation status
STS_MB_TCP	BOOL	Modbus TCP delay activation status
TSAM_MAX	TIME	Maximum cycle time reached by the application since Power ON
TSAM_ACT	TIME	Last actual cycle time of the application
TSAM_MIN	TIME	Minimum cycle time reached by the application since Power ON

Description

This function block allows to define the operational communication parameters needed to manage the delay to execute the Modbus RTU and/or Modbus TCP communication agents. The function block provides also information on the application actual cycle time and the maximum and minimum cycle time reached during normal operations. Of course, it should not be used in a timer task otherwise you will get the task execution time.

Caution

In order to avoid communications and/or operational problems, SYS_OPRS_MNGT must be used in a sort of “one-shot” mode at CPU Power-ON (and not cyclically).

Default Values Table

Input	Default Value
EN_MB_RTU	FALSE
EN_MB_TCP	FALSE
DELAY_MB_RTU	100
DELAY_MB_TCP	100

5-3-40 TCP/IP Port

FB Prototype

TCP_IP_Port

Input parameters

Label	Type	Description
ENABLE	BOOL	Command to enable the TCP/IP port operations
PORT	UINT	Required server TCP/IP port (default value = 502)
USESEC	BOOL	Enable/Disable security functions (default value = FALSE)
T_OUT	UINT	Communication time out factor (this value is a multiple of 100ms - eg. 10 means 10 x 100ms = 1 s)

Output parameters

Label	Type	Description
ERROR	BOOL	TCP/IP port communication error status
ERCODE	USINT	TCP/IP port communication error code

Description

This function block configure the required TCP/IP Port and the parameters to be used with the selected Modbus TCP/IP Agent.

Reference table

Input

Label	Options
ENABLE	FALSE = TCP Port DISABLE TRUE = TCP Port ENABLE
PORT	Default value = 502
T_OUT	0... 65535. Deafult value 10 (1000 ms)

Output

Label	Options
ERCODE	0 = Communication OK 1 = Server not available 2 = Resource not available 3 = TCP/IP problems

Appendix A
Reference documents

[1] "*Infoteam OpenPCS programming system – user manual*" – version 6.0 english

[2] "*IEC 61131-3: Programming Industrial Automation Systems*" – Karl-Heinz John, Michael Tiegelkamp - Springer

[3] "*Ascon Firmware Function Block Library*"

[4] "*IEC 61131-3 Function Block Library*".

[5] "*Estensioni per gestire porte di comunicazione dell'ambiente OpenPCS*" V1.0 – Maurizio Grassi

[6] *CANopen Extension for IEC61131 – Software manual*" – Edition March 2005 – Systec Electronic

[7] "*CiA Draft Standard 405 – CANopen Interface and Device Profile for IEC61131-3 Programmable Devices*" – version 2.0

[8] "*CANopen Application Layer and Communication Profile - CiA DS301 v 4.02*"

[9] "*Modbus Messaging on TCP/IP implementation guide*" - http://www.Modbus-IDA.org

[10] "*MODBUS over Serial Line Specification & Implementation guide*" - http://www.Modbus-IDA.org

[11] "*MODBUS APPLICATION PROTOCOL SPECIFICATION*" - http://www.Modbus-IDA.org

[12] "*CU-02 Installation manual*" (code: J30 - 658 - 1ACU02 E).

[13] "*CU-02 User manual*" (code: J30 - 478 - 1ACU02 E).

[14] "*MP-01 Installation manual*" (code: J30 - 658 - 1AMP01 E).

[15] "*MP-01 User manual*" (code: J30 - 478 - 1AMP01 E).

[16] "*micro**PAC** I/O modules Installation Manuals*".

[17] "*micro**PAC** I/O modules User Manuals*".