- PROFIBUS

smar
First in Fieldbus

FUNCTION BLOCKS
INSTRUCTION MANUAL

MAR / 22
VERSION 2

PROFI® BUS

FBLOC- PAME

smar
NOVA SMAR S/A
www.smar.com.br

TABLE OF CONTENTS

Section 1

INTRODUCTION TO FUNCTION BLOCK APPLICATION

Overview

Functional block applications are defined as plant or factory applications that perform one or more automatic monitoring and control functions.

Function Block

Functional blocks represent the basic automation functions performed by the functional block application. Each functional block processes input parameters according to a specified algorithm and an internal set of control parameters. They produce output parameters that are available for use within the same functional block application or by other functional block applications.

Transducer Block

Transducer blocks insulate functional blocks from the specifics of I/O devices, such as sensors, actuators, and switches. Transducer blocks control access to I/O devices through a device independent interface defined for use by functional blocks. Transducer blocks also perform functions, such as calibration and linearization, on I/O data to convert it to a device independent representation. Their interface to functional blocks is defined as one or more implementation independent I/O channels.

Physical Block

Physical blocks are used to define hardware specific characteristics of functional block applications. Similar to transducer blocks, they insulate functional blocks from the physical hardware by containing a set of implementation independent of hardware parameters.

Function Block Definitions

Functional blocks are defined by their inputs, outputs, control parameters, and by the algorithm that operates on these parameters. Functional blocks are identified using a name (Tag) and a numeric index.

Tags provide a symbolic reference to functional blocks. They are unambiguous within the scope of a fieldbus system. Numeric index are numbers assigned to optimize access to functional blocks. As opposed to functional block tags, which are global, numeric index have meaning only within the application that contains the functional block.

Functional block parameters define the inputs, outputs, and the data used to control functional block operation. They are visible and accessible over the network. Additional parameters, called “contained within” parameters are used to define the private data of a functional block. Although visible over the network, they may not participate in cyclic data exchanges.

Cyclic data Exchange

The cyclic data exchange indicates that an input parameter of one functional block obtains its value from specific output parameters of another functional block in other device cyclically. There are no internal links between functional blocks into the device.

In general, a functional block of transmitter or actuator devices exchange data cyclically with the controller (for instance, a master PLC). Typically, the transmitter gets the data from the sensor and the Controller device requests these data, makes some calculation and sends the result from an actuator that will take some actions in the process.

In order to configure which information will be exchanged, the master gets the information about the devices consulting the GSD file. For each device there is a GSD file. This file has an identifier that is unique and identifies the device for the master equipment. The GSD file has all the information of the device, e.g. transmitter types, how many and the kind of blocks the device have and the possible cyclic configuration that the equipment supports. One example of the GSD file is given in the Figure 1.1.

```
;*******************************************************
;**   GSD file for LD303 - Pressure Transmitter          **
;**   smar0895.GSD                                      **
;*******************************************************
#Profibus_DP
GSD_Revision                  = 2
Vendor_Name                   = "SMAR"
Model_Name                    = "LD303"
Revision                      = "1.0"
Ident_Number                  = 0x0895 ; 0x9740
:                               :
:                               :
;Modules for Analog Input
Module    = "Analog Input (short)   "   0x94
EndModule
Module    = "Analog Input (long)    "   0x42, 0x84, 0x08, 0x05
EndModule

;Module for Totalizer
Module   = "Total "                 0x41, 0x84, 0x85
               EndModule
Module   = "Total_Settot "          0xC1, 0x80, 0x84, 0x85
EndModule
Module   = "Total_Settot_Modetot "  0xC1, 0x81, 0x84, 0x85
EndModule

;Empty module
Module    = "EMPTY_MODULE"          0x00
EndModule
```

Figure 1.1 - Example of the GSD file

Configuration Data (CFG_DATA)

In the cyclic data exchange there are different parameters for functional blocks (for further details, refer to the Cyclic CFG_DATA parameter of each functional block). The differences come from different user needs regarding the necessary scope of information (with feed back of the actual position of the output or without) and the way of integration in the control task (with or without remote cascade). During configuration the operator chooses the parameter combination and the tools in order to concatenate an internal configuration string (in the figure 1, the config string or *identifier byte* are those numbers defined in each module).

The Master looks at the GSD file searching for the supported configuration for a specific block. For example, according to the GSD file of the figure 1, the user can configure the Output of Analog Input block ("Analog Input (short) ") or the Output of the Totalizer ("Total ") or both.

In the GSD file there is a section where is defined all supported configuration that is possible for the device. Each permitted configuration is started with the “Module” word and finished with “End_Module” word. In the description of each “Modules” there is a string name and some configuration data number (CFG_DATA). These numbers are internal to the Master and describe the parameter combination (how many parameters, datatype, length, etc.) of the functional block. For Example,

Module = "Total_Settot " 0xC1, 0x80, 0x84, 0x85

String describing the parameter of the cyclic configuration

Configuration Data (CFG_DATA)

There is a special module “EMPTY_MODULE” indicating that one specific Functional block will not participate from the cyclic data exchange.

For each functional block that support cyclic data exchanged it is necessary to configure one (and only one) cyclic configuration combination for this block OR the “EMPTY_MODULE” (indicating that the user does not desire to use the block).

According to the Profibus-PA Profile 3.0, there are two kinds of configuration for a parameter combination: the Identifier byte (or short identifier) and the Extended Identifier Format (or long identifier). Some combination has only one kind of configuration and other has both. The device of the Smar supported both kind of configuration. Then, in the example above the user can choose either "Analog Input (short)" or "Analog Input (long)" and would have the same result for the configuration. In same configuration the user can mixed both kind of configuration: long and short.

If there are more than one functional blocks with more than one cyclic parameter, the data elements are concatenated either in the Input or in the Output data frame (this depends on the configuration). The order of the parameters for one functional block in the input and output data frame is from the lower to the higher Relative Index in the Parameter Attribute table of the functional blocks (see example in the figure 3). If there are more then one FB with the same type in a device (for example 3 AI FBs) the order of the cyclic parameters in the Input and Output data frame is the same as the order of the functional block into the directory of the device (see example in the Figure 2).

All the function blocks of the device need to be configured in the “Config data” in the same order of the directory into the device (see in the table “FB set and FB type availability” the order of the blocks for each device).

For instance, the LD303 device has one AI and one TOT functional block, the order to the “Config data” must be AI and TOT respectively. The IF303 device has 3 AI and 3 TOT, the order of the “Config data” must be AI1, AI2, AI3, TOT1, TOT2, TOT3 respectively.

The Figures below illustrates the data exchange between the master and the slave devices. In the Figure 1.2, in the FI303 device (3 AO blocks) only the First AO is configured (the order of the configuration must be respected). Then in the configuration, the user needs to indicate that the AO2 and AO3 block are not being used (configuring with “Empty Module”). For the AO1 block it is chose “RCAS_IN + RCAS_OUT” data exchange, where the output data from the master functional block goes to the RCAS_IN input of the slave device. And the RCAS_OUT output of the AO1 functional block goes to the Input of the master functional block.

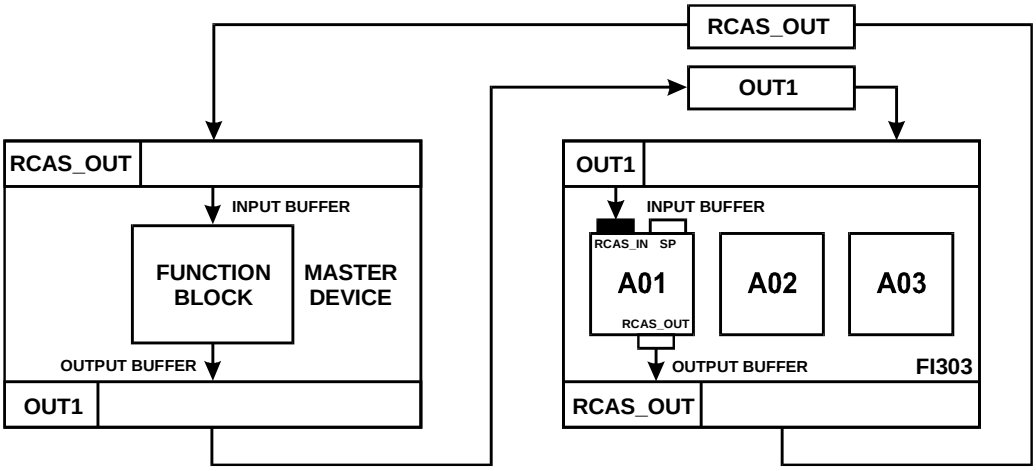

Figure 1.2 – Example of the cyclic data exchange - Configuration Data “Rcas_In + Rcas_Out” + “Empty_Module” + “Empty_Module” to the FI303 device

In the Figure 1.3, in the FY303 device (1 AO block) it is chose “SP + RB + RCASIN + RCASOUT + POSD + CB” data exchange, where there are 2 inputs and 4 outputs to be transported from the slave to the master. In that case, the order in the input and output buffer depends on the relative index of the block parameter. For example, the input data to the slave OUT1 and OUT2 (in the frame) respects the ascending order of the AO block: SP receives the OUT1 and RCAS_IN receives the OUT2 (see the block parameters table to check the relative index of the parameter). The same occurs with the 4 outputs of the AO block. The order of the output buffer is READBACK, RCAS_OUT, POS_D and CHECKBACK.

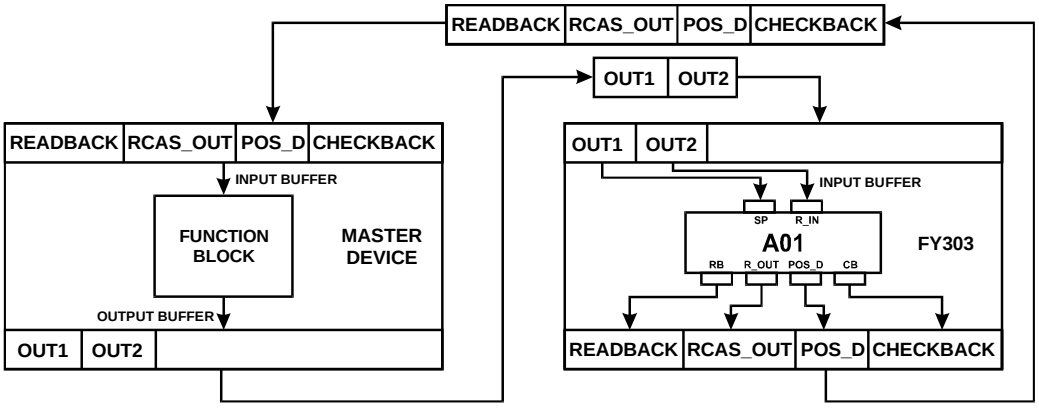

Figure 1.3 – Example of the cyclic data exchange - Configuration Data “SP + RB + RCASIN + RCASOUT + POSD + CB” to the FY303 device

Example of the Configuration

1) Considering that the LD303 transmitter has 2 functional blocks: 1 AI and 1 TOT. The following settings are valid for the configuration:

- Configuring the output of the AI block and the output of the TOT block:

"Analog Input (short) " and “Total” or
“0x94, 0x41, 0x84, 0x85”

- Configuring the output of AI block only:

"Analog Input (long) " and “Empty module” or
“0x42, 0x84, 0x08, 0x05,0x00”

- Configuring the output of TOT block only:

“Empty module” and “Total_Settot” or
“0x00, 0xC1, 0x80, 0x84, 0x85”

2) Considering that the FI303 transmitter has 3 functional blocks, 3 AO blocks. The following settings are valid for the configuration:

- Configuring the RcasIn input, RcasOut and checkback outputs of the AO1 only (first Analog Input block):

"RCAS_IN + RCAS_OUT + CB" , “EMPTY_MODULE” and “EMPTY_MODULE” or
“0x97,0xA4,0x00,0x00”

- Not configure the AO1, configuring the SP input, Readback and PosD outputs of the AO2, and the SP the AO3, it results:

“EMPTY_MODULE”, "SP + RB + POSD " and “SP” or
“0x00,0x96,0xA4,0xA4”

- Configuring the RCAS_IN input and RCAS_OUT output of the 3 AOs (AO1,AO2 and AO3):

“RCAS_IN + RCAS_OUT”, “ RCAS_IN + RCAS_OUT”, “ RCAS_IN + RCAS_OUT” or
“0xB4,0xB4,0xB4”

Notes:

- In the example 1, the order of the functional blocks in the LD303 is: AI and TOT respectively. Then in the configuration is necessary choosing one valid configuration for the AI block and TOT block. If the user chooses the TOT config first, the cyclic connection will be not established. In the example 2 is necessary to configure the AO1, AO2 and AO3 respectively in this order.

- Some configurators use the description of the module in the GSD file, e.g., “Analog Input (short)”. Others use the identifier number, e.g., “0x94”. Then in the example are showed both kind of configuration.
- In the configuration using identifier numbers all functional blocks has already included in the string, e.g. “0x00,0x96,0xA4,0xA4” of the example 2, has “0x00” (Empty module) for the first block, “0x96,0xA4” (SP, READBACK,POS_D) for the second block, and “0xA4” (SP) for the third block.
- The Short or the Long identifier gives the same effect.

Information Access

Functional block information may be grouped for access depending on how it is to be used. For while, there is defined for access purposes only the dynamic operation data view.

To support access of operator interface information during functional block execution, two levels of network access are defined, one for operational traffic and one for background traffic. Operator interface traffic is transferred as background traffic to prevent it from interfering with the operation of time-critical functional blocks.

Function Block Application Structure

Functional block applications are modeled as a set of functional blocks coordinated to execute a related set of operations.

Functional block model is real-time algorithm that transforms input parameters into output parameters. Its operation is controlled through setting the control parameters.

The transmitter and actuator devices have functional blocks that are modeled to provide or get process value with functional blocks from controller device.

The interoperation between functional blocks from different devices is modeled though the data exchange between input parameter of one functional block to an output parameter of another. Functional blocks can be bound together within and across devices. Interfaces between functional blocks located in the same functional block application are locally defined. Those interfaces between functional blocks in different devices use the communication services.

To support functional block operation, the functional block architecture also provides transducer and resource blocks, and display objects.

Functional block Application Process represents the functional block application as an integrated set of these components accessed to its network interface.

Block Object

A block object represents a logical processing unit composed of a set of input, processing, and control parameters and an associated algorithm.

During system operation, a short hand reference, known as a numeric index is used for block access purposes. A block’s numeric index is unique only within the functional block application where it exists.

The algorithm of a block is identified by its type and the revision level of its type. This information indicates how the execution of the algorithm is affected by control parameters.

Block Parameters

Parameters define the inputs, outputs, and control data for a block. Their relationship to each other and to the block algorithm is shown below.

Parameter Identifiers

Parameter names are unique within a block. Within a system, a parameter can be unambiguously identified by qualifying its name with the tag of its block.

Parameter Usage

Parameters are defined for a block for a specific purpose. Each is defined for use as an input, an output, or a control parameter. Control parameters are also referred to as “contained” parameters because they may not be connected cyclically with parameters in other blocks. Each type of usage is defined as follows:

Contained

A contained parameter is a parameter whose value is configured, set by an operator, higher level device, or calculated. It may not be linked to another functional block input or output. The mode parameter is an example of a contained parameter common to all blocks.

Output

An output parameter is a parameter that may be connected cyclically to an input parameter of another functional block. In general, output parameters contain status. The output status indicates the quality of the parameter value and the mode of the block when it was generated.

The value of an output parameter may not be obtained from a source external to the block. It may be generated by the block algorithm, but does not have to be.

The values of certain output parameters are dependent on the value of the mode parameter of the block. These output parameters may be referred to as mode-controlled output parameters.

Blocks whose purpose is to generate a single output contain one parameter designed as the primary output parameter. Primary outputs are used by other blocks for control or calculation purposes. These blocks also contain secondary output parameters such as alarm and event parameters that represent a supporting role to the primary output parameter.

Input

An input parameter obtains its value from a source external to the block. An input parameter may be connected cyclically to an output parameter of another functional block. Its value may be used by the algorithm of the block.

In general, input parameter values are accompanied by status. When an input parameter is connected cyclically to an output parameter, the status will be provided as the status of the output parameter (when the parameter has a status). When it is not connected cyclically to an output parameter, the status will indicate that the value was not provided by an output parameter. When an expected input parameter value is not received, the functional block supported services responsible for delivering the data will set the status of the input parameter to indicate the failure.

If an input parameter is not connected cyclically to an output parameter, then it will be treated as a constant value by the functional block application. The difference between not connected cyclically input parameters and contained parameters is that input parameters have the capability to support a cyclic connection and contained parameters do not.

Blocks whose purpose is to transform or operate on a single input will contain one parameter designed as the primary input parameter. One input parameter of some types of blocks is designated as the primary input parameter. Primary inputs are used for control or calculation purposes. These blocks may also contain secondary input parameters that support processing done on the primary input parameter.

Parameter Relationships

The execution of a block involves the inputs, outputs, contained parameters, and the algorithm of the block. The execution time for a block’s algorithm is defined as a parameter of the block. Its value is dependent on how the block was implemented.

The input parameters are used by the algorithm in conjunction with the state of the functional block application containing the block to determine if the algorithm can achieve the target mode established for it. The TARGET_MODE parameter indicates what mode of operation is desired for the block. It is normally set by a control device or the operator.

Under certain operating condition a block may not be able to function in the requested mode. In such cases, the actual mode reflects the mode it is able to achieve. Comparison of the actual against the target indicates whether the target was achieved.

The values for the mode parameter for a block are defined by the Permitted Mode parameter. Thus, the modes available for controlling a block may vary with each block.

Once the actual mode is determined, the block execution progresses and the outputs are generated.

Parameter Status

The Status argument of the input and output parameters is made to show to another blocks the current status of the source block.

The Status field is composed of three parts: Quality, Sub-Status and Limits.

Quality – It indicates the quality of the parameter value.

- Good Cascade – The quality of the value is good, and it may be part of a cascade structure.
- Good Non-Cascade – The quality of the value is good, and the block doesn't support a cascade path.
- Uncertain – The quality of the value is less than normal, but the value may still be useful.
- Bad – The value is not useful.

Sub-Status – The sub-status is a complement of the quality status and takes information to initialize or break a cascade control, alarms and others. There are different sets of sub-status for each quality.

Limits – It provides information whether the associated value is limited or not, as well the direction. The limits are classified as: Not Limited, High Limited, Low Limited, Constant.

When an input parameter is "linked" to an output parameter through the cyclic data exchange, the whole structure (status and value) is received from the bus. If the input is not "linked", then the status may be set manually by the user, as well the value.

Composition of Status

The Status has the following composition:

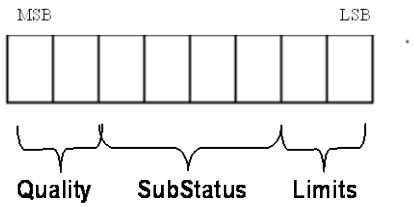

Figure 1.4 – Composition of Status

The quality, sub-status, and limit components of status are defined as follows:

Quality - The quality used will be determined by the highest priority condition:
0 = Bad
1 = Uncertain
2 = Good (Non-cascade)
3 = Good (Cascade)

Sub-status - Sub-status values in the status attribute are defined as shown in the table 1.

Limits - The following limit conditions will be always available in the status attribute.
0 = Not limited
1 = Low limited
2 = High limited
3 = Constant

Examples:
0xC1 (in hexadecimal) is “Good-Cascade Non Specific and Low Limited” status
0xCF(in hexadecimal) is “Good-Cascade Not invited and Constant ” status
0x4E(in hexadecimal) is “Uncertain Initial Value and High Limited” status

Quality	Sub-Status	Limit	Hexa Value	Decimal Value	Not in cascade	Forward path of cascade	Backward path of cascade
GoodNC	0 = ok **(lowest priority)**	0 – Not Limited 1 – Low Limited 2 – High Limited 3 - Constant	0x80	128	X		
GoodNC	1 = Active Update Event	0 – Not Limited 1 – Low Limited 2 – High Limited 3 - Constant	0x84	132	X		
GoodNC	2 = Active Advisory Alarm	0 – Not Limited 1 – Low Limited 2 – High Limited 3 - Constant	0x88	136	X		
GoodNC	3 = Active Critical Alarm	0 – Not Limited 1 – Low Limited 2 – High Limited 3 - Constant	0x8C	140	X		
GoodNC	4 = Unacknowledged Update Event	0 – Not Limited 1 – Low Limited 2 – High Limited 3 - Constant	0x90	144	X		
GoodNC	5 = Unacknowledged Advisory Alarm	0 – Not Limited 1 – Low Limited 2 – High Limited 3 - Constant	0x94	148	X		
GoodNC	6 = Unacknowledged Critical Alarm	0 – Not Limited 1 – Low Limited 2 – High Limited 3 - Constant	0x98	152	X		
GoodNC	8 = Initiate Fail Safe (IFS)	0 – Not Limited 1 – Low Limited 2 – High Limited 3 - Constant	0xA0	160	X		
GoodNC	9= Maintenance required	0 – Not Limited 1 – Low Limited 2 – High Limited 3 - Constant	0xA4	164	X		
Uncertain	0 = Non-specific	0 – Not Limited 1 – Low Limited 2 – High Limited 3 - Constant	0x40	64	X		

Uncertain	1 = Last Usable Value	0 – Not Limited 1 – Low Limited 2 – High Limited 3 - Constant	0x44	68	X		
Uncertain	2 = Substitute	0 – Not Limited 1 – Low Limited 2 – High Limited 3 - Constant	0x48	72	X		
Uncertain	3 = Initial Value	0 – Not Limited 1 – Low Limited 2 – High Limited 3 - Constant	0x4C	76	X		
Uncertain	4 = Sensor Conversion not Accurate	0 – Not Limited 1 – Low Limited 2 – High Limited 3 - Constant	0x50	80	X		
Uncertain	5 = Engineering Unit Range Violation	0 – Not Limited 1 – Low Limited 2 – High Limited 3 - Constant	0x54	84	X		
Uncertain	6 = Sub-normal	0 – Not Limited 1 – Low Limited 2 – High Limited 3 - Constant	0x58	88	X		
Uncertain	7 = Configuration Error	0 – Not Limited 1 – Low Limited 2 – High Limited 3 - Constant	0x5C	92	X		
Uncertain	8 = Simulated Value	0 – Not Limited 1 – Low Limited 2 – High Limited 3 - Constant	0x60	96	X		
Uncertain	9 = Sensor Calibration	0 – Not Limited 1 – Low Limited 2 – High Limited 3 - Constant	0x64	100	X		
GoodC	0 = ok	0 – Not Limited 1 – Low Limited 2 – High Limited 3 - Constant	0xC0	192		X	X
GoodC	1 = Initialization Acknowledged(IA)	0 – Not Limited 1 – Low Limited 2 – High Limited 3 - Constant	0xC4	196		X	

GoodC	2 = Initialization Request(IR)	0 – Not Limited 1 – Low Limited 2 – High Limited 3 - Constant	0xC8	200			X
GoodC	3 = Not Invited (NI)	0 – Not Limited 1 – Low Limited 2 – High Limited 3 - Constant	0xCC	204			X
GoodC	5 = Do Not Selected(NS)	0 – Not Limited 1 – Low Limited 2 – High Limited 3 - Constant	0xD4	212		X	
GoodC	6 = Local Override(LO)	0 – Not Limited 1 – Low Limited 2 – High Limited 3 - Constant	0xD8	216			X
GoodC	8 = Initiate Fail Safe (IFS)	0 – Not Limited 1 – Low Limited 2 – High Limited 3 - Constant	0xE0	224		X	
Bad	0 = Non-specific	0 – Not Limited 1 – Low Limited 2 – High Limited 3 - Constant	0x00	0	X	X	X
Bad	1 = Configuration Error	0 – Not Limited 1 – Low Limited 2 – High Limited 3 - Constant	0x04	4	X	X	X
Bad	2 = Not Connected	0 – Not Limited 1 – Low Limited 2 – High Limited 3 - Constant	0x08	8			
Bad	3 = Device Failure	0 – Not Limited 1 – Low Limited 2 – High Limited 3 - Constant	0x0C	12	X	X	X
Bad	4 = Sensor Failure	0 – Not Limited 1 – Low Limited 2 – High Limited 3 - Constant	0x10	16	X	X	X
Bad	5 = No Communication, with last usable value	0 – Not Limited 1 – Low Limited 2 – High Limited 3 - Constant	0x14	20			

Bad	6 = No Communication, with no usable value	0 – Not Limited 1 – Low Limited 2 – High Limited 3 - Constant	0x18	24			
Bad	7 = Out of Service **(highest priority)**	0 – Not Limited 1 – Low Limited 2 – High Limited 3 - Constant	0x1C	28			

Table 1.1 – Parameter Identifiers

Example: Conversion from Enumeration to Number

The following formula is used to obtain the enumeration number of a determinate status attribute:

$$\text{Decimal Value Status} = 64 * \text{Quality} + 4 * \text{Sub} - \text{Status} + \text{Limits}$$

For example, considering the following status:
"Uncertain - Initial Value - High Limited"
Where:
Quality = "uncertain" = 1
Sub-Status = "Initial Value" = 3
Limit = "High Limited " = 2
Applying the formula:

Decimal Value Status = 64 * 1 + 4 * 3 + 2 = 78 (in decimal) or 0x4E (in Hexadecimal)

Example: Conversion from Number to Enumeration

There are many ways to convert the enumerate number to the status string. Below is shown two forms to do this.

1) Expressing the number in binary:
   Hex Value Status = 78 = 0x4E = 01001110 (in binary)

Dividing this binary number in Quality, Sub-status and Limit fields:
Quality = 01 = 1 = "Uncertain"
Sub-Status = 0011 = 3 = "Initial Value"
Limit = 10 = 2 = "High Limited"
The corresponding status is "Uncertain - Initial Value - High Limited".

2) Using the value of status in decimal format:
   Decimal Value Status = 78

Divided the number by 64. The quotient will be the Quality and save the remainder:
   Quality = 78 / 64 = 1
   Remainder = 14

Divided the remainder by 4. The quotient will be the Sub-Status and the remainder will be the limit:
   SubStatus = 14 / 4 = 3
   Limit = 2

Channel Handling

The Functional blocks Analog Input, Analog Output and Totalizer are connected to the transducer block through the CHANNEL.

The CHANNEL is an Unsigned16 parameter and its value means a pointer to the related transducer block and its parameter. It consists of 2 elements:

- The TransducerID (First Byte, it is 1 for the first transducer, 2 for the second, etc. Depend on the order in the directory).

- The Relative Index of the used Transducer Block parameter (Second Byte).

Optionally, the channel may be disconnected setting its value to zero (0x00).

Channel of Input Function Blocks

Typically, a transducer block from a transmitter device has three parameters that can be connected with the input functional block: Primary Value (PV), Secondary Value 1 (SV1) and Secondary Value 2 (SV2).

The functional blocks connected to the transducer may be connected with each one of these outputs, although it depends on the device and the FB. There are some rules described below:

- The Totalizer block can be configured only with the PV parameter;
- Different kinds of functional blocks (e.g., AI and TOT) can be configured with the same channel value, but functional blocks of the same type (e.g., AI and AI or TOT and TOT) cannot reference the same transducerID in the channel;
- The Relative index of the transducer output parameters (PV, SV1, SV2) changes for each device type.

Channel of Output Function Blocks

The Transducer from an Actuator device has only one reference parameter, and the Relative Index of the related functional block channel parameter shall be 0 (zero).

The AO block has two channels: one to connect the AO block to the transducer (AO → TRD), OUT_CHANNEL, where the AO block sends the calculate value to the transducer. The other is to connect the TRD to the AO (TRD → AO), IN_CHANNEL, where the transducer sends the actual position to the READBACK parameter of the Analog Output block. But, the Smar devices do not need 2 channels, then it is recommended to use both the channels with the same value.

Example Using the Channel

1) Configuring the channel of the LD303

Considering the LD device that has one transducer and 2 functional blocks: 1 AI and 1 TOT. The possible channels of this FBs could be:
1. There is only one transducer block then the first byte is 1.
2. The TOT block must be configured with the PV parameter. Then the second byte of the channel needs to be PV. (in the LD device the PV of the transducer has the relative index 18 (0x12))
3. In the AI block the second byte of the channel may be PV, SV1 or SV2 (Relative index of SV1 = 29 (0x1D), Relative index of SV1 = 31(0x1F))

The channel of the AI and TOT block would be:
AI.CHANNEL = 0x011D (if the transducer output chose was SV1) or 0x011F (If the output was SV2) or 0x0112 (if the parameter chose was PV)
TOT.CHANNEL = 0x0112 (PV output)

2) Configuring the channel of the FI303

Considering the FI device that has 3 transducers and 3 AO Functional blocks. The possible channels of these FBs could be:
1. There are 3 transducers then the first byte may be 1 (for the first transducer), 2 (for the second), and 3 (for the third).
4. The second byte to the actuator device is ever zero (0).

The CHANNEL for the AO blocks would be:
AO1.IN_CHANNEL = AO1.OUT_CHANNEL = 0x0100 (first AO configured with the first transducer)
AO2.IN_CHANNEL = AO2. OUT_CHANNEL = 0x0200 (second AO configured with the second transducer)
AO3.IN_CHANNEL = AO3. OUT_CHANNEL = 0x0300 (Third AO configured with the third transducer)

Output Calculation

When the actual mode is AUTO or RCAS, the normal algorithm is executed. This calculation is specific for each Functional block type. If the mode is a “manual” mode, the output is just following a value provided by the user (Man or LO).

In Manual mode the status of the output will be “Uncertain Simulate Value”, it will be indicating that the user are writing the output.

Remote Cascade Control

In a remote cascade, the upper control block provides an output value and status, which becomes the remote cascade input to the lower block.

The lower block in the remote cascade provides an output value, which is communicated to the upper block as back-calculation input.

Based on the following example, it will be shown the process of remote cascade initialization.

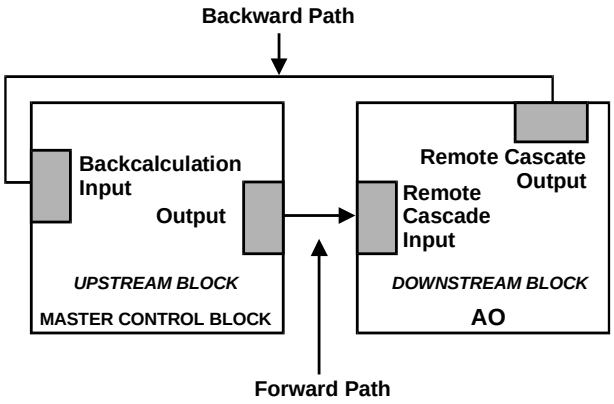

Figure 1.5 – Example of the Remote Cascade

There are four steps to complete a remote cascade initialization:

1. Not cascade mode – As the AO block is in Auto mode, it indicates to the upstream block (in the master device) that it is not using the output of the master. Then the downstream block (AO) sends the actual using value (SP value) and the status GoodC-Not Invited backward.

AO
Target_Mode = Auto
RCAS_IN.Status = Any status (because the block is not in cascade mode)
MODE_BLK.Actual = Auto
BKCAL_OUT.Status = GoodC-Not Invited

2. Initialize – The user changes the target mode of the downstream block (AO) to RCas, then the AO block send the “GoodC-IR” status in RCAS_OUT output. The value of RCAS_OUT is the initial value for the upstream block starts to calculate.

The AO block waits to the “Good-Initiate Acknowledge” status in its RCAS_IN input. This input is connected to the output of the upstream block. The AO block continue running the algorithm with the last good value from SP.

AO
Target_Mode = Rcas
RCAS_IN.Status = Good Cascade and Sub-status different from Initiate Acknowlegde
MODE_BLK.Actual = Auto
BKCAL_OUT.Status = GoodC-Initialization Request (IR)

3. Initialization complete – The AO block goes to RCas, because the upstream block sent "GoodC-IA" status to the RCAS_IN input. Now, the AO block starts to get the SP value from the RCAS_IN input.

AO
Target_Mode = RCas
RCAS_IN.Status = Good Cascade - Initiate Acknowlegde (IA)
MODE_BLK.Actual = RCas
RCAS_OUT.Status = GoodC-Ok

4. Cascade complete – The upstream block changes the status of output from GoodC-IA to GoodC-ok. The AO block gets the SP value from the RCAS_IN input.

AO
Target_Mode = RCas
RCAS_IN.Status = Good Cascade - Ok
MODE_BLK.Actual = RCas
RCAS_OUT.Status = Good Cascade - Ok

Note:
Lower block can not execute in RCas: the target mode of lower block is not RCas, or there is any condition forcing the lower block to a higher priority mode as fail Safe conditions. (See details in the section Mode Parameter).

Mode Parameter

a) Mode types

The operation of the block is summarized for each mode type as follows:

Out of Service (O/S):

The block is not being evaluated. The output is kept at last value or, in the case of power failure, the output can be programmed to keep a determined value.

Local Override (LO):

The block output is not being calculated, although it may be limited. It applies to control block that supports a track input parameter. When the block is in LO, its output is tracking the value set by the user locally (by the magnetic keys). The user cannot change the outputs from the host remote.

Manual (Man):

The block output is not being calculated, although it may be limited. The operator may set directly the outputs of the block.

Automatic (Auto):

The normal algorithm calculates the block output. If the block has a setpoint, it will be used a local value that may be written by an operator through an interface device.

The block output is calculated using the input from the transducer block, for an input Functional block and using a setpoint value provide by a host or an operator through an interface device in case of an output Functional block.

Remote Cascade (RCas):

The block setpoint is set by a Control Application through the remote cascade parameter RCAS_IN. The normal algorithm calculates the block output based on that setpoint.

Auto, and RCas are the "automatic" modes, that calculate the primary output using the normal algorithm. The "manual" modes are LO and Man.

Mode type	Source of SP	Source of OUT
O/S	User	User
LO	User	User
Man	User	User
Auto	User	Block algorithm
Rcas	Control Application running on an interface device	Block algorithm

Table 1.2 – Mode Parameters

b) Elements of Mode Block

The mode block is composed of two parameters TARGET_MODE and MODE_BLK that is defined in all blocks. These parameters are explained bellow:

- **TARGET_MODE** - This is the mode requested by the operator. Only one mode from those allowed by the permitted mode parameter may be requested, that check will be done by the device.

- **MODE_BLK** - This parameter is calculating by the algorithm based in the inputs and the Target mode. Therefore the user can not write in the attributes of this parameter. It is defined as having three elements:
  - **Actual** - This is the current mode of the block, which may differ from the TARGET_MODE based on operating conditions and block configuration, as input parameter status and bypass configuration, for example. Its value is *always* calculated as part of block execution.
  - **Permitted** – It defines the modes that are allowed for an instance of the block. It is like a list of mode types selected from the supported modes.
  - **Normal** - This is the mode which the block should be set to during normal operating conditions. The normal attribute is used as a *reminder*. It does not affect the algorithm calculation.

The execution of a functional block will be controlled through the mode parameter. The user sets the TARGET_MODE, which indicates what mode of operation is desired for the block. Then, the algorithm evaluates if the block can be executed in the *requested mode or the nearest higher priority mode possible*. The actual mode reflects the mode of block operation.

c) Priority of mode

The concept of priority is used when the block calculates the actual mode.

Mode	Description	Priority
O/S	Out of Service	7 – highest
LO	Local Override	5
Man	Manual	4
Auto	Automatic	3
Rcas	Remote Cascade	1 - lowest

Table 1.3 – Priority of the Mode

d) Mode Calculation

The actual mode will be calculated based on the following:

- Each mode type has some conditions that force the actual mode to be of higher priority than the target mode.
- Starting from the highest priority mode (O/S), it is analyzed its corresponding conditions. If they are present, then the actual mode will be this one, otherwise it is necessary to check the conditions for the next lower priority mode (LO, Man, Auto and Rcas) till the target mode, exclusive. For instance, if the target mode is RCas, it is necessary to check the conditions for O/S, LO, Man and Auto, in this order. If all those conditions are false, the actual mode will be the target mode.

Mode	Conditions
O/S	- Target Mode is O/S
LO	- Target Mode is LO.
Man	- Target mode is Man. - Target mode is RCas, in the previous execution the actual mode was Man and the input RCAS_IN status is not Good_IA.
Auto	- Target mode is Auto. - Target mode is RCas and in the previous execution the actual mode was LO or O/S. - Target and in the previous execution the actual mode was Rcas and block goes to fail safe. (RCAS_IN.status is Good_IFS or RCAS_IN status is bad for the time greater than FSAFE_TIME) - Target mode is Rcas and in the previous execution the actual mode was Auto and the input RCAS_IN status is not Good_IA.
RCas	- Target mode is Cas and cascade initialization has just completed (the input RCAS_IN status is Good_IA. - Actual mode of last execution was Rcas.

Table 1.4 – Mode Calculation

e) Specific Information for Driver Developers

Internally, for each mode attribute is assigned within the bitstring in the following manner (the bitstring is the same to the TARGET_MODE and the elements Actual, Normal of the MODE_BLK.

Mode	Hex value	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
O/S	0x80	1	0	0	0	0	0	0	0
LO	0x20	0	0	1	0	0	0	0	0
Man	0x10	0	0	0	1	0	0	0	0
Auto	0x08	0	0	0	0	1	0	0	0
Rcas	0x02	0	0	0	0	0	0	1	0

Table 1.5 – Mode Bitstring

The element supported of the MODE_BLK is logic OR with the respective supported modes. For example, the Analog Input block has the Supported mode = O/S, AUTO, MAN, then the supported mode will be:

Mode	Hex value	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
O/S,Auto,Man	0x98	1	0	0	1	1	0	0	0

Table 1.6 – Supported Mode Bitstring of the AI block

Scaling Parameters

The scaling parameter defines the operating range and the engineering units associated with a parameter. It also defines the number of digits to the right of the decimal point, which should be used by an interface device in displaying that parameter.

Scaling information is used for two purposes. Display devices need to know the range for bar graphs and trending, as well as the unit code. Control blocks need to know the range to use internally as percent of span, so that the tuning constants may remain dimensionless.

The AI Functional block uses the OUT_SCALE to convert the transducer value to the engineering unit utilized in the application.

The AO block uses the OUT_SCALE to convert the SP value to the engineering unit expected by the output transducer block that is also the engineering units of the READBACK value.

The following fields form the scale:

- Engineering Units at 100% of scale - The value that represents the upper end of range in engineering unit.
- Engineering Units at 0% of scale - The value that represents the lower end of range in engineering unit.
- Units Index – Code Index description for the Engineering Unit.
- Decimal Point - The number of digits to the right of the decimal point which should be used by an interface device in displaying the specified parameter.

Example Using Scale Parameter

The AO algorithm works internally with values in percent of span. Therefore the AO block converts the SP to percentage (PV_SCALE), it calculates the output in percentage, and then it converts to engineering unit of output (OUT_SCALE).

1. The AO takes the SP and converts to percentage of the PV_SCALE:

$$SP\% = \frac{(SP - EU\_0) * 100}{(EU\_100 - EU\_0)} \quad [PV\_SCALE]$$

PV_SCALE:
EU at 100% = 20
EU at 0% = 4
Units Index = mA
Decimal point = 2
SP = 15 mA

The values of SP in percentage are:

$$SP\% = \frac{(15 - 4) * 100}{(20 - 4)}$$

$$SP\% = 68.75\%$$

2. The output value is converted from percentage to engineering units of the OUT_SCALE:

$$OUT = \frac{SP\%}{100} * (EU\_100\% - EU\_0\%) + EU\_0\% \quad [OUT\_SCALE]$$

OUT_SCALE:
EU at 100% = 15
EU at 0% = 3
Units Index = psi
Decimal point = 2

The output value of this example is:

$$OUT = \frac{68.75}{100} * (15 - 3) + 3$$

$$OUT = 11.25 \text{ psi}$$

Fail Safe Handling

The Fail Safe is a special state that allows the functional block to do safe action when it has been detected an abnormal situation.

The abnormal situation occurs when there is an unusable input (bad sensor, for example) or the loss of the communication between functional blocks longer than a specified time (FSAFE_TIME).

When the condition that activated the Fault State is normalized, the Fault State is cleared and the block returns to the normal operation.

Conditions that activate the Fail Safe

When the Input or Output functional blocks detects an abnormal condition, the block goes to a fail safe. These abnormal situations are detected by different forms in inputs functional blocks and outputs functional blocks.

Input functional blocks (e.g., AI and TOT) are connected with an upstream block (transducer block) by the channel. When the transducer output has a bad status (for example, bad sensor) the fail safe condition of the functional block is activated.

Output functional blocks (e.g., AO block) receiving input values from the upstream blocks though "cyclic connection". These upstream blocks are control functional blocks and in generally are in a control device. In that case, the fail safe is activated when one of the following conditions are reached:

- Loss of communication to RCAS_IN for a time that exceeds FSAFE_TIME;
- Loss of communication to SP for a time that exceeds FSAFE_TIME;
- IFS status in the RCAS_IN input when the target mode is RCas;
- IFS status in the SP when the target mode is Auto;

Fail Safe Actions

The actions that the input or output blocks can make when the block is in Fail Safe can be chose by the user though the FSAFE_TYPE parameter in the AI and AO blocks, or using the FAIL_TOT parameter in the TOT block.

In the FSAFE_TYPE parameter the following options are available:

- Use FSAFE_VALUE – In this case, the AI and AO blocks use the safe value provides by the FSAFE_VALUE parameter to the calculation when the Fail Safe is active. The status of the output goes to "Uncertain, substitute value";
- Use Last Usable Value – In that case, the AI and AO blocks use the Last usable value to the algorithm calculation. The status will be "Uncertain Last Usable Value". If there is not any good value yet, use Initial Value to the output. The status will be "Uncertain, Initial Value";
- "Use the wrong value" (only to AI block) – The AI block uses the wrong and status values to the calculation;
- Use the ACTUATOR_ACTION (only to AO block) – The AO block goes to the safe position based in a discrete parameter ACTUATOR_ACTION into the transducer block.

In FAIL_TOT parameter (only using in totalizer functional block) the following options are available:

- Hold: Stop the totalization in the last value. The status of the output goes uncertain, non-specific".
- Memory: Use Last Usable Value to the totalization. The status will be "Uncertain Last Usable Value". If there is not a first good status in the memory yet, use Initial Value to the totalization. The status will be "Uncertain, Initial Value";
- Run: The totalization is continued. The wrong input value and status will be used for the output.

Diagnosis

In order to provide some information about the device to the control application and the human interface, there are some diagnosis parameters into the device.

The diagnosis parameters have a bitstring datatype and there is a mask parameter indicating which diagnosis is supported by the device.

The diagnosis bitstring and its supported diagnosis are showed in the "BitString Description".

Diagnosis of the Device Characteristics

In the Physical block the DIAGNOSIS parameter has the information about the “alerts” into the device (for instance, device not initialized, power up, factory init, hardware failure, etc). The DIAGNOSIS_MASK has the diagnosis supported by the device.

Diagnosis of the Actuator

In the AO block there is special output parameter to diagnosis purpose. The CHECKBACK parameter is a read-only bitstring parameter that has the resume of the main information about the functional block and the transducer block.

The CHECKBACK can be used for the configuration in a cyclic communication and the discrete information provided by the parameter can be used by a control device to detect some abnormal situations with the actuator.

Alarms and Events – Alert Processing

Alarms and events, known as alerts, represent state changes within functional block applications.

a) Alarm parameter (X_ALM parameter)

The alarm parameter is provided in a block to capture the dynamic information associated with an alarm. The following fields form the alarm parameter:

- Unacknowledged - Not Used.
- Alarm state - This field gives an indication of whether the alert is active or no. The Alarm State will have the following states:
    - 0 – Clear;
    - 1 – Active.

The alarm state is cleared when the block goes to out of service mode.

- Time stamp – Not Used.
- Subcode – Not Used.
- Value - The value of the associated parameter at the time the alert was detected.

b) Alarm limit (X_LIM parameter)

An analog alarm occurs when a value reaches or exceeds a limit. For a high alarm, an alarm is true when the analog value is greater than the limit. The status of the alarm remains true until the value drops below the limit minus the alarm hysteresis.

The alarm type can be disabled setting its respective alarm limit parameter to +/- INF, which is the default of all alarm limits.

The output parameter of the functional blocks that will be compared to alarm limit depends.

c) Alarm hysteresis (ALARM_HYS parameter)

Amount the OUT must return within the alarm limits before the alarm condition clears. In the AI block the Alarm Hysteresis is expressed as a percent of the OUT span (OUT_SCALE). In the TOT block the Hysteresis is expressed in engine unit.

d) Alert key (ALERT_KEY parameter)

It is an identification number of the plant unit. This information may be used in the host for sorting alarms, etc.

e) Alarm Summary (ALM_SUM parameter)

The parameter Alarm Summary summarizes the status of up to 16 process alarms of the same block. For each alarm, the current status, unacknowledged status, unreported status, and disabled status are maintained. It has four attributes:

- Current Alarms - the Active status of each alarm.
- Unacknowledged - the Unacknowledged status of each alarm.
- Unreported - the Unreported status of each alarm.
- Disabled - the Disabled status of each alarm.

f) Update Event

The Update Event is a special alarm that occurs when the static parameter is changed. When the static parameter is changed the bit correspondent in the ALARM_SUM is set and the output of the block goes to “Good (NC) Active Update Event” (if the actual status has lower priority than this). And the block stays in this state for 10 seconds. After that, the block returns to the normal situation (Output with the last status and the bit Update_Event in the ALARM_SUM parameter clear).

Data Type and Data Structure Definition

In this section are defined every data structure and data types used in the system.

Code Data Type	Data Type	Size	Description
1	Boolean	1	True or false
2	Integer8	1	
3	Integer16	2	
4	Integer32	4	
5	Unsigned8	1	
6	Unsigned16	2	
7	Unsigned32	4	
8	FloatingPoint	4	
9	VisibleString	1,2,3,..	they are one byte per character, and include the 7 bit ASCII character set.
10	OctetString	1,2,3,…	Octet strings are binary
-	Date	-	
-	TimeofDay	-	
-	TimeDifference	-	
-	BitString	-	
-	DataTimeValue	-	

Table 1.7 – Data Type and Data Structure Definition

Block Object – DS-32

This data structure consists of the attributes of a block.

E	Element Name	Data Type	Size
1	Reserved	Unsigned8	1
2	Block Object	Unsigned8	1
3	Parent Class	Unsigned8	1
4	Class	Unsigned8	1
5	DD REFERENCE	Unsigned32	4
6	DD REVISION	Unsigned16	2
7	Profile	OctetString	2
8	Profile Revision	Unsigned16	2
9	Execution Time	Unsigned8	1
10	Number_of_Parameters	Unsigned16	2
11	ADDRESS OF VIEW_1	Unsigned16	2
12	Number of Views	Unsigned8	1

Table 1.8 – Block Object DS-32

Value & Status - Floating Point Structure – DS-33

This data structure consists of the value and status of floating point parameters that are Inputs or Outputs.

E	Element Name	Data Type	Size
1	Value	Float	4
2	Status	Unsigned8	1

Table 1.9 – Floating Point Structure DS-33

Value & Status - Discrete Structure – DS-34

This data structure consists of the value and status of discrete value parameters.

E	Element Name	Data Type	Size
1	Value	Unsigned8	1
2	Status	Unsigned8	1

Table 1.10 – Discrete Structure DS-34

Scaling Structure – DS-36

This data structure consists of the static data used to scale floating point values for display purposes.

E	Element Name	Data Type	Size
1	EU at 100%	Float	4
2	EU at 0%	Float	4
3	Units Index	Unsigned16	2
4	Decimal Point	Integer8	1

Table 1.11 – Scaling Structure DS-36

Mode Structure – DS-37

This data structure consists of bit strings for actual, permitted, and normal modes.

E	Element Name	Data Type	Size
1	Actual	Bitstring	1
2	Permitted	Bitstring	1
3	Normal	Bitstring	1

Table 1.12 – Mode Structure DS-37

Alarm Float Structure – DS-39

This data structure consists of data that describes floating point alarms.

E	Element Name	Data Type	Size
1	Unacknowledged	Unsigned8	1
2	Alarm State	Unsigned8	1
3	Time Stamp	Time Value	8
4	Subcode	Unsigned16	2
5	Value	Float	4

Table 1.13 – Alarm Float Structure DS-39

Alarm Discrete Structure – DS-40

This data structure consists of data that describes discrete alarms.

E	Element Name	Data Type	Size
1	Unacknowledged	Unsigned8	1
2	Alarm State	Unsigned8	1
3	Time Stamp	Time Value	8
4	Subcode	Unsigned16	2
5	Value	Unsigned8	1

Table 1.14 – Alarm Discrete Structure DS-40

Alarm Update Structure – DS-41

This data structure consists of data that describes a static revision alarm.

E	Element Name	Data Type	Size
1	Unacknowledged	Unsigned8	1
2	Update State	Unsigned8	1
3	Time Stamp	Time Value	8
4	Subcode	Unsigned16	2
5	Relative Index	Unsigned16	2

Table 1.15 – Alarm Update Structure DS-41

Alarm Summary Structure – DS-42

This data structure consists of data that summarizes 16 alerts.

E	Element Name	Data Type	Size
1	Current	Bit String	2
2	Unacknowledged	Bit String	2
3	Unreported	Bit String	2
4	Disabled	Bit String	2

Table 1.16 – Alarm Summary Structure DS-42

Simulate - Floating Point Structure – DS-50

This data structure consists of a simulate and transducer floating point value and status and a simulate enable/disable discrete.

E	Element Name	Data Type	Size
1	Simulate Status	Unsigned8	1
2	Simulate Value	Float	4
3	Simule En/Disable	Unsigned8	1

Table 1.17 – Simulate - Floating Point Structure DS-50

Simulate - Discrete Structure – DS-51

This data structure consists of a simulate and transducer discrete value and status and a simulate enable/disable discrete.

E	Element Name	Data Type	Size
1	Simulate Status	Unsigned8	1
2	Simulate Value	Unsigned8	1
5	Simule En/Disable	Unsigned8	1

Table 1.18 – Simulate - Discrete Structure DS-51

Batch Structure – DS-67

This data structure contains the structure of the Batch parameter.

E	Element Name	DataType	Size
1	BATCH_ID	Unsigned32	4
2	RUP	Unsigned16	2
3	OPERATION	Unsigned16	2
4	PHASE	Unsigned16	2

Table 1.17 – Batch Structure DS-67

BLOCK LIBRARY

The table shows the block description:

BLOCK	DESCRIPTION
RES	PHYSICAL – This block contains data that is specific to the hardware that is associated with the resource.
AI	ANALOG INPUT – This block takes the input data from the transducer block and makes it available to other functional blocks. It has scaling conversion, filtering and fail safe mechanism.
AO	ANALOG OUTPUT – The AO block provides a value to an output transducer block. It provides value, scaling conversion, fail safe mechanism and other features.
TOT	TOTALIZER – It integrates a variable in function of the time. It has scaling conversion and fail safe mechanism.

Table 2.1 – Block Library

PHY – Physical Block

Description

This block contains data that is specific to the hardware that is associated with the resource. All data is modeled as Contained, so there are no outputs or inputs to this block. The data is not processed in the way that a functional block processes data, so there is no function schematic.

This parameter set is intended to be the minimum required for the functional block Application associated with the resource in which it resides. Some parameters that could be in the set, e.g. calibration data and ambient temperature, are more part of their respective transducer blocks.

FACTORY_RESET Parameter

This parameter allows degrees of initialization of the resource. They are:

Option	Name	Description
1	Restart with defaults	It is intended to clear the configuration memory. It works like a factory initialization.
2506	Restart processor	It provides a way to hit the reset button on the processor associated with the resource.
2712	Restart bus address	It initializes the device address to the default value, 126

Table 2.2 – FACTORY_RESET Parameter

Non-volatile memory

The Smar devices do not support cyclic saving of non-volatile parameters to a non-volatile memory. On the other hand, the Smar devices have a mechanism to save non-volatile parameters into a non-volatile memory during the power down, and they will be recovered in the power up.

Write lock by software

If WRITE_LOCKING parameter is “locked”, will prevent any external change to the data base in the Functional block Application of the resource. Block connections and calculation results will proceed normally, but the configuration will be locked. If it is set “Write Unlocked” the write is allowed again.

Diagnosis

The DIAGNOSIS parameter has a checkup of main points of the resource to the device work properly.

Identifier Number Selector

The IDENT_NUMBER_SELECTOR parameter allows the user select a different identifier number if the resource will permit it.

If a device is switched to the profile Ident_Number, the device shall interact with the profile features of the GSD file.

Supported Modes

AUTO.

Parameters

Idx	Parameter	DataType (length)	Valid Range/ Options	Default Value	Units	Store / Mode	Description
1	ST_REV	Unsigned16		0	None	S/RO	It will be incremented each time that occur a change in a static parameter in the physical block.
2	TAG_DESC	OctString(32)		Spaces	Na	S	Tag Name of the Block. This parameter must be unique in the configuration.
3	STRATEGY	Unsigned16		0	None	S	It is a user supplied value to identify a configuration.
4	ALERT_KEY	Unsigned8	1 to 255	0	None	S	
5	TARGET_MODE	Unsigned8	AUTO	AUTO	None	S	It contains the desired mode to the block.
6	MODE_BLK	DS-37			Na	D/RO	See Mode Parameter
7	ALARM_SUM	DS-42		0,0,0,0	None	D/RO	It contains the current states of the block alarms.
8	SOFTWARE_REVISION	Visiblestring(16)	Set by manufacturer		None	S/RO	Revision-number of the software of the field device.
9	HARDWARE_REVISION	Visiblestring(16)	Set by manufacturer		None	S/RO	Revision- number of the hardware associated with the resource.
10	DEVICE_MAN_ID	Unsigned16	Set by manufacturer		None	S/RO	Manufacturer identification number.
11	DEVICE_ID	Visiblestring(16)	Set by manufacturer		None	S/RO	Manufacturer's model number associated with the resource.
12	DEV_SER_NUM	Visiblestring(16)	Set by manufacturer		None	S/RO	Hardware serial-number of the field device.
13	DIAGNOSIS	Octetstring(4)			None	D/RO	Bitstring indicating the diagnosis of the device. See Diagnosis item.
14	DIAGNOSIS_EXTENSION	Octetstring(6)			None	D/RO	Not used.
15	DIAGNOSIS_MASK	Octetstring(4)			None	D/RO	Bitstring indicating with kind of diagnosis the device support.
16	DIAGNOSIS_MASK_EXTENSION	Octetstring(6)			None	D/RO	Not used.
17	DEVICE_CERTIFICATION	Visiblestring(32)			Na	S/RO	Certifications of the field devices.
18	WRITE_LOCKING	Unsigned16	0: Write locked 2457: Write Unlocked	2457	Na	S	If Locked, no writes from anywhere are allowed, except to clear WRITE_LOCK. Cyclic block inputs will continue to be updated.

Idx	Parameter	DataType (length)	Valid Range/ Options	Default Value	Units	Store / Mode	Description
19	FACTORY_RESET	Unsigned16	1: Restart with default 2506: Restart processor 2712: Recover default address to the device	0	Na	S	Allows a manual restart of the device. Several degrees of restart are possible.
20	DESCRIPTOR	Octetstring(32)				S	It is a user supplied description of the block in the application.
21	DEVICE_MESSAGE	Octetstring(32)			None	S	It is a user supplied Message of the block in the application.
22	DEVICE_INSTALL_DATE	Octetstring(16)				S	Date of the device installation.
23	LOCAL_OP_ENA	Unsigned8		1	None	N	Not Used.
24	IDENT_NUMBER_SELECTOR	Unsigned8	0: Profile specific Ident_Num 1: Mnf specific Ident_Number 2: Mnf specific Ident_Number of V2.0 3: Ident_Number of Multi_Variable device			S	It permits the user change the IDENT_NUMBER of the device.
25	HW_WRITE_PROTECTION	Unsigned8			None	D/RO	Not Used.
Legend: E – Enumerated parameter; na – Dimensionless parameter; RO – Read only; D – dynamic; N – non-volatile; S - static							

Table 2.3 – Physical Block Parameters

AI – Analog Input

Overview

The Analog Input block takes the input data from the Transducer block, selected by channel number, and makes it available to other functional blocks at its output.

Schematic

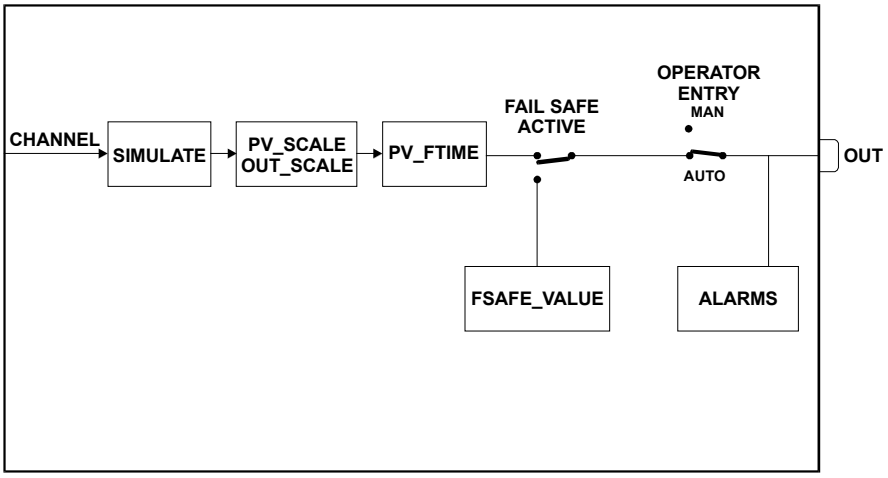

Figure 2.1 – Analog Input Block

Description

The transducer block provides the PV unit to Analog Input, and when the PV unit is changed into the transducer, the PV_SCALE unit is converted.

Optionally, a filter may be applied in the process value signal, whose time constant is PV_FTIME. Considering a step change to the input, this is the time in seconds to the PV reaches 63.2% of the final value. If the PV_FTIME value is zero, the filter is disabled.

Simulate

The SIMULATE parameter is used for the diagnostics and checkout purposes. When it is active, the transducer value and status will be overridden by the simulate value and status.

The SIMULATE structure is composed by the following attributes:
- Simulate Value and Status
- Simulate Enable

When simulation is enable, the Transducer Input parameter will be calculated based on the attribute Simulate Value/Status of the SIMULATE parameter. Otherwise it will be that one supplied by the transducer block.

Supported Modes

O/S, MAN and AUTO.

Status Handling

The AI block does not support cascade path. Then, the output status has not a cascade sub-status.

Cyclic CFG_DATA

Configuration supported	Short config (Identifier Byte)	Long Config (Extended Identifier format)
OUT	0x94	0x42,0x84,0x08,0x05

Table 2.4 – Cyclic CFG_DATA

Parameters

Idx	Parameter	DataType (length)	Valid Range/ Options	Default Value	Units	Store / Mode	Description
1	ST_REV	Unsigned16		0	None	S/RO	It will be incremented each time that occur a change in a static parameter in that block.
2	TAG_DESC	OctString(32)		Spaces	Na	S	Tag Name of the Block. This parameter must be unique in the configuration.
3	STRATEGY	Unsigned16		0	None	S	It is a user supplied value to identify a configuration.
4	ALERT_KEY	Unsigned8	1 to 255	0	None	S	
5	TARGET_MODE	Unsigned8	O/S, MAN and AUTO.	AUTO	None	S	It contains the desired mode to the block.
6	MODE_BLK	DS-37			Na	D/RO	See Mode Parameter
7	ALARM_SUM	DS-42		0,0,0,0	None	D/ RO	It contains the current states of the block alarms.
8	BATCH	DS-67		0,0,0,0	None	S	It is intended to use in a distributed fieldbus system to identify used and available channels. There is no algorithm related with it.
10	OUT	DS-33	OUT_SCALE		OUT	D / Man	The analog value calculated as a result of executing the function.
11	PV_SCALE	2 Floats		100,0	Trd output	S	The high and low scale values, to transducer for a specified channel.
12	OUT_SCALE	DS-36		100,0,-,-	OUT	S	The high and low scale values to the OUT parameter.
13	LIN_TYPE	Unsigned8		0	None	S	Not Used.

Idx	Parameter	DataType (length)	Valid Range/ Options	Default Value	Units	Store / Mode	Description
14	CHANNEL	Unsigned16		0	None	S	The number of the logical hardware channel to the transducer that is connected to this I/O block.
16	PV_FTIME	Float	Non-Negative	0	Sec	S	Time constant of a single exponential filter for the PV, in seconds.
17	FSAFE_TYPE	Unsigned8	0:Use FSAFE_VALUE 1:Use Last Usable Value 2:Use Wrong Value	1	E	S	Define the reaction of the device in Fail Safe condition.
18	FSAFE_VALUE	Float	OUT_SCALE	0	OUT	S	Preset Value to set the Output when the Fail Safe is active.
19	ALARM_HYS	Float	0 to 50 %	0.5%	%	S	Alarm hysteresis parameter. In order to clear the alarm the amount the PV must return within the alarm limit plus hysteresis.
21	HI_HI_LIM	Float	OUT_SCALE, +INF	INF	OUT	S	The setting for high high alarm in engineering units.
23	HI_LIM	Float	OUT_SCALE, +INF	INF	OUT	S	The setting for high alarm in engineering units.
25	LO_LIM	Float	OUT_SCALE, -INF	INF	OUT	S	The setting for low alarm in engineering units.
27	LO_LO_LIM	Float	OUT_SCALE, -INF	INF	OUT	S	The setting for low low alarm in engineering units.
30	HI_HI_ALM	DS-39				D	The status for high high alarm.
31	HI_ALM	DS-39				D	The status for high alarm.
32	LO_ALM	DS-39				D	The status for low alarm.
33	LO_LO_ALM	DS-39				D	The status for low low alarm.
34	SIMULATE	DS-50	0: Disable ; ≠0: Active are the Enable options.	Disable		S	Allows the transducer value to be manually supplied when simulate is enabled.
35	OUT_UNIT_TEXT	OctString(16)			None	S	It is used when the specific unit of OUT is not in the code list. Then the user can supply a textual unit definition.
Legend: E – Enumerated parameter; na – Dimensionless parameter; RO – Read only; D – dynamic; N – non-volatile; S – static							

Table 2.5 - Analog Input Parameters

AO – Analog Output

Overview

The Analog Output Block is a functional block used by devices that work as output elements in a control loop, e.g. valves, actuators, positioners, etc. The AO block receives a signal from another functional block and passes its results to an output transducer block through an internal channel reference.

Schematic

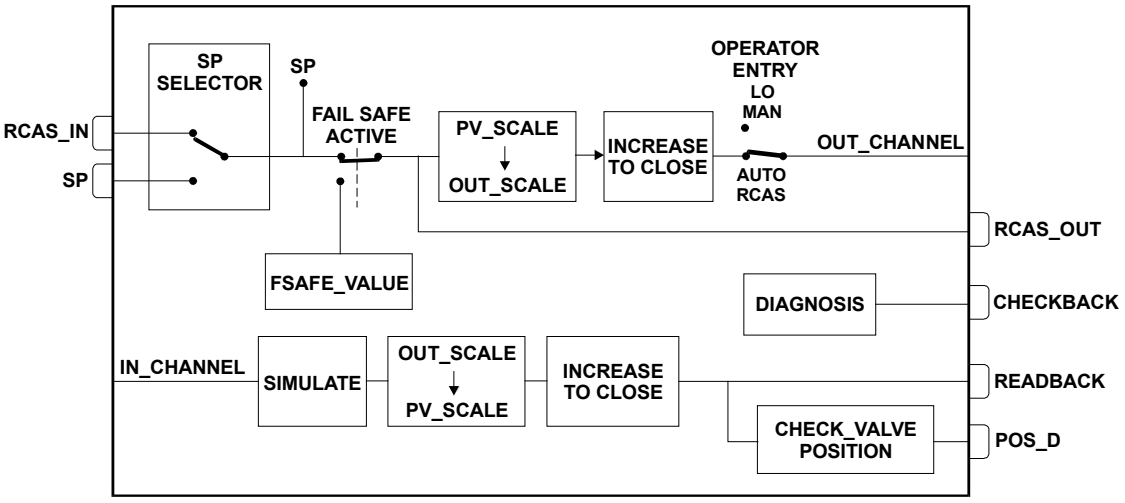

Figure 2.2 – Analog Output Block

Description

Treatment of Input Values

The SP value may be controlled automatically through a remote cascade control or manually by an operator. The PV_SCALE is used to do the scaling conversion of the SP.

$$SP\% = \frac{SP}{(EU\_100\% - EU\_0\%) + EU\_0\%}[PV\_SCALE]$$

Treatment of Output Values

The output scaling (OUT_SCALE) is used to convert percent of span to the number used by the transducer. This allows portions of the SP span to cause full span movement of the output.

$$OUT = SP\% * (EU\_100\% - EU\_0\%) + EU\_0\%[OUT\_SCALE]$$

Increase To Close

The INCREASE_CLOSE parameter allows the output to be inverted relative to the span of the input value. For example, if the SP is 100. (PV_SCALE=0-100%; OUT_SCALE = 3-15Psi):

- If the INCREASE_CLOSE = 0 (“Rising”), SP converted to OUT_SCALE will be15 psi. Therefore, the actuator type will be “air to open”.
- If the INCREASE_CLOSE = 1 (“Falling”), SP converted to OUT_SCALE will be 3 psi. Therefore, the actuator type will be “air to close”.

Simulate

The SIMULATE parameter is used for the diagnostics and checkout purposes. When it is active, the transducer value and status will be overridden by the simulate value and status.

The SIMULATE structure is composed by the following attributes:

- Simulate Value and Status
- Simulate Enable

Readback parameters

The Readback value from the transducer block is composed for two parameters: READBACK and POS_D.
The READBACK is the analog return from the transducer, the valve position for example. The POS_D is a discrete state: open, close, or intermediate position.

If the hardware supports a Readback value, such as valve position, then the value will be read by the transducer block. If not supported, the Transducer Value/Status is generated from AO.OUT by the transducer block.

When simulation is active, the Readback values and status will be calculated based on the attribute Simulate Value/Status of the SIMULATE parameter. Otherwise, it will be that one supplied by the transducer block.

Supported Modes

O/S, LO, MAN, AUTO, and RCAS.

Cyclic CFG_DATA

Configuration supported	Short config (Identifier Byte)	Long Config (Extended Identifier format)
RCASIN / RCASOUT	0xB4	0xC4,0x84,0x84,0x08,0x05,0x08,0x05
RCASIN / RCASOUT / CHECKBACK	0x97,0xA4	0xC5,0x84,0x87,0x08,0x05,0x08,0x05,0x0A
SP	0xA4	0x82,0x84,0x08,0x05
SP / CHECKBACK	0x92,0xA4	0xC3,0x84,0x82,0x08,0x05,0x0A
SP / READBACK / POSD	0x96,0XA4	0xC6,0x84,0x86,0x08,0x05,0x08,0x05,0x05,0x05
SP / READBACK / POSD / CHECKBACK	0x99,0xA4	0xC7,0x84,0x89,0x08,0x05,0x08,0x05,0x05,0x05,0x0A
SP / READBACK / RCASIN / RCASOUT/ POSD / CHECKBACK	0x9E,0xA9	0xCB,0x89,0x8E,0x08,0x05,0x08,0x05,0x08,0x05,0x08,0x05,0x05,0x05,0x0A

Table 2.6 – Cyclic CFG_DATA

Parameters

Idx	Parameter	Data Type (length)	Valid Range/ Options	Default Value	Units	Store / Mode	Description
1	ST_REV	Unsigned16		0	None	S/RO	It will be incremented each time that occur a change in a static parameter in that block.
2	TAG_DESC	OctString(32)		Spaces	Na	S	Tag Name of the Block. This parameter must be unique in the configuration.
3	STRATEGY	Unsigned16		0	None	S	It is a user supplied value to identify a configuration.
4	ALERT_KEY	Unsigned8	1 to 255	0	None	S	
5	TARGET_MODE	Unsigned8	O/S, LO, MAN, AUTO, and RCAS.	AUTO	None	S	It contains the desired mode to the block.
6	MODE_BLK	DS-37			Na	D/RO	See Mode Parameter
7	ALARM_SUM	DS-42		0,0,0,0	None	D/RO	It contains the current states of the block alarms.
8	BATCH	DS-67		0,0,0,0	None	S	It is intended to use in a distributed fieldbus system to identify used and available channels. There is no algorithm related with it.
9	SP	DS-33	PV_SCALE		PV	N/Auto	The analog set point. Can be set manually, automatically through the interface device or another field device.
11	PV_SCALE	DS-36		0-100%	PV	S	The high and low scale values to the SP parameter.
12	READBACK	DS-33	PV_SCALE		PV	D/RO	Indicate the readback of the actual position of the transducer.

Idx	Parameter	Data Type (length)	Valid Range/ Options	Default Value	Units	Store / Mode	Description
14	RCAS_IN	DS-33			PV	D	Remote setpoint value and status provided by a supervisory Host to an analog control or output block.
21	IN_CHANNEL	Unsigned16		0	None	S	The number of the logical hardware channel **FROM** the transducer that is connected to this I/O block. See channel handling.
22	OUT_CHANNEL	Unsigned16		0	None	S	The number of the logical hardware channel **TO** the transducer that is connected to this I/O block. See channel handling.
23	FSAFE_TIME	Float	Positive	0	Sec	S	Time from the detection of failure to the action of the block, if there is a failure condition.
24	FSAFE_TYPE	Unsigned8	0:Use FSAFE_VALUE 1:Use Last Usable Value 2:Goes to ACTUATOR_ACTION position	1	E	S	Define the reaction of the device in a Fail Safe condition.
25	FSAFE_VALUE	Float	OUT_SCALE	0	OUT	S	Preset Value to set the Output when the Fail Safe is active and the FSAFE_TYPE = 0.
27	RCAS_OUT	DS-33			PV	D/RO	The value and status required by an upper block. So the upper block may prevent reset windup and provide bumpless transfer to closed loop control.
31	POS_D	DS-34	0: not initialized 1: closed 2: opened 3: intermediate	0	E	D/RO	The current position of the valve.
32	SETP_DEVIATION	Float		0	PV	D/RO	Difference between SP and Readback.
33	CHECK_BACK	OctString(3)		0	Bitwise	D/RO	Information of the current device state. See Check Back Options.
34	CHECK_BACK_MASK	OctString(3)		0	Bitwise	S/RO	Supported Check Back information.
35	SIMULATE	DS-50	0: Disable ; ≠0: Active are the Enable options.	Disable		S	Allows the readback value to be manually supplied when simulate is enabled. In this case, the simulate value and status will be the PV value.
36	INCREASE_CLOSE	Unsigned8	0: Rising 1: Falling	0	E	S	Direction of positioner in automatic mode.
37	OUT	DS-33	OUT_SCALE		OUT	N/Man	The output value result to the transducer block.
38	OUT_SCALE	DS-36		0-100%	Trd input	S	The high and low scale values, to transducer for a specified channel.
Legend: E – Enumerated parameter; Na – Dimensionless parameter; RO – Read only; D – dynamic; N – non-volatile; S - static							

Table 2.7 – Analog Output Parameters

TOT - Totalizer

Overview

The Totalizer functional block takes the input data from the Transducer block, selected by channel number, and integrates over the time. This block is normally used to totalize flow, giving total mass or volume over a certain time, or totalize power, giving the total energy.

Schematic

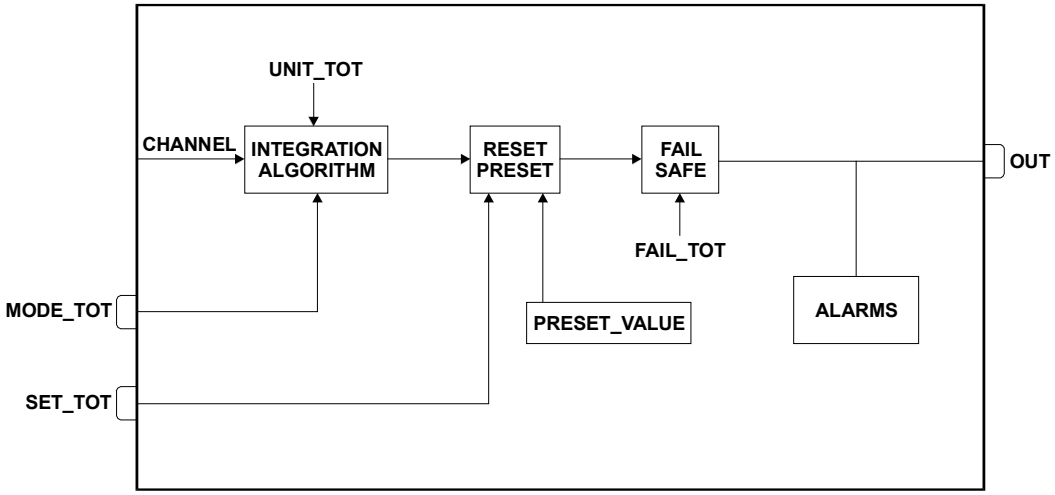

Figure 2.3 – Totalizer Block

Description

The totalizer functional block integrates a variable (e.g. flow rate or power) in function of the time to the corresponding quantity (e.g., volume, mass or distance).

The rate unit of the Totalizer is providing by the transducer block. Internally, the time units are converted in rate units per second. Each rate, multiplied by the block execution time, gives the mass, volume or energy increment per block execution.

The TOTAL is the totalized quantity. The engineering unit used in the output is the UNIT_TOT. The unit of the output must be compatible with the unit of the input provided by the transducer by the channel. Then, if the input the rate is mass flow (e.g. Kg/s, g/min, ton/h) the unit of the output must be mass (e.g. kg, g, ton, lb, etc.).

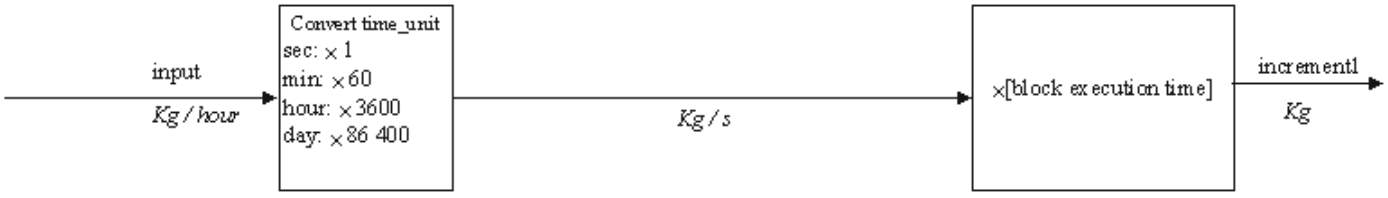

Figure 2.4 – Increment Calculation with Rate Input

Flow Totalization

In order to distinguish forward and reverse flows, the totalizer block considers a negative sign as an indication of reverse flow.

The net flow is obtained by adding the two increments. The net increment will have a positive or negative signal to indicate the net flow direction. In order to integrate the difference between the inflow and outflow of a tank, for example, the second one can be assigned to be negative. The net flow direction to be considered in the totalization is defined in MODE_TOT. The following options are available:

- Positive only – only positive flows are totalized. The negative values will be treated as zero.
- Negative only – only negative flows are totalized. The positive values will be treated as zero.
- Balanced – both positive and negative values will be totalized.
- Hold – the value totalized will be constant.

Reset e Preset

The special parameter SET_TOT is used to reset or preset the integration. This parameter is level sensitive. When the SET_TOT is “Reset” the output will be zero. While this parameter has the “Reset” value, the block will be old reset. When the SET_TOT is “Preset”, the output receives the value of the PRESET_TOT parameter, and while the value is “Preset” the output remains the same. Therefore, it will start to integrate only after it goes to “Totalize” again.

Starting the Block

In order to start the Totalizer block to working properly the following steps must be attended:

- Channel parameter must be set to the PV;
- Primary_Value_Type of the Transducer Block must be set to “Flow”;
- Linearization_Type of the Transducer Block must be set to the “Square Root”;
- Primary_Value_Unit of the Transducer Block must be set to a valid flow unit;
- UNIT_TOT of the Totalizer block must be set to a corresponding mass or volume unit equivalent of the flow input unit;

Supported Modes

O/S, AUTO

Cyclic CFG_DATA

Configuration supported	Long Config (Extended Identifier format)
TOTAL	0x41,0x84,0x85
TOTAL / SETTOT	0xC1,0x80,0x84,0x85
TOTAL / SETTOT / MODETOT	0xC1,0x81,0x84,0X85

Table 2.8 – Cyclic CFG_DATA

Parameters

Idx	Parameter	Data Type (length)	Valid Range/ Options	Default Value	Units	Store / Mode	Description
1	ST_REV	Unsigned16		0	None	S/RO	It will be incremented each time that occur a change in a static parameter in that block.
2	TAG_DESC	OctString(32)		Spaces	Na	S	Tag Name of the Block. This parameter must be unique in the configuration.
3	STRATEGY	Unsigned16		0	None	S	It is a user supplied value to identify a configuration.
4	ALERT_KEY	Unsigned8	1 to 255	0	None	S	
5	TARGET_MODE	Unsigned8	O/S, AUTO.	AUTO	None	S	It contains the desired mode to the block.
6	MODE_BLK	DS-37			Na	D/RO	See Mode Parameter
7	ALARM_SUM	DS-42		0,0,0,0	None	D/RO	It contains the current states of the block alarms.
8	BATCH	DS-67		0,0,0,0	None	S	It is intended to use in a distributed fieldbus system to identify used and available channels. There is no algorithm related with it.
10	TOTAL	DS-33			OUT	N/RO	The primary analog value calculated as a result of executing the function. It is the result of integration.

Idx	Parameter	Data Type (length)	Valid Range/ Options	Default Value	Units	Store / Mode	Description
11	UNIT_TOT	Unsigned16			OUT	S	The engineering unit of the output.
12	CHANNEL	Unsigned16		0	None	S	The number of the logical hardware channel to the transducer that is connected to this I/O block.
13	SET_TOT	Unsigned8	0: Totalize 1: Reset 2: Preset	Totalize	E	N	Resets the totalizer output or set the output to PRESET_TOT. It is level sensitive.
14	MODE_TOT	Unsigned8	0: Balanced 1: Positive only 2: Negative only 3: Hold	Balanced	E	N	Defines the type of counting (positive or negative or hold the last value) of the integration.
15	FAIL_TOT	Unsigned8	0: Run 1: Hold 2: Memory	Run	E	S	Defines the procedure to have in fail safe condition.
16	PRESET_TOT	Float		0	OUT	S	Value of the output when the SET_TOT is Preset.
17	ALARM_HYS	Float		0	OUT	S	Alarm hysteresis parameter. In order to clear the alarm the amount the PV must return within the alarm limit plus hysteresis.
18	HI_HI_LIM	Float		INF	OUT	S	The setting for high high alarm in engineering units.
19	HI_LIM	Float		INF	OUT	S	The setting for high alarm in engineering units.
20	LO_LIM	Float		INF	OUT	S	The setting for low alarm in engineering units.
21	LO_LO_LIM	Float		INF	OUT	S	The setting for low low alarm in engineering units.
22	HI_HI_ALM	DS-39				D	The status for high high alarm.
23	HI_ALM	DS-39				D	The status for high alarm.
24	LO_ALM	DS-39				D	The status for low alarm.
25	LO_LO_ALM	DS-39				D	The status for low low alarm.
Legend: E – Enumerated parameter; Na – Dimensionless parameter; RO – Read only; D – dynamic; N – non-volatile; S - static							

Table 2.9 – Totalizer Parameters

DO - Discrete Outputs

Overview

Discrete Output Function Block represents e.g. discrete valves, relay outputs, transistor outputs. etc

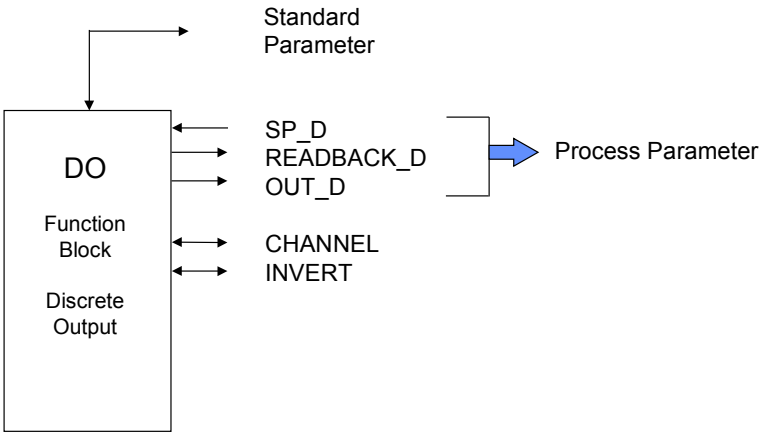

Figure 2.5 - Summary of the Parameters of Discrete Output Function Block

The DI schematic is shown in the Figure 2.6.

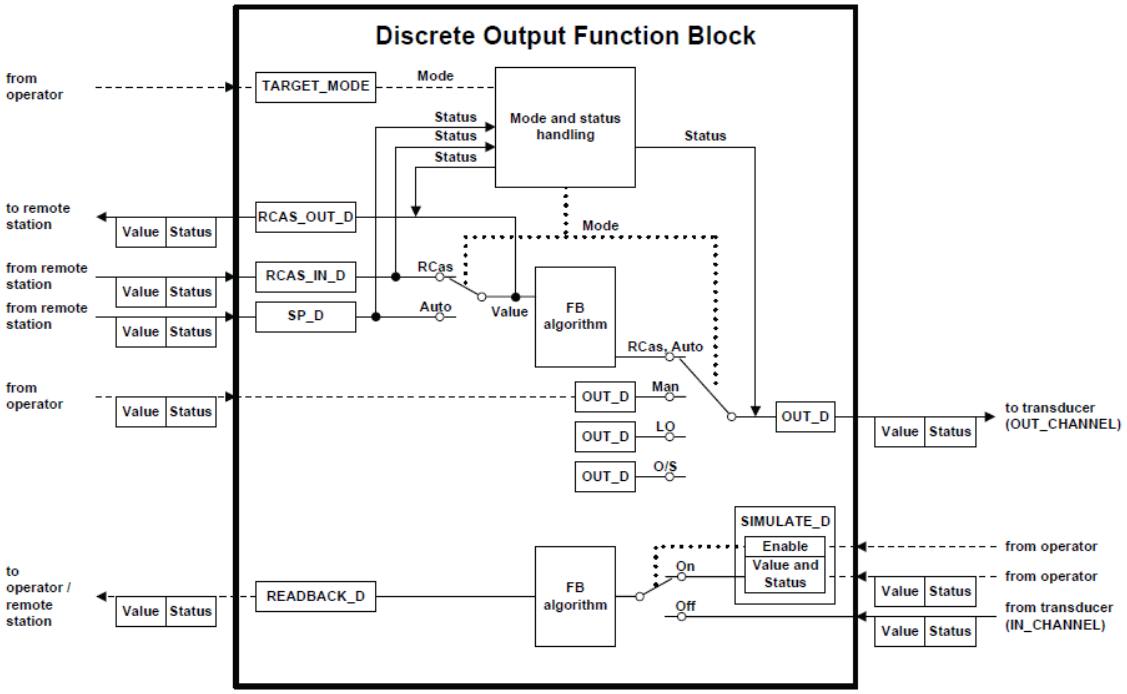

Figure 2.6 - Schematic of the Discrete Output Function Block

Description

The Transducer Block receives de OUTD_D value from DO Block. The Target Mode is set by the operator according to permitted Mode Block. The DO Block receives the setpoint discrete value from the Host according to the Mode Block.

Simulate

The SIMULATE_D parameter is used for the diagnostics and checkout purposes. When it is active, the readback value and status will be overridden by the simulate value and status. The SIMULATE_D structure is composed by the following attributes:

- Simulate Value and Status;
- Simulate Enable.

When simulation is enabled, the Transducer Block and de DO Block will be disconnected.

Supported Modes

O/S, MAN, RCAS, LO and AUTO.

Status Handling

The DO block does support cascade path. Then, the output status has a cascade sub-status.

Cyclic CFG_DATA

Function Block	Parameter	Identifier Byte	Extended Identifier format
DO block	SP_D / READBACK_D		0xC1,0x81,0x81,0x83
	SP_D / CHECK_BACK_D		0xC1,0x81,0x82,0x92
	SP_D / READBACK_D / CHECK_BACK_D		0xC1,0x81,0x84,0x93
	RCAS_IN_D / RCAS_OUT_D		0xC1,0x81,0x81,0x8C
	RCAS_IN_D / RCAS_OUT_D / CHECK_BACK_D		0xC1,0x81,0x84,0x9C
	SP_D / READBACK_D / RCAS_IN_D / RCAS_OUT_D / CHECK_BACK_D		0xC1,0x83,0x86,0x9F

Table 2.10 - Cyclic CFG_DATA

Discrete Output Function Block Parameter Descriptions

Parameter	Description
IN_CHANNEL	Reference to the active Transducer Block and its parameter which provides the discrete value for the final control element.
CHECKBACK	Detailed information of the device, bitwize coded. More than one message possible at once.
CHECK_BACK_MASK	Definition of supported CHECK_BACK information bits 0 = not supported 1 = supported
FSAFE_TIME	Time in seconds from detection of failure of the actual used setpoint (SP_D = bad or RCAS_IN <> Good) to the action of the block if the condition still exists.
FSAFE_TYPE	Defines reaction of the device, if failure of actual used setpoint is still detected after FSFVE_TIME or if the status of actual used setpoint is Initiate Fail Safe. The calculated ACTUAL MODE is AUTO respectively. 0 = value FSAVE_VALUE is used as setpoint status of OUT_D = UNCERTAIN - Substitute Value 1 = storing last valid setpoint status of OUT_D = UNCERTAIN - Last usable Value or BAD - No communication, no LUV 2 = actuator goes to fail-safe position defined by ACTUATOR_ACTION, status of OUT_D = BAD - non specific
FSAFE_VAL_D	OUT_D used if FSAFE_TYPE = 0 and FSAFE is activated.
INVERT	Indicates whether the SP_D should be logically inverted before writing to OUT_D in mode AUTO or RCAS. List of valid values : 0 = not inverted 1 = invert

Parameter	Description
OUT_D	This parameter is the process variable of the discrete output block in AUTO, and RCas mode and is the value specified by the operator/engineer in MAN and LO.
READBACK_D	Can be the actual position of the final control element and its sensors.
RCAS_IN_D	Target Setpoint and status provided by a supervisory Host to the discrete output block used in MODE RCAS.
RCAS_OUT_D	Function Block Setpoint and status provided to a supervisory Host for monitoring / back calculation and to allow action to be taken under limited conditions or mode change.
SIMULATE	For commissioning and maintenance reasons, it is possible to simulate the READBACK by defining the value and the status. That means that the Transducer Block and the DO-FB will be disconnected.
SP_D	Setpoint of function block used in MODE AUTO.

Table 2.11 - Discrete Output Function Block Parameter Description

Discrete Output Function Block Parameter Attributes

The following table presents an overview of all parameters and their attributes of Discrete Output Function Blocks defined in Class B.

Relative Index	Parameter Name	Object type	Data type	Store	Size	Access	Param. Usage/ Kind of transport	Default values
Function Block Parameter for Discrete Output DO								
9	SP_D	Record	DS-34	D	2	r,w	I/a, cyc	-
10	OUT_D	Record	DS-34	D	2	r,w	C/a	-
12	READBACK_D	Record	DS-34	D	2	r	O/a, cyc	-
14	RCAS_IN_D	Record	DS-34	D	2	r,w	I/a, cyc	-
17	CHANNEL	Simple	Unsigned16	S	2	r,w	C/a	-
18	INVERT	Simple	Unsigned 8	S	1	r,w	C/a	0
19	FSAVE_TIME	Simple	Float	S	4	r,w	C/a	0
20	FSAVE_TYPE	Simple	Unsigned 8	S	1	r,w	C/a	2
21	FSAVE_VAL_D	Simple	Unsigned 8	S	1	r,w	C/a	0
22	RCAS_OUT_D	Record	DS-34	D	2	r	O/a, cyc	-
24	SIMULATE	Record	DS-51	S	3	r,w	C/a	disable
33	CHECK_BACK	Simple	OctetString	D	3	r	C/a, cyc	-
34	CHECK_BACK_ MASK	Simple	OctetString	Cst	3	r	C/a	-
35-44	reserved by PNO							
45	first manufacture specific parameter							

Table 2.12 - Parameter attributes of the Discrete Output Function Block

DI - Discrete Input

Overview

Discrete Input Function Blocks represent e.g. inductive, optical, capacitive, ultrasonic, proximity switches.

The Discrete Input block takes the input discrete data from the Transducer block, selected by channel number, and makes it available to other functional blocks at its output.

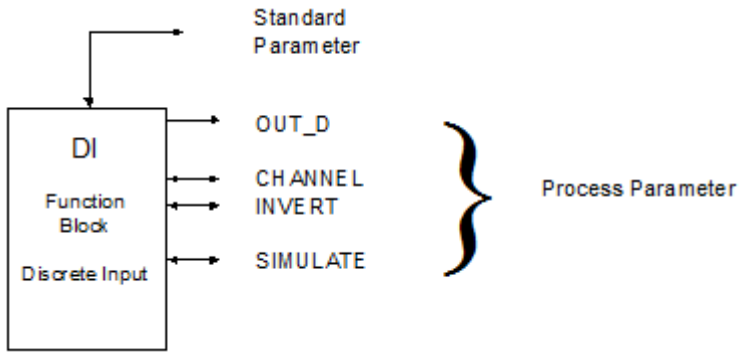

Figure 2.7 - Summary of the Parameters of Discrete Input Function Blocks

The DI schematic is shown in the figure 2.

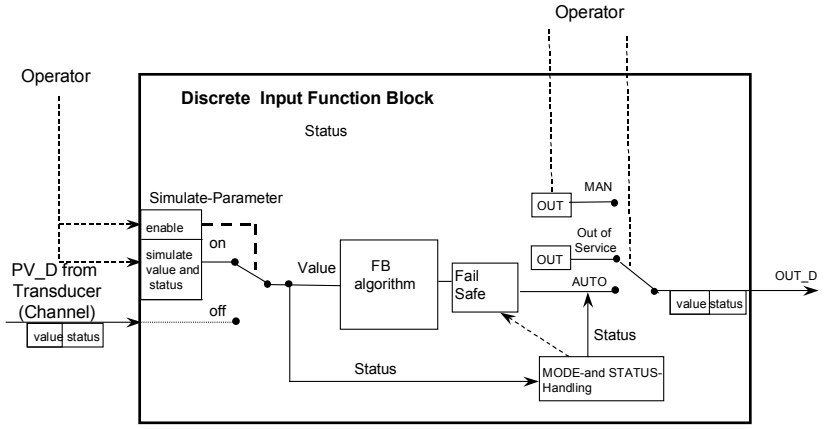

Figure 1.8 - Schematic of the Discrete Input Function Block

Description

The Transducer Block process variable status is visible at the Transducer Block. The Target Mode is set by the operator according to permitted Mode Block.

The Status (OUT_D) is coupled with the OUT_D value from the block.

Simulate

The SIMULATE_D parameter is used for the diagnostics and checkout purposes. When it is active, the transducer value and status will be overridden by the simulate value and status.

The SIMULATE_D structure is composed by the following attributes:

- Simulate Value and Status
- Simulate Enable

When simulation is enabled, the Transducer Input parameter will be calculated based on the attribute Simulate Value/Status of the SIMULATE_D parameter. Otherwise it will be that one supplied by the transducer block.

Supported Modes

O/S, MAN and AUTO.

Status Handling

The DI block does not support cascade path. Then, the output status has not a cascade sub-status.

Cyclic CFG_DATA

Function Block	Parameter	Identifier Byte	Extended Identifier format
Discrete Input (DI)	**OUT_D**	**0x91**	

Table 2.13 – Cyclic CFG_DATA

Discrete Function Block Parameter Descriptions

Parameter	Description
CHANNEL	Reference to the active Transducer Block which provides the measurement value to the Function Block.
INVERT	Indicates whether the input value of the PV_D should be logically inverted before it is stored in the OUT_D. List of valid values : 0 = not inverted 1 = invert
OUT_D	OUT_D is the output of the function block. The value is specified by the operator in MODE MAN.
FSAFE_TYPE	Defines reaction of device, if a fault is detected. 0 = value FSAVE_VALUE is used as OUT_D Status – UNCERTAIN_Initial Value , 1 = use of stored last valid OUT_D value Status - UNCERTAIN_LastUsableValue , if there is no valid value available, then UNCERTAIN- Inital_Value 2 = OUT_D has the wrong calculated value and status Status = BAD
FSAFE_VALUE	Default value for the OUT_D parameter, if a fault is detected.
SIMULATE	For commissioning and test purposes the input value from the Transducer Block in the Discrete Input Function Block DI-FB can be modified. That means that the Transducer and DI-FB will be disconnected.

Table 2.14 - Discrete Input Function Block Parameter Description

Discrete Input Function Block Parameter Attributes

Relative Index	Parameter Name	Object type	Data type	Store (S;D;N)	Size	Access	Param. Usage/ Kind of Transport	Default Values
Function Block Parameter for Discrete Input DI								
10	OUT_D	Record	DS-34	D	2	r,w	O/cyc	measured of the value, state
14	CHANNEL	Simple	Unsigned16	S	2	r,w	C/a	-
15	INVERT	Simple	Unsigned 8	S	1	r,w	C/a	0
20	FSAVE_TYPE	Simple	Unsigned 8	S	1	r,w	C/a	1
21	FSAVE_VAL_D	Simple	Unsigned 8	S	1	r,w	C/a	0
24	SIMULATE	Record	DS-51	S	3	r,w	C/a	disable
25-34	reserved by PNO							
35	first manufacture specific parameter							

Table 2.15 - Parameter attributes of the Discrete Input Function Block

BitStrings Description

DIAGNOSIS (PHYSICAL BLOCK)

Bit	Mnemonic	Meaning
0	DIA_HW_ELECTR	Hardware failure of the electronic components.
1	DIA_HW_MECH	Hardware failure of the mechanics parts.
2	DIA_TEMP_MOTOR	Motor temperature too high.
3	DIA_TEMP_ELECTR	Electronic temperature too high.
4	DIA_MEM_CHKSUM	Memory error.
5	DIA_MEASUREMENT	Failure in the Measurement.
6	DIA_NOT_INIT	Device not Initialized.
7	DIA_INIT_ERR	SelfCalibration failed.
10	DIA_ZERO_ERR	Zero point error.
11	DIA_SUPPLY	Power supply failed.
12	DIA_CONF_INVAL	Configuration not valid.
13	DIA_WARM_START	Re_startup carried out (Power Up). This bit is set to True when Power up occurs, and it will be automatically reset after 10 seconds.
14	DIA_COLD_START	New_startup carried out (Factory Init). This bit is set to True when factory init occurs, and it will be automatically reset after 10 seconds.
15	DIA_MAINTAINANCE	Maintenance required.
16	DIA_CHARACT	Characterization invalid.
17	IDENT_NUMBER_VIOLATION	Set to 1 if the Ident_Number of the running cyclic data transfer and the value of physical block IDENT_NUMBER_SELECTOR parameter are different.
37	EXTENSION AVAIABLE	More diagnosis information is available. The information of these diagnosis will be in the DIAGNOSIS_EXT.
Note: For more details about the parameters, consult the specific manual of each device.		

Table 2.16 – Diagnoses of Physical Block

CHECK_BACK (ANALOG OUTPUT BLOCK)

Bit	Mnemonic	Meaning	CHECK_BACK_MASK
0	CB_FAIL_SAFE	Field device in Fail safe active.	Used
1	CB_REQ_LOC_OP	Request for local Operation.	Not used
2	CB_LOCAL_OP	Field device under local control, LOCKED OUT switch is in gear.	Not used
3	CB_OVERRIDE	Emergency override active.	Not used
4	CB_DISC_DIR	Actual position feedback different from expected position.	Used
5	CB_TORQUE_D_OP	Torque limit in OPEN direction is exceeded.	Not used
6	CB_TORQUE_D_CL	Torque limit in CLOSE direction is exceeded.	Not used
7	CB_TRAV_TIME	Indicates status of travel monitoring equipment, if YES, travel time for actuator has exceeded.	Used
8	CB_ACT_OPEN	Actuator is moving towards open direction.	Used
9	CB_ACT_CLOSE	Actuator is moving towards close direction.	Used
10	CB_UPDATE_EVT	Alert generated by any change to the static data (Function and Transducer Block).	Used
11	CB_SIMULATE	Simulation of process values is enabled.	Used
13	CB_CONTR_ERR	Internal control loop disturbed.	Not used
14	CB_CONTR_INACT	Positioner inactive (OUT status = BAD).	Used
15	CB_SELFTEST	Device under self test.	Used
16	CB_TOT_VALVE_TRAV	Indicates that total valve travel limit is exceeded.	Used
17	CB_ADD_INPUT	Indicates that an additional input (i.e. for diagnostics) is activated.	Used
Note: For more details about the parameters, consult the specific manual of each device.			

Table 2.17 – Check Back of Analog Output Block

FB Set and FB Type Availability

Order Into the Directory	Block\s	LD		TT		DT		IF		FI		TP		FY		FP		LD293	
		FB type	Qtt	FB type	Qtt	FB type	Qtt	FB type	Qtt	FB type	Qtt	FB type	Qtt	FB type	Qtt	FB type	Qtt	FB type	Qtt
1	PHY	Yes	1	Yes	1	Yes	1	Yes	1	Yes	1	Yes	1	Yes	1	Yes	1	Yes	1
2	AO									Yes	3			Yes	1	Yes	1		
2	AI	Yes	1	Yes	2	Yes	1	Yes	3			Yes	1					Yes	1
3	TOT	Yes	1					Yes	3			Yes	1						
4	TRD-LD	Yes	1																
4	TRD-TP											Yes	1						
4	TRD-TT			Yes	2														
4	TRD-IF							Yes	3										
4	TRD-FI									Yes	3								
4	TRD-FP															Yes	1		
4	TRD-FY													Yes	1				
4	TRD-DT					Yes	1												
4	TRD-LD293																	Yes	1
5	TRD-DSP	Yes	1	Yes	1			Yes	1	Yes	1	Yes	1	Yes	1	Yes	1	Yes	1

Note 1 – The column “FB type” indicates which functional block type is available for each type of device.
Note 2 – The column “Qtt” indicates the number of functional blocks instantiated during the factory initialization for each type of device.
Note 3 – The column “Order into the directory” indicates the internal order of the blocks into the device. (This information is useful in the cyclic configuration – CFG_data config). Some blocks have the same number of others, because the device does not have both AO and AI functional blocks, and the transducer is specific for one device.

Table 2.18a – FB Set and FB Type Availability

Order Into the Directory	Blocks	DC303		FRI303	
		FB type	Qtt	FB type	Qtt
1	PHY	Yes	1	Yes	1
2	DO	Yes	8	Yes	2
3	DI	Yes	16	Yes	2
4	TRD_DC	Yes	1		
4	TRD_FR			Yes	1
5	TRD-DSP			Yes	1

Table 2.18b – FB Set and FB Type Availability for DC303 and FRI303