Chapter 6

INPUTS

Edition 4

Overview

Contents (continued)

Overview

This chapter describes the INPUTS function block class.

Input Function Blocks are automatically created as part of the PC3000 Hardware Definition i.e. the process of declaring which I/O module type resides in each position within the rack. Each Input Function Block is 'attached' to a physical I/O channel. Once defined they may be manipulated in the same way as function blocks from other classes.

PIM Module

The PIM Module is handled in a different way using the Mk1 Module.

The following is a guide to creating a user program involving a Mk2 PIM.

Do NOT request a PIM module in the 'Required Module' section of the Hardware Definition Screen. Instead, create an instance of the Mk2_PIM2 function block (to be found under MODULES in the function block 'Class List').

The address parameter must be set to the address of the module, for example a PIM Mk2 in slot 5 of rack 3 will have an address of '3:5'. This address is not automatically generated as with other PC3000 I/O.

Create an instance of a Mk2_PI_Ctr for each required counter channel, found under INPUTS in the class list. The address parameter must again be explicitly entered, this time with either a '....°1 ' or '....°2', depending upon the channel number. For example, channel 2 of a PIM Mk2 in slot 12 of rack 7 would have an address of '7:12:2'.

DIGITAL_IN FUNCTION BLOCK

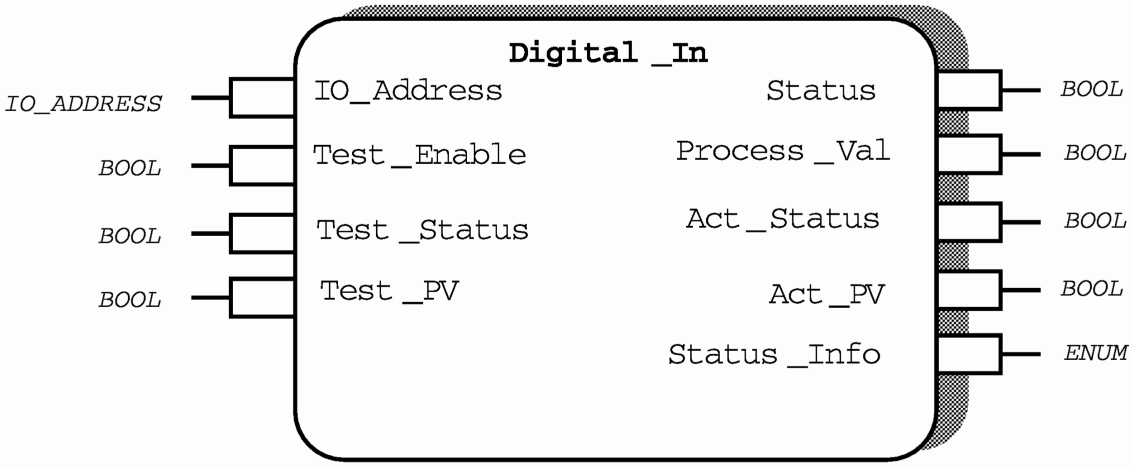


Figure 6-1 Digital_In Function Block

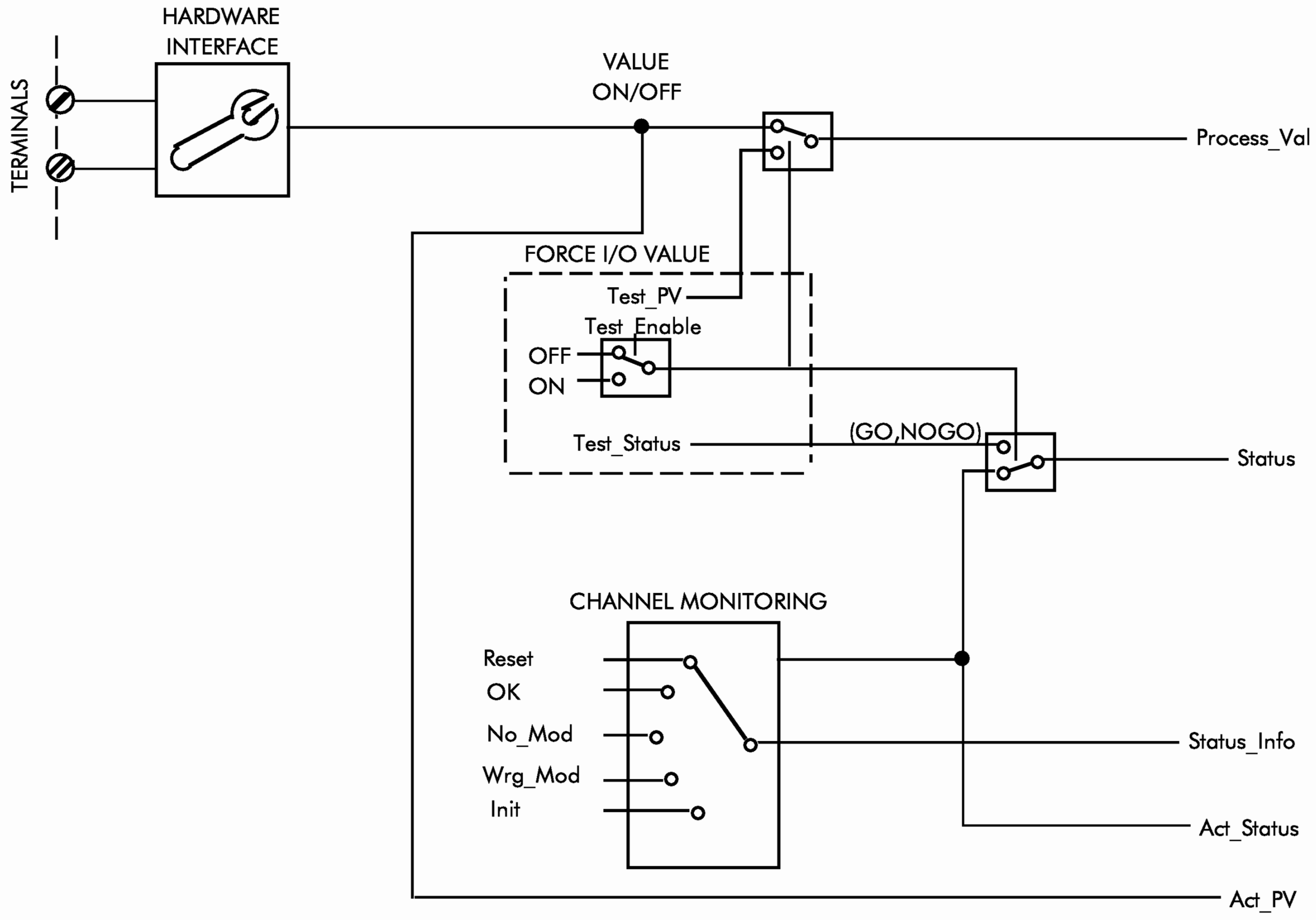
Functional Description

The Digital_In function block provides the interface to the Digital Input hardware module. It provides a boolean output parameter that defines the sense of an associated physical digital input.

Test facilities are provided to allow the physical input state and status to be overridden by test values.

Function Block Attributes

Type:..................................	10 10
Class:	INPUTS
Default Task:	Task_1
Short List:	Process_Val, Act_PV, Status, Status_Info
Memory Requirements	12 Bytes



Parameter Descriptions

IO_Address (IOA)

IO_Address associates the function block instance with the physical connections on the hardware module to which it is referenced. Its value is assigned automatically when the function block instance is defined. Its value takes the form X:YY:ZZ, where X represents the number of the rack in which the module is resident, YY represents the number of the slot in the rack and ZZ represents the number of the channel within the module. For example, 1:02:03 would mean that the function block instance references the third channel of a module which sits in the second slot of the first rack of the PC3000 system.

Test_Enable (TEN)

Test_Enable allows the user to switch the value read by Process_Val between the hardware input channel and the Test_PV input to the function block. If Test_Enable is set to Off (0), Process_Val will read its value from the hardware input and Status will reflect the status of the hardware. If Test_Enable is set to

On (1), Process_Val will read its value from Test_PV and Status will be driven by the Test_Status input.

Test_Status (TST)

The value of Test_Status is copied to Status when Test_Enable is On (1). When Test_Enable is set to Off (0), Test_Status is not used.

Test_PV (TPV)

Test_PV is copied to the Process_Val output when Test_Enable is On (1). When Test_Enable is set to Off (0), Test_PV is not used.

Status

When Test_Status is set to Off (0), the parameter Status reflects the status of the hardware channel being addressed by the function block. If Test_Status is set to On (1), Status will read its value directly from Test_Status. Status can take the values NOGO (0) or Go (1).

Process_Val (PV)

During normal operation, when Test_Enable is set to Off (0), Process_Val reflects the input to the digital input channel being addressed by the function block. If Test_Enable is set to On (1), Process_Val reads its value directly from Test_PV. This is the value used by the control strategy.

Act_Status (AST)

Act_Status reflects the status of the hardware channel being addressed by the function block. Act_Status always reflects the hardware status, whereas Status can be switched between the hardware input and Test_Status in test mode. This parameter should be used for diagnostic purposes only.

Act_PV (APV)

Act_PV reflects the input to the digital input channel being addressed by the function block. Act_PV always reflects the hardware input, whereas Process_Val can be switched between the hardware input and Test_PV in test mode. This parameter should be used for diagnostic purposes only.

Status_Info (STI)

Status_Info is a diagnostic parameter which is used to explain the state of Status. It can have 5 possible states:

Reset (0): User program is not running.

Ok (1): The channel is functioning normally.

No_Mod (2): There is no module in the hardware slot being addressed by the function block.

Wrg_Mod (3): An incorrect module type has been fitted in the slot addressed by the function block.

Init (4): The module or channel is being initialised. This condition is transient and may be seen at start up.

Parameter Attributes

Name	**Type**	**Cold Start**	**Read Access**	**Write Access**	**Type Specific Information**	
Act_PV	**BOOL**	Off (0)	Config	Block	Senses	Off (0) On (1)
Act_Status	**BOOL**	NOGO (0)	Config	Block	Senses	NOGO(0) Go (1)
IO_Address	**IO_ADDRESS**		Config	Config		
Process_Val	**BOOL**	Off (0)	Oper	Block	Senses	Off (0) On(1)

Status	**BOOL**	NOGO (0)	Oper	Block	Senses	NOGO(0)
						Go(1)

Status_Info	**ENUM**	Reset (0)	Oper	Block	Senses	See Parameter Description

Test_Enable	**BOOL**	Off (0)	Config	Config	Senses	Off(0)
						On(1)

Test_PV	**BOOL**	Off (0)	Config	Config	Senses	Off(0)
						On(1)

Test_Status	**BOOL**	NOGO (0)	Config	Config	Senses	NOGO(0)
						Go(1)

Table 6-1 Parameter Attributes

DEBOUNCE_IN FUNCTION BLOCK

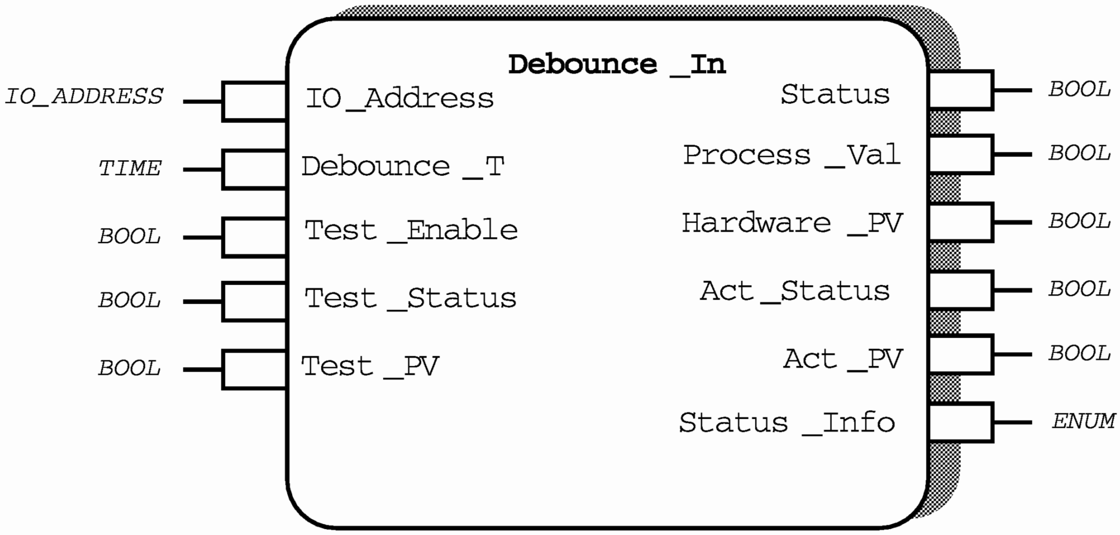


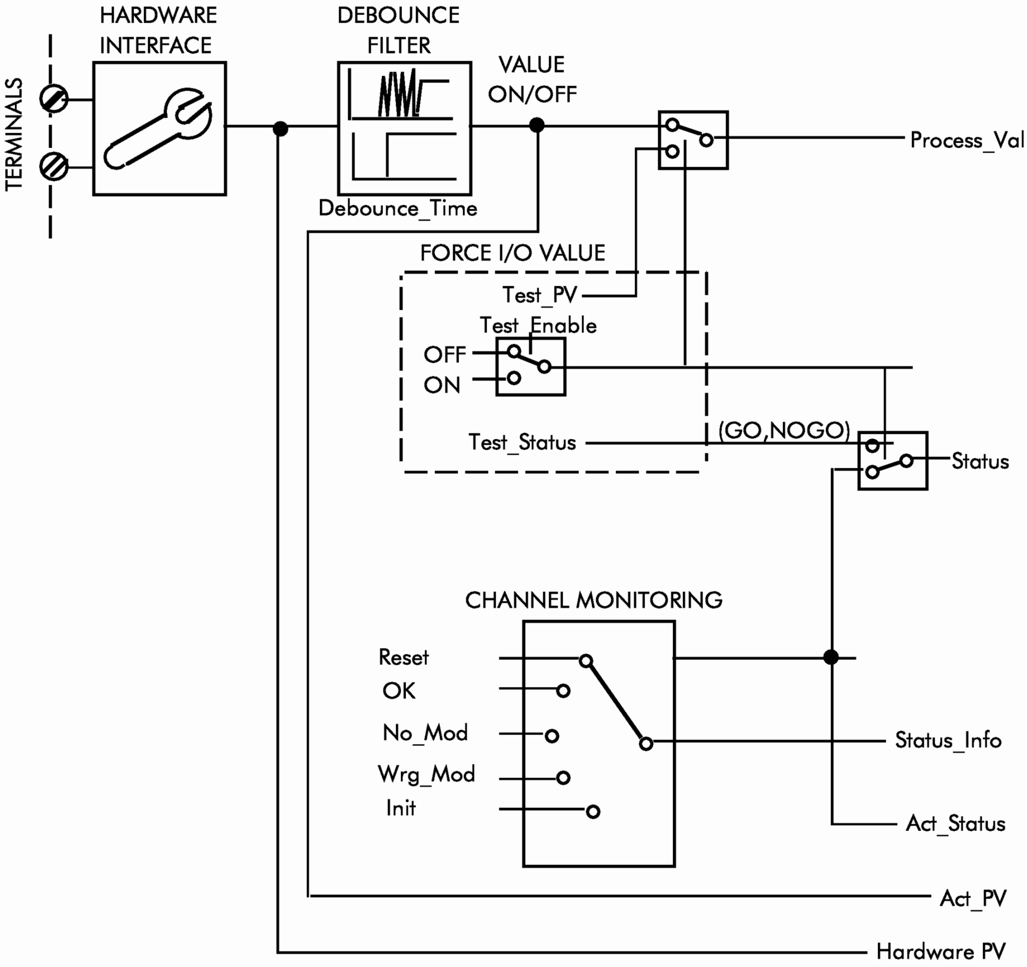
Figure 6-2 Debounce_In Function Block

Functional Description

The Debounce_In function block provides the interface to the Digital Input hardware module. Debounce functionality is included to compensate for contact bounce. When debounce is employed, the Process_Val changes on the first edge detected and is guaranteed not to change again for the debounce time.

Function Block Attributes

Type:..................................	10 14
Class:..................................	INPUTS
Default Task:	Task_1
Short List:	Process_Val, Act_PV, Status, Status_Info
Memory Requirements:	22 Bytes



Parameter Descriptions

IO_Address (IOA)

IO_Address associates the function block instance with the physical connections on the hardware module to which it is referenced. Its value is assigned automatically when the function block instance is defined. Its value takes the form X:YY:ZZ, where X represents the number of the rack in which the module is resident, YY represents the number of the slot in the rack and ZZ represents the number of the channel within the module. For example, 1:02:03 would mean that the function block instance references the third channel of a module which sits in the second slot of the first rack of the PC3000 system.

Debounce_T (DT)

Debounce_T defines the amount of time for which the Process_Val is held following detection of an edge to the input channel.

Test_Enable (TEN)

Test_Enable allows the user to switch the value read by Process_Val between the hardware input channel and the Test_PV input to the function block. If Test_Enable is set to Off (0), Process_Val and Hardware_PV will read their value from the hardware input and Status will reflect the status of the hardware.

If Test_Enable is set to On (1), Process_Val and Hardware_PV will read their value from Test_PV and Status will be driven by the Test_Status input.

Test_Status (TST)

The value of Test_Status is copied to Status when Test_Enable is On (1). When Test_Enable is set to Off (0), Test_Status is not used.

Test_PV (TPV)

Test_PV is copied to the Process_Val and Hardware_PV outputs when Test_Enable is On (1). When Test_Enable is set to Off (0), Test_PV is not used.

Status

When Test_Status is set to Off (0), the parameter Status reflects the status of the hardware channel being addressed by the function block. If Test_Status is set to On (1), Status will read its value directly from Test_Status. Status can take the values NOGO (0) or Go (1).

Process_Val (PV)

During normal operation, with Test_Enable is set to Off (0), Process_Val reflects the input to the digital input channel being addressed by the function block. If Test_Enable is set to On (1), Process_Val reads its value directly from Test_PV.

In both modes, the debounce functionality acts to ensure that Process_Val responds only to the first edge of the changing input. This is the value used by the control strategy.

Hardware_PV (HPV)

Hardware_PV reflects the process value before the debounce filter.

Act_Status (AST)

Act_Status reflects the status of the hardware channel being addressed by the function block. Act_Status always reflects the hardware status, whereas Status can be switched between the hardware input and Test_Status in test mode. This parameter should be used for diagnostic purposes only.

Act_PV (APV)

Act_PV reflects the input to the digital input channel being addressed by the function block, with debounce functionality included to ensure that Act_PV responds only to the first edge of the changing input.

Act_PV always reflects the hardware input, whereas Process_Val can be switched between the hardware input and Test_PV in test mode. This parameter should be used for diagnostic purposes only.

Status_Info (STI)

Status_Info is a diagnostic parameter which is used to explain the state of Status. It can have 5 possible states:

Reset (0): The user program is not running

Ok (1): The channel is functioning normally.

No_Mod (2): There is no module in the hardware slot being addressed by the function block.

Wrg_Mod (3): An incorrect module type has been fitted in the slot addressed by the function block

Init (4): The module or channel is being initialised. This is a transient state which may be seen at start up.

Parameter Attributes

Name	Type	Cold Start	Read Access	Write Access	Type Specific Information	

Act_PV	**BOOL**	Off (0)	Config	Block	Senses	Off(0)
						On(1)

Act_Status	**BOOL**	NOGO (0)	Config	Block	Senses	NOGO(0)
						Go(1)

Debounce_T	**TIME**	0	Oper	Oper	High Limit Low Limit	1d_3h_46m_0

Hardware_PV	**BOOL**	Off (0)	Oper	Block	Senses	Off(0)
						On(1)

IO_Address	**IO_ADDRESS**		Config	Config		

Process_Val	**BOOL**	Off (0)	Oper	Block	Senses	Off(0)
						On(1)

Status	**BOOL**	NOGO (0)	Oper	Block	Senses	NOGO(0)
						Go(1)

Status_Info	**ENUM**	Reset (0)	Oper	Block	Senses	See Parameter Description

Test_Enable	**BOOL**	Off (0)	Config	Config	Senses	Off(0)
						On(1)

Test_PV	**BOOL**	Off (0)	Config	Config	Senses	Off(0)
						On(1)

Test_Status	**BOOL**	NOGO (0)	Config	Config	Senses	NOGO(0)
						Go(1)

Table 6-2 Debounce_In Parameter Attributes

ANALOG_IN FUNCTION BLOCK

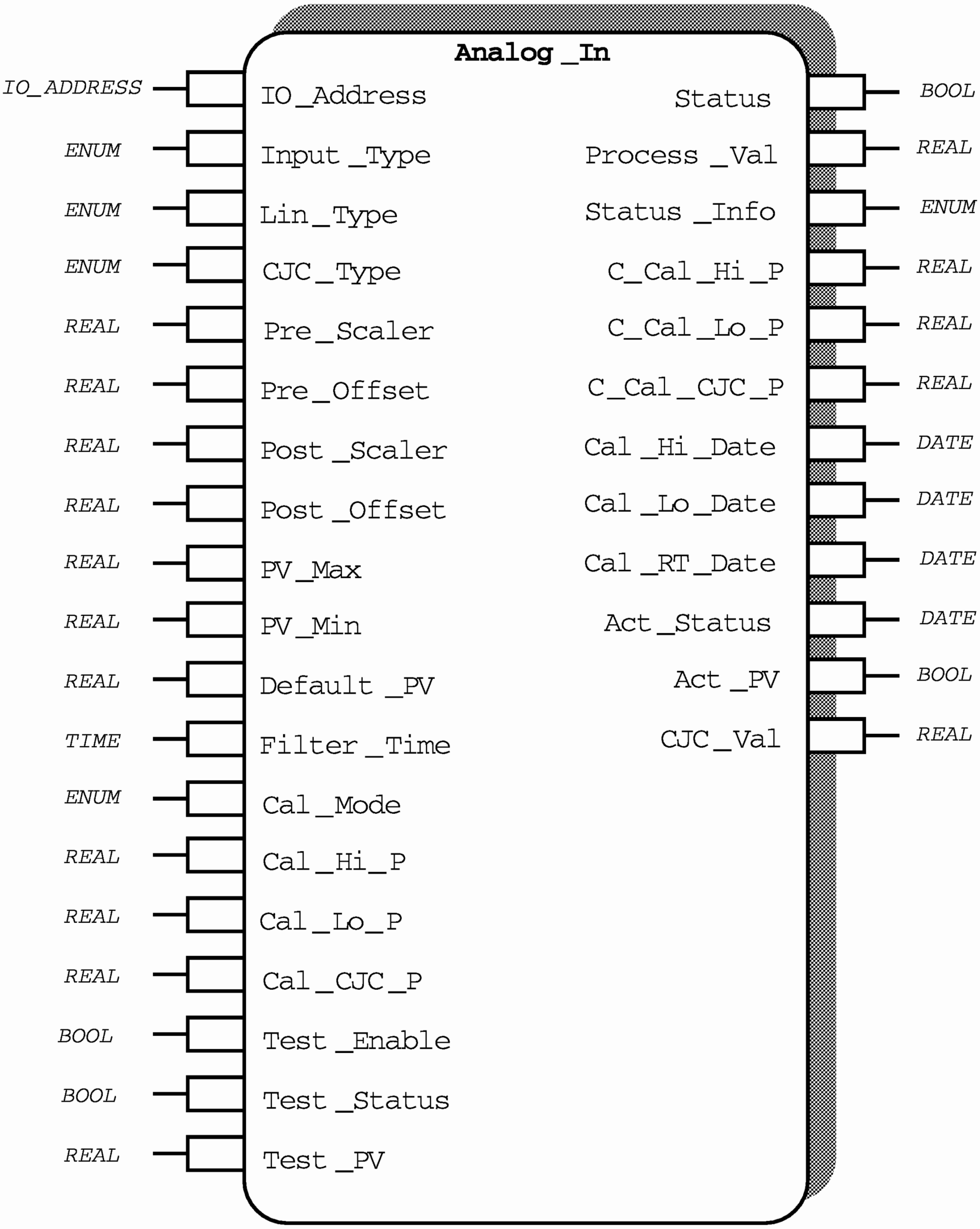


Figure 6-3 Analog_In Function Block Diagram

Functional Description

The Analog_In function block provides the interface to the Analogue Input hardware module. Input linearisation and low pass filtering is provided along with scaling and offsetting of the measured value. There are also test facilities to allow the physical measured value and status to be overridden by test values.

The analogue function block is used with both 2 and 4 channel hardware modules.

Function Block Attributes

Type:	16 32
Class:	INPUTS
Task:	Task_2
Short List:	Process_Val, Act_PV, Status, Status_Info
Memory Requirement:	126 Bytes
Execution Time:	175 µ Secs

Parameter Descriptions

Parameters marked with '*' **must** be configured prior to running the program. All other parameters are optional.

IO_Address (IOA)

IO_Address associates the function block instance with the physical connections on the hardware module to which it is referenced. Its value is assigned automatically when the function block instance is defined. Its value takes the form X:YY:ZZ, where X represents the number of the rack in which the module is resident, YY represents the number of the slot in the rack and ZZ represents the number of the channel within the module.

For example, 1:02:03 would mean that the function block instance references the third channel of a module which sits in the second slot of the first rack of the PC3000 system.

Input_Type (IT) *

The input range of an analogue input channel is set by means of a parameter (Input_Type). The input range is considered to be normal maximum working range of the input. In pratice there is approximately 20-25% headroom above the maximum range setting after which the input channel will declare a hardware overrange condition.

This parameter is not automatically linked to thermocouple type or maximum and minimum sensor range; these must be set independently selecting the linearisation type (Lin_Type), the maximum expected temperature or input (PV_Max) and the minimum input value (PV_Min).

Two Channel Range Setting

The 2 channel module has a universal input, allowing a mix of different input types on the same card.

The following settings apply:

'Old' Range	Range	Hardware Range
mV50	Range_1	0 to 50mV linear or tc inputs
mV100	Range_2	0 to 100mV linear or tc inputs
mV50RT	Range_3	2 wire Resistance Thermometer 0 to 500 ohm
mV100RT	Range_4	3 wire Resistance Thermometer 0 to 500 ohm
V0_5	Range_5	0 to 5V linear
V0_10	Range_6	0 to 10V linear

Four Channel Range Setting

The 4 channel modules come in 4 different derivatives. The setting depends upon the module type:

Four Channel Millivolt Input Variant:

PC3000/AI/VERSION2/MV4 and PC3000/A1/V2/MV3/HIZ1

Range	Hardware Range	
	MV4	MV3/HIZ1*
Range_1	-10 to 10mV	-1 to +1V
Range_2	-10 to 20mV	-1 to +2V
Range_3	-10 to 50mV	-1 to +5V
Range_4	-10 to 100mV	-1 to +10V
Range_5	N/A	N/A
Range_6	N/A	N/A

* Channel 1 only.

The LCM firmware and PS Hardware Definition see the MV3/HIZ1 module just as MV4.

Four Channel Milliamp Input Variant:

PC3000/AI/VERSION2/MA4

Assume module is fitted with 5 ohm burden resistors.

Range	Hardware Range
Range_1	0 to 2mA (Not used)
Range_2	0 to 4mA (Not used)
Range_3	0 to 10mA
Range_4	0 to 20mA
Range_5	N/A
Range_6	N/A

Four Channel Voltage Input Variant:

PC3000/AI/VERSION2/V4 and PC3000/A1/V2/FV2

Note bipolar input range

Range	Hardware Range	
	V4	V2/FV2*
Range_1	-1 to 1V	10Hz - 1KHz
Range_2	-2 to 2V	10HZ - 2KHz
Range_3	-5 to 5V	10Hz - 5KHz
Range_4	-10 to 10V	10Hz - 10KHz
Range_5	N/A	N/A
Range_6	N/A	N/A

* Channels 1 and 3 only. The LCM firmware and PS Hardware Editor see V2/FV2 module just as V4.

Four Channel Resistance Thermometer Input Variant:

PC3000/AI/VERSION2/RT4

Range	Hardware Range
Range_1	0 to 50 Ohms
Range_2	0 to 100 Ohms
Range_3	0 to 250 Ohms
Range_4	0 to 500 Ohms
Range_5	N/A
Range_6	N/A

Lin_Type (LT) *

Lin_Type defines the type of input linearisation employed by the function block. The parameter is an enumerated type which can take on the values shown in the following table:

Value	Mnemonic	Linearisation Type	Range
0	**Linear**	None	
1	**J**	Fe / Const thermocouple	-210 to 1200 °C
2	**L**	Fe / Const (DIN) thermocouple	-21 to 600 °C
3	**K**	NiCr / NiAl thermocouple	-265 to 1372 °C
4	**T**	Cu / Const thermocouple	-270 to 400 °C
5	**R**	Pt13%Rh / Pt thermocouple	-50 to 1767 °C
6	**S**	Pt10%Rh / Pt thermocouple	-50 to 1767 °C
7	**B**	Pt30%Rh / Pt6%Rh thermocouple	0 to 1820 °C
8	**E**	NiCr / Const thermocouple	-270 to 1000 °C
9	**Pt_100**	Pt 100Ω resistance thermocouple	-200 to 800 °C
10	**G1**	W / W26%Re thermocouple	0 to 2300 °C
11	**WRe5_26**	W5%Re / W26%Re thermocouple	0 to 2300 °C
12	**PR10_40**	Pt10%Rh / Pt40%Rh thermocouple	200 to 1800 °C
13	**C**	W5%Re / W26%Re thermocouple	0 to 2300 °C
14	**PR20_40**	Pt20%Rh / Pt40%Rh thermocouple	0 to 2000 °C
15	**Plat_II**	Platinel II thermocouple	0 to 1200 °C

Table 6-3 Input Linearisation

Value	Mnemonic	Linearisation Type	Range
16	**G2**	W / W26%Re thermocouple	0 to 2200 °C
17	**NC_NM**	Ni / Ni18%Mo thermocouple	0 to 1100 °C
18	**D**	W3%Re / W25%Re thermocouple	0 to 2400 °C
19	**WRe26_5**	WRe5% / WRe26% thermocouple	0 to 2000 °C
20	**Nic_Nis**	Nicrosil / Nisil thermocouple	0 to 1300 °C
21	**Q004**	Q004 land pyrometer	700 to 1600 °C
22	**Q003**	Q003 land pyrometer	600 to 1500 °C
23	**R_026**	R026 pyrometer	0 to 500 °C
24	**IVDI**	IVDI pyrometer	1000 to 2500 °C
25	**DT1**	DT1 pyrometer	750 to 2500 °C
26	**DT1_10**	DT1 / 10 pyrometer	1000 to 3000 °C
27	**R_023**	R023 pyrometer	700 to 1700 °C
28	**FP_GP10**	FP / GP 10 pyrometer	450 to 900 °C
29	**FP_GP11**	FP / GP 11 pyrometer	600 to 1300 °C
30	**FP_GP12**	FP / GP 12 pyrometer	750 to 1850 °C
31	**FP_GP20**	FP / GP 20 pyrometer	300 to 750 °C
32	**FP_GP21**	FP / GP 21 pyrometer	500 to 1100 C°
33	**SQRT**	Square root function	

Table 6-3 Input Linearisation

CJC_Type (CT)

CJC_Type defines the type of cold junction compensation employed by the function block. This is necessary to compensate for the Seebeck effect, which occurs at the connection between the thermocouple cable and the instrument. Cold junction compensation is only invoked by the function block when a thermocouple linearisation has been selected at the parameter Lin_Type. CJC_Type can be set to one of three options. These are:

Intern (0):

This should be used when the thermocouple is connected directly to the PC3000 analogue input module.

Ext_0C (1):

This should be used when the thermocouple is connected to an external cold junction fixed at 0°C. The Ext_0C (1) algorithm compensates for a cold junction temperature of 0°C.

Ext_50C (2):

This should be used when the thermocouple is connected to an external cold junction fixed at 50°C. The Ext_50C (1) algorithm compensates for a cold junction temperature of 50°C.

Selecting Process Value Range *

Set by the parameters, PV_Max and PV_Min. PV_Max represents the maximum value that the channels Process_Value can take without the modules' Status reverting from Go (1) to NOGO (0). PV_Min is the minimum value of the Process_Value.

The values of PV_Max and PV_Min must be set taking into account any pre or post linearisation scaling and offset values that may have been set up. As an example, a 0 to 5V input scaled to provide a linear output in the range 0 to 100% would be configured as follows:

Parameter	**Value**
Input_Type	Range_5
Pre_Scaler	20
Pre_Offset	0
PV_Max	110% (10% headroom)
PV_Min	-10% (10% headroom)

Scalers and offsets are discussed in a later section.

Note:

PV_Max and PV_Min must be set up explicitly. They are not linked to the Linearisation table selected from Lin_Type.

PV_Max and PV_Min should be set to the table limits (Refer to the technical specification) or to the maximum working range.

If the defined limits are exceeded the parameter, Status_Info, indicates the reason; Over_R(8) if input exceeds maximum limit defined by PV_Max or the hardware input limit, or Under_R(9) if input exceeds minimum limit defined by PV_Min or the hardware input limit.

PV_Max (PMX) *

PV_Max is used to define the upper limit of the Process_Val. This limit is invoked after pre and post input scaling and offsetting has been performed.

If Process_Val is greater than PV_Max, Status will be set to NOGO (0), Staus_Info will be set to Over_R (8) and Process_Val will be set to Default_PV.

PV_Min (PMN) *

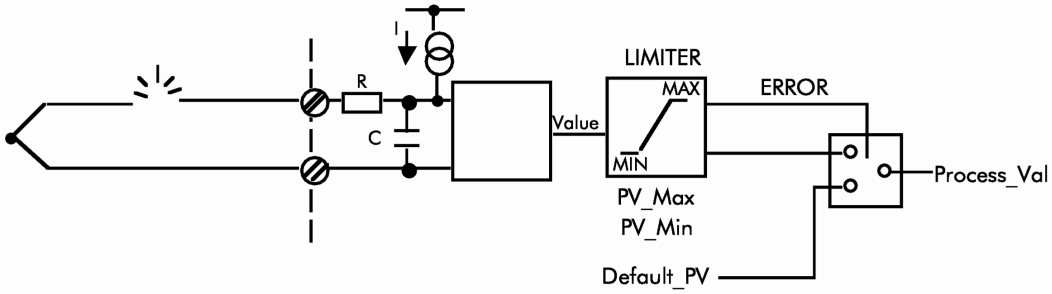
PV_Min is used to define the lower limit of the Process_Val. This limit is invoked after pre and post input scaling and offsetting have been performed.

If Process_Val is less than PV_Min, Status will be set to NOGO (0), Staus_Info will be set to Under_R (9) and Process_Val will be set to Default_PV.

Sensor Break *

The parameter, Default_PV may be used in conjunction with PV_Max and PV_Min to set a 'safe' value on the Process_Value when the defined PV limits are exceeded.

As a recommendation it is normal practice to set the Default_PV to the value of PV_Max-1. This will ensure that in the event of sensor break the PV will be forced upscale and the output power via the PID Function Block will be turned off.



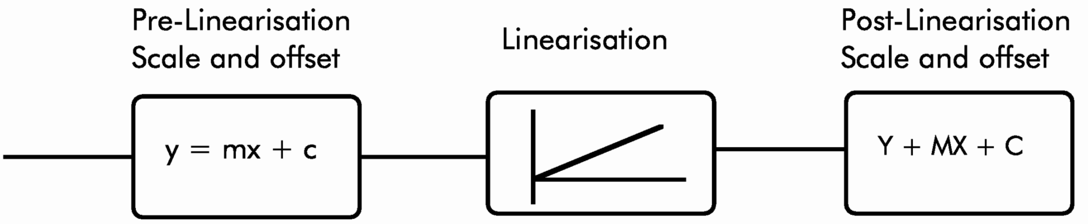
Default_PV (DPV) *

Default_PV is used to drive the Process_Val when the Status is NOGO (0).

Pre-Linearisation Scaler and Offset

Two parameters, Pre_Scaler and Pre_Offset, provide a means of scaling the input signal prior to performing linearisation.

The pre-linearisation scaler and offset act as follows:


Pre-Linearisation Scale and offset
Linearisation
Post-Linearisation Scale and offset
y = mx + c
Y + MX + C

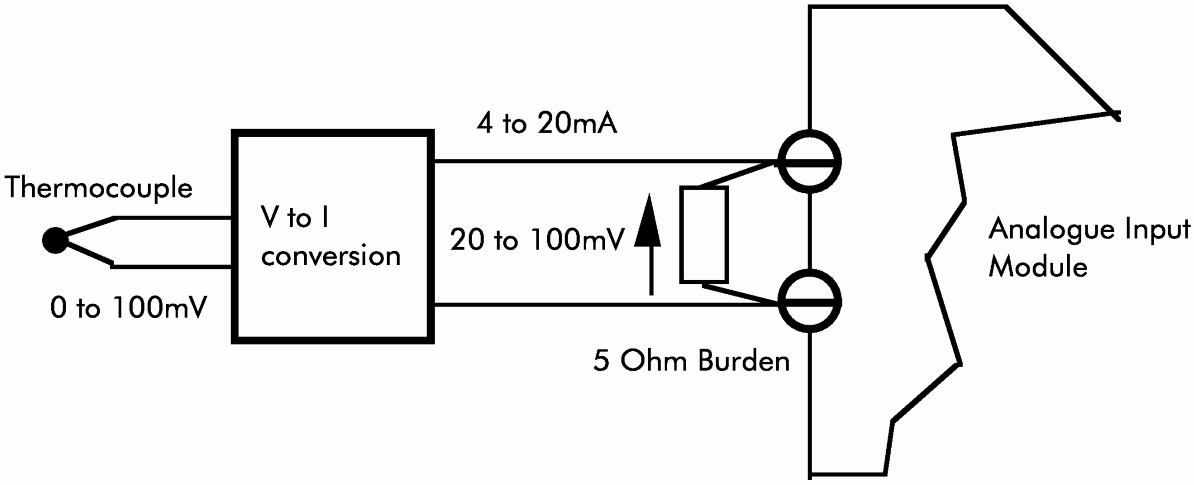
m = Pre-Scaler

c = Pre-Offset

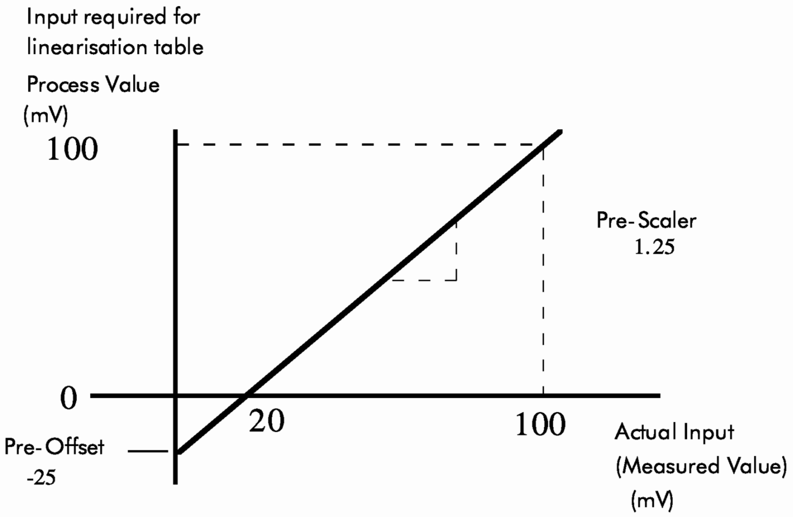
M = Post-Scaler

C = Post-Offset

As an example consider a 4 to 20mA input signal from a remote thermocouple signal conditioner. The remote thermocouple produces a 0 to 100mV input signal which is subsequently delivered to the PC3000 as 4 to 20mA.



The PC3000 input requires a 5 Ohm burden resistor to convert this input to 20 to 100mV. The linearisation table converts millivolts to degrees Centigrade. The pre-linearisation scaler and offset must therefore be set:

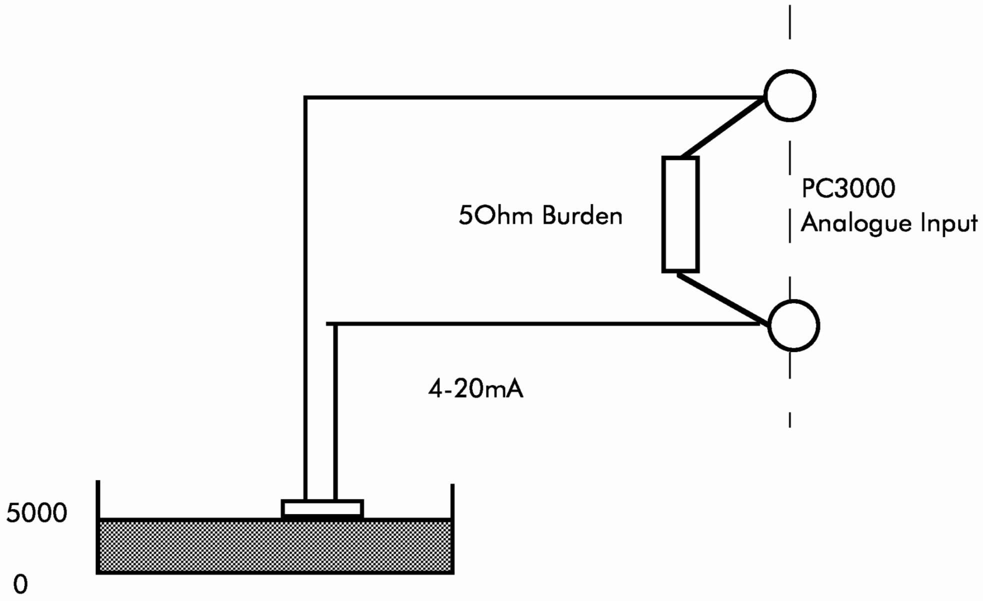


Pre-scaler = 100/(100-20) = 1.25

Pre-offset = 100-(1.25*100) = -25

(from y = mx + c)

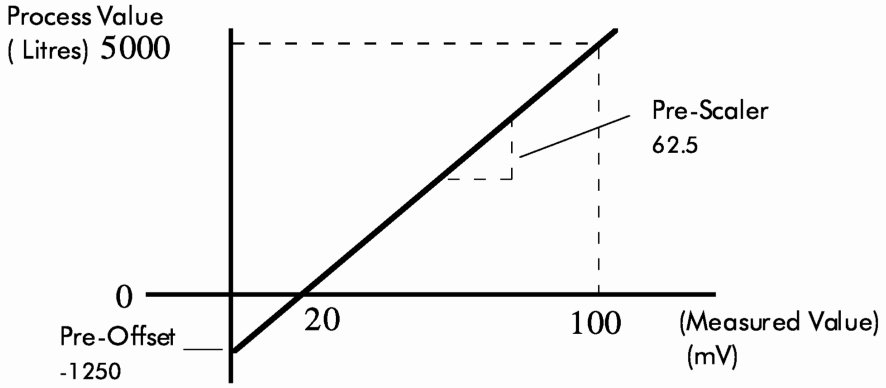
Another example might be:



Tank fills to 5000 litres

Transmitter delivers 4 to 20mA

Input signal to PC3000 is 20 to 100mV due to 5 Ohm burden



Pre-Scaler = 5000/(100-20) = 62.5

Pre-Offset = 5000/ - (62.5 * 100) = -1250

Rule:

Pre-Scaler = (MaxPV - MinPV) / (MaxMV - MinMV)

Pre_Offset = (MaxPV) - (Pre_Scaler * MaxMV)

where: MaxPV is the maximum process value i.e. 5000 litres

MinPV is the minimum process value i.e. 0 litres

MaxMV is the maximum measured value (at the PC3000 input) i.e. 100mV

MinMV is the minimum measured value (at the PC3000 input) i.e. 20mV

Pre_Scaler (PRS)

Pre_Scaler is a scaler which is applied to the input signal before the linearisation algorithm is invoked.

Pre_Offset (PRO)

Pre_Offset is an offset which is applied to the input signal before the linearisation algorithm is invoked.

Post -Linearisation Scaler and Offset

Two parameters, Post_Scaler and Post_Offset, provide a means of scaling the input signal after performing linearisation.

The post-linearisation scaler and offset act as indicated in the diagram on page 6-23

As an example consider a thermocouple signal which produces an output value of 873.5°C. The post-linearisation scaler and offset may be used to provide a value in °F. The post-linearisation scaler and offset must therefore be set:

post-scaler = 9/5

post-offset = 32

from y = mx + c

Post_Scaler (POS)

Post_Scaler is a scaler which is applied to the input signal after the linearisation algorithm has been invoked.

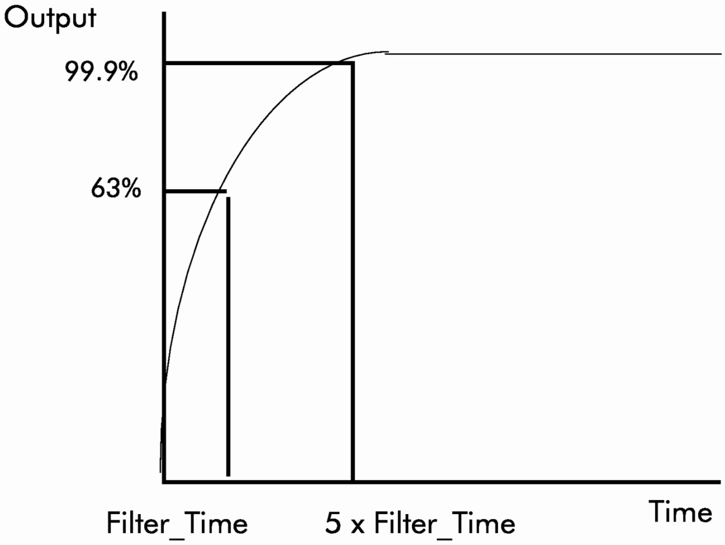
Post_Offset (POO)

Post_Offset is an offset which is applied to the input signal after the linearisation algorithm has been invoked.

Filter_Time (FT)

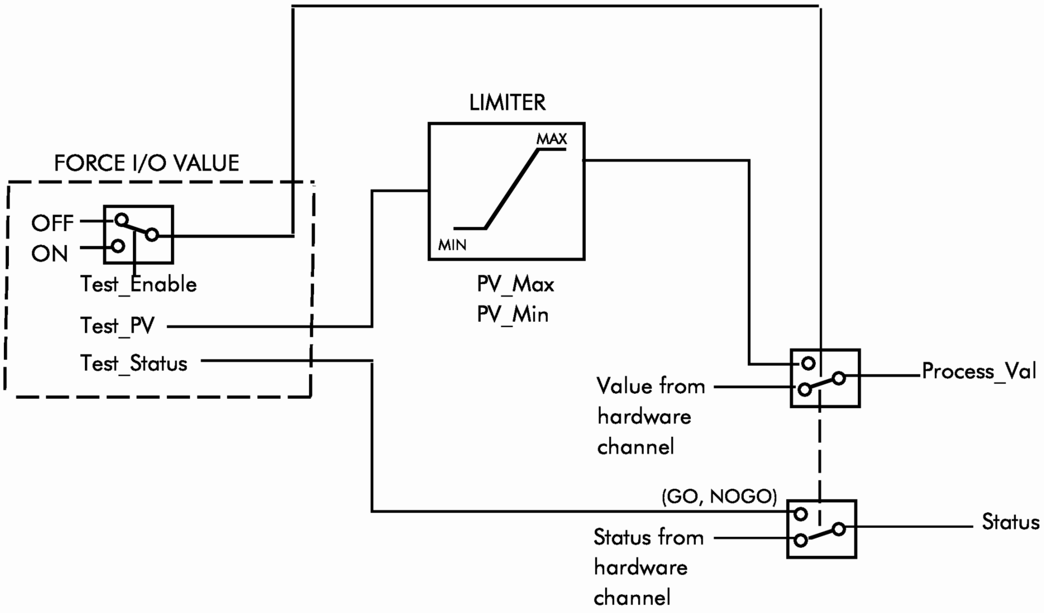
The input module includes a programmable low pass filter. The time constant of this filter may be set by means of this parameter. By default the filter time is set to zero, turning the filter action off.

The filter may be used to reduce the effect of random noise on the measured value. An increase in effective resolution may be gained at the expense of input response time by averaging the input over a longer period of time.



Forcing I/O Value

These parameters provide the ability to decouple the program from the real value of the I/O. This allows values acted upon by the PC3000 program to be 'forced' independent of the condition that exists in the real world. This means that failure mechanisms may be easily tested or commissioning problems ,such as 'limit switch not yet installed' may be overcome.



Test_Enable (TEN)

This must be set to the On (1) state to force the Process_Value. Once set the values of Status and Process_Value are set by the following parameters.

Test_Status (TST)

With the channel placed in test mode, this parameter may be used to directly control the Status parameter of the channel.

> **Note:** The value of the Status_Info parameter will always indicate OK in this mode of operation.

Test_PV (TPV)

With the channel placed in test mode, this parameter may be used to directly control the Process_Value parameter of the channel.

Status (ST)

When Test_Status is set to Off (0), the parameter Status reflects the status of the hardware channel being referenced by the function block. If Test_Status is set to On (1), Status will read its value directly from Test_Status.

Status can be NOGO (0) or Go (1)

Process_Val (PV)

During normal operation, when Test_Enable is set to Off (0), Process_Val reflects the input to the analog input channel being addressed by the function block, after linearisation, scaling and offsetting have been performed. If Test_Enable is set to On (1), Process_Val reads its value directly from Test_PV.

This is the value used by the control strategy.

Status_Info (STI)

Status_Info is a diagnostic parameter which is used to explain the state of Status. It can have nine possible states:

Reset (0): The user program is not running.

Ok (1)) The channel is functioning normally.

No_Mod (2): There is no module in the hardware slot being addressed by the function block.

Wrg_Mod (3): An incorrect module type has been fitted in the slot addressed by the function block

Init (4): The module or channel is being initialised

_ (5): No Status_Info function

No_Cal (6): The input range which has been selected at Input_Type has not been calibrated

Calib (7): The module is currently in calibration mode

Over_R (8): Either Process_Val is greater than PV_Max, or the hardware input range has been exceeded.

Under_R (9): Either Process_Val is less than PV_Min, or the hardware input range has been exceeded.

Act_PV (APV) and Act_Status (AST)

These are outputs from the Function Block. Act_PV indicates the value returned by the channel. In normal operation the value Act_PV will match that of the Process_Value. Act_PV includes the effect of linearisation, and any scalers and offsets which may have been set. Similarly, Act_Status will normally match the status indicated by Status. However, the following conditions may occur:

(1) Value produced by Function Block outside of limits defined by, PV_Max and PV_Min.

PV_Max represents the maximum value that the channels Process_Value can take without the modules' Status reverting from Go(1) to NOGO(0). PV_Min is the minimum value of the Process_Value.

As an example if the following parameter values have been set:

Input_Type	Range 1 (50mV)
Lin_Type	J
PV_Max	500°C
PV_Min	-10°C

All other parameters assume default values. If the input millivolt signal causes the value returned by the block to exceed 500°C then the parameters assume the following values:

Act_PV	634.3
Act_Status	Go
Process_Value	Default_PV
Status	NOGO

If the thermocouple had failed, the Act_Status would also be set to NOGO(0) and the ACT_PV would be frozen at the last value before overrange occurred. The parameter, Status_Info records the reason for the NOGO(0) status.

(2) Use of Test Mode

If the Process_Value is forced using the test mode, the data produced by the channel is still indicated using Act_PV and Act_Status. See section on 'Forcing I/O' for further details.(

3) During Calibration

If the channel is placed in calibration mode by setting its' mode to I/P_Hi, the data produced by the channel is indicated using Act_PV and Act_Status. See section on 'Calibration' for further details.

Whilst calibrating a channel the parameters assume the following values:

Act_PV	Calibration input
Act_Status	NOGO
Process_Value	Default_PV
Status	NOGO
Status_Info	Calib(rating)

The Act_PV now indicates the 'raw' value produced by the hardware channel, since the effect of linearisation and any scalers or offsets which may have been set up are switched off during hardware calibration.

Calibration Parameters

The parameters listed below are used for calibrating the hardware module.

Cal_Mode
Cal_Hi_P
Cal_Lo_P
Cal_CJC_P
C_Cal_Hi_P
C_Cal_Lo_P
C_Cal_CJC_P
Cal_Hi_Date
Cal_Lo_Date
Cal_CJC_Date
Cal_RT_Date
CJC_Val

See 'PC3000 Installation' handbook HA022231 for full details of calibration procedure.

Parameter Attributes

Name	Type	Cold Start	Read Access	Write Access	Type Specific Information	
Act_PV	**REAL**	0.0	Config	Block	High Limit Low Limit	PV_Max PV_Min
Act_Status	**BOOL**	NOGO (0)	Config	Block	Senses	NOGO (0) Go (1)
C_Cal_CJC_P	**REAL**	0	Config	Block	High Limit Low Limit	1000°C -273°C
C_Cal_Hi_P	**REAL**	0.0	Config	Block	High Limit Low Limit	100,000 -10,000
C_Cal_Lo_P	**REAL**	0.0	Config	Block	High Limit Low Limit	100,000 -10,000
Cal_CJC_Date	**DATE**	00:00:00 Jan 1 1970	Config	Block	High Limit Low Limit	Jan 19 2038 00:00:00 Jan 1 1970 00:00:00 Note 1
Cal_CJC_P	**REAL**	0.0	Config	Config	High Limit Low Limit	1000 °C -1000 °C
Cal_Hi_Date	**DATE**	00:00:00 Jan 1 1970	Config	Block	High Limit Low Limit	Jan 19 2038 00:00:00 Jan 1 1970 00:00:00 Note 1
Cal_Hi_P	**REAL**	0.0	Config	Config	High Limit Low Limit	100,000 Cal_Lo_P
Cal_Lo_Date	**DATE**	00:00:00 Jan 1 1970	Config	Block	High Limit Low Limit	Jan 19 2038 00:00:00 Jan 1 1970 00:00:00 Note 1
Cal_Lo_P	**REAL**	0.0	Config	Config	High Limit Low Limit	Cal_Hi_P -10,000
Cal_Mode	**ENUM**	Run (0)	Config	Config	Senses	Run (0) Save (1) I/P_Hi (2) I/P_Lo (3) CJC (4)

Table 6-4 Analog_In Parameter Attributes (Continued)

Name	Type	Cold Start	Read Access	Write Access	Type Specific Information	
Cal_RT_Date	**DATE**	Jan 1 1970 00:00:00	Config	Block	High Limit Low Limit	Jan 19 2038 00:00:00 Jan 1 1970 00:00:00 Note 1
CJC_Type	**ENUM**	Intern (0)	Config	Config	Senses	See Parameter Description
CJC_Val	**REAL**	0.0	Config	Block	High Limit Low Limit	10,000°C - 273°C
Default_PV	**REAL**	0.0	Config	Config	High Limit Low Limit	100,000 -100,000
Filter_Time	**TIME**	0	Super	Super	]High Limit Low Limit	1d_3h_46m_40s _000ms 0
Input_Type	**ENUM**	Range_1 (0)	Config	Config	Senses	See Parameter Description
IO_Address	**IO_Address**		Super	Super		
Lin_Type	**ENUM**	None (0)	Config	Config	Senses	See Parameter Description
Post_Offset	**REAL**	0.0	Config	Config	High Limit Low Limit	100,000 -100,000
Post_Scaler	**REAL**	1.0	Config	Config	High Limit Low Limit	100,000 -100,000
Pre_Offset	**REAL**	0.0	Config	Config	High Limit Low Limit	100,000 -100,000
Pre_Scaler	**REAL**	1.0	Config	Config	High Limit Low Limit	100,000 -100,000
Process_Val	**REAL**	0.0	Oper	Block	High Limit Low Limit	PV_Max PV_Min
PV_Max	**REAL**	0.0	Config	Config	High Limit Low Limit	100,000 -100,000
PV_Min	**REAL**	0.0	Config	Config	High Limit Low Limit	100,000 -100,000
Status	**BOOL**	NOGO (0)	Oper	Block	Senses	NOGO (0) Go (1)

Table 6-4 Analog_In Parameter Attributes (continued)

Name	**Type**	**Cold Start**	**Read Access**	**Write Access**	**Type Specific Information**	
Status_Info	**ENUM**	Reset (0)	Oper	Block	Senses	See Parameter Description
Test_Enable	**BOOL**	Off (0)	Config	Config	Senses	Off (0) On (1)
Test_PV	**REAL**	0.0	Config	Config	High Limit Low Limit	PV_Max PV_Min
Test_Status	**BOOL**	NOGO (0)	Config	Config	Senses	NOGO (0) Go (1)

Table 6-4 Analog_In Parameter Attributes (continued)

Note 1: Calibration dates are not currently supported.

xFAST_AN_I FUNCTION BLOCK (Not for new designs)

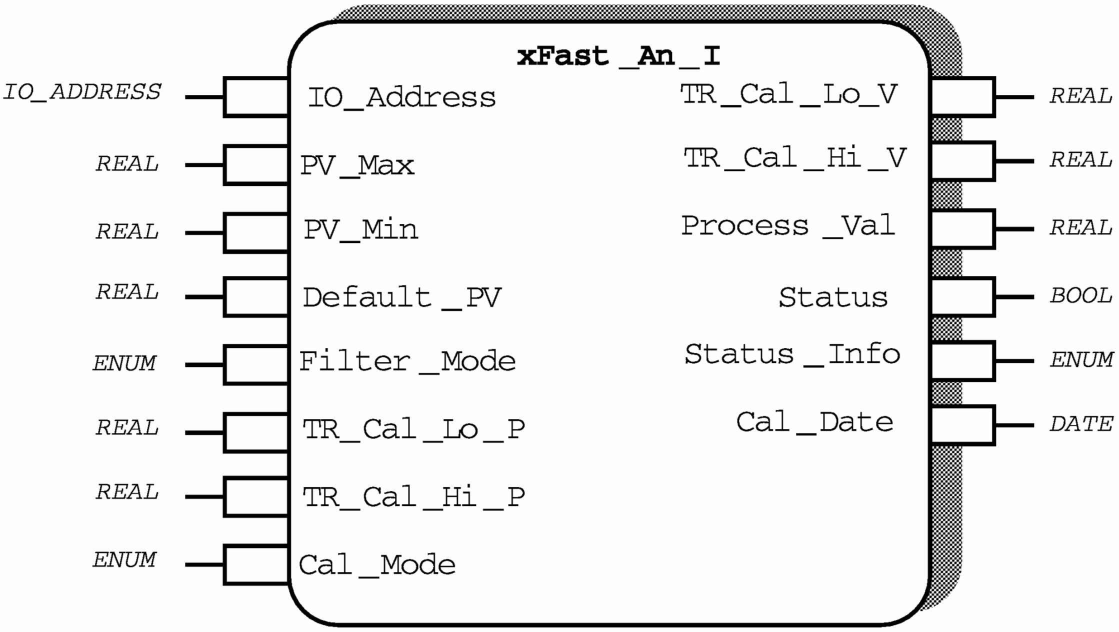


Figure 6-4 xFast_An_I Function Block Diagram

Functional Description

The xFast_An_I function block provides the interface to any hardware which can support a Fast Analogue Input channel., for example, the fast analogue IO module (Fast_An_8).

It is suitable for situations where an analogue input is to be processed at scan rates less than 100 ms.

Note that the x_I_Fast_An_I block should be considered in order to reduce the floating point arithmetic overhead for scan rates less than 50 ms.

Function Block Attributes

Type:	10 30
Class:	INPUTS
Default Task:	Task_2
Short List:	Process_Val, Status, Status_Info
Memory Required:	86 Bytes

Parameter Descriptions

Calibration Parameters

The parameters listed below are used for calibration of the hardware module.

TR_Cal_Lo_P

TR_Cal_Hi_P

Cal_Mode

TR_Cal_Lo_V

TR_Cal_Hi_V

Cal_Date

IO_Address (IOA)

IO_Address associates the function block instance with the physical connections on the hardware module to which it is referenced. Its value is assigned automatically when the function block instance is defined. Its value takes the form X:YY:ZZ, where X represents the number of the rack in which the module is resident, YY represents the number of the slot in the rack and ZZ represents the number of the channel within the module. For example, 1:02:03 would mean that the function block instance references the third channel of a module which sits in the second slot of the first rack of the PC3000 system.

PV_Max (PMX)

PV_Max is used to define the upper limit of the Process_Val. If the input from the hardware is greater than this value, Status will be set to NOGO (0), Status_Info will be set to Over_R (6) and Process_Val will be set equal to Default_PV.

PV_Min (PMN)

PV_Min is the lower input of the input value. If the input from the hardware is less than PV_Min, Status will be set to NOGO (0), Status_Info will be set to Under_R (7) and Process_Val will be set to Default_PV.

Default_PV (DPV)

Default_PV is used to drive the Process_Val when Status is NOGO (0).

Filter_Mode (FM)

Filter_Mode defines the corner frequency of the two pole low pass filter which is employed by the hardware channel. The options available include 16 Hz (0), 32 Hz (1), 80 Hz (2) and 160 Hz (3).

Process_Val (PV)

Process_Val reflects the analogue value input to the hardware channel addressed by the function block.

Status (ST)

Status reflects the status of the hardware channel being addressed by the function block.

Status_Info (STI)

Status_Info is a diagnostic parameter which explains the state of Status. It can have eight possible states:

Reset (0): The user program is not running

Ok (1): The Status is Go (1)

No_Mod (2): There is no module in the hardware slot being addressed by the function block.

Wrg_Mod (3): An incorrect module type has been fitted in the slot addressed by the function block

Calib (4): The module is currently in calibration mode.

Init (5): The module or channel is currently being initialised.

Over_R (6): The analogue value input to the hardware channel is greater than PV_Max.

Under_R (7): The analogue value input to the hardware channel is less than PV_Min.

Parameter Attributes

Name	Type	Cold Start	Read Access	Write Access	Type Specific	
Cal_Date	**DATE**		Config	Config		
Cal_Mode	**ENUM**	Run (0)	Config	Config	Senses	Run (0) Save (1) Hcal_Hi (2) Hcal_Lo (3) Tcal_Hi (4) Tcal_Lo (5) Dflt_TR (6)
Default_PV	**REAL**	0	Config	Config	High Limit Low Limit	110 -110
Filter_Mode	**ENUM**	16Hz (0)	Oper	Oper	EnumeratedValues	16Hz (0) 32 Hz(1) 80Hz (2) 160Hz (3)
IO_Address	**IO_ADDRESS**		Config	Config		
Process_Val	**REAL**	0 %	Oper	Oper	High Limit Low Limit	Set by PV_Max Set by PV_Min
PV_Max	**REAL**	110	Config	Config	High Limit Low Limit	110 -110
PV_Min	**REAL**	0	Config	Config	High Limit Low Limit	110 -110
Status	**BOOL**	Go (1)	Oper	Oper	Senses	NOGO (0) Go (1)
Status_Info	**ENUM**	Ok (1)	Oper	Oper	Senses	See Parameter Description
TR_Cal_Hi_P	**REAL**	100 %	Config	Config	High Limit Low Limit	110 TR_Cal_Lo_P
TR_Cal_Hi_V	**REAL**	100 %	Config	Config	High Limit Low Limit	110 -110
TR_Cal_Lo_P	**REAL**	0 %	Config	Config	High Limit Low Limit	TR_Cal_Hi_P -110
TR_Cal_Lo_V	**REAL**	0 %	Config	Config	High Limit Low Limit	110 -110

Table 6-5 xFast_An_I Parameter Attributes

xI_FAST_AN_I FUNCTION BLOCK (Not for new designs)

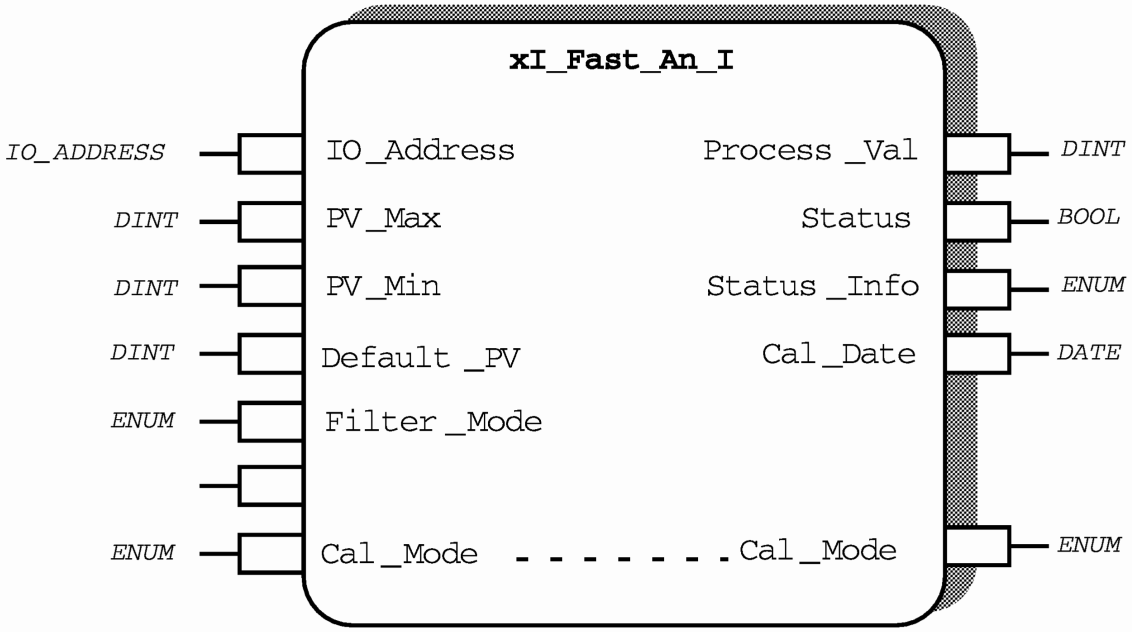


Figure 6-5 xl_Fast_An_I Function Block Diagram

Functional Description

The xI_Fast_An_I function block provides the interface to any hardware which can support a Fast Analogue Input channel. E.g. the fast analogue IO Module (FAST_AN_8). The xI_Fast_An_I function block is identical to the Fast_An_In function block except that it is purely integer and has no transducer calibration features. It represents a lower performance overhead than Fast_An In and Analogue_In and should be used where a high speed analogue input is required, i.e. scan rates less than 100ms.The integer output has a range of -2000 to +2000 to represent an input range of -10 volts to +10 volts.

Function Block Attributes

Type:..................................	10 38
Class:..................................	INPUTS
Default Task:	Task_2
Short List:	Process_Val, Status, Status_Info
Memory Required:..............	86 Bytes

Parameter Descriptions

Calibration Parameters

The parameters listed below are used for calibration of the hardware module.

Cal_Mode

Cal_Date

IO_Address (IOA)

IO_Address associates the function block instance with the physical connections on the hardware module to which it is referenced. Its value is assigned automatically when the function block instance is defined. Its value takes the form X:YY:ZZ, where X represents the number of the rack in which the module is resident, YY represents the number of the slot in the rack and ZZ represents the number of the channel within the module. For example, 1:02:03 would mean that the function block instance references the third channel of a module which sits in the second slot of the first rack of the PC3000 system.

PV_Max (PMX)

PV_Max is used to define the upper limit of the Process_Val. If the input from the hardware is greater than this value, Status will be set to NOGO (0), Status_Info will be set to Over_R (6) and Process_Val will be set equal to Default_PV.

PV_Min (PMN)

PV_Min is the lower input of the input value. If the input from the hardware is less than PV_Min, Status will be set to NOGO (0), Status_Info will be set to Under_R (7) and Process_Val will be set to Default_PV.

Default_PV (DPV)

Default_PV is used to drive the Process_Val when Status is NOGO (0).

Filter_Mode (FM)

Filter_Mode defines the corner frequency of the two pole low pass filter which is employed by the hardware channel. The options available include 16 Hz (0), 32 Hz (1), 80 Hz (2) and 160 Hz (3).

Process_Val (PV)

Process_Val reflects the analogue value input to the hardware channel addressed by the function block.

Status (ST)

Status reflects the status of the hardware channel being addressed by the function block.

Status_Info (STI)

Status_Info is a diagnostic parameter which explains the state of Status. It can have eight possible states:

Reset (0): The user program is not running

Ok (1): The Status is Go (1)

No_Mod (2): There is no module in the hardware slot being addressed by the function block.

Wrg_Mod (3): An incorrect module type has been fitted in the slot addressed by the function block

Calib (4): The module is currently in calibration mode.

Init (5): The module or channel is currently being initialised.

Over_R (6): The analogue value input to the hardware channel is greater than PV_Max.

Under_R (7): The analogue value input to the hardware channel is less than PV_Min.

PULSE INPUT MODULE MK2

OVERVIEW

Count Channel 1

i1 Count input when in normal count mode, or Phase A count input in quadrature count mode

D1 Phase B count input in quadrature mode

0V Common

Note that all the connections for this channel share the same 'common' connection which is internally connected to the common for Channel 2.

All inputs have switching levels of 30% and 70% of the encoder supply voltage, which in turn depends upon the position of the user-selectable jumper which switches between 5V and 12V. The thresholds have built-in hysteresis.

Count Channel 2

i2 Count input when in normal count mode, or Phase A count input in quadrature count mode

D2 Phase B count input in quadrature mode

0V Common

Note that all the connections for this channel share the same 'common' connection, which is internally connected to the common for Channel 1.

All inputs have switching levels of 30% and 70% of the encoder supply voltage, which in turn depends upon the position of the user-selectable jumper which switches between 5V and 12V. The thresholds have built-in hysteresis. It is not possible to individually select threshold levels for each channel.

Encoder Supply

V+ Encoder Supply with 12V @ 150mA or 5V @ 500mA capability, including short-circuit protection.

12V/5V Configuration

The PIM Mk2 module is supplied as standard with 12V logic thresholds on all inputs, as well as a 12V encoder power supply. The encoder interfaces are disabled, and only the count inputs, (marked as i1 and i2) are used.

If 5V operation is required, then the module should be withdrawn from the rack, and the plastic covers removed. The 0.1" jumper should then be removed from LK2, which is situated alongside the module earthing point at the front of the module, on its bottom edge. No other configuration is necessary.

Status Indication

There are 16 LEDs at the front of the PIM, arranged as part of the normal 4x4 matrix used on other PC3000 modules.

Assuming that LED 1 is top left when the board is in the rack, and working from left to right, and from top to bottom, the LED functions are as follows:

LED	Function	Colour
1	Module fault indication	Red
2	SIBUS communications active	Green
3	Module run indication	Green
4	Diagnostic LED	Green
5	Channel 1 count input active	Yellow
6	Channel 1 direction input active	Yellow
7	Channel 2 count input active	Yellow
8	Channel 2 direction input active	Yellow
9		Yellow
10		Yellow
11		Yellow
12		Yellow
13		Yellow
14		Yellow
15		Yellow
16		Yellow

The LEDs reflect the logic levels found at the plant terminals, i.e. if the level is high, the LED is ON, if the level is low, the LED is OFF. Thus, if nothing is connected to the module, all the LEDs will be illuminated.

PI_SMPLCTR FUNCTION BLOCK

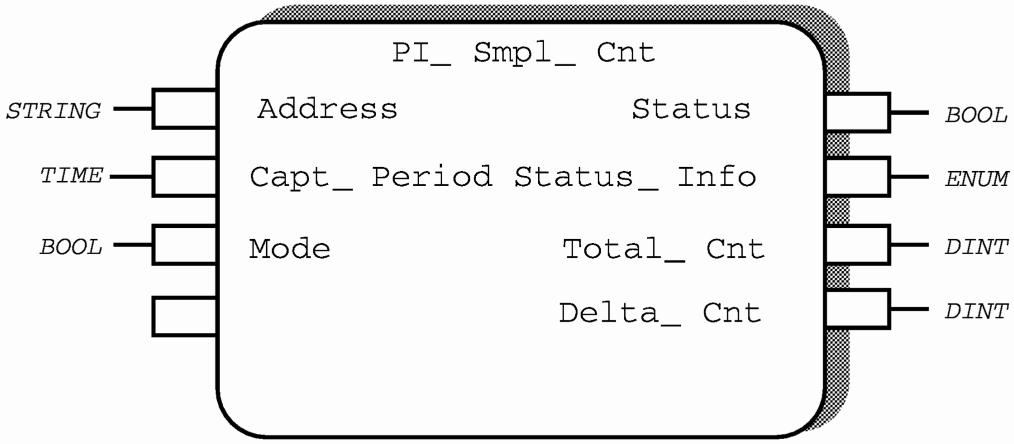


Figure 6-6 PI_Smpl_Ctr function block

Functional Description

The **PI_Smpl_Ctr** function block interfaces with the Pulse Input Module (PIM) to provide a pulse input counter. The module must be declared under 'MODULES'. This is a mechanism to read a count value at accurately sampled intervals, with the intervals being timed by the PIM. The function block outputs the total count value and the difference between the latest captured value and the previous captured value.

The **PI_Smpl_Ctr** function block receives its pulse information from the PIM via the PIM function block which has been assigned to the hardware module. The **PI_Smpl_Ctr** will only function if it has been declared in conjunction with a PIM function block and a properly configured PIM.

Function Block Attributes

Type....................................	10 E2
Class:	INPUTS
Default_Task:	Task_2
Short List:	Total_Cnt, Delta_Cnt, Mode, Status
Memory Requirements:	42 Bytes

Parameter Descriptions

Address (ADD)

Rack, slot and channel address as per normal PC3000 I/O module address conventions.

E.g. **2:7:2** refers to the second channel of the module in the seventh slot in the second rack.

Capt_Period (CP)

The Capt_Period is the time between successive internal count captures upon which the **Delta_Count** and **Frequency** parameter values are calculated. This parameter must be set to at least twice the task time and preferably to at least 1 second.

Mode (M)

This controls the active/inactive state of the counting channel.

Possible states:

Reset	(0)	The channel is inactive and all parameters are reset.
Run	(1)	The channel is active.

Status (S)

This gives the current status of the channel function block.

Possible states:

Nogo	(0):	There is a fault which is preventing normal operation of the channel function block. More information is given in the **Status_Info** parameter.
Go	(1):	The channel function block is operating normally.

Status_Info(S)

This lists the possible reason for a status of Nogo.

Possible states:

Reset	(0):	The user program is not currently running.
OK	(1)	There is no fault condition, Status is OK.
Mod_Err	(2)	There is a module-related error. The error is specified in the module function block.
Bad_Add	(3)	An invalid or non-existent address has been specified.
Dup_Add	(4)	An address has been used twice in the user program.

Total_Cnt (TC)

This parameter gives the 24 bit absolute count value stored in the module. It is always positive, and wraps around between 16777216 and 0, when a roll-over occurs.

Delta_Cnt

This parameter gives the latest interval count, either positive or negative, as passed from the module. The interval is that specified in the **Capt_Period** parameter. Note that the module continually updates this value, and thus any overruns in the user program execution will not result in a module lock-up condition.

Parameter Attributes

Name	Type	Cold Start	Read Access	Write Access	Type Specific Information	
Address	**STRING**		Oper	Oper		
Capt_Period	**TIME**	0	Oper	Oper	High Limit Low Limit	10s 100ms
Mode	**BOOL**	Reset (0)	Oper	Oper	Senses	Reset (0) Run (1)
Status	**BOOL**		Oper	Oper	Senses	Nogo (0) Go (1)
Status_Info	**ENUM**		Oper	Oper	Senses	Reset 0) OK (1) Mod_Err (1) Bad_Add (2) Bad_Add (3) Dup_Add (4)
Total_Cnt	**DINT**	0	Oper	Oper	High Limit Low limitt	16777216 0
Delta_Cnt	**DINT**	0	Oper	Oper	High Limit Low limit	16777216 -16777216

Table 6-7 PI_Smpl_Ctr Parameter Attributes