Advantys Reflex Actions Reference Guide

890USE18300 Version 1.0

31004635 00

a brand of Schneider Electric

Telemecanique

Table of Contents

Safety Information

Important Information

NOTICE

Read these instructions carefully, and look at the equipment to become familiar with the device before trying to install, operate, or maintain it. The following special messages may appear throughout this documentation or on the equipment to warn of potential hazards or to call attention to information that clarifies or simplifies a procedure.

The addition of this symbol to a Danger or Warning safety label indicates that an electrical hazard exists, which will result in personal injury if the instructions are not followed.

This is the safety alert symbol. It is used to alert you to potential personal injury hazards. Obey all safety messages that follow this symbol to avoid possible injury or death.

DANGER

DANGER indicates an imminently hazardous situation, which, if not avoided, **will result** in death, serious injury, or equipment damage.

WARNING

WARNING indicates a potentially hazardous situation, which, if not avoided, **can result** in death, serious injury, or equipment damage.

CAUTION

CAUTION indicates a potentially hazardous situation, which, if not avoided, **can result** in injury or equipment damage.

PLEASE NOTE

All pertinent state, regional, and local safety regulations must be observed when installing and using this product. For reasons of safety and to ensure compliance with documented system data, only the manufacturer should perform repairs to components.

When controllers are used for applications with technical safety requirements, please follow the relevant instructions.

No responsibility is assumed by Schneider Electric for any consequences arising out of the use of this material. This document is not intended as an instruction manual for untrained persons.

About the Book

At a Glance

Document Scope

This manual describes the individual reflex actions supported by the Advantys configuration software. It describes the configuration requirements for each action and gives illustrative examples.

Validity Note

The data and illustrations found in this book are not binding. We reserve the right to modify our products in line with our policy of continuous product development. The information in this document is subject to change without notice and should not be construed as a commitment by Schneider Electric.

Related Documents

Title of Documentation	Reference Number
The Advantys STB System Planning and Installation Guide	890USE17100
The Advantys STB System Hardware Components Reference Guide	890USE17200
The Advantys STB Profibus DP Network Interface Applications Guide	890USE17300
The Advantys STB INTERBUS Network Interface Applications Guide	890USE17400
The Advantys STB DeviceNet Network Interface Applications Guide	890USE17500
The Advantys STB CANopen Network Interface Applications Guide	890USE17600
The Advantys STB Ethernet TCP/IP Modbus Network Interface Applications Guide	890USE17700
The Advantys STB Modbus Plus Network Interface Applications Guide	890USE17800
The Advantys STB Fipio Network Interface Applications Guide	890USE17900
The Advantys Configuration Software Quick Start User Guide	890USE18000

Product Related Warnings

All pertinent state, regional, and local safety regulations must be observed when installing and using this product. For reasons of safety and to ensure compliance with documented system data, only the manufacturer should perform repairs to components.
When controllers are used for applications with technical safety requirements, please follow the relevant instructions.
Failure to use Schneider Electric software or approved software with our hardware products may result in injury, harm, or improper operating results.
Failure to observe this product related warning can result in injury or equipment damage.

User Comments

We welcome your comments about this document. You can reach us by e-mail at TECHCOMM@modicon.com

Introduction to Reflex Actions

1

At a Glance

Overview

This chapter describes the general features and functions of the Advantys reflex actions. It lists the types and variations of reflex blocks that can be created using the Advantys configuration software and explains how two blocks may be combined in a nested reflex action.

What's in this Chapter?

This chapter contains the following topics:

What Is a Reflex Action?

Summary

Reflex actions are small routines that perform dedicated logical functions directly on the Advantys island bus. They allow output modules on the island to act on data and drive field actuators directly, without requiring the intervention of the fieldbus master.

A typical reflex action comprises one or two function blocks that perform:

- Boolean AND or exclusive-OR operations
- comparisons of an analog input value to user-specified threshold values
- up- or down-counter operations
- timer operations
- the triggering of a latch to hold a digital value high or low
- the triggering of a latch to hold an analog value at a specific value

The island bus optimizes reflex response time by assigning the highest transmission priority to its reflex actions. Reflex actions take some of the processing workload off the fieldbus master, and they offer a faster, more efficient use of system bandwidth.

How Reflex Actions Behave

Reflex actions are designed to control outputs independently of the fieldbus master controller. They may continue to turn outputs on and off even when power is removed from the fieldbus master. Use prudent design practices when you use reflex actions in your application.

WARNING

UNEXPECTED OUTPUT OPERATION.

For outputs that are configured to respond to reflex actions, the output state represented in the island's network interface module (NIM) may not represent the actual states of the outputs.

- Turn off field power before you service any equipment connected to the island.
- For digital outputs, view the echo register for the module in the process image to see the actual output state.
- For analog outputs, there is no echo register in the process image. To view an actual analog output value, connect the analog output channel to an analog input channel.

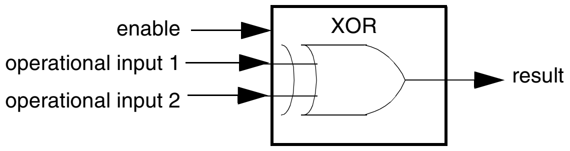
Failure to follow this precaution can result in death, serious injury, or equipment damage.

Configuring a Reflex Action

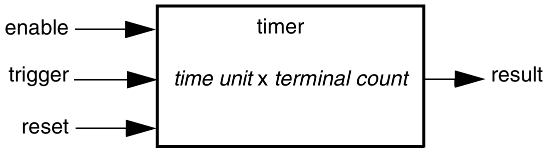
Each block in a reflex action must be configured using the Advantys configuration software.
Each block must be assigned a set of inputs and a result. Some blocks also require that you specify one or more user-preset values—a compare block, for example, requires that you preset threshold values and a delta value for hysteresis.

Inputs to a Reflex Action

The inputs to a reflex block include an enable input and one or more operational inputs.The inputs may be constants or they may come from other I/O modules on the island, from virtual modules or outputs from another reflex block. For example, an XOR block requires three inputs—the enable and two digital inputs that contain the Boolean values to be XORed:



Some blocks, such as the timers, require reset and/or trigger inputs to control the reflex action. The following example shows a timer block with three inputs:



The trigger input starts the timer at 0 and accumulates *time units* of 1, 10, 100 or 1000 ms for a specified number of counts. The reset input causes the timer accumulator to be reset.
An input to a block may be a Boolean value, a word value, or a constant, depending on the type of reflex action it is performing. The enable input is either a Boolean or a constant *always enabled* value. The operational input to an block such as a digital latch must always be a Boolean, whereas the operational input to an analog latch must always be a 16-bit word.
You will need to configure a source for the block's input values. An input value may come from an I/O module on the island or from the fieldbus master via a virtual module in the NIM.

Note: All inputs to a reflex block are sent on a change-of-state basis. After a change-of-state event has occurred, the system imposes a 10 ms delay before it accepts another change of state (input update). This feature is provided to minimize jitter in the system.

Result of a Reflex Block

Depending on the type of reflex block that you use, it will output either a Boolean or a word as its result. Generally, the result is mapped to an *action module*, as shown in the following table:

Reflex Action	**Result**	**Action Module Type**
Boolean logic	Boolean value	digital output
integer compare	Boolean value	digital output
counter	16-bit word	first block in a nested reflex action
timer	Boolean value	digital output
digital latch	Boolean value	digital output
analog latch	16-bit word	analog output

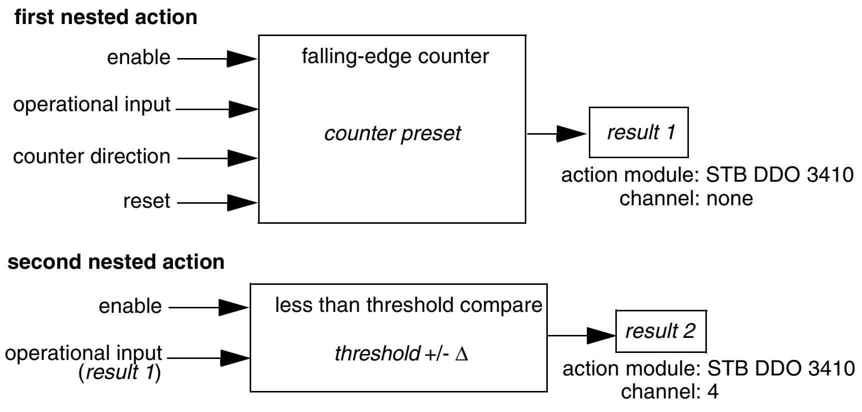
The result from a block is usually mapped to an individual channel on an output module. Depending on the type of result that the block produces, this action module may be an analog channel or a digital channel.
When the result is mapped to a digital or analog output channel, that channel becomes dedicated to the reflex action and can no longer use data from the fieldbus master to update its field device.
The exception is when a reflex block is the first of two actions in a nested reflex action.

Nesting

The Advantys configuration software allows you to create nested reflex actions. One level of nesting is supported—i.e., two reflex blocks, where the result of the first block is an operational input to the second block.

When you nest a pair of blocks, you need to map the results of both to the same action module. Choose the action module type that is appropriate for the result of the second block. This may mean that in some cases you will need to choose an action module for the first result that does not seem to be appropriate according to the table above.

For example, say you want to combine a counter block and a compare block in a nested reflex action. You want the result of the counter to be the operational input to the compare block. The compare block will then produce a Boolean as its result:



Result 2 (from the compare block) is the result that the nested reflex action will send to an actual output. Because the result of a compare block needs to be mapped to a digital action module, *result 2* is mapped to channel 4 on an STB DDO 3410 digital output module.

Result 1 is used only inside the module—it provides the 16-bit operational input to the compare block. It is mapped to the same STB DDO 3410 digital output module that is the action module for the compare block.

Instead of specifying a physical channel on the action module for *result 1*, the channel is set to *none*. In effect, you are sending *result 1* to an internal reflex buffer where it is stored temporarily until it is used as the operational input to the second block. You are not really sending an analog value to a digital output channel.

Number of Reflex Blocks on an Island

An island can support up to 10 reflex blocks. A nested reflex action consumes two blocks.
An individual output module can support up to two reflex blocks. Supporting more than one block requires that you manage your processing resources efficiently. If you are not careful with your resources, you may be able to support only one block on an action module.
Processing resources are consumed quickly when a reflex block receives its inputs from multiple sources (different I/O modules on the island and/or virtual modules in the NIM). The best way to preserve processing resources is to:

- use the *always enabled* constant as the enable input whenever possible
- use the same module to send multiple inputs to a block whenever possible

An Overview of Reflex Action Types

Summary

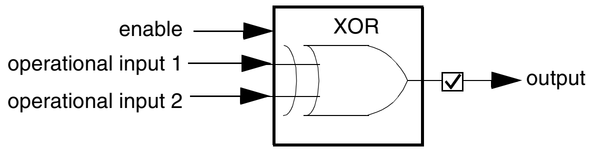
There are seven types of reflex blocks available in the Advantys configuration software:

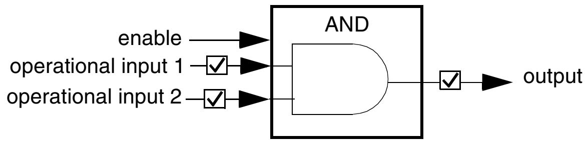
- Boolean logic blocks (See *Boolean Reflex Blocks, p. 41*)
- integer compare blocks (See *Integer Compare Reflex Blocks, p. 55*)
- unsigned compare blocks (See *Unsigned Compare Reflex Blocks, p. 71*)
- counter blocks (See *Counter Reflex Blocks, p. 93*)
- timer blocks (See *Timer Reflex Blocks, p. 107*)
- digital latches (See *Digital Latch Reflex Blocks, p. 147*)
- analog latches (See *Analog Latch Reflex Blocks, p. 129*)

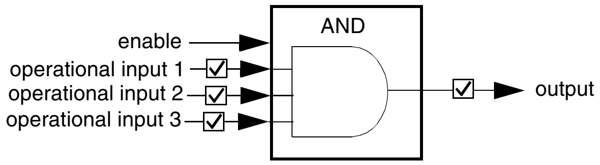
Each block supports a series of variations called *action types*.

Boolean Logic Action Types

Three fundamental Boolean logic action types are supported—the exclusive-OR (XOR) block, the two-input AND block, and the three-input AND block:







Boolean logic blocks require two types of inputs—an enable input and two or three operational inputs. All the inputs need to be digital (Boolean) values from sources that you must specify in the reflex editor. The output from any of these action types is also a Boolean value.
Notice the check boxes on the operational input lines to the AND blocks and on the output lines from all the Boolean blocks. When you place a check mark in one or more of these boxes, you invert the input or output value(s). When you invert an input to an action block, a value of 0 is treated as a 1 and a value of 1 is treated as a 0. In other words, you turn a Boolean false condition into a Boolean true condition or vice versa. If you invert the output from an XOR block, it becomes an XNOR action; if you invert the output from an AND block, it becomes a NAND action. Because of all the possibilities that can result from combinations of standard and inverted inputs and outputs, there are a large number of variations to the three basic Boolean action types. These variations are illustrated in truth tables.

Compare Action Types

A compare block takes a word as its operational input and compares that value against a predefined threshold value or a window of values. An integer compare block accepts operational inputs with integer values in the range -32 768 to +32 767. An unsigned compare block accepts operational inputs with integer values in the range 0 to 65 535.

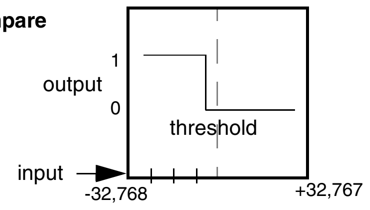
- Integer compare blocks generally take their operational inputs from Advantys STB analog input modules. Advantys analog modules use the IEC format for handling data. In this format, the most significant bit is always a dedicated sign bit, and the remaining 15 bits are able to represent values up to 32 767.
- Unsigned compare blocks generally take their operational inputs from virtual modules (See *The Virtual Module, p. 28*) or from the outputs produced by counter reflex actions (See *Nesting Two Reflex Blocks, p. 36*). These input sources produce unsigned values with 16-bit resolution (values as high as 65 535).

Integer compare blocks and unsigned compare blocks both support four action types:

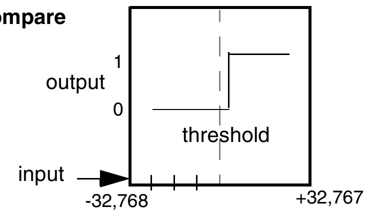
- less-than-threshold compares, where the output is a Boolean 1 when the operational input value is less than a user-defined threshold value
- greater-than-threshold compares, where the output is a Boolean 1 when the operational input value is greater than a user-defined threshold value
- inside-the-window compares, where the output is a Boolean 1 when the operational input value is within a range of values bounded by two user-defined thresholds
- outside-the-window compares, where the output is a Boolean 1 when the operational input value is outside a range of values bounded by two user-defined thresholds

The following illustration shows how the four action types compare the input to the thresholds, using the integer compare block as an example:

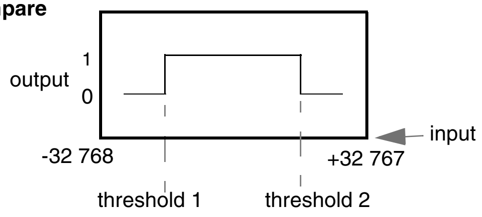
Less-than-threshold compare



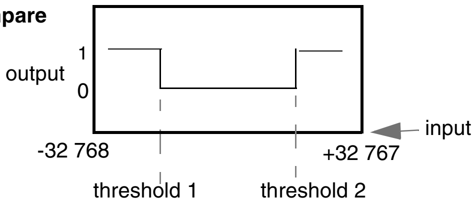
Greater-than-threshold compare



Inside-the-window compare



Inside-the-window compare



For all of the above action types, you may also specify a delta (Δ) value, which acts as an hysteresis around the threshold value(s).
The integer compare action types are described in *Integer Compare Reflex Blocks, p. 55*. The unsigned compare action types are described in *Unsigned Compare Reflex Blocks, p. 71*.

Counter Action Types

A counter block takes a series of digital inputs and accumulates a running count of the number of transitions either from 0 to 1 or from 1 to 0. You can configure the counter block to count up or down from a user-specified preset value. The output from the block is the current count—an unsigned integer value in the range 0 to 65 535.

Counter blocks support two action types:

- a rising-edge counter, where the counter increments or decrements each time the input value transitions from 0 to 1
- a falling-edge counter, where the counter increments or decrements each time the input value transitions from 1 to 0

Note: Counter blocks are different from other reflex actions in that they never map their output results to physical analog output channels. A counter block is designed to be coupled with an unsigned compare block in a nested reflex action (See *Nesting Two Reflex Blocks, p. 36*). The counter block is always the first block in the nested action, and its output is used as the operational input to the compare block.

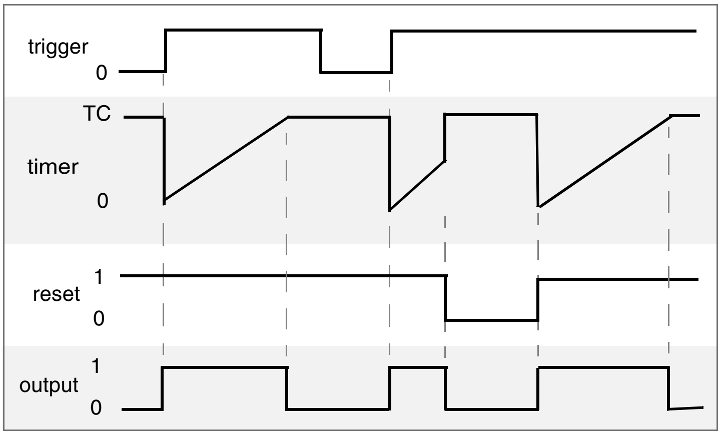
Timer Action Types

Timer blocks support four action types:
- delay-to-start timers
- delay-to-stop timers
- rising-edge timers
- falling-edge timers

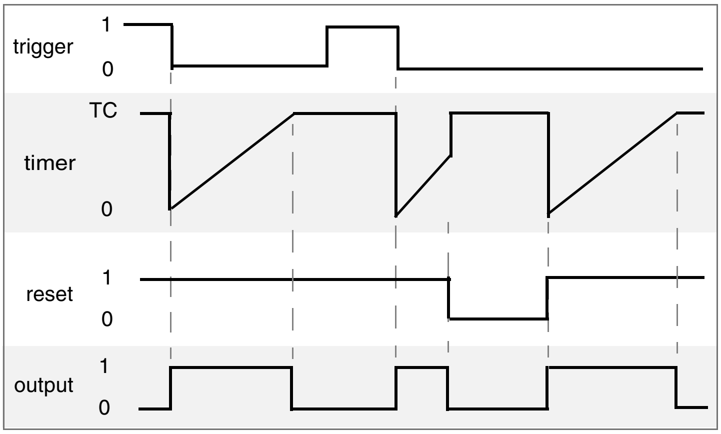
Timer blocks respond to a digital trigger input. A block begins accumulating time units on either the rising edge or falling edge of the trigger input and accumulates counts until it reaches a user-specified terminal count.

For a rising- or falling-edge timer, the accumulator holds the terminal count until the rising or falling edge of the trigger starts a new counting operation or until the block receives a reset input:

rising-edge timer

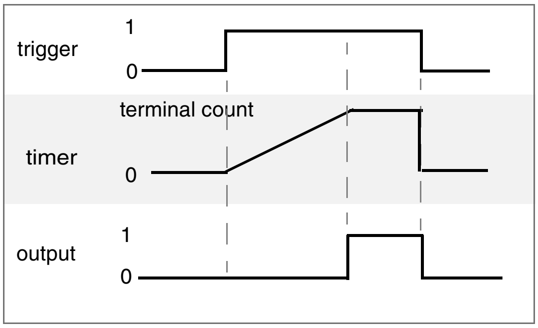


falling-edge timer

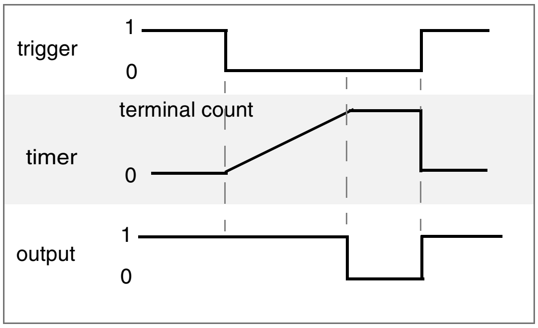


The output from a delay timer goes high or low when the timer reaches its terminal count and stays high or low while the terminal count is being held:

delay-to-start timer

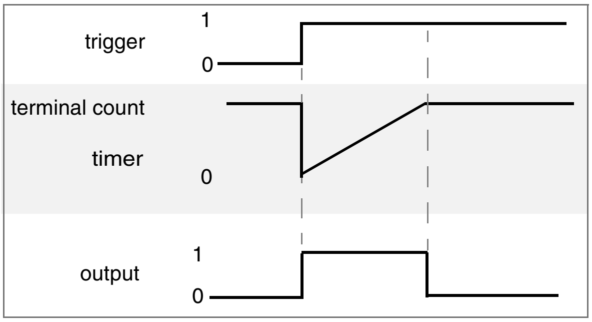


delay-to-stop timer

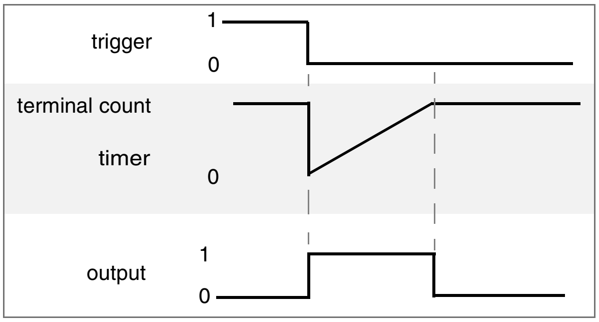


The output from an edge timer goes high while the timer is accumulating time counts and goes low when the terminal count is reached:

rising-edge timer



falling-edge timer



The outputs from all four timer action types may be inverted. When you invert an output from an action block, a value of 0 is treated as a 1 and a value of 1 is treated as a 0—in other words, you turn a Boolean false condition into a Boolean true condition or vice versa.

Latch Types

Latch blocks respond to a digital trigger input by latching to an operational input value on either the rising edge or falling edge of the trigger input. The block produces an output that is equal to value of the input at the moment it was latched, and that output remains until the trigger latches another value on its rising or falling edge. The operational input may be either of Boolean values (digital latches) or word values (analog latches).

Digital and analog latches both support four action types:

- a falling-edge latch, where the block latches the output value to the value of the operational input at the moment when the trigger transitions from 1 to 0
- a rising-edge latch, where the block latches the output value to the value of the operational input at the moment when the trigger transitions from 0 to 1
- a low-level latch, where the block latches the output to the value of the operational input when the trigger is at 0 and unlatches the output when the trigger is at 1
- a high-level latch, where the block latches the output to the value of the operational input when the trigger is at 1 and unlatches the output when the trigger is at 0

When an output is unlatched, the value of the output echoes the value of the operational input.

The output from a digital latch may be inverted. The output from an analog latch cannot be inverted. When you invert an output from a digital latch block, a value of 0 is treated as a 1 and a value of 1 is treated as a 0—in other words, you turn a Boolean false condition into a Boolean true condition or vice versa.

Configuring a Reflex Block

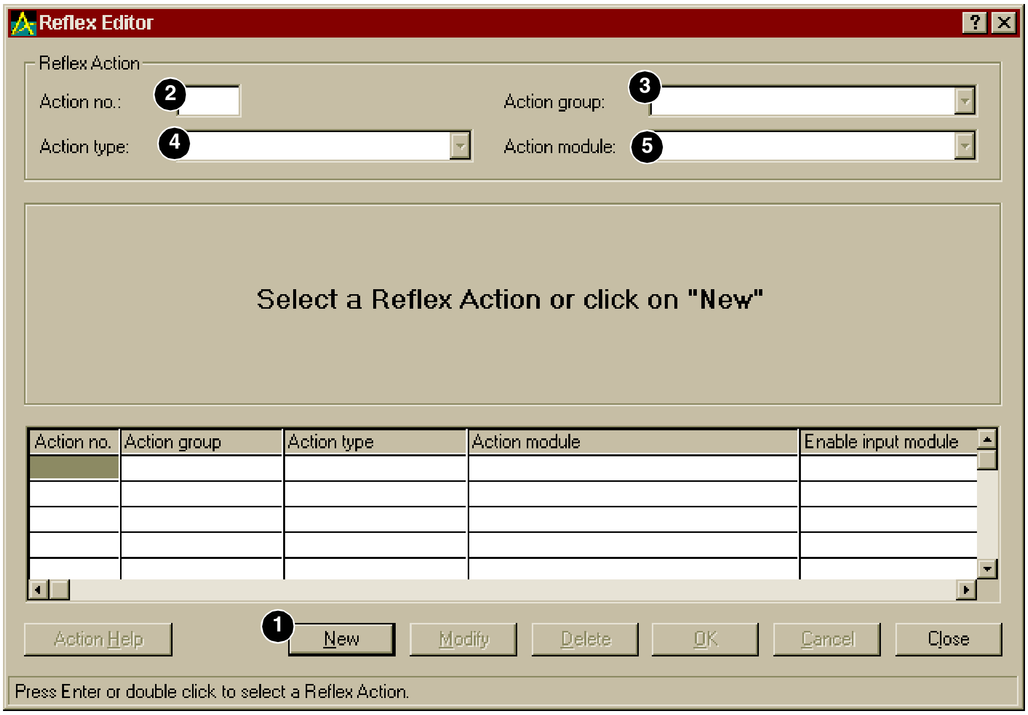
Summary

To create a reflex block and map it to an action module on your island bus, you need to use the reflex editor in the Advantys configuration software. The following procedure describes the basic parameters that need to be specified in the editor.

Opening the Reflex Editor

To open the reflex editor, click the following icon in the island toolbar:

The reflex editor will open in your workspace. If it is opening for the first time, the editor will look like this:



Select a Reflex Action or click on "New"

Action no.	Action group	Action type	Action module	Enable input module

Press Enter or double click to select a Reflex Action.

Note: If the **New** button (item 1 above) in the reflex editor is disabled, then the island selected in the workspace is locked.
To unlock the island, close the reflex editor and click the key icon on the island toolbar:

Some island configurations are password-protected. If the configuration on which you are working is protected, you will need to enter the password to unlock the island. If the configuration is not protected, it will unlock as soon as you click the key icon once.

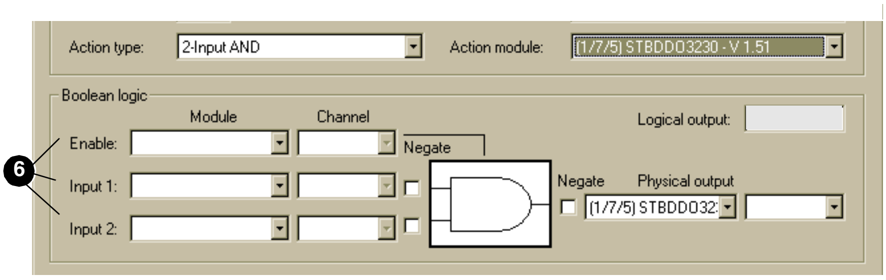
Defining the Reflex Block

The following steps describe how to select and define a reflex block and its action module in the reflex editor:

Step	Action	Result
1	Click the **New** button (item 1 above).	The **Action No.** field (item 2 above) fills with the number of the new reflex block, and the **Action group** field (item 3 above) is selected.
2	From the **Action group** pull-down menu, select one of the seven reflex blocks (See *An Overview of Reflex Action Types, p. 15*).	A block diagram appears in the center pane of the reflex editor with empty field for the inputs, outputs and any user-specified preset values.
3	From the **Action type** pull-down menu (item 4 above), select the appropriate block type.	
4	From the **Action module** pull-down menu (item 5 above), select an output module from your island bus configuration.	The module you specify here automatically appears in the **Physical output** list box in the block diagram in the center pane of the editor.
5	Now you can configure the action's input values and output destination.	All reflex blocks require a set of input values, a physical output and a logical output, as described in the following discussions.

Configuring the Inputs to a Reflex Block

Every block requires that you configure a set of input values. The block diagram that appears in the center pane of the reflex editor displays the input fields in a column on the right (as in item 6 below). The following example shows a two-input AND block:



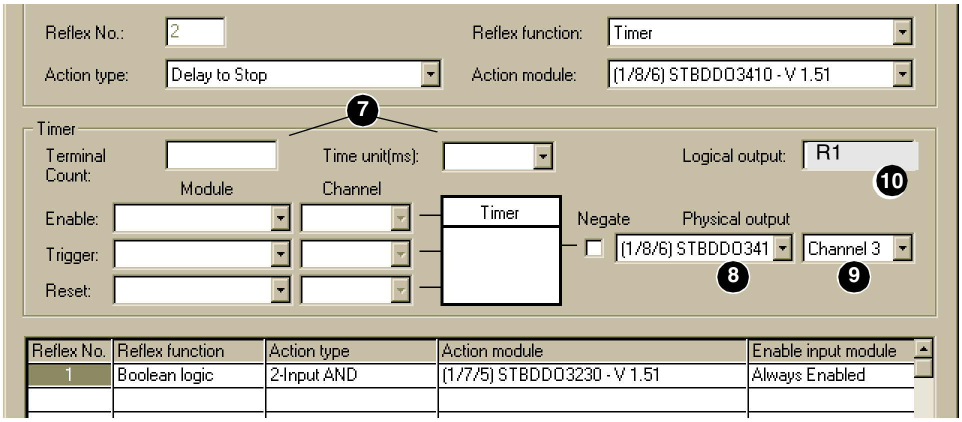
This example shows a block with three inputs—an **Enable** and two operational inputs (**Input 1** and **Input 2**). Each input has its own pull-down list, from which you will configure the source of each input. Generally, inputs can be derived from one of four sources:

- from another input module on the island bus
- from a constant value that you specify (e.g., always enabled, up-count direction)
- from the fieldbus master, in the form of the virtual module (See *The Virtual Module, p. 28*) or the action module (See *Using the Action Module as an Input to a Block, p. 32*)
- from a reflex block

Configuring Preset Values for a Reflex Block

Some reflex actions also have some user-specified preset values that you will need to configure. For example, a timer block requires a timing unit and a terminal count preset. When preset values are required, the reflex editor displays them above the reflex block (as in item 7 below).

The following example shows the block display for a delay-to-stop timer:



Reflex No.: 2
Reflex function: Timer
Action type: Delay to Stop
Action module: (1/8/6) STBDDO3410 - V 1.51
Timer
Terminal Count:
Time unit(ms):
Logical output: R1
Physical output: (1/8/6) STBDDO341 · Channel 3

Reflex No.	Reflex function	Action type	Action module	Enable input module
1	Boolean logic	2-Input AND	(1/7/5) STBDDO3230 - V 1.51	Always Enabled

Notice the two preset fields across the top of the block—a text box where you need to enter a **Terminal Count** value and a list box for selecting a **Time unit**.

Configuring the Physical Output from a Reflex Block

Notice in both examples above that the module listed in the **Physical output** field (item 8 above) is the module you chose as the **Action module**. The physical output module is always the action module. You then need to specify the channel on the action module to which the physical output will be written (as in item 9 above). You may select either an available channel on the action module or *None*:

- Choose a channel number if you want to map the output from the action to a real physical output. After you have configured this physical channel, it will be dedicated to the reflex action.
- Choose *None* only if you are configuring the first of two blocks in a nested reflex action. The output will be written to a temporary memory buffer, then used as an input to the second block in the nested action.

The Logical Output

The software automatically assigns the output a tag name, referred to as the *logical output*. The **Logical output** field appears above the **Physical output** fields (see item 10 above). The field contains a fixed-text box with an assigned name in the range R1 through R10.

The logical output is particularly useful in a nested reflex because the string from the first reflex action appears in the pull-down menus as an input channel selection for the inputs to the second reflex action.

The Virtual Module

Summary

Because reflex actions are designed to operate independently from the fieldbus master, inputs to the reflex blocks generally come from local input modules. In some applications, however, you may want the fieldbus master to provide an input value to a block. One way to do this is via the virtual module.

The Advantys configuration software provides three words in the output process image where the fieldbus master may write digital and/or analog values for use exclusively as inputs to the reflex actions. These three words comprise the *virtual module*.

Virtual Module Structure

If you choose to use the virtual module, it may be one, two or three words in length:

- If you want to use the virtual module only for digital inputs, the virtual module will be one word long. It provides 16 bits where the fieldbus master can write up to 16 digital inputs to the reflex actions.
- If you want to use the virtual module only for analog inputs, the virtual module will be two words long. It will provide two words where the fieldbus master can write up to two analog input values to the reflex actions.
- If you want to use the virtual module for both digital and analog inputs, the virtual module will be three words long. The first word provides 16 bits for digital inputs and the second and third words are for two analog input values to the reflex actions.

Note: If you look at the vitrtual module data in the Modbus View of the Advantys configuration software, the 16 bits of digital virtual data will be displayed in the low bytes of two separate registers. If you look at the virtual module data in the Fieldbus View, the 16 bits of digital virtual data may be displayed either in a single 16-bit word or in the low bytes of two contiguous words, depending on the fieldbus. The STB NMP 2212 Modbus Plus NIM and the STB NIP 2212 Ethernet NIM display virtual digital data the way it is displayed in the Modbus View.

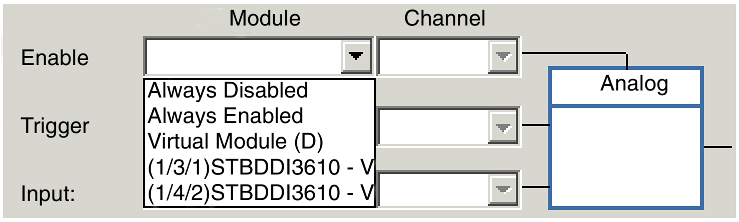
The word or words used for the virtual module are always the last words in the output process image. If all three words are used, the digital word will appear first followed by the two analog words.

Selecting the Virtual Module

The size of the virtual module in your process image is determined by your selection of inputs to the reflex actions in your island configuration.

For example, suppose you are setting up a falling-edge analog latch (See *Falling-edge Analog Latch Block, p. 130*). The latch has three inputs—an enable input, a latch trigger and an analog operational input. The enable input and the trigger need to be Boolean (digital) inputs, and the operational input needs to be an analog word.

When you are selecting the source for the enable and the trigger inputs, one of the choices that will appear in both input list boxes is *Virtual Module (D)*:



Always Disabled
Always Enabled
Virtual Module (D)
(1/3/1)STBDDI3610 - V
(1/4/2)STBDDI3610 - V

If you select an input called *Virtual Module (D)*, the virtual-module word for digital inputs becomes part of the output process image. This means that the fieldbus master needs to write to one of the virtual module's 16 available bits to control the enable input and/or the trigger input.

Suppose you are configuring the operational input to that falling-edge analog latch. The operational input must be an analog integer value.When you go to the list box to specify the source of the operational input, one of you choices will be *Virtual Module (A)*:

If you select *Virtual Module (A)* as the input, the two virtual-module words for analog inputs become part of the output process image. The fieldbus master needs to write an operational input to the first word in the analog latch block.

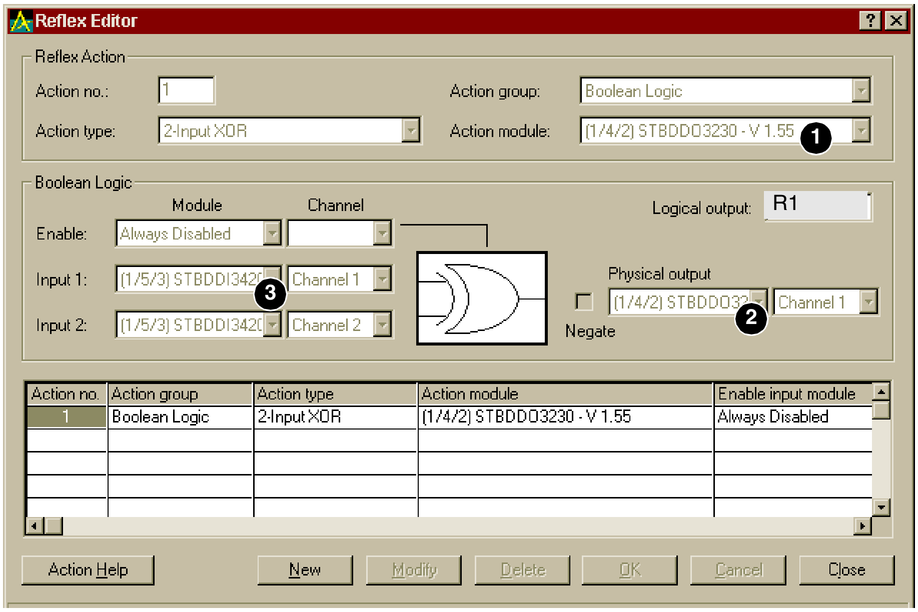
The Action Module

Summary

When you configure a reflex block, you must assign it to an *action module*. The action module is always one of the output modules in your island configuration. Usually there is a direct relationship between the action module that you select and the type of output that the reflex action will produce. If the reflex action produces a Boolean result as its output, the action module is generally a digital output module. If the reflex action produces an analog output, an analog output module is generally the action module.
The exception is when you nest two reflex blocks together. In that case, both actions need to have the same action module, and the action module type needs to match the output expected from the second block in the nested action.

Mapping a Reflex Output to a Physical Output

When you configure a reflex block to write its output to a field actuator, choose an action module and specify the channel on the action module that will send the output to the actuator.
For example, suppose you want to configure a Boolean XOR action that writes outputs to a field actuator connected to channel 1 on an STB DDO 3230 output module. This STB DDO 3230 module is located at address 2 on the island bus. Here is how the configuration might look in the reflex editor:



Reflex Action

Action no.:	1	Action group:	Boolean Logic
Action type:	2-Input XOR	Action module:	(1/4/2) STBDDO3230 - V 1.55

Boolean Logic

	Module	Channel
Enable:	Always Disabled	
Input 1:	(1/5/3) STBDDI342(	Channel 1
Input 2:	(1/5/3) STBDDI342(	Channel 2

Logical output: R1

Physical output: (1/4/2) STBDDO32 Channel 1

Negate

Action no.	Action group	Action type	Action module	Enable input module
1	Boolean Logic	2-Input XOR	(1/4/2) STBDDO3230 - V 1.55	Always Disabled

This action is designed to XOR the Boolean inputs produced on channel 1 and channel 2 of the STB DDI 3420 digital input module at address 3 on the island bus (item 3 above). The output from the action will be written to channel 1 on the STB DDO 3230 digital output module at address 2 on the island bus.

Item 1 above shows the action module selected from the list box. The entry lists three things:

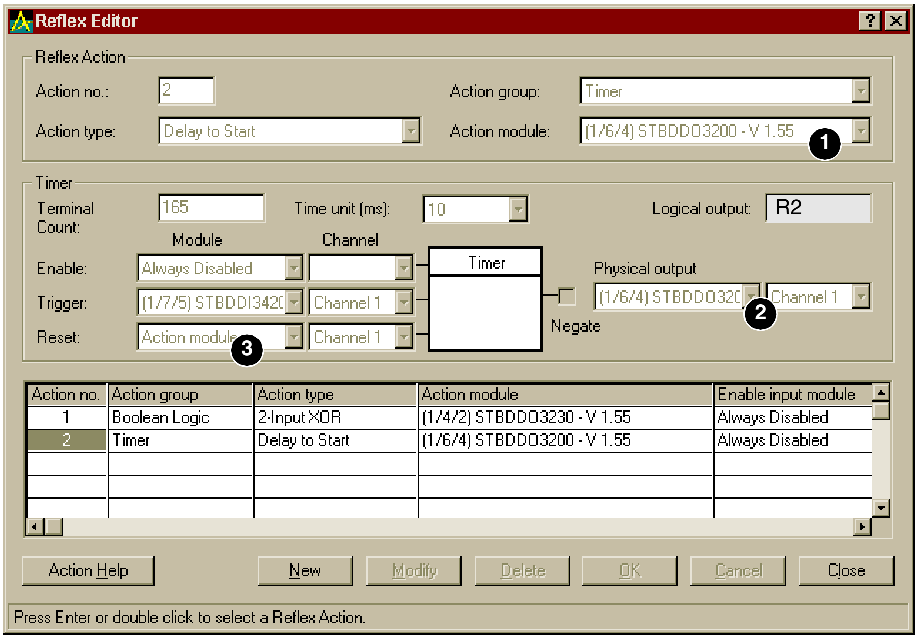
- the model number of the action module
- the version of the module (in this case, v 1.55)
- a position code (1/4/2)

The position code tells you that the STB DDO 3230 module that you have selected as your action module is located in the primary segment (1), at physical location 4 and logical address 2 on the island bus. The discrepancy between logical address 2 and the physical location 4 is caused by the physical presence on the island bus of the NIM and a PDM, two modules that do not have logical addresses.

Item 2 above indicates exactly where the configuration will map the reflex output. The physical output module is the action module. The action channel is selected from the list box on the right. Since an STB DDO 3230 is a two-channel output module, the channel choices in the drop-down list box are *None*, *Channel 1* and *Channel 2*. For this configuration, channel 1 has been selected as the action channel.

Using the Action Module as an Input to a Block

Once you have mapped the output from a block to a physical channel on the action module and downloaded the configuration, this channel becomes dedicated to that reflex action. The fieldbus master can no longer drive the physical output. However, the channel address is still present in the output process image, and the fieldbus master can write data to this address location. You may use this channel address to deliver fieldbus data as an input to a reflex block.
For example, suppose you want to configure a delay-to-start timer (See *Delay-to-start Timer Block, p. 108*), and you want the fieldbus master to provide the reset input to the reflex block. Here is how the configuration might look in the reflex editor:



Action no.	Action group	Action type	Action module	Enable input module
1	Boolean Logic	2-Input XOR	(1/4/2) STBDDO3230 - V 1.55	Always Disabled
2	Timer	Delay to Start	(1/6/4) STBDDO3200 - V 1.55	Always Disabled

Note: This is one way of preserving processing resources (See *Number of Reflex Blocks on an Island, p. 14*), since you are reusing resources from the same output module that is already involved in the reflex action.

The action module is specified as the STB DDO 3200 output module located at logical address 4 on the island bus (item 1 above). The physical output is mapped to channel 1 of the action module (item 2 above).
For efficiency, you may reuse the bit in the output process image previously assigned to channel 1 of the action module as the reset input. Functionally, this means that the fieldbus master will be able to reset (stop) the timer accumulator by writing a value of 0 to that data bit in the output process image. To make this happen, select *Action Module* from the Reset input pull-down list box (item 3 above), then select *Channel 1* as the Reset input channel.

Selecting None as the Physical Output Channel

In the two examples above, you always selected a channel (1, 2, 3, etc.) along with the action module as the physical output from the reflex action. Among the channel options in the pull-down list box is the entry *None*. Select this entry only when the action block you are configuring is the first block in a nested reflex action (See *Nesting Two Reflex Blocks, p. 36*).
When you select *None* as the physical output channel, the output from the block goes to a temporary storage buffer, and it can be used as an input to the second block in the nested reflex action.

How Action Modules Respond to Fallback Conditions

Fallback Conditions

Advantys STB output modules are designed to send their output data to a predictable *fallback* state in the event of a communications failure between the island and the fieldbus. In this state, output data is replaced with pre-configured fallback values so that a module's output data values are known when the system recovers from a communications failure.

Because reflex blocks are able to operate independently from the fieldbus master, there are some circumstances where the fallback scenario for an action module will be different from an output module that does not involve reflex actions. The following discussion points out these circumstances.

Action Module Behaviors

An action module is an Advantys STB output module that has at least one of its channels dedicated to the result of a reflex block. Typically, an action module will behave like any other output module on the island. It will send its output channels to their configured fallback states when communication between the island and the fieldbus master is lost.

An action module is not usually in a situation where its regular output channels are in their configured fallback states while a channel dedicated to the reflex action continues to operate. The exception is a two-channel action module where:
- each output channel supports an independent reflex block
- neither reflex block is receiving any inputs from the NIM (inputs to the reflex blocks are not coming from the virtual module (See *The Virtual Module, p. 28*) or from the action module itself (See *Using the Action Module as an Input to a Block, p. 32*))

If both these conditions are true, the action module will continue to run if communication between the island and the fieldbus master is lost. The green RDY LED on the action module will display a special three-blink pattern to indicate when this condition is occurring.

If the action module has more than two output channels, the behavior described above does not apply. An Advantys STB output module cannot be configured to support any more than two reflex blocks.

When Inputs Fail

If an input module on the island bus is providing an input to a reflex block and that input module loses sensor power from the PDM, the reflex block immediately acts upon a 0 value coming from that input. After a delay of up to 1.5 ms, the reflex action acknowledges that PDM power has been lost and puts the reflex channel in its fallback state.

Note: For more information about fallback conditions, refer to the output module descriptions in the *Advantys STB Hardware Reference Guide* (890 USE 172).

Nesting Two Reflex Blocks

Summary

The Advantys configuration software allows you to create one level of nesting for reflex actions. You can nest two reflex blocks, where the output from the first block is used as an operational input to the second block. Both reflex blocks must be nested within the same action module.

The Action Module

In a nested reflex action, the output from the first reflex block is used internally—as an operational input to the second reflex block. The output from the second reflex block is used to update the physical output channel of the *action module*.

When you nest a pair of reflex blocks, you need to map the outputs from both to the same action module. Choose the action module type that is appropriate for the output from the second nested action block. In some cases, this means that you may need to choose an action module for the first block that does not seem to be appropriate for its output.

For example, say you want to nest counter-compare action. To do this, you need to configure two action blocks with the reflex editor. The first block is the counter action (See *Counter Reflex Blocks, p. 93*), and the second block is an unsigned compare action (See *Unsigned Compare Reflex Blocks, p. 71*).

The output from a counter is always a 16-bit word value, and the output from the unsigned compare is always a binary (Boolean) value. Intuitively, you might assume that because the counter produces a word as its output it should be mapped to an analog action module. However, since the counter is the first block in the nested action and since the output from the second action—the unsigned compare—is a Boolean, you need to select a digital output module as the action module.

The Physical Outputs

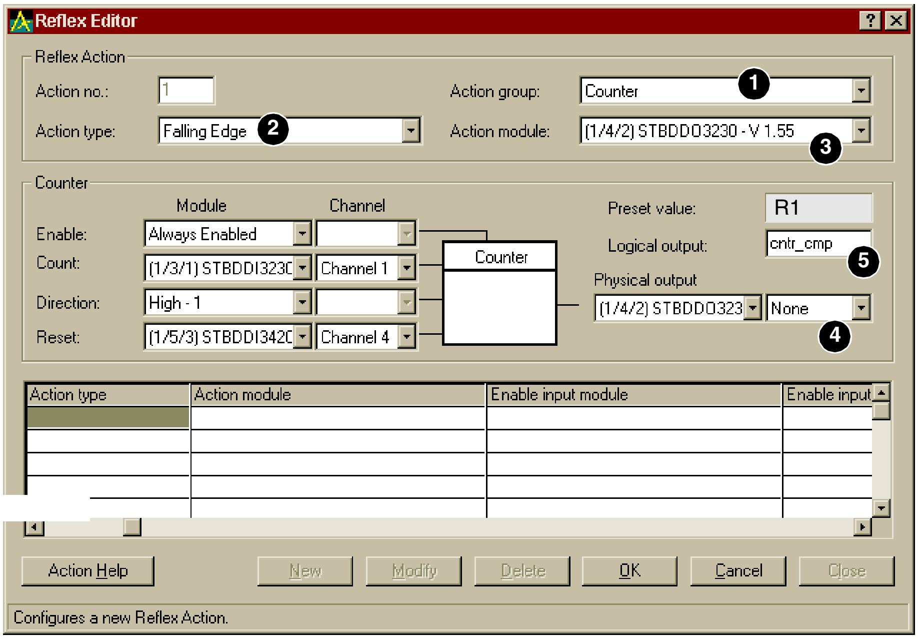
The reflex editor requires that you specify the physical and logical output of each reflex block that you configure. Generally, the physical output is the channel on the action module to which the output of the action will be written. The physical output is always mapped this way when an action is not part of a nesting; it is also how the output from the second action block in a nested action is mapped. For the first block in a nested action, however, the physical output is sent to a temporary memory buffer. Instead of specifying an output channel on the action module, you need to specify the physical output as *None*.

The Logical Outputs

The output from each block also needs to be assigned a logical output. The logical output is a tag name for the output—a text string between one and eight characters long. The characters may be any combination of standard keyboard characters—alpha numerics, underscores, and/or standard symbols (!,?, /, >, etc.).
The logical output can be particularly useful in a nested reflex because the text string of the first action block will appear in the pull-down menu as an input to the second action block.

A Counter-Compare Configuration

To clarify the process of configuring a nested action, let's look at the way you might configure the first of the two action blocks in the reflex editor of the Advantys configuration software:


Reflex Editor

Reflex Action

Action no.:	1	Action group:	Counter ❶
Action type:	Falling Edge ❷	Action module:	(1/4/2) STBDDO3230 - V 1.55 ❸

Counter

	Module	Channel
Enable:	Always Enabled	
Count:	(1/3/1) STBDDI323C	Channel 1
Direction:	High - 1	
Reset:	(1/5/3) STBDDI342C	Channel 4

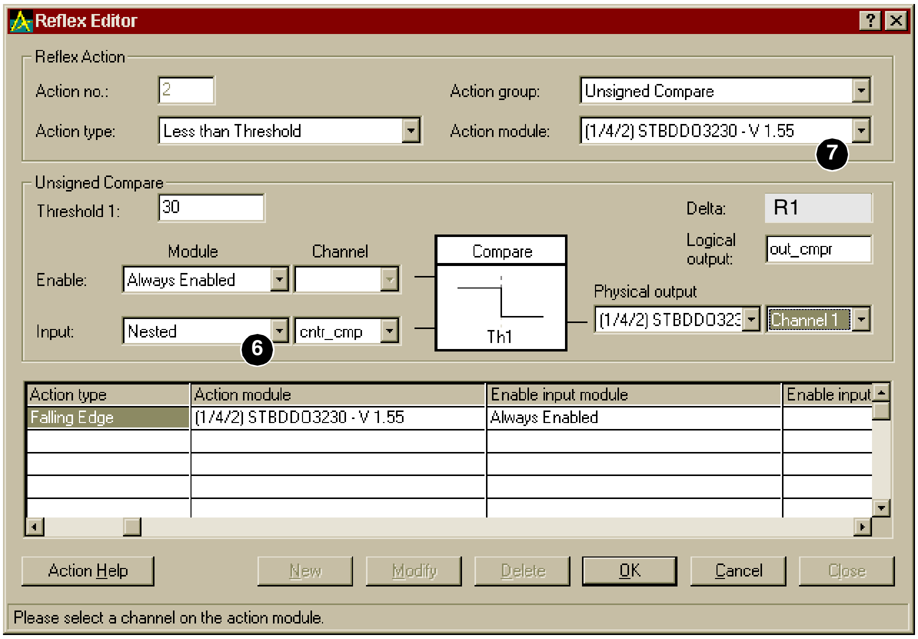
Preset value:	R1
Logical output:	cntr_cmp ❺
Physical output	(1/4/2) STBDDO323 / None ❹

Action type	Action module	Enable input module	Enable input

Configures a new Reflex Action.

Action no. 1 is a falling-edge counter, as items 1 and 2 above indicate. The action module is the STB DDO 3230 digital output module at island bus address 2 (item 3 above). The action module needs to be a digital output module because the ultimate result of the nested action will be Boolean.
Item 4 above shows the physical output channel on the action module as *None*. The output from the falling-edge counter is sent to a temporary memory buffer. The output value is then used as an operational input to the second block in the nested action.

The logical output string assigned to the falling-edge counter output (item 5 above) is *in_cmpr*. The logical output from the first block is used as the operational input to the second block, as shown in the following illustration:


Reflex Editor

Reflex Action

Action no.: 2

Action group: Unsigned Compare

Action type: Less than Threshold

Action module: (1/4/2) STBDDO3230 - V 1.55

Unsigned Compare

Threshold 1: 30

Delta: R1

Logical output: out_cmpr

	Module	Channel
Enable:	Always Enabled	
Input:	Nested	cntr_cmp

Physical output: (1/4/2) STBDDO323 Channel 1

Action type	Action module	Enable input module	Enable input
Falling Edge	(1/4/2) STBDDO3230 - V 1.55	Always Enabled	

Please select a channel on the action module.

Action no. 2 is an unsigned less-than-threshold compare block. Item 6 shows that the operational input to the compare block is *cntr_cmp*, the logical output from action no. 1. The action module (item 7 above) for the less-than-threshold compare block is the STB DDO 3230 digital output module at island bus address 2—the same action module as the one for the falling-edge counter.

Reflex Action Start-up States

Summary

All reflex blocks are initially at fallback when the island starts up after a power cycle or any other reconfiguration sequence. However, the fallback mode and fallback value applied to each output channel is the factory-default (*predefined* state, *off*), not the user-configured parameters downloaded with the configuration. The user-configured parameters are applied only after all inputs have been received and a condition that triggers fallback occurs.

After all the reflex inputs have been received (even with a status error), the reflex blocks will enter the Run state. If there is a status error, that reflex block output channel will enter fallback with the user-configured value.

Enable inputs have the same effect as normal inputs for the purposes of entering and leaving the fallback mode.

Consequences

Some of the consequences of this start-up state behavior are:

- If one or more peer input module(s) is/are missing, no reflex blocks in the module will run. The factory-default fallback mode and fallback values will remain in effect.
- Issuing a **Stop** and a **Run** command from the Advantys configuration software will reset the reflex blocks so that the output channels will start in their user-configured fallback modes and states.
- Removing a mandatory module from the island and then replacing it will also reset the reflex blocks so that the output channels will start in their user-configured fallback modes and states.

Note: In a two-channel module where both channels are used for enabled reflex blocks and where only peer inputs are used, removing and replacing the fieldbus cable will not have an effect on the reflex blocks (since the module stays operational while the rest of the modules go to their fallback states).

For reflex errors to be cleared (as indicated by the LED and the Advantys configuration software described below), all configured reflex blocks must be successfully executed. This requires that all input data be present (without any status errors) and that the *enable* input is high at least once during the period when all input data are present.

Reflex Action LED Error State

When a reflex block is in error or is not running because all its inputs have not been received, the green RDY LED on the action module will blink in a special pattern—three blinks followed by a pause, repeatedly until the condition is cleared.
Reflex errors are also indicated by emergency messages and emergency error codes. These errors appear in the Advantys configuration software as a node error (error register = 0*x*80 in the I/O module diagnostics window).

Enable Behavior

If the enable input to a reflex block is the Always Enabled constant, the block always becomes operational immediately at start-up.
If the enable input to a reflex block is the Always Disabled constant, the block always starts up in its fallback state. The action module flashes the fallback LED pattern described above for the action channel. Other nonreflex outputs remain operational. The module sends an emergency message indicated by the *Node Error* diagnostics bit. If the Advantys configuration software is connected to the physical island, the module image in the island editor will flash in red. The error in that block will never clear. *Always Disabled* might be used while you are commissioning the island.
If the enable input to a reflex block is the signal from an input module on the island bus, the reflex block starts up in its fallback state when the input value is 0. The action module flashes the fallback LED pattern described above for the action channel. Other nonreflex outputs remain operational. The module sends an emergency message indicated by the *Node Error* diagnostics bit. If the Advantys configuration software is connected to the physical island, the module image in the island editor will flash in red. As soon as the enable input is sensed, the reflex block becomes operational, the LED goes on steady, and the module image in the island editor stops flashing in red.
If the enable input to a reflex block is either part of the action module or the digital virtual module, the reflex starts up in its fallback state when the input value is 0. The action module flashes the fallback LED pattern described above for the action channel. Other nonreflex outputs remain operational. The module sends an emergency message indicated by the *Node Error* diagnostics bit. If the Advantys configuration software is connected to the physical island, the module image in the island editor will flash in red. As soon as the enable input is sensed, the reflex block becomes operational, the LED goes on steady, and the module image in the island editor stops flashing in red.

Boolean Reflex Blocks

2

At a Glance

Overview

This chapter describes three Boolean logic reflex blocks—an exclusive-OR (XOR) and two logical ANDs. XOR blocks operate on two input values; AND blocks can operate on either two or three inputs. Because the software allows you to invert the results of these blocks and sometimes their operational inputs, several variations of the three block types are supported.

What's in this Chapter?

This chapter contains the following topics:

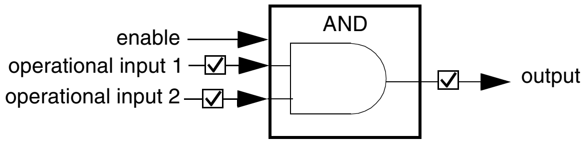
Two-input AND Blocks

Summary

A two-input AND block performs a logical AND operation on two Boolean operational inputs. The output is a Boolean true or false, expressed as a value of 1 or 0, respectively. You may invert the value(s) of one or both inputs. You may also invert the value of the output, in which case the action becomes a logical NAND.

Structure of a Two-input AND Block

A block diagram for a two-input AND is shown below:



The AND block has three inputs—one enable input and two operational inputs. The enable input turns the block on or off. The operational inputs send two Boolean values to the block. The inputs are ANDed together when the block is enabled, and the result is a Boolean output.
The checkboxes on the two input lines and the output line provide the mechanism by which one or more of the values can be inverted. When you click on one of these boxes, a check mark toggles on or off. When a box is checked, the value of the associated input or output is inverted—i.e., a 1 becomes a 0, a 0 becomes a 1.

Enable Input

An AND block can be enabled either by a Boolean 1 or an *always enabled* constant. It can be disabled by a Boolean 0 or an *always disabled* constant.
If the enable input is a Boolean, it may be produced by:
- a digital input from a module on the island
- a digital output from the virtual module (See *The Virtual Module, p. 28*)
- an output on the action module (See *Using the Action Module as an Input to a Block, p. 32*) written to by the fieldbus master

When the enable input is a Boolean 0 or an *always disabled* constant, the block is disabled—the action does not execute and the output is frozen in the state it was in when the block became disabled. The block continues to process inputs but does not act on them. If the block becomes enabled, it immediately begins acting on the latest set of inputs received.

Operational Inputs

Every two-input AND block requires two operational input values. Each input is a Boolean 1 or 0. These inputs may come from some combination of:

- constant values
- digital inputs from modules on the island
- digital outputs from the virtual module (See *The Virtual Module, p. 28*)
- an output on the action module (See *Using the Action Module as an Input to a Block, p. 32*) written to by the fieldbus master
- if the AND is the second block in a nested reflex action (See *Nesting Two Reflex Blocks, p. 36*), one of the operational inputs may be the output from the first reflex block

Physical Output

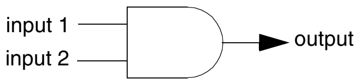
The output from a two-input AND block is a Boolean true (1) or false (0), as shown in the truth tables that follow. The physical output (See *Configuring the Physical Output from a Reflex Block, p. 27*) needs to be mapped to an action module:

- The action module may be a digital output module on the island bus. In this case, you need to specify one of the digital output channels as the destination for the reflex output.
- If the AND is the first block in a nested reflex action, the action module needs to be the same as the one specified for the second reflex block. Specify the channel as *None*.

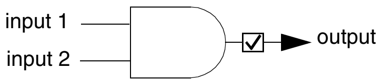
When the output of a reflex block is mapped to a channel on a digital output module, that channel becomes dedicated to the reflex action and can no longer use data from the fieldbus master to update its field device. The fieldbus master still has the ability to write data to this bit address in the NIM, and the reflex action editor lets you use this data from the fieldbus master as an input to the block.

Truth Tables

In its simplest form, a two-input AND block looks like this:



and an inverted AND (a NAND) block looks like this:



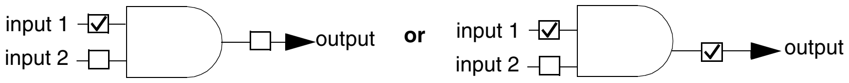
The following truth table shows the possible outputs of this AND operation:

If input 1 is:	and input 2 is:	then the standard output is:	and the inverted output is:
0	0	0	1
0	1	0	1
1	0	0	1
1	1	1	0

Inverted Operational Inputs

One or both of the operational inputs may be inverted. An inversion is indicated in the Advantys configuration software as a check mark in a box on an input line.

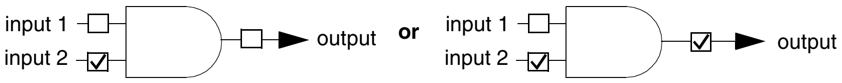
When input 1 is inverted:



the truth table yields the following:

If input 1 is:	and input 2 is:	then the standard output is:	and the inverted output is:
0	0	0	1
0	1	1	0
1	0	0	1
1	1	0	1

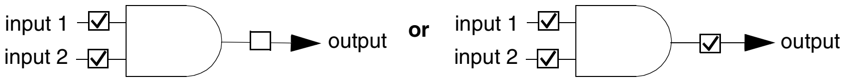
When input 2 is inverted:



the truth table yields the following:

If input 1 is:	and input 2 is:	then the standard output is:	and the inverted output is:
0	0	0	1
0	1	0	1
1	0	1	0
1	1	0	1

When both inputs are inverted:



the truth table yields the following:

If input 1 is:	and input 2 is:	then the standard output is:	and the inverted output is:
0	0	1	0
0	1	0	1
1	0	0	1
1	1	0	1

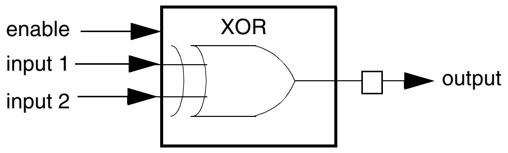
XOR Blocks

Summary

An XOR block performs an exclusive-OR operation on two Boolean operational inputs. The output from the block is Boolean true or false, expressed as a value of 1 or 0, respectively. You may invert the value of the output, in which case the action becomes an exclusive-NOR (XNOR).

Structure of an XOR Block

An XOR block diagram is shown below:



The block has three inputs—one enable input and two operational inputs. The enable input turns the XOR block on or off. The operational inputs send two Boolean values to the block. The inputs are XORed with each other when the block is enabled, and the result is a Boolean output.
The checkbox on the output line provides the mechanism by which the output value may be inverted. When you click on this box, a check mark toggles on or off. When the box is checked, the value of the associated output is inverted—i.e., a 1 becomes a 0, a 0 becomes a 1.

Enable Input

An XOR block can be enabled either by a Boolean 1 or an *always enabled* constant. It can be disabled by a Boolean 0 or an *always disabled* constant.
If the enable input is a Boolean, it may be produced by:

- a digital input from a module on the island
- a digital output from the virtual module (See *The Virtual Module, p. 28*)
- an output on the action module (See *Using the Action Module as an Input to a Block, p. 32*) written to by the fieldbus master

When the enable input is a Boolean 0 or an *always disabled* constant, the block is disabled—the action does not execute and the output is frozen in the state it was in when the block became disabled. The block continues to process inputs but does not act on them. If the block becomes enabled, it immediately begins acting on the latest set of inputs received.

Operational Inputs

Every XOR block requires two operational input values. These inputs may come from some combination of:

- constant values
- digital inputs from modules on the island
- digital outputs from the virtual module (See *The Virtual Module, p. 28*)
- an output on the action module (See *Using the Action Module as an Input to a Block, p. 32*) written to by the fieldbus master
- if the XOR is the second block in a nested reflex action (See *Nesting Two Reflex Blocks, p. 36*), an operational input may be the output from the first reflex block

Physical Output

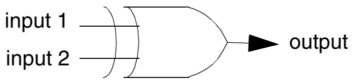
The output from an XOR block is a Boolean true (1) or false (0), as shown in the truth tables that follow. The physical output (See *Configuring the Physical Output from a Reflex Block, p. 27*) needs to be mapped to an action module:

- The action module may be a digital output module on the island bus. In this case, you need to specify one of the digital output channels as the destination for the reflex output.
- If the XOR is the first block in a nested reflex action, the action module needs to be the same as the one specified for the second reflex block. Specify the channel as *None*.

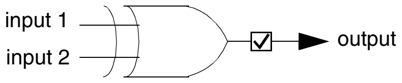
When the output from a block is mapped to a channel on a digital output module, that channel becomes dedicated to the reflex action and can no longer use data from the fieldbus master to update its field device. The fieldbus master still has the ability to write data to this bit address in the NIM, and the reflex action editor lets you use this data from the fieldbus master as an input to the block.

Truth Table

In its simplest form, a standard XOR block looks like this:



and an inverted XOR (an XNOR) block looks like this:



The following truth table shows the possible outputs:

If input 1 is:	and input 2 is:	then the standard output is:	and the inverted output is:
0	0	0	1
0	1	1	0
1	0	1	0
1	1	0	1

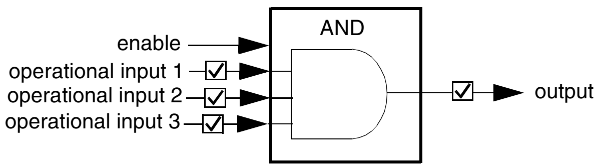
Three-input AND Blocks

Summary

A three-input AND block performs a logical AND operation on three Boolean operational inputs. The output is Boolean true or false, expressed as a value of 1 or 0, respectively. Optionally, you may invert one or more inputs. You may also invert the value of the output, in which case the action becomes a logical NAND.

Structure of a Three-input AND Block

A block diagram for a three-input AND is shown below:



The block has four inputs—one enable input and three operational inputs. The enable input turns the block on or off. The operational inputs send three Boolean values to the block. The inputs are ANDed together when the block is enabled, and the result is a Boolean output.
The checkboxes on the three input lines and the output line provide the mechanism by which one or more of the input/output values can be inverted. When you click on one of these boxes, a check mark toggles on or off. When a box is checked, the value of the associated input or output is inverted—i.e., a 1 becomes a 0, a 0 becomes a 1.

Enable Input

An AND block can be enabled either by a Boolean 1 or an *always enabled* constant. It can be disabled by a Boolean 0 or an *always disabled* constant.
If the enable input is a Boolean, it may be produced by:
- a digital input from a module on the island
- a digital output from the virtual module (See *The Virtual Module, p. 28*)
- an output on the action module (See *Using the Action Module as an Input to a Block, p. 32*) written to by the fieldbus master

When the enable input is a Boolean 0 or an *always disabled* constant, the block is disabled—the action does not execute and the output is frozen in the state it was in when the block became disabled. The block continues to process inputs but does not act on them. If the block becomes enabled, it immediately begins acting on the latest set of inputs received.

Operational Inputs

Every three-input AND requires three operational input values. Each input is a Boolean 1 or 0. These inputs may come from some combination of

- constant values
- digital inputs from modules on the island
- digital outputs from the virtual module (See *The Virtual Module, p. 28*)
- an output on the action module (See *Using the Action Module as an Input to a Block, p. 32*) written to by the fieldbus master
- if the AND is the second part of a nested reflex action (See *Nesting Two Reflex Blocks, p. 36*), one of the operational inputs may be the output from the first reflex block

Physical Output

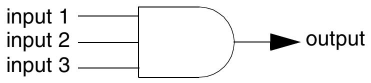
The output from a three-input AND action is a Boolean true (1) or false (0), as shown in the truth tables that follow. The physical output (See *Configuring the Physical Output from a Reflex Block, p. 27*) needs to be mapped to an action module:

- The action module may be a digital output module on the island bus. In this case, you need to specify one of the digital output channels as the destination for the reflex output.
- If the AND is the first action in a nested reflex action, the action module needs to be the same action module as the one specified for the second reflex block. Specify the channel as *None*.

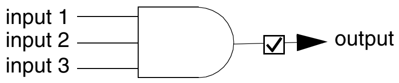
When the output from a block is mapped to a channel on a digital output module, that channel becomes dedicated to the reflex action and can no longer use data from the fieldbus master to update its field device. The fieldbus master still has the ability to write data to this bit address in the NIM, and the reflex action editor lets you use this data from the fieldbus master as an input to the block.

Truth Tables

In its simplest form, a three-input AND block looks like this:



and an inverted AND (a NAND) block looks like this:



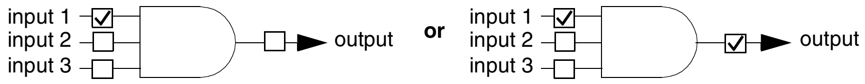
The following truth table shows the possible putouts of this AND operation:

If input 1 is:	**and input 2 is:**	**and input 3 is:**	**then the standard output is:**	**and the inverted output is:**
0	0	0	0	1
0	0	1	0	1
0	1	0	0	1
0	1	1	0	1
1	0	0	0	1
1	0	1	0	1
1	1	0	0	1
1	1	1	1	0

Inverted Operational Inputs

One or more of the operational inputs may be inverted. An inversion is indicated in the Advantys configuration software as a check mark in a box on an input line.

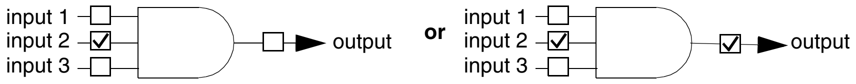
When input 1 is inverted:



The truth table yields the following:

If input 1 is:	and input 2 is:	and input 3 is:	then the standard output is:	and the inverted output is:
0	0	0	0	1
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	0	1
1	1	0	0	1
1	1	1	0	1

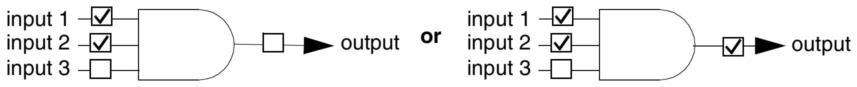
When input 2 is inverted:



the truth table yields the following:

If input 1 is:	and input 2 is:	and input 3 is:	then the standard output is:	and the inverted output is:
0	0	0	0	1
0	0	1	0	1
0	1	0	0	1
0	1	1	0	1
1	0	0	0	1
1	0	1	1	0
1	1	0	0	1
1	1	1	0	1

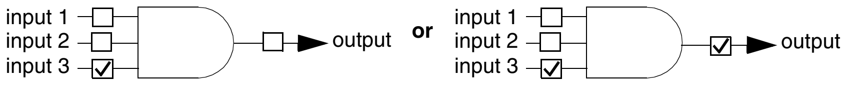
When inputs 1 and 2 are both inverted:



the truth table yields the following:

If input 1 is:	and input 2 is:	and input 3 is:	then the standard output is:	and the inverted output is:
0	0	0	0	1
0	0	1	1	0
0	1	0	0	1
0	1	1	0	1
1	0	0	0	1
1	0	1	0	1
1	1	0	0	1
1	1	1	0	1

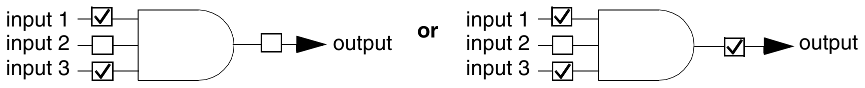
When input 3 is inverted:



the truth table yields the following:

If input 1 is:	and input 2 is:	and input 3 is:	then the standard output is:	and the inverted output is:
0	0	0	0	1
0	0	1	0	1
0	1	0	0	1
0	1	1	0	1
1	0	0	0	1
1	0	1	0	1
1	1	0	1	0
1	1	1	0	1

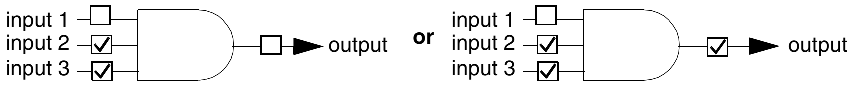
When inputs 1 and 3 are both inverted:



the truth table yields the following:

If input 1 is:	and input 2 is:	and input 3 is:	then the standard output is:	and the inverted output is:
0	0	0	0	1
0	0	1	0	1
0	1	0	1	0
0	1	1	0	1
1	0	0	0	1
1	0	1	0	1
1	1	0	0	1
1	1	1	0	1

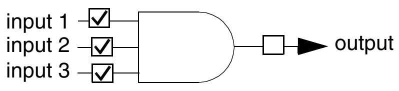
When inputs 2 and 3 are both inverted:



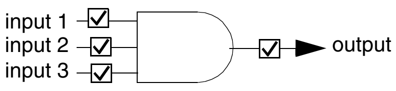
the truth table yields the following:

If input 1 is:	and input 2 is:	and input 3 is:	then the standard output is:	and the inverted output is:
0	0	0	0	1
0	0	1	0	1
0	1	0	0	1
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	0	1

When all three inputs are inverted:



or



the truth table yields the following:

If input 1 is:	and input 2 is:	and input 3 is:	then the standard output is:	and the inverted output is:
0	0	0	1	0
0	0	1	0	1
0	1	0	0	1
0	1	1	0	1
1	0	0	0	1
1	0	1	0	1
1	1	0	0	1
1	1	1	0	1

Integer Compare Reflex Blocks

3

At a Glance

Overview

This chapter describes four integer compare reflex blocks. Two of these blocks compare an analog input value to a single threshold value and produce a specific Boolean result when the input is greater than or less than the threshold. The other two blocks compare an analog input value against a window defined by two threshold values and produce a specific Boolean result when the input value is either inside or outside that window.

What's in this Chapter?

This chapter contains the following topics:

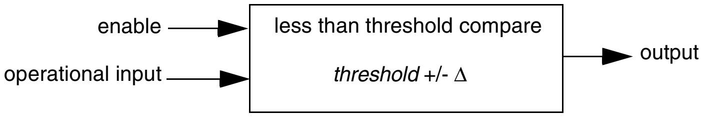
Less-than-threshold Integer Compare Block

Summary

A less-than-threshold integer compare performs a comparison between an analog input value and a threshold value that you specify. The analog input value is represented as an integer in the range -32 768 to +32 767. The software allows you to assign a delta (Δ), which acts as an hysteresis around the threshold value. The block produces a Boolean result as its output.

Structure of a Less-than-threshold Compare Block

A block diagram for a less-than-threshold integer compare is shown below:



The block has two inputs—an enable input and one operational input. The enable input turns the block on or off. The operational input sends a word value to the block that will be compared against the threshold.
The block also has two preset values (See *Configuring Preset Values for a Reflex Block, p. 27*)—a threshold value against which the operational input value will be compared and a Δ for hysteresis around the threshold. You must specify these presets.
The output is a Boolean 1 when the operational input value is less than threshold - Δ and a Boolean 0 when the input is greater than or equal to threshold + Δ. The output remains unchanged when the operational input is greater than or equal to threshold - Δ and less than threshold + Δ.

Enable Input

A less-than-threshold compare block is enabled either by a Boolean 1 or an *always enabled* constant. It can be disabled by a Boolean 0 or an *always disabled* constant. If the enable input is a Boolean, it may be produced by:
- a digital input from a module on the island
- a digital output from the virtual module (See *The Virtual Module, p. 28*)
- an output on the action module (See *Using the Action Module as an Input to a Block, p. 32*) written to by the fieldbus master

When the enable input is a Boolean 0 or an *always disabled* constant, the block is disabled—the action does not execute and the output is frozen in the state it was in when the block became disabled. The block continues to process inputs but does not act on them. If the block becomes enabled, it immediately begins acting on the latest set of inputs received.

Operational Input

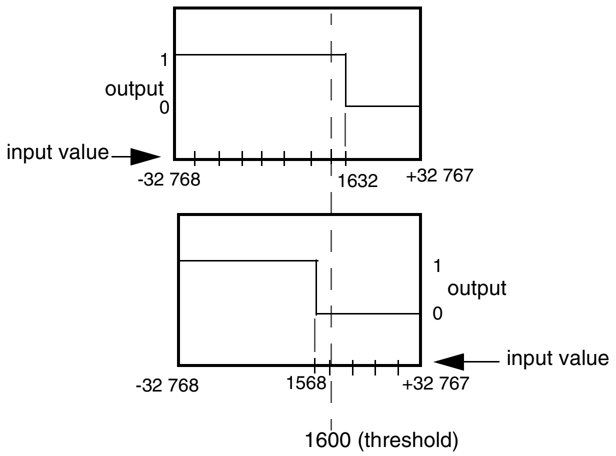
A less-than-threshold integer compare block uses one operational input. This input needs to be a word that holds a signed integer value in the range -32,768 to +32,767. The input can come from:
- an analog input from a module on the island
- an analog output from the virtual module (See *The Virtual Module, p. 28*)
- if the less-than-threshold compare is the second block in a nested reflex action (See *Nesting Two Reflex Blocks, p. 36*), the operational input may be configured as the output of the first reflex block

Threshold and Δ

You need to enter two preset values—a threshold and a Δ. The threshold is the value against which the operational input is compared. You can add a Δ value to the threshold, which acts as an hysteresis.

Note: To be valid, threshold + Δ and threshold - Δ must be integers in the range -32,768 to +32,767.

For example, say you assign a threshold value of 1600 to the compare block. You then assign a Δ value of 32 to that threshold. If your operational input value is less than threshold - Δ (1568), the output from the block is 1. If your operational input is greater than or equal to threshold + Δ (1632), the output is 0:



While the input value is within the 2Δ band, it holds its last value.
For example, if the input value is increasing from a value less than 1568, the output will be 1 until the input value reaches 1632. When it exceeds 1632, the output drops to 0. If the input value then begins to decrease after the output has dropped to 0, the output will remain at 0 until the input value decreases to 1568, at which point it will rise to 1.

Physical Output

The block produces as its output a Boolean 1 when the input value is less than threshold - Δ and a Boolean 0 when the input is greater than or equal to threshold + Δ. The physical output (See *Configuring the Physical Output from a Reflex Block, p. 27*) needs to be mapped to an action module:

- The action module may be a digital output module on the island bus. In this case, you need to specify one of the digital output channels as the destination for the reflex output.
- If the compare is the first block in a nested reflex action (See *Nesting Two Reflex Blocks, p. 36*), the action module needs to be the same action module as the one specified for the second block. Specify the channel as *None*.

When the output of a block is mapped to a channel on a digital output module, that channel becomes dedicated to the reflex action and can no longer use data from the fieldbus master to update its field device. The fieldbus master still has the ability to write data to this bit address in the NIM, and the reflex action editor lets you use this data from the fieldbus master as an input to the block.

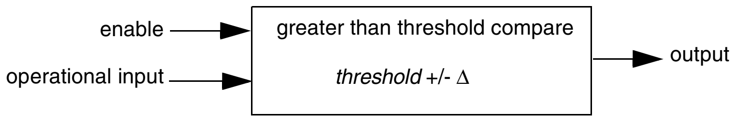
Greater-than-threshold Integer Compare Block

Summary

A greater-than-threshold integer compare block performs a comparison between an analog input value and a threshold value that you specify using the Advantys configuration software. The analog input value is represented as an integer in the range -32 768 to +32 767. The software allows you to assign a delta (Δ) value, which acts as an hysteresis around the threshold value. The action produces a Boolean result as its output.

Structure of a Greater-than-threshold Compare Block

A block diagram for a greater-than-threshold integer compare is shown below:



The block has two inputs—an enable input and one operational input. The enable input turns the block on or off. The operational input sends a word value to the block that will be compared against the threshold.
The block also has two preset values (See *Configuring Preset Values for a Reflex Block, p. 27*)—a threshold value against which the operational input value will be compared and a Δ for hysteresis around the threshold. You must specify these presets.
The output is a Boolean 1 when the operational input value is greater than threshold + Δ and a Boolean 0 when the input is less than or equal to threshold - Δ. The output remains unchanged when the operational input is greater than threshold - Δ and less than or equal to threshold + Δ.

Enable Input

A greater-than-threshold compare block can be enabled either by a Boolean 1 or an *always enabled* constant. It can be disabled by a Boolean 0 or an *always disabled* constant.
If the enable input is a Boolean, it may be produced by:
- a digital input or output from a module on the island
- a digital output from the virtual module (See *The Virtual Module, p. 28*)
- an output on the action module (See *Using the Action Module as an Input to a Block, p. 32*) written to by the fieldbus master

When the enable input is a Boolean 0 or an *always disabled* constant, the block is disabled—the action does not execute and the output is frozen in the state it was in when the block became disabled. The block continues to process inputs but does not act on them. If the block becomes enabled, it immediately begins acting on the latest set of inputs received.

Operational Input

A greater-than-threshold integer compare block uses one operational input. This input needs to be a word with a signed integer value in the range -32,768 to +32,767. The input can come from:

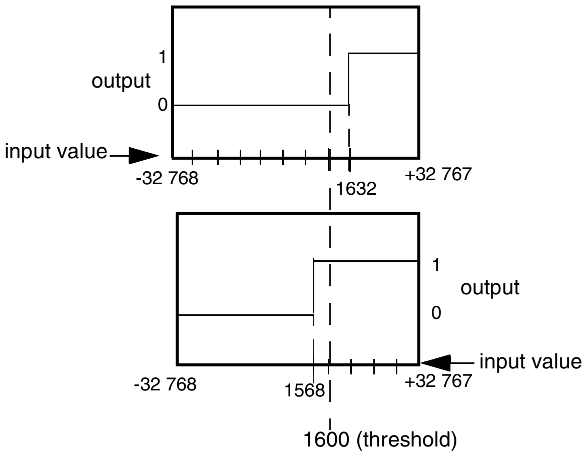
- an analog input from a module on the island
- an analog output from the virtual module
- if the compare is the second block in a nested reflex action (See *Nesting Two Reflex Blocks, p. 36*), the operational input may be configured as the output of the first reflex block

Threshold and Δ

You need to enter two values—threshold and the Δ. The threshold is the value against which the operational input is compared. You can also add a Δ value to the threshold, which acts as an hysteresis.

Note: To be valid, threshold + Δ and threshold - Δ must be a value in the range -32,768 to +32,767.

For example, say you assign a threshold value of 1600 to the block. You then assign a Δ value of 32 to that threshold. If your operational input value is less than or equal to threshold - Δ (1568), the output is 0. If your operational input is greater than threshold + Δ (1632), the output is 1:



While the input value is within the 2Δ band, it holds its last value.
For example, if the input value is increasing from a value less than or equal to 1568, the output will be 0. When it exceeds 1632, the output rises to 1. If the input value then begins to decrease after the output has risen to 1, the output will remain at 1 until the input value decreases to 1568, at which point it will drop to 0.

Physical Output

The block produces a Boolean 1 as its output when the input value is greater than threshold + Δ and a Boolean 0 as its output when the input value is less than or equal to threshold - Δ. The physical output (See *Configuring the Physical Output from a Reflex Block, p. 27*) needs to be mapped to an action module:

- The action module may be a digital output module on the island bus. In this case, you need to specify one of the digital output channels as the destination for the block's output.
- If the compare is the first block in a nested reflex action (See *Nesting Two Reflex Blocks, p. 36*), the action module needs to be the same action module as the one specified for the second reflex block. Specify the channel as *None*.

When the output from a block is mapped to a channel on a digital output module, that channel becomes dedicated to the reflex action and can no longer use data from the fieldbus master to update its field device. The fieldbus master still has the ability to write data to this bit address in the NIM, and the reflex action editor lets you use this data from the fieldbus master as an input to the block.

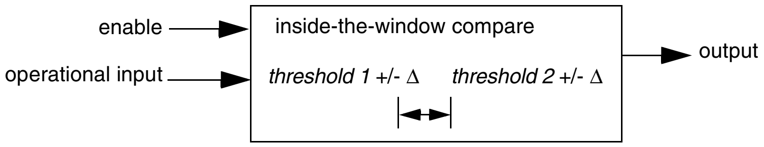
Inside-the-window Integer Compare Block

Summary

An inside-the-window integer compare block performs a comparison between an analog input value and a window bounded by two thresholds. The input value is represented as an integer in the range -32 768 to +32 767. The software lets you assign values to the two thresholds (TH 1 and TH 2) along with a delta (Δ) value, which acts as an hysteresis around TH 1 and TH 2. The block produces a Boolean result as its output.

Structure of an Inside-the-window Compare Block

A block diagram for an inside-the-window integer compare is shown below:



The block has two inputs—an enable input and one operational input. The enable input turns the block on or off. The operational input sends a word value to the block that will be compared against the thresholds.
The block also has three preset values (See *Configuring Preset Values for a Reflex Block, p. 27*)—TH 1, TH 2 and a Δ for hysteresis around the TH 1 and TH 2 values. The range of values between TH 1 - Δ and TH 2 + Δ comprises the window against which the operational input value will be compared. You must specify these presets.
The output is a Boolean 1 when the operational input value is inside the window (greater than TH1 + Δ but less than TH2 - Δ) and a Boolean 0 when the input value is not inside the window (less than or equal to TH1 - Δ or greater than or equal to TH2 + Δ). The output remains unchanged when the operational input is greater than TH1 - Δ but less than or equal to TH1 + Δ, or when it is greater than or equal to TH2 - Δ but less than TH2 + Δ.

Enable Input

An inside-the-window integer compare block can be enabled either by a Boolean 1 or an *always enabled* constant. It can be disabled by a Boolean 0 or an *always disabled* constant.
If the enable input is a Boolean, it may be produced by:
- a digital input from a module on the island
- a digital output from the virtual module (See *The Virtual Module, p. 28*)
- an output on the action module (See *Using the Action Module as an Input to a Block, p. 32*) written to by the fieldbus master

When the enable input is a Boolean 0 or an *always disabled* constant, the block is disabled—the action does not execute and the output is frozen in the state it was in when the block became disabled. The block continues to process inputs but does not act on them. If the block becomes enabled, it immediately begins acting on the latest set of inputs received.

Thresholds

Inside-the-window compares require two threshold values, which define the upper and lower bounds of the window. Each TH value needs to be a signed integer value in the range -32 768 to +32 767. TH 1 defines the lower boundary of the window; TH 2 defines the upper boundary.

Note: The value of TH 2 must be greater than the value of TH 1.

Operational Input

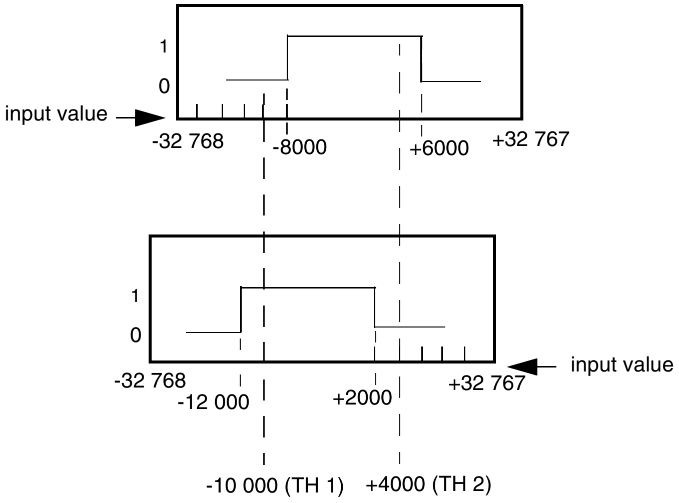
An inside-the-window compare uses one operational input. It must be a word with an integer value in the range -32 768 to +32 767. The input can come from:
- an analog input from a module on the island
- an analog output from the virtual module (See *The Virtual Module, p. 28*)
- if the inside-the-window compare is the second block in a nested reflex action (See *Nesting Two Reflex Blocks, p. 36*), the operational input may be configured as the output of the first reflex block

Delta (Δ)

You can also add a Δ value to an inside-the-window compare. The Δ acts as an hysteresis around the two thresholds.

Note: To be valid, TH 2 - TH 1 must be greater than 2Δ. For example, say that TH 1 = -10 000 and TH 2 = +4000. The Δ value you assign to the reflex action must therefore be less than 7000.

Suppose you have a window defined by TH 1 = -10 000 and TH 2 = +4000. To that window, you specify a Δ of 2000. If your operational input value is less than or equal to TH1 - Δ (less than or equal to -12000)and is increasing in value, the reflex result is 0. The result remains 0 until the input value exceeds -8000, at which point the result rises to 1. If the input value continues to increase, the result remains at 1 until the input value reaches TH 2 + Δ (+6000). When the input value reaches +6000, the result drops back to 0.
If, on the other hand, the input value is decreasing from a value greater than or equal to TH 2 + Δ (greater than or equal to +6000), the reflex result is 0 until the point where the input value becomes lower than +2000. At this point, the result rises to 1 and remains at 1 as the input value decreases to TH 1 - Δ (-12 000). At this point, the result drops back to 0:



While the input value is inside the area defined by the threshold and the Δ, it holds its last value.
For example, if the input value is inside the window and increasing, the result will be 1. When it hits +6000, the result drops to 0. If the input value then begins to decrease after the result has dropped, the result will remain there until the input value decreases below +2000, at which point it will rise to 1 and will remain there until the input value decreases to -12 000.

Physical Output

The block produces a Boolean 1 when the input value is inside the window and a Boolean 0 when the input value is not inside that window. The physical output (See *Configuring the Physical Output from a Reflex Block, p. 27*) needs to be mapped to an action module:

- The action module may be a digital output module on the island bus. In this case, you need to specify one of the digital output channels as the destination for the block's output.
- If the compare is the first block in a nested reflex action (See *Nesting Two Reflex Blocks, p. 36*), the action module needs to be the same action module as the one specified for the second reflex block. Specify the channel as *None*.

When the output from a block is mapped to a channel on a digital output module, that channel becomes dedicated to the reflex action and can no longer use data from the fieldbus master to update its field device. The fieldbus master still has the ability to write data to this bit address in the NIM, and the reflex action editor lets you use this data from the fieldbus master as an input to the block.

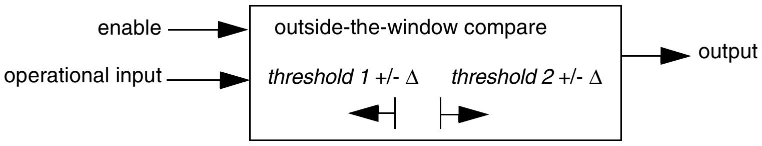
Outside-the-window Integer Compare Block

Summary

An outside-the-window integer compare block performs a comparison between an analog input value and a window of values bounded by two thresholds. The input value is represented as an integer in the range -32 768 to +32 767. The software lets you assign values to the two thresholds (TH 1 and TH 2) along with a delta (Δ) value, which acts as an hysteresis around TH 1 and TH 2. The block produces a Boolean result as its output.

Structure of an Outside-the-window Compare Block

A block diagram for an outside-the-window integer compare is shown below:



The block has two inputs—an enable input and one operational input. The enable input turns the block on or off. The operational input sends a word value to the block that will be compared against the thresholds.
The block also has three preset values (See *Configuring Preset Values for a Reflex Block, p. 27*), which you must specify—TH 1, TH 2 and a Δ for hysteresis around the TH 1 and TH 2 values. The range of values between TH 1 - Δ and TH 2 + Δ comprises the window against which the operational input value will be compared.
The output is a Boolean 1 when the operational input value is outside the window (less than TH 1 - Δ or greater than TH 2 + Δ) and a Boolean 0 when the input value is not outside the window (greater than or equal to TH 1 + Δ but less than or equal to TH 2 - Δ). The output remains unchanged when the operational input is greater than or equal to TH 1 - Δ but less than TH 1 + Δ, or when it is greater than TH 2 - Δ but less than or equal to TH 2 + Δ.

Enable Input

An outside-the-window integer compare block can be enabled either by a Boolean 1 or an *always enabled* constant. It can be disabled by a Boolean 0 or an *always disabled* constant.
If the enable input is a Boolean, it may be produced by:
- a digital input from a module on the island
- a digital output from the virtual module (See *The Virtual Module, p. 28*)
- an output on the action module (See *Using the Action Module as an Input to a Block, p. 32*) written to by the fieldbus master

When the enable input is a Boolean 0 or an *always disabled* constant, the block is disabled—the action does not execute and the output is frozen in the state it was in when the block became disabled. The block continues to process inputs but does not act on them. If the block becomes enabled, it immediately begins acting on the latest set of inputs received.

Thresholds

Outside-the-window compares require two threshold values, which define the upper and lower bounds of the window. Each TH value needs to be a signed integer in the range -32 768 to +32 767. TH 1 defines the lower boundary of the window; TH 2 defines the upper boundary.

Note: The value of TH 2 must be greater than the value of TH 1.

Operational Input

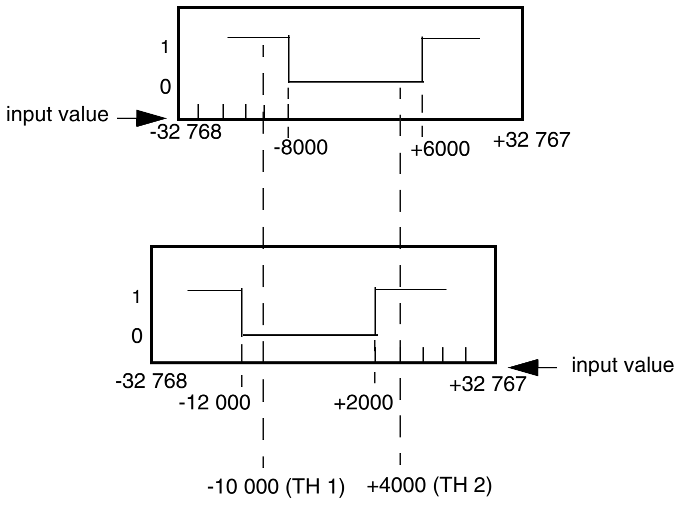
An outside-the-window compare block uses one operational input. It must be a word with an integer value in the range -32 768 to +32 767. The input can come from:
- an analog input from a module on the island
- an analog output from the virtual module
- if the outside-the-window compare is the second block in a nested reflex action (See *Nesting Two Reflex Blocks, p. 36*), the operational input may be configured as the output of the first reflex block

Delta (Δ)

You can also add a Δ value to an outside-the-window compare, which acts as an hysteresis around the two thresholds.

Note: To be valid, TH 2 - TH 1 must be greater than 2Δ. For example, say that TH 1 = -10 000 and TH 2 = +4000. The Δ value you assign to the reflex action must therefore be less than 7000.

Suppose you have a window defined by TH 1 = -10 000 and TH 2 = +4000. To that window, you specify a Δ of 2000. If your operational input value is less than or equal to TH 1 - Δ (less than or equal to -12000)and is increasing in value, the reflex output is 1. The output remains 1 until the input value reaches -8000, at which point the output drops to 0. If the input value continues to increase, the output remains at 0 until the input value exceeds TH 2 + Δ (+6000). When the input value exceeds +6000, the output rises back to 1.
If, on the other hand, the input value is decreasing from a value greater than TH 2 + Δ (greater than +6000), the reflex output is 1 until the point where the input value reaches +2000. At this point, the output drops to 0 and remains at 0 as the input value decreases beyond TH 1 - Δ (less than -12 000). At this point, the output rises back to 1:



While the input value is inside the area defined by the threshold and the Δ, it holds its last value.
For example, if the input value is inside the window and increasing, the output will be 0. When it exceeds +6000, the output rises to 1. If the input value then begins to decrease after the output has dropped, the output will remain there until the input value decreases to +2000, at which point it will drop to 0 and will remain there until the input value decreases beyond -12 000.

Physical Output

The block produces a Boolean 1 when the input value is outside the window and a Boolean 0 when the input value is not outside that window. The physical output (See *Configuring the Physical Output from a Reflex Block, p. 27*) needs to be mapped to an action module:

- The action module may be a digital output module on the island bus. In this case, you need to specify one of the digital output channels as the destination for the reflex output.
- If the compare is the first block in a nested reflex action (See *Nesting Two Reflex Blocks, p. 36*), the action module needs to be the same action module as the one specified for the second reflex block. Specify the channel as *None*.

When the output from a block is mapped to a channel on a digital output module, that channel becomes dedicated to the reflex action and can no longer use data from the fieldbus master to update its field device. The fieldbus master still has the ability to write data to this bit address in the NIM, and the reflex action editor lets you use this data from the fieldbus master as an input to the block.

Unsigned Compare Reflex Blocks

4

At a Glance

Overview

This chapter describes four unsigned compare reflex blocks. Two of these blocks compare an analog input value to a single threshold value and produce a specific Boolean result when the input is greater that or less than the threshold. The other two blocks compare an analog input value against a window defined by two threshold values and produce a specific Boolean result when the input value is either inside or outside that window.

What's in this Chapter?

This chapter contains the following topics:

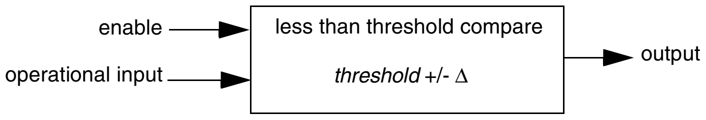
Less-than-threshold Unsigned Compare Block

Summary

A less-than-threshold unsigned compare block performs a comparison between an analog input value and a threshold value. The input value is represented as an integer in the range 0 to 65 535. The software lets you assign the threshold value along with a delta (Δ) value, which acts as an hysteresis for the threshold. The action produces a Boolean result as its output.

Structure of a Less-than-threshold Comparison Block

A block diagram for a less-than-threshold unsigned compare is shown below:



The block has two inputs—an enable input and one operational input. The enable input turns the block on or off. The operational input sends a word value to the block that will be compared against the threshold.
The block also has two preset values (See *Configuring Preset Values for a Reflex Block, p. 27*), which you must specify—a threshold value against which the operational input value will be compared and a Δ for hysteresis around the threshold value.
The output is a Boolean 1 when the operational input is less than threshold - Δ and a Boolean 0 when the input is greater than or equal to threshold + Δ. The output remains unchanged when the operational input is greater than or equal to TH - Δ but less than TH + Δ.

Enable Input

A less-than-threshold unsigned compare block can be enabled either by a Boolean 1 or an *always enabled* constant. It can be disabled by a Boolean 0 or an *always disabled* constant.
If the enable input is a Boolean, it may be produced by:
- a digital input from a module on the island
- a digital output from the virtual module (See *The Virtual Module, p. 28*)
- an output on the action module (See *Using the Action Module as an Input to a Block, p. 32*) written to by the fieldbus master

When the enable input is a Boolean 0 or an *always disabled* constant, the block is disabled—the action does not execute and the output is frozen in the state it was in when the block became disabled. The block continues to process inputs but does not act on them. If the block becomes enabled, it immediately begins acting on the latest set of inputs received.

Operational Input

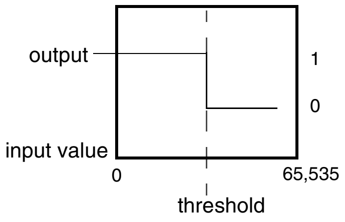
A less-than-threshold unsigned compare block uses one operational input. It must be a word with an unsigned integer value in the range 0 to 65 535. The input can come from:

- an analog input from a module on the island
- an analog output from the virtual module (See *The Virtual Module, p. 28*)
- if the less-than-threshold compare is the second block in a nested reflex action (See *Nesting Two Reflex Blocks, p. 36*), the operational input may be configured as the output of the first reflex block

Note: Unsigned compare blocks are often nested together with counter blocks (See *Counter Reflex Blocks, p. 93*). The unsigned compare is always the second block in the nested action, and the analog output from the counter is used as its operational input. These two action types complement each other well because the output from a counter is always unsigned with 16-bit resolution.

Note: Do not use a word that contains a signed negative integer value as the operational input to this comparison. The reflex action will misinterpret a value of 1 in the sign bit position (bit 15) as part of the integer value. Avoid the use of modules such as the STB AVI 1270 analog input module, which produces an input with a possible negative integer value, as the source for the operational input to your reflex action.

The illustration below shows a simple case of how the block works:



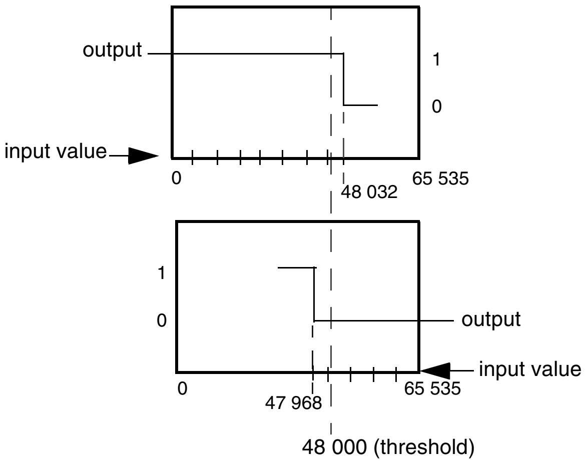
If the operational input is less than the threshold value, the output is 1. If the operational input is greater than or equal to the threshold value, the output is 0.

Threshold and Δ

You need to enter two values in a compare action, the threshold and the Δ. The threshold is the value against which the operational input is compared, as shown in the examples above. The Δ value acts as an hysteresis around the threshold.

Note: To be valid, threshold + Δ and threshold - Δ must be integers in the range 0 to 65 535.

For example, say you assign a threshold value of 48,000 to the comparison action. You then assign a Δ value of 32 to that threshold. Thus, if your operational input value is less than the threshold - Δ (47 968), the reflex output is 1. If your operational input is greater than or equal to threshold + Δ (48 032), the reflex output is 0:



While the input value is within the Δ band, it holds its last value.
For example, if the input value is increasing from a value less than 47 968, the output will be 1. When it hits 48 032, the output drops to 0. If the input value then begins to decrease after the output has dropped to 0, the output will remain at 0 until the input value decreases below 47 968, at which point it will rise to 1.

Physical Output

The block produces a Boolean 1 when the input is less than threshold - Δ and a Boolean 0 when the input is greater than or equal to threshold + Δ. The physical output (See *Configuring the Physical Output from a Reflex Block, p. 27*) needs to be mapped to an action module:

- The action module may be a digital output module on the island bus. In this case, you need to specify one of the digital output channels as the destination for the block's output.
- If the compare is the first block in a nested reflex action (See *Nesting Two Reflex Blocks, p. 36*), the action module needs to be the same action module as the one specified for the second reflex block. Specify the channel as *None*.

When the output from a block is mapped to a channel on a digital output module, that channel becomes dedicated to the reflex action and can no longer use data from the fieldbus master to update its field device. The fieldbus master still has the ability to write data to this bit address in the NIM, and the reflex action editor lets you use this data from the fieldbus master as an input to the block.

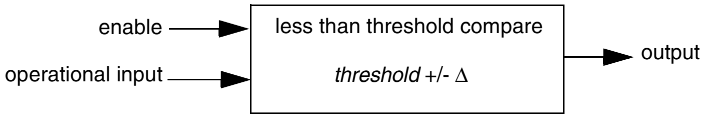
Greater-than-threshold Unsigned Compare Block

Summary

A greater-than-threshold unsigned compare block performs a comparison between an analog input value and a threshold value (TH). The input value is represented as an integer in the range 0 to 65 535. The software lets you configure the TH value along with a delta (Δ) value, which acts as an hysteresis for the threshold. The action produces a Boolean result as its output.

Structure of a Greater-than-threshold Compare Block

A block diagram for a greater-than-threshold unsigned compare is shown below:



The block has two inputs—an enable input and one operational input. The enable input turns the block on or off. The operational input sends a word value to the block that will be compared against the threshold.
The block also has two preset values (See *Configuring Preset Values for a Reflex Block, p. 27*)—a threshold value against which the operational input value will be compared and a Δ for hysteresis around the threshold value. You must specify these presets.
The output is a Boolean 1 when the operational input is greater than TH + Δ and a Boolean 0 when the input is less than or equal to the TH - Δ. The output remains unchanged when the operational input is greater than TH - Δ but less than or equal to TH + Δ.

Enable Input

A greater-than-threshold unsigned compare block can be enabled either by a Boolean 1 or an *always enabled* constant. It can be disabled by a Boolean 0 or an *always disabled* constant.
If the enable input is a Boolean, it may be produced by:
- a digital input from a module on the island
- a digital output from the virtual module (See *The Virtual Module, p. 28*)
- an output on the action module (See *Using the Action Module as an Input to a Block, p. 32*) written to by the fieldbus master

When the enable input is a Boolean 0 or an *always disabled* constant, the block is disabled—the action does not execute and the output is frozen in the state it was in when the block became disabled. The block continues to process inputs but does not act on them. If the block becomes enabled, it immediately begins acting on the latest set of inputs received.

Operational Input

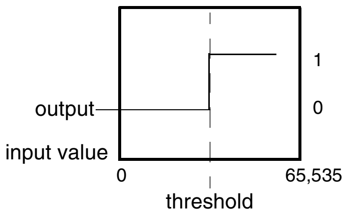
A greater-than-threshold unsigned compare block uses one operational input. It must be a word with an unsigned integer value in the range 0 to 65 535. The input can come from:

- an analog input channel on the island
- an analog output from the virtual module (See *The Virtual Module, p. 28*)
- if the greater-than-threshold compare is the second block in a nested reflex action (See *Nesting Two Reflex Blocks, p. 36*), the operational input may be configured as the output of the first reflex block

Note: Unsigned compares are often nested together with counter blocks (See *Counter Reflex Blocks, p. 93*). The unsigned compare is always the second block in the nested action, and the analog output from the counter is used as its operational input. These two action types complement each other well because the output from a counter is always unsigned with 16-bit resolution.

Note: Do not use a word that contains a signed negative integer value as the operational input to an unsigned integer comparison. The block will misinterpret a value of 1 in the sign bit position (bit 15) as part of the integer value. Avoid the use of modules such as the STB AVI 1270 analog input module, which produces an input with a possible negative integer value, as the source for the operational input to the block.

The illustration below shows the behavior of the block when Δ is 0:



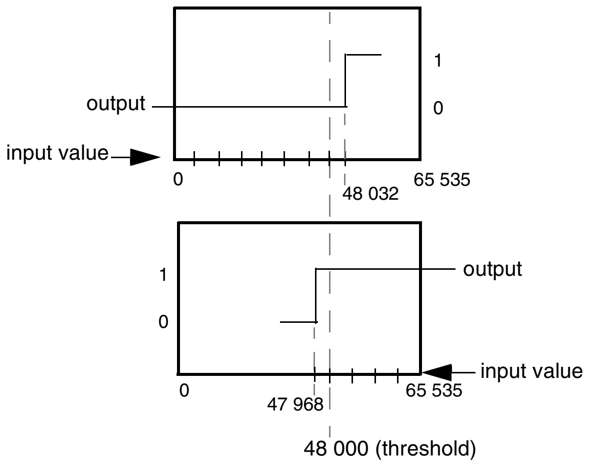
If the operational input is less than or equal to the threshold value, the output is 0. If the operational input value is greater than the threshold value, the output is 1

Threshold and Δ

You need to enter two values—threshold and the Δ. The threshold is the value against which the operational input is compared. You can also add a Δ value to the threshold, which acts as an hysteresis.

Note: To be valid, TH + Δ and TH - Δ must be integers in the range 0 to 65 535.

For example, say you assign a threshold value of 48,000 to the comparison action. You then assign a Δ value of 32 to that threshold. Thus, if your operational input value is less than or equal to TH - Δ (47 968), the output is 0. If your operational input is greater than TH + Δ (48 032), the output is 1:



While the input value is within the 2Δ band, it holds its last value.
For example, if the input value is increasing from a value less than or equal to 47 068, the output will be 0. When it exceeds 48 032, the output rises to 1. If the input value then begins to decrease after the output has risen to 1, the output will remain at 1 until the input value decreases to 47 968, at which point it will drop to 0.

Physical Output

The block produces a Boolean 1 when the input is greater than TH + Δ and a Boolean 0 when the input is less than or equal to TH - Δ. The physical output (See *Configuring the Physical Output from a Reflex Block, p. 27*) needs to be mapped to an action module:

- The action module may be a digital output module on the island bus. In this case, you need to specify one of the digital output channels as the destination for the block's output.
- If the compare is the first block in a nested reflex action (See *Nesting Two Reflex Blocks, p. 36*), the action module needs to be the same action module as the one specified for the second reflex block. Specify the channel as *None*.

When the output from a block is mapped to a channel on a digital output module, that channel becomes dedicated to the reflex action and can no longer use data from the fieldbus master to update its field device. The fieldbus master still has the ability to write data to this bit address in the NIM, and the reflex action editor lets you use this data from the fieldbus master as an input to the block.

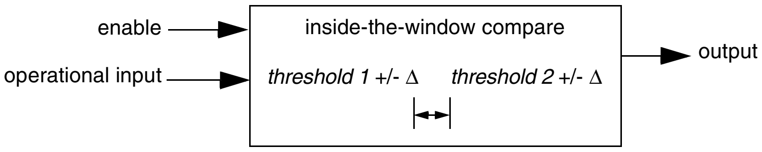
Inside-the-window Unsigned Compare Block

Summary

An inside-the-window unsigned compare block performs a comparison between an analog input value and a window of values bounded by two thresholds. The input value is represented as an integer in the range 0 to 65 535. The software lets you assign values to the two thresholds (TH 1 and TH 2) along with a delta (Δ) value, which acts as an hysteresis around TH 1 and TH 2. The block produces a Boolean result as its output.

Structure of an Inside-the-window Compare Block

A block diagram for an inside-the-window unsigned compare is shown below:



The block has two inputs—an enable input and one operational input. The enable input turns the block on or off. The operational input sends a word value to the block that will be compared against the thresholds.

The block also has three preset values (See *Configuring Preset Values for a Reflex Block, p. 27*)—TH 1, TH 2 and a Δ for hysteresis around the TH 1 and TH 2 values. The range of values between TH 1 - Δ and TH 2 + Δ comprises the window against which the operational input value will be compared. You must specify these presets.

The output is a Boolean 1 when the operational input value is inside the window ((greater than TH 1 + Δ but less than TH 2 - Δ) and a Boolean 0 when the input value is not inside the window (less than or equal to TH 1 - Δ or greater than or equal to TH 2 + Δ). The output remains unchanged when the operational input is greater than TH 1 - Δ but less than or equal to TH 1 + Δ, or when its is greater than or equal to TH 2 - Δ but less than TH 2 + Δ.

Enable Input

An inside-the-window unsigned compare block can be enabled either by a Boolean 1 or an *always enabled* constant. It can be disabled by a Boolean 0 or an *always disabled* constant.

If the enable input is a Boolean, it may be produced by:

- a digital input from a module on the island
- a digital output from the virtual module (See *The Virtual Module, p. 28*)
- an output on the action module (See *Using the Action Module as an Input to a Block, p. 32*) written to by the fieldbus master

When the enable input is a Boolean 0 or an *always disabled* constant, the block is disabled—the action does not execute and the output is frozen in the state it was in when the block became disabled. The block continues to process inputs but does not act on them. If the block becomes enabled, it immediately begins acting on the latest set of inputs received.

Thresholds

Inside-the-window compare blocks require two threshold values, which serve as the upper and lower bounds of the window. Each TH value needs to be an unsigned integer value in the range 0 to 65 535. TH 1 defines the lower boundary of the window; TH 2 defines the upper boundary.

Note: The value of TH 2 must be greater than the value of TH 1.

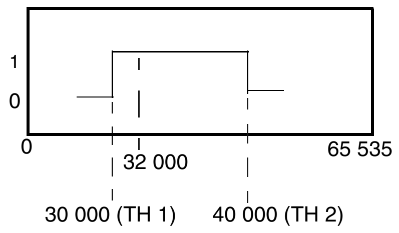
Operational Input

An inside-the-window compare block uses one operational input. It must be a word with an unsigned integer in the range 0 to 65 535. The input can come from:
- an analog input from a module on the island
- an analog output from the virtual module (See *The Virtual Module, p. 28*)
- if the less-than-threshold compare is the second block in a nested reflex action (See *Nesting Two Reflex Blocks, p. 36*), the operational input may be configured as the output of the first reflex block

Note: Unsigned compare blocks are often nested together with counter blocks (See *Counter Reflex Blocks, p. 93*). The unsigned compare is always the second block in the nested action, and the analog output from the counter is used as its operational input. These two action types complement each other well because the output from a counter is always unsigned with 16-bit resolution.

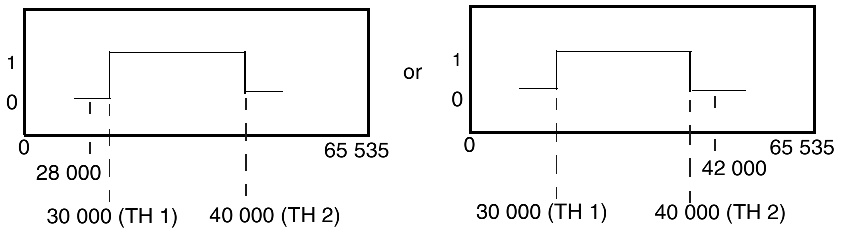
Note: Do not use a word that contains a signed negative integer value as the operational input to an unsigned integer compare. The block will misinterpret a value of 1 in the sign bit position (bit 15) as part of the integer value. Avoid the use of modules such as the STB AVI 1270 analog input module, which produce an input with a possible negative integer value, as the source for the operational input to the block.

Suppose that you have two threshold values, where TH 1 = 30 000 and TH 2 = 40 000. Then suppose that the operational input is 32 000 and Δ = 0:



Because the value of the operational input falls inside the window defined by TH 1 and TH 2, the block produces a Boolean 1 as its output.

Alternately, if the value of the operational input is less than TH 1 (say, 28 000) or greater than TH 2 (say 42 000):



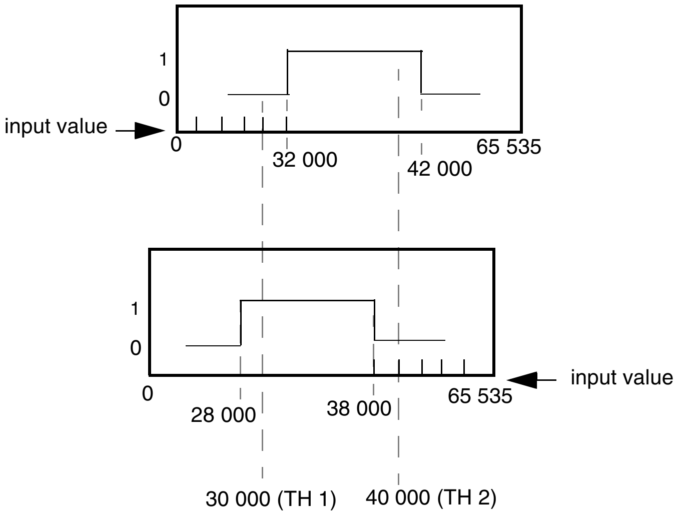
then the block produces a Boolean 0 as its output because the input value is outside the window.

Delta (Δ)

You can also add a Δ value to an inside-the-window compare block, which acts as an hysteresis around the two thresholds.

Note: To be valid, TH 2 - TH 1 must be greater than 2Δ. For example, say that TH 1 = 30 000 and TH 2 = 40 000. The Δ value you assign to the block must therefore be less than 5000.

Suppose you have a window defined by TH 1 = 30 000 and TH 2 = 40 000. To that window, you specify a Δ of 2000. If your operational input value is less than or equal to TH 1 - Δ (28 000) and is increasing in value, the output is 0. The output remains 0 until the input value exceeds 32 000, at which point the output rises to 1. If the input value continues to increase, the output remains at 1 until the input value reaches TH 2 + Δ (42 000). When the input value reaches 42 000, the output drops back to 0. If the input value is decreasing from a value greater than or equal to TH 2 + Δ (42 000), the output is 0 until the input value becomes less than 38 000. At this point, the output rises to 1 and remains there as the input value decreases to TH 1 - Δ (28 000). At this point, the output drops back to 0:



While the input value is within the window defined by the threshold and the Δ, it holds its last value.
For example, if the input value is inside the window and increasing, the output will be 1. When it hits 42 000, the output drops to 0. If the input value then begins to decrease after the output has dropped, the output will remain at 0 until the input value decreases to less than 38 000, at which point it will rise to 1 and will remain there until the input value decreases to 28 000.

Physical Output

The block produces a Boolean 1 when the input value is within the window and a Boolean 0 when the input value is outside that window. The physical output (See *Configuring the Physical Output from a Reflex Block, p. 27*) needs to be mapped to an action module:

- The action module may be a digital output module on the island bus. In this case, you need to specify one of the digital output channels as the destination for the block's output.
- If the compare is the first block in a nested reflex action (See *Nesting Two Reflex Blocks, p. 36*), the action module needs to be the same action module as the one specified for the second block. Specify the channel as *None*.

When the output from a block is mapped to a channel on a digital output module, that channel becomes dedicated to the reflex action and can no longer use data from the fieldbus master to update its field device. The fieldbus master still has the ability to write data to this bit address in the NIM, and the reflex action editor lets you use this data from the fieldbus master as an input to the block.

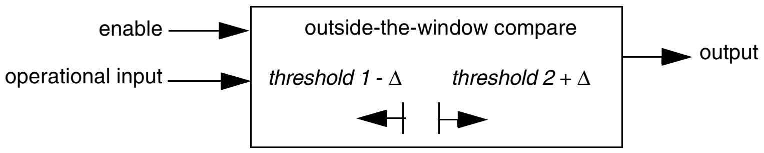
Outside-the-window Unsigned Compare Block

Summary

An outside-the-window unsigned compare block performs a comparison between an analog input value and a window of values bounded by two thresholds. The input value is represented as an integer in the range 0 to 65 535. The software lets you assign values to the two thresholds (TH 1 and TH 2) along with a delta (Δ) value, which acts as an hysteresis around TH 1 and TH 2. The block produces a Boolean result as its output.

Structure of an Outside-the-window Compare Block

A block diagram for an outside-the-window unsigned compare is shown below:



The block has two inputs—an enable input and one operational input. The enable input turns the block on or off. The operational input sends a word value to the block that will be compared against the thresholds.
The block also has three preset values (See *Configuring Preset Values for a Reflex Block, p. 27*)—TH 1, TH 2 and a Δ for hysteresis around the TH 1 and TH 2 values. The range of values between TH 1 - Δ and TH 2 + Δ comprises the window against which the operational input value will be compared. You must specify these presets.
The output is a Boolean 1 when the operational input value is outside the window (less than TH 1 - Δ or greater than TH 2 + Δ) and a Boolean 0 when the input value is not outside the window (greater than or equal to TH 1 + Δ but less than or equal to TH 2 - Δ). The output remains unchanged when the operational input is greater than or equal to TH 1 - Δ but less than TH 1 + Δ, or when it is greater than TH 2 - Δ but less than or equal to TH 2 + Δ.

Enable Input

An outside-the-window unsigned compare block can be enabled either by a Boolean 1 or an *always enabled* constant. It can be disabled by a Boolean 0 or an *always disabled* constant.

If the enable input is a Boolean, it may be produced by:

- a digital input from a module on the island
- a digital output from the virtual module (See *The Virtual Module, p. 28*)
- an output on the action module (See *Using the Action Module as an Input to a Block, p. 32*) written to by the fieldbus master

When the enable input is a Boolean 0 or an *always disabled* constant, the block is disabled—the action does not execute and the output is frozen in the state it was in when the block became disabled. The block continues to process inputs but does not act on them. If the block becomes enabled, it immediately begins acting on the latest set of inputs received.

Thresholds

Outside-the-window compare blocks require two threshold values, which serve as the upper and lower bounds of the window. Each TH value needs to be an unsigned integer in the range 0 to 65 535. TH 1 defines the lower boundary of the window; TH 2 defines the upper boundary.

Note: The value of TH 2 must be greater than the value of TH 1.

Operational Input

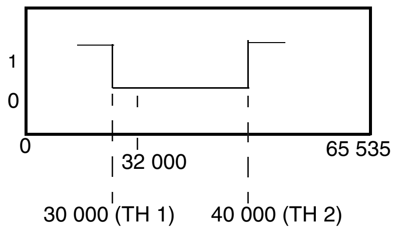
An outside-the-window compare block uses one operational input. It must be a word with an unsigned integer in the range 0 to 65 535. The input can come from:

- an analog input from a module on the island
- an analog output from the virtual module (See *The Virtual Module, p. 28*)
- if the less-than-threshold compare is the second block in a nested reflex action (See *Nesting Two Reflex Blocks, p. 36*), the operational input may be configured as the output of the first reflex block

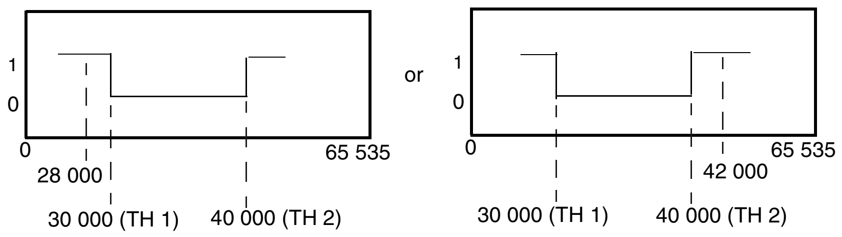
Note: Unsigned compare blocks are often nested together with counter blocks (See *Counter Reflex Blocks, p. 93*). The unsigned compare is always the second block in the nested action, and the analog output from the counter is used as its operational input. These two action types complement each other well because the output from a counter is always unsigned with 16-bit resolution.

Note: Do not use a word that contains a signed negative integer value as the operational input to an unsigned integer compare. The block will misinterpret a value of 1 in the sign bit position (bit 15) as part of the integer value. Avoid the use of modules such as the STB AVI 1270 analog input module, which produces an input with a possible negative integer value, as the source for the operational input to the block.

Suppose that you have two threshold values, where TH 1 = 30 000 and TH 2 = 40 000. Then suppose that the operational input is 32 000 and Δ = 0:



Because the value of the operational input falls inside the window defined by TH 1 and TH 2, the block produces a Boolean 0 as its result.
Alternately, if the value of the operational input is less than TH 1 (say, 28 000) or greater than TH 2 (say 42 000):



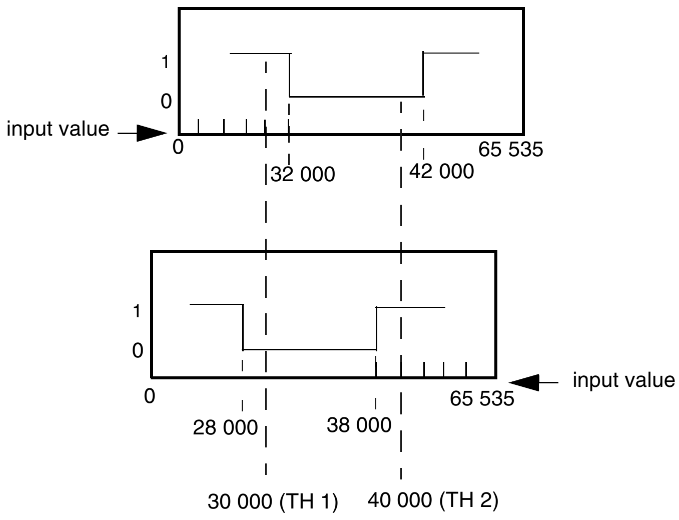
then the block produces a Boolean 1 as its result because the input value is outside the window.

Delta (Δ)

You can also add a Δ value to an outside-the-window compare block, which acts as an hysteresis around the two thresholds.

Note: To be valid, TH 2 - TH 1 must be greater than 2Δ. For example, say that TH 1 = 30 000 and TH 2 = 40 000. The Δ value you assign to the block must therefore be less than 5000.

Suppose you have a window defined by TH 1 = 30 000 and TH 2 = 40 000. To that window, you specify a Δ of 2000. If your operational input value is less than the TH 1 - Δ (28 000) and is increasing in value, the result is 1. The result remains 1 until the input value exceeds 32 000, at which point the result drops to 0. If the input value continues to increase, the result remains at 0 until the input value reaches TH 2 + Δ (42 000). When the input value reaches 42 000, the result rises back to 1.
If the input value is decreasing from a value greater than TH 2 + Δ (42 000), the reflex result is 1 until the input value reaches 38 000. At this point, the result drops to 0 and remains there as the input value becomes less than TH 1 - Δ (28 000). At this point, the result rises back to 1:



While the input value is within the window defined by the threshold and the Δ, it holds its last value.
For example, if the input value is inside the window and increasing, the result will be 0. When it exceeds 42 000, the result rises to 1. If the input value then begins to decrease after the result has dropped, the result will remain at 1 until the input value decreases to 38 000, at which point it will drop to 0 and will remain there until the input value decreases to less than 28 000.

Physical Output

The block produces a Boolean 1 when the input value is outside the window and a Boolean 0 when the input value is within that window. The physical output (See *Configuring the Physical Output from a Reflex Block, p. 27*) needs to be mapped to an action module:

- The action module may be a digital output module on the island bus. In this case, you need to specify one of the digital output channels as the destination for the block's output.
- If the compare is the first block in a nested reflex action (See *Nesting Two Reflex Blocks, p. 36*), the action module needs to be the same action module as the one specified for the second reflex block. Specify the channel as *None*.

When the output from a block is mapped to a channel on a digital output module, that channel becomes dedicated to the reflex action and can no longer use data from the fieldbus master to update its field device. The fieldbus master still has the ability to write data to this bit address in the NIM, and the reflex action editor lets you use this data from the fieldbus master as an input to the block.

Counter Reflex Blocks

5

At a Glance

Overview

This chapter describes two counter reflex blocks that count Boolean inputs either up or down from a preset value. The result from these counter blocks is a word value. One counter increments or decrements on the rising edge of the operational input, and the other increments or decrements on the falling edge of the operational input.

What's in this Chapter?

This chapter contains the following topics:

Falling-edge Counter Block

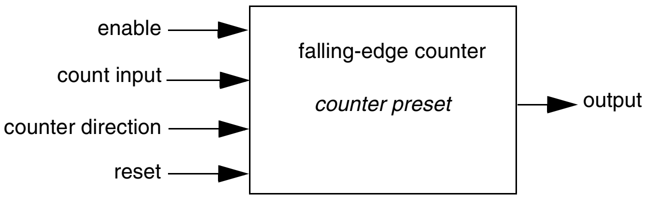
Summary

A falling-edge counter block counts up (increments) or down (decrements) each time its count input falls from 1 to 0. The count begins at a user-specified counter preset value and continues up or down until the block receives a reset input. A reset sends the counter back to its preset value and starts a new counting sequence. The block produces an unsigned analog word as its output.

Note: Unlike other reflex actions, a counter block is designed to act exclusively as the first block in a nested reflex action (See *Nesting Two Reflex Blocks, p. 36*). The output from a counter block is used as an analog input to an unsigned compare block (See *Unsigned Compare Reflex Blocks, p. 71*). As a result, the reflex editor lets you map the output only to a digital action module, even though the output value is analog.

Structure of a Falling-edge Counter Block

A block diagram for a standard falling-edge counter is shown below:

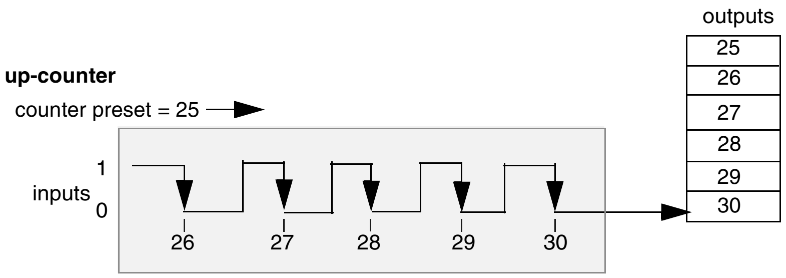


The block has four inputs:
- The enable input turns the counter on or off.
- The count input sends a Boolean value to the block that will generate a count input when it transitions from 1 to 0.
- The counter direction input defines whether the block will increment or decrement on each count.
- The reset will restart the counting operation at the predefined counter preset value.

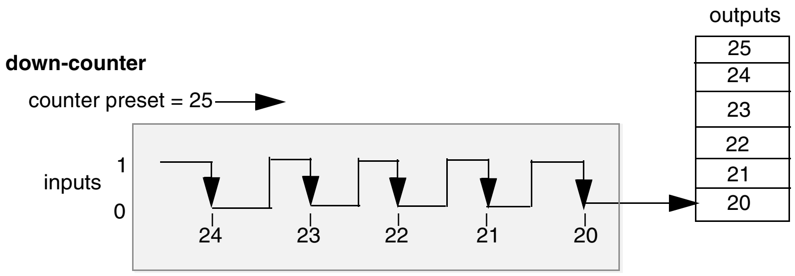
The block also has a counter preset value (See *Configuring Preset Values for a Reflex Block, p. 27*)—an integer value that defines the starting point for each counting operation. You must specify this preset.
The block produces a 16-bit word output on each count. The word holds an unsigned integer value in the range 0 to 65 535. On each count, the output equals the counter preset plus the incremented count or minus the decremented count.

Counter Preset

You must specify the counter preset value before implementing a counter operation. The preset must be an unsigned integer in the range 0 to 65 535. A counting sequence always begins at this counter preset value, then increments or decrements from it each time the count input value falls from 1 to 0.
For example, say you configure an up-counter with a counter preset at 25. The block will start a counting sequence at 25 and will increment by 1 each time the count input falls from 1 to 0:



If you are using a down-counter with a counter preset at 25, the counter will start a counting sequence at 25 and will decrement by 1 each time the count input drops from 1 to 0:



Enable Input

A falling-edge counter block can be enabled either by a Boolean 1 or an *always enabled* constant. It can be disabled by a Boolean 0 or an *always disabled* constant. If the enable input is a Boolean, it may be produced by:

- a digital input from a module on the island
- a digital output from the virtual module (See *The Virtual Module, p. 28*)
- an output on the action module (See *Using the Action Module as an Input to a Block, p. 32*) written to by the fieldbus master

When the enable input is a Boolean 0 or an *always disabled* constant, the block is disabled—the action does not execute and the output is frozen in the state it was in when the block became disabled. The block continues to process inputs but does not act on them. If the block becomes enabled, it immediately begins acting on the latest set of inputs received.

Note: If the count input is 0 when the enable input transitions from 0 to 1, the counter assumes that a falling-edge transition has just taken place and increments or decrements once. If the count input is 1 when the enable input transitions from 0 to 1, the block waits for the next falling-edge transition before it starts counting.

Count Input

A falling-edge counter block receives a stream of Boolean 1s and 0s as an count input. The counter increments or decrements each time the input value falls from 1 to 0. The inputs may come from:

- a constant
- a digital input from a module on the island
- a digital output from the virtual module (See *The Virtual Module, p. 28*)
- an output on the action module (See *Using the Action Module as an Input to a Block, p. 32*) written to by the fieldbus master

Note: At start-up, make sure that the count input provides a 1 to the counter block. If the count input is 0 when the block is enabled, the counter will assume that a falling-edge transition has just taken place and will increment or decrement once. If the count input is 1 when the count is enabled, the block will wait for the next falling-edge transition before it starts counting.

Count Direction Input

Every falling-edge counter block needs to count in a direction—either up or down. Using the Advantys configuration software, set the direction of the counter as a constant value of either 0 or 1, where:
- 0 = an up-counter
- 1 = a down-counter

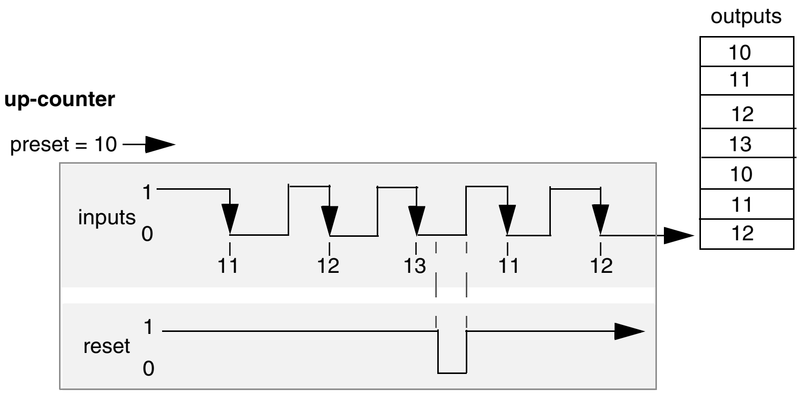
The inputs may come from:
- a constant
- a digital input from a module on the island
- a digital output from the virtual module (See *The Virtual Module, p. 28*)
- an output on the action module (See *Using the Action Module as an Input to a Block, p. 32*) written to by the fieldbus master

Reset Input

Every falling-edge counter action has a reset input. The reset input is a Boolean value. A reset value of 0 returns the counter to the specified preset value. A reset value of 1 allows the counter to continue to increment or decrement. While reset is low, the block does not count.

The reset input can be configured to come from:
- a constant
- a digital input from a module on the island
- a digital output from the virtual module (See *The Virtual Module, p. 28*)
- an output on the action module (See *Using the Action Module as an Input to a Block, p. 32*) written to by the fieldbus master

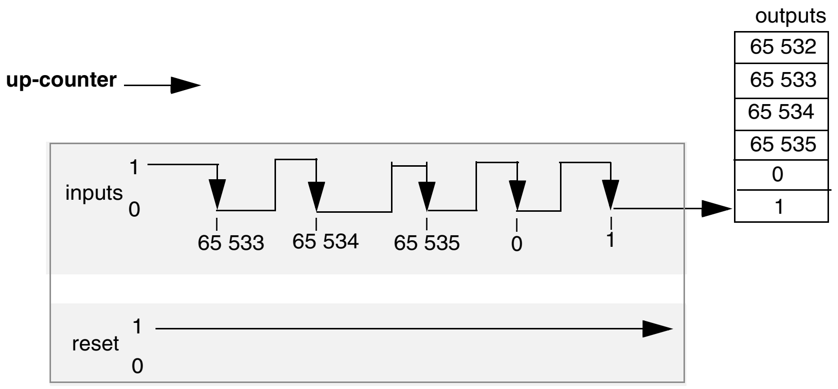
For example, say you have an up-counter with a preset value of 10. The counter will start its counting sequence at 10 and will increment by 1 each time the count input drops from 1 to 0. Suppose that the reset input drops to 0 after three counts:



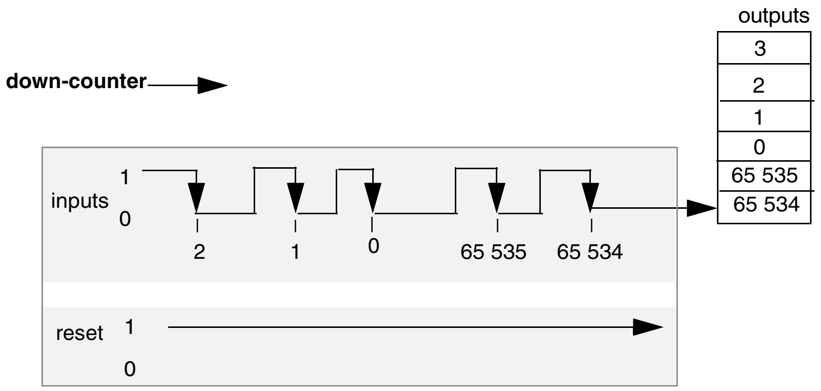
The reset input causes the counter to return to the preset value (10) and to start the up-counting process over again.

Wrap-arounds

If an up-counter increments up to 65 535 and does not receive a reset input, it will wrap to 0 and continue to increment from there until it is reset. At reset, the counter will return to the preset value and start a new incremental up-count.



If a down-counter decrements down to 0 and does not receive a reset input, it will wrap to 65 535 and continue to decrement from there until it is reset. At reset, the counter will return to the preset value and start a new incremental down-count.



Physical Output

The output of a falling-edge counter is a word that holds an unsigned integer value in the range 0 to 65 535. The physical output (See *Configuring the Physical Output from a Reflex Block, p. 27*) needs to be mapped to a digital action module.

Note: A counter is always the first block in a nested reflex action. The action module must always be a digital output module, which will be configured to perform a compare action (preferably an unsigned compare, since values greater than 32 767 would be misinterpreted by a standard integer compare action). The reflex editor does not allow you to map the counter's output to an analog module.

You need to specify the channel to which the counter output will be mapped as *none*—the output will be stored temporarily in an internal reflex buffer, then used as the count input to the compare block.

Power-up and Fallback

Upon power-up, the counter's output data is set to the preset value (if the enable in put is on).
If an error causes the counter block to go to its fallback state, the output freezes in its last active state. Upon removal of the error condition, the counter starts counting again at the point where the output was frozen.

Rising-edge Counter Block

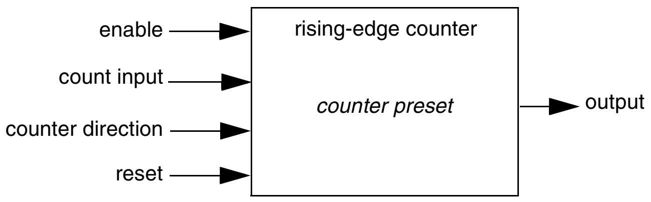
Summary

A rising-edge counter block counts up (increments) or down (decrements) each time an count input to the action rises from 0 to 1. The count begins at a user-specified preset value and continues counting up or down until the block receives a reset input. A reset sends the counter back to its preset value and starts a new counting sequence. The block produces an unsigned analog word as its output.

Note: Unlike other reflex actions, a counter block is designed to act exclusively as the first block in a nested reflex action (See *Nesting Two Reflex Blocks, p. 36*). The output from a counter block is used as an analog input to an unsigned compare block (See *Unsigned Compare Reflex Blocks, p. 71*). As a result, the reflex editor allows you to map the output only to a digital action module, even though the output value is analog.

Structure of a Rising-edge Counter Block

A block diagram for a standard rising-edge counter is shown below:



The counter block has four inputs:
- The enable input turns the counter on or off.
- The count input sends a Boolean value to the block that will generate a count when it transitions from 0 to 1.
- The counter direction input defines whether the action will increment or decrement on each count.
- The reset will restart the counting operation at the predefined counter preset value.

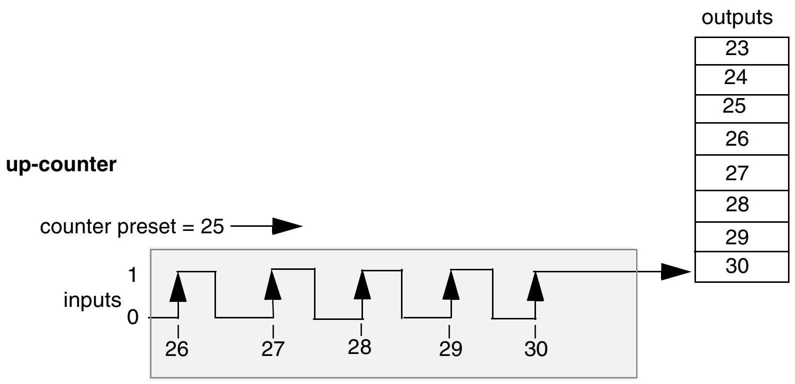
The block also has a counter preset value (See *Configuring Preset Values for a Reflex Block, p. 27*)—an integer value that defines the starting point for each counting operation. You must specify this preset.

The block produces a 16-bit word output on each count. The word holds an unsigned integer value in the range 0 to 65 535. On each count, the output equals the counter preset plus the incremented count or minus the decremented count.

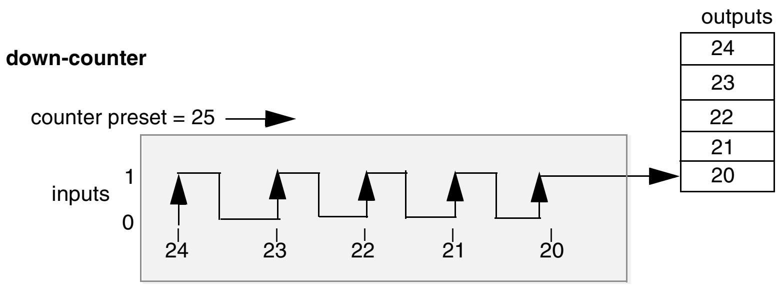
Counter Preset

You must specify the counter preset value before implementing a counter operation. The preset must be an unsigned integer in the range 0 to 65 535. A counting sequence always begins at the counter preset, then increments or decrements from there each time the count input value transitions from 0 to 1.

For example, say you have an up-counter with a preset value of 25. The counter will start a counting sequence at 25 and will increment by 1 each time the count input rises from 0 to 1:



If you are using a down-counter with a preset value of 25, the counter will start a counting sequence at 25 and will decrement by 1 each time the count input rises from 0 to 1:



Enable Input

A rising-edge counter block can be enabled either by a Boolean 1 or an *always enabled* constant. It can be disabled by a Boolean 0 or an *always disabled* constant. If a Boolean input is used, its value may be produced by:

- a digital input or output from a module on the island
- a digital output from the virtual module (See *The Virtual Module, p. 28*)
- an output on the action module (See *Using the Action Module as an Input to a Block, p. 32*) written to by the fieldbus master

When the enable input is a Boolean 0 or an *always disabled* constant, the block is disabled—the action does not execute and the output is frozen in the state it was in when the block became disabled. The block continues to process inputs but does not act on them. If the block becomes enabled, it immediately begins acting on the latest set of inputs received.

Note: If the count input is 1 when the enable input transitions from 0 to 1, the counter assumes that a rising-edge transition has just taken place and increments or decrements once. If the count input is 0 when the enable input transitions from 0 to 1, the block waits for the next rising edge transition before it starts counting.

Count Input

A rising-edge counter block has one count input—a stream of Boolean 1s and 0s. The counter increments or decrements each time the input value rises from 1 to 0. The inputs can come from:

- a constant
- a digital input from a module on the island
- a digital output from the virtual module (See *The Virtual Module, p. 28*)
- an output on the action module (See *Using the Action Module as an Input to a Block, p. 32*) written to by the fieldbus master

Note: At start-up, make sure that the count input provides a 0. If the count input is 1 when the counter becomes enabled, the block will assume that a rising-edge transition has just taken place and will increment or decrement once. If the count input is 0 when the count becomes enabled, the block will wait for the next rising-edge transition before it starts counting.

Count Direction Input

Every rising-edge counter block needs to count in a direction—either up or down. Using the Advantys configuration software, you can set the direction of the counter as a constant value of either 0 or 1, where:

- 0 = an up-counter
- 1 = a down-counter

The input can come from:

- a constant
- a digital input from a module on the island
- a digital output from the virtual module (See *The Virtual Module, p. 28*)
- an output on the action module (See *Using the Action Module as an Input to a Block, p. 32*) written to by the fieldbus master

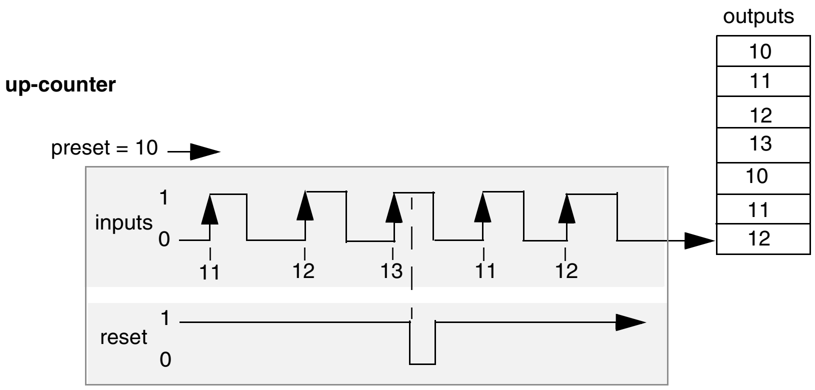
Reset Input

Every rising-edge counter block has a reset input. The reset input is a Boolean value. A reset value of 0 returns the counter to the specified preset value. A reset value of 1 allows the counter to continue to increment or decrement. While reset is low, the counter will not count.

The reset input can be configured to come from:

- a constant
- a digital input from a module on the island
- a digital output from the virtual module (See *The Virtual Module, p. 28*)
- an output on the action module (See *Using the Action Module as an Input to a Block, p. 32*) written to by the fieldbus master

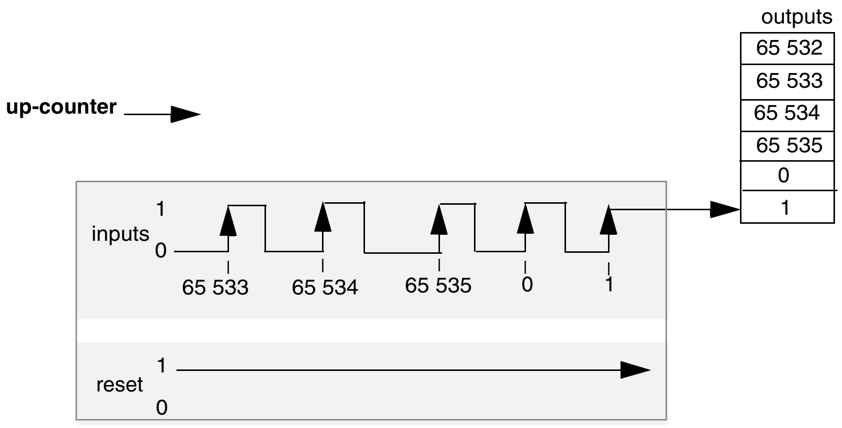
For example, say you have an up-counter with a preset value of 10. The counter will start its counting sequence at 10 and will increment by 1 each time the count input rises from 0 to 1. Suppose that the reset input drops to 0 after three counts:



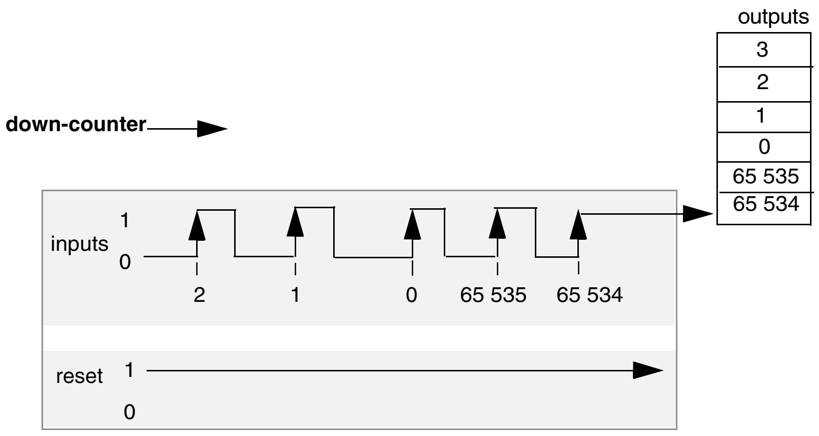
The reset input causes the counter to return to the preset value (10) and to start the up-counting process over again.

Wrap-arounds

If an up-counter increments up to 65 535 and does not receive a reset input, it will wrap to 0 and continue to increment from there until it is reset. At reset, the counter will return to the preset value and start a new incremental up-count.



If a down-counter decrements down to 0 and does not receive a reset input, it will wrap to 65 535 and continue to decrement from there until it is reset. At reset, the counter will return to the preset value and start a new incremental down-count.



Physical Output

The output of a rising-edge counter block is a word that holds an unsigned integer value in the range 0 to 65 535. The physical output (See *Configuring the Physical Output from a Reflex Block, p. 27*) needs to be mapped to a digital action module.

Note: A counter is always the first block in a nested reflex action. The action module must always be a digital output module, which will be configured to perform a compare action (preferably an unsigned compare, since values greater than 32 767 would be misinterpreted by a standard integer compare action). The reflex editor does not allow you to map the counter's output to an analog module.

You need to specify the channel to which the counter output will be mapped as *none*—the output will be stored temporarily in an internal reflex buffer, then used as the count input of the compare block.

Power-up and Fallback

Upon power-up, the counter's output data is set to the preset value (if the enable in put is on).
If an error causes the counter block to go to its fallback state, the output freezes in its last active state. Upon removal of the error condition, the counter starts counting again at the point where the output was frozen.

Timer Reflex Blocks

6

At a Glance

Overview

This chapter describes two types of timer blocks—delay timers and edge timers.
Delay timer blocks start timing when a timer trigger is set, count timing intervals for some specified number of counts, then hold the terminal count value until the trigger launches another timing operation.
Edge time blocks start timing when a timer trigger is set, count timing intervals for some specified number of counts, then return to their start state until the trigger launches a new timing operation.

What's in this Chapter?

This chapter contains the following topics:

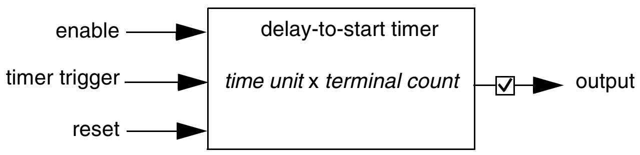
Delay-to-start Timer Block

Summary

A delay-to-start timer block starts a timing operation when its *trigger* rises from 0 to 1. The timer needs to be preset to accumulate a user-specified *time unit* for a specified number of counts (the *terminal count*). The output from a delay-to-start timer block is a Boolean value that rises to 1 when the terminal count is reached and stays at 1 as long as the terminal count is held. You may invert the value of the output.

Structure of a Delay-to-start Timer Block

A block diagram for a delay-to-start timer is shown below:



The timer block has three inputs—an enable input, a timer trigger and a reset. The enable input allows or stops the output from being updated. The timer trigger is essentially a timer start command. The reset input is a Boolean value that stops the timer operation when it is 0.

The block also has two preset values (See *Configuring Preset Values for a Reflex Block, p. 27*)—a time unit and a terminal count. The time unit needs to be specified as some number of ms. The terminal count is a user-defined number of time units. When a timing operation starts, it will accumulate time units from 0 up to the terminal count (as long as the reset is value is 1). When the timer reaches the terminal count, the output turns on and stay on until the timer looses the terminal count. When the timer looses the terminal count, it turns off.

The output is a Boolean value. The standard output is 1 while the block holds the terminal count and 0 when it looses the terminal count. The output may be inverted.

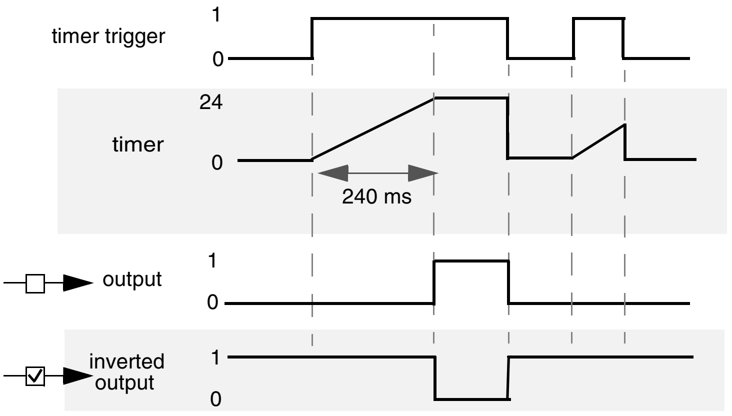
Time Units and Terminal Count

You need to preset the timer block to accumulate in one of the following time units:
- 1 ms
- 10 ms
- 100 ms
- 1000 ms
- 10 000 ms

When the timer is enabled and the trigger starts the accumulation, the block will count a specified number of time units. The maximum number of unit counts allowed is called the *terminal count*. The terminal count is a user-specified integer value in the range 1 to 32 767.

When the timer reaches the terminal count, the accumulator stops counting time units and the output turns on (1 if the output is standard, 0 if the output is inverted). The output remains on as long as the timer accumulator holds the terminal count.

For example, suppose you specify a time unit of 10 ms and a terminal count of 24. When the timer trigger input rises from 0 to 1, the timer accumulates to 240 ms, then stops and holds its terminal count until the trigger drops to 0.



As the timing diagram above shows, a standard output rises to 1 (the inverted output falls to 0) when the terminal count is reached, remains 1 as long as the timer holds the terminal count, and falls to 0 when the timer’s accumulator looses the terminal count. An inverted output falls to 0 when the terminal count is reached, remains 0 as long as the timer holds the terminal count, and rises to 1 when the timer’s accumulator looses the terminal count.

Enable Input

A delay-to-start timer block can be enabled either by a Boolean 1 or an *always enabled* constant. It can be disabled by a Boolean 0 or an *always disabled* constant.
If the enable input is a Boolean, it may be produced by:
- a digital input from a module on the island
- a digital output from the virtual module (See *The Virtual Module, p. 28*)
- an output on the action module (See *Using the Action Module as an Input to a Block, p. 32*) written to by the fieldbus master

When the enable input is a Boolean 0 or an *always disabled* constant, the block is disabled. The timer finishes a timing cycle if it has already started it, but it does not change the output—the output is frozen in the state it was in when the block became disabled. The block continues to process inputs but does not act on them. If the block becomes enabled, it immediately begins acting on the latest set of inputs received.

Timer Trigger Input

The trigger input is a set of Boolean 1s and 0s. The rising edge of the trigger input starts a timing operation, and the falling edge of the trigger input causes the timer accumulator to drop to 0.
The value of the trigger input is important for the output of the block. If the trigger drops to 0 before the timer reaches the terminal count, the timer stops accumulating and drops to 0. When this happens, the output never turns on. If the trigger remains at 1 after the terminal count has been reached, the timer accumulator holds the terminal count value and the output rises to 1.

Note: At start-up, make sure that the trigger input provides a 0 to the timer block. If the trigger is 1 when the timer becomes enabled, the timer assumes that a rising-edge transition has just taken place and starts accumulating counts immediately. If the trigger input is 0 when the block is enabled, the timer waits for the next rising-edge transition before it starts the time-unit accumulation.

The timer trigger input may be produced by:
- a constant
- a digital input from a module on the island
- a digital output from the virtual module (See *The Virtual Module, p. 28*)
- an output on the action module (See *Using the Action Module as an Input to a Block, p. 32*) written to by the fieldbus master
- if the timer is the second block in a nested reflex action (See *Nesting Two Reflex Blocks, p. 36*), its trigger input may be configured as the output of the first reflex block

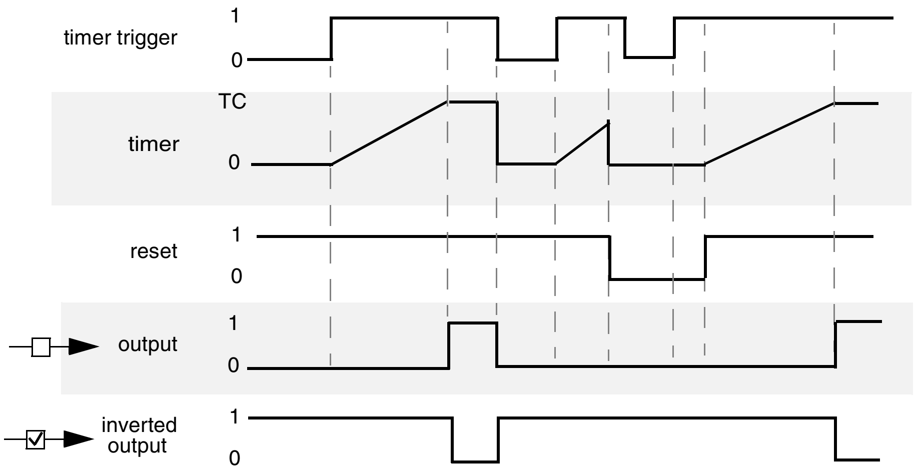
Timer Reset Input

The reset input is essentially a timer override mechanism. It may be a Boolean 1 or 0. The timer is operational when the reset value is 1; it does not operate when the reset value is 0.

The reset input may be produced by:

- a constant
- a digital input from a module on the island
- a digital output from the virtual module (See *The Virtual Module, p. 28*)
- an output on the action module (See *Using the Action Module as an Input to a Block, p. 32*) written to by the fieldbus master

The following timing diagram shows how the value of the reset input effects the output from the timer block:



At the beginning of the timing sequence, when the reset input is 1, the standard output is 0 (the inverted output is 1) while the timer is accumulating. The standard output rises to 1 (the inverted output drops to 0) when the terminal count (TC) is reached. When the trigger drops to 0, the timer and the standard output drop to 0 (or the inverted output rises to 1).

The second time the trigger input rises to 1, the timer begins to accumulate again. But before TC is reached the second time, the reset input drops to 0, thereby resetting the timer. The standard output remains at 0 (or the inverted output remains at 1) during this second timing sequence.

When the reset input rises back to 1, the timer begins to accumulate again starting at 0. The reason that the reset input is able to restart the timer is because the timer trigger input is 1 when the reset rises to 1. Once TC has been reached, the standard output rises to 1 again and stays there (or the inverted output drops to 0 and stays there) as long as both the trigger input and the reset input hold the terminal count.

Physical Output

The output from a delay-to-start timer block is a Boolean 1 or 0.
If the output is not inverted, the output goes to 1 when the block reaches its specified terminal count and stays at 1 as long as the timer accumulator holds the terminal count. The output falls to 0 when the block looses the terminal count.
If the output is inverted, the output goes to 0 when the block reaches its specified terminal count and stays at 0 as long as the timer accumulator holds the terminal count. The output is 1 when the block looses the terminal count.
The physical output (See *Configuring the Physical Output from a Reflex Block, p. 27*) needs to be mapped to an action module:

- The action module may be a digital output module on the island bus. In this case, you need to specify one of the digital output channels as the destination for the block's output.
- If the timer is the first block in a nested reflex action, the action module needs to be the same action module as the one specified for the second reflex block. Specify the channel as *None*.

When the output of a block is mapped to a channel on a digital output module, that channel becomes dedicated to the reflex action and can no longer use data from the fieldbus master to update its field device. The fieldbus master still has the ability to write data to this bit address in the NIM, and the reflex action editor lets you use this data from the fieldbus master as an input to the block.

Power-up and Fallback

Upon power-up, the timer's output data is reset to 0.
If an error causes the timer block to go to its fallback state, the output freezes in its last active state. Upon removal of the error condition, the timer is reset to 0.

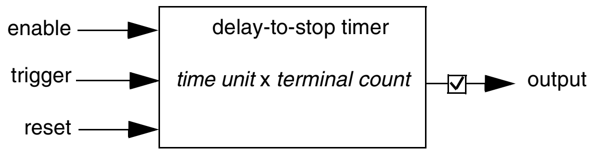
Delay-to-stop Timer Block

Summary

The delay-to-stop timer block starts a timing operation when its *trigger* falls from 1 to 0. The timer needs to be preset to accumulate a user-specified *time unit* for a specified number of counts (the *terminal count*). The output of a delay-to-stop timer block is a Boolean that goes to 0 as soon as the terminal count is reached and stays at 0 as long as the terminal count is held. Optionally, you may invert the value of the output.

Structure of a Delay-to-stop Timer Block

A block diagram for a delay-to-stop timer is shown below:



The timer block has three inputs—an enable input, a timer trigger and a reset. The enable input allows or stops the output from being updated. The timer trigger is essentially a timer start command. The reset input is a Boolean value that stops the timer operation when it is 0.
The block also has two preset values (See *Configuring Preset Values for a Reflex Block, p. 27*)—a time unit and a terminal count. The time unit needs to be specified as some number of ms. The terminal count is a user-defined number of time units. When a timing operation starts, it will accumulate time units from 0 up to the terminal count. When the timer reaches the terminal count, the output turns off until the timer looses the terminal count. When the timer looses the terminal count, it turns on. The output is a Boolean value. The standard output is 0 while the timer holds the terminal count and 1 when the timer looses the terminal count. The output may be inverted.

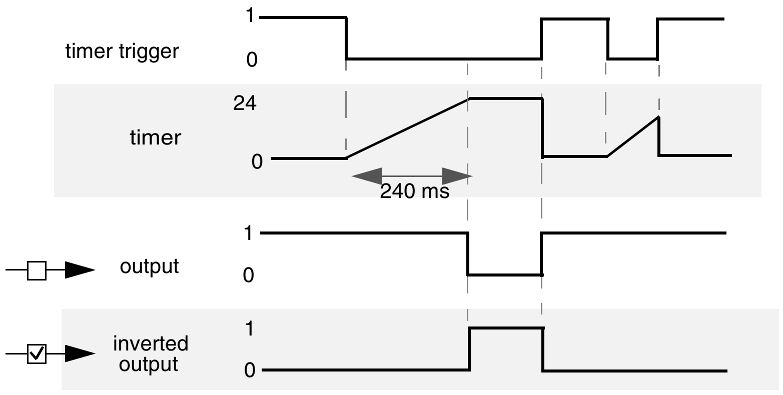
Time Units and Terminal Count

You need to preset the timer block to accumulate in one of the following time units:
- 1 ms
- 10 ms
- 100 ms
- 1000 ms
- 10 000 ms

When the timer block is enabled and the trigger starts the accumulation, the block will count a specified number of time units. The maximum number of unit counts allowed is called the *terminal count*. The terminal count is a user-specified integer value in the range 1 to 32 767.

When the block reaches the terminal count, the accumulator stops counting time units and the output of the action turns off (0 if the output is standard, 1 if the output is inverted). The output remains off as long as the timer accumulator holds the terminal count.

For example, suppose you specify a time unit of 10 ms and a terminal count of 24. When the timer trigger input drops from 1 to 0, the timer accumulates to 240 ms, then stops and holds its terminal count as long as the trigger remains at 0.



As the timing diagram above shows, the standard output falls to 0 (the inverted output rises to 1) when the terminal count is reached. The standard output remains 0 (the inverted output remains 1) while the timer holds the terminal count. The standard output is 1 (the inverted output is 0) whenever the timer is not at the terminal count.

Enable Input

A delay-to-stop timer block can be enabled either by a Boolean 1 or an *always enabled* constant. It can be disabled by a Boolean 0 or an *always disabled* constant.
If the enable input is a Boolean, it may be produced by:

- a digital input from a module on the island
- a digital output from the virtual module (See *The Virtual Module, p. 28*)
- an output on the action module (See *Using the Action Module as an Input to a Block, p. 32*) written to by the fieldbus master

When the enable input is a Boolean 0 or an *always disabled* constant, the block is disabled. The timer finishes a timing cycle if it has already started it, but it does not change the output—the output is frozen in the state it was in when the block became disabled. The block continues to process inputs but does not act on them. If the block becomes enabled, it immediately begins acting on the latest set of inputs received.

Timer Trigger Input

The falling edge of the trigger input starts a timing operation, and the rising edge of the trigger input causes the timer accumulator to drop to 0. The trigger input may be a Boolean 1 or 0.
For a delay-to-stop timer block, the value of the trigger input is important for the output from the block. If the trigger rises to 1 before the timer reaches the terminal count, the timer stops accumulating and drops to 0. When this happens, the output never turns off. If the trigger remains at 0 after the terminal count has been reached, the timer accumulator holds the terminal count value and the output turns off.
The timer trigger value may be produced by:

- a constant
- a digital input from a module on the island
- a digital output from the virtual module (See *The Virtual Module, p. 28*)
- an output on the action module (See *Using the Action Module as an Input to a Block, p. 32*) written to by the fieldbus master
- if the timer is the second part of a nested reflex action (See *Nesting Two Reflex Blocks, p. 36*), its trigger input may be configured as the output of the first reflex action

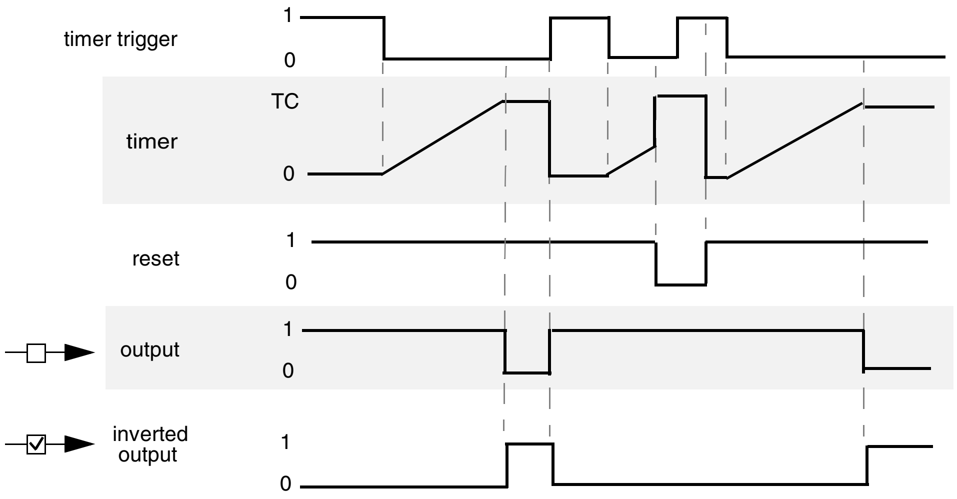
Timer Reset Input

The reset input is essentially a timer override mechanism. It may be a Boolean 1 or 0. The block is operational when the reset value is 1; it does not operate when the reset value is 0.

The reset input may be produced by:

- a constant
- a digital input from a module on the island
- a digital output from the virtual module (See *The Virtual Module, p. 28*)
- an output on the action module (See *Using the Action Module as an Input to a Block, p. 32*) written to by the fieldbus master

The following timing diagram shows how the value of the reset input effects the inverted output from the timer block:



At the beginning of the timing sequence, while the reset input is 1, the standard output is 1 (the inverted output is 0) before and while the block is accumulating. The standard output goes to 0 (the inverted output goes to 1) when TC is reached. When the trigger rises to 1, the timer drops to 0, and the standard output rises to 1 (the inverted output falls to 0).

The second time the trigger input falls to 0, the timer begins to accumulate again. But before the accumulation completes the second time, the reset input drops to 0, thereby sending the timer to 0. The standard output remains at 1 (the inverted output remains at 0).

When the timer trigger value drops to 0 for the third time, the timer begins to accumulate again. Once TC has been reached, the output drops to 0 again and stays at 0 as long as the trigger input is 0 and the reset input is 1.

Physical Output

The output from a delay-to-stop timer block is a Boolean 1 or 0.
If the output is not inverted, the output will go to 0 when the block has reached its specified terminal count and it will stay at 0 as long as the timer accumulator holds the terminal count. The output is 1 when the block looses the terminal count.
If the output is inverted, the output will go to 1 when the block has reached its specified terminal count and will stay at 1 as long as the timer accumulator holds the terminal count. The output is 0 when the block looses the terminal count.
The physical output (See *Configuring the Physical Output from a Reflex Block, p. 27*) needs to be mapped to an action module:

- The action module may be a digital output module on the island bus. In this case, you need to specify one of the digital output channels as the destination for the block's output.
- If the timer is the first block in a nested reflex action, the action module needs to be the same action module as the one specified for the second reflex block. Specify the channel as *None*.

When the output from a reflex block is mapped to a channel on a digital output module, that channel becomes dedicated to the reflex action and can no longer use data from the fieldbus master to update its field device. The fieldbus master still has the ability to write data to this bit address in the NIM, and the reflex action editor lets you use this data from the fieldbus master as an input to the block.

Power-up and Fallback

Upon power-up, the timer's output data is reset to the terminal count.
If an error causes the timer block to go to its fallback state, the output freezes in its last active state. Upon removal of the error condition, the timer is reset to the terminal count.

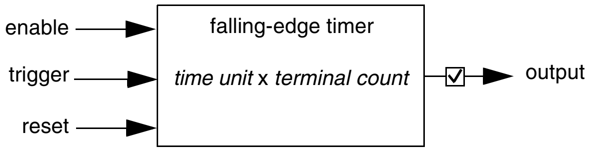
Falling-edge Timer Block

Summary

A falling-edge timer block starts a timing operation when its *trigger* falls from 1 to 0. The timer needs to be preset to accumulate at a user-specified *time unit* for a specified number of counts (the *terminal count*). The output from a falling-edge timer block is a Boolean that goes to 1 while the timer is accumulating and 0 when the timer is not accumulating time units (when the accumulator is at the terminal count). Optionally, you may invert the value of the output.

Structure of a Falling-edge Timer Block

A block diagram for a standard falling-edge timer is shown below:



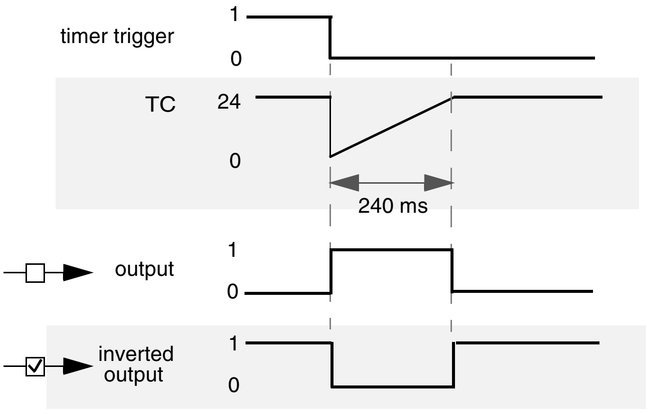
The timer block has three inputs—an enable input, a timer trigger and a reset. The enable input allows or stops the output from being updated. The timer trigger is essentially a timer start command. The reset input is a Boolean value that stops the timer operation when it is 0.
The block also has two preset values (See *Configuring Preset Values for a Reflex Block, p. 27*)—a time unit and a terminal count. The time unit needs to be specified as some number of ms. The terminal count is a user-defined number of time units. When a timing operation starts, it will accumulate time units from 0 up to the terminal count. While the timer is counting, the output turns on. As soon as the timer reaches the terminal count, the output turns off and remains off until the trigger starts a new counting sequence.
The output value is a Boolean. The standard output is 1 while the timer is accumulating counts and 0 when it is not counting. The output may be inverted.

Time Units and Terminal Count

You need to preset the timer block to accumulate in one of the following time units:
- 1 ms
- 10 ms
- 100 ms
- 1000 ms
- 10 000 ms

When the timer is enabled and the trigger starts the accumulation, the block will count a specified number of time units. This number is called the *terminal count*. The terminal count is a user-specified integer value in the range 0 to 32 767.

For example, suppose you specify a time unit of 10 ms and a terminal count of 24. When the timer trigger input drops from 1 to 0, the timer begins accumulating from 0 in 10 ms time units. It accumulates 24 time units (240 ms), then stops accumulating and holds its terminal count.



As the timing diagram above shows, a standard output rises to 1 while the timer is accumulating and falls to 0 whenever the timer is not accumulating. An inverted output falls to 0 while the timer is accumulating and rises to 1 whenever the timer is not accumulating.

Enable Input

A falling-edge timer block can be enabled either by a Boolean 1 or an *always enabled* constant. It can be disabled by a Boolean 0 or an *always disabled* constant. If the enable input is a Boolean, it may be produced by:
- a digital input from a module on the island
- a digital output from the virtual module (See *The Virtual Module, p. 28*)
- an output on the action module (See *Using the Action Module as an Input to a Block, p. 32*) written to by the fieldbus master

When the enable input is a Boolean 0 or an *always disabled* constant, the block is disabled. The timer finishes a timing cycle if it has already started it, but it does not change the output—the output is frozen in the state it was in when the block became disabled. The block continues to process inputs but does not act on them. If the block becomes enabled, it immediately begins acting on the latest set of inputs received.

Timer Trigger Input

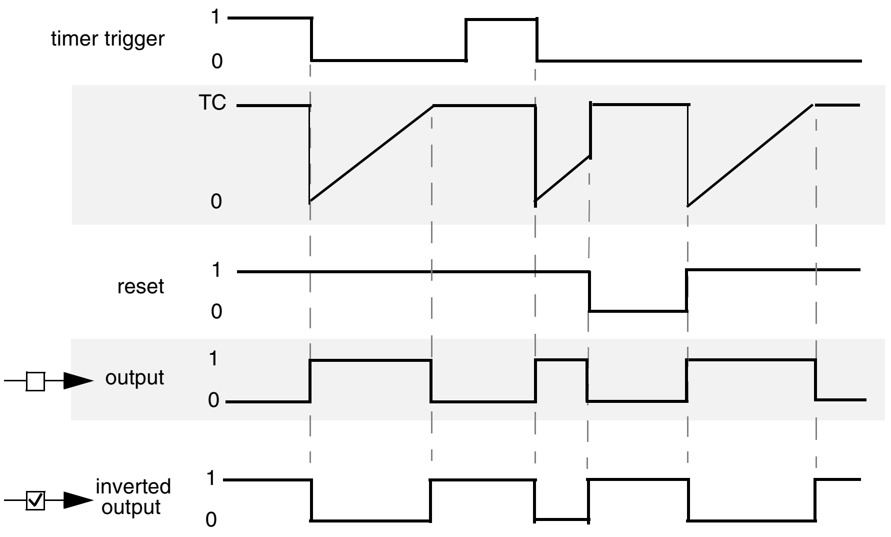
The timer trigger is essentially a timer start command. It may be a Boolean 1 or 0. The block starts accumulating time units when the timer trigger drops from 1 to 0.

Note: At start-up, make sure that the trigger input provides a 1 value to the timer block. If the trigger is 0 when the timer becomes enabled, the timer will assume that a falling edge transition has just taken place and will start accumulating time counts immediately. If the trigger input is 1 when the timer becomes enabled, the timer will wait for the next falling edge transition before it starts the time-unit accumulation.

The timer trigger value may be produced by:
- a constant
- a digital input from a module on the island
- a digital output from the virtual module (See *The Virtual Module, p. 28*)
- an output on the action module (See *Using the Action Module as an Input to a Block, p. 32*) written to by the fieldbus master
- if the timer is the second block in a nested reflex action (See *Nesting Two Reflex Blocks, p. 36*), its trigger input may be configured as the output of the first reflex block

Timer Reset Input

The reset input is essentially a timer override mechanism. It may be a Boolean 1 or 0. The timer is operational when the reset value is 1; it does not operate when the reset value is 0.
The following timing diagram shows how the value of the reset input affects the block's output:



At the beginning of the timing sequence, while the reset input is high, the standard output rises to 1 (the inverted output drops to 0) while the block is accumulating. The standard output drops to 0 (the inverted output rises to 1) when the terminal count (TC) is reached.
The second time that the trigger drops to 0, the block begins to accumulate and the standard output rises to 1 (the inverted output drops to 0). But before TC is reached the second time, the reset input drops to 0, thereby stopping the timer and sending the standard output back to 0 (the inverted output back to 1.
When the reset input rises back to 1, the block begins to accumulate again starting at 0, and the standard output rises again to 1 (the inverted output drops to 0). The reset input is able to restart the timer because the state of the timer trigger input is 0 when the reset rises to 1.

Note: If timer trigger is low when reset goes high, the timer will start. It will not start if the trigger is high.

Physical Output

The output of a falling-edge timer is a Boolean 1 or 0.

If the output is not inverted, the output rises to 1 when the block is accumulating time units and to 0 when the timer is at 0 or at the terminal count.

If the output is inverted, the output drops to 0 when the timer is accumulating time units and to 1 when the timer is at 0 or at the terminal count.

The physical output (See *Configuring the Physical Output from a Reflex Block, p. 27*) needs to be mapped to an action module:

- The action module may be a digital output module on the island bus. In this case, you need to specify one of the digital output channels as the destination for the block's output.
- If the timer is the first block in a nested reflex action, the action module needs to be the same action module as the one specified for the second reflex block. Specify the channel as *None*.

When the output from a block is mapped to a channel on a digital output module, that channel becomes dedicated to the reflex action and can no longer use data from the fieldbus master to update its field device. The fieldbus master still has the ability to write data to this bit address in the NIM, and the reflex action editor lets you use this data from the fieldbus master as an input to the block.

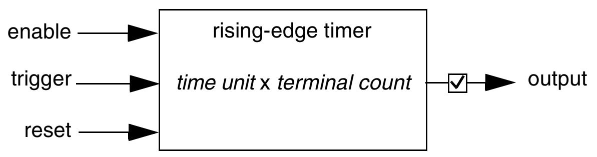
Rising-edge Timer Block

Summary

A rising-edge timer block starts a timing operation when its *trigger* rises from 0 to 1. The block needs to be preset to accumulate at a user-specified *time unit* for a specified number of counts (the *terminal count*). The output from a rising-edge timer block is a Boolean that rises to 1 while the timer is accumulating and drops to 0 when the accumulator is at the terminal count. You may invert the value of the output.

Structure of a Rising-edge Timer Block

A block diagram for a standard rising-edge timer is shown below:

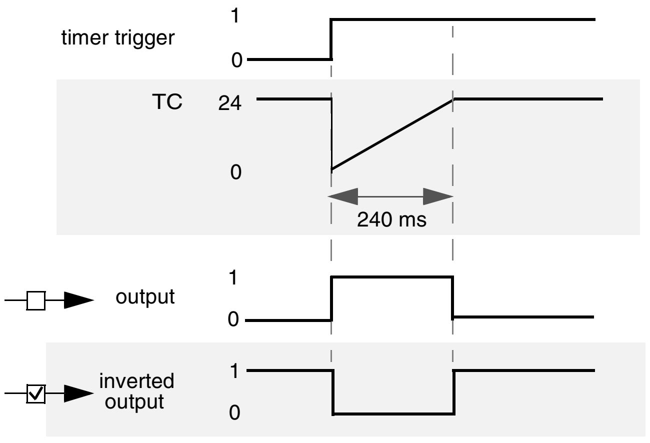


The timer block has three inputs—an enable input, a timer trigger and a reset. The enable input allows or stops the output from being updated. The timer trigger is essentially a timer start command. The reset input is a Boolean value that will make the timer inoperable when it is set to 0.
The block also has two preset values (See *Configuring Preset Values for a Reflex Block, p. 27*)—a time unit and a terminal count. The time unit needs to be specified as some number of ms. The terminal count is a user-defined number of time units. When a timing operation starts, it will accumulate time units from 0 up to the terminal count. While the block is counting, the output turns on. As soon as the timer reaches the terminal count, the output turns off and remains off until the trigger starts a new counting sequence.
The output is a Boolean value. The standard output is 1 while the block is accumulating counts and 0 when it is not counting. The output may be inverted.

Time Units and Terminal Count

You need to preset the timer block to accumulate in one of the following time units:
- 1 ms
- 10 ms
- 100 ms
- 1000 ms
- 10 000 ms

When the timer is enabled and the trigger starts the accumulation, the block will count a specified number of time units. This number is called the *terminal count*. The terminal count is a user-specified integer value in the range 0 to 32 767.
For example, suppose you specify a time unit of 10 ms and a terminal count of 24. When the timer trigger input rises from 0 to 1, the timer begins accumulating from 0 in 10 ms time units. It accumulates 24 time units (240 ms), then stops accumulating and holds its terminal count.



As the timing diagram above shows, a standard output rises to 1 while the timer is accumulating and drops to 0 whenever the timer is not accumulating. An inverted output drops to 0 while the timer is accumulating and rises to 1 whenever the timer is not accumulating.

Enable Input

A rising-edge timer block can be enabled either by a Boolean 1 or an *always enabled* constant. It can be disabled by a Boolean 0 or an *always disabled* constant.
If the enable input is a Boolean, it may be produced by:
- a digital input from a module on the island
- a digital output from the virtual module (See *The Virtual Module, p. 28*)
- an output on the action module (See *Using the Action Module as an Input to a Block, p. 32*) written to by the fieldbus master

When the enable input is a Boolean 0 or an *always disabled* constant, the block is disabled. The timer finishes a timing cycle if it has already started it, but it does not change the output—the output is frozen in the state it was in when the block became disabled. The block continues to process inputs but does not act on them. If the block becomes enabled, it immediately begins acting on the latest set of inputs received.

Timer Trigger Input

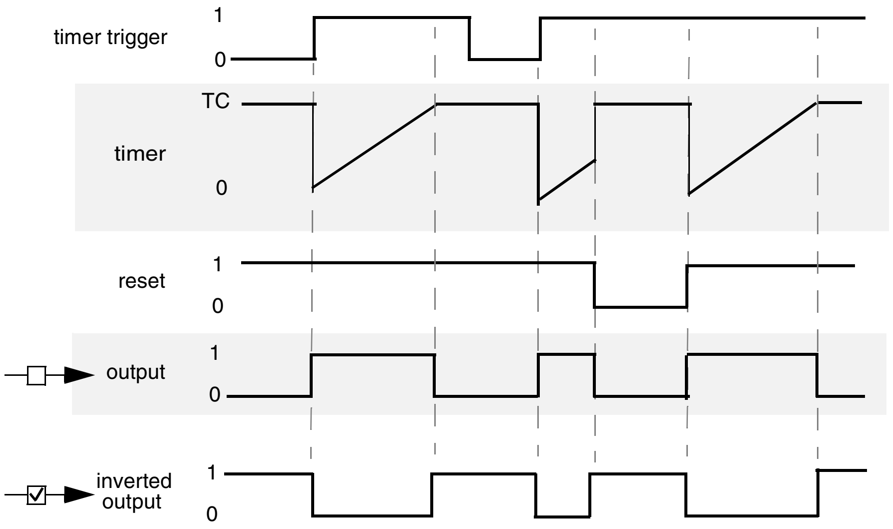
The timer trigger is essentially a timer start command. It may be a Boolean 1 or 0. The timer starts accumulating time units when the timer trigger rises from 0 to 1.

Note: At start-up, make sure that the trigger input provides a 0 to the timer block. If the trigger is 1 when the timer becomes enabled, the block will assume that a rising edge transition has just taken place and will start accumulating time counts immediately. If the trigger input is 0 when the timer becomes enabled, the block will wait for the next rising edge transition before it starts the time-unit accumulation.

The timer trigger value may be produced by:
- a constant
- a digital input from a module on the island
- a digital output from the virtual module (See *The Virtual Module, p. 28*)
- an output on the action module (See *Using the Action Module as an Input to a Block, p. 32*) written to by the fieldbus master
- if the timer is the second block in a nested reflex action (See *Nesting Two Reflex Blocks, p. 36*), its trigger input may be configured as the output of the first reflex block

Timer Reset Input

The reset input is essentially a timer override mechanism. It may be a Boolean 1 or 0. The block is operational when the reset value is 1; it does not operate when the reset value is 0.
The following timing diagram shows how the value of the reset input effects the output from the block:



At the beginning of the timing sequence, while the reset input is 1, the standard output rises to 1 (the inverted output drops to 0) while the block is accumulating The standard output drops to 0 (the inverted output rises to 1) when the terminal count (TC) is reached.
The second time that the trigger drops to 0, the timer begins to accumulate and the standard output rises to 1 again (the inverted output drops to 0 again). But before TC is reached the second time, the reset input drops to 0, thereby stopping the block and sending the standard output to 0 (the inverted output to 1).
When the reset input rises back to 1, the timer begins to accumulate again starting at 0, and the standard output rises again to 1 (the inverted output falls to 0). The reset input restarts the timer because the state of the timer trigger input is high when the reset goes high.

Physical Output

The output of a falling-edge timer is a Boolean 1 or 0.
If the output is not inverted, the output rises to 1 when the timer is accumulating time units and drops to 0 when the timer is at 0 or the terminal count.
If the output is inverted, the output drops to 0 when the timer is accumulating time units and rises to 1 when the timer is at 0 or the terminal count.
The physical output (See *Configuring the Physical Output from a Reflex Block, p. 27*) needs to be mapped to an action module:

- The action module may be a digital output module on the island bus. In this case, you need to specify one of the digital output channels as the destination for the block's output.
- If the timer is the first block in a nested reflex action (See *Nesting Two Reflex Blocks, p. 36*), the action module needs to be the same action module as the one specified for the second reflex block. Specify the channel as *None*.

When the output from a block is mapped to a channel on a digital output module, that channel becomes dedicated to the reflex action and can no longer use data from the fieldbus master to update its field device. The fieldbus master still has the ability to write data to this bit address in the NIM, and the reflex action editor lets you use this data from the fieldbus master as an input to the block.

Analog Latch Reflex Blocks

7

At a Glance

Overview

Two types of analog latch blocks are described in this chapter—edge latches and level latches.
Edge latches latch an analog value on either the rising edge or falling edge of the block's trigger. The output from the block remains latched until the trigger causes another input value to be latched. The output is always a latched value.
Level latches produce an output that is latched when the trigger is at one level (1 or 0) and unlatched when the trigger is not at that level.

What's in this Chapter?

This chapter contains the following topics:

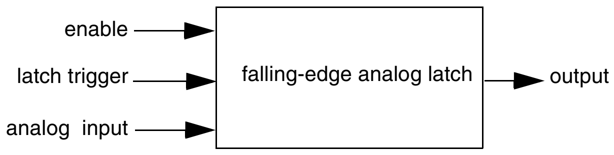
Falling-edge Analog Latch Block

Summary

A falling-edge analog latch block produces an output that latches the value of an analog input when the *trigger* drops from 1 to 0. The output remains latched while the trigger is at 0 and while it transitions back to 1. If the latch trigger transitions from 1 to 0 again, the block latches the output to the value of the analog input at the time of the second transition. The output is always a latched analog value in the form of a 16-bit word.

Structure of a Falling-edge Analog Latch Block

A block diagram for a falling-edge analog latch is shown below:



output latched to the value of the analog input at the latest falling edge of the latch trigger

The latch block has three inputs—an enable input, a latch trigger and an analog input. The enable input turns the block on or off. The latch trigger causes the block to latch onto the value of the analog input at the moment that it transitions from 1 to 0. The analog input is an integer value that is latched when the trigger fires. It may be an unsigned integer value in the range 0 to 65 535 or a signed integer value in the range -32 768 to +32 767.
The output of the block is the latched value. It may be an unsigned integer value in the range 0 to 65 535 or a signed integer value in the range -32 768 to +32 767.

Enable Input

A falling-edge analog latch can be enabled either by a Boolean 1 or an *always enabled* constant. It can be disabled by a Boolean 0 or an *always disabled* constant. If the enable input is a Boolean, it may be produced by:
- a digital input from a module on the island
- a digital output from the virtual module (See *The Virtual Module, p. 28*)

When the enable input is a Boolean 0 or an *always disabled* constant, the block is disabled—the action does not execute and the output is frozen in the state it was in when the block became disabled. The block continues to process inputs but does not act on them. If the block becomes enabled, it immediately begins acting on the latest set of inputs received.

Latch Trigger Input

The latch trigger may be a Boolean 1 or 0. When the value of the trigger falls from 1 to 0, the block latches the value of the analog input and the latched value becomes the block's output. The latched output value remains set until the trigger falls again from 1 to 0, producing a new latched output.

The latch trigger value may be produced by:

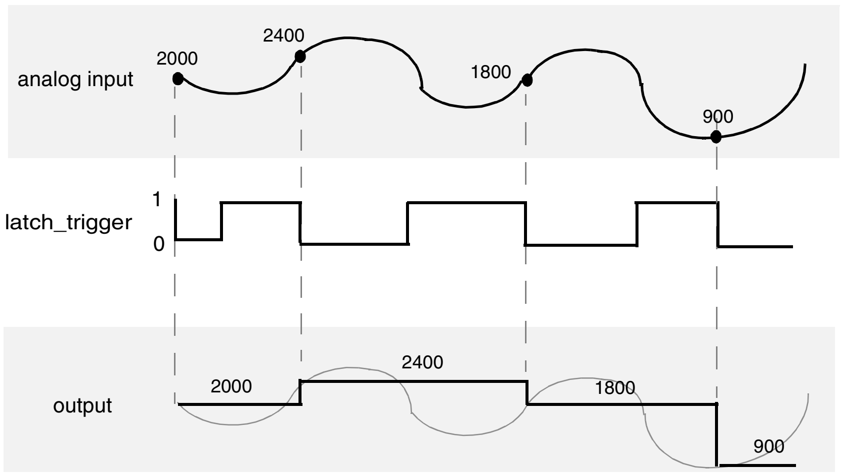
- a constant
- a digital input from a module on the island
- a digital output from the virtual module (See *The Virtual Module, p. 28*)
- if the latch is the second block in a nested reflex action (See *Nesting Two Reflex Blocks, p. 36*), the latch trigger input may be the output from the first reflex block

Note: At start-up, make sure that the trigger input provides a 1 to the latch block. If the trigger is 0 when the latch becomes enabled, the latch assumes that a falling-edge transition has just taken place and immediately latches the value. If the trigger input is 1 when the block is enabled, the latch waits for the next falling-edge transition before it latches a value.

Analog Input

The analog input may be an unsigned integer value in the range 0 to 65 535 or a signed integer value in the range -32 768 to +32 767. The value of the input will be latched by the trigger when it falls from 1 to 0.
The input value may be produced by:
- an analog input from a module on the island
- an analog output from the virtual module (See *The Virtual Module, p. 28*)

The following timing diagram shows how the value of the trigger effects the output of the latch block:



At the beginning of the latch sequence, the analog input value is 2000 when the trigger falls from 1 to 0. The output is latched at 2000.
At the moment when the latch trigger falls from 1 to 0 the second time, the analog input is at 2400. The output is latched at 2400.
When the latch trigger falls from 1 to 0 the third time, the analog input is at 1800. The output is latched at 1800. On the fourth transition of the trigger from 1 to 0, the analog input is at 900. The output is then latched to 900.

Physical Output

The output from a falling-edge analog latch block is a 16-bit word. It may be an unsigned integer in the range 0 to 65 535 or a signed integer in the range -32 768 to +32 767. The output value is the value of the analog input at the moment of the last falling edge of the latch trigger.

Note: The type of output value from this block matches the type of input value—e.g., if you input an unsigned integer value, the output will be an unsigned integer value. The block itself does not discriminate between an unsigned value of 65 535 and a signed value of -32 768. You need to make sure that the block output is being sent to an output module that can handle the output value correctly.

The physical output (See *Configuring the Physical Output from a Reflex Block, p. 27*) needs to be mapped to an action module:

- The action module may be an analog output module on the island bus. In this case, you need to specify one of the analog output channels as the destination for the block's output.
- If the latch is the first block in a nested reflex action (See *Nesting Two Reflex Blocks, p. 36*), the action module needs to be the same action module as the one specified for the second reflex block. Specify the channel as *None*.

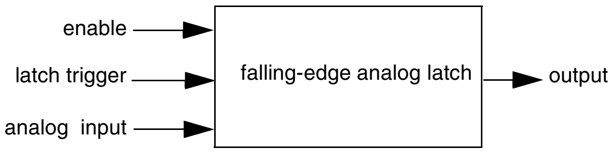
Rising-edge Analog Latch Block

Summary

A rising-edge analog latch block produces a output that latches the value of an analog input when the block's *trigger* rises from 0 to 1. The output remains latched to this value while the trigger is at 1 and while it transitions back to 0. If the latch trigger transitions from 0 to 1 again, the block latches the output to the value of the analog input at the time of the second transition. The output of a rising-edge analog latch action is a latched analog value in the form of a 16-bit word.

Structure of a Rising-edge analog Latch Block

A block diagram for a standard rising-edge analog latch is shown below:



output latched to the value of the analog input at the latest rising edge of the latch trigger

The latch block has three inputs—an enable input, a latch trigger and an analog input. The enable input turns the block action on or off. The latch trigger causes the block to latch onto the value of the analog input when it transitions from 1 to 0. The analog input is an integer value in that is latched when the trigger fires. It may be an unsigned integer value in the range 0 to 65 535 or a signed integer value in the range -32 768 to +32 767.
The output is the current latched value of the block. It may be an unsigned integer value in the range 0 to 65 535 or a signed integer value in the range -32 768 to +32 767.

Enable Input

A rising-edge analog latch can be enabled either by a Boolean 1 or an *always enabled* constant. It can be disabled by a Boolean 0 or an *always disabled* constant. If the enable input is a Boolean, it may be produced by:
- a digital input from a module on the island
- a digital output from the virtual module (See *The Virtual Module, p. 28*)

When the enable input is a Boolean 0 or an *always disabled* constant, the block is disabled—the action does not execute and the output is frozen in the state it was in when the block became disabled. The block continues to process inputs but does not act on them. If the block becomes enabled, it immediately begins acting on the latest set of inputs received.

Latch Trigger Input

The latch trigger may be a Boolean 1 or 0. When the value of the trigger rises from 0 to 1, the block latches the value of the analog input and that latched value becomes the block's output. The latched output value remains set until the trigger rises again from 0 to 1, producing a new latched output.
The latch trigger value may be produced by:

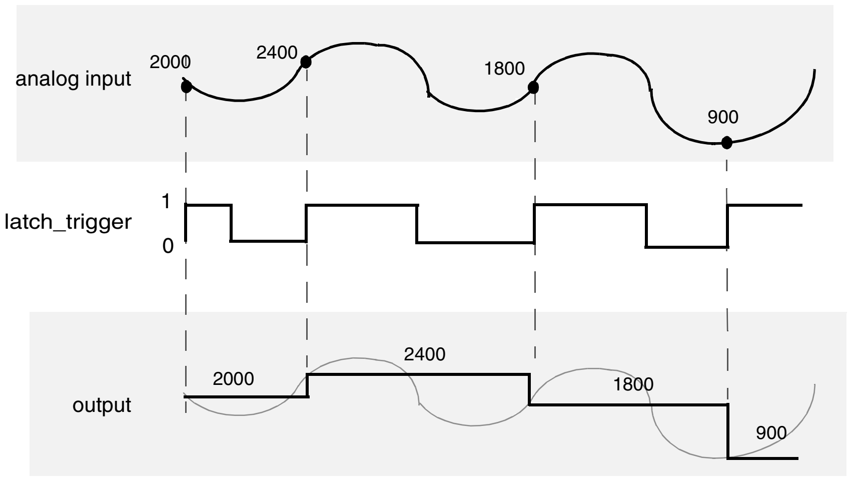
- a constant
- a digital input from a module on the island
- a digital output from the virtual module (See *The Virtual Module, p. 28*)
- if the latch is the second block in a nested reflex action (See *Nesting Two Reflex Blocks, p. 36*), the latch trigger input may be the output from the first reflex block

Note: At start-up, make sure that the trigger input provides a 0 to the latch block. If the trigger is 1 when the latch becomes enabled, the latch assumes that a rising-edge transition has just taken place and immediately latches the value. If the trigger input is 0 when the block is enabled, the latch waits for the next rising-edge transition before it latches a value.

Analog Input

The analog input may be an unsigned integer value in the range 0 to 65 535 or a signed integer value in the range -32 768 to +32 767. The value of the input is latched by the trigger value when it rises from 0 to 1.
The input value may be produced by:
- an analog input from a module on the island
- an analog output from the virtual module (See *The Virtual Module, p. 28*)

The following timing diagram shows how the value of the trigger effects the output from the block:



At the beginning of the latch sequence, the analog input is at a value of 2000 when the trigger rises to 1. The output of the action is latched at 2000.
At the moment when the latch trigger rises to 1 the second time, the analog input is at 2400. The output is then latched at 2400.
When the latch trigger rises to 1 the third time, the analog input is at 1800. The output is latched at 1800. On the fourth transition of the trigger from 0 to 1, the analog input is at 900. The output is then latched to 900.

Physical and Logical Output

The output of a rising-edge analog latch block is a 16-bit word. It may be an unsigned integer in the range 0 to 65 535 or a signed integer in the range -32 768 to +32 767. The output is the value of the analog input at the moment of the last falling edge of the latch trigger.

Note: The type of output value from this block matches the type of input value—e.g., if you input an unsigned integer value, the output will be an unsigned integer value. The block itself does not discriminate between an unsigned value of 65 535 and a signed value of -32 768. You need to make sure that the block output is being sent to an output module that can handle the output value correctly.

The physical output (See *Configuring the Physical Output from a Reflex Block, p. 27*) needs to be mapped to an action module:
- The action module may be an analog output module on the island bus. In this case, you need to specify one of the analog output channels as the destination for the block's output.
- If the latch is the first block in a nested reflex action (See *Nesting Two Reflex Blocks, p. 36*), the action module needs to be the same action module as the one specified for the second reflex block. Specify the channel as *None*.

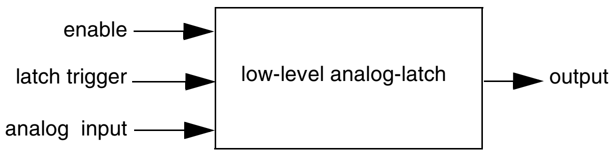
Low-level Analog Latch Block

Summary

A low-level analog latch block produces a latched output when the its *trigger* is 0 and an unlatched output when the trigger is 1. When the action is unlatched, the value of the output is identical to the value of the analog input. When the action is latched, the value of the output is latched to the value of the analog input at the moment when the latch trigger fell from 1 to 0. The output of a low-level analog latch action is a latched or unlatched analog value in the form of a 16-bit word.

Structure of a Low-level Analog Latch Block

A block diagram for a low-level analog latch is shown below:



output equal to the analog input when latch trigger = 1

output latched to the analog input when latch trigger = 0

The latch block has three inputs—an enable input, a latch trigger and an analog input. The enable input turns the latch block action on or off. The latch trigger causes the block to latch onto the value of the analog input when it transitions from 1 to 0. The analog input is an integer value that is latched when the trigger fires. It may be an unsigned integer value in the range 0 to 65 535 or a signed integer value in the range -32 768 to +32 767.
The output is the value of the block, either latched or unlatched. It may be an unsigned integer value in the range 0 to 65 535 or a signed integer value in the range -32 768 to +32 767.

Enable Input

A low-level analog latch can be enabled either by a Boolean 1 or an *always enabled* constant. It can be disabled by a Boolean 0 or an *always disabled* constant.
If the enable input is a Boolean, it may be produced by:
- a digital input from a module on the island
- a digital output from the virtual module (See *The Virtual Module, p. 28*)

When the enable input is a Boolean 0 or an *always disabled* constant, the block is disabled—the action does not execute and the output is frozen in the state it was in when the block became disabled. The block continues to process inputs but does not act on them. If the block becomes enabled, it immediately begins acting on the latest set of inputs received.

Latch Trigger Input

The latch trigger may be a Boolean 1 or 0. When the value is 0, the block latches the value of the analog input, and that latched value becomes the output from the block. When the trigger value is 1, the output is unlatched and equal to the value of the analog input.

The latch trigger value may be produced by:

- a constant
- a digital input from a module on the island
- a digital output from the virtual module
- if the latch is the second block in a nested reflex action (See *Nesting Two Reflex Blocks, p. 36*), the latch trigger input may be the output from the first reflex block

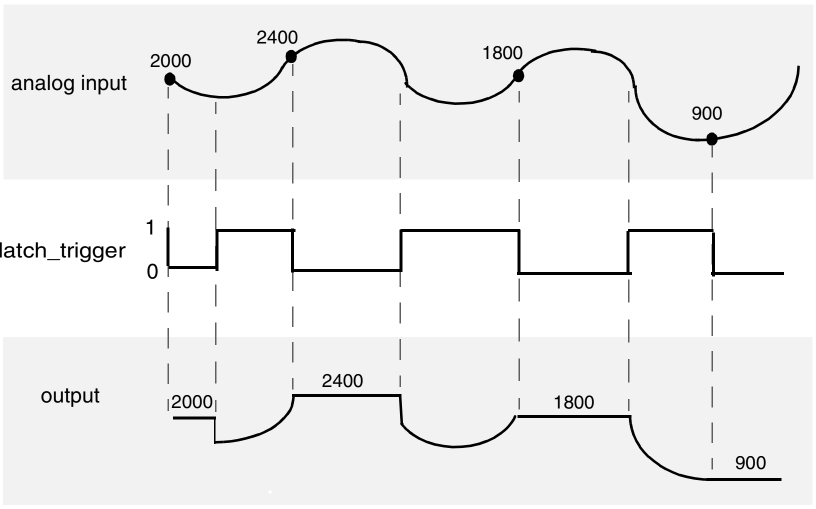
Analog Input

The analog input may be an unsigned integer value in the range 0 to 65 535 or a signed integer value in the range -32 768 to +32 767. The value of the input is latched when the value of the latch trigger is 0 and unlatched when the value of the latch trigger is 1.

The input value may be produced by:

- an analog input from a module on the island
- an analog output from the virtual module (See *The Virtual Module, p. 28*)

The following timing diagram shows how the value of the trigger effects the block's output:



At the beginning of the latch sequence, the analog input is at a value of 2000 when the trigger drops from 1 to 0. The block's output is latched at 2000.

At the moment when the latch trigger falls from 1 to 0 the second time, the analog input is at 2400. The output is then latched at 2400.

When the latch trigger falls from 1 to 0 the third time, the analog input is at 1800. The output is latched at 1800. On the fourth transition of the trigger from 1 to 0, the analog input is at 900. The output is then latched to 900.

Physical Output

The output of a low-level analog latch block is a 16-bit word. It may be an unsigned integer in the range 0 to 65 535 or a signed integer in the range -32 768 to +32 767. The output is latched to the value of the analog input when the value of the latch trigger is 0, and it is unlatched when the value of the trigger is 1.

Note: The type of output value from this block matches the type of input value—e.g., if you input an unsigned integer value, the output will be an unsigned integer value. The block itself does not discriminate between an unsigned value of 65 535 and a signed value of -32 768. You need to make sure that the block output is being sent to an output module that can handle the output value correctly.

The physical output (See *Configuring the Physical Output from a Reflex Block, p. 27*) needs to be mapped to an action module:

- The action module may be an analog output module on the island bus. In this case, you need to specify one of the analog output channels as the destination for the block's output.
- If the latch is the first block in a nested reflex action (See *Nesting Two Reflex Blocks, p. 36*), the action module needs to be the same action module as the one specified for the second reflex block. Specify the channel as *None*.

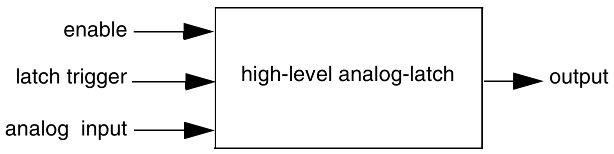
High-level Analog Latch Block

Summary

A high-level analog latch block produces a latched output when the block's *trigger* is 1 and an unlatched output when the trigger is 0. When the block is unlatched, the value of the output is identical to the value of the analog input. When the block is latched, the value of the output is latched to the value of the analog input when the latch trigger rises from 0 to 1. The output is an analog value in the form of a 16-bit word.

Structure of a High-level Analog Latch Block

A block diagram for a high-level analog latch is shown below:



output equal to the analog input when latch trigger = 0

output latched to the analog input when latch trigger = 1

The latch block has three inputs—an enable input, a latch trigger and an analog input. The enable input turns the block on or off. The latch trigger causes the block to latch onto the value of the analog input at the moment that it transitions from 1 to 0. The analog input is an integer value that is latched to a value when the trigger fires. It may be an unsigned integer value in the range 0 to 65 535 or a signed integer value in the range -32 768 to +32 767.
The output is the value of the block, either latched or unlatched. It may be an unsigned integer value in the range 0 to 65 535 or a signed integer value in the range -32 768 to +32 767.

Enable Input

A high-level analog latch block can be enabled either by a Boolean 1 or an *always enabled* constant. It can be disabled by a Boolean 0 or an *always disabled* constant. If the enable input is a Boolean, it may be produced by:
- a digital input from a module on the island
- a digital output from the virtual module (See *The Virtual Module, p. 28*)

When the enable input is a Boolean 0 or an *always disabled* constant, the block is disabled—the action does not execute and the output is frozen in the state it was in when the block became disabled. The block continues to process inputs but does not act on them. If the block becomes enabled, it immediately begins acting on the latest set of inputs received.

Latch Trigger Input

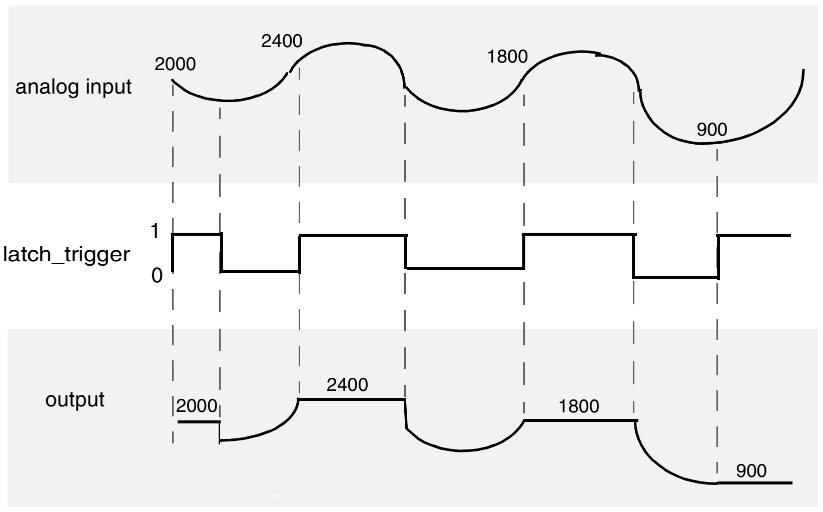
The latch trigger may be a Boolean 1 or 0. When the trigger is 1, the value of the analog input is latched, and that value becomes the output from the block as long as the trigger is 1. When the trigger value is 0, the block latches the value of the analog input and that latched value becomes the block's output. When the trigger value is 1, the output is unlatched and equal to the value of the analog input.
The latch trigger value may be produced by:

- a constant
- a digital input from a module on the island
- a digital output from the virtual module
- if the latch is the second block in a nested reflex action (See *Nesting Two Reflex Blocks, p. 36*), the latch trigger input may be the output from the first reflex block

Analog Input

The analog input may be an unsigned integer value in the range 0 to 65 535 or a signed integer value in the range -32 768 to +32 767. The value of the input is latched when the latch trigger is 1 and unlatched when the latch trigger is 0.
The input value may be produced by:
- an analog input from a module on the island
- an analog output from the virtual module (See *The Virtual Module, p. 28*)

The following timing diagram shows how the value of the trigger effects the output from the block:



At the beginning of the timing sequence, the analog input is at a value of 2000 when the trigger rises from 0 to 1. The output of the action is latched at 2000.
At the moment when the latch trigger rises from 0 to 1 the second time, the analog input is at 2400. The output is then latched at 2400.
When the latch trigger rises from 0 to 1 the third time, the analog input is at 1800. The output is latched at 1800. On the fourth transition of the trigger from 0 to 1, the analog input is at 900. The output is then latched to 900.

Physical Output

The output of a high-level analog latch block is a 16-bit word. It may be an unsigned integer in the range 0 to 65 535 or a signed integer in the range -32 768 to +32 767. The output is latched to the value of the analog input when the latch trigger is 1 and unlatched when the trigger is 0.

Note: The type of output value from this block matches the type of input value—e.g., if you input an unsigned integer value, the output will be an unsigned integer value. The block itself does not discriminate between an unsigned value of 65 535 and a signed value of -32 768. You need to make sure that the block output is being sent to an output module that can handle the output value correctly.

The physical output (See *Configuring the Physical Output from a Reflex Block, p. 27*) needs to be mapped to an action module:

- The action module may be an analog output module on the island bus. In this case, you need to specify one of the analog output channels as the destination for the reflex output.
- If the latch is the first block in a nested reflex action (See *Nesting Two Reflex Blocks, p. 36*), the action module needs to be the same action module as the one specified for the second reflex block. Specify the channel as *None*.

Digital Latch Reflex Blocks

8

At a Glance

Overview

Two types of digital latch blocks are described in this chapter—edge latches and level latches.
Edge latch blocks latch a digital value on the rising edge or falling edge of the block's trigger. The output remains latched until the trigger causes another input value to be latched; the output is always a latched value.
Level latches produce an output that is latched when the trigger value is at one level (1 or 0) and unlatched when the trigger value is not at that level.

What's in this Chapter?

This chapter contains the following topics:

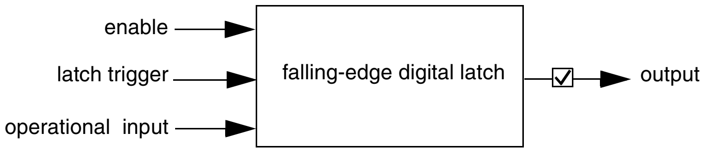
Falling-edge Digital Latch Block

Summary

A falling-edge digital latch block produces an output that latches the value of an operational input when the *trigger* falls from 1 to 0. The output remains latched to that digital value. The output from the block is a Boolean 1 or 0. You may invert the value of the output.

Structure of a Falling-edge Digital Latch Block

A block diagram for a falling-edge digital latch is shown below:



standard output latched to the value of the operational input at the latest falling edge of the latch trigger

inverted output latched to the inverse value of the operational input at the latest falling edge of the latch trigger

The latch block has three inputs—an enable input, a latch trigger and an operational input. The enable input turns the block on or off. The latch trigger causes the block to latch onto a digital value. The operational input is a stream of Boolean values on which the latch operates.
The standard output is the value of the operational input at the moment when the latch trigger falls from 1 to 0. The output may be inverted so that it is a Boolean 0 when the latched input is 1 and a Boolean 1 when the latched input is 0. By placing a check mark in the checkbox on the output line, you invert the output value.

Enable Input

A falling-edge digital latch block can be enabled either by a Boolean 1 or an *always enabled* constant. It can be disabled by a Boolean 0 or an *always disabled* constant. If the enable input is a Boolean, it may be produced by:
- a digital input from a module on the island
- a digital output from the virtual module (See *The Virtual Module, p. 28*)
- an output on the action module (See *Using the Action Module as an Input to a Block, p. 32*) written to by the fieldbus master

When the enable input is a Boolean 0 or an *always disabled* constant, the block is disabled—the action does not execute and the output is frozen in the state it was in when the block became disabled. The block continues to process inputs but does not act on them. If the block becomes enabled, it immediately begins acting on the latest set of inputs received.

Latch Trigger Input

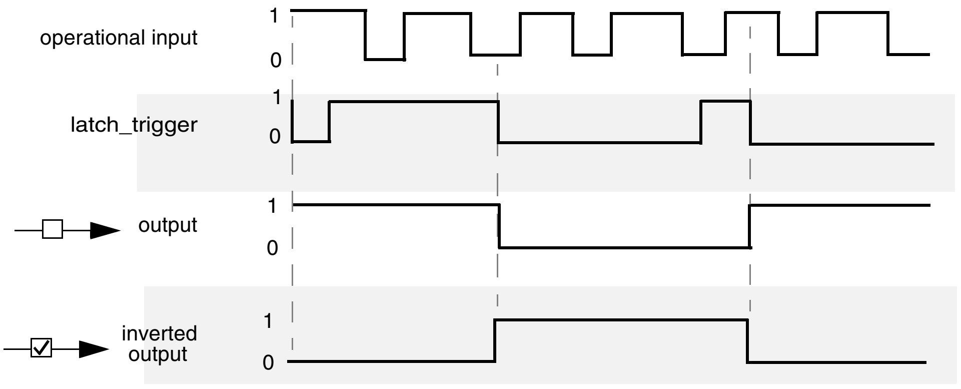
The latch trigger may be a Boolean 1 or 0. When it transitions from 1 to 0, the value of the operational input is latched. If the output is standard, the value of the operational input becomes the output from the block; if the output is inverted, the inverted value of the operational input becomes the output from the block.
The latch trigger value may be produced by:
- a constant
- a digital input from a module on the island
- a digital output from the virtual module (See *The Virtual Module, p. 28*)
- an output on the action module (See *Using the Action Module as an Input to a Block, p. 32*) written to by the fieldbus master
- if the latch is the second block in a nested reflex action (See *Nesting Two Reflex Blocks, p. 36*), the latch trigger input may be the output from the first reflex block

Operational Input

The operational input is a stream of Boolean 1s and 0s that is latched by the falling edge of the latch trigger. It may be produced by:

- a constant
- a digital input from a module on the island
- a digital output from the virtual module (See *The Virtual Module, p. 28*)
- an output on the action module (See *Using the Action Module as an Input to a Block, p. 32*) written to by the fieldbus master
- if the latch is the second block in a nested reflex action (See *Nesting Two Reflex Blocks, p. 36*), the latch trigger input may be the output from the first reflex block

The following timing diagram shows how the value of the trigger effects the output of the latch action:



At the beginning of the sequence, the operational input is high when the latch trigger falls from 1 to 0. The standard output is latched at 1 (the inverted output is latched at 0).

The output value remains latched until the latch trigger falls from 1 to 0 a second time. At that moment, the operational input is low. The standard output is latched at 0 (the inverted output is latched at 1).

When the latch trigger falls from 1 to 0 the third time, the operational input is high. The standard output is latched again at 1 (the inverted output is latched at 0).

Physical Output

The output from a falling-edge digital latch block is a Boolean 1 or 0. The output is always a latched value, determined by the value of the operational input when the trigger transitions from 1 to 0.

If the output is not inverted, the output latches the value of the operational input when the latch trigger falls from 1 to 0.

If the output is inverted, the output latches the inverse of the value of the operational input when the latch trigger falls from 1 to 0.

The physical output (See *Configuring the Physical Output from a Reflex Block, p. 27*) needs to be mapped to an action module:

- The action module may be a digital output module on the island bus. In this case, you need to specify one of the digital output channels as the destination for the block's output.
- If the latch is the first block in a nested reflex action (See *Nesting Two Reflex Blocks, p. 36*), the action module needs to be the same action module as the one specified for the second reflex block. Specify the channel as *None*.

When the output from a block is mapped to a channel on a digital output module, that channel becomes dedicated to the reflex action and can no longer use data from the fieldbus master to update its field device. The fieldbus master still has the ability to write data to this bit address in the NIM, and the reflex action editor lets you use this data from the fieldbus master as an input to the block.

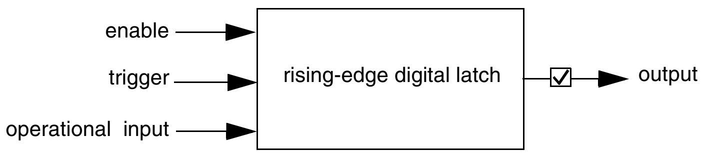
Rising-edge Digital Latch Block

Summary

A rising-edge digital latch block produces a output that latches the value of an operational input when the block's *trigger* rises from 0 to 1. The output remains latched when the trigger falls to 0 and until the trigger rises to 1 again. The output is a Boolean 1 or 0. You may invert the value of the output.

Structure of a Rising-edge Digital Latch Block

A block diagram for a rising-edge digital latch is shown below:



standard output latched to the value of the operational input at the latest falling edge of the latch trigger

inverted output latched to the inverse value of the operational input at the latest falling edge of the latch trigger

The latch block has three inputs—an enable input, a latch trigger and an operational input. The enable input turns the block on or off. The latch trigger causes the block to latch onto a digital value. The operational input is the stream of Boolean values on which the latch operates.
The standard output is the value of the operational input at the moment when the latch trigger rises from 0 to 1. The output may be inverted so that it is a Boolean 0 when the latched operational input value is 1 and a Boolean 1 when the latched operational input value is 0. By placing a check mark in the checkbox on the output line, you invert the output value.

Enable Input

A rising-edge digital latch block can be enabled either by a Boolean 1 or an *always enabled* constant. It can be disabled by a Boolean 0 or an *always disabled* constant.
If the enable input is a Boolean, it may be produced by:
- a digital input from a module on the island
- a digital output from the virtual module (See *The Virtual Module, p. 28*)
- an output on the action module (See *Using the Action Module as an Input to a Block, p. 32*) written to by the fieldbus master

When the enable input is a Boolean 0 or an *always disabled* constant, the block is disabled—the action does not execute and the output is frozen in the state it was in when the block became disabled. The block continues to process inputs but does not act on them. If the block becomes enabled, it immediately begins acting on the latest set of inputs received.

Latch Trigger Input

The latch trigger may be a Boolean 1 or 0. When it rises from 0 to 1, the value of the operational input is latched. If the output is standard, the value of the operational input becomes the output of the action; if the output is inverted, the inverted value of the operational input becomes the output of the action.
The latch trigger value may be produced by:

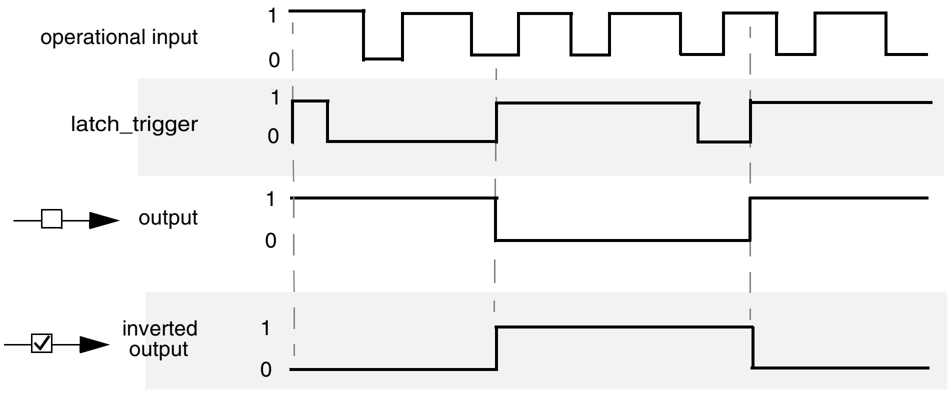
- a constant
- a digital input from a module on the island
- a digital output from the virtual module
- an output on the action module written to by the fieldbus master
- if the latch is the second block in a nested reflex action (See *Nesting Two Reflex Blocks, p. 36*), the latch trigger input may be the output from the first reflex block

Operational Input

The operational input is a pulse train of Boolean 1s and 0s that will be latched at any time by the rising edge of the latch trigger. It may be produced by:

- a digital input or output from a module on the island
- a digital output from the virtual module
- an input data bit from a channel on the action module
- if the latch is the second block in a nested reflex action (See *Nesting Two Reflex Blocks, p. 36*), the latch trigger input may be the output from the first reflex block

The following timing diagram shows how the value of the trigger effects the output from the block:



At the beginning of the timing sequence, the operational input is high when the trigger rises from 0 to 1. The standard output is latched at 1(the inverted output is latched at 0).

The output value remains latched until the latch trigger rises from 0 to 1 a second time. At that moment, the operational input is low. The standard output is latched at 0 (the inverted output is latched at 1).

When the latch trigger rises from 0 to 1 the third time, the operational input is high. The standard output is latched again at 1 (the inverted output is latched at 0).

Physical Output

The output from a rising-edge digital latch block is a Boolean 1 or 0. The output is always a latched value, determined by the value of the operational input when the trigger rises from 0 to 1.

If the output is not inverted, the output latches the value of the operational input when the latch trigger rises from 0 to 1.

If the output is inverted, the output latches the inverse of the value of the operational input when the latch trigger rises from 0 to 1.

The physical output (See *Configuring the Physical Output from a Reflex Block, p. 27*) needs to be mapped to an action module:

- The action module may be a digital output module on the island bus. In this case, you need to specify one of the digital output channels as the destination for the block's output.
- If the latch is the first block in a nested reflex action (See *Nesting Two Reflex Blocks, p. 36*), the action module needs to be the same action module as the one specified for the second reflex block. Specify the channel as *None*.

When the output from a block is mapped to a channel on a digital output module, that channel becomes dedicated to the reflex action and can no longer use data from the fieldbus master to update its field device. The fieldbus master still has the ability to write data to this bit address in the NIM, and the reflex action editor lets you use this data from the fieldbus master as an input to the block.

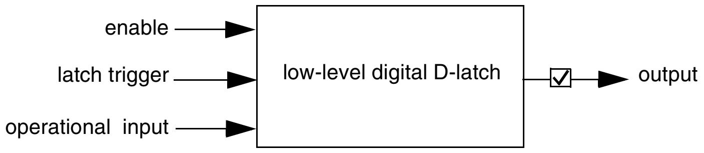
Low-level Digital D-latch Block

Summary

A low-level digital D-latch block produces a latched output when the block's *trigger* is 0 and an unlatched output when the trigger is 1. The output is a Boolean 1 or 0. You may invert the value of the output.

Structure of a Low-level Digital D-latch Block

A block diagram for a standard low-level digital D-latch is shown below:



standard output latched to a specific operational input value when the trigger = 0; unlatched and equal to the current operational input when the trigger = 1

inverted output latched to the inversion of a specific operational input value when the trigger = 0; unlatched and equal to the inverse of the current operational input when the trigger = 1

The latch block has three inputs—an enable input, a latch trigger and an operational input. The enable input turns the block on or off. The latch trigger causes the block to latch onto or unlatch from a digital value. The operational input is a stream of Boolean values on which the block operates.
When a standard output is unlatched, it will echoes the current operational input. When an inverted output is unlatched, it will echoes the inverse of the current operational input.
When a standard output is latched, it holds the value of the operational input when the latch trigger drops from 1 to 0. When an inverted output is latched, it will hold the inverse of the value of the operational input when the trigger drops from 1 to 0.

Enable Input

A low-level digital D-latch can be enabled either by a Boolean 1 or an *always enabled* constant. It can be disabled by a Boolean 0 or an *always disabled* constant. If the enable input is a Boolean, it may be produced by:
- a digital input from a module on the island
- a digital output from the virtual module (See *The Virtual Module, p. 28*)
- an output on the action module (See *Using the Action Module as an Input to a Block, p. 32*) written to by the fieldbus master

When the enable input is a Boolean 0 or an *always disabled* constant, the block is disabled—the action does not execute and the output is frozen in the state it was in when the block became disabled. The block continues to process inputs but does not act on them. If the block becomes enabled, it immediately begins acting on the latest set of inputs received.

Latch Trigger Input

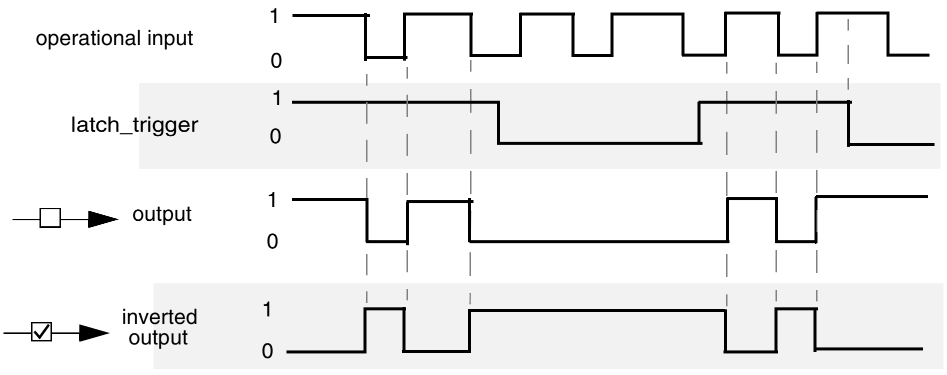
The latch trigger may be a Boolean 1 or 0. When it is 0, the value of the operational input is latched. When the trigger value is 1, the output is unlatched.
The latch trigger value may be produced by:
- a constant
- a digital input from a module on the island
- a digital output from the virtual module
- an output on the action module written to by the fieldbus master
- if the latch is the second block in a nested reflex action (See *Nesting Two Reflex Blocks, p. 36*), the latch trigger input may be the output from the first reflex block

Operational Input

The operational input is a stream of Boolean 1s and 0s that can be latched and unlatched by the latch trigger. It may be produced by:

- a constant
- a digital input from a module on the island
- a digital output from the virtual module
- an output on the action module written to by the fieldbus master
- if the latch is the second block in a nested reflex action (See *Nesting Two Reflex Blocks, p. 36*), the latch trigger input may be the output from the first reflex block

The following timing diagram shows how the value of the trigger effects the output from the block:



At the beginning of the timing sequence, the standard output echoes the value of the operational input (the inverted output echoes the inverse of the operational input) as long as the trigger is high. When the trigger drops from 1 to 0 the first time, the operational input is low. The trigger latches the standard output at 0 (the inverted output at 1), and holds it there as long as the trigger stays low.
When the latch trigger rises back to 1, the standard output begins to echo the operational input again (the inverted output echoes the inverse of the operational input again).
When the latch trigger falls from 1 to 0 the second time, the operational input is high. The trigger latches the standard output at 1 (the inverted output at 0).

Physical Output

The output from a low-level digital D-latch block is a Boolean 1 or 0. The output is latched when the trigger value is 0 and unlatched when the trigger value is 1.

If the output is not inverted, it echoes the current operational input when the latch trigger is high, and it latches the value of the operational input at the moment that the trigger drops from 1 to 0.

If the output is inverted, it echoes the inverse of the current operational input when the latch trigger is high, and it latches a value that is the inverse of the value of the operational input at the moment that the trigger drops from 1 to 0.

The physical output (See *Configuring the Physical Output from a Reflex Block, p. 27*) needs to be mapped to an action module:

- The action module may be a digital output module on the island bus. In this case, you need to specify one of the digital output channels as the destination for the block's output.
- If the latch is the first block in a nested reflex action (See *Nesting Two Reflex Blocks, p. 36*), the action module needs to be the same action module as the one specified for the second reflex block. Specify the channel as *None*.

When the output from a block is mapped to a channel on a digital output module, that channel becomes dedicated to the reflex action and can no longer use data from the fieldbus master to update its field device. The fieldbus master still has the ability to write data to this bit address in the NIM, and the reflex action editor lets you use this data from the fieldbus master as an input to the block.

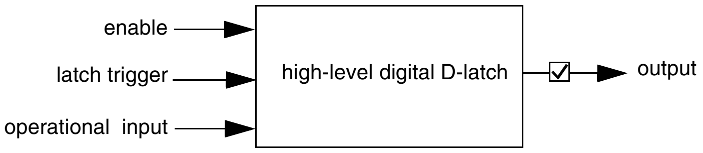
High-level Digital D-latch Block

Summary

A high-level digital D-latch block produces a latched output when the block's *trigger* is 1 and an unlatched output when the trigger is 0. The output is a Boolean 1 or 0. You may invert the value of the output.

Structure of a High-level Digital D-latch Block

A block diagram for a standard high-level digital D-latch is shown below:



standard output latched to a specific operational input value when the trigger = 1; unlatched and equal to the current operational input when the trigger = 0

inverted output latched to the inversion of a specific operational input value when the trigger = 1; unlatched and equal to the inverse of the current operational input when the trigger = 0

The latch block has three inputs—an enable input, a latch trigger and an operational input. The enable input turns the block on or off. The latch trigger causes the block to latch onto or unlatch from a digital value. The operational input is a stream of Boolean values on which the block operates.
When a standard output is unlatched, it echoes the current operational input. When an inverted output is unlatched, it echoes the inverse of the current operational input. When a standard output is latched, it holds the value of the operational input when the latch trigger rises from 0 to 1. When an inverted output is latched, it hold the inverse of the value of the operational input when the trigger rises from 0 to 1.

Enable Input

A high-level digital D-latch block can be enabled either by a Boolean 1 or an *always enabled* constant. It can be disabled by a Boolean 0 or an *always disabled* constant. If the enable input is a Boolean, it may be produced by:
- a digital input from a module on the island
- a digital output from the virtual module (See *The Virtual Module, p. 28*)
- an output on the action module (See *Using the Action Module as an Input to a Block, p. 32*) written to by the fieldbus master

When the enable input is a Boolean 0 or an *always disabled* constant, the block is disabled—the action does not execute and the output is frozen in the state it was in when the block became disabled. The block continues to process inputs but does not act on them. If the block becomes enabled, it immediately begins acting on the latest set of inputs received.

Latch Trigger Input

The latch trigger may be a Boolean 1 or 0. When it is 1, the value of the operational input is latched. When the trigger value is 0, the output is unlatched.
The latch trigger value may be produced by:

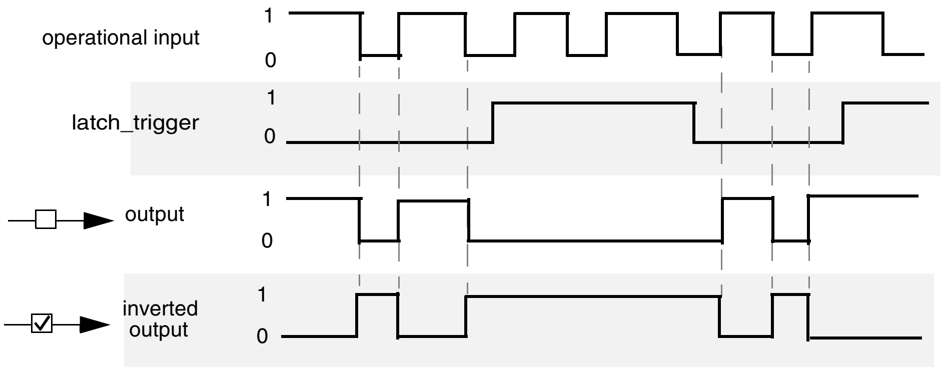
- a constant
- a digital input from a module on the island
- a digital output from the virtual module (See *The Virtual Module, p. 28*)
- an output on the action module (See *Using the Action Module as an Input to a Block, p. 32*) written to by the fieldbus master
- if the latch is the second block in a nested reflex action (See *Nesting Two Reflex Blocks, p. 36*), the latch trigger input may be the output from the first reflex block

Operational Input

The operational input is a stream of Boolean 1s and 0s that will be latched and unlatched by the trigger value. It may be produced by:

- a constant
- a digital input from a module on the island
- a digital output from the virtual module (See *The Virtual Module, p. 28*)
- an output on the action module (See *Using the Action Module as an Input to a Block, p. 32*) written to by the fieldbus master
- if the latch is the second block in a nested reflex action (See *Nesting Two Reflex Blocks, p. 36*), the latch trigger input may be the output from the first reflex block

The following timing diagram shows how the value of the trigger effects the output from the block:



At the beginning of the timing sequence, the standard output echoes the value of the operational input (the inverted output echoes the inverse of the operational input) as long as the trigger is low. When the trigger transitions from 0 to 1 the first time, the operational input is low. The trigger latches the standard output at 0 (the inverted output at 1), and holds it there as long as the trigger stays high.

At the moment when the latch trigger transitions back to 0, the standard output begins to echo the operational input again (the inverted output echoes the inverse of the operational input again).

When the latch trigger transitions from 0 to 1 the second time, the operational input is high. The trigger latches the standard output at 1 (the inverted output at 0).

Physical Output

The output from a high-level digital D-latch block is a Boolean 1 or 0. The output is a latched value when the trigger is 1 and unlatched when the trigger is 0.

If the output is not inverted, it echoes the current operational input when the latch trigger is low, and it latches the value of the operational input at the moment that the trigger rises from 0 to 1.

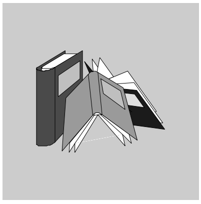
If the output is inverted, it echoes the inverse of the current operational input when the latch trigger is low, and it latches a value that is the inverse of the value of the operational input at the moment that the trigger rises from 0 to 1.

The physical output (See *Configuring the Physical Output from a Reflex Block, p. 27*) needs to be mapped to an action module:

- The action module may be a digital output module on the island bus. In this case, you need to specify one of the digital output channels as the destination for the block's output.
- If the latch is the first block in a nested reflex action (See *Nesting Two Reflex Blocks, p. 36*), the action module needs to be the same action module as the one specified for the second reflex block. Specify the channel as *None*.

When the output from a block is mapped to a channel on a digital output module, that channel becomes dedicated to the reflex action and can no longer use data from the fieldbus master to update its field device. The fieldbus master still has the ability to write data to this bit address in the NIM, and the reflex action editor lets you use this data from the fieldbus master as an input to the block.

Glossary



!

10Base-T	An adaptation of the IEEE 802.3 (Ethernet) standard, the 10Base-T standard uses twisted-pair wiring with a maximum segment length of 100 m (328 ft) and terminates with an RJ-45 connector. A 10Base-T network is a baseband network capable of transmitting data at a maximum speed of 10 Mbit/s.
802.3 frame	A frame format, specified in the IEEE 802.3 (Ethernet) standard, in which the header specifies the data packet length.

A

agent	**1.** SNMP—the SNMP application that runs on a network device. **2.** Fipio—a slave device on a network.
analog input	A module that contains circuits that convert analog DC input signals to digital values that can be manipulated by the processor. By implication, these analog inputs are usually direct—i.e., a data table value directly reflects the analog signal value.
analog output	A module that contains circuits that transmit an analog DC signal proportional to a digital value input to the module from the processor. By implication, these analog outputs are usually direct—i.e., a data table value directly controls the analog signal value.
application object	In CAN-based networks, application objects represent device-specific functionality, such as the state of input or output data.

ARP	*address resolution protocol.* IP's network layer protocol uses ARP to map an IP address to a MAC (hardware) address.
auto baud	The automatic assignment and detection of a common baud rate as well as the ability of a device on a network to adapt to that rate.
auto-addressing	The assignment of an address to each island bus I/O module and preferred device.
auto-configuration	The ability of island modules to operate with predefined default parameters. A configuration of the island bus based completely on the actual assembly of I/O modules.

B

BootP	*bootstrap protocol.* A UDP/IP protocol that allows an internet node to obtain its IP parameters based on its MAC address.
BOS	*beginning of segment.* When more than one segment of I/O modules is used in an island, an STB XBE 1200 BOS module is installed in the first position in each extension segment. Its job is to carry island bus communications to and generate logic power for the modules in the extension segment.
bus arbitrator	A master on a Fipio network.

C

CAN	*controller area network.* The CAN protocol (ISO 11898) for serial bus networks is designed for the interconnection of smart devices (from multiple manufacturers) in smart systems for real-time industrial applications. CAN multi-master systems ensure high data integrity through the implementation of broadcast messaging and advanced error mechanisms. Originally developed for use in automobiles, CAN is now used in a variety of industrial automation control environments.
CANopen protocol	An open industry standard protocol used on the internal communication bus. The protocol allows the connection of any standard CANopen device to the island bus.
CI	*command interface.*

CiA	*CAN in Automation*. CiA is a non-profit group of manufacturers and users dedicated to developing and supporting CAN-based higher layer protocols.
COB	*communication object.* A communication object is a unit of transportation (a "message") in a CAN-based network. Communication objects indicate a particular functionality in a device. They are specified in the CANopen communication profile.
COMS	*island bus scanner.*
configuration	The arrangement and interconnection of hardware components within a system and the hardware and software selections that determine the operating characteristics of the system.
CRC	*cyclic redundancy check.* Messages that implement this error checking mechanism have a CRC field that is calculated by the transmitter according to the message's content. Receiving nodes recalculate the field. Disagreement in the two codes indicates a difference between the transmitted message and the one received.

D

DeviceNet protocol	DeviceNet is a low-level, connection-based network that is based on CAN, a serial bus system without a defined application layer. DeviceNet, therefore, defines a layer for the industrial application of CAN.
DHCP	*dynamic host configuration protocol.* A TCP/IP protocol that allows a server to assign an IP address based on a role name (host name) to a network node.
differential input	A type of input design where two wires (+ and -) are run from each signal source to the data acquisition interface. The voltage between the input and the interface ground are measured by two high-impedance amplifiers, and the outputs from the two amplifiers are subtracted by a third amplifier to yield the difference between the + and - inputs. Voltage common to both wires is thereby removed. Differential design solves the problem of ground differences found in single-ended connections, and it also reduces the cross-channel noise problem.
digital I/O	An input or output that has an individual circuit connection at the module corresponding directly to a data table bit or word that stores the value of the signal at that I/O circuit. It allows the control logic to have discrete access to the I/O values.
DIN	*Deutsche industrial norms.* A German agency that sets engineering and dimensional standards and now has worldwide recognition.

E

EDS	*electronic data sheet.* The EDS is a standardized ASCII file that contains information about a network device's communications functionality and the contents of its object dictionary. The EDS also defines device-specific and manufacturer-specific objects.
EIA	*Electronic Industries Association.* An organization that establishes electrical/ electronic and data communication standards.
EMC	*electromagnetic compatibility.* Devices that meet EMC requirements can operate within a system's expected electromagnetic limits without error.
EMI	*electromagnetic interference.* EMI can cause an interruption, malfunction, or disturbance in the performance of electronic equipment. It occurs when a source electronically transmits a signal that interferes with other equipment.
EOS	*end of segment.* When more than one segment of I/O modules is used in an island, an STB XBE 1000 EOS module is installed in the last position in every segment except the final segment on the island. Its job is to extend island bus communications and send 24 VDC for logic power to the next segment.
Ethernet	A LAN cabling and signaling specification used to connect devices within a defined area, e.g., a building. Ethernet uses a bus or a star topology to connect different nodes on a network.0
Ethernet II	A frame format in which the header specifies the packet type, Ethernet II is the default frame format for STB NIP 2212 communications.

F

fallback state	A safe state to which an Advantys STB I/O module can return in the event that its communication connection fails.
fallback value	The value that a device assumes during fallback. Typically, the fallback value is either configurable or the last stored value for the device.
FED_P	*Fipio extended device profile.* On a Fipio network, the standard device profile type for agents whose data length is more than eight words and equal to or less than thirty-two words.

Fipio	*Fieldbus Interface Protocol (FIP).* An open fieldbus standard and protocol that conforms to the FIP/World FIP standard. Fipio is designed to provide low-level configuration, parameterization, data exchange, and diagnostic services.
Flash memory	Flash memory is nonvolatile memory that can be overwritten. It is stored on a special EEPROM that can be erased and reprogrammed.
FRD_P	*Fipio reduced device profile.* On a Fipio network, the standard device profile type for agents whose data length is two words or less.
FSD_P	*Fipio standard device profile.* On a Fipio network, the standard device profile type for agents whose data length is more than two words and equal to or less than eight words.
full scale	The maximum level in a specific range—e.g., in an analog input circuit the maximum allowable voltage or current level is at full scale when any increase beyond that level is over-range.
function block	A function block performs a specific automation function, such as speed control. A function block comprises configuration data and a set of operating parameters.
function code	A function code is an instruction set commanding one or more slave devices at a specified address(es) to perform a type of action, e.g., read a set of data registers and respond with the content.

G

gateway	A program or /hardware that passes data between networks.
global_ID	*global_identifier.* A 16-bit integer that uniquely identifies a device's location on a network. A global_ID is a symbolic address that is universally recognized by all other devices on the network.
GSD	*generic slave data* (file). A device description file, supplied by the device's manufacturer, that defines a device's functionality on a Profibus DP network.

H

HMI	**human-machine interface** An operator interface, usually graphical, for industrial equipment.
HMI	*human-machine interface* An operator interface, usually graphical, for industrial equipment.
hot swapping	Replacing a component with a like component while the system remains in operation.
HTTP	*hypertext transfer protocol.* The protocol that a web server and a client browser use to communicate with one another.

I

I/O base	A mounting device, designed to seat an I/O module, hang it on a DIN rail, and connect it to the island bus. It provides the connection point where the module can receive either 24 VDC or 115/230 VAC from the input or output power bus distributed by a PDM.
I/O module	In a programmable controller system, an I/O module interfaces directly to the sensors and actuators of the machine/process. This module is the component that mounts in an I/O base and provides electrical connections between the controller and the field devices. Normal I/O module capacities are offered in a variety of signal levels and capacities.
I/O scanning	The continuous polling of the Advantys STB I/O modules performed by the COMS to collect data bits, status, error, and diagnostics information.
IEC	*International Electrotechnical Commission Carrier.* Founded in 1884 to focus on advancing the theory and practice of electrical, electronics, and computer engineering, and computer science. IEC 1131 is the specification that deals with industrial automation equipment.
IEC type 1 input	Type 1 digital inputs support sensor signals from mechanical switching devices such as relay contacts and push buttons operating in normal environmental conditions.

IEC type 1+ input	Type 1+ digital inputs support sensor signals from mechanical switching devices such as relay contacts, push buttons (in normal-to-moderate environmental conditions), three-wire proximity switches and two-wire proximity switches that have: ● a voltage drop of no more than 8 V ● a minimum operating current capability less than or equal to 2 mA ● a maximum off-state current less than or equal to 0.8 mA
IEC type 2 input	Type 2 digital inputs support sensor signals from solid state devices or mechanical contact switching devices such as relay contacts, push buttons (in normal or harsh environmental conditions), and two- or three-wire proximity switches.
IEEE	*Institute of Electrical and Electronics Engineers, Inc.* The international standards and conformity assessment body for all fields of electrotechnology, including electricity and electronics.
industrial I/O	An Advantys STB I/O module designed at a moderate cost for typical continuous, high-duty-cycle applications. Modules of this type often feature standard IEC threshold ratings, usually providing user-configurable parameter options, on-board protection, good resolution, and field wiring options. They are designed to operate in moderate-to-high temperature ranges.
input filtering	The amount of time that a sensor must hold its signal on or off before the input module detects the change of state.
input polarity	An input channel's polarity determines when the input module sends a 1 and when it sends a 0 to the master controller. If the polarity is *normal*, an input channel will send a 1 to the controller when its field sensor turns on. If the polarity is *reverse*, an input channel will send a 0 to the controller when its field sensor turns on.
input response time	The time it takes for an input channel to receive a signal from the field sensor and put it on the island bus.
INTERBUS protocol	The INTERBUS fieldbus protocol observes a master/slave network model with an active ring topology, having all devices integrated in a closed transmission path.
IP	*internet protocol.* That part of the TCP/IP protocol family that tracks the internet addresses of nodes, routes outgoing messages, and recognizes incoming messages.

L

LAN	*local area network.* A short-distance data communications network.
light industrial I/O	An Advantys STB I/O module designed at a low cost for less rigorous (e.g., intermittent, low-duty-cycle) operating environments. Modules of this type operate in lower temperature ranges with lower qualification and agency requirements and limited on-board protection; they usually have limited or no user-configuration options.
linearity	A measure of how closely a characteristic follows a straight-line function.
LSB	*least significant bit, least significant byte.* The part of a number, address, or field that is written as the rightmost single value in conventional hexadecimal or binary notation.

M

MAC address	*media access control address.* A 48-bit number, unique on a network, that is programmed into each network card or device when it is manufactured.
mandatory module	When an Advantys STB I/O module is configured to be mandatory, it must be present and healthy in the island configuration for the island to be operational. If a mandatory module fails or is removed from its location on the island bus, the island will go into a pre-operational state. By default, all I/O modules are not mandatory. You must use the Advantys configuration software to set this parameter.
master/slave model	The direction of control in a network that implements the master/slave model is always from the master to the slave devices.
Modbus	Modbus is an application layer messaging protocol. Modbus provides client and server communications between devices connected on different types of buses or networks. Modbus offers many services specified by function codes.
MSB	*most significant bit, most significant byte.* The part of a number, address, or field that is written as the leftmost single value in conventional hexadecimal or binary notation.

N

N.C. contact	*normally closed* contact. A relay contact pair that is closed when the relay coil is de-energized and open when the coil is energized.
N.O. contact	*normally open.* contact. A relay contact pair that is open when the relay coil is de-energized and closed when the coil is energized.
NEMA	*National Electrical Manufacturers Association.*
network cycle time	The time that a master requires to complete a single scan of all of the configured I/O modules on a network device; typically expressed in microseconds.
NIM	*network interface module.* This module is the interface between an island bus and the fieldbus network of which the island is a part. The network interface module's built-in power supply provides 5 V logic power to the Advantys STB I/O modules as well as 24 V source power, as needed, to the support I/O modules. The NIM also has an RS-232 interface that is the connection point for the Advantys configuration software.
NMT	*network management.* NMT protocols provide services for network initialization, error control, and device status control.

O

object dictionary	Sometimes called the "object directory," this part of the CANopen device model is a map to the internal structure of CANopen devices (according to CANopen profile DS-401). A given device's object dictionary is a lookup table that describes the data types, communications objects, and application objects the device uses. By accessing a particular device's object dictionary structure through the CANopen fieldbus, you can predict its network behavior and, therefore, build a distributed application that implements it.
open industrial communication network	A distributed communication network for industrial environments based on open standards (EN 50235, EN50254, and EN50170, and others) that allows the exchange of data between devices from different manufacturers.
output filtering	The amount that it takes an output channel to send change-of-state information to an actuator after the output module has received updated data from the NIM.

output polarity	An output channel's polarity determines when the output module turns its field actuator on and when it turns the actuator off. If the polarity is *normal*, an output channel will turn its actuator on when the master controller sends it a 1. If the polarity is *reverse*, an output channel will turn its actuator on when the master controller sends it a 0.
output response time	The time it takes for an output module to take an output signal from the island bus and send it to its field actuator.

P

parameterize	To supply the required value for an attribute of a device at run-time.
PDM	*power distribution module.* A module that distributes either AC or DC field power to a cluster of I/O modules directly to its right on the island bus. A PDM delivers field power separately to the input modules and the output modules. It is important that all the I/O clustered directly to the right of a PDM be in the same voltage group—either 24 VDC, 115 VAC, or 230 VAC.
PDO	*process data object.* In CAN-based networks, PDOs are transmitted as unconfirmed broadcast messages or sent from a producer device to a consumer device. The transmit PDO from the producer device has a specific identifier that corresponds to the receive PDO of the consumer devices.
PE	*protective earth.* A return line across the bus for fault currents generated at a sensor or actuator device in the control system.
peer-to-peer communications	In peer-to-peer communications, there is no master/slave or client/server relationship. Messages are exchanged between entities of comparable or equivalent levels of functionality, without having to go through a third party (like a master device).
PLC	*programmable logic controller.* The PLC is the brain of an industrial manufacturing process. It automates a process as opposed to relay control systems. PLCs are computers suited to survive the harsh conditions of the industrial environment.

preferred module	An I/O module that functions as an auto-addressable node on an Advantys STB island but is not in the same form factor as a standard Advantys STB I/O module and therefore does not fit in an I/O base. A preferred device connects to the island bus via an STB XBE 1000 EOS module and a length of STB XCA 100*x* bus extension cable. It can be extended to another preferred module or back into a standard island segment. If it is the last device on the island, it must be terminated with a 120 Ω terminator.
premium network interface	An Advantys STB network interface module designed at a relatively high cost to support high module densities, high transport data capacity (e.g., for web servers), and more diagnostics on the island bus.
prioritization	Prioritization is an optional feature that allows you to selectively identify digital input modules to be scanned more frequently during the NIM's logic scan of the island bus.
process I/O	An Advantys STB I/O module designed for operation at extended temperature ranges in conformance with IEC type 2 thresholds. Modules of this type often feature high levels of on-board diagnostics, high resolution, user-configurable parameter options, and higher levels of agency approval.
process image	A part of the NIM firmware that serves as a real-time data area for the data exchange process. The process image includes an input buffer that contains current data and status information from the island bus and an output buffer that contains the current outputs for the island bus, from the fieldbus master.
producer/ consumer model	In networks that observe the producer/consumer model, data packets are identified according to their data content rather than by their physical location. All nodes "listen" on the network and consume those data packets that have appropriate identifiers.
Profibus DP	*Profibus Decentralized Peripheral.* An open bus system that uses an electrical network based on a shielded two-wire line or an optical network based on a fiber-optic cable. DP transmission allows for high-speed, cyclic exchange of data between the controller CPU and the distributed I/O devices.

R

reflex action	The execution of a simple, logical command function configured locally at an island bus I/O module. Reflex actions are executed by island bus modules on data from various island locations, like input and output modules or the NIM. Examples of reflex actions include compare and copy operations.

repeater	An interconnection device that extends the permissible length of a bus.
reverse polarity protection	Use of a diode in a circuit to protect against damage and unintended operation in the event that the polarity of the applied power is accidentally reversed.
rms	*root mean square.* The effective value of an alternating current, corresponding to the DC value that produces the same heating effect. The rms value is computed as the square root of the average of the squares of the instantaneous amplitude for one complete cycle. For a sine wave, the rms value is 0.707 times the peak value.
role name	A customer-driven, unique logical personal identifier for an Ethernet Modbus TCP/ IP NIM. A role name is created either as a combination of a numeric rotary switch setting and the STB NIP 2212 part number or by modifying text on the Configure Role Name web page. After the STB NIP 2212 is configured with a valid role name, the DHCP server will use it to identify the island at power up.
RTD	*resistive temperature detect.* An RTD device is a temperature transducer composed of conductive wire elements typically made of platinum, nickel, copper, or nickel-iron. An RTD device provides a variable resistance across a specified temperature range.
Rx	*reception.* For example, in a CAN-based network, a PDO is described as an RxPDO of the device that receives it.

S

SAP	*service access point.* The point at which the services of one communications layer, as defined by the ISO OSI reference model, is made available to the next layer.
SCADA	*supervisory control and data acquisition.* Typically accomplished in industrial settings by means of microcomputers.
SDO	*service data object.* In CAN-based networks, SDO messages are used by the fieldbus master to access (read/write) the object directories of network nodes.
segment	A group of interconnected I/O and power modules on an island bus. An island must have at least one segment and may have as many as seven segments. The first (leftmost) module in a segment needs to provide logic power and island bus communications to the I/O modules on its right. In the primary segment, that function is filled by a NIM. In an extension segment, that function is filled by an STB XBE 1200 BOS module.

SELV	*safety extra low voltage.* A secondary circuit designed and protected so that the voltage between any two accessible parts (or between one accessible part and the PE terminal for Class 1 equipment) does not exceed a specified value under normal conditions or under single-fault conditions.
SIM	*subscriber identification module.* Originally intended for authenticating users of mobile communications, SIMs now have multiple applications. In Advantys STB, configuration data created or modified with the Advantys configuration software can be stored on a SIM and then written to the NIM's Flash memory.
single-ended inputs	An analog input design technique whereby a wire from each signal source is connected to the data acquisition interface, and the difference between the signal and ground is measured. Two conditions are imperative to the success of this design technique—the signal source must be grounded, and the signal ground and data acquisition interface ground (the PDM lead) must have the same potential.
sink load	An output that, when turned on, receives DC current from its load.
size 1 base	A mounting device, designed to seat an STB module, hang it on a DIN rail, and connect it to the island bus. It is 13.9 mm wide and 128.25 mm high.
size 2 base	A mounting device, designed to seat an STB module, hang it on a DIN rail, and connect it to the island bus. It is 18.4 mm wide and 128.25 mm high.
size 3 base	A mounting device, designed to seat an STB module, hang it on a DIN rail, and connect it to the island bus. It is 28.1 mm wide and 128.25 mm high.
slice I/O	An I/O module design that combines a small number of channels (usually between two and six) in a small package. The idea is to allow a system developer to purchase just the right amount of I/O and to be able to distribute it around the machine in an efficient, mechatronics way.
SM_MPS	*state management_message periodic services.* The applications and network management services used for process control, data exchange, error reporting, and device status notification on a Fipio network.
SNMP	*simple network management protocol.* The UDP/IP standard protocol used to manage nodes on an IP network.
snubber	A circuit generally used to suppress inductive loads—it consists of a resistor in series with a capacitor (in the case of an RC snubber) and/or a metal-oxide varistor placed across the AC load.
source load	A load with a current directed into its input; must be driven by a current source.

standard network interface	An Advantys STB network interface module designed at moderate cost to support the kind of configuration capabilities and throughput capacity suitable for most standard applications on the island bus.
STD_P	*standard profile.* On a Fipio network, a standard profile is a fixed set of configuration and operating parameters for an agent device, based on the number of modules that the device contains and the device's total data length. Three types of standard profiles are available—Fipio reduced device profile (FRD_P), Fipio standard device profile (FSD_P), and the Fipio extended device profile (FED_P).
stepper motor	A specialized DC motor that allows discrete positioning without feedback.
subnet	A part of a network that shares a network address with the other parts of a network. A subnet may be physically and/or logically independent of the rest of the network. A part of an internet address called a subnet number, which is ignored in IP routing, distinguishes the subnet.
surge suppression	The process of absorbing and clipping voltage transients on an incoming AC line or control circuit. Metal-oxide varistors and specially designed RC networks are frequently used as surge suppression mechanisms.

T

TC	*thermocouple.* A TC device is a bimetallic temperature transducer that provides a temperature value by measuring the voltage differential caused by joining together two different metals at different temperatures.
TCP	*transmission control protocol.* A connection-oriented transport layer protocol that provides reliable full-duplex data transmission. TCP is part of the TCP/IP suite of protocols.
telegram	A data packet used in serial communication.
TFE	*transparent factory Ethernet.* Schneider Electric's open automation framework based on TCP/IP.
Tx	*transmission.* For example, in a CAN-based network, a PDO is described as a TxPDO of the device that transmits it.

U

UDP	*user datagram protocol.* A connectionless mode protocol in which messages are delivered in a datagram to a destination computer. The UDP protocol is typically bundled with the Internet Protocol (UPD/IP).

V

varistor	A two-electrode semiconductor device with a voltage-dependant nonlinear resistance that drops markedly as the applied voltage is increased. It is used to suppress transient voltage surges.
voltage group	A grouping of Advantys STB I/O modules, all with the same voltage requirement, installed directly to the right of the appropriate power distribution module (PDM) and separated from modules with different voltage requirements. Never mix modules with different voltage requirements in the same voltage group.

W

watchdog timer	A timer that monitors a cyclical process and is cleared at the conclusion of each cycle. If the watchdog runs past its programmed time period, it generates a fault.

Index