Original instructions

PLUTO Safety-PLC

Description of Function Blocks

Library: Func06.fps

Table of contents:

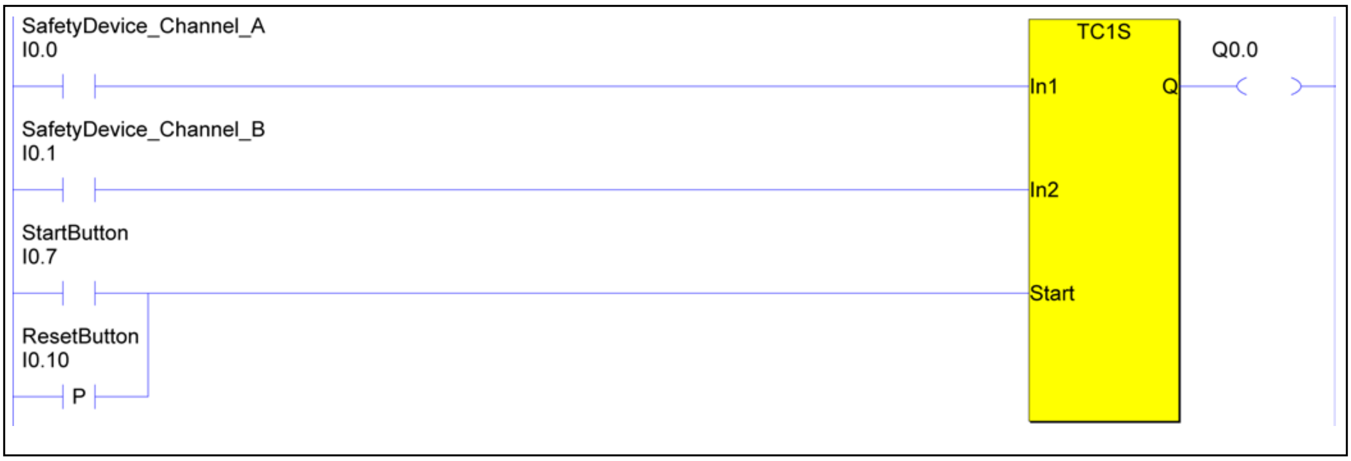
Basic

TC1S

Two-channel function with input for start

- In1 and In2 are primary inputs where safety devices can be connected.
- Start is a start condition which can be used for monitoring or reset.
  In order to get monitored reset the input to Start can be set as pulse/edge.

Description:
The function block works as a conventional safety relay with redundant and monitored inputs (In1, In2).

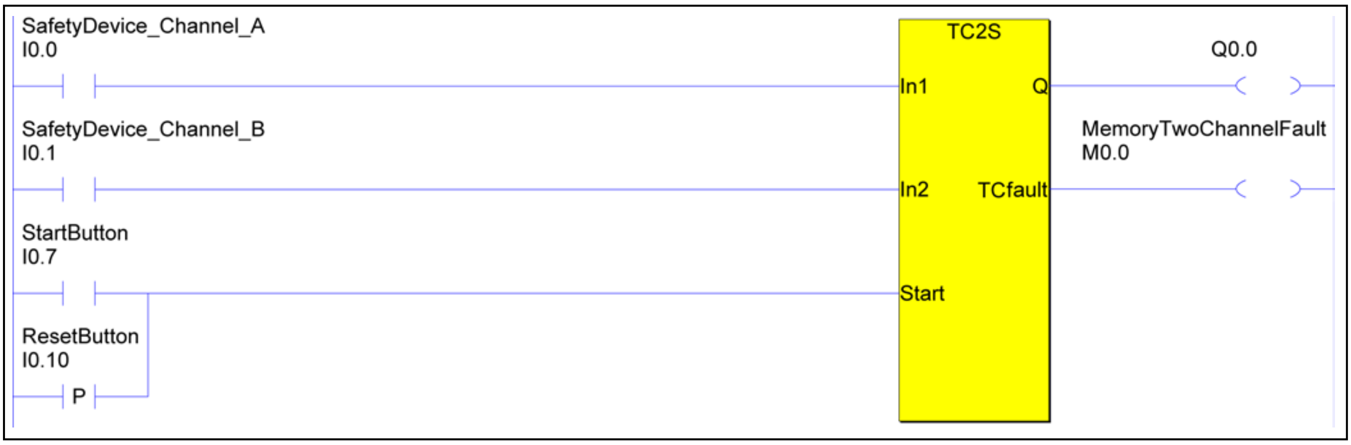


TC2S

Two-channel function with input for start

- In1 and In2 are primary inputs where safety devices can be connected.
- Start is a start condition which can be used for monitoring or reset.
  In order to get monitored reset the input to Start can be set as pulse/edge.
- TCfault is on at two channel fault, i.e. when the function block is on, and only one of In1 or In2 switches off and on again.

<u>Description:</u>
The function bock works as a conventional safety relay with redundant and monitored inputs (In1, In2).

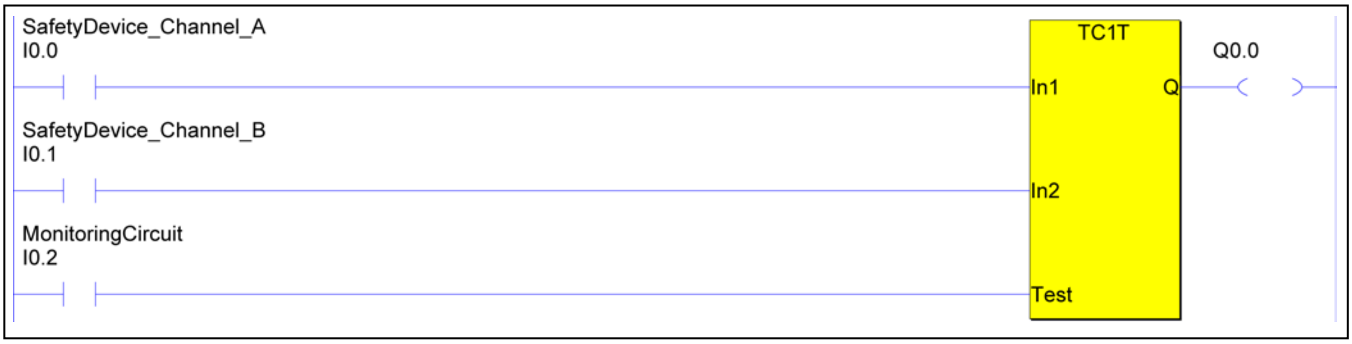


TC1T

Two-channel function with test input

- In1 and In2 are primary inputs where safety devices can be connected.
- Test is an input condition which must be true at the start up moment and can be used for monitoring of external components. Test must switch on before In1 and In2, i.e. the function block cannot be initiated by Test.

Description:
The function block works as a conventional safety relay with redundant and monitored inputs (In1, In2).

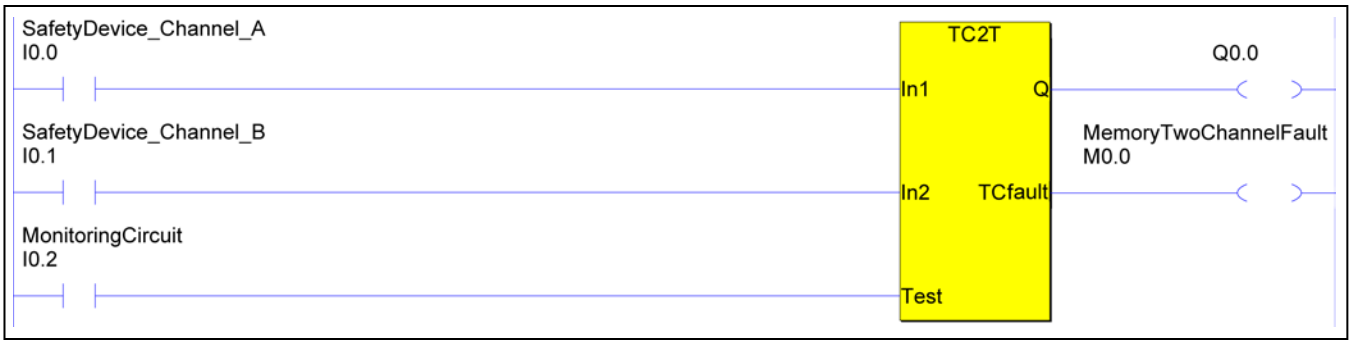


TC2T

Two-channel function with test input

- In1 and In2 are primary inputs where safety devices can be connected.
- Test is an input condition which must be true at the start up moment and can be used for monitoring of external components. Test must switch on before In1 and In2, i.e. the function block cannot be initiated by Test.
- TCfault is on at two channel fault, i.e. when the function block is on, and only one of In1 or In2 switches off and on again.

Description:
The function block works as a conventional safety relay with redundant and monitored inputs (In1, In2).



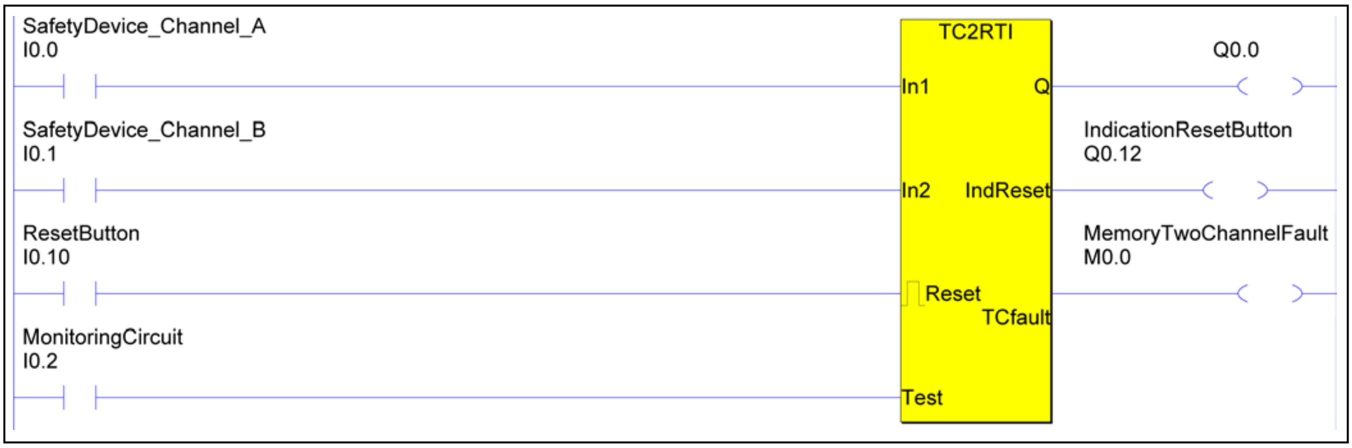
TC2RTI

Two-channel function with test and reset inputs.

- In1 and In2 are primary inputs where safety devices can be connected.
- Test is an input condition which must be true at the start up moment and can be used for monitoring of external components. Test must switch on before In1 and In2, i.e. the function block cannot be initiated by Test.
- Reset is a monitored reset input which must switch on-off (reset on negative edge) after all other inputs, in order to switch on the output of the function block.

- IndReset is an output for indication which is ON when the function block is switched off and In1, In2 or Test is off, and flashes when the function block is ready for activation with reset.
- TCfault is on at two channel fault, i.e. when the function block is on, and only one of In1 or In2 switches off and on again.

Description:
The function block works as a conventional safety relay with redundant and monitored inputs (In1, In2).

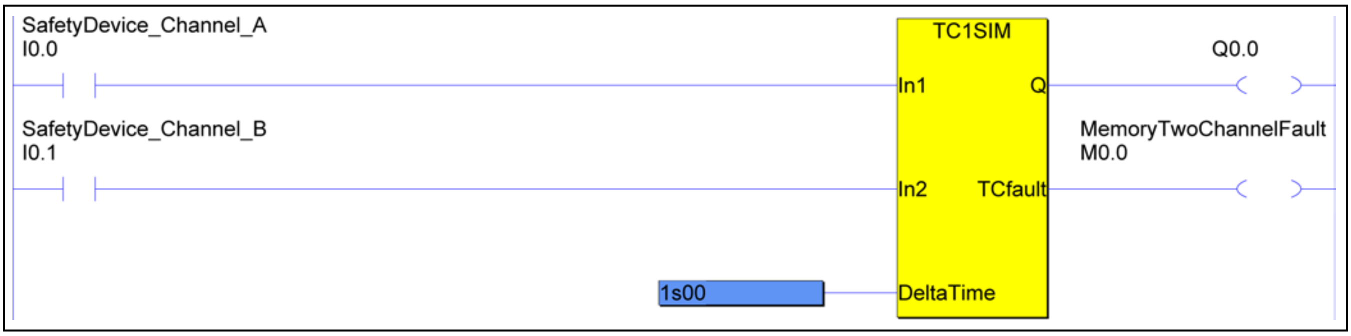


TC1SIM

Two-channel function with monitoring of simultaneous operation and debounce filtering.

- In1 and In2 are primary inputs where safety devices can be connected.
- DeltaTime is an input for time constant specifying the max time difference between switching on In1 and In2.
- TCfault is on at two channel fault, i.e. when the function block is on, and only one of In1 or In2 switches off and on again.

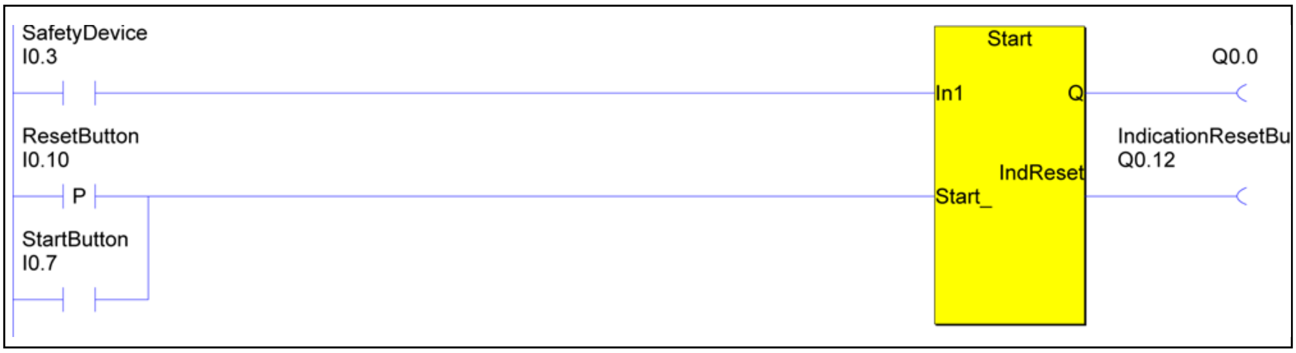
Description:
The function block works as a conventional safety relay with redundant and monitored inputs (In1, In2) which must switch on simultaneously within the time “DeltaTime”.



Start

Single-channel function with start input

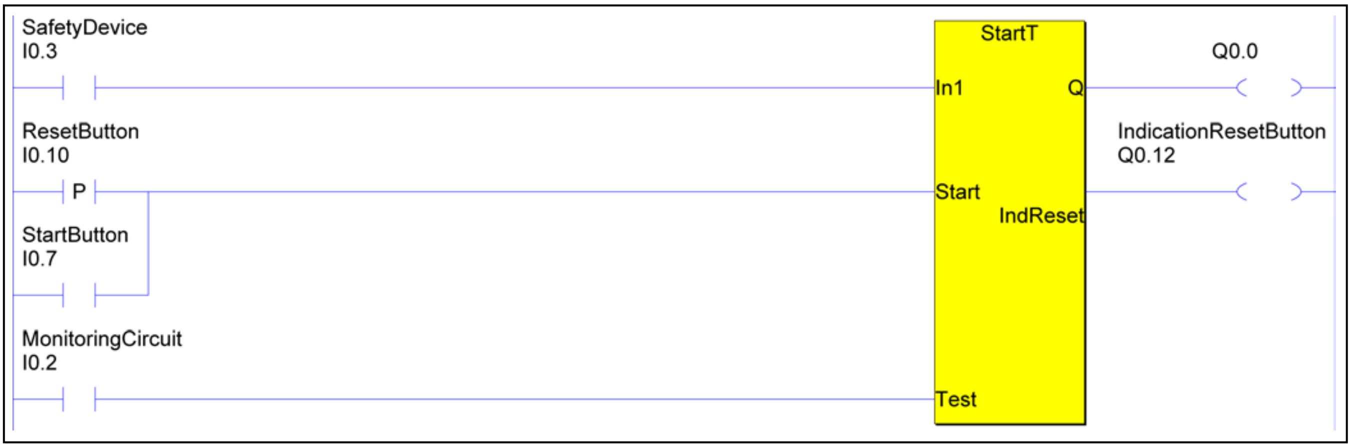
- In1 is primary input to which the output of a safety devices, output of other function blocks or memories can be connected.
- Start is a start condition and can be used as monitor or reset input.
  In order to get monitored reset, the input to Start can be set as pulse/edge.
- IndReset is an output for indication which is on when the function block is switched off and In1 is off, and flashes when the function block is ready for activation with Start.



StartT

Single-channel function with start and test input

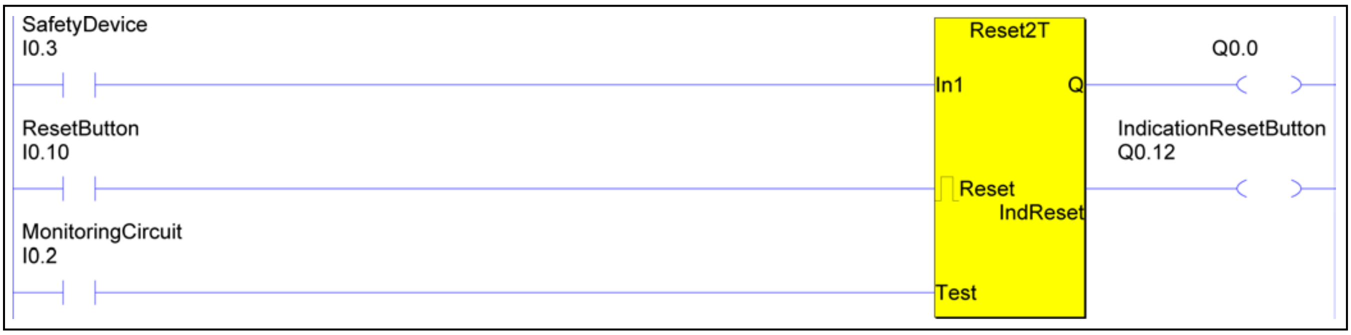
- In1 is primary input to which the output of a safety devices, output of other function blocks or memories can be connected.
- Start is a start condition and can be used as reset input.
  In order to get monitored reset, the input to Start can be set as pulse/edge.
  Start can be bridged in order to get automatic reset by connection of a line directly from "common".
- Test is an input condition which must be true at the start up moment and can be used for monitoring of external components. Test must be on before In1 or Start (if Start is bridged to "common", Test must be on before In1), i.e. the function block cannot be initiated by Test.
  If input Test is not needed it can be bridged by connection of a line direct from the right "common".
- IndReset is an output for indication which is on when the function block is switched off and In1 is off, and flashes when the function block is ready for activation with Start.



Reset2T

Single-channel function with reset and test input

- In1 is primary input to which the output of safety devices, output of other function blocks or memories can be connected.
- Test is an input condition which must be true at the start up moment and can be used for monitoring of external components. Test must switch on before Reset, i.e. the function block cannot be initiated by Test.
- Reset is a monitored reset input which must be activated on-off (reset on negative edge) after all other inputs in order to switch on the output of the function block.
- IndReset is an output for indication which is on when the function block is switched off and In1 and Test is off, and flashes when the function block is ready for activation with Start.

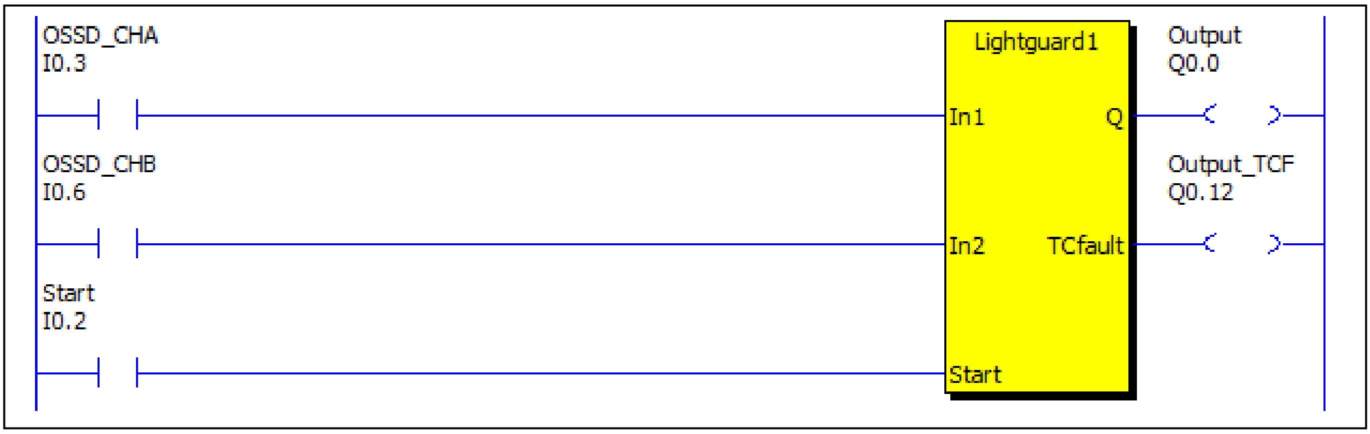


Lightguard1

Two-channel function for optical safety device with OSSD outputs with test pulses

- In1 and	
- In2	Safety inputs where optical safety device OSSD-outputs can be connected.
- Start	Start condition which can be used for monitoring, restart or reset. In order to get monitored reset the function block can be further connected to a reset block.
- TCfault	Output for two channel fault, i.e. when the function block is on and just one of In1 and In2 switches off and on again.

Description:
The function block is intended for optical safety devices with two monitored output channels. Short circuit between output channels or short circuit to power supply must be monitored by the optical safety device itself.

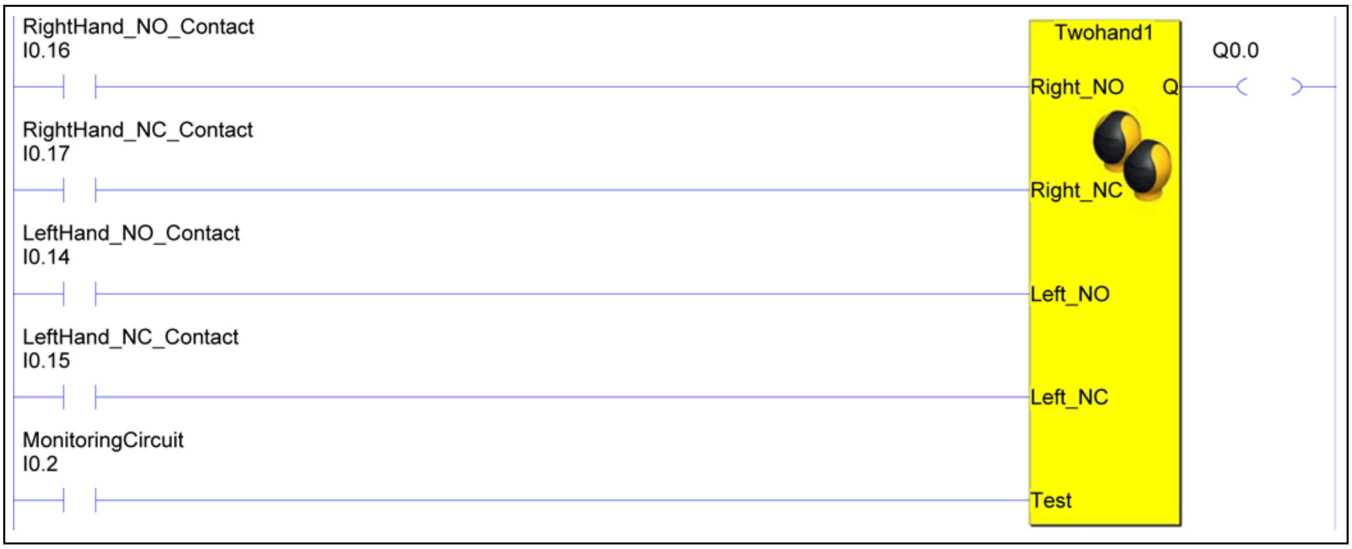


Twohand1

Two-hand control for actuators with NO/NC + NO/NC contacts.

- Right_NO is Right hand NO-contact
- Right_NC is Right hand NC-contact etc.
- Test is an input condition that must be on before any of the other inputs changes from the initial position and can be used for monitoring of external components.

Function:
In initial position shall:
Right_NO be off,
Right_NC be on,
Left_NO be off and
Left_NC be on
To start the output (Q), these four inputs must change state in 0.5 seconds and remain in that position. After stop, all inputs must come to the initial position to enable restart.



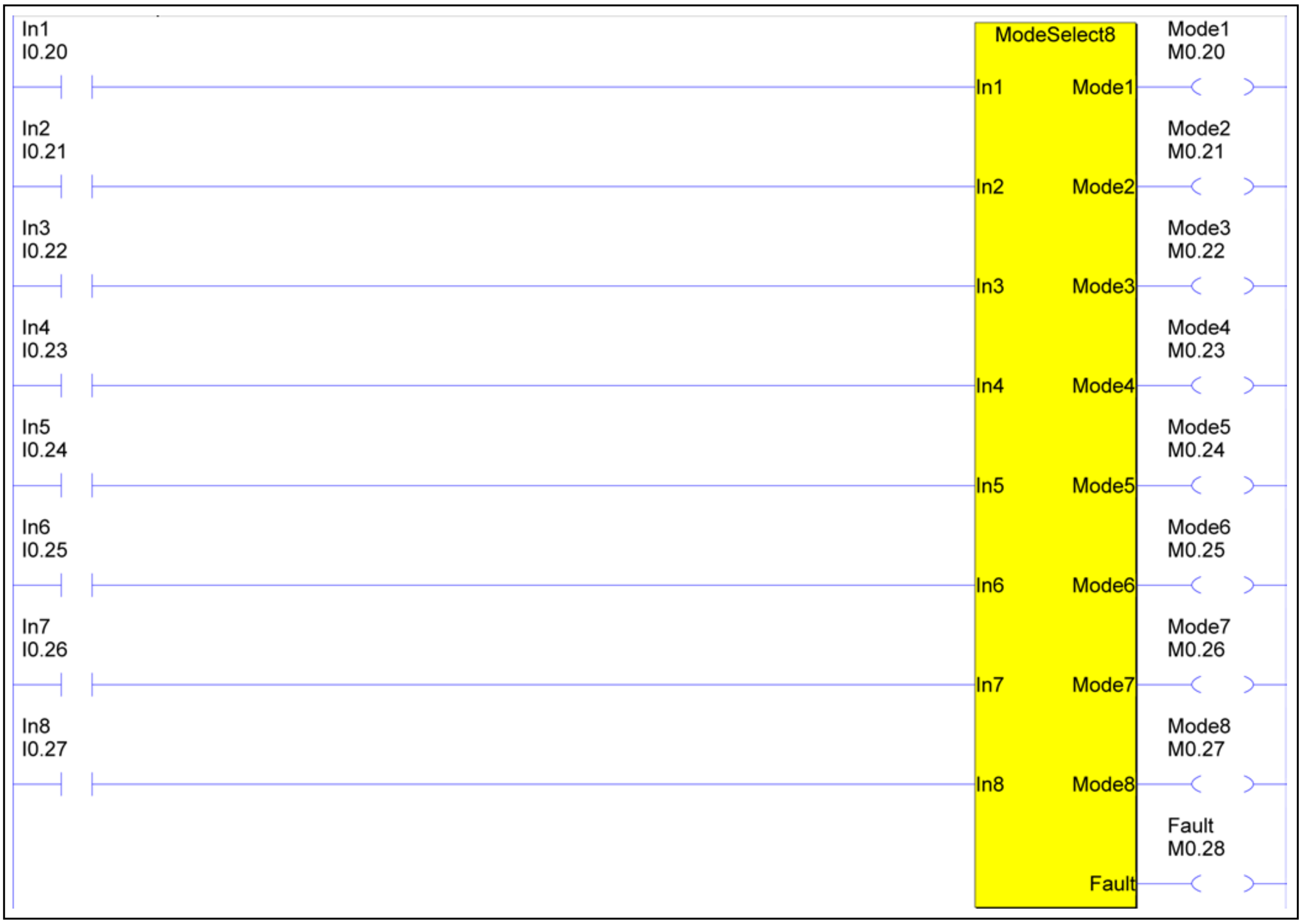
ModeSelect8

Monitoring of selector switch with 8 positions

- In1..In8 are inputs from the selector switch.
- Mode1..Mode8 are outputs which are set corresponding to the inputs In1..In8.
  At faulty behaviour in the selector switch none of Mode1..Mode8 is set.
- Fault is set if:
  more than one of In1..In8 are set or
  by overlap in the selector switch, so not all of In1..In8 are off before a new input is set.

Description:

The block is intended to monitor a selector switch. The requirement is that only one position may be activated (ON) and by changing position all inputs must switch off before a new input is set.



Advanced

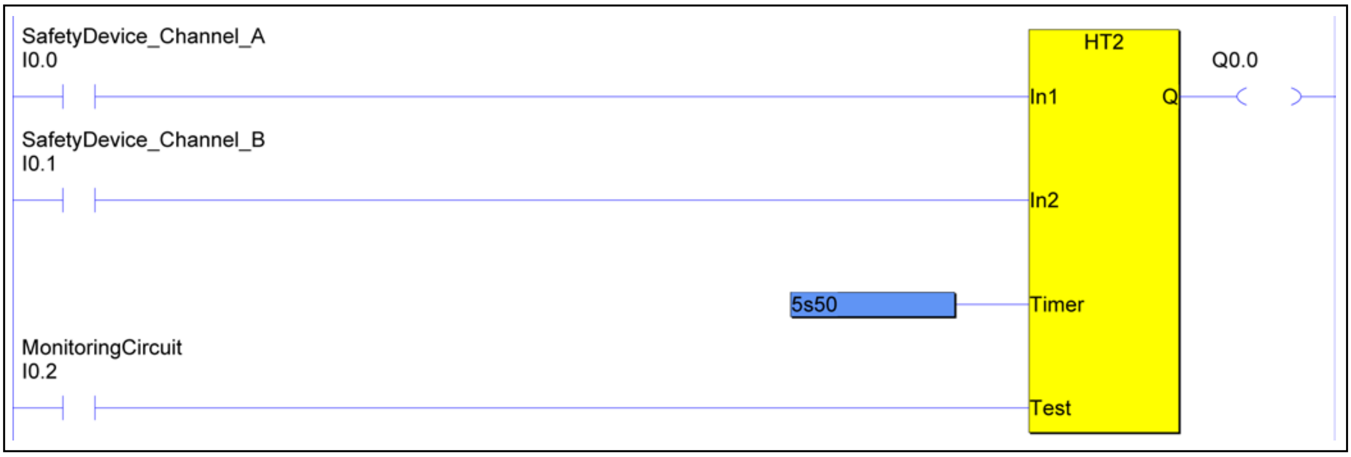
HT2

Two-channel function with time limitation.

- In1 and In2 are primary inputs where safety devices can be connected.
- Test is an input condition which must be true at the start up moment and can be used for monitoring of external components. Test must switch on before In1 and In2, i.e. the function block cannot be initiated by Test.
- Timer is an input for a time constant with the value for the desired time limit.

Description:
When both inputs In1 and In2 switches 0 to1, the function block switches on and remains in that state until the time is elapsed or one of In1 or In2 goes off. The timer starts when both In1 and In2 are on.

The LED indicators in the front panel for In1 and In2 flashes at the input that does not switch on if the other does and also if one of them opens and closes and the other remains on.



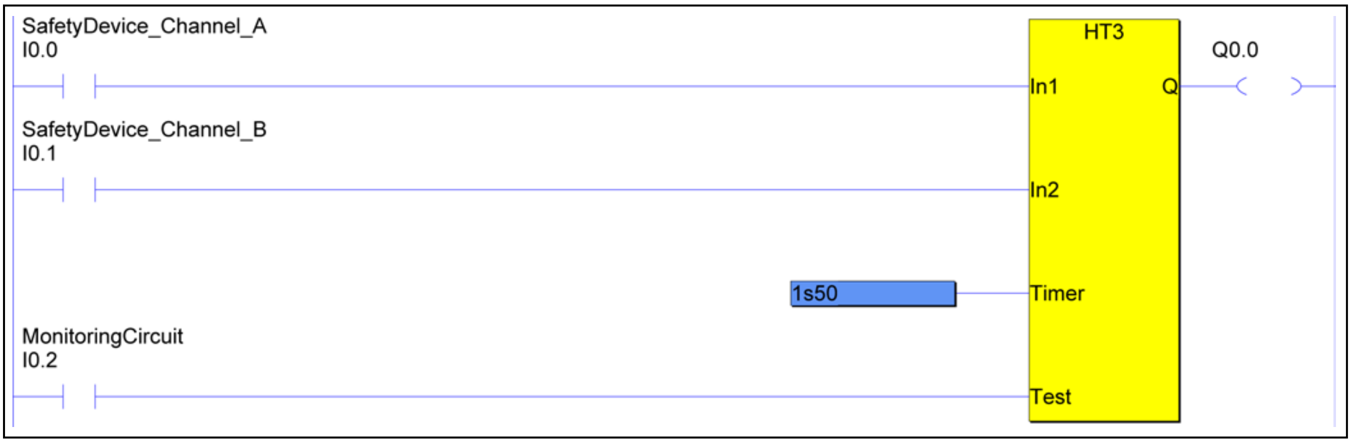
HT3

Two-channel function with time limitation.

- In1 and In2 are primary inputs where safety devices can be connected.
- Test is an input condition which must be true at the start up moment and can be used for monitoring of external components. Test must switch on before In1 and In2, i.e. the function block cannot be initiated by Test.
- Timer is an input for a time constant with the value for the desired time limit.

Description:
When both inputs In1 and In2 switches 0 to1, the function block switches on and remains in that state until the time is elapsed or one of In1 or In2 goes off.
The timer starts when one of In1 and In2 is on. (Different from HT2)

The LED indicators in the front panel for In1 and In2 flashes at the input that does not switch on if the other does and also if one of them opens and closes and the other remains on.

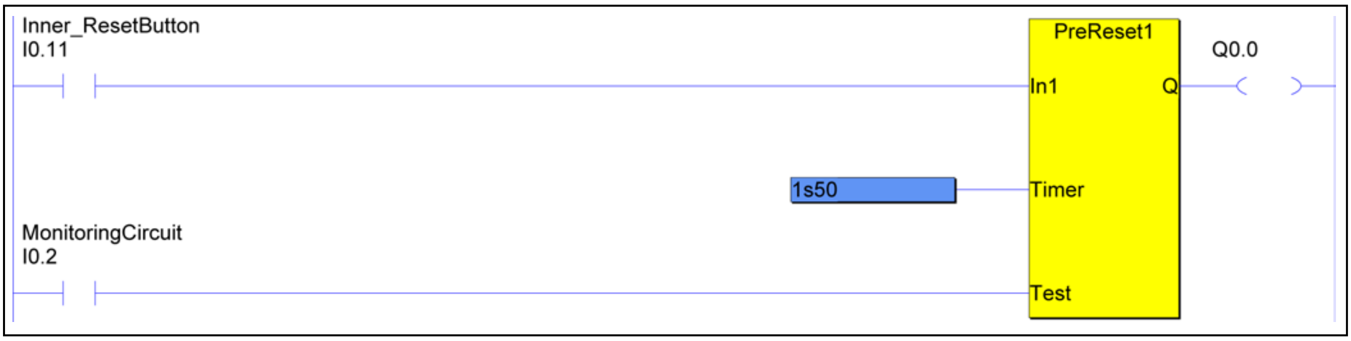


PreReset1

Single-channel pulse function (intended for inner reset)

- In1 is an input to which for example a push button for prereset can be connected.
- Test is an input condition which must be true at the start up moment and can be used for monitoring of external components. Test must switch on before In1, i.e. the function block cannot be initiated by Test.
- Timer is an input for a time constant with the value for the desired pulse time.

Description:
When In1 switches on, the output of the function block switches on and remains on until the time is elapsed, even if In1 goes low before.

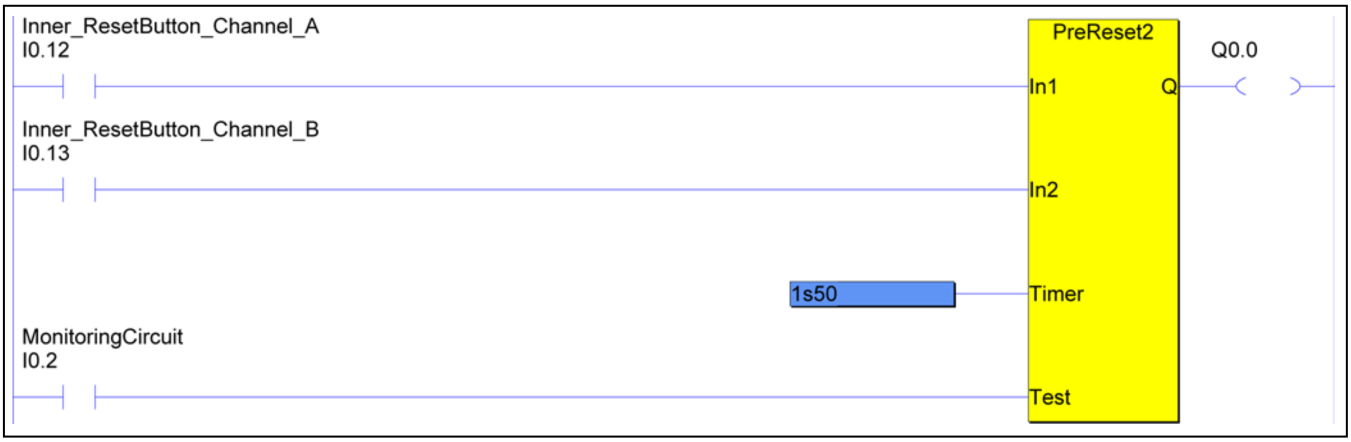


PreReset2

Two-channel pulse function (intended for inner reset)

- In1 and In2 are inputs to which for example a two-channel push button for prereset can be connected.
- Test is an input condition which must be true at the start up moment and can be used for monitoring of external components. Test must switch on before In1 and In2, i.e. the function block cannot be initiated by Test.
- Timer is an input for a time constant with the value for the desired pulse time.

Description:
When In1 and In2 switches on, the output of the function block switches on and remains on until the time is elapsed, even if In1 and In2 goes low before.
The timing starts when both In1 and In2 are switched on.

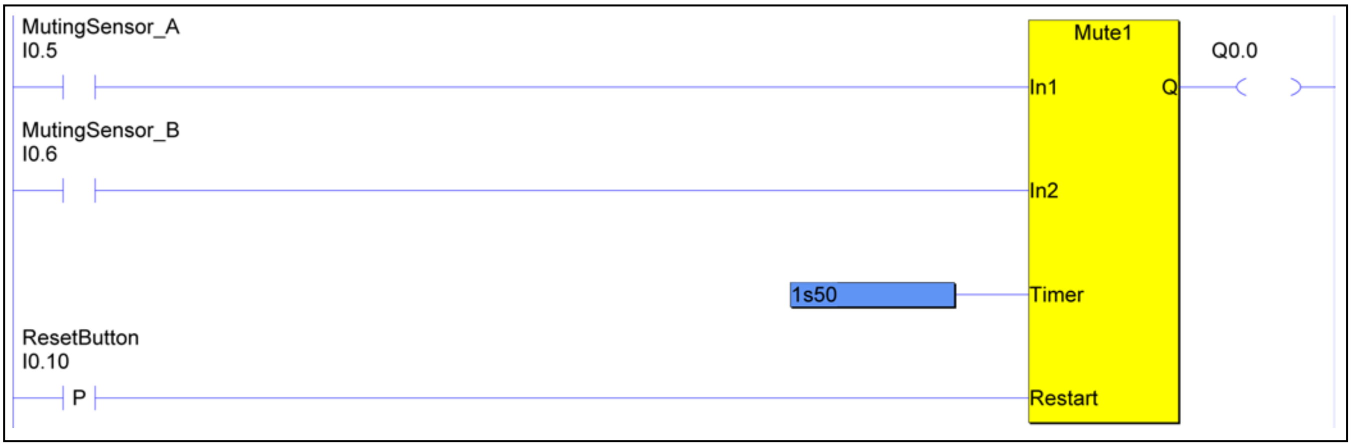


Mute1

Muting function with two-channel input.

- In1 and In2 are inputs for connection of for example two muting sensors.
- Restart is an input which can be used for restart button.
- Timer is an input for a time constant with the value for the maximal muting time.

Description:
When both inputs In1 and In2 switches 0 to1, the function block switches on and remains in that state until the time is elapsed or one of In1 or In2 goes off. The timer starts when both In1 and In2 are on.
By activation of the Restart input, the muting can be restarted if it has switched off. The only condition for restart is that In1 and In2 are on. Restart is activated on positive edge of the input signal.

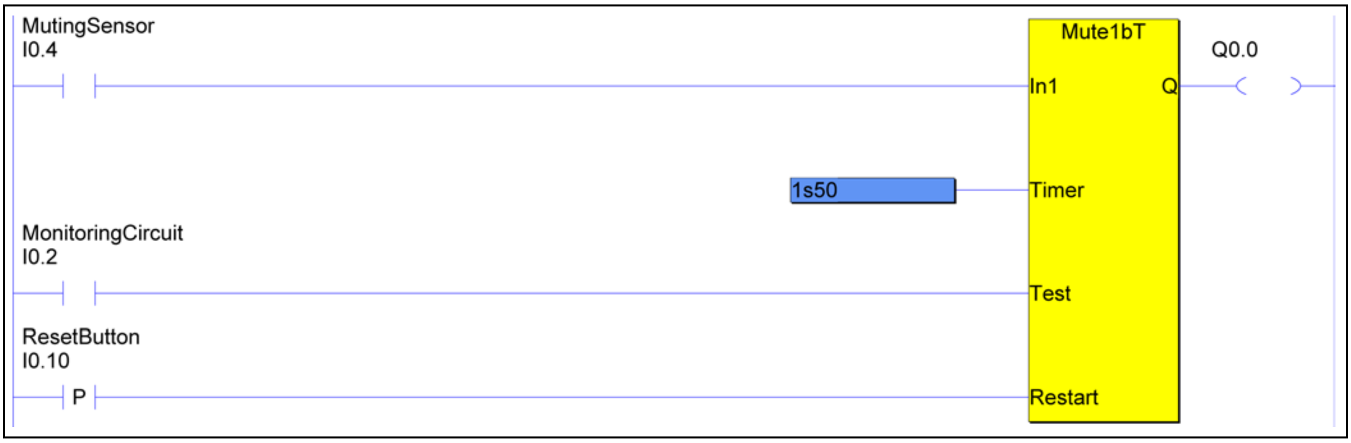


Mute1bT

Muting function with single-channel input.

- In1 is inputs for connection of muting sensor, output from other block etc.
- Test is an input condition which must be true at the start up moment and can be used for monitoring of external components. Test must switch on before In1 and Restart, i.e. the function block cannot be initiated by Test.
- Restart is an input which can be used for restart button.
- Timer is an input for a time constant with the value for the maximal muting time.

Description:
When input In1 switches 0 to1, the function block switches on and remains in this state until the time is elapsed or In1 goes off.
By activation of the Restart input, the muting can be restarted if the time is elapsed without switching off In1 first. Restart is activated on positive edge of the input signal.

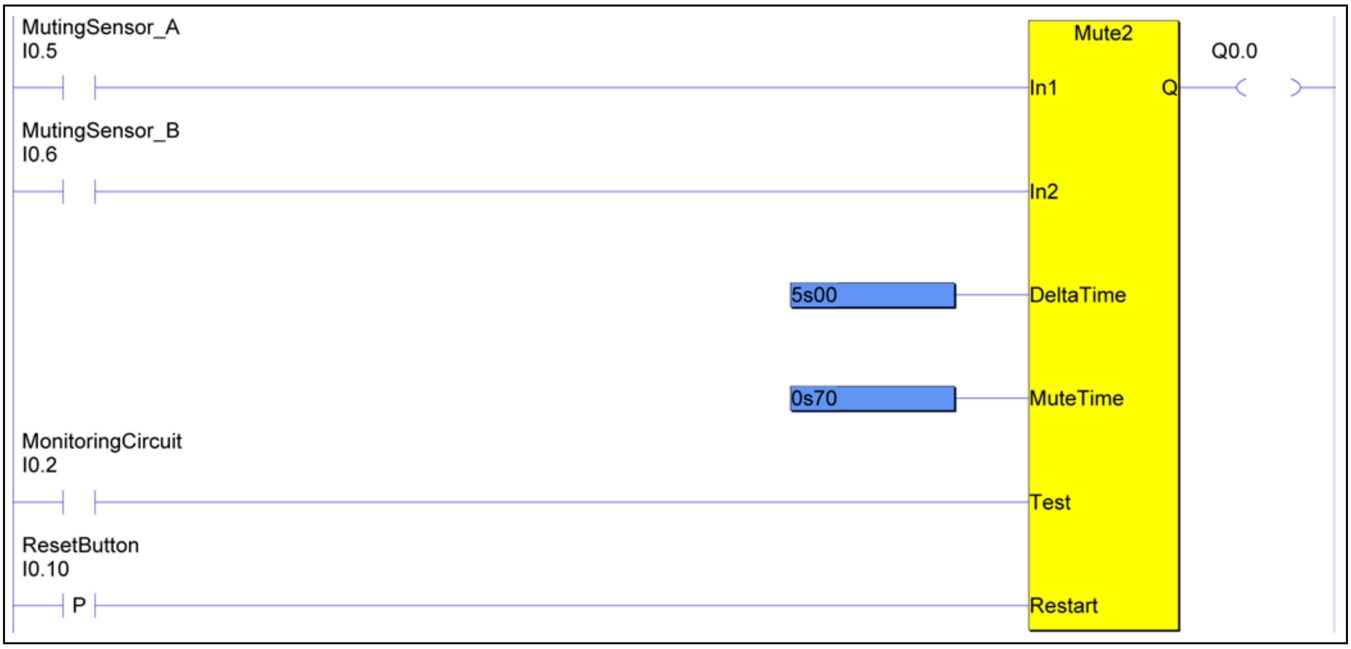


Mute2

Muting function with two-channel input, time limitation and timer for simultaneous activation of inputs.

- In1 and In2 are inputs for connection of for example two muting sensors.
- Test is an input condition which must be true at the start up moment and can be used for monitoring of external components. Test must switch on before In1, In2 and Restart, i.e. the function block cannot be initiated by Test.
- Restart is an input which can be used for restart button.
- DeltaTime is an input for a time constant with the value for the maximal difference in simultaneity between In1 and In2.
- MuteTime is an input for a time constant with the value for the maximal muting time.

Description:
When both inputs In1 and In2 switches 0 to1, the function block switches on and remains in that state until Mute Time is elapsed or one of In1 or In2 goes off. The timer starts when both In1 and In2 is on.
By activation of the Restart input the muting can be restarted if it has switched off. The only condition for restart is that In1, In2 and Test are on. Restart is activated on positive edge of the input signal.
If the time difference between switching on In1 and In2 exceeds the value of DeltaTime the function block does not switch on.



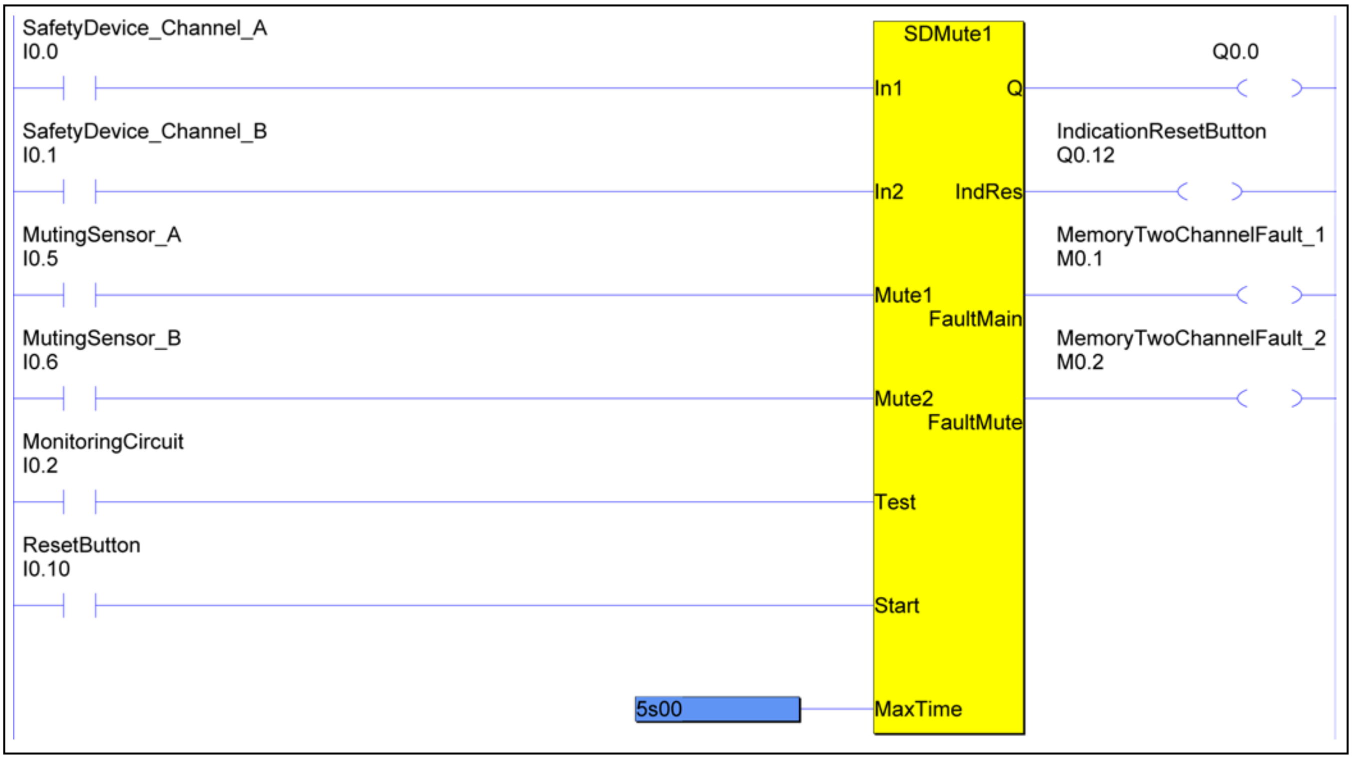
SDMute1

Safety function with two-channel input and integrated two-channel muting function

- In1 and In2 are primary input channels where safety devices can be connected.
- Mute1 and Mute2 are inputs for connection of for example two muting sensors.
- Test is an input condition which must be true at the start up moment and can be used for monitoring of external components. Test must switch on before the start attempt, i.e. the function block cannot be initiated by Test.
- Start is a start condition and can be used as reset input.
  In order to get monitored reset, the input to Start can be set as pulse/edge.
- MaxTime is an input for a time constant with the value for the maximal muting time.

- Output IndReset is on when the function block is off and flashes when it is ready for activation with Start.
- Output FaultMain is on at two-channel fault at In1/In2.
- Output FaultMute is on at two-channel fault at Mute1/Mute2

Description:
The main function In1/In2 works as a conventional safety relay with redundant and monitored inputs.
The muting function Mute1/Mute2 works as a time limited muting function. When the muting function is activated the timing starts and the inputs to In1 and In2 are allowed to switch off. Before the time is elapsed In1 and In2 must switch on again, if not the output of the function block switches off.
Start is an input function for start and restart of the function block. Start can be made when the inputs (In1, In2) or (Mute1, Mute2) are active and can be made even after the time has elapsed or a two channel fault exists. The input to Start can be set as pulse/edge to achieve monitored reset.



SeqMute1

Sequential muting function during transport of material.

In- and output description:

LightGuard	Input for light beam, light curtain etc.
Mute_1	Input for muting sensor furthest from the light beam. (Activated sensor = Input ON)
Mute_2	Input for muting sensor closest to the light beam. (Activated sensor = Input ON)
Enable	Input for enabling of the muting session
MaxTime	Input for the maximum time in seconds between Mute_1 ON and LightGuard ON, must be between 3..300 seconds
Restart	Input to restart the sequence after a failure or a timeout

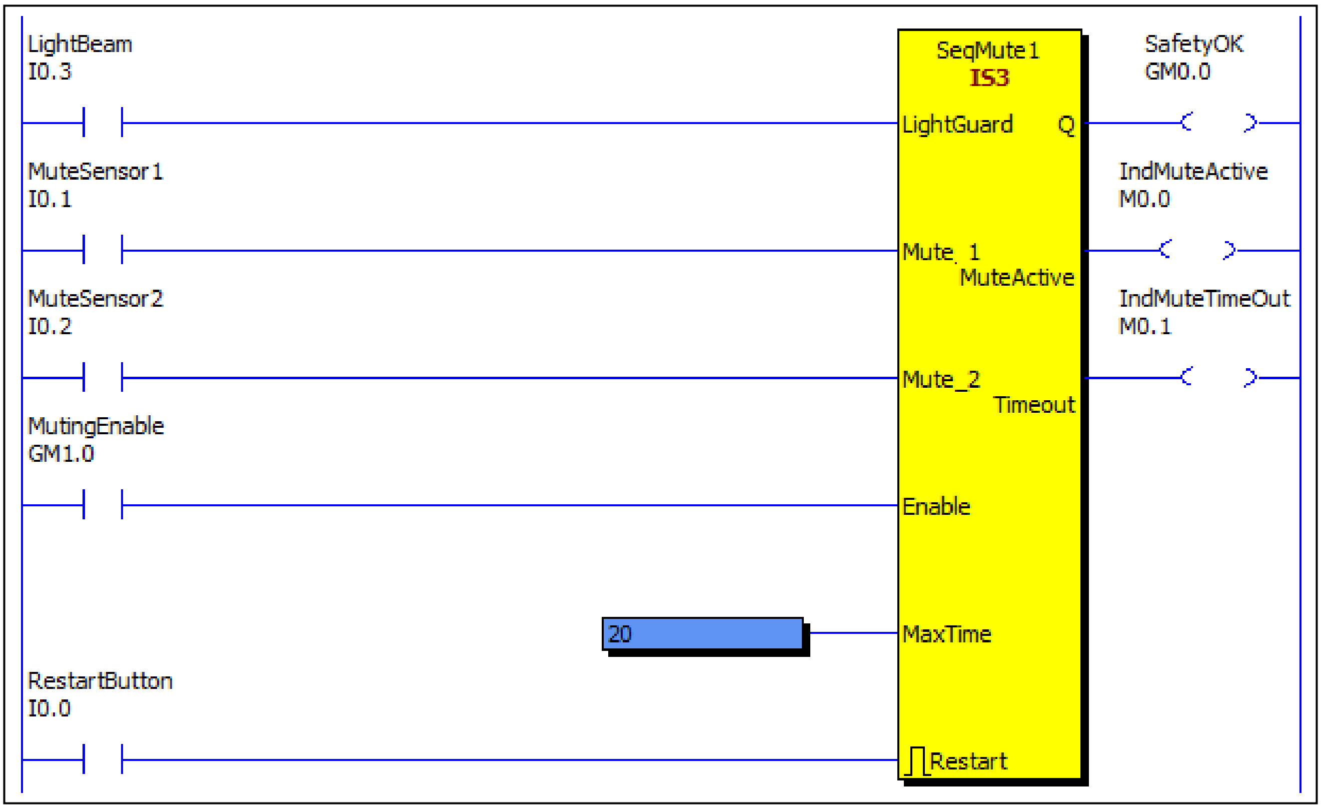
Q	Main output for safety ON, indicating LightGuard ON or muting activated
MuteActive	Output indicating that the light beam is muted.
Timeout	Output indicating that the maximum time for the muting sequence has been exceeded

Description of the function:

The function is made for muting of a light beam during transport of material.
Muting sensors shall be mounted so they are activated at different positions during the transport.
The requirements for initialization of the muting sequence are: Light beam ON (not broken), Mute_1 and Mute_2 OFF and Enable ON.
Muting is then initiated after activation of input Mute_1 and then Mute_2.

The complete muting sequence can be described as following:
- 0) Mute_1 OFF, Mute_2 OFF, LightGuard ON
- 1) Mute_1 ON
- 2) Mute_2 ON after a minimum of 0,1s
- 3) Wait 0,1s
- 4) Muting Active - LightGuard can be OFF/BROKEN
- 0) When LightGuard ON+1s

The Enable input must be ON for enabling the muting. Enable can be used for a control signal from the machine. For example to allow muting only when a feeder is running. If Enable is not used the input can be set continuously high.
Until the LightGuard is OFF the muting sequence can be reversed so for example if Mute_2 is activated it can be switched off and on again.
If the muting is interrupted by a failure, the sequence is initiated and the muting is deactivated.
After a failure or a timeout the muting can be restarted by activating the Restart input for a duration of between 50ms and 2s.

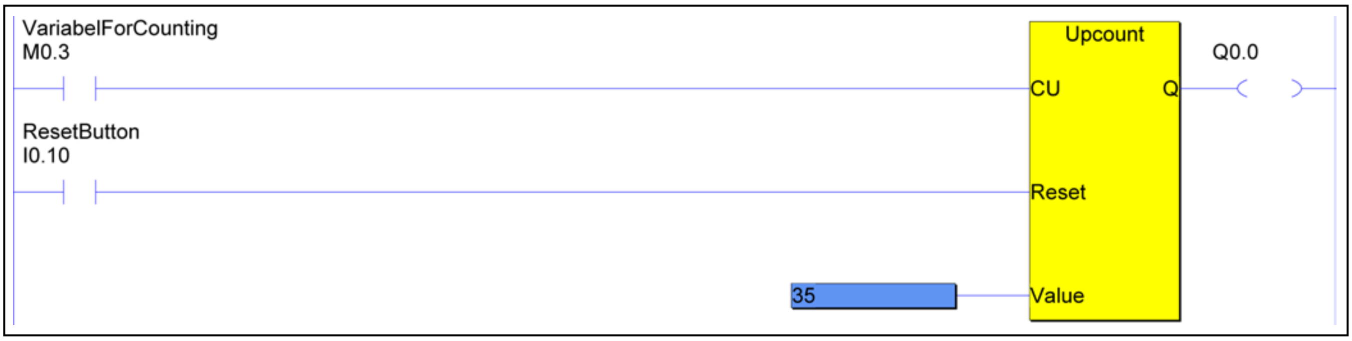

LightBeam
I0.3
MuteSensor1
I0.1
MuteSensor2
I0.2
MutingEnable
GM1.0
20
RestartButton
I0.0
SeqMute1
IS3
LightGuard
Q
Mute_1
MuteActive
Mute_2
Timeout
Enable
MaxTime
Restart
SafetyOK
GM0.0
IndMuteActive
M0.0
IndMuteTimeOut
M0.1

Upcount

Counter which counts up to preset value

- CU, counter input
- Reset, resets the counter to 0
- To Value shall a constant 0 - 32767 be connected, representing the preset value.

Function:
The counter is incremented on positive edge at the counter input CU.
When the counter reaches the preset value the output switches on and remains on until it is reset.
When the Reset input is on the counter is set to 0. A reset of the counter is also done at the first scan of the program step where the counter is programmed.

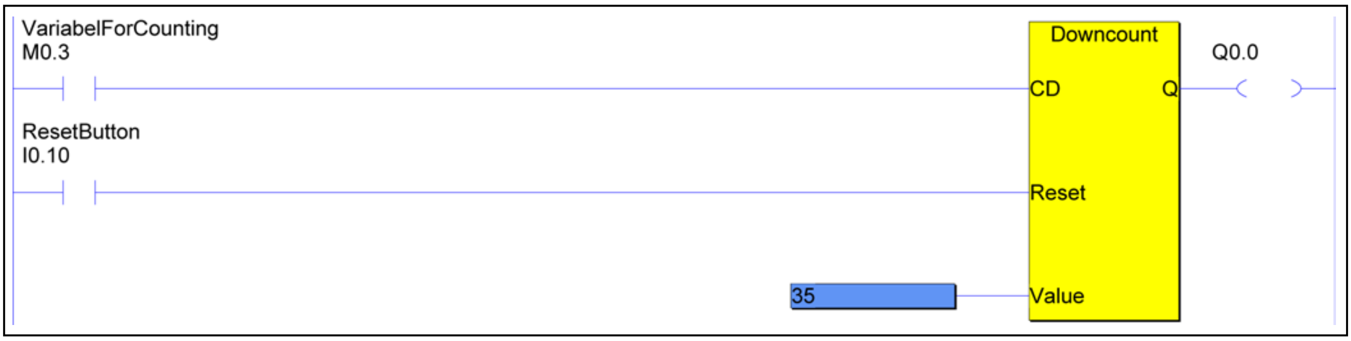


Downcount

Counter which counts down from preset value to 0.

- CD, counter input
- Reset sets the counter to preset value.
- To Value shall a constant 0 - 32767 be connected, representing the preset value.

Function:
The counter is decremented on positive edge at the counter input CD.
When the counter reaches 0 the output switches on and remains on until Reset is activated.
When the Reset input is on the counter is set to preset value. A reset of the counter is also done at the first scan of the program step where the counter is programmed.



LightCurtain1

Function block for light curtains with three main modes of operation:
Guard only, Single break, Double break mode.

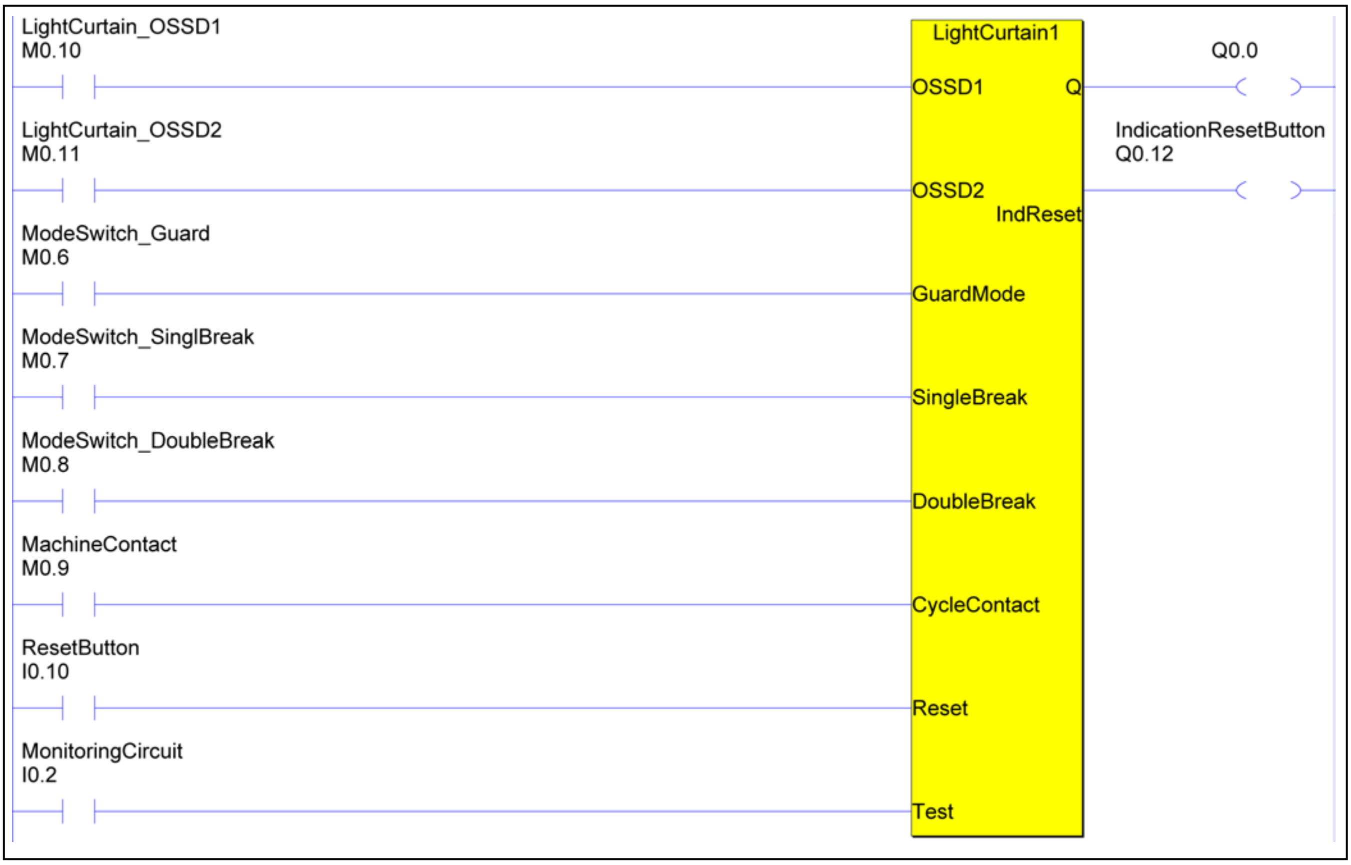
Function description:
In “Guard only” mode the input “GuardMode” shall be set. The output Q switches on after that input “Reset” is activated and the light curtain is green (OSSDs on).
The input “CycleContact” works as enable function in this mode. Reset is not required after operation of the “CycleContact”

“Single break” and “Double break” modes are intended for machines where the light curtain is used for initiation of a new cycle (typically for presses).
When the machine is prepared for a new cycle it shall set the input “CycleContact”. If “SingleBreak” is set output Q is not set before the light curtain has been interrupted/red once and in “DoubleBreak” mode it must be interrupted/red two times.
If “CycleContact” is set in Single and Double break mode for more than 30 seconds a timeout function will trip out in order to avoid unintended start. Reset is required to restore normal operation.

Reset must be activated after following cases:
- After change of mode, “Guard only”, “Single break”, “Double break”.
- If input Test is not activated when output “Q” shall switch on.
- When the light curtain is not activated within 30 sec. in single and double break modes.
- After intrusion in the light curtain when the output “Q” is set. (When the machine is stopped by the light curtain.)

When the light curtain is expecting interruption in Single and Double break modes Reset is not required.

Description of in- and outputs:
- OSSD1: Input connected to light curtain output
- OSSD2: Input connected to light curtain output
- GuardMode: Mode selection, Guard only
- SingleBreak: Mode selection, Single break mode
- DoubleBreak: Mode selection, Double break mode
- CycleContact: Input for initiation of counter for single and double break.
  After the input is set the counter is decremented at each RED/GREEN transition.
- Reset: Input for reset push button.
- Test: Input which must be set in the moment when the output is switching. When the output has switched on the input can be off.
- IndReset: Output for indicator in reset pushbutton. Flashing when reset is possible, constantly on if conditions for reset is not fulfilled.


LightCurtain_OSSD1
M0.10
LightCurtain_OSSD2
M0.11
ModeSwitch_Guard
M0.6
ModeSwitch_SinglBreak
M0.7
ModeSwitch_DoubleBreak
M0.8
MachineContact
M0.9
ResetButton
I0.10
MonitoringCircuit
I0.2
LightCurtain1
OSSD1
Q
OSSD2
IndReset
GuardMode
SingleBreak
DoubleBreak
CycleContact
Reset
Test
Q0.0
IndicationResetButton
Q0.12

LightCurtain2

Function block for light curtains with three main modes of operation:
“Guard only”, “Single break”, “Double break” mode and possibility for selection of reset mode.

Function description:
In “Guard only” mode the input “GuardMode” shall be set. The input “CycleContact” works as enable function in this mode.

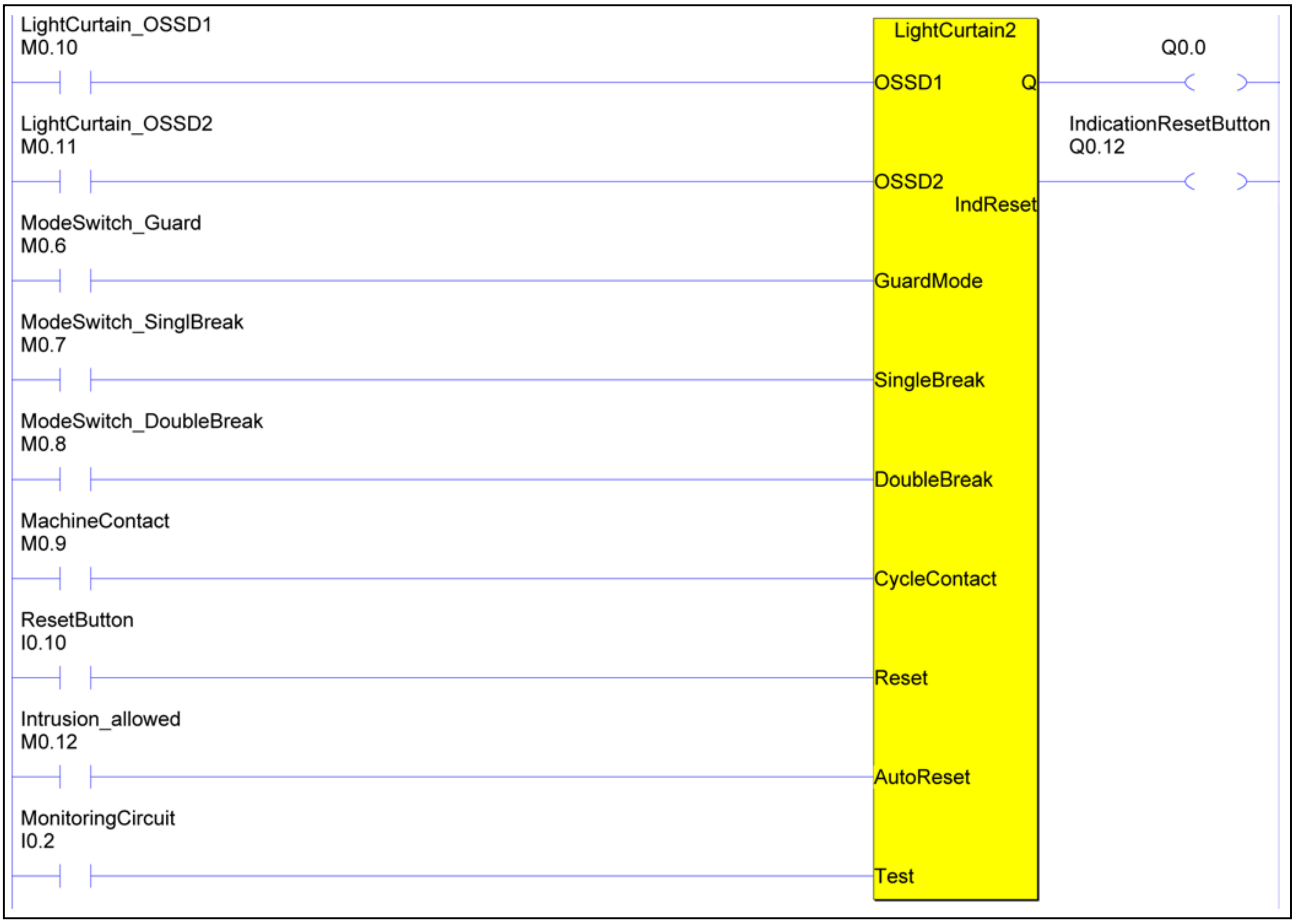
If AutoReset is on, the output Q swtiches on directly when the light curtain is green (OSSDs on).
If AutoReset is off, Q switches on after input “Reset” is activated.

Single break and Double break modes are intended for machines where the light curtain is used for initiation of a new cycle (typically for presses).
When the machine is prepared for a new cycle it shall set the input “CycleContact”. If “SingleBreak” is set output Q is not set before the light curtain has been interrupted/red once and in “DoubleBreak” mode it must be interrupted/red two times.
If “CycleContact” is set in Single and Double break mode for more than 30 seconds a timeout function will trip out in order to avoid unintended start. Reset is required to restore normal operation.

Reset is normally connected to a Reset push button. If the input “AutoReset” is on, no activation of the input “Reset” is required after an intrusion in the light curtain (OSSD inputs off/red light curtain). If “AutoReset” is off, Reset must be activated after intrusion in the light curtain when the output “Q” is set. (When the machine is stopped by the light curtain.) When the light curtain is expecting interruption in Single and Double break modes Reset is not required.

Reset must be activated after following cases, even if “AutoReset” is set:
- After change of mode, “Guard only”, “Single break”, “Double break”.
- If input Test is not activated when output “Q” shall switch on.
- When the light curtain is not activated within 30 sec. in single and double break modes.

Description of in- and outputs:
- OSSD1: Input connected to light curtain output
- OSSD2: Input connected to light curtain output
- GuardMode: Mode selection, Guard only
- SingleBreak: Mode selection, Single break mode
- DoubleBreak: Mode selection, Double break mode
- CycleContact: Input for initiation of counter for single and double break.
  After the input is set the counter is decremented at each RED/GREEN transition.
- Reset: Input for reset push button.
- AutoReset: When input is set no reset is required after intrusion in the light curtain sensing field.
- Test: Input which must be set in the moment when the output is switching. When the output has switched on the input can be off.
- IndReset: Output for indicator in reset pushbutton. Flashing when reset is possible, constantly on if conditions for reset is not fulfilled.


LightCurtain_OSSD1
M0.10
LightCurtain2
OSSD1
Q
Q0.0
LightCurtain_OSSD2
M0.11
OSSD2
IndReset
IndicationResetButton
Q0.12
ModeSwitch_Guard
M0.6
GuardMode
ModeSwitch_SinglBreak
M0.7
SingleBreak
ModeSwitch_DoubleBreak
M0.8
DoubleBreak
MachineContact
M0.9
CycleContact
ResetButton
I0.10
Reset
Intrusion_allowed
M0.12
AutoReset
MonitoringCircuit
I0.2
Test

ValueComp1

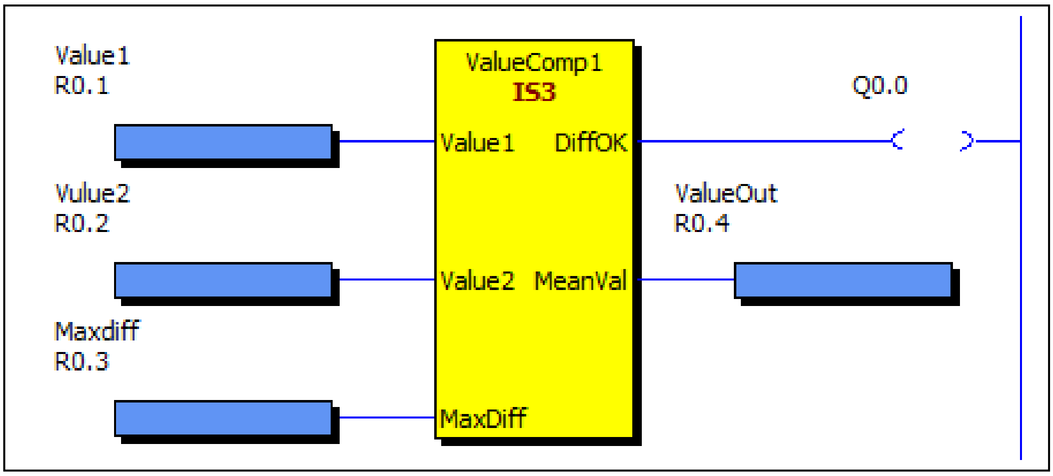
Comparison of two registers and mean value calculation

Description of in- and outputs:

- Value1: Input value 1. Can be register (R) or constant.
- Value2: Input value 2. Can be register (R) or constant.
- MaxDiff: Maximum allowed difference between Value1 and Value2. Can be register (R) or constant, positive value.
- DiffOK: Output for result if the difference between Value1 and Value2 is less or equal to MaxDiff.
- MeanVal: Mean value of Value1 and Value2. Shall be connected to register (R).

Description:

Output DiffOK is high "1" if the difference between Value1 and Value2 is less or equal to MaxDiff.
It does not matter if Value1 is greater or smaller the value2.
The output MeanVal shows the mean value of Value1 and Value2. (Even if DiffOK is off)



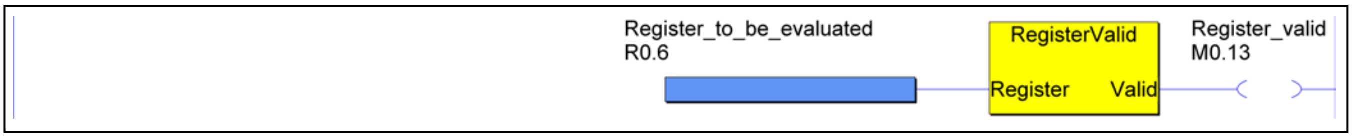
RegisterValid

Valid value zero for register over bus (R=0)

- Register	Register to be evaluated (e.g. R0.0, R10.3).
- Valid	Output which is set “1” if the received register value is valid.

Description:
The block is to be used when a register is exported from a Pluto (marked "E"), and the value is sent via bus to other Pluto-units.
At a bus interruption the receiving Pluto can not read the register and the value is then set to “0”. Sometimes it is important to know if the value really is “0”, or if this is caused by an error. By adding the block RegisterValid in the receiving Pluto a memory (M) will indicate if the value is valid or not. The memory can then be used together with an ordinary register comparison.

For evaluation of double registers the block DRegisterValid shall be used.

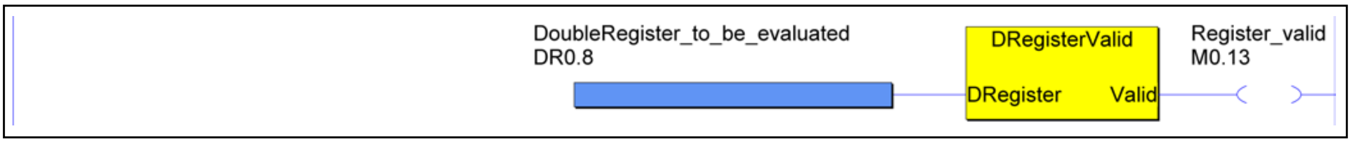


DRegisterValid

Valid value zero for double register over bus (DR=0)

- DRegister	Double register for evaluation (e.g. DR0.0, DR10.3).
- Valid	Output which is set “1” if the received double register value is valid.

Description:
The block is to be used when a register is exported from a Pluto (marked "E"), and the value is sent via bus to other Pluto-units.
At a bus interruption the receiving Pluto can not read the register and the value is then set to “0”. Sometimes it is important to know if the value really is “0”, or if this is caused by an error. By adding the block DRegisterValid in the receiving Pluto a memory (M) will indicate if the value is valid or not. The memory can then be used together with an ordinary register comparison.



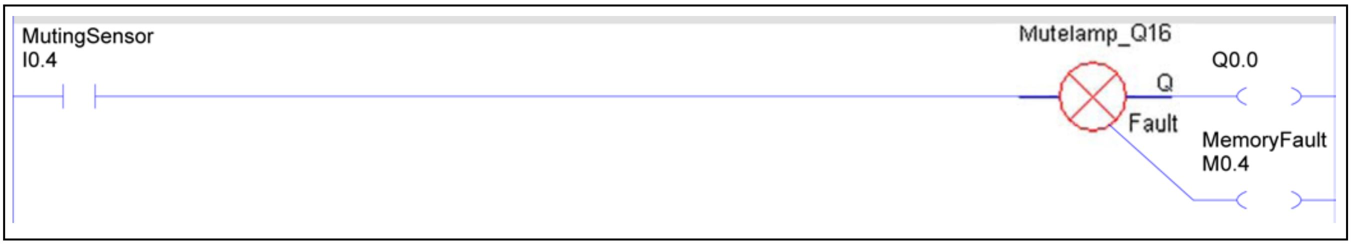
A20

Mutelamp_Q16

Monitoring of current to muting indicator at output Q16. (Only Pluto A20)

- In1 is primary input to which the output of a muting function can be connected.
- Fault is an output indicating that no current is fed to Q16.

Description:
The block is made for monitoring of the current in a muting indicator connected to Q16.
The current limits min. and max are set to 56 and 800 mA for min 2W bulb.
When In1 is activated the indicators are switched on and when the current is within limits the output of the block “Q” is set. The current may not afterwards pass the limits. If no current is detected or exceeds the upper limit the output switches off and the output 'Fault' is activated.

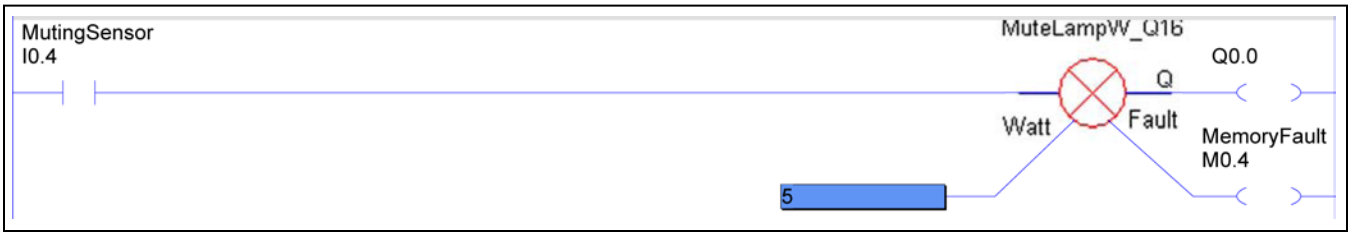


MutelampW_Q16

Monitoring of current to muting indicator at Q16. (Only Pluto A20)

- In1 is primary input to which the output of a muting function can be connected.
- Watt_Q16 is a constant with the value in Watt for the bulb connected to Q16.
- IndQ16 is an output from the block which if used must be connected to Q16 where also the muting indicator must be connected.
- Fault is an output indicating that no current is fed to Q16.

Function:
The block is made for monitoring of current in a muting indicator connected to Q16.
At the input Watt_Q16 the value of the bulb sizes at Q16 can be declared. This value is then translated to a current limit. 1 Watt gives 28 mA, 2 Watt 56 mA etc.
When In1 is activated the indicator is switched on and when the current has exceeded the min limit value the output of the block is set. The current may not afterwards drop below the min limit or exceed the max limit (800mA). If no current is detected the output switches off and the output Fault is activated.



Analog

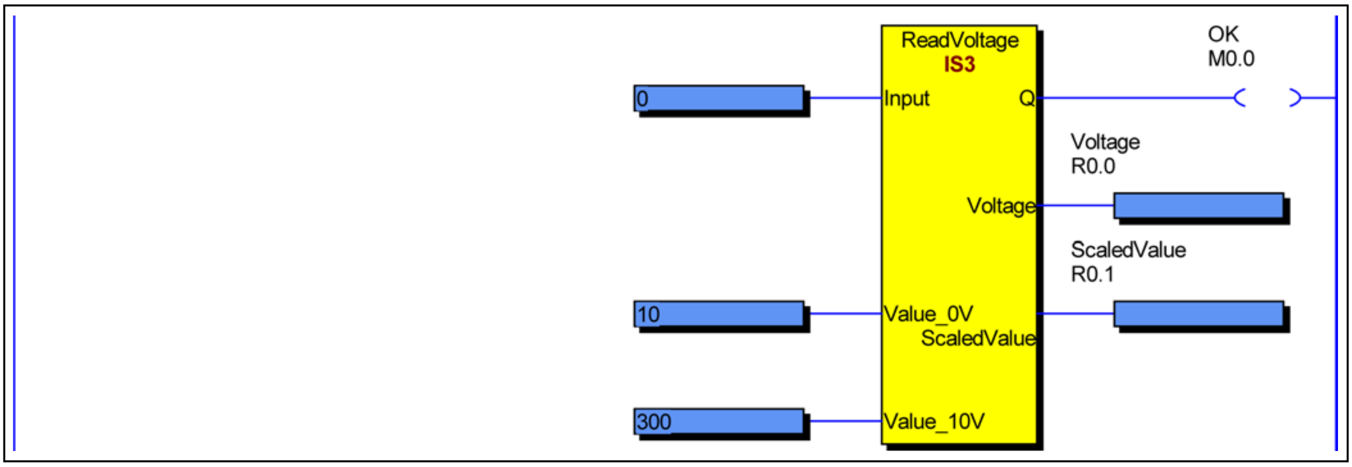
ReadVoltage

Reading of analogue input, 0-10 Volt

In- and outputs:
- Input Constant for selection of analogue input (0=IA0, 1=IA1, …)
- Value_0V Scaling Value for what the output ScaledValue shall show at 0 volt on the input.
- Value_10V Scaling Value for what the output ScaledValue shall show at 10 volt on the input.

- OK OK output. Value is within range.
- Voltage Measured input value in millivolt (mV). Shall be connected to a register (R).
- ScaledValue Output for ScaledValue. Shall be connected to a register (R).
Scaling according to Value_0V and Value_10V.

Description:
The function block reads an analogue input configured for 0-10V.
The output Voltage shows the real input voltage in mV.
The output ScaledValue shows a scaled value whose scaling is determined by the inputs Value_0V and Value_10V.
Example: For a pressure sensor giving 0V at 10 Bar and 10V at 300 Bar the values shall be set: Value_0V=10, Value_10V=300.
Max value is 32767, for higher values the block ReadVoltage32 can be used.



ReadVoltage32

Reading of analogue input, 0-10 Volt

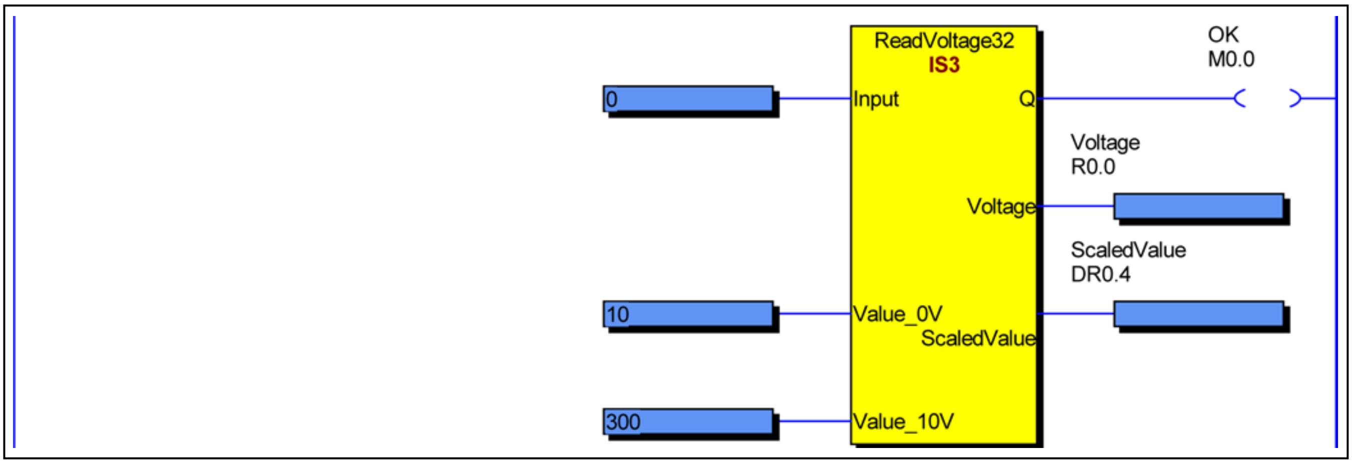
<u>In- and outputs:</u>

- Input Constant for selection of analogue input (0=IA0, 1=IA1, …)
- Value_0V Scaling Value for what the output ScaledValue shall show at 0 volt on the input.
- Value_10V Scaling Value for what the output ScaledValue shall show at 10 volt on the input.

- OK OK output. Value is within range.
- Voltage Measured input value in millivolt (mV). Shall be connected to a register (R).
- ScaledValue Output for ScaledValue. Shall be connected to a double register (DR).
  Scaling according to Value_0V and Value_10V.

<u>Description:</u>

The function block reads an analogue input configured for 0-10V.
The output Voltage shows the real input voltage in mV.
The output ScaledValue shows a scaled value whose scaling is determined by the inputs Value_0V and Value_10V.
Example: For a pressure sensor giving 0V at 10 Bar and 10V at 300 Bar the values shall be set: Value_0V=10, Value_10V=300.
There is a variant of the block named ReadVoltage where ScaledValue is connected to a single register (R) instead of a double register (DR).



ReadCurrent

Reading of analogue input, 4-20 mA

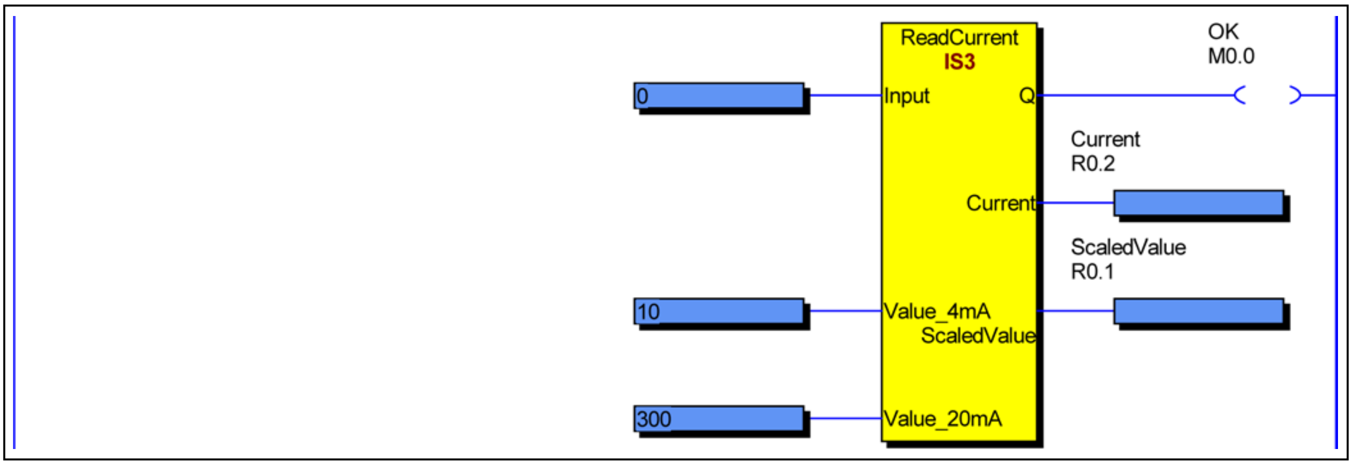
<u>In- and outputs:</u>

- Input Constant for selection of analogue input (0=IA0, 1=IA1, …)
- Value_4mA Scaling Value for what the output ScaledValue shall show at 4 mA on the input.
- Value_20mA Scaling Value for what the output ScaledValue shall show at 20 mA on the input.

- OK OK output. Value is within range.
- Current Measured input value in microampere (µA). Shall be connected to a register (R).
- ScaledValue Output for ScaledValue. Shall be connected to a register (R).
  Scaling according to Value_4mA and Value_20mA.

<u>Description:</u>

The function block reads an analogue input configured for 4-20 mA.
The output Current shows the real input voltage in µA.
The output ScaledValue shows a scaled value whose scaling is determined by the inputs Value_4mA and Value_20mA.
Example: For a pressure sensor giving 4mA at 10 Bar and 20mA at 300 Bar the values shall be set: Value_4mA=10 and Value_20mA=300.
Max value is 32767, for higher values the block Read Current32 can be used.



ReadCurrent32

Reading of analogue input, 4-20 mA

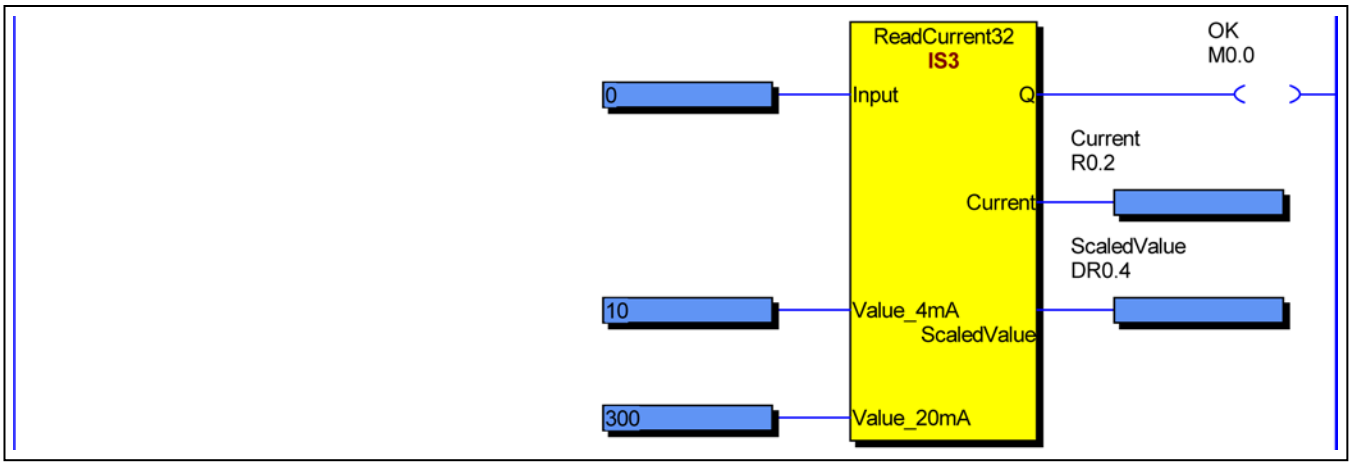
<u>In- and outputs:</u>

- Input Constant for selection of analogue input (0=IA0, 1=IA1, …)
- Value_4mA Scaling Value for what the output ScaledValue shall show at 4 mA on the input.
- Value_20mA Scaling Value for what the output ScaledValue shall show at 20 mA on the input.

- OK OK output. Value is within range.
- Current Measured input value in microampere (µA). Shall be connected to a register (R).
- ScaledValue Output for ScaledValue. Shall be connected to a double register (DR).
Scaling according to Value_4mA and Value_20mA.

<u>Description:</u>

The function block reads an analogue input configured for 4-20 mA.
The output Current shows the real input voltage in µA.
The output ScaledValue shows a scaled value whose scaling is determined by the inputs Value_4mA and Value_20mA.
Example: For a pressure sensor giving 4mA at 10 Bar and 20mA at 300 Bar the values shall be set: Value_4mA=10 and Value_20mA=300.
There is a variant of the block named ReadCurrent where ScaledValue is connected to a single register (R) instead of a double register (DR).



ASi

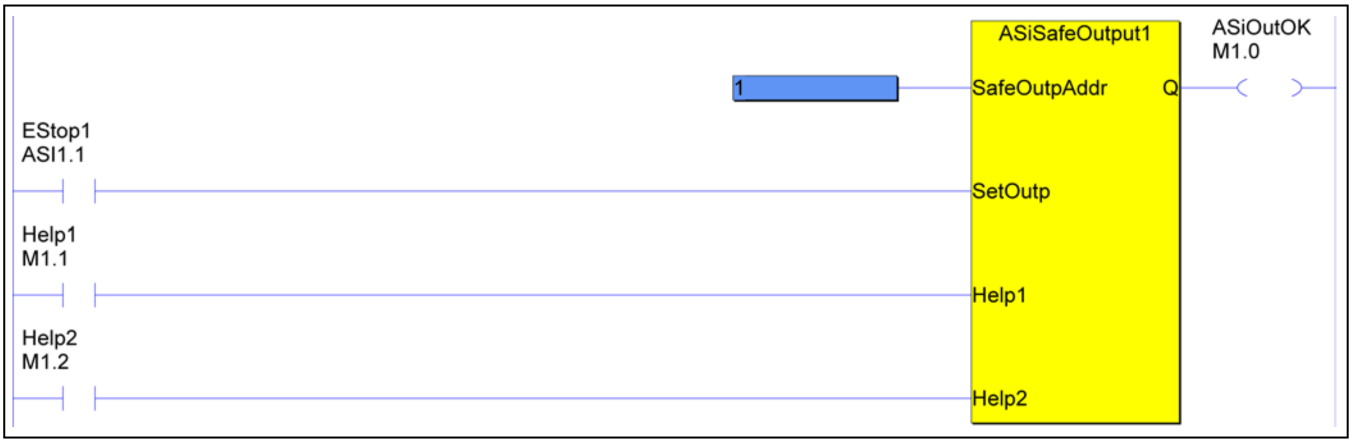
ASiSafeOutput1

Control of AS-i safe output slave (with state ON/OFF)

In- and outputs:
- SafeOutpAddr is an input for a constant setting the address of the output slave (1-31).
- SetOutp is an input for switching the safe output in the slave on and off.
- Help1 and Help2 are two inputs for sending "help"-signals to the slave.
  How and if the help-signals shall be used is individual for different slaves on the market.
  For information see user instruction for the actual output slave.
- Q is an output that is set if the slave is demanded to set, but also that it is declared on the page "AS-i options" and the block is correct executed.

Description:
The function block controls the safe output in the slave with the block input "SetOutp".
The address of the output slave shall be given to the block by connecting a constant to the input "SafeOutpAddr".
On the configuration page "AS-i Options" the corresponding address must be configured as "Safe Output". The physical output slave unit must also be addressed. This is performed either by an AS-i addressing tool or by using Pluto and the function "Change AS-i slave address" in Pluto Manager.

The signals Help1 and Help2 shall be used according to the manufacturer’s instructions since the usage can differ. In some cases they are used for reset of different faults.

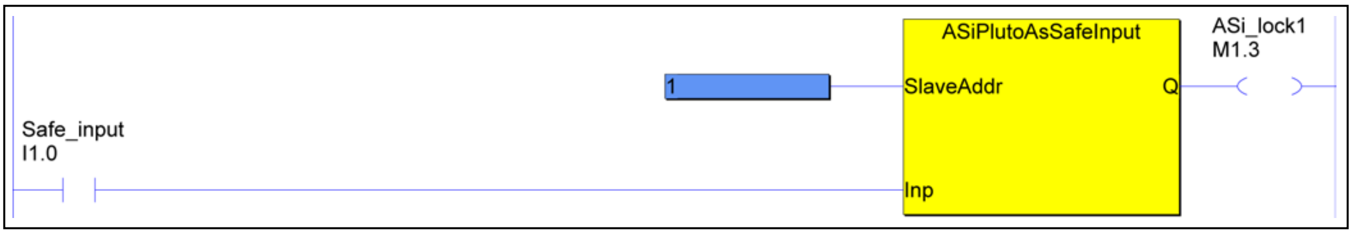


ASiPlutoAsSafeInput

Pluto simulates a safe input slave

In- and outputs:
- SlaveAddr is an input for a constant setting the address of the simulated slave (1-31).
- Inp is an input for switching on and off.
- Q is an output that is set if the simulated slave is on, but also that it is declared on the page "AS-i options" and the block is correctly executed.

Description:
The function block makes the Pluto simulate a safe input slave which can be read by another Pluto AS-i connected to the AS-i bus. It is then possible to transfer safe variables to other Pluto units over the AS-i bus.
The address of the output slave shall be given to the block by connecting a constant to the input "SlaveAddr". On the configuration page "AS-i Options" the corresponding address must be configured as "Pluto as Safe Input".



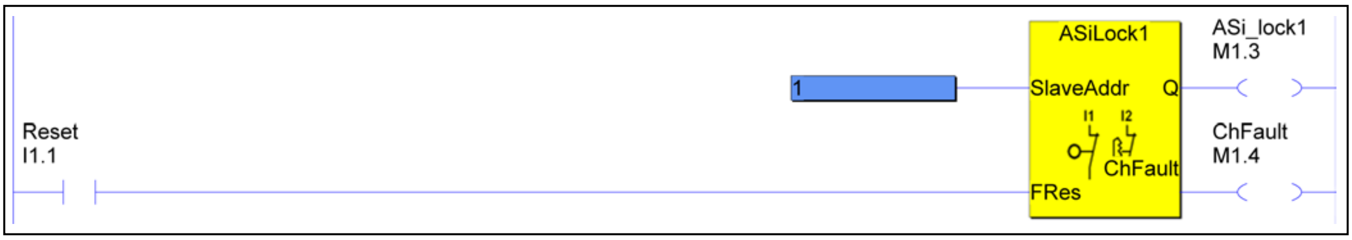
ASiLock1

As-i lock

This block performs channel monitoring on one channel only, typically a door lock:
Channel 1 is used for the guard (protective switch).
Channel 2 is used for the lock monitor.
If the lock is opened, the guard can also be opened. This sequence of events is monitored.
It is an error if the protective switch opens while the lock monitor is active.

The AS-i lock slave shall be configured as Safe input with no channel monitoring on the AS-i options page. The channel monitoring is done inside the block.

- SlaveAddr is input for a constant specifying the lock's AS-i address (1-31).
- FRes is an input that resets error (positive edge).
- Q is an output which is set when the lock is activated, the door is closed and no error is active. The AS-i input must also be configured correctly.
- ChFault gets activated if channel 2 is ON while channel 1 is OFF.



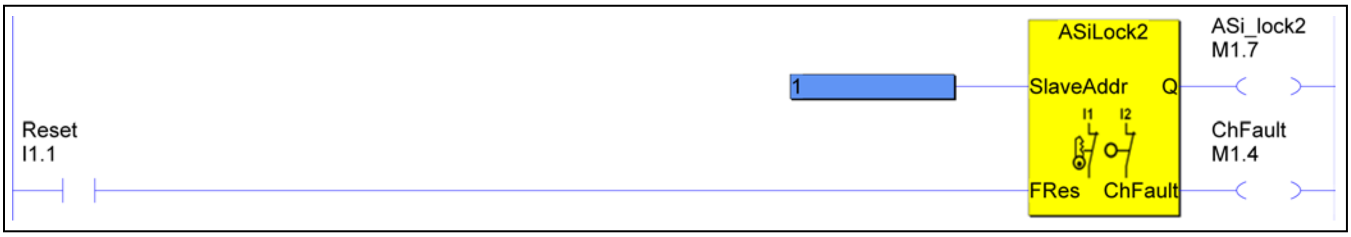
ASiLock2

As-i lock

This block performs channel monitoring on one channel only, typically a door lock:
Channel 2 is used for the guard (protective switch).
Channel 1 is used for the lock monitor.
If the lock is opened, the guard can also be opened. This sequence of events is monitored.
It is an error if the protective switch opens while the lock monitor is active.

The AS-i lock slave shall be configured as Safe input with no channel monitoring on the AS-i options page. The channel monitoring is done inside the block.

- SlaveAddr is input for a constant specifying the lock's AS-i address (1-31).
- FRe is an input that resets error (positive edge).
- Q is an output which is set when the lock is activated, the door is closed and no error is active.
  The AS-i input must also be configured correctly.
- ChFault gets activated if channel 1 is ON while channel 2 is OFF.



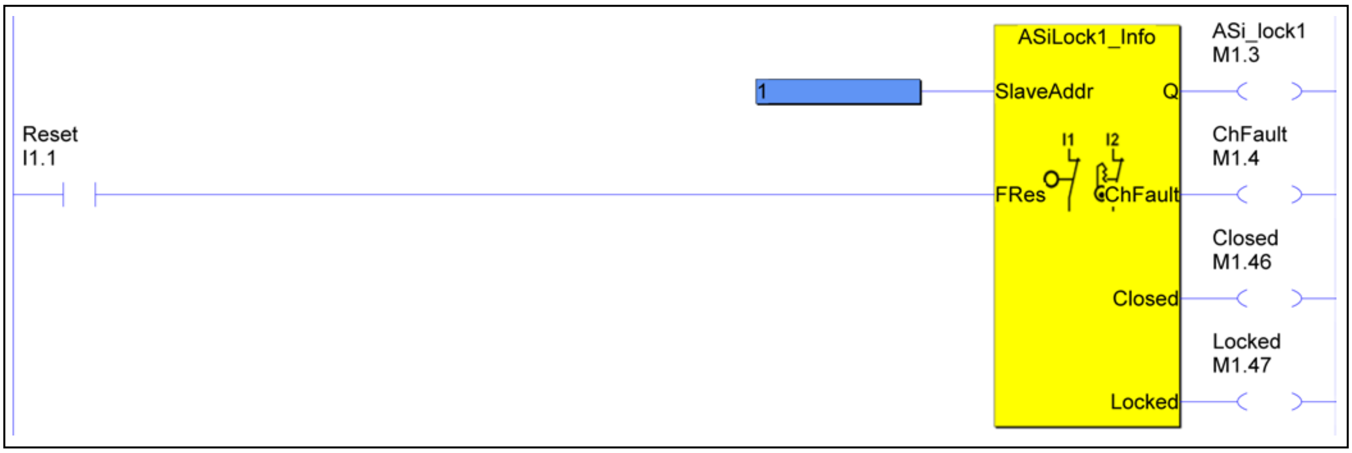
ASiLock1_Info

As-i lock

In- and outputs
- SlaveAddr is input for a constant specifying the lock's AS-i address (1-31).
- FRes is an input that resets error (positive edge).
- Q is an output which is set when the lock is activated, the door is closed and no error is active. The AS-i input must also be configured correctly.
- ChFault gets activated if channel 2 is ON while channel 1 is OFF.
- Closed is a non-safe output indicating that the door is closed.
- Locked is a non-safe output indicating that the door is locked.

Description:
This block performs channel monitoring on one channel only, typically a door lock:
  Channel 1 is used for the guard (protective switch).
  Channel 2 is used for the lock monitor.
If the lock is opened, the guard can also be opened. This sequence of events is monitored.
It is an error if the protective switch opens while the lock monitor is active.

The AS-i lock slave shall be configured as Safe input with no channel monitoring on the AS-i options page. The channel monitoring is done inside the block.



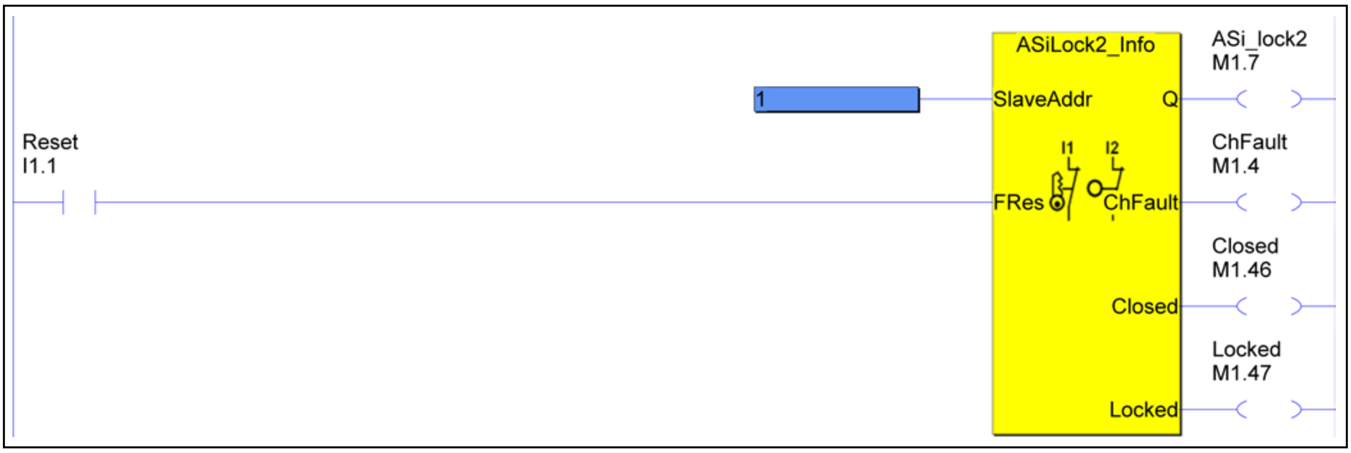
ASiLock2_Info

As-i lock

In- and outputs:
- SlaveAddr is input for a constant specifying the lock's AS-i address (1-31).
- FRes is an input that resets error (positive edge).
- Q is an output which is set when the lock is activated, the door is closed and no error is active. The AS-i input must also be configured correctly.
- ChFault gets activated if channel 1 is ON while channel 2 is OFF.
- Closed is a non-safe output indicating that the door is closed.
- Locked is a non-safe output indicating that the door is locked.

Description:
This block performs channel monitoring on one channel only, typically a door lock:
Channel 2 is used for the guard (protective switch).
Channel 1 is used for the lock monitor.
If the lock is opened, the guard can also be opened. This sequence of events is monitored.
It is an error if the protective switch opens while the lock monitor is active.

The AS-i lock slave shall be configured as Safe input with no channel monitoring on the AS-i options page. The channel monitoring is done inside the block.



ASiAnalogInput

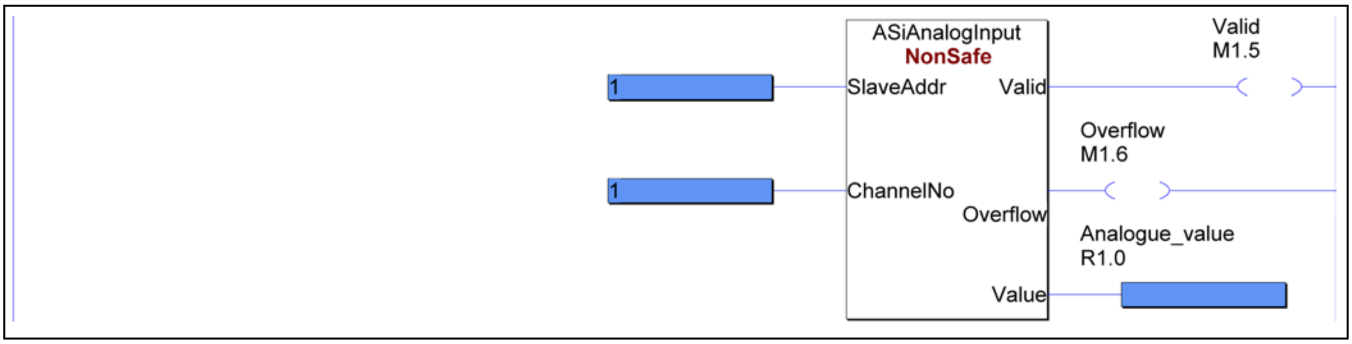
For reading of analogue input slave. (Non-safe)
Profiles: S-7.3.C, S-7.3.D, S-7.3.E and S-7.3.F

In- and outputs:

- SlaveAddr is an input for a constant setting the address of the analogue slave (1-31).
- ChannelNo is an input for a constant setting which channel that shall be read (1-4).
- Valid is an output that is set when the slave is correctly declared, Pluto is communicating with it and it has no internal faults.
- Overflow is an output that is set when the analogue value is out of range.
- Value is the output for the analogue value. The output shall be connected to a register.

Description:
The function block reads the analogue value in an AS-i analogue input slave. The value shall be connected to a register. By further evaluation of the register in the PLC program it should be used together with the output Valid which indicates that the value is reliable.

Note: The analogue value is not safety related.



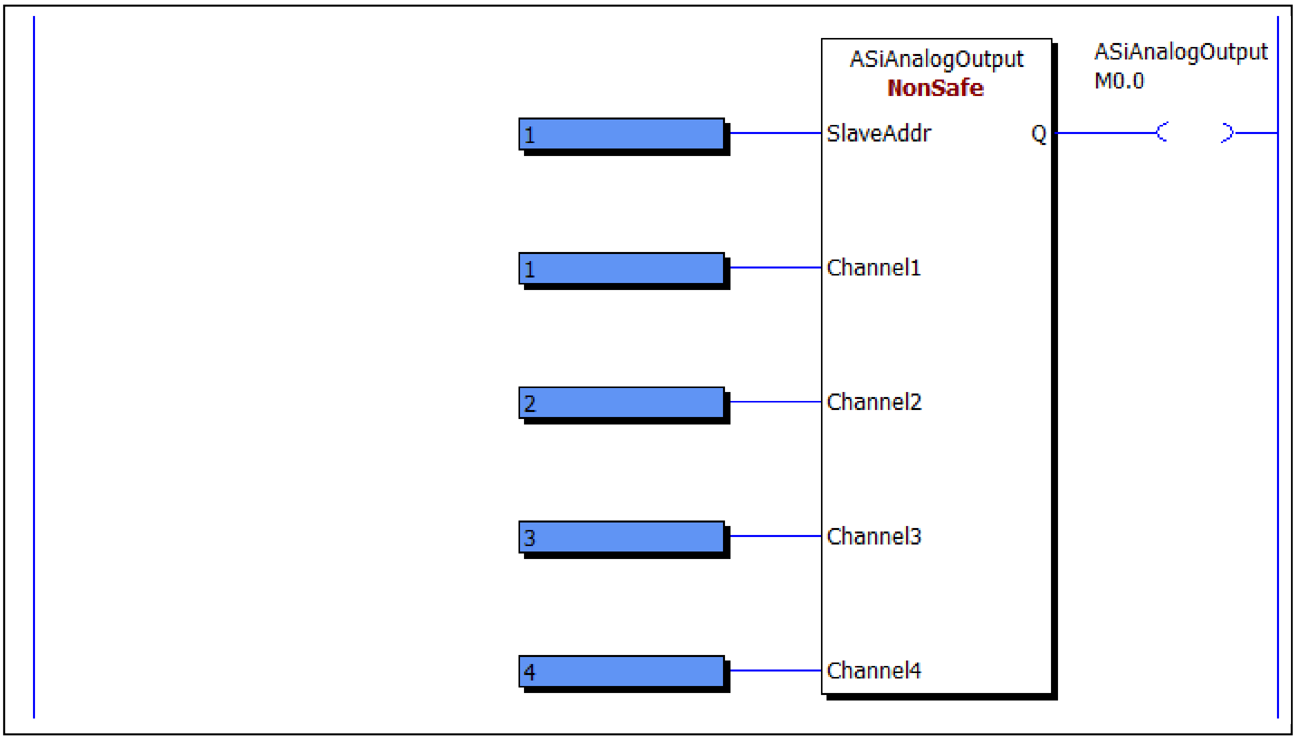
ASiAnalogOutput

For writing to analogue output slave. (Non-safe)
Profiles: S-7.3.4, S-7.3.5 and S-7.3.6

Inputs:
- SlaveAddr is an input for a constant setting the address of the analogue slave (1-31).
- Channel1 is the value to be written to channel 1.
- Channel2 is the value to be written to channel 2.
- Channel3 is the value to be written to channel 3.
- Channel4 is the value to be written to channel 4.

Description:
The function block writes analogue values to an AS-i analogue output slave. Up to 4 channels can be written.

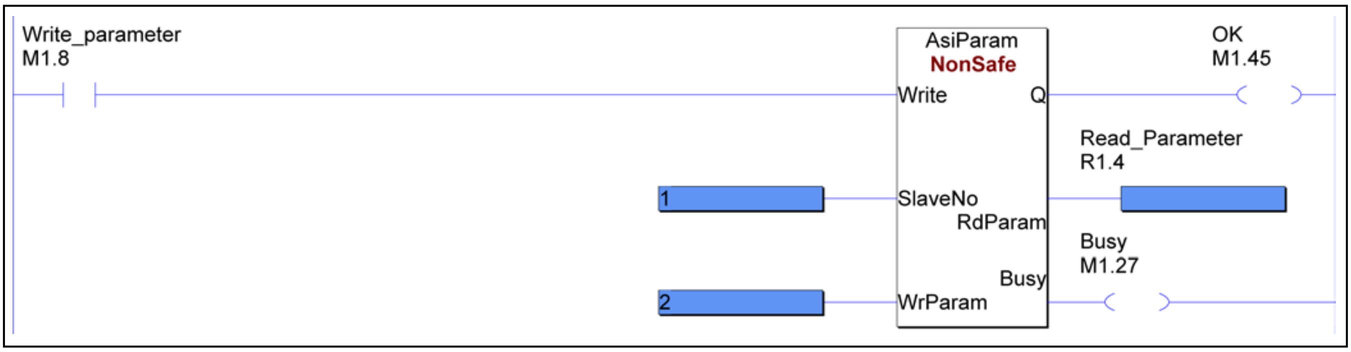
Note: The analogue value is not safety related.



ASiParam

Write parameter to AS-i slave and get parameter result

In- and outputs:
- Write is an input for write request on positive edge.
- SlaveNo is an input for AS-i slave no. B slaves are coded as 32+slave no.
- WrParam is an input for Parameter to be written (0..15).
- Q is an output which is set at successful operation.
- RdParam is an output for Parameter value read back.
- Busy is an output which is set when operation is in progress or other slave is being handled.



Counters

HS_SpeedCount_Dir

Speed measurement with direction. For sensors with A/B pulses connected to counter inputs.

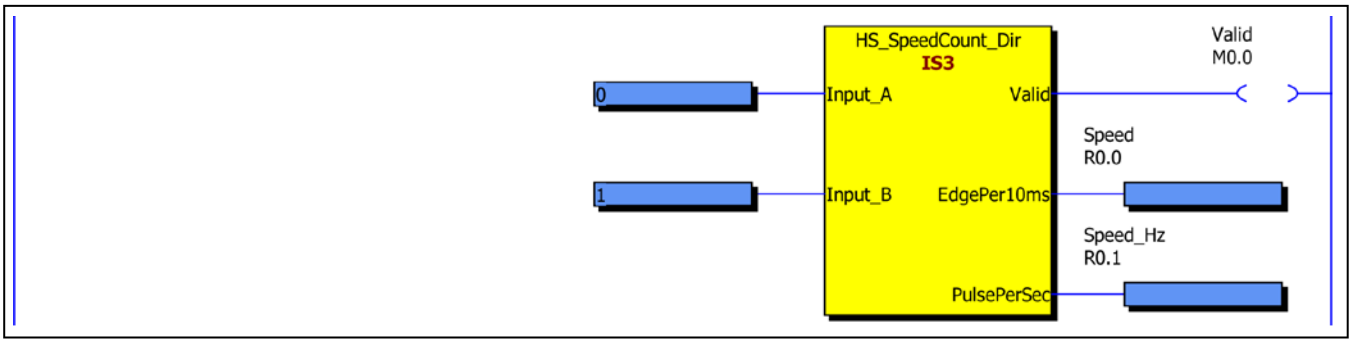
In- and outputs:

- Input_A and Input_B: Constants for selection of which input shall be read.
  For IA0-IA1: Input A=0, Input B=1
  For IA2-IA3: Input A=2, Input B=3
- EdgePer10ms: Output for speed value in edges/10ms. Shall be connected to a register (R). (An incremental encoder with A/B-pulses rated 1024 ppr gives 4096 edges per revolution.)
- PulsePerSec: Output for speed value in pulses/sec. Shall be connected to a register (R). (An incremental encoder with A/B-pulses rated 1024 ppr shows 1024 at 1 rev/sec.)
- Valid: Output which shall be on (1). At fault it is set low (0).

Description:
The function block reads a high speed counter input configured as “Counter input – Up/Down” in I/O Options.

To the input pair, a sensor with A/B pulses shall be connected. This is a common standard for incremental encoders meaning two signals 90° phase shifted to each other. The direction of a motion can then be determined which is shown by a positive or negative value at the outputs EdgePer10ms and PulsePerSec.
Pluto and the block counts the edges that the A and B signals generate together during 10 ms. Each period contains 4 edges. This value is shown at the output “EdgePer10ms”. The output “PulsesPerSec” is a recalculation to pulses per second corresponding to the rating of an encoder. (An incremental encoder with A/B pulses is normally marked with pulses per revolution, ppr.)
Note that the resolution is 25. The relation between the inputs is: 1 EdgePer10ms = 25 PulsePerSec (100/4). Example: A sensor rated 512 ppr rotates 1 revolution / second. The output PulsePerSec shows then 512 and EdgePer10ms shows 20 (20.48).

When the block detects faults the output Valid switches off and the output values show 32 767. Such fault can be too high frequency. Note that too high frequency also can give 0 speed because of electrical reasons such as type of sensor output.



HS_SpeedCount_Up

Measurement of speed at counter inputs

<u>In- and outputs:</u>

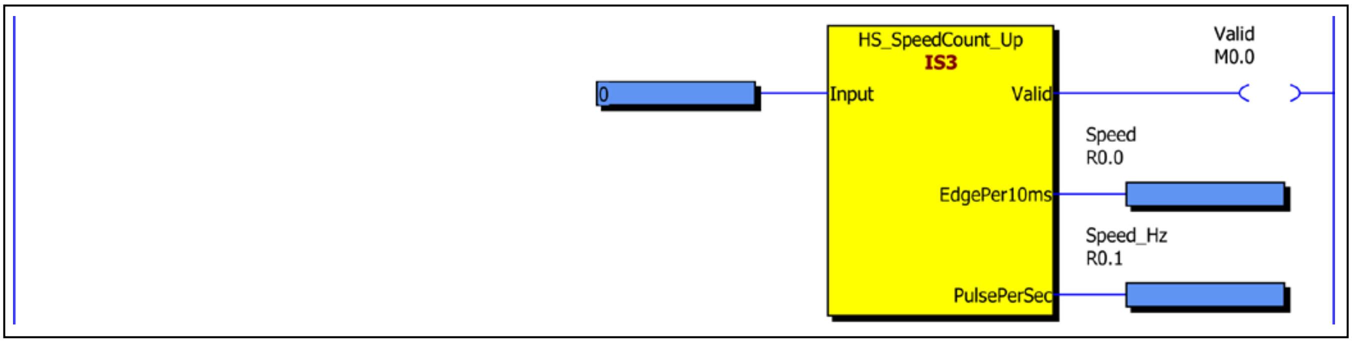
- Input Constant for selection of input. (0=IA0, 1=IA1, …)
- EdgePer10ms Output for speed value in edges/10ms. Shall be connected to a register (R).
- PulsePerSec Output for speed value in pulses/sec. Shall be connected to a register (R).
- Valid Output which shall be on (1). At fault it is set low (0).

<u>Description:</u>

The function block reads a high speed counter input configured as “Counter input – Up” in I/O Options.

To the input a sensor generating a square wave pulse shall be connected. Example of sensor can be a proximity switch, a photo cell or an incremental encoder.
The function block counts the amount of edges on the square wave pulse during 10 ms (1 program cycle). This value is shown at the output “EdgePer10ms”. One pulse/period has two edges. The output “PulsePerSec is a recalculation to pulses per second. Note that the resolution is 50. The relation between the output is: 1 EdgePer10ms = 50 PulsePerSec (100/2). The outputs always show a positive value since direction is not possible to detect with one input.

When the block detects faults the output Valid switches off and the output values show 32 767. Such fault can be too high frequency. Note that too high frequency also can give 0 speed because of electrical reasons such as type of sensor output.



SpeedMon1

Dual channel speed monitoring with two input registers

In- and outputs:

- Speed Input register for speed value (Primary speed input).
- CompSpeed Input register for monitoring of the value main speed connected to the input “Speed”.
- MaxDiff Input for the maximum allowed difference between “Speed” and “CompSpeed”.
- DiffDelay Timer input for off-delay of output SpeedValid (and ValidSpeed) by comparison.
- LowSpeedLim Limit value for safe low speed.
- SpeedValid Output for when the two speed values are within the limit of “MaxDiff”.
- ValidSpeed Normally equal to the input “Speed”. At fault 32767.
- StandStill Output set at standstill.
- SafeLowSpeed Output set when speed is less than input “LowSpeedLim”. Shall be a positive value.

Description:

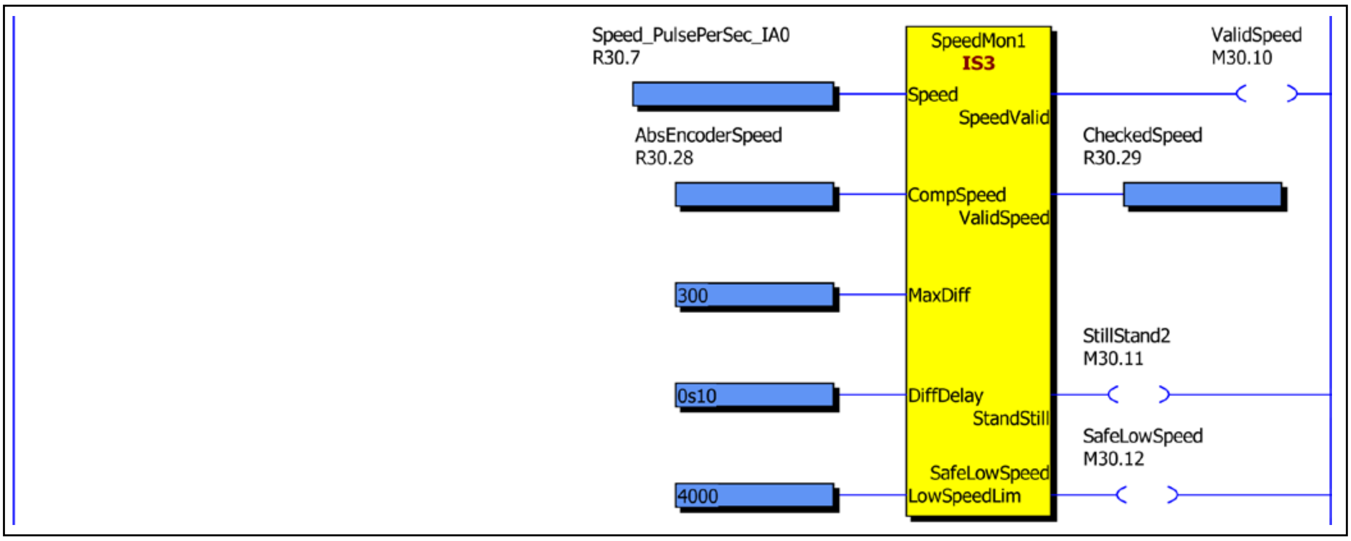
The block is intended to give a redundant speed monitor function but can also be used for other purposes. It has two inputs for speed values. These input registers can take their values from different sources such as the function blocks for incremental encoders, absolute encoders, analogue inputs etc. The input “Speed” is a primary input for a speed value and “CompSpeed” is a secondary channel for monitoring the correctness of the primary speed value.

The function block has three safety functions:

- Compares a register “Speed” with a second register for speed “CompSpeed”, and checks that the difference is not more than the value set at the input register “MaxDiff”. If the difference is within the limit the output “SpeedValid” is set, and the output “ValidSpeed” will be equal to “Speed”. If “MaxDiff” is exceeded “SpeedValid” will go off and “ValidSpeed” will be set to 32767.
  The timer input "DiffDelay" is an off-delay for the comparison. The block allows the two values “Speed” and “CompSpeed” to differ more than MaxDiff during this time. This timer can be useful if "CompSpeed" and "Speed" have different response time.
  It is also a fault if "Speed" is -32768 or 32767 (limit values for registers, R_). SpeedValid trips then immediately.

- Stand still monitoring of input "Speed" with hysteresis. The output "StandStill" is set when the value at the input "Speed" has been 0 for 0.7 sec. After that the "Speed" value is allowed to increase/decrease three times in either direction before "StandStill" switches off again and remains off for minimum 2 seconds. (If the block is used in a system without detection of direction the output will switch off when the sensor has switched four times.)

- Safe limit speed (SLS). The output SafeLowSpeed is set when the input value at "Speed" is less than the input value "LowSpeedLim". The "LowSpeedLim" value is valid also for negative speed.


Speed_PulsePerSec_IA0
R30.7
SpeedMon1
IS3
ValidSpeed
M30.10
Speed
SpeedValid
AbsEncoderSpeed
R30.28
CheckedSpeed
R30.29
CompSpeed
ValidSpeed
300
MaxDiff
StillStand2
M30.11
0s10
DiffDelay
StandStill
SafeLowSpeed
M30.12
SafeLowSpeed
4000
LowSpeedLim

Encoders

SafeEncoder

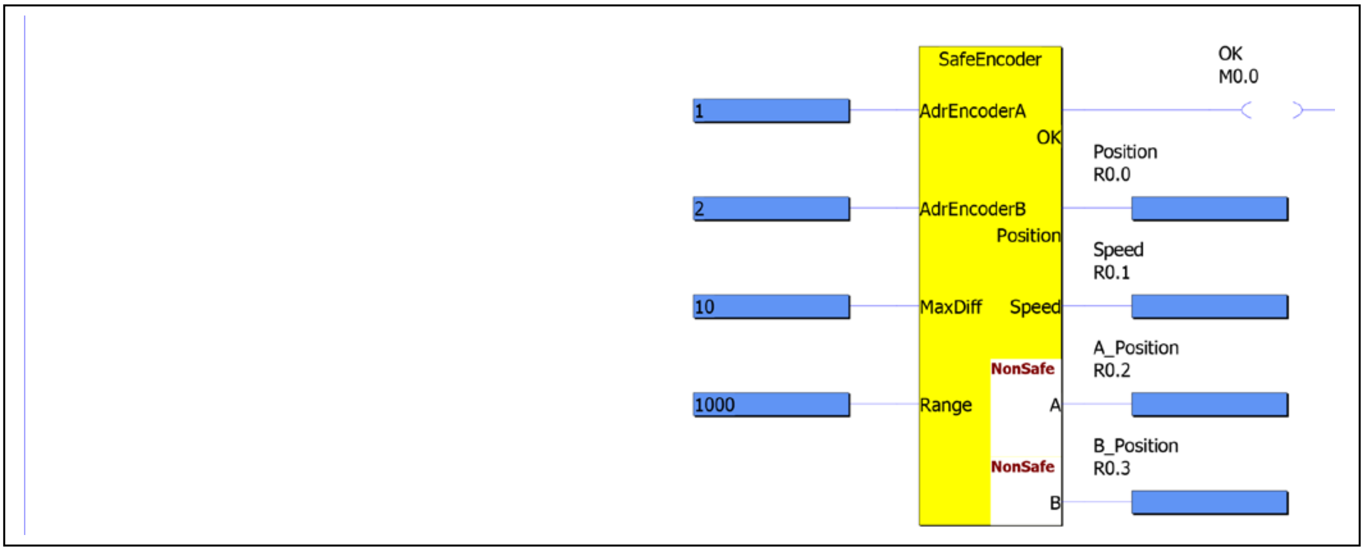
Function block for singleturn encoders generating safe position and speed value out of two absolute encoders.

Function:
The function block reads and evaluates two absolute encoders. The average of the two encoders is calculated and set to the output “Position”. The output ”Speed” is also an average value given in increments/10ms. It is also monitored that the two encoder values do not deviate more than the value set by the input “MaxDiff”.
If something is wrong the output “OK” is set “0”'. In some application the values “Position” and 'Speed' must be used together with the “OK” output.

Description of in- and outputs:
- AdrEncoderA: Encoder A node address
- AdrEncoderB: Encoder B node address
- MaxDiff: Max allowed deviation between the encoders (max 2% of Range)
- Range: Number of increments per revolution
- OK: Set when encoders are working OK and the position values are within the margin set by “MaxDiff”
- Position: Position value
- Speed: Speed value as increments/10ms
- A: Encoder A position. Must not be used in PLC program!
- B: Encoder B position. Must not be used in PLC program!

NOTE! Position values from single encoders are only available for adjustment purposes and must NOT be used for safety.

NOTE! When error occurs “Position” = -1, “Speed” = -32768 and the OK output will be reset.



SafeEncoderMult

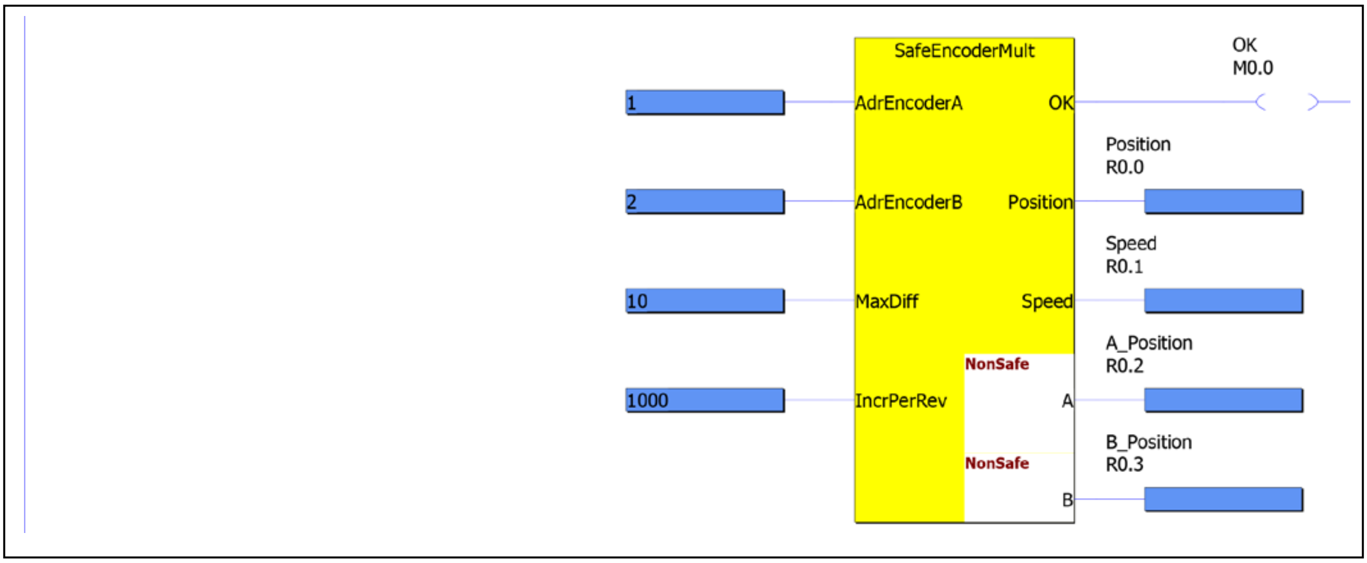
Function block for multiturn encoders generating safe position and speed value out of two absolute encoders.

Function:
The function block reads and evaluates two absolute multiturn encoders. The average of the two encoders is calculated and set to the output “Position”.
The output “Speed” is also an average value given in increments/10ms.
It is also monitored so the two encoder values do not deviate more than the value set by the input “MaxDiff”.
If something is wrong the output “OK” is set “0”. In some application the values “Position” and “Speed” must be used together with the “OK” output.

Description of in- and outputs:
- AdrEncoderA: Encoder A node address
- AdrEncoderB: Encoder B node address
- MaxDiff: Max allowed deviation between the encoders (max 2% of IncrPerRev)
- IncrPerRev: Number of increments per revolution
- OK: Set when encoders are working OK and the position values are within the margin set by “MaxDiff”
- Position: Position value
- Speed: Speed value as increments/10ms
- A: Encoder A position. Must not be used in PLC program!
- B: Encoder B position. Must not be used in PLC program!

NOTE! Position values from single encoders are only available for adjustment purposes and must NOT be used for safety.

NOTE! When error occurs “Position” = -1, “Speed” = -32768 and the OK output will be reset.



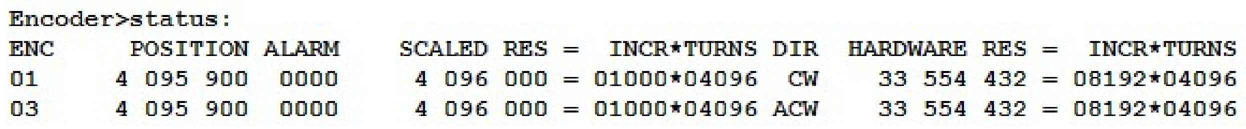
SafeEncoderMult32

Same as SafeEncoderMult but for Pluto with “Instruction set 3” (see Pluto Programming Manual). Can handle double registers.

Function:
The function block reads and evaluates two absolute multiturn encoders. The average of the two encoders is calculated and set to the output “Position”.
The output “Speed” is also an average value given in increments/10ms.
It is also monitored so the two encoder values do not deviate more than the value set by the input “MaxDiff”.
If something is wrong the output “OK” is set “0”. In some application the values “Position” and “Speed” must be used together with the “OK” output.

Description of in- and outputs:
- AdrEncoderA: Encoder A node address
- AdrEncoderB: Encoder B node address
- MaxDiff: Max allowed deviation between the encoders (max 2% of IncrPerRev)
- IncrPerRev: Number of increments per revolution
- Range: This value should be set to the total number of increments (IncrPerRev * number of turns). *See example below.
- OK: Set when encoders are working OK and the position values are within the margin set by “MaxDiff”
- Position: Position value
- Speed: Speed value as increments/10ms
- A: Encoder A position. Must not be used in PLC program!
- B: Encoder B position. Must not be used in PLC program!

*Example: An encoder has a resolution of 1000 increments per revolution (IncrPerRev) and 4096 turns. In this case “Range” should be set to 4096000 and “IncrPerRev” to 1000. In terminal window this can be seen by typing “encoder”, followed by “s”.

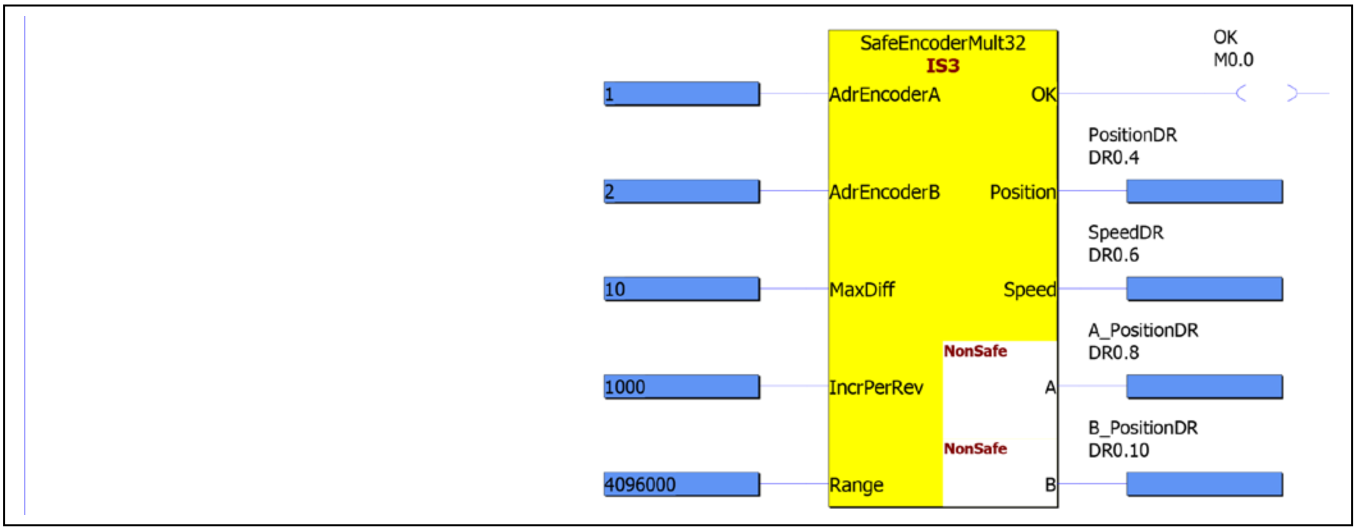


```
Encoder>status:
ENC      POSITION ALARM    SCALED RES =  INCR*TURNS DIR  HARDWARE RES =  INCR*TURNS
01      4 095 900  0000     4 096 000 = 01000*04096  CW     33 554 432 = 08192*04096
03      4 095 900  0000     4 096 000 = 01000*04096 ACW     33 554 432 = 08192*04096
```

*Example: If IncrPerRev = 1000, Range shall be set to 4096000 (1000*4096).*

NOTE! Position values from single encoders are only available for adjustment purposes and must NOT be used for safety.

NOTE! When error occurs “Position” = -1, “Speed” = -2147483648 and the OK output will be reset.

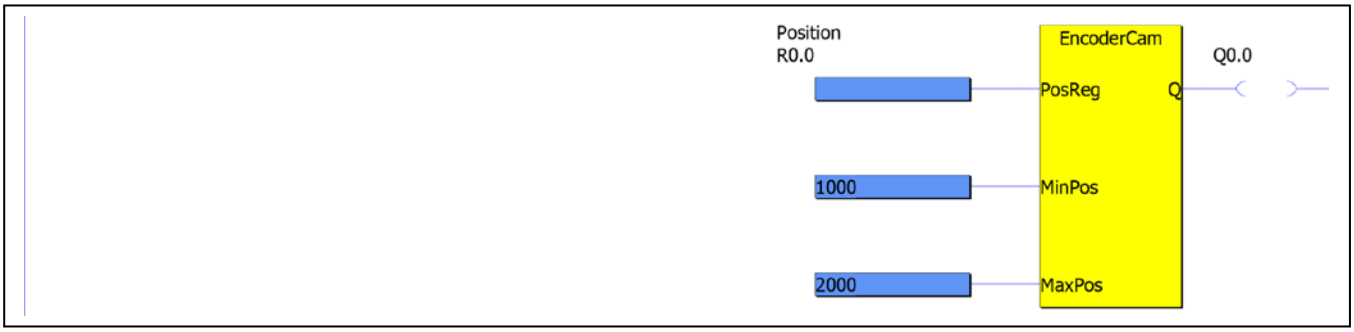

SafeEncoderMult32
IS3
1
AdrEncoderA
OK
OK
M0.0
PositionDR
DR0.4
2
AdrEncoderB
Position
SpeedDR
DR0.6
10
MaxDiff
Speed
A_PositionDR
DR0.8
NonSafe
1000
IncrPerRev
A
B_PositionDR
DR0.10
NonSafe
4096000
Range
B

EncoderCam

Function block for electronic cam limit switch.

Function:
The output Q is set when the value in the input register PosReg is within the limits MinPos and MaxPos. The input value is normally Generated by the function block SafeEncoder.
NOTE! It is possible to specify a cam which passes through the zero position of an encoder. Position<0 is forbidden.
Example: If MinPos=3000 and MaxPos=200, Q is set when the position is greater than 2999 or less than 201.

Description of in- and outputs:
- PosReg: Register where the position is stored
- MinPos: Min allowed value
- MaxPos: Max allowed value

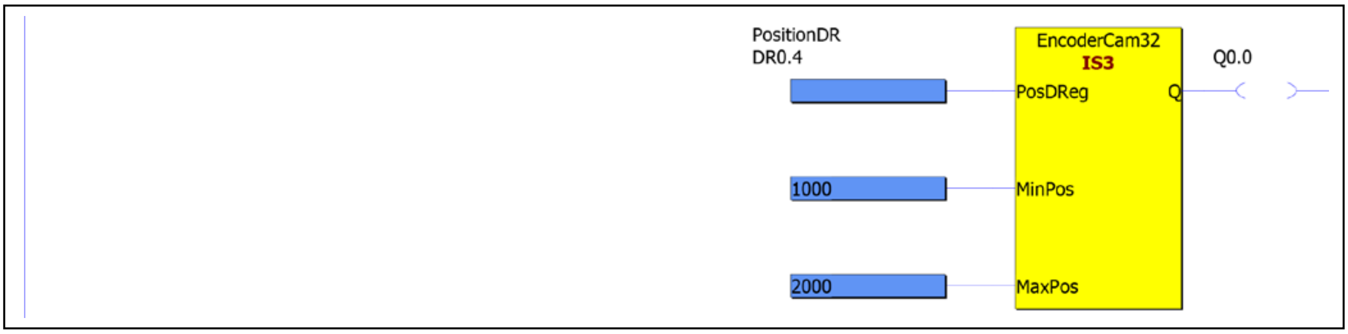


EncoderCam32

Same as EncoderCam but can handle double registers.
Needs “Instruction set 3” (see Pluto Programming Manual).

Function:
The output Q is set when the value in the input register PosReg is within the limits MinPos and MaxPos. The input value is normally Generated by the function block SafeEncoder.
NOTE! It is possible to specify a cam which passes through the zero position of an encoder. Position<0 is forbidden.
Example: If MinPos=3000 and MaxPos=200, Q is set when the position is greater than 2999 or less than 201.

Description of in- and outputs:
- PosReg: Register where the position is stored
- MinPos: Min allowed value
- MaxPos: Max allowed value

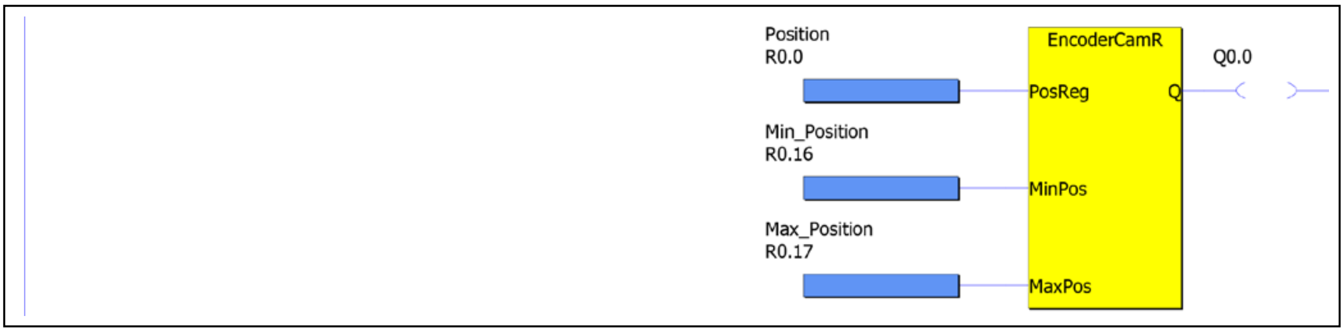


EncoderCamR

Function block for electronic cam limit switch where inputs MinPos and MaxPos also can be registers.

Function:
The output Q is set when the value in the input register PosReg is within the limits MinPos and MaxPos.
NOTE! It is possible to specify a cam which passes through the zero position of an encoder. Position<0 is forbidden.
Example: If MinPos=3000 and MaxPos=200, Q is set when the position is greater than 2999 or less than 201.

Description of in- and outputs:
- PosReg: Register for the encoder position.
- MinPos: Min/start value. Constant or register
- MaxPos: Max/stop value. Constant or register
(Only positive values are allowed)



External Comm

Ext_Sig

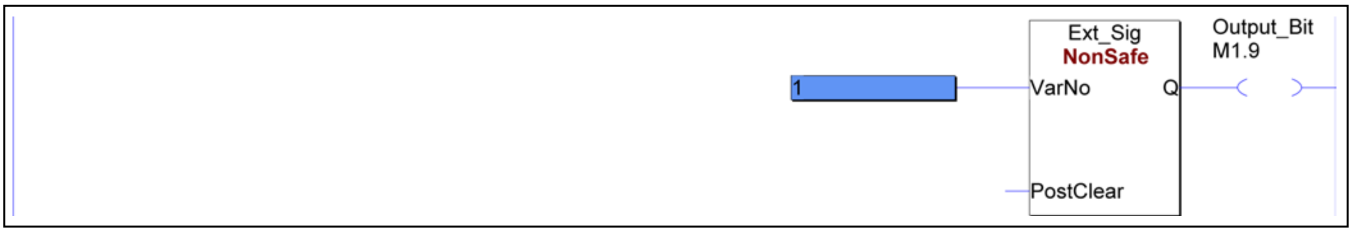
(Non-safe)

The block reads signals (1 bit) from external devices such as gateway, HMI etc.
The external devices can either be connected to the Pluto CAN-bus or the serial programming port.
By connection to the programming port Modbus ASCII communication protocol is used.

In- and outputs:
- VarNo is an input for a constant which specifies the variable number (0..63).
- Q is output and can control a M, GM or Q according to what the external device transmits.
- PostClear is an input for setting the output "Q" to 0 next cycle.
  If PostClear is not connected Q will keep its value until the external variable is set to 0.
  If time out is selected (under External Communication) Q will be set to 0 when the set time value has expired.

Description:
An external device can transmit up to 64 1-bit variables numbered 0..63. This block can take care of one of these variables and set a memory or output (M, GM or Q) with the output "Q" from the block.
For Modbus devices each "VarNo" corresponds to a Modbus "Output coil":
VarNo: 0 to Output coil 0,
VarNo: 1 to Output coil 1,
...
VarNo: 63 to Output coil 63.
So if for example a HMI controls output coil 4 the constant to input "VarNo" shall be set to "4".

PostClear sets the output "Q" to 0 in next PLC cycle, if it is unconditionally set "Q" will be set to "1" during one cycle if the external device writes "1" to the variable (output coil). If time out is selected (under External Communication) Q will be set to 0 when the set time value has expired.



Ext_Val

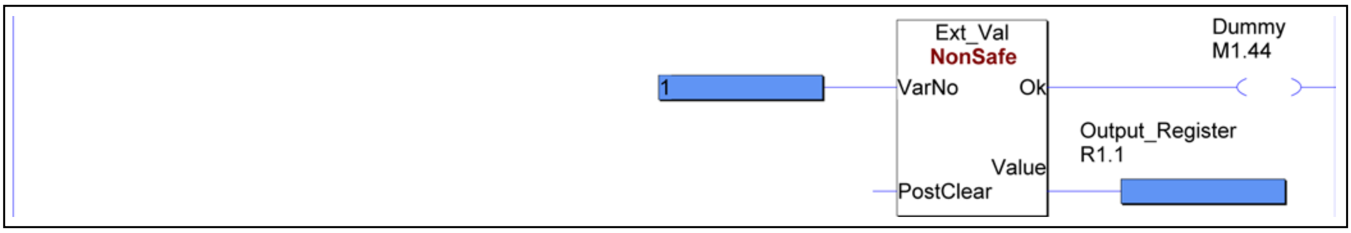
(Non-safe)

The block reads values (16 bit) from external devices such as gateways, HMI etc.
The external devices can either be connected to the Pluto CAN-bus or the serial programming port.
By connection to the programming port Modbus ASCII communication protocol is used.

In- and outputs:
- VarNo is an input for a constant which specifies the variable number (0..7).
- Value is the output for the value from the external device and must be connected to a Register (R).
- PostClear is an input for setting the output "Value" to 0 in next PLC cycle.
  If PostClear is not connected the output "Value" is frozen until it is updated by the external variable.
  If time out is selected (under External Communication) "Value" will be set to 0 when the set time value has expired.
- Q is mainly a dummy output which is normally 1. However it must be connected to a M, GM or Q.

Description:
An external device can transmit up to 8 16-bit variables numbered 0..7. This block can take care of one of these variables and write it to a register (R) with the output "Value" from the block.
For Modbus devices each "VarNo" corresponds to a Modbus "Holding register":
 VarNo: 0 to Holding reg. 400000,
 VarNo: 1 to Holding reg. 400001,
 ...
 VarNo: 7 to Holding reg. 400007.
So if for example a HMI controls output coil HREG 400002 the constant to input "VarNo" shall be set to "2".

PostClear sets the output "Value" to 0 in next PLC cycle. If the input is unconditionally set, "Value" will be set to the value written by the external device during one cycle. Then it will be reset to "0" again. If time out is selected (under External Communication) "Value" will be set to 0 when the set time value has expired.



ExtVarBlock

(Non-safe)

The block reads the variables from external devices such as gateways, HMI etc. and links them to the PLC code.
The external devices can either be connected to the Pluto CAN-bus or the serial programming port. By connection to the programming port Modbus ASCII communication protocol is used.

In- and outputs:

- BlockNo is an input for a constant specifying which "External comm blocks" (0..3) that are used.
- Bit_0...Bit_15 are outputs and can control M, GM or Q according to what the external device transmits.
- Reg_0 and Reg_1 are outputs for the registers from the external device and must be connected to Register (R).
- OK is mainly a dummy output which is normally 1. However it must be connected to a M, GM or Q.

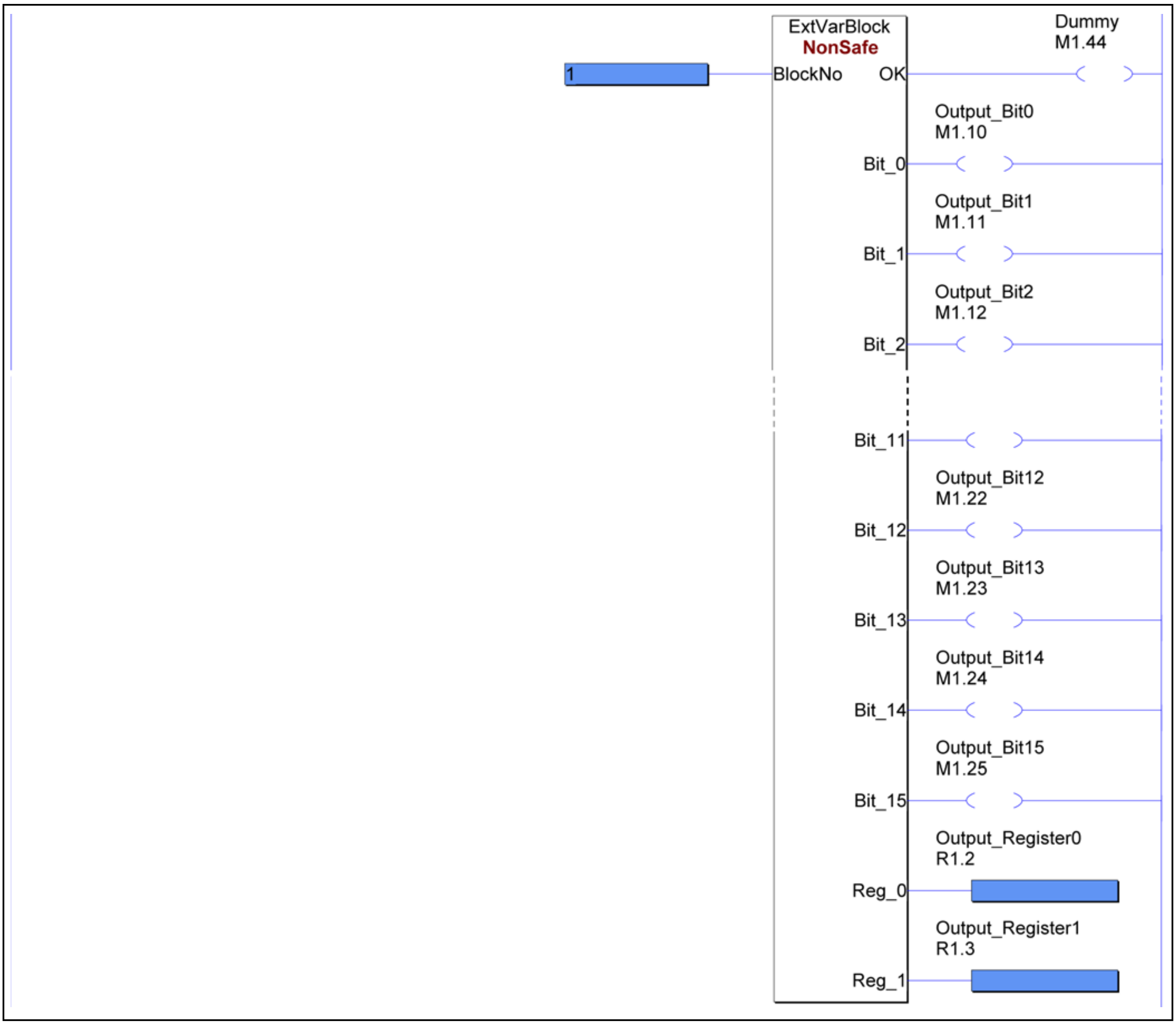
Description:

Pluto can receive up to 64 1-bit variables and 8 registers from external devices.
The data is received into four "External comm blocks". These four blocks can receive data from different sources which means that it is possible for each Pluto to receive data from four different external devices at the same time.
ExtVarBlock reads all the variables in one of the "External comm blocks" which are enabled for each Pluto under the button "External Communication".
Each block has 16 1-bit variables and 2 registers.
Timeout: If the connection with the external device is lost the variables are set to 0 after the timeout time has elapsed. The timeout is set on the page "External Communication"

For Modbus devices:

Bit_0...Bit_15 corresponds to a Modbus "Output coil":
BlockNo: 0, Bit_0 to Output coil 0,
BlockNo: 0, Bit_1 to Output coil 1,
...
BlockNo: 1, Bit_0 to Output coil 16,
...
BlockNo: 3, Bit_15 to Output coil 63,

Reg_0 and Reg_1 corresponds to a Modbus "Holding register":
BlockNo: 0, Reg_0 to Holding reg. 400000,
BlockNo: 0, Reg_1 to Holding reg. 400001,
BlockNo: 1, Reg_0 to Holding reg. 400002,
...
BlockNo: 3, Reg_1 to Holding reg. 400007.


ExtVarBlock
NonSafe
1
BlockNo
OK
Dummy
M1.44
Output_Bit0
M1.10
Bit_0
Output_Bit1
M1.11
Bit_1
Output_Bit2
M1.12
Bit_2
Bit_11
Output_Bit12
M1.22
Bit_12
Output_Bit13
M1.23
Bit_13
Output_Bit14
M1.24
Bit_14
Output_Bit15
M1.25
Bit_15
Output_Register0
R1.2
Reg_0
Output_Register1
R1.3
Reg_1

ToGateway_User_A

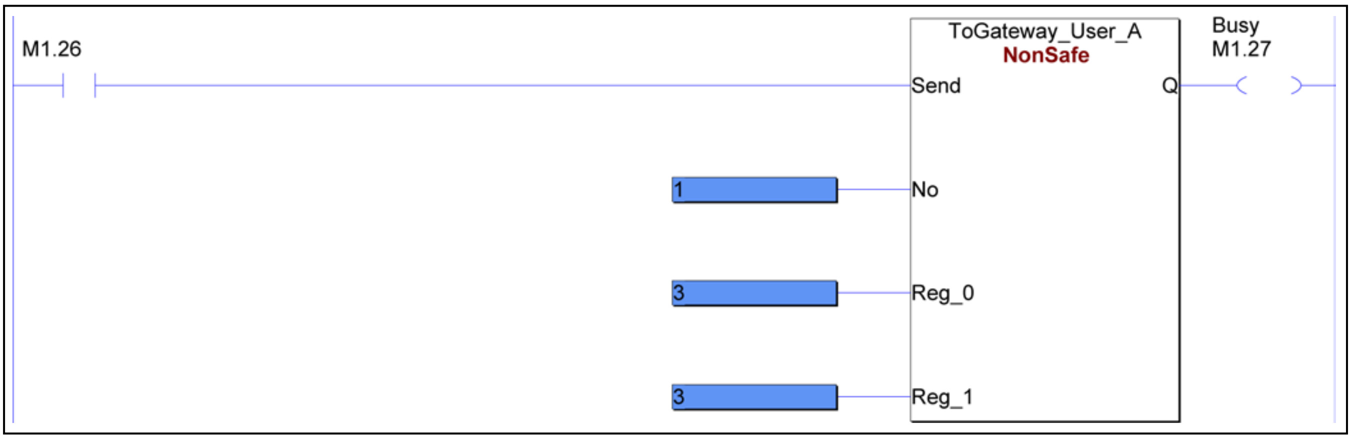
(Non-safe)

Send to gateway: 2 registers free of choice.

The block sends the variables which are connected to the block inputs.

- Reg_0 input for a register (R, SR)
- Reg_1 input for a register (R, SR)
- Send: Enable send. Data is transmitted when input is "1"
- Q: Is "1" during transmission.

Byte 0 = Reg_0 low byte
Byte 1 = Reg_0 high byte
Byte 2 = Reg_1 low byte
Byte 3 = Reg_1 high byte



ToGateway_User_B

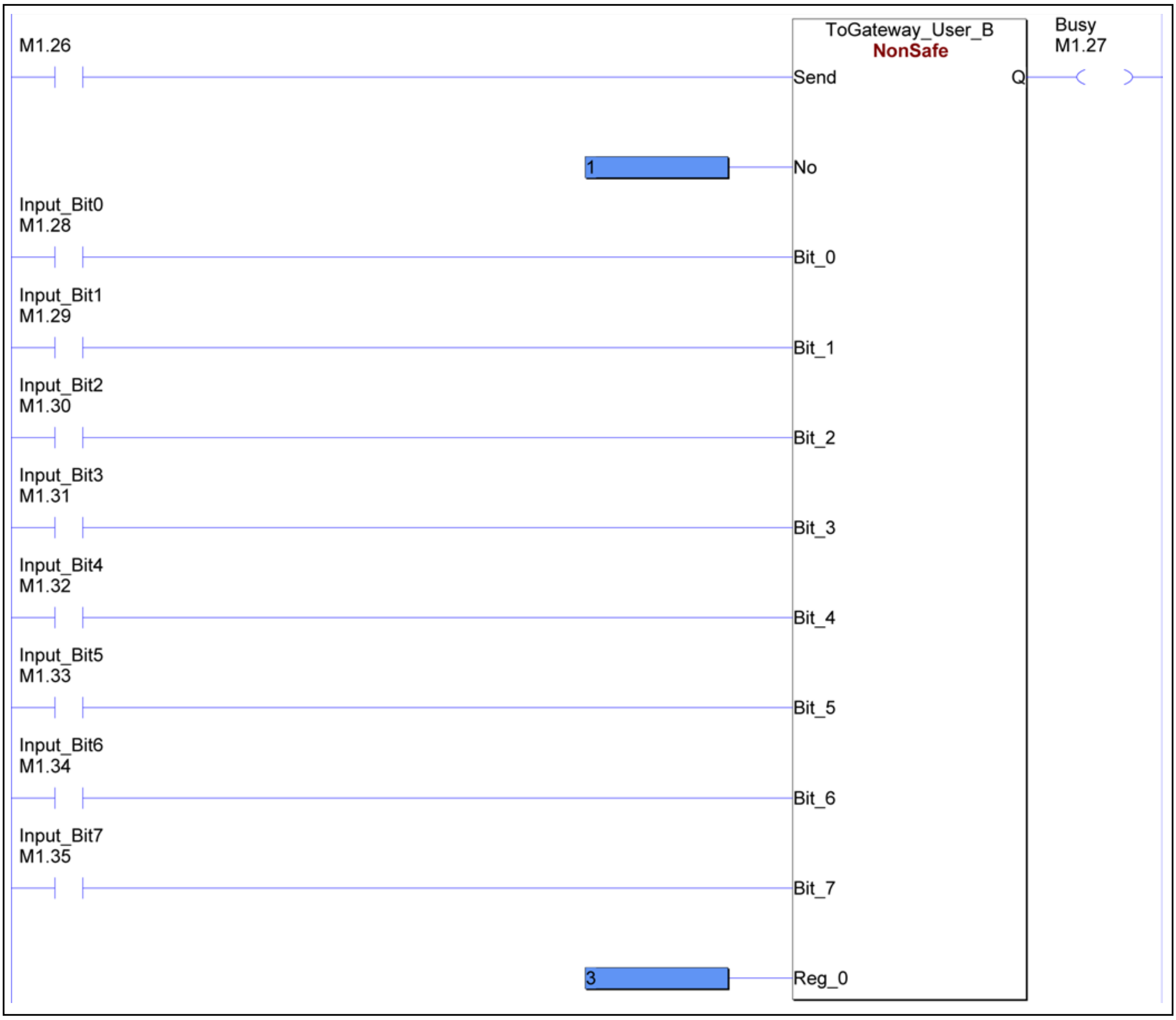
(Non-safe)

Send to gateway: 8 bits and 1 register free of choice and Error code.

The block sends the variables which are connected to the block inputs.

- Bit_0...Bit_7 inputs for bit variables (I, Q, M, GM, SM)
- Reg_0 input for a register (R, SR)
- Send: Enable send. Data is transmitted when input is "1"
- Q: Is "1" during transmission.

Byte 0 = Reg_0 low byte
Byte 1 = Reg_0 high byte
Byte 2 = Bit_7 Bit_6 Bit_5 Bit_4 Bit_3 Bit_2 Bit_1 Bit_0 (LSB)
Byte 3 = Error code



ToGateway_User_C

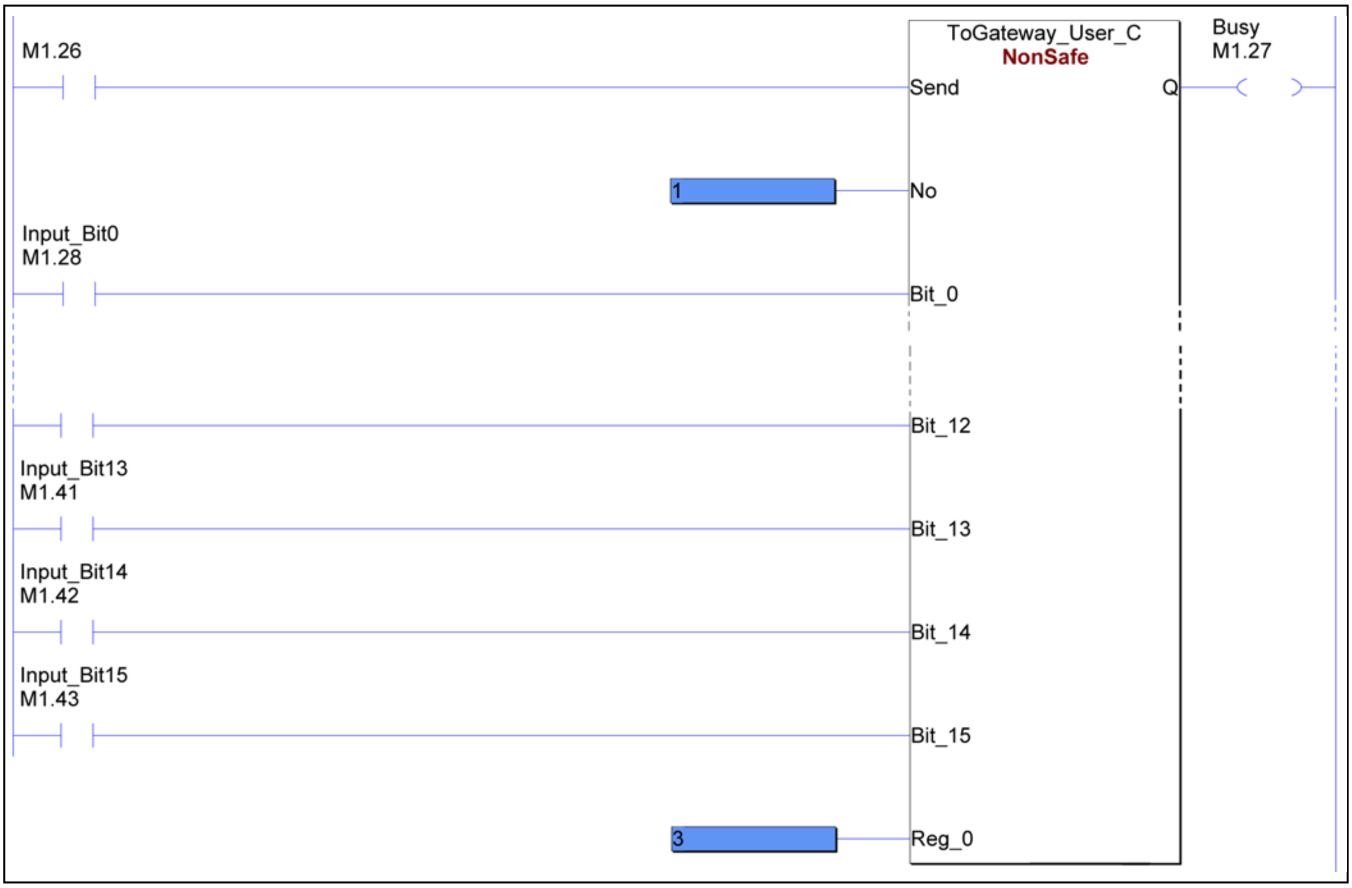
(Non-safe)

Send to gateway: 16 bits and 1 register.

The block sends the variables which are connected to the block.

- Bit_0...Bit_15 inputs for bit variables as inputs, outputs, memories (I, Q, M, GM, SM)
- Reg_0 input for a register
- Send: Enable send. Data is transmitted when input is "1"
- Q: Is "1" during transmission.

Byte 0 = Reg_0 low byte
Byte 1 = Reg_0 high byte
Byte 2 = Bit_7 Bit_6 Bit_5 Bit_4 Bit_3 Bit_2 Bit_1 Bit_0 (LSB)
Byte 3 = Bit_15 Bit_14 Bit_13 Bit_12 Bit_11 Bit_10 Bit_9 Bit_8 (LSB)



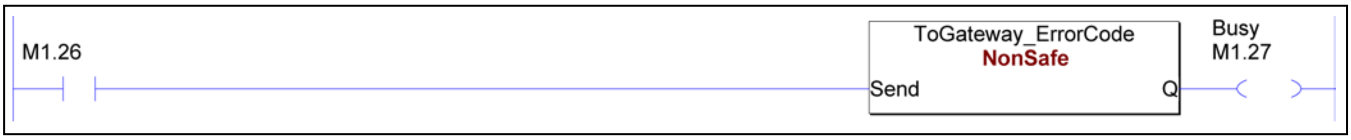
ToGateway_ErrorCode

(Non-safe)

Send error code to gateway.

- Send: Enable send. Data is transmitted when input is "1"
- Q: Is "1" during transmission.

Byte 0 = -
Byte 1 = -
Byte 2 = -
Byte 3 = Error code



ToGateway_B46_I20_I47

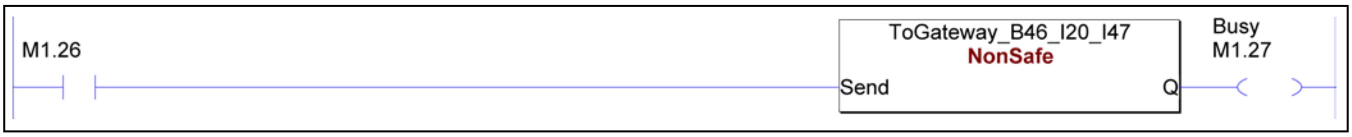
(Non-safe)

Send to gateway inputs I20..I27, I30..I37, I40..I47 and Error code.

Only for B46/S46

- Send: Enable send. Data is transmitted when input is "1"
- Q: Is "1" during transmision.

Byte 0 = I20..I27:	I27	I26	I25	I24	I23	I22	I21	I20	(LSB)
Byte 1 = I30..I37:	I37	I36	I35	I34	I33	I33	I31	I30	(LSB)
Byte 2 = I40..I47:	I47	I46	I45	I44	I43	I44	I41	I40	(LSB)
Byte 3 = Error code									



ToGateway_D45_I20_I47

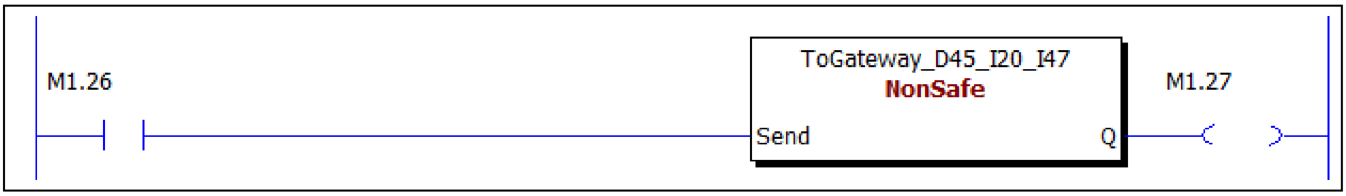
(Non-safe)

Send to gateway inputs I20..I26, I30..I37, I40..I47 and Error code.

Only for D45

- Send: Enable send. Data is transmitted when input is "1"
- Q: Is "1" during transmision.

Byte 0 = I20..I26: I26 I25 I24 I23 I22 I21 I20 (LSB)
Byte 1 = I30..I37: I37 I36 I35 I34 I33 I33 I31 I30 (LSB)
Byte 2 = I40..I47: I47 I46 I45 I44 I43 I44 I41 I40 (LSB)
Byte 3 = Error code



ToGateway_B42_I20_I47

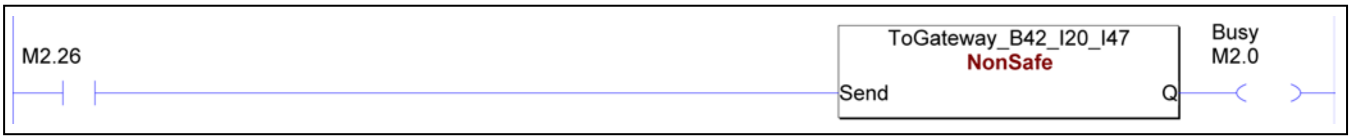
(Non-safe)

Send to gateway inputs I20..I27, I30..I37, I40..I47 and Error code.

Only for B42 AS-i

- Send: Enable send. Data is transmitted when input is "1"
- Q: Is "1" during transmision.

Byte 0 = I20..I27: I27 I26 I25 I24 I23 I22 I21 I20 (LSB)
Byte 1 = I30..I37: I37 I36 I35 I34 I33 I33 I31 I30 (LSB)
Byte 2 = I40..I47: I47 I46 I45 I44 I43 I44 I41 I40 (LSB)
Byte 3 = Error code



ToGateway_ASi_1_15_Safe

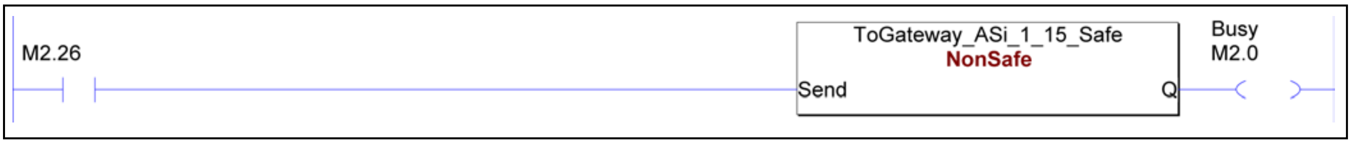
(Non-safe)

Send to gateway: Inputs of AS-i safe slaves 1...15.

Only for Pluto B42 AS-i

- Send: Enable send. Data is transmitted when input is "1"
- Q: Is "1" during transmision.

Byte 0 = Inputs:	I17	I16	I15	I14	I13	I12	I11	I10	(LSB)
Byte 1 = Asi1..Asi7:	Asi7	Asi6	Asi5	Asi4	Asi3	Asi2	ASi1	...	(LSB)
Byte 2 = Asi8..Asi15:	Asi15	Asi14	Asi13	Asi12	Asi11	Asi10	Asi9	Asi8	(LSB)
Byte 3 = Error code									



ToGateway_ASi_16_31_Safe

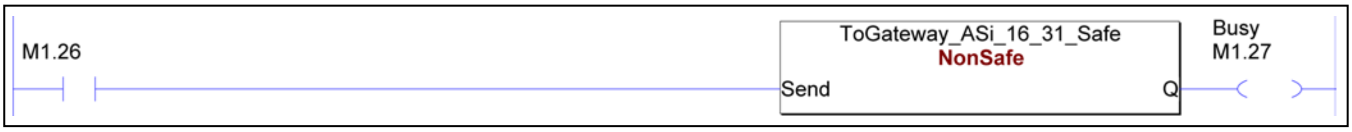
(Non-safe)

Send to gateway: Inputs in Pluto I1...I3, I10...13 and inputs of AS-i safe slaves 16...31.

Only for Pluto AS-i

- Send: Enable send. Data is transmitted when input is "1"
- Q: Is "1" during transmision.

Byte 0 = Inputs:	I13	I12	I11	I10	I3	I2	I1	--	(LSB)
Byte 1 = ASi16..ASi23:	ASi23	ASi22	ASi21	ASi20	ASi19	ASi18	ASi17	ASi16	(LSB)
Byte 2 = ASi24..ASi31:	ASi31	ASi30	ASi29	ASi28	ASi27	ASi26	ASi25	ASi24	(LSB)
Byte 3 = Error code									



ToGateway_ASi_1_3_NonSafe_In

(Non-safe)

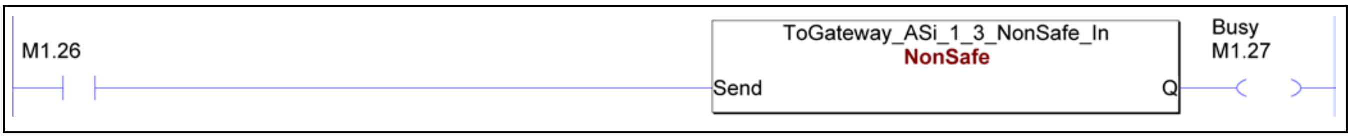
Send ASi inputs of nonsafe slaves 1..3 to gateway.

Only for Pluto AS-i

- Send: Enable send. Data is transmitted when input is "1"
- Q: Is "1" during transmission.

Byte 0 = Reserved
Byte 1 = Inputs AS-i slave 1 (A/B): 1B4 1B3 1B2 1B1 1A4 1A3 1A2 1A1 (LSB)
Byte 2 = Inputs AS-i slave 2 (A/B): 2B4 2B3 2B2 2B1 2A4 2A3 2A2 2A1 (LSB)
Byte 3 = Inputs AS-i slave 3 (A/B): 3B4 3B3 3B2 3B1 3A4 3A3 3A2 3A1 (LSB)

Note: Standard slaves and A-slaves are stored in same position



ToGateway_ASi_4_7_Nonsafe_In

(Non-safe)

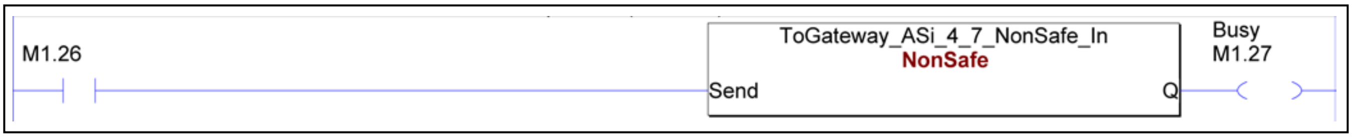
Send ASi inputs of nonsafe slaves 4..7 to gateway.

Only for Pluto AS-i

- Send: Enable send. Data is transmitted when input is "1"
- Q: Is "1" during transmission.

Byte 0 = Inputs AS-i slave 4 (A/B): 4B4 4B3 4B2 4B1 4A4 4A3 4A2 4A1 (LSB)
Byte 1 = Inputs AS-i slave 5 (A/B): 5B4 5B3 5B2 5B1 5A4 5A3 5A2 5A1 (LSB)
Byte 2 = Inputs AS-i slave 6 (A/B): 6B4 6B3 6B2 6B1 6A4 6A3 6A2 6A1 (LSB)
Byte 3 = Inputs AS-i slave 7 (A/B): 7B4 7B3 7B2 7B1 7A4 7A3 7A2 7A1 (LSB)

Note: Standard slaves and A-slaves are stored in same position (low nibble).



ToGateway_ASi_8_11_NonSafe_In

(Non-safe)

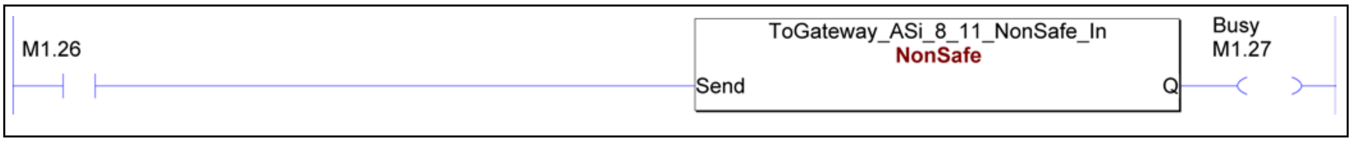
Send ASi inputs of nonsafe slaves 8..11 to gateway.

Only for Pluto AS-i

- Send: Enable send. Data is transmitted when input is "1"
- Q: Is "1" during transmission.

Byte 0 = Inputs AS-i slave 8 (A/B):	8B4	8B3	8B2	8B1	8A4	8A3	8A2	8A1 (LSB)
Byte 1 = Inputs AS-i slave 9 (A/B):	9B4	9B3	9B2	9B1	9A4	9A3	9A2	9A1 (LSB)
Byte 2 = Inputs AS-i slave 10 (A/B):	10B4	10B3	10B2	10B1	10A4	10A3	10A2	10A1 (LSB)
Byte 3 = Inputs AS-i slave 11 (A/B):	11B4	11B3	11B2	11B1	11A4	11A3	11A2	11A1 (LSB)

Note: Standard slaves and A-slaves are stored in same position (low nibble).



ToGateway_ASi_12_15_NonSafe_In

(Non-safe)

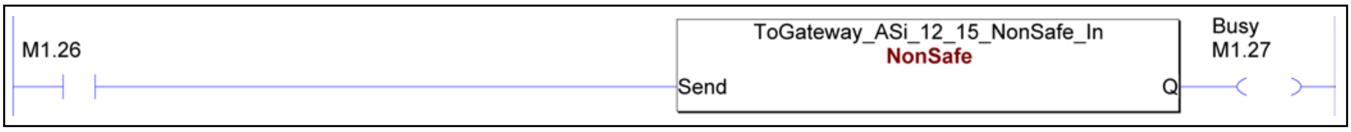
Send ASi inputs of nonsafe slaves 12..15 to gateway.

Only for Pluto AS-i

- Send: Enable send. Data is transmitted when input is "1"
- Q: Is "1" during transmission.

Byte 0 = Inputs AS-i slave 12 (A/B):	12B4	12B3	12B2	12B1	12A4	12A3	12A2	12A1 (LSB)
Byte 1 = Inputs AS-i slave 13 (A/B):	13B4	13B3	13B2	13B1	13A4	13A3	13A2	13A1 (LSB)
Byte 2 = Inputs AS-i slave 14 (A/B):	14B4	14B3	14B2	14B1	14A4	14A3	14A2	14A1 (LSB)
Byte 3 = Inputs AS-i slave 15 (A/B):	15B4	15B3	15B2	15B1	15A4	15A3	15A2	15A1 (LSB)

Note: Standard slaves and A-slaves are stored in same position (low nibble).



ToGateway_ASi_16_19_NonSafe_In

(Non-safe)

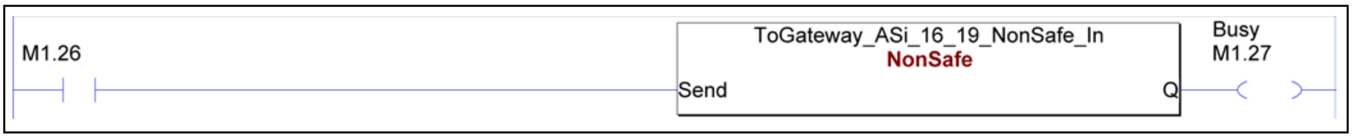
Send ASi inputs of nonsafe slaves 16..19 to gateway.

Only for Pluto AS-i

- Send: Enable send. Data is transmitted when input is "1"
- Q: Is "1" during transmission.

Byte 0 = Inputs AS-i slave 16 (A/B): 16B4 16B3 16B2 16B1 16A4 16A3 16A2 16A1 (LSB)
Byte 1 = Inputs AS-i slave 17 (A/B): 17B4 17B3 17B2 17B1 17A4 17A3 17A2 17A1 (LSB)
Byte 2 = Inputs AS-i slave 18 (A/B): 18B4 18B3 18B2 18B1 18A4 18A3 18A2 18A1 (LSB)
Byte 3 = Inputs AS-i slave 19 (A/B): 19B4 19B3 19B2 19B1 19A4 19A3 19A2 19A1 (LSB)

Note: Standard slaves and A-slaves are stored in same position (low nibble).



ToGateway_ASi_20_23_NonSafe_In

(Non-safe)

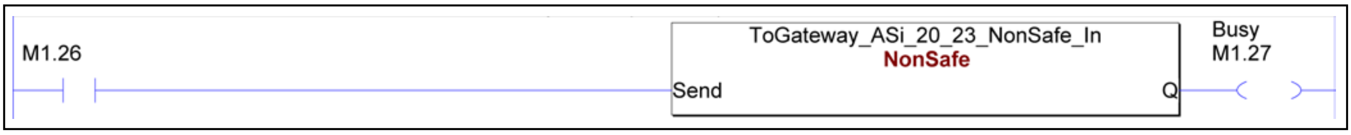
Send ASi inputs of nonsafe slaves 20..23 to gateway.

Only for Pluto AS-i

- Send: Enable send. Data is transmitted when input is "1"
- Q: Is "1" during transmission.

Byte 0 = Inputs AS-i slave 20 (A/B): 20B4 20B3 20B2 20B1 20A4 20A3 20A2 20A1 (LSB)
Byte 1 = Inputs AS-i slave 21 (A/B): 21B4 21B3 21B2 21B1 21A4 21A3 21A2 21A1 (LSB)
Byte 2 = Inputs AS-i slave 22 (A/B): 22B4 22B3 22B2 22B1 22A4 22A3 22A2 22A1 (LSB)
Byte 3 = Inputs AS-i slave 23 (A/B): 23B4 23B3 23B2 23B1 23A4 23A3 23A2 23A1 (LSB)

Note: Standard slaves and A-slaves are stored in same position (low nibble).



ToGateway_ASi_24_27_NonSafe_In

(Non-safe)

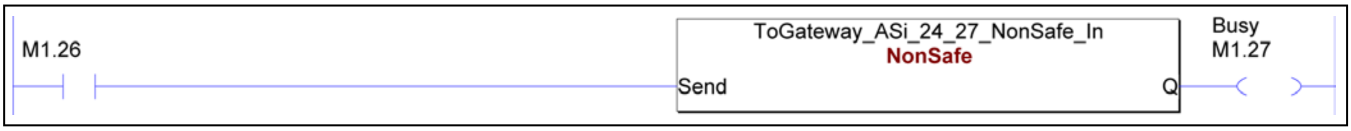
Send ASi inputs of nonsafe slaves 24..27 to gateway.

Only for Pluto AS-i

- Send: Enable send. Data is transmitted when input is "1"
- Q: Is "1" during transmission.

Byte 0 = Inputs AS-i slave 24 (A/B):	24B4	24B3	24B2	24B1	24A4	24A3	24A2	24A1 (LSB)
Byte 1 = Inputs AS-i slave 25 (A/B):	25B4	25B3	25B2	25B5	25A4	25A3	25A2	25A1 (LSB)
Byte 2 = Inputs AS-i slave 26 (A/B):	26B4	26B3	26B2	26B1	26A4	26A3	26A2	26A1 (LSB)
Byte 3 = Inputs AS-i slave 27 (A/B):	27B4	27B3	27B2	27B1	27A4	27A3	27A2	27A1 (LSB)

Note: Standard slaves and A-slaves are stored in same position (low nibble).



ToGateway_ASi_28_31_NonSafe_In

(Non-safe)

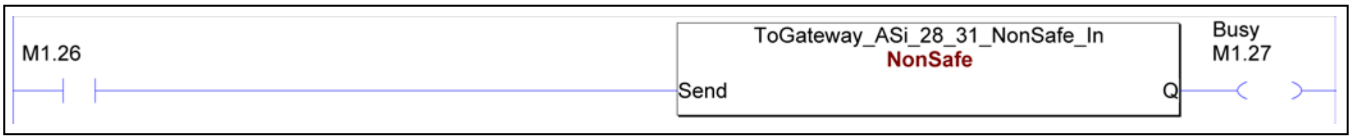
Send ASi inputs of nonsafe slaves 28..31 to gateway.

Only for Pluto AS-i

- Send: Enable send. Data is transmitted when input is "1"
- Q: Is "1" during transmission.

Byte 0 = Inputs AS-i slave 28 (A/B):	28B4	28B3	28B2	28B1	28A4	28A3	28A2	28A1 (LSB)
Byte 1 = Inputs AS-i slave 29 (A/B):	29B4	29B3	29B2	29B1	29A4	29A3	29A2	29A1 (LSB)
Byte 2 = Inputs AS-i slave 30 (A/B):	30B4	30B3	30B2	30B1	30A4	30A3	30A2	30A1 (LSB)
Byte 3 = Inputs AS-i slave 31 (A/B):	31B4	31B3	31B2	31B1	31A4	31A3	31A2	31A1 (LSB)

Note: Standard slaves and A-slaves are stored in same position (low nibble).



StatusBus

StatusBusOn1

(Non-safe)

Read ON/OFF status for slaves 1-15.

Description of in and outputs:

- IQ: Constant for selection of StatusBus IQ, must be 10-13. (10=IQ10, 11=IQ11, …)

- Slave1: Set if StatusBus slave 1 is ON.

  ..

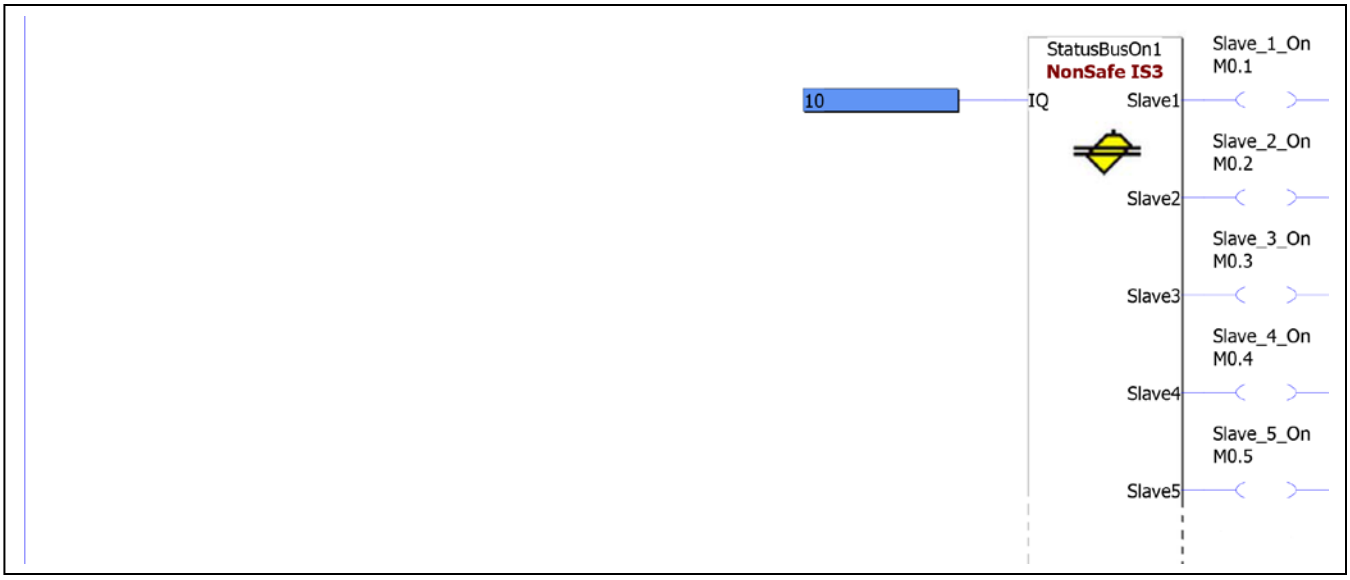
- Slave15: Set if StatusBus slave 15 is ON.

Description:

StatusBus is one wire (pin 5 in M12 connector) running beside the dynamic safety signal, connected to one of IQ10..13. It gives non-safe ON/OFF status for each individual safety component in a dynamic circuit.

Which of IQ10..13 that shall be read is selected with the block input “IQ”.
The outputs “Slave1…Slave15” shows the status for the corresponding slave. 0=OFF and 1=ON.

Note: The intended function is to give the user information on displays or other kinds of HMI.
The information is non-safe and must not be used for safety related functions.
Since the response time is varying depending on jitter, number of slaves and bus speed, it is often not suitable for machine control functions.



StatusBusOn2

(Non-safe)

Read ON/OFF status for slaves 16-30.

Description of in and outputs:

- IQ: Constant for selection of StatusBus IQ, must be 10-13. (10=IQ10, 11=IQ11, …)

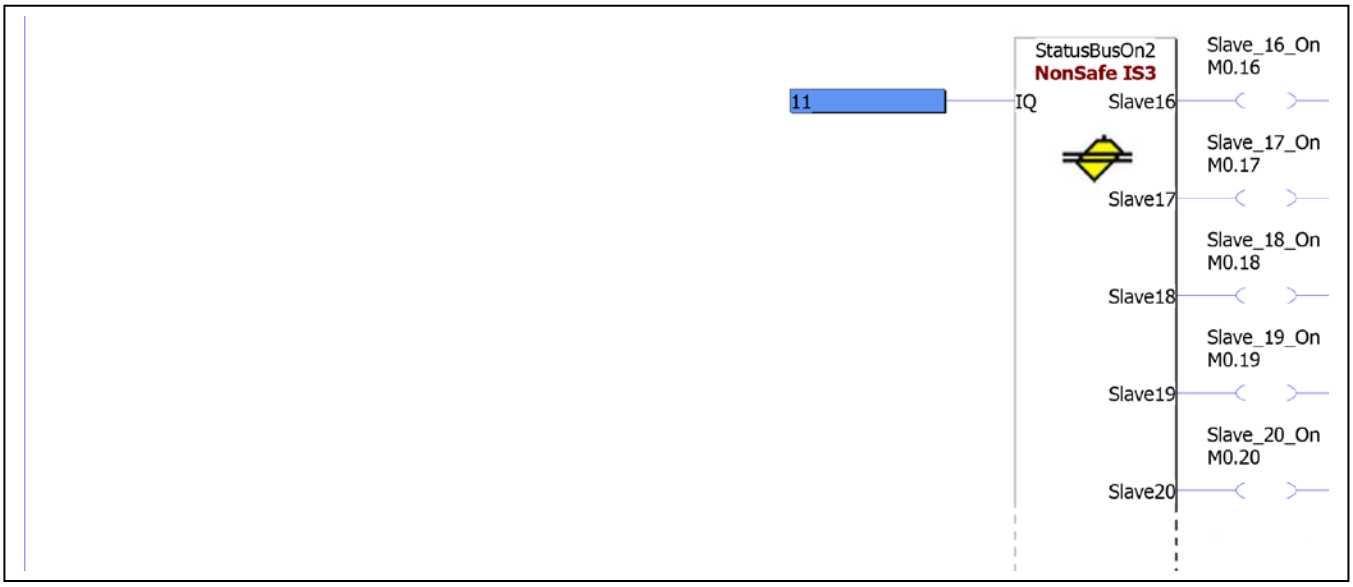
- Slave16: Set if StatusBus slave 16 is ON.
  ..
- Slave30: Set if StatusBus slave 30 is ON.

Description:

StatusBus is one wire (pin 5 in M12 connector) running beside the dynamic safety signal, connected to one of IQ10..13. It gives non-safe ON/OFF status for each individual safety component in a dynamic circuit.

Which of IQ10..13 that shall be read is selected with the block input "IQ".
The outputs "Slave16…Slave30" shows the status for the corresponding slave. 0=OFF and 1=ON.

Note: The intended function is to give the user information on displays or other kinds of HMI.
The information is non-safe and must not be used for safety related functions.
Since the response time is varying depending on jitter, number of slaves and bus speed, it is often not suitable for machine control functions.



StatusBusOnReg

(Non-safe)

ON/OFF status for all slaves to registers.

<u>Description of in and outputs:</u>

- IQ: Constant for selection of StatusBus IQ, must be 10-13. (10=IQ10, 11=IQ11, …)

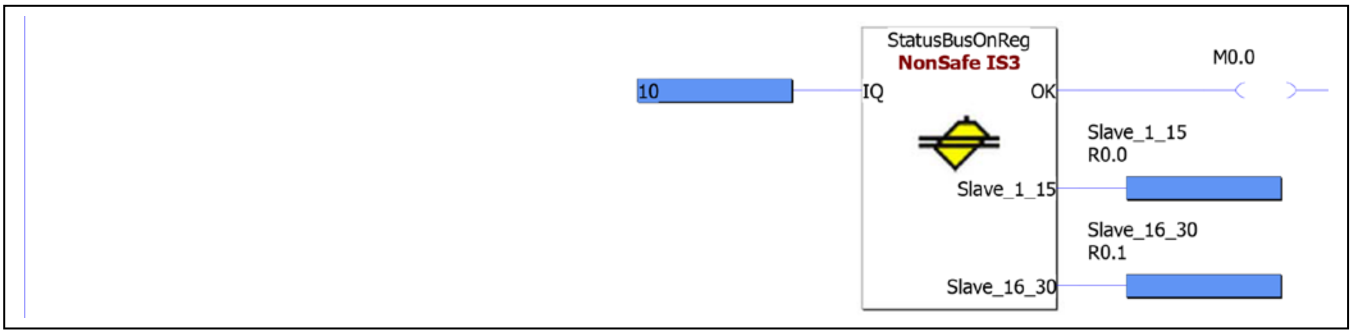
- Ok: Always set (Dummy)
- Slave_1_15: Output to register (R) for slave 1..15, one bit/slave.
  Example: If slave 1 is on then Slave_1_15.1 (bit 1) is on. (Bit 0 is empty)
- Slave_16_30: Output to register (R) for slave 16..30, one bit/slave.
  Example: If slave 16 is on then Slave_16_30.0 (bit 0) is on. (Bit 15 is empty)

<u>Description:</u>

StatusBus is one wire (pin 5 in M12 connector) running beside the dynamic safety signal, connected to one of IQ10..13. It gives non-safe ON/OFF status for each individual safety component in a dynamic circuit.

Which of IQ10..13 that shall be read is selected with the block input “IQ”.
The register outputs “Slave_1_15 and Slave_16_30” shows the status for the corresponding slaves. Each bit represents one slave, 0=OFF and 1=ON.

Note: The intended function is to give the user information on displays or other kinds of HMI.
The information is non-safe and must not be used for safety related functions.
Since the response time is varying depending on jitter, number of slaves and bus speed, it is often not suitable for machine control functions.



StatusBusSlave

(Non-safe)

StatusBus info for one slave.

Description of in and outputs:

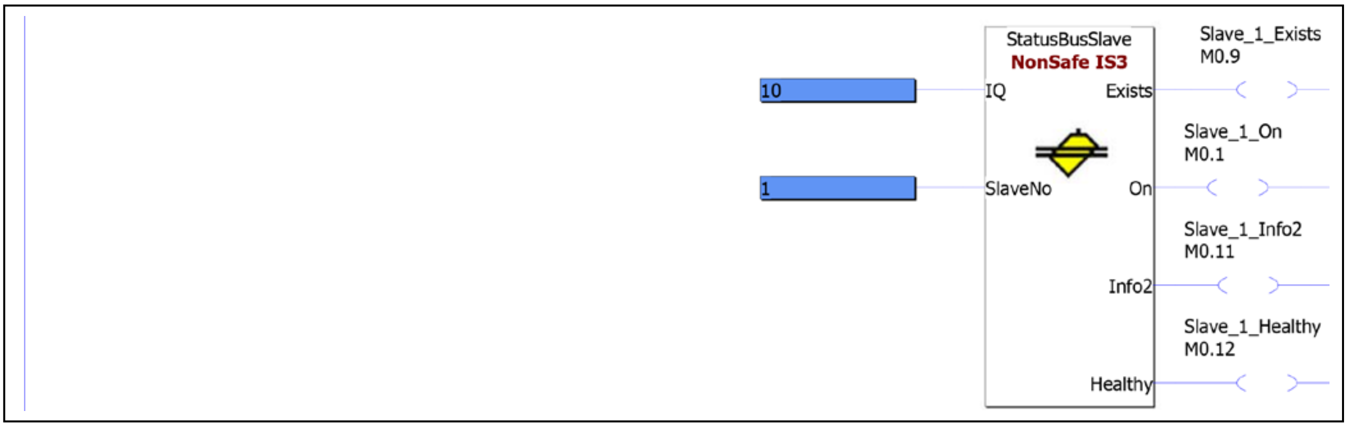
- IQ: Constant for selection of StatusBus IQ, must be 10-13. (10=IQ10, 11=IQ11, …)
- SlaveNo: Address of the slave, 1-30.

- Exists: Slave is present in the system and communication on StatusBus.
- On: Set when the slave safety is ON.
- Info2: Secondary info, dependent on slave type.
- Healthy: No problem for slave.

Description:
StatusBus is one wire (pin 5 in M12 connector) running beside the dynamic safety signal, connected to one of IQ10..13.
This block reads all information available for a slave, three bit data and if it is present on the bus.

With the block inputs “IQ” and “SlaveNo” the Pluto I/O (IQ10..13) and slave number is selected.
The output “On” shows the main status, if the safety function is OK/ON. 0=OFF and 1=ON.
“Info2” is secondary information individual for different slave types but in many cases not used.
The “Healthy” bit is a diagnostic bit, also individual for different slave types. Typical meaning can be low supply voltage on the slave. 1=OK/No problem.

Note: The intended function is to give the user information on displays or other kinds of HMI.
The information is non-safe and must not be used for safety related functions.
Since the response time is varying depending on jitter, number of slaves and bus speed, it is often not suitable for machine control functions.



Compatibility

TC1RTI

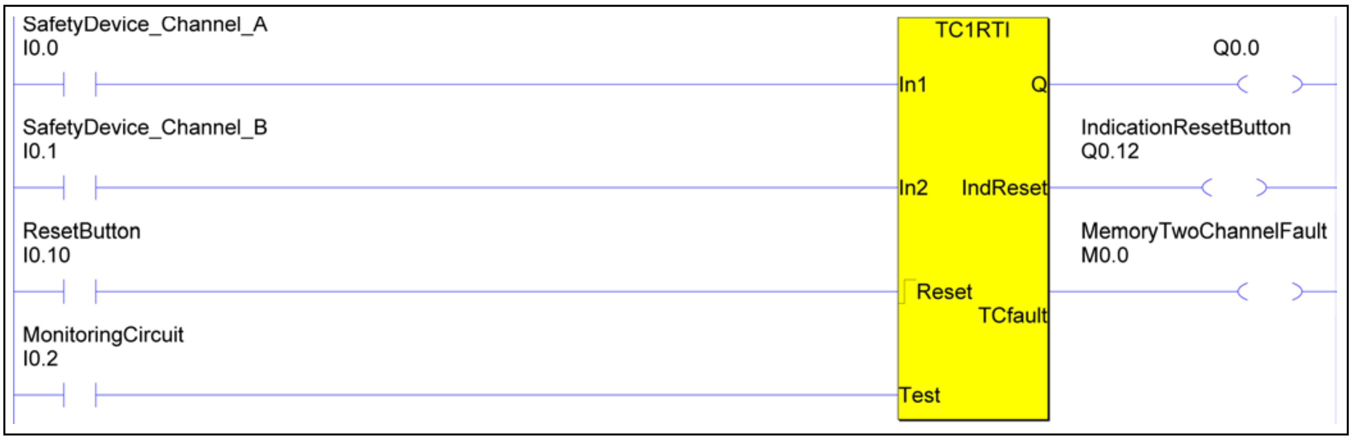
(Not recommended for new programs. Use TC2RTI instead.)

Two-channel function with test and reset inputs.

- In1 and In2 are primary inputs where safety devices can be connected.
- Test is an input condition which must be true at the start up moment and can be used for monitoring of external components. Test must switch on before In1 and In2, i.e. the function block cannot be initiated by Test.
- Reset is a monitored reset input which must be activated (positive edge) after all other inputs, in order to switch on the output of the function block.

- IndReset is an output for indication which is ON when the function block is switched off and In1, In2 or Test is off, and flashes when the function block is ready for activation with reset.
- TCfault is on by two channel fault, i.e. when the function block is on and just one of In1 and In2 switches off and on again.

Description:
The function block works as a conventional safety relay with redundant and monitored inputs (In1, In2).

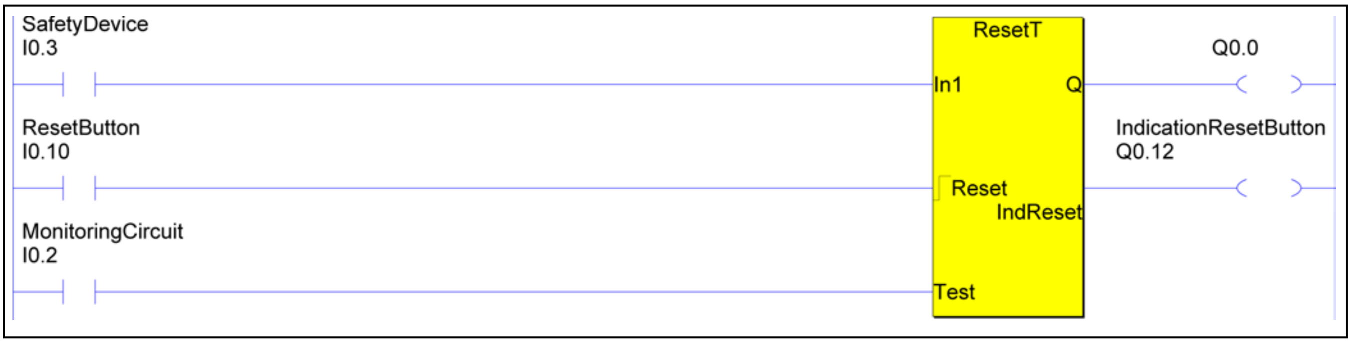


ResetT

(Not recommended for new programs. Use Reset2T instead.)

Single-channel function with reset and test input

- In1 is primary input to which the output of safety devices, output of other function blocks or memories can be connected.
- Test is an input condition which must be true at the start up moment and can be used for monitoring of external components. Test must switch on before Reset, i.e. the function block cannot be initiated by Test.
- Reset is a monitored reset input which must be activated (positive edge) after all other inputs in order to switch on the output of the function block.
- IndReset is an output for indication which is on when the function block is switched off and In1 and Test is off, and flashes when the function block is ready for activation with Start.



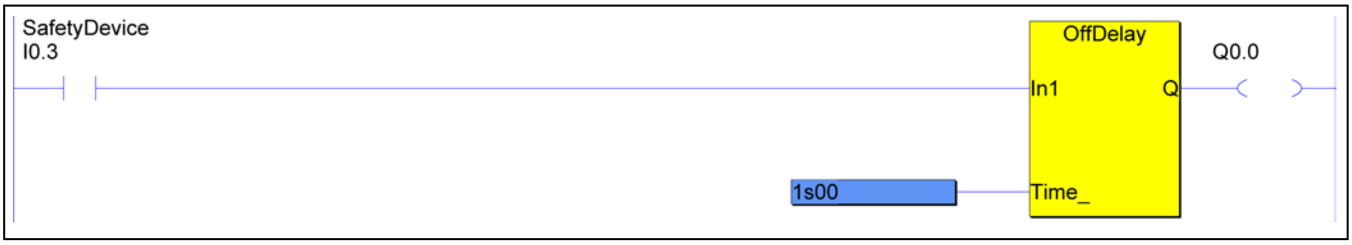
Instruction set 2

OffDelay

Off delay timer function

- In1 is input.
- Time_ is input for time constant.

Function:
The output Q switches on immediately when In1 switches on. When In1 switches off Q switches off after the time has elapsed. The time is specified by the time constant connected to the input “Time_”.



Multiply

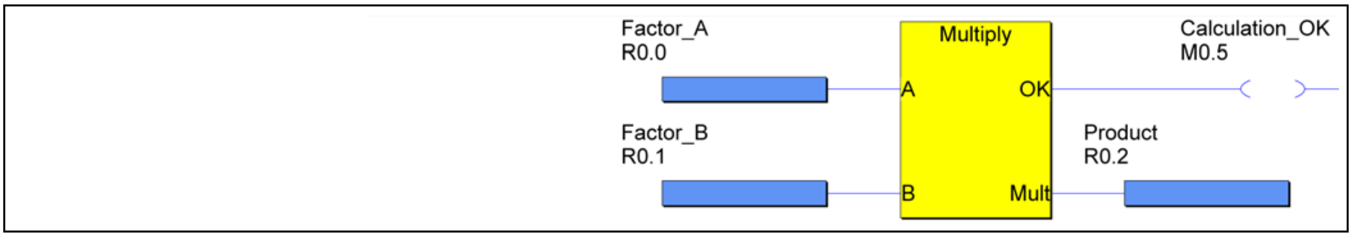
Integer multiplication

Mult = A*B

- A and B are inputs for the factors. They can either be registers or constants.
- Mult is the result of the calculation. It can be assigned to a register.
- OK is an output indicating that the multiplication was successful.

Overflow:
If the multiplication exceeds the limits -32768 and 32767 (overflow) the OK output switches off and the value of Mult is not relevant. It can be connected to a register (R). This means that the variable connected to OK should be used in combination with the output register later in the PLC code.

NOTE! Pluto system software ver 2.2.3 or higher neeeded.



Divide

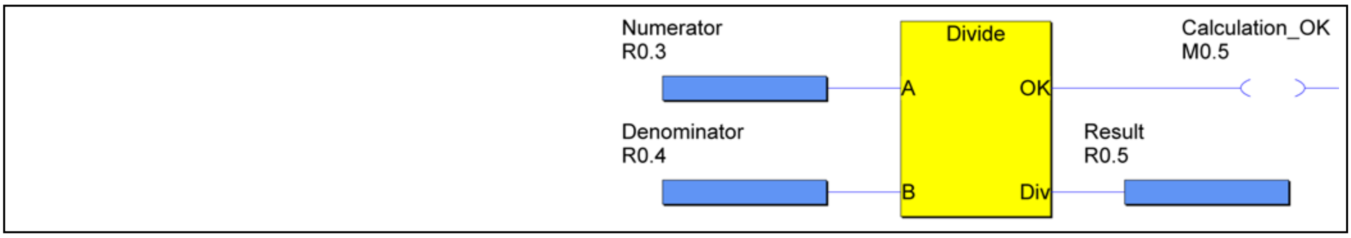
Integer division

Div = A/B

- A is inputs for the numerator. It can either be a register or a constant.
- B is inputs for the denominator. It can either be a register or a constant.
- Div is the result of the calculation. It can be assigned to a register.
- OK is an output indicating that the division was successful. It can be connected to variable types M, GM, Q.

Division with 0:
If B is 0 the calculation cannot be performed and is indicated by that the "OK" output is switched off. The output "Div" is then 0. This means that the variable connected to OK should be used in combination with the output register later in the PLC code.

NOTE! Pluto system software ver 2.2.3 or higher neeeded.



Reg_to_Binary

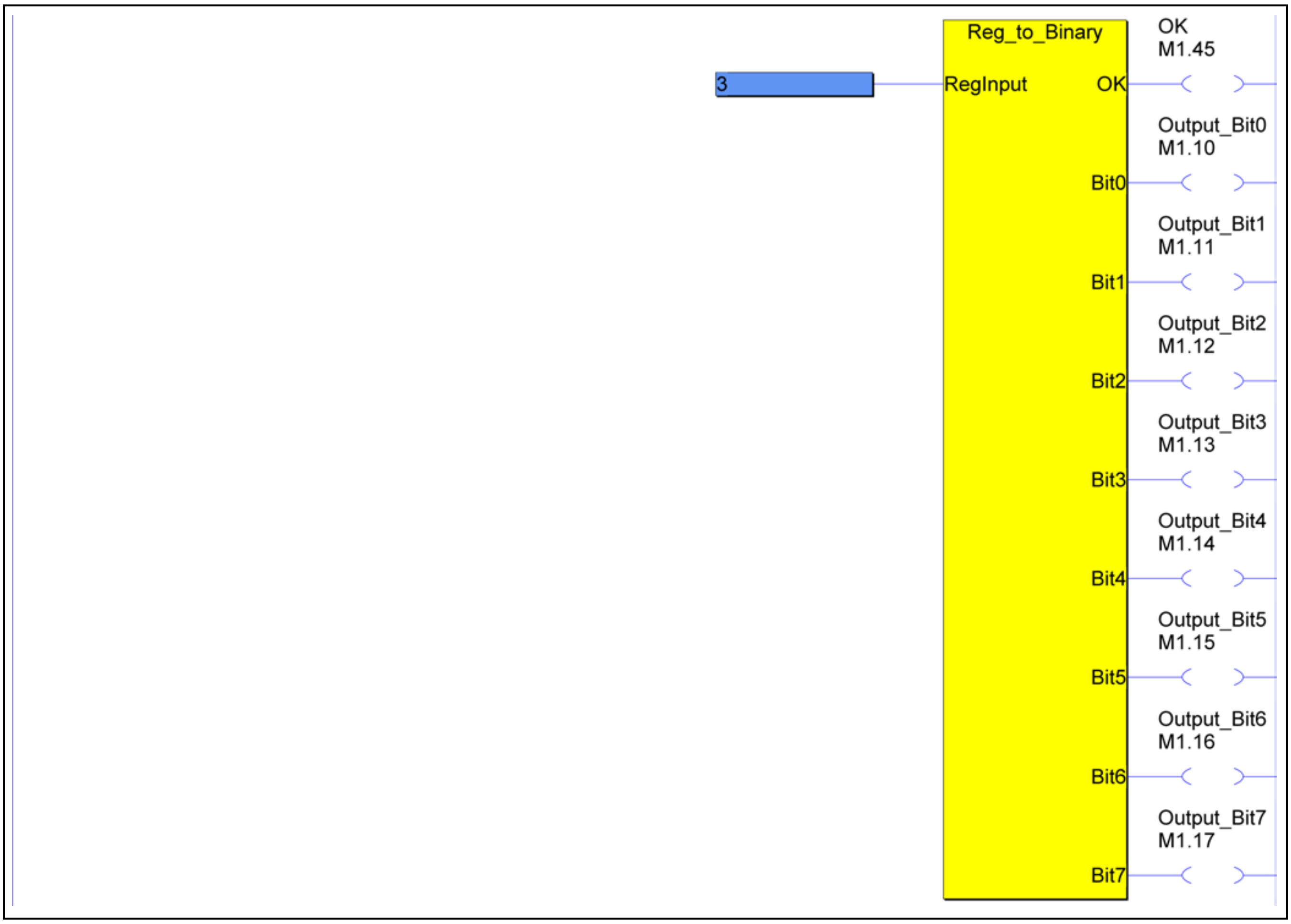
Conversion of a register to 8 output bits

- RegInput is the input value that shall be converted. It can be connected to a register or a constant.
- Bit0, Bit1, ... Bit7 are output bits for the result of the conversion.
- OK is on as long as the input value is within 0..255

Description:
The function block codes the 8 lowest bits (LSB) in a register to the 8 outputs Bit0..7

Examples:
RegInput = 1 gives Bit0 = 1, Bit1..7 = 0 (00000001)
RegInput = 2 gives Bit1 = 1, Bit0, Bit2..7 = 0 (00000010)
RegInput = 255 gives all of Bit0..7 = 1 (11111111)

Since the registers are 16 bit long the input value is limited to 0..255.
If the value of the input register is outside the limit the output OK and all of Bit0..7 are set to 0 (off).



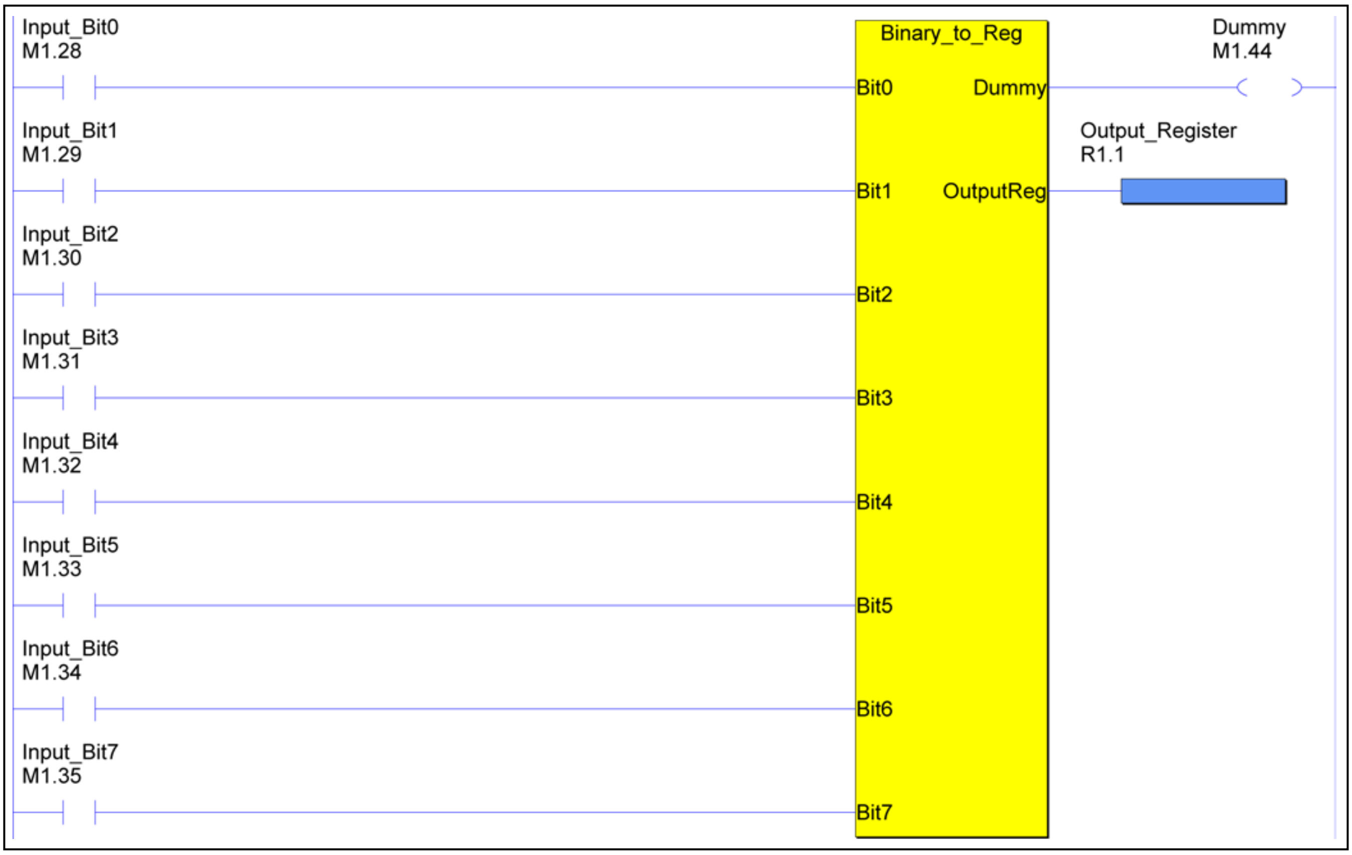
Binary_to_Reg

Conversion of 8 input bits to a decimal register

- Bit0..7 are the input bits which shall be converted to a register value.
- OutputReg is the output for the result of the conversion. It can be connected to a register.
- Q is a dummy output which is always 1 (on)

Description:
The block converts the inputs Bit0..7 from binary to a decimal value. The value is stored in the register to which the output OutputReg is connected.

Examples:
Bit0 = 1, Bit1..7 = 0 (00000001) gives OutputReg = 1
Bit1..2 = 1, Bit0, Bit3..7 = 0 (00000110) gives OutputReg = 6
All of Bit0..7 = 1 (11111111) gives OutputReg = 255



Binary16_to_Reg

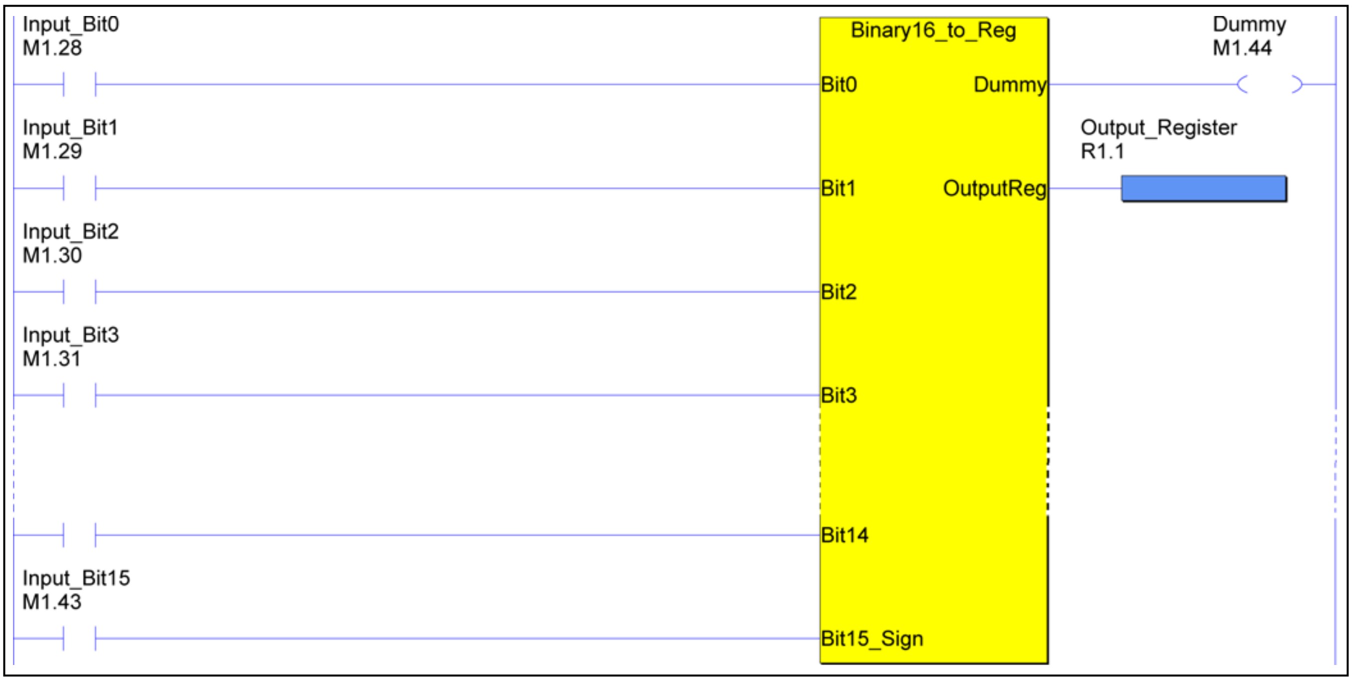
Conversion of 16 input bits to a decimal register

- Bit0..15 are the input bits which shall be converted to a register value -32768..+32767.
- OutputReg is the output for the result of the conversion. It can be connected to a register.
- Q is a dummy output which is always 1 (on).

Description:
The block converts the inputs Bit0..15 from binary to a decimal value. The value is stored in the register to which the output OutputReg is connected.

Note: The Pluto registers are of type "Signed 16 bit" which means that if bit 15 is set the output register will get a negative value.

Examples:
Bit0 = 1, Bit1..15 = 0 (00000000 00000001) gives OutputReg = 1
Bit1..2 = 1, Bit0, Bit3..15 = 0 (00000000 00000110) gives OutputReg = 6
Bit0..7 = 1, Bit8..15 = 0 (00000000 11111111) gives OutputReg = 255
Bit0..15 = 1, Bit8..15 = 0 (11111111 11111111) gives OutputReg = -1
Bit15 = 1, Bit0..14 = 0 (10000000 00000000) gives OutputReg = -32768


Input_Bit0
M1.28
Input_Bit1
M1.29
Input_Bit2
M1.30
Input_Bit3
M1.31
Input_Bit15
M1.43
Binary16_to_Reg
Bit0
Bit1
Bit2
Bit3
Bit14
Bit15_Sign
Dummy
OutputReg
Dummy
M1.44
Output_Register
R1.1