Modicon M221
Logic Controller
Programming Guide

04/2014

EIO0000001360.01

www.schneider-electric.com

Schneider Electric

The information provided in this documentation contains general descriptions and/or technical characteristics of the performance of the products contained herein. This documentation is not intended as a substitute for and is not to be used for determining suitability or reliability of these products for specific user applications. It is the duty of any such user or integrator to perform the appropriate and complete risk analysis, evaluation and testing of the products with respect to the relevant specific application or use thereof. Neither Schneider Electric nor any of its affiliates or subsidiaries shall be responsible or liable for misuse of the information contained herein. If you have any suggestions for improvements or amendments or have found errors in this publication, please notify us.

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

When devices are used for applications with technical safety requirements, the relevant instructions must be followed.

Failure to use Schneider Electric software or approved software with our hardware products may result in injury, harm, or improper operating results.

Failure to observe this information can result in injury or equipment damage.

Table of Contents

Safety Information

Important Information

NOTICE

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

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

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

⚠ DANGER

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

⚠ WARNING

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

⚠ CAUTION

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

NOTICE

NOTICE is used to address practices not related to physical injury.

PLEASE NOTE

Electrical equipment should be installed, operated, serviced, and maintained only by qualified personnel. No responsibility is assumed by Schneider Electric for any consequences arising out of the use of this material.

A qualified person is one who has skills and knowledge related to the construction and operation of electrical equipment and its installation, and has received safety training to recognize and avoid the hazards involved.

About the Book

At a Glance

Document Scope

This document describes the configuration and programming of the Modicon M221 Logic Controller for SoMachine Basic. For further information, refer to the separate documents provided in the SoMachine Basic online help.

Validity Note

This document has been updated with the release of SoMachine Basic V1.1.

Related Documents

Title of Documentation	Reference Number
SoMachine Basic - Operating Guide	EIO0000001354 (ENG) EIO0000001355 (FRA) EIO0000001356 (GER) EIO0000001357 (SPA) EIO0000001358 (ITA) EIO0000001359 (CHS) EIO0000001366 (POR) EIO0000001367 (TUR)
SoMachine Basic Generic Functions - Library Guide	EIO0000001474 (ENG) EIO0000001475 (FRA) EIO0000001476(GER) EIO0000001477 (SPA) EIO0000001478 (ITA) EIO0000001479 (CHS) EIO0000001480 (POR) EIO0000001481 (TUR)
Modicon M221 Logic Controller - Hardware Guide	EIO0000001384 (ENG) EIO0000001385 (FRA) EIO0000001386 (GER) EIO0000001387 (SPA) EIO0000001388 (ITA) EIO0000001389 (CHS) EIO0000001370 (POR) EIO0000001371 (TUR)

Title of Documentation	Reference Number
Modicon TMC2 Cartridge - Programming Guide	EIO0000001782 (ENG) EIO0000001783 (FRA) EIO0000001784 (GER) EIO0000001785 (SPA) EIO0000001786 (ITA) EIO0000001787 (CHS) EIO0000001788 (POR) EIO0000001789 (TUR)
Modicon TMC2 Cartridge - Hardware Guide	EIO0000001768 (ENG) EIO0000001769 (FRE) EIO0000001770 (GER) EIO0000001771 (SPA) EIO0000001772 (ITA) EIO0000001773 (CHS) EIO0000001775 (TUR) EIO0000001774 (POR)
Modicon TM3 Expansion Modules Configuration - Programming Guide	EIO0000001396 (ENG) EIO0000001397 (FRA) EIO0000001398 (GER) EIO0000001399 (SPA) EIO0000001400 (ITA) EIO0000001401 (CHS) EIO0000001374 (POR) EIO0000001375 (TUR)
Modicon TM3 Digital I/O Modules - Hardware Guide	EIO0000001408 (ENG) EIO0000001409 (FRA) EIO0000001410 (GER) EIO0000001411 (SPA) EIO0000001412 (ITA) EIO0000001413 (CHS) EIO0000001376 (POR) EIO0000001377 (TUR)
Modicon TM3 Analog I/O Modules - Hardware Guide	EIO0000001414 (ENG) EIO0000001415 (FRA) EIO0000001416 (GER) EIO0000001417 (SPA) EIO0000001418 (ITA) EIO0000001419 (CHS) EIO0000001378 (POR) EIO0000001379 (TUR)

Title of Documentation	Reference Number
Modicon TM3 Expert Modules - Hardware Guide	EIO0000001420 (ENG) EIO0000001421 (FRA) EIO0000001422 (GER) EIO0000001423 (SPA) EIO0000001424 (ITA) EIO0000001425 (CHS) EIO0000001380 (POR) EIO0000001381 (TUR)
Modicon TM3 Transmitter and Receiver Modules - Hardware Guide	EIO0000001426 (ENG) EIO0000001427 (FRA) EIO0000001428 (GER) EIO0000001429 (SPA) EIO0000001430 (ITA) EIO0000001431 (CHS) EIO0000001382 (POR) EIO0000001383 (TUR)
Modicon TM2 Expansion Modules Configuration - Programming Guide	EIO0000000396 (ENG) EIO0000000397 (FRE) EIO0000000398 (GER) EIO0000000399 (SPA) EIO0000000400 (ITA) EIO0000000401 (CHS)
Modicon TM2 Digital I/O Modules - Hardware Guide	EIO0000000028 (ENG) EIO0000000029 (FRA) EIO0000000030 (GER) EIO0000000031 (SPA) EIO0000000032 (ITA) EIO0000000033 (CHS)
Modicon TM2 Analog I/O Modules - Hardware Guide	EIO0000000034 (ENG) EIO0000000035 (FRA) EIO0000000036 (GER) EIO0000000037 (SPA) EIO0000000038 (ITA) EIO0000000039 (CHS)

You can download these technical publications and other technical information from our website at www.schneider-electric.com.

Product Related Information

⚠ WARNING

LOSS OF CONTROL

- The designer of any control scheme must consider the potential failure modes of control paths and, for certain critical control functions, provide a means to achieve a safe state during and after a path failure. Examples of critical control functions are emergency stop and overtravel stop, power outage and restart.
- Separate or redundant control paths must be provided for critical control functions.
- System control paths may include communication links. Consideration must be given to the implications of unanticipated transmission delays or failures of the link.
- Observe all accident prevention regulations and local safety guidelines.[1]
- Each implementation of this equipment must be individually and thoroughly tested for proper operation before being placed into service.

Failure to follow these instructions can result in death, serious injury, or equipment damage.

[1] For additional information, refer to NEMA ICS 1.1 (latest edition), "Safety Guidelines for the Application, Installation, and Maintenance of Solid State Control" and to NEMA ICS 7.1 (latest edition), "Safety Standards for Construction and Guide for Selection, Installation and Operation of Adjustable-Speed Drive Systems" or their equivalent governing your particular location.

⚠ WARNING

UNINTENDED EQUIPMENT OPERATION

- Only use software approved by Schneider Electric for use with this equipment.
- Update your application program every time you change the physical hardware configuration.

Failure to follow these instructions can result in death, serious injury, or equipment damage.

Part I
Introduction

Overview

This part provides general information about the Modicon M221 Logic Controller and its configuration and programming features.

What Is in This Part?

This part contains the following chapters:

Chapter	Chapter Name	Page
1	About the Modicon M221 Logic Controller	15
2	Configuration Features	25

Chapter 1
About the Modicon M221 Logic Controller

What Is in This Chapter?

This chapter contains the following topics:

TM221C Logic Controller Description

Overview

The TM221C Logic Controller has various powerful features and can service a wide range of applications.

Software configuration, programming, and commissioning are accomplished with the SoMachine Basic software described in the SoMachine Basic Operating Guide and the M221 Logic Controller - Programming Guide.

Programming Languages

The M221 Logic Controller is configured and programmed with the SoMachine Basic software, which supports the following IEC 61131-3 programming languages:

- IL: Instruction List
- LD: Ladder Diagram
- Grafcet (List)

Power Supply

The power supply of the TM221C Logic Controller is 24 Vdc or 100...240 Vac.

Real Time Clock

The M221 Logic Controller includes a Real Time Clock (RTC) system.

Run/Stop

The M221 Logic Controller can be operated externally by the following:

- a hardware Run/Stop switch
- a Run/Stop operation by a dedicated digital input, defined in the software configuration (for more information, refer to Configuring Digital Inputs *(see page 64)*.)
- SoMachine Basic software (for more information, refer to Toolbar *(see SoMachine Basic, Operating Guide)*).

Memory

This table describes the different types of memory:

Memory Type	Size	Used to
RAM	512 Kbyte, of which 256 Kbyte available for the application.	execute the application and contain data
Flash	1.5 Mbyte, of which 256 Kbyte is used to backup the user application and data in case of power outage.	save the application

Embedded Inputs/Outputs

The following embedded I/O types are available, depending on the controller reference:

- Regular inputs
- Fast inputs associated with counters
- Regular sink/source transistor outputs
- Fast sink/source transistor outputs associated with pulse generators
- Relay outputs
- Analog inputs
- Analog outputs

Removable Storage

The M221 Logic Controllers include an embedded SD card slot.

The main uses of the SD card are:

- Initializing the controller with a new application
- Updating the controller firmware

Embedded Communication Features

The following types of communication ports are available depending on the controller reference:

- Ethernet
- USB Mini-B
- Serial Line 1

TM221C Logic Controllers

Reference	Digital Inputs	Digital Outputs	Analog Inputs	Communication Ports	Power Supply
TM221C16R	5 regular inputs(1) 4 fast inputs (HSC)(2)	7 relay outputs	Yes	1 serial line port 1 USB programming port	100...240 Vac
TM221CE16R			Yes	1 serial line port 1 USB programming port 1 Ethernet port	

NOTE: All TM221C Logic Controller logic controllers use removable screw terminal blocks.

(1) The regular inputs have a maximum frequency of 5 kHz.
(2) The fast inputs can be used either as regular inputs or as fast inputs for counting or event functions.
(3) The fast transistor outputs can be used either as regular transistor outputs, or for PWM, PLS functions, or reflex outputs for HSC.

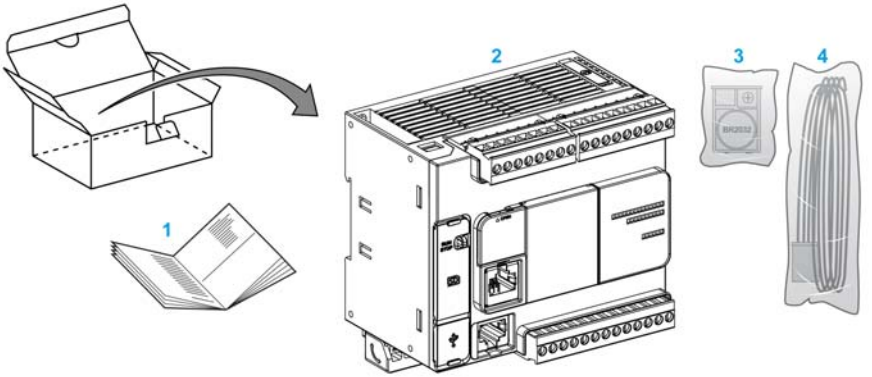
Reference	Digital Inputs	Digital Outputs	Analog Inputs	Communication Ports	Power Supply
TM221C16T	5 regular inputs(1) 4 fast inputs (HSC)(2)	Source outputs 5 regular transistor outputs 2 fast outputs (PWM/PLS)(3)	Yes	1 serial line port 1 USB programming port	24 Vdc
TM221CE16T			Yes	1 serial line port 1 USB programming port 1 Ethernet port	
TM221C24R	10 regular inputs(1) 4 fast inputs (HSC)(2)	10 relay outputs	Yes	1 serial line port 1 USB programming port	100...240 Vac
TM221CE24R			Yes	1 serial line port 1 USB programming port 1 Ethernet port	
TM221C24T		Source outputs 8 regular transistor outputs 2 fast outputs (PWM/PLS)(3)	Yes	1 serial line port 1 USB programming port	24 Vdc
TM221CE24T			Yes	1 serial line port 1 USB programming port 1 Ethernet port	
TM221C40R	20 regular inputs(1) 4 fast inputs (HSC)(2)	16 relay outputs	Yes	1 serial line port 1 USB programming port	100...240 Vac
TM221CE40R			Yes	1 serial line port 1 USB programming port 1 Ethernet port	
TM221C40T		Source outputs 14 regular transistor outputs 2 fast outputs (PWM/PLS)(3)	Yes	1 serial line port 1 USB programming port	24 Vdc
TM221CE40T			Yes	1 serial line port 1 USB programming port 1 Ethernet port	

NOTE: All TM221C Logic Controller logic controllers use removable screw terminal blocks.

(1) The regular inputs have a maximum frequency of 5 kHz.
(2) The fast inputs can be used either as regular inputs or as fast inputs for counting or event functions.
(3) The fast transistor outputs can be used either as regular transistor outputs, or for PWM, PLS functions, or reflex outputs for HSC.

Delivery Content

The following figure shows the content of the delivery for a TM221C Logic Controller:



1 TM221C Logic Controller Instruction Sheet
2 TM221C Logic Controller
3 Battery holder with lithium carbon monofluoride battery, type Panasonic BR2032.
4 Analog cable

TM221M Logic Controller Description

Overview

The TM221M Logic Controller has various powerful features and can service a wide range of applications.

Hardware configuration, programming, and commissioning are accomplished with the SoMachine Basic software described in the SoMachine Basic - Operating Guide.

Programming Languages

The M221 Logic Controller is configured and programmed with the SoMachine Basic software, which supports the following IEC 61131-3 programming languages:

- IL: Instruction List
- LD: Ladder Diagram
- Grafcet (List)

Power Supply

The power supply of the TM221M Logic Controller is 24 Vdc *(see Modicon M221 Logic Controller, Hardware Guide)*.

Real Time Clock

The M221 Logic Controller includes a Real Time Clock (RTC) system.

Run/Stop

The M221 Logic Controller can be operated externally by the following:

- a hardware Run/Stop switch
- a Run/Stop operation by a dedicated digital input, defined in the software configuration (for more information, refer to Configuring Digital Inputs *(see page 64)*)
- SoMachine Basic software (for more information, refer to Toolbar *(see SoMachine Basic, Operating Guide)*).

Memory

This table describes the different types of memory:

Memory Type	Size	Used to
RAM	512 Kbyte, of which 256 Kbyte available for the application.	execute the application and contains data
Flash	1.5 Mbyte, of which 256 Kbyte is used to backup the user application and data in case of power outage.	save the application

Embedded Inputs/Outputs

The following embedded I/O types are available, depending on the controller reference:

- Regular inputs
- Fast inputs (HSC)
- Regular transistor outputs
- Fast transistor outputs (PWM/PLS)
- Relay outputs
- Analog inputs
- Analog outputs

Embedded Communication Features

The following communication ports are available on the front panel of the controller, depending on the controller reference:

- Ethernet
- USB Mini-B
- SD Card
- Serial Line 1
- Serial Line 2

TM221M Logic Controllers

Reference	Digital Input	Digital Output	Analog Input	Communication Port	Terminal Type
TM221M16R	4 regular inputs(1) 4 fast inputs (HSC)(2)	8 relay outputs	Yes	2 serial line ports 1 USB programming port	Removable screw terminal blocks
TM221M16RG	4 regular inputs(1) 4 fast inputs (HSC)(2)	8 relay outputs	Yes	2 serial line ports 1 USB programming port	Removable spring terminal blocks
TM221ME16R	4 regular inputs(1) 4 fast inputs (HSC)(2)	8 relay outputs	Yes	1 serial line port 1 USB programming port 1 Ethernet port	Removable screw terminal blocks
TM221ME16RG	4 regular inputs(1) 4 fast inputs (HSC)(2)	8 relay outputs	Yes	1 serial line port 1 USB programming port 1 Ethernet port	Removable spring terminal blocks

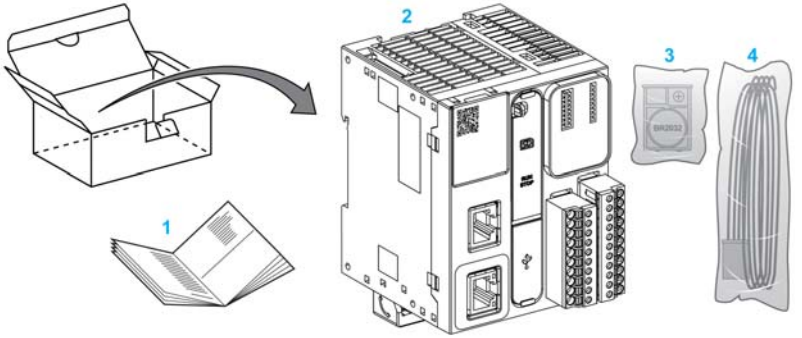
(1) The regular inputs have a maximum frequency of 5 kHz.
(2) The fast inputs can be used either as regular inputs or as fast inputs for counting or event functions.
(3) The fast transistor outputs can be used as regular transistor outputs, or for PWM, PLS functions, or reflex outputs for HSC.

Reference	Digital Input	Digital Output	Analog Input	Communication Port	Terminal Type
TM221M16T	4 regular inputs(1) 4 fast inputs (HSC)(2)	6 regular transistor outputs 2 fast transistor outputs (PWM/PLS)(3)	Yes	2 serial line ports 1 USB programming port	Removable screw terminal blocks
TM221M16TG	4 regular inputs(1) 4 fast inputs (HSC)(2)	6 regular transistor outputs 2 fast transistor outputs (PWM/PLS)(3)	Yes	2 serial line ports 1 USB programming port	Removable spring terminal blocks
TM221ME16T	4 regular inputs(1) 4 fast inputs (HSC)(2)	6 regular transistor outputs 2 fast transistor outputs (PWM/PLS)(3)	Yes	1 serial line port 1 USB programming port 1 Ethernet port	Removable screw terminal blocks
TM221ME16TG	4 regular inputs(1) 4 fast inputs (HSC)(2)	6 regular transistor outputs 2 fast transistor outputs (PWM/PLS)(3)	Yes	1 serial line port USB programming port 1 Ethernet port	Removable spring terminal blocks
TM221M32TK	12 regular inputs(1) 4 fast inputs (HSC)(2)	14 regular transistor outputs 2 fast outputs (PWM/PLS)(3)	Yes	2 serial line ports 1 USB programming port	HE10 (MIL 20) connectors
TM221ME32TK	12 regular inputs(1) 4 fast inputs (HSC)(2)	14 regular outputs 2 fast outputs (PWM/PLS)(3)	Yes	1 serial line port 1 USB programming port 1 Ethernet port	HE10 (MIL 20) connectors

(1) The regular inputs have a maximum frequency of 5 kHz.
(2) The fast inputs can be used either as regular inputs or as fast inputs for counting or event functions.
(3) The fast transistor outputs can be used as regular transistor outputs, or for PWM, PLS functions, or reflex outputs for HSC.

Delivery Content

The following figure shows the content of the delivery for a TM221M Logic Controller:



1 TM221M Logic Controller Instruction Sheet
2 TM221M Logic Controller
3 Battery holder with lithium carbon monofluoride battery, type Panasonic BR2032.
4 Analog cable

Chapter 2
Configuration Features

Introduction

This chapter provides information related to M221 Logic Controller memory mapping, task, states, behaviors, objects, and functions. The topics explained in this chapter allow the operator to understand the featured specifications of M221 Logic Controller that are primarily needed to configure and program the controller in SoMachine Basic.

What Is in This Chapter?

This chapter contains the following sections:

Section 2.1
Objects

What Is in This Section?

This section contains the following topics:

Objects

Overview

In SoMachine Basic, the term *object* is used to represent an area of logic controller memory reserved for use by an application. Objects can be:

- Simple software variables, such as memory bits and words
- Addresses of digital or analog inputs and outputs
- Controller-internal variables, such as system words and system bits
- Predefined system functions or function blocks, such as timers and counters.

Controller memory is either pre-allocated for certain object types, or automatically allocated when an application is downloaded to the logic controller.

Objects can only be addressed by a program once memory has been allocated. Objects are addressed using the prefix `%`. For example, `%MW12` is the address of a memory word, `%Q0.3` is the address of an embedded digital output, and `%TM0` is the address of a `Timer` function block.

Object Types

Introduction

The language objects for the M221 Logic Controller are classified as follows in the SoMachine Basic software:

- Memory objects
- System objects
- I/O objects
- Software objects

The language object types are described in the following table:

Object Type	Object	Object Function	Description
Memory objects	%M	Memory bits	Stores memory bit.
	%MW	Memory words	Stores 16 bit memory word.
	%MD	Memory double words	Stores 4 byte memory word.
	%MF	Memory floating point	Stores memory floating point in a mathematical argument which has a decimal in its expression.
	%KW	Constant words	Stores 16 bit constant word.
	%KD	Constant double words	Stores 4 byte constant word.
	%KF	Constant floating points	Stores constant floating point in a mathematical argument which has a decimal in its expression.
System objects	%S	System bits *(see page 150)*	Stores system bit.
	%SW	System words *(see page 158)*	Stores system word.
I/O objects	%I	Input bits *(see page 110)*	Stores value of the digital input.
	%Q	Output bits *(see page 111)*	Stores value of the digital output.
	%IW	Input words *(see page 112)*	Stores value of the analog input.
	%FC	Fast counters *(see page 116)*	Serves as either up-counter or down-counter and counts the rising edge of discrete inputs in single word or double word computational mode.
	%HSC	High speed counters *(see page 122)*	Counts of discrete input in single word or double word computational mode.
	%PLS	Pulse generator output *(see page 135)*	Generate square wave signals at dedicated output channels and allows only a single signal width, or duty cycle, of 50%.
	%PWM	Pulse width modulation *(see page 142)*	Generate square wave signals at dedicated output channels and allows variable width and, so, duty cycle.

Object Type	Object	Object Function	Description
Software objects	%TM	Timers	Specifies a time before triggering an action.
	%C	Counters	Provides up and down counting of actions.
	%MSG	Messages	Stores the status message at the communication port.
	%R	LIFO/FIFO registers	Stores memory up to 16 words of16 bits each in 2 different ways, queue, and stacks.
	%DR	Drum registers	Operates on a principle similar to an electromechanical drum controller which changes step according to external events.
	%SBR	Shift bit registers	Provides a left or right shift of binary data bits (0 or 1).
	%SC	Step counters	Provides a series of steps to which actions can be assigned.
	SCH	Schedule blocks	Controls actions at a predefined month, day, and time.
	PID	PID control	Provides a generic control loop feedback in which output is proportional, integral, and derivative of the input.

Memory objects and software objects are generic objects used in SoMachine Basic, whereas system objects and I/O objects are controller-specific. All controller-specific objects are discussed in the Programming *(see page 103)* section.

For programming details of memory objects and software objects, refer to the SoMachine Basic Generic Functions Library Guide.

Addressing I/O Objects

Addressing Examples

This table shows addressing examples for various object types:

Object Type	Syntax	Example	Description
Memory objects			
Memory bits	%M*i*	%M25	Internal memory bit 25.
Memory words	%MW*i*	%MW15	Internal memory word 15.
Memory double words	%MD*i*	%MD16	Internal memory double word 16.
Memory floating points	%MF*i*	%MF17	Internal memory floating point 17.
Constant words	%KW*i*	%KW26	Constant word 26.
Constant double words	%KD*i*	%KD27	Internal constant double word 27.
Constant floating points	%KF*i*	%KF28	Internal constant floating point 28.
System objects			
System bits	%S*i*	%S8	System bit 8.
System words	%SW*i*	%SW30	System word 30.
I/O objects			
Digital inputs	%I*y.z*	%I0.5	Digital input 5 on the controller (embedded I/O).
Digital outputs	%Q*y.z*	%Q3.4	Digital output 4 on the expansion module at address 3 (expansion module I/O).
Analog inputs	%IW*y.z*	%IW0.1	Analog input 1 on the controller (embedded I/O).
Fast counters	%FC*i*	%FC2	Fast counter 2 on the controller.
High speed counters	%HSC*i*	%HSC1	High speed counter 1 on the controller.
Pulse	%PLS*i*	%PLS0	Pulse output 0 on the controller.
Pulse width modulation	%PWM*i*	%PWM1	Pulse width modulation output 1 on the controller.
Software objects			
Timers	%TM*i*	%TM5	Timer instance 5.
Counters	%C*i*	%C2	Counter instance 2.
Message	%MSG*i*	%MSG1	Program compilation status message 1.
LIFO/FIFO registers	%R*i*	%R3	FIFO/LIFO registers instance 3.

i Object instance identifier that indicates the instance of the object on the controller.
y Indicates the I/O type. It is 0 for the controller and 1,2, and so on, for the expansion modules.
z Channel number on the controller or expansion module.

Object Type	Syntax	Example	Description
Drum controllers	%DR*i*	%DR6	Drum controller 6 on the controller.
Shift bit registers	%SBR*i*	%SBR5	Shift bit register 5 on the controller.
Step counters	%SC*i*	%SC5	Step counter 5 on the controller.
Schedule blocks	SCH *i*	SCH 3	Schedule block 3 on the controller.
PID control	PID *i*	PID 7	PID feedback object 7 on the controller.

i Object instance identifier that indicates the instance of the object on the controller.
y Indicates the I/O type. It is 0 for the controller and 1,2, and so on, for the expansion modules.
z Channel number on the controller or expansion module.

Maximum Number of Objects

Description

This table provides information about the maximum number of objects supported by the M221 Logic Controller:

Objects	M221 Logic Controller References			
	Modular References		Compact References	
	TM221M16R• TM221ME16R•	TM221M16T• TM221ME16T• TM221M32TK TM221ME32TK	TM221C••R TM221CE••R	TM221C••T TM221CE••T
Memory objects				
%M	512	512	512	512
%MW	8000	8000	8000	8000
%MD %MF	7999	7999	7999	7999
%KW	512	512	512	512
%KD %KF	511	511	511	511
System objects				
%S	160	160	160	160
%SW	234	234	234	234
I/O objects				
%I	8	8 (for TM221M16T• and TM221ME16T•)	9 (for TM221C16• and TM221CE16•)	9 (for TM221C16• and TM221CE16•)
		16 (for TM221M32TK and TM221ME32TK)	14 (for TM221C24• and TM221CE24•)	14 (for TM221C24• and TM221CE24•)
			24 (for TM221C40• and TM221CE40•)	24 (for TM221C40• and TM221CE40•)

Objects	M221 Logic Controller References			
	Modular References		Compact References	
	TM221M16R• TM221ME16R•	TM221M16T• TM221ME16T• TM221M32TK TM221ME32TK	TM221C••R TM221CE••R	TM221C••T TM221CE••T
%Q	8	8 (for TM221M16T• and TM221ME16T•)	7 (for TM221C16• and TM221CE16•)	7 (for TM221C16• and TM221CE16•)
		16 (for TM221M32TK and TM221ME32TK)	10 (for TM221C24• and TM221CE24•)	10 (for TM221C24• and TM221CE24•)
			16 (for TM221C40• and TM221CE40•)	16 (for TM221C40• and TM221CE40•)
%IW	2	2	2	2
%FC	4	4	4	4
%HSC	2	2	2	2
%PLS %PWM	0	2	0	2
Software objects				
%TM	255	255	256	256
%C	255	255	256	256
%MSG	2	2	1 (for TM221C••R)	1 (for TM221C••T)
			2 (for TM221CE••R)	2 (for TM221CE••T)
%R	4	4	4	4
%DR	8	8	8	8
%SBR	8	8	8	8
%SC	8	8	8	8
%SCH	16	16	16	16
PID	14	14	14	14

Section 2.2
Task Structure

What Is in This Section?

This section contains the following topics:

Tasks and Scan Modes

Overview

The Modicon TM221M Logic Controller supports the following task types:

- Master task
- Periodic task
- Event task

The master tasks can be configured in either of the following scan modes:

- Freewheeling mode
- Periodic mode

For more information, refer to the Configuring Program Behavior and Tasks *(see SoMachine Basic, Operating Guide)*.

Tasks

Master tasks are triggered by continuous cyclic scanning or by the software times by specifying the scan period 2...150 ms (default 100 ms) in the periodic mode.

Periodic tasks are triggered by software timers, so are configured by specifying the scan period 5...255 ms (default 255 ms) in the periodic mode.

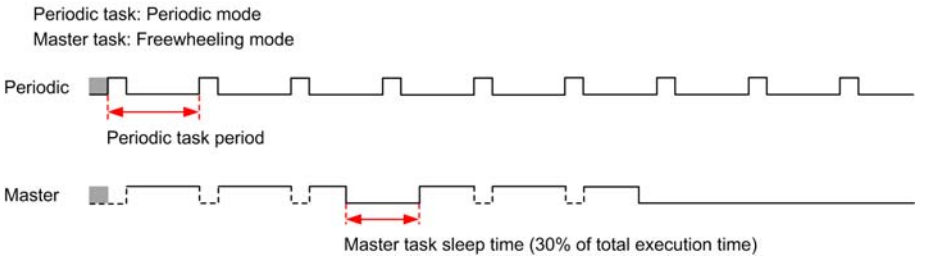
Event tasks are triggered by the physical inputs or the HSC function blocks. These events are associated with embedded digital inputs (%I0.2...%I0.5) (rising, falling or both edges) or with the high speed counters (%HSC0 and %HSC1) (when the count reaches the high speed counter threshold). You can configure 2 events for each HSC function block.

You must configure one priority for each event task. The priority range is 0...7 and the priority 0 has the highest priority.

Scan Modes

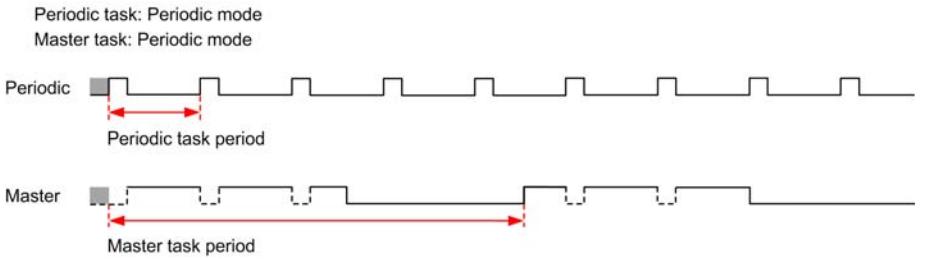
The freewheeling mode is a continuous cyclic scanning mode. In this mode, a new scan starts immediately after the previous scan has completed.

This figure shows the relationship between master tasks and periodic tasks when the master task is in freewheeling mode:



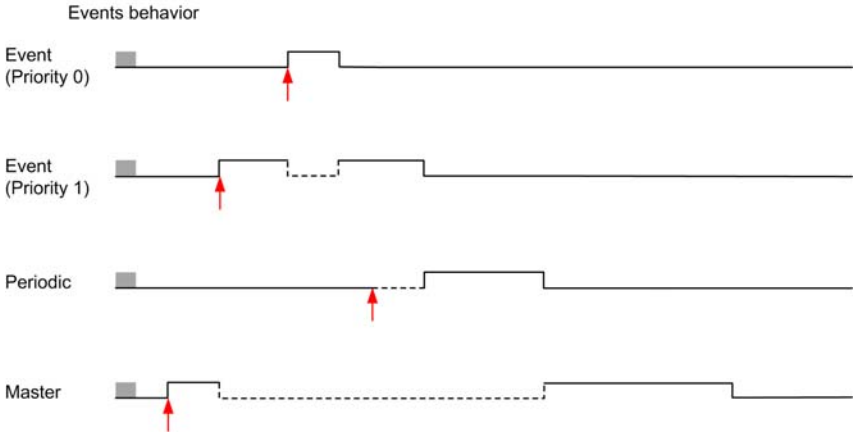
In periodic mode, the logic controller waits until the configured scan time has elapsed before starting a new scan. Every scan is therefore the same duration.

This figure shows the relationship between master tasks and periodic tasks when the master task is in periodic mode:



Event priorities control the relationship between the event tasks, master tasks, and periodic tasks. The event task interrupts the master task and periodic task execution.

This figure shows the relationship between event tasks, master tasks, and periodic tasks in the periodic mode:



The event tasks are triggered by a hardware interruption that sends a task event to the event task.

Watchdog Timer

You can configure a specific watchdog timer for the master task and periodic tasks. If the task execution time exceeds the configured watchdog timer period, the logic controller goes to the HALTED state. This watchdog timer is managed by the software timers.

A system watchdog timer checks if the user logic is loading more than 80% of the CPU processing capacity.

Maximum Number of Tasks and Priorities

Description

This table summarizes the task types, available scan modes for each task, scan mode triggering conditions, operator configurable ranges, maximum number of each task, and their execution priorities:

Task Type	Scan Mode	Triggering Condition	Configurable Range	Maximum Number of Tasks	Priority
Master	Freewheeling	Normal	Not applicable	1	Lowest
	Periodic	Software timer	2...150 ms		
Periodic	Periodic	Software timer	5...255 ms	1	Higher than master task and lower than event tasks
Event	Periodic	Physical inputs	%I0.2...%I0.5	4	Highest
		%HSC function blocks	2 events per %HSC object	4	

Section 2.3
Controller States and Behaviors

Introduction

This section provides you with information on controller states, state transitions, and behaviors in response to system events. It begins with a detailed controller state diagram and a description of each state. It then defines the relationship of output states to controller states before explaining the commands and events that result in state transitions. It concludes with information about persistent variables and the effect of SoMachine Basic task programming options on the behavior of your system.

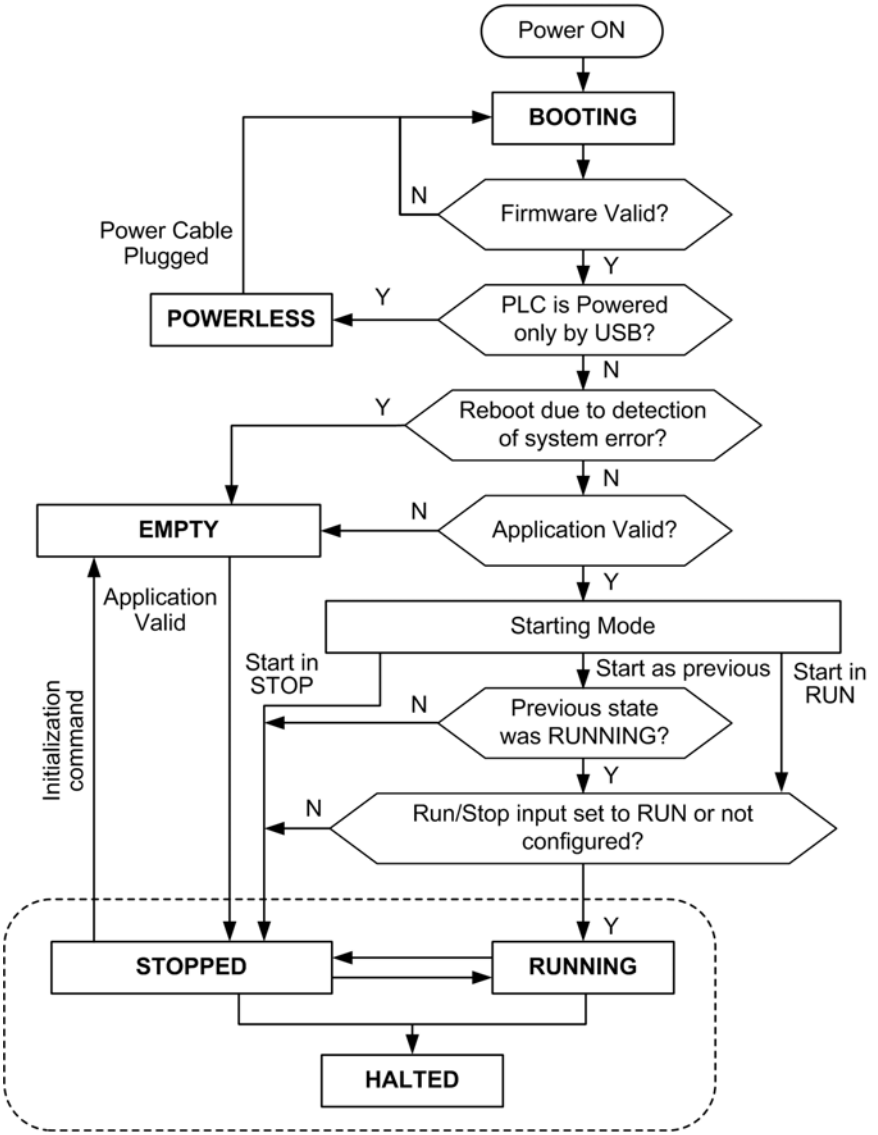
What Is in This Section?

This section contains the following topics:

Controller State Diagram

Controller State Diagram

This figure describes the controller operating states:



Controller States Description

Introduction

This section provides a detailed description of the controller states.

⚠ WARNING

UNINTENDED EQUIPMENT OPERATION

- Never assume that your controller is in a certain controller state before commanding a change of state, configuring your controller options, uploading a program, or modifying the physical configuration of the controller and its connected equipment.
- Before performing any of these operations, consider the effect on all connected equipment.
- Before acting on a controller, always positively confirm the controller state by viewing its LEDs, confirming the condition of the Run/Stop input, checking for the presence of output forcing, and reviewing the controller status information via SoMachine Basic.

Failure to follow these instructions can result in death, serious injury, or equipment damage.

When using Automatic Start in Run, the controller will start executing program logic when power is applied to the equipment. It is essential to know in advance how automatic reactivation of the outputs will affect the process or machine being controlled. Configure the Run/Stop input to help control the Automatic Start in Run feature. In addition, the Run/Stop input is designed to give local control over remote RUN commands. If the possibility of a remote RUN command after the controller had been stopped locally by SoMachine would have unintended consequences, you must configure and wire the Run/Stop input to help control this situation.

⚠ WARNING

UNINTENDED MACHINE START-UP

- Confirm that the automatic reactivation of the outputs does not produce unintended consequences before using the Automatic Start in Run setting.
- Use the Run/Stop input to help avoid an unwanted restart in Run mode and to help prevent the unintentional start-up from a remote location.
- Verify the state of security of your machine or process environment before applying power to the Run/Stop input or before issuing a Run command from a remote location.

Failure to follow these instructions can result in death, serious injury, or equipment damage.

Controller States Table

This table provides detailed description of the controller operating states:

Controller State	Description	Communication	Application Execution	LED		
				PWR	RUN	ERR
BOOTING	This state indicates that the logic controller is running the BOOT firmware. The logic controller executes its own internal autotests. The communication channels are enabled to allow updating of the runtime firmware. It is not possible to login with SoMachine Basic. Outputs are set to initialization values *(see page 48)*.	Restricted	No	On	Flashing	On
	The logic controller does not have valid firmare. The communication channels are enabled to allow updating of the runtime firmware. It is not possible to login with SoMachine Basic. Outputs are set to initialization values *(see page 48)*.	Restricted	No	On	Off	On
EMPTY	This state indicates that there is not a valid application. It is possible to login with SoMachine Basic (*download/watchlist*). Inputs are forced to 0. Outputs are set to initialization values *(see page 48)*.	Yes	No	On	Off	1 flash
STOPPED	This state indicates that the logic controller has a valid application which is stopped. Inputs are read. Outputs are set to fallback values *(see page 48)*, or forced values *(see page 49)* from SoMachine Basic. Alarm output is set to 0.	Yes	No	On	Flashing	Off

Controller State	Description	Communication	Application Execution	LED		
				PWR	RUN	ERR
RUNNING	This state indicates that the logic controller is executing the application. There may be a non-fatal detected error but the logic controller does not stop the application. Inputs are read by the application tasks. Outputs are managed by the application tasks, or from SoMachine Basic in online mode (watchlist, output forcing *(see page 49)*). Alarm output is set to 1.	Yes	Yes	On	On	Off
HALTED	This state indicates that the application is stopped because of an application software detected error. *(see page 45)* Objects retain their current values, allowing analysis of the cause of the detected error. All tasks are stopped on the current instruction. The communication capabilities are the same as in STOPPED state. Inputs are not read, and keep their last values. Outputs are set to fallback values *(see page 48)*. Alarm output is set to 0.	Yes	No	On	Flashing	On

Controller State	Description	Communication	Application Execution	LED		
				PWR	RUN	ERR
POWERLESS	This state indicates that the logic controller is powered only by the USB cable. This mode can only be used to update the firmware (by USB) or to download/upload the user application (by USB). In this state, all state transitions are not allowed, so the only way to change the state of the logic controller is to connect the main power, but in this case the logic controller will boot and reload all installed components. It is possible to login with SoMachine Basic (*download/upload/watchlist*). Inputs are forced to 0. Outputs are set to initialization values *(see page 48)*.	Yes (only USB)	No	Off	Flashing	Off

NOTE: The system word %SW6 indicates the logic controller state (EMPTY, STOPPED, RUNNING, HALTED, and POWERLESS).

Controller State Transitions

Boot Controller

Effect: Command a reboot of the controller, for details about power-on sequence see the controller state diagram *(see page 39)*.

Methods:

- Power cycle
- Reboot by script
  - The script on an SD card can issue a REBOOT as its final command.

Application Download

Effect: Download the application into the controller memory.

Methods:

- SoMachine Basic online button:
  - Select the **PC to controller (download)** command.
    Effect: Erase the current application and set the controller in `EMPTY` state. If download is successful, a Cold Start is done and the controller is set in `STOPPED` state.
- Application file transfer by SD card:
  - Effect: At the next reboot, erase the current application and copies the application files from the SD card to the controller memory. If download is successful, a Cold Start is done and the controller is set in `STOPPED` state.

Initialize Controller

Effect: Sets the controller in `EMPTY` state, and then after a Cold Start in `STOPPED` state.

Methods:

- SoMachine Basic online button:
  - Select the **Initialize controller** command.

RUN Controller

Effect: Command a transition to the `RUNNING` controller state.

Methods:

- Run/Stop *(see Modicon M221 Logic Controller, Hardware Guide)* switch on front face:
  - It commands a transition to `RUNNING` state on rising edge.
- Run/Stop *(see Modicon M221 Logic Controller, Hardware Guide)* input:
  - The input must be configured in the application (Configuring Digital Inputs *(see page 64)*).
  - It commands a transition to `RUNNING` state on rising edge.
- SoMachine Basic online button:
  - Select the **Run Controller** command.
- Application starting mode *(see SoMachine Basic, Operating Guide)* setting:
  - **Start in Run** or **Start in Previous State**.

STOP Controller

Effect: Command a transition to the STOPPED controller state.

Methods:

- Run/Stop *(see Modicon M221 Logic Controller, Hardware Guide)* switch on front face:
  - It forces a transition to STOPPED state on low level.
- Run/Stop *(see Modicon M221 Logic Controller, Hardware Guide)* input:
  - The input must be configured in the application (Configuring Digital Inputs *(see page 64)*).
  - It forces a transition to STOPPED state on low level.
- SoMachine Basic online button:
  - Select the **Stop Controller** command.
- Application starting mode *(see SoMachine Basic, Operating Guide)* setting:
  - **Start in Stop** or **Start in Previous State**.
- **Download** command:
  - It needs the controller to be set in STOPPED state (after the download the controller is in STOPPED state).

Error Detected (Transition to HALTED State)

Effect: Command a transition to the HALTED controller state.

Reasons for switching to HALTED state:

- Watchdog (Application Behavior *(see SoMachine Basic, Operating Guide)*)
- System overrun, over 80% of the CPU processing capacity (Watchdog Timer *(see page 36)*)
- If download is successful, a Cold Start is done and the controller is set in STOPPED state.

Cold Start

Cold Start is defined to be a power-up with all data initialized to its default values, and User Logic started from the beginning of the program with all program variables cleared. In a Cold Start everything is initialized: software and hardware settings.

Cold Start occurs for the following reasons:

- Boot controller without validated application online modification.
- PLC without a charged backup battery always powers up via a Cold Start.
- Application Download
- Initialize controller

Effects of the Cold Start:

- Initialize all the function blocks.
- Clear all the user memory.
- Put the objects %S and %SW in the initial values.
- Reload parameters from post configuration (it means that the changes in the post configuration will be effective)
- Restore user application from flash (it means that unsaved online changes are lost).
- Restart all the internal components of the controller.

Warm Start

The Warm Start simply resume running the USER logic program, in its last operating state, with all counters, function blocks, and system words and bits preserved.

Persistent Variables

Automatic Saved on Power Outage

The controller automatically saves the first 50 Memory Words (%MW0 to %MW49) in the internal data flash following a power outage. The date is restored to the memory word region during the initialization, even if the controller performs a cold start due to a battery malfunction.

These *automatically saved* persistent variables are reinitialized in case of a new download, INIT command or %S0 activation (refer to System Bits *(see page 150)*).

Saved by User Request

You can save up to 1000 memory words (%MW50 up to %MW1049) in the flash memory. The number of words saved or restored is specified in the system word %SW148 (refer to System Words *(see page 158)*).

To perform the SAVE operation, set system bit %S93 to 1 (refer to System Bits *(see page 150)*). The flash region is erased at the end of the MAST cycle. The operation exclusively occupies the controller for approximately 40 ms.

The writing of memory words to flash memory is performed in stages between MAST cycles. The system bit%S92 (refer to System Bits *(see page 150)*) is set to 1 to signal the end of operation.

The restore is performed by setting %S94 to 1 (refer to System Bits *(see page 150)*). The restore operation is performed completely at the end of the MAST cycle.

Output Behavior

Introduction

The controller defines output behavior in response to commands and system events in a way that allows for greater flexibility. An understanding of this behavior is necessary before discussing the commands and events that affect controller states.

The possible output behaviors and the controller states to which they apply are:

- Managed by application program
- Initialization values
- Fallback Mode *(see SoMachine Basic, Operating Guide)*
  - Maintain values
  - Fallback values
- Output forcing

Managed by Application Program

Your application program manages outputs normally. This applies in the `RUNNING` state.

Initialization Values

This output state applies in the `BOOTING`, `EMPTY` and `POWERLESS` states.

In the initialization state, the outputs assume the following values:

- For embedded outputs:
  - Fast transistor output: 0 Vdc
  - Regular transistor output: 0 Vdc
  - Relay output: Open
  - Expert I/O functions (PLS/PWM, HSC): 0 Vdc
- For expansion module outputs:
  - Regular transistor output: 0 Vdc
  - Relay output: Open

Fallback Values

This output state applies in the `STOPPED` and `HALTED` states.

In the fallback mode, the outputs assume the following values:

- For embedded outputs:
  - Fast transistor output: according to fallback setting (Fallback Behavior *(see SoMachine Basic, Operating Guide)*)
  - Regular transistor output: according to fallback setting (Fallback Behavior)
  - Relay output: according to fallback setting (Fallback Behavior)
  - Expert I/O functions (PLS/PWM, HSC):
    - Fallback value: according to fallback setting (Fallback Behavior)
    - **Maintain values**: 0 Vdc

- For expansion module outputs:
  - Regular transistor output: according to fallback setting (Fallback Behavior)
  - Relay output: according to fallback setting (Fallback Behavior)

Output Forcing

The controller allows you to force the state of selected outputs to a defined value for the purposes of system testing, commissioning, and maintenance.

You are only able to force the value of an output while your controller is connected to SoMachine Basic.

To do so, use the **Force** command in an animation table.

Output forcing overrides all other commands to an output irrespective of the task programming that is being executed.

The forcing is not released by online change or logout of SoMachine Basic.

The forcing is automatically released by Cold Start *(see page 45)* and Download application *(see page 44)* command.

The forcing does not apply for expert I/O functions (PLS, PWM, and HSC).

⚠ WARNING

UNINTENDED EQUIPMENT OPERATION

- You must have a thorough understanding of how forcing will affect the outputs relative to the tasks being executed.
- Do not attempt to force I/O that is contained in tasks that you are not certain will be executed in a timely manner, unless your intent is for the forcing to take affect at the next execution of the task whenever that may be.
- If you force an output and there is no apparent affect on the physical output, do not exit SoMachine Basic without removing the forcing.

Failure to follow these instructions can result in death, serious injury, or equipment damage.

Part II
Configuring the M221 Logic Controller

Overview

This part provides information about how to configure the M221 Logic Controller references.

What Is in This Part?

This part contains the following chapters:

Chapter 3
How to Configure a Controller

Overview

This chapter describes how to build a configuration in SoMachine Basic and configure the M221 Logic Controller.

What Is in This Chapter?

This chapter contains the following topics:

Building a Configuration

Introduction

Configure a controller by building a configuration in SoMachine Basic. To build a configuration, first create a new project or open an existing project.

Refer to *SoMachine Basic Operating Guide* for information on how to:

- create or open an existing project
- replace the default logic controller
- add an expansion module to the logic controller
- add a cartridge to the logic controller
- save the project.

Some general information about the SoMachine Basic user interface is provided below.

Start Page

The start page window is always displayed when you launch SoMachine Basic. Use this window to register the SoMachine Basic software, manage the connection to the logic controller, and create or select a project to work with.

SoMachine Basic Window

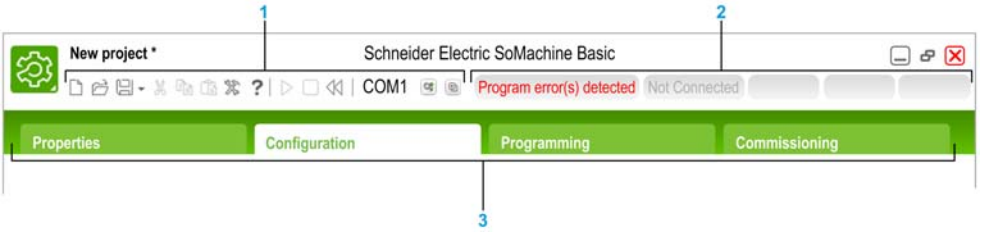
Once you have selected a project to work with, SoMachine Basic displays the main window.

At the top of the main window, a toolbar contains icons that allow you to perform common tasks, including returning to the start page window.

Next to the toolbar, the status bar displays informational messages about the current state of the connection to the logic controller.

Below the toolbar and the status bar, the main window is divided into a number of *modules*. Each module controls a different stage of the development cycle, and is accessible by clicking the module tab.

This figure shows the toolbar, status bar, and the module tabs in the main window:



1 Toolbar
2 Status bar
3 Module tabs

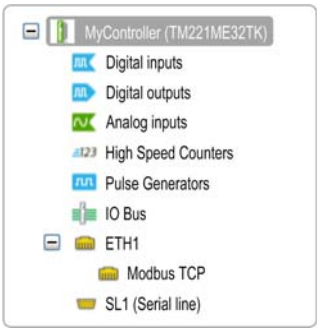
Item	Description
Toolbar	Provides easy access to commonly used functions. For more information, refer to the Toolbar .
Status bar	Displays status and information messages on the current system status. For more information, refer to the Status bar.
Module tabs	To develop an application, work your way through the module tabs from left to right: ● **Properties** Set up the project properties. ● **Configuration** Replicate and configure the hardware configuration of the logic controller and associated expansion modules. ● **Programming** Develop the program in one of the supported programming languages. ● **Commissioning** Manage the connection between SoMachine Basic and the logic controller, upload/download applications, test, and commission the application.

Hardware Tree

The hardware tree is displayed on left-hand side in the **Configuration** window. It shows a structured view of the current hardware configuration. When you add a controller, an expansion module, or a cartridge to the project, several nodes are automatically added to the hardware tree.

NOTE: The nodes in the hardware tree are specific to the controller and the hardware configuration. These nodes depend on the I/O functions that the controller, expansion modules, and cartridges provide.

This figure shows the hardware tree of the controller configuration:



Item	Description
Digital inputs	Use to configure the embedded digital inputs of the logic controller.
Digital outputs	Use to configure the embedded digital outputs of the logic controller.
n Serial line number (1 or 2, controller-specific).	

Item	Description
Analog inputs	Use to configure the embedded analog inputs of the logic controller.
High Speed Counters	Use to configure the embedded high speed counting functions (HSC).
Pulse Generators	Use to configure the embedded pulse generator functions (PLS/PWM).
IO Bus	Use to configure the expansion modules and cartridges connected to the logic controller.
ETH1	Use to configure the embedded Ethernet communications.
Modbus TCP	Use to configure the Modbus TCP for the Ethernet communications.
SLn **(Serial line)**	Use to configure the embedded serial line or the serial line added using a cartridge.
n Serial line number (1 or 2, controller-specific).	

Editor

The editor area is displayed in center of the **Configuration** window. It displays the graphical representation of hardware configuration of the devices. The hardware configuration in a project can be:

- only a controller
- a controller with cartridge
- a controller with expansion modules
- a controller with cartridge and expansion modules.

The editor area displays:

- a short description about the device when you click the device image or when you click the device node in the hardware tree.
- configuration properties of the item selected in the hardware tree.

If you add an expansion module to the configuration, the expansion module appears at the right-hand side of the controller or the previously added expansion module. Cartridges are added on the controller in the cartridge slot.

When configuring a controller, a cartridge, or an expansion module, the configuration properties of the node currently selected in the hardware tree are displayed below the graphical configuration. These properties allow you to configure the device.

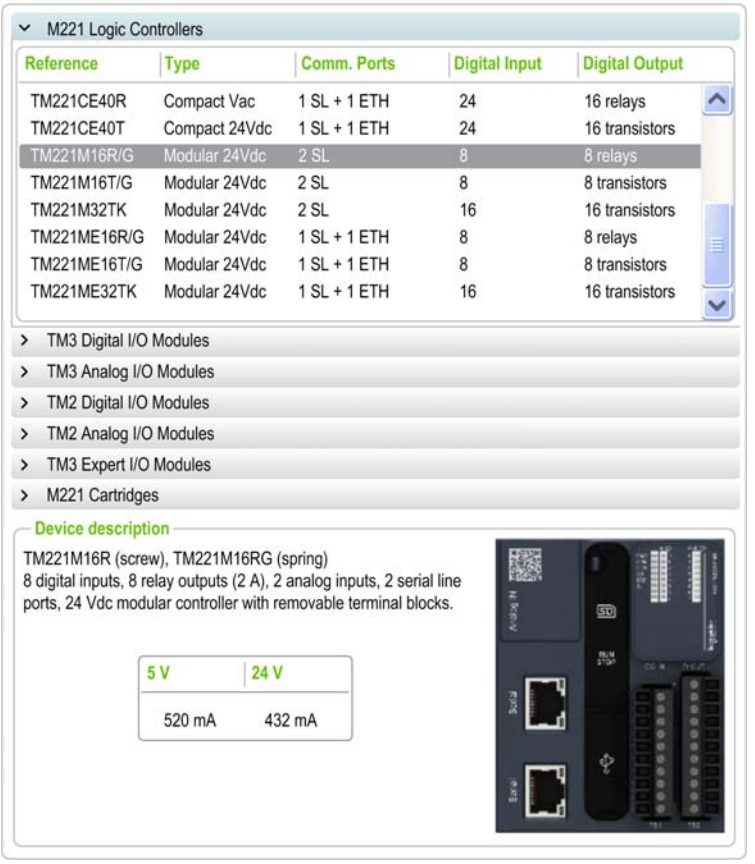
This figure shows the configuration of a controller with an expansion module (the controller is selected):

Catalog

The catalog area is displayed on right-hand side in the **Configuration** window. It displays the complete range of the logic controllers, expansion modules, and cartridges that can be configured using SoMachine Basic. It also provides a short description of the selected device.

You can drag-and-drop the objects from the catalog area to the editor area. You can also replace the existing controller by a different controller with simple drag-and-drop from the catalog.

This figure shows the catalog of the logic controllers and the expansion modules:



Configuring the M221 Logic Controller

Controller Configuration

Controller configuration depends on the number and type of embedded input/outputs, I/O objects, and communication ports.

Use the **Configuration** tab to configure the properties of your controller and the expansion modules. Select a node in the hardware tree to configure the properties of the controller.

This table shows the available configurations of the M221 Logic Controller:

Reference	Digital Input	Digital Output	Analog Input	High Speed Counter	Pulse Generator	Ethernet	Serial Line
TM221M16R• TM221C••R	X	X	X	X			X
TM221ME16R• TM221CE••R	X	X	X	X		X	X
TM221M16T• TM221M32TK TM221C••T	X	X	X	X	X		X
TM221ME16T• TM221ME32TK TM221CE••T	X	X	X	X	X	X	X

X Available for configuration in SoMachine Basic. For information on how to configure:

- Digital inputs, refer to Configuring Digital Inputs *(see page 64)*
- Digital outputs, refer to Configuring Digital Outputs *(see page 68)*
- Analog inputs, refer to Configuring Analog Inputs *(see page 70)*
- High speed counters, refer to Configuring High Speed Counters *(see page 72)*
- Pulse generators, refer to Configuring Pulse Generators *(see page 80)*
- Ethernet, refer to Configuring Ethernet *(see page 92)*
- Serial lines, refer to Configuring Serial Line *(see page 98)*.

Downloading Firmware Updates with an SD Card

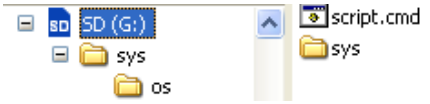
Overview

You can use an SD card to download firmware updates directly to the logic controller.

Refer to Controller States and Behavior *(see page 38)* for information on the logic controller operating states and status of the LEDs.

Downloading Firmware to the Controller

This table describes how to download the firmware in the logic controller with an SD card:

Step	Action
1	Stop the logic controller and unplug the USB programming cable if connected.
2	Insert an empty SD card into the PC that is running SoMachine Basic.
3	Copy the firmware update files to the root folder of the SD card. For example: 
4	Remove the SD card from the PC and insert it into the SD card slot of the logic controller.
5	Start the logic controller. **Result:** Copying of the firmware file begins. During the operation, the SD system LED on the logic controller is On. Do not stop the logic controller while the operation is in progress. **NOTE:** The firmware download process has a low priority in order to minimize impact on the user logic and communication performance of the logic controller. Depending on the amount of free time in your program, the operation may take considerably longer to complete if the logic controller is in RUN or STOP mode than if it is in BOOTING mode.
6	When the SD system LED is turned Off, remove the SD card.
7	Reconnect the USB programming cable to the logic controller and login to the logic controller with the SoMachine Basic software.

Updating Firmware using Executive Loader Wizard

Overview

You can update the executives of the controller using the executive loader wizard (OS loader).

Refer to Controller States and Behavior *(see page 38)* for information on the logic controller operating states and status of the LEDs.

Updating the Firmware of the Controller

To launch the Exec Loader Wizard, follow these steps:

Step	Action
1	Close all Windows applications, including virtual machines.
2	Click **Start** →**Programs** →**Schneider Electric** →**SoMachine Basic** →**M221 Firmware Update** or run the *ExecLoaderWizard.exe* from *SoMachine Basic installation folder\Execloader* folder.

Chapter 4
Embedded Input/Output Configuration

Overview

This chapter describes how to configure the embedded I/O objects of the M221 Logic Controller.

The number of embedded inputs and outputs depends on the controller reference. For more information, refer to the tables TM221C Logic Controllers *(see page 17)* and TM221M Logic Controller *(see page 21)*s.

What Is in This Chapter?

This chapter contains the following topics:

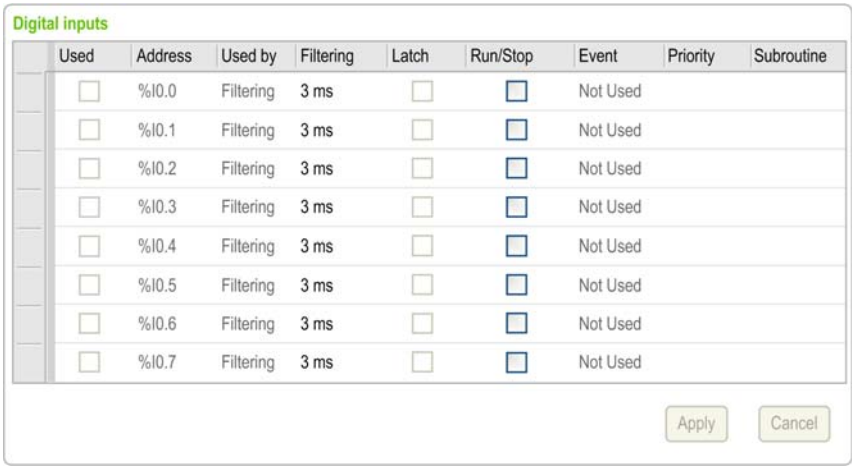
Configuring Digital Inputs

Introduction

By default, all digital inputs are used as regular inputs. Some of the digital inputs are fast and can be used by configuring the high speed counters *(see page 72)* while other inputs can be configured as event sources.

Digital Inputs Configuration

This table describes how to configure the digital inputs:

Step	Action
1	Click the **Digital inputs** node in the hardware tree to display the digital input properties. This figure shows the properties of the digital inputs in the editor area: 
2	Edit the properties to configure the digital inputs. For detailed information on the digital input configuration parameters, refer to the table below.

This table describes each parameter of the digital input configuration:

Parameter	Editable	Value	Default Value	Description
Used	No	True/False	False	Indicates whether the input channel is being used in a program or not.
Address	No	%I0.x	–	Displays the address of the digital input on the controller, where x represents the channel number. If the controller has 8 digital input channels, x varies from 0...7. If the controller has 16 digital input channels, x varies from 0...15. For example, `%I0.2` is the third digital input channel of the logic controller.
Used by	No	*any*	**Filtering**	Displays the name of the component that uses the input channel. For example, if the input channel is used by a subroutine, this field displays **User logic**. The possible values in this field are: ● **User logic** ● **Filtering** ● **Latch** ● **Run/Stop** ● **Event** ● **%HSCx** where x is the high speed counter instance on the controller ● **%FC**y where y is the fast counter instance on the controller If an input is being used by more than one operation, all values, separated by commas, are displayed in this field.
Filtering	Yes	**No Filter** **3 ms** **12 ms**	**3 ms**	Allows you to select the noise filter duration for the input channel. Using a filter for the digital inputs reduces the noise on the controller input. If you select filter for an input, you cannot configure that input for: ● **Latch** ● **Event**

Parameter	Editable	Value	Default Value	Description
Latch	Yes	True/False	False	Allows you to enable or disable latching for the inputs configured as events (%I0.2...%I0.5). By default, this option is disabled due to default value of **Filtering**. Set the **Filtering** to **No Filter** to enable the **Latch** option. Latching enables pulses with a duration shorter than the controller scan time to be memorized. When a pulse duration is shorter than a scan time and has a value greater than or equal to 1 ms, the controller latches the pulse, which is then updated in the next scan. If you enable **Latch** for an input, you cannot configure that input for: • **Filtering** • **Run/Stop** • **Event**
Run/Stop	Yes	True/False	False	Allows you to configure 1 digital input as an additional Run/Stop switch. If you configure a digital input as Run/Stop switch, you cannot use the input in any other function block (for example, high speed counter function block, fast counter function block, and so on). If you enable **Run/Stop** for an input, you cannot configure that input for: • **Latch** • **Event**
Event	Yes	**Not Used** **Falling Edge** **Rising Edge** **Both edges**	**Not Used**	Allows you to select an event that triggers the inputs %I0.2...%I0.5. By default, this option is disabled due to default value of **Filtering**. Set the **Filtering** to **No Filter** to enable the **Event** option. If you select an event from the drop-down list (other than **Not Used**), the **Priority** parameter enables for editing to set the priority of the event.

Parameter	Editable	Value	Default Value	Description
Priority	Yes	0...7	7	Allows you to set the priority of the triggering event for the inputs `%I0.2`...`%I0.5`. You can set the priority of each event using the **Priority** parameter that is editable only for the inputs configured as event. Assign each configured event a different priority: if 2 events have same priority, a detected error message appears in the window.
Subroutine	No	*any*	*empty*	Displays the number of the subroutine associated with an input configured as an event.

Additional configuration details are displayed in the **Programming** tab. For more information, refer to Digital Inputs (%I) *(see page 110)*.

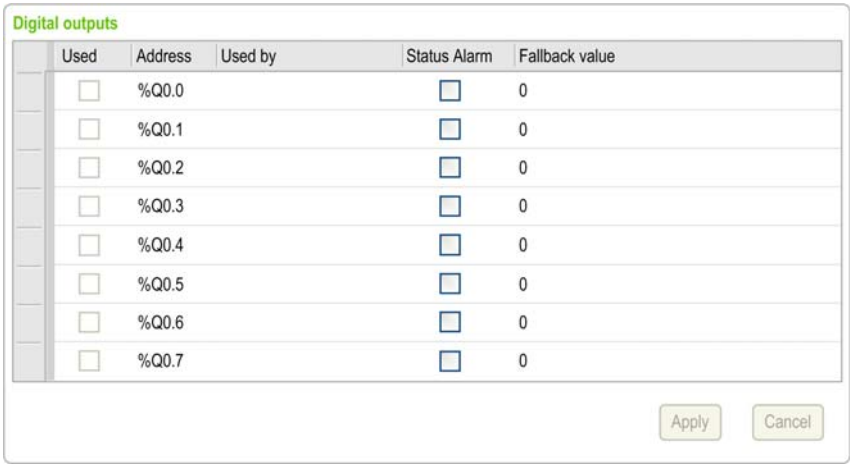
Configuring Digital Outputs

Introduction

By default, all digital outputs are used as regular outputs. For controllers equipped with transistor outputs, 2 outputs are fast transistor outputs and can be used by configuring the pulse generators *(see page 80)*.

Digital Outputs Configuration

This table describes how to configure the digital outputs:

Step	Action
1	Click the **Digital outputs** node in the hardware tree to display the digital output properties. This figure shows the properties of the digital outputs in the editor area: 
2	Edit the properties to configure the digital outputs. For detailed information on the digital output configuration parameters, refer to the table below.

This table describes each parameter of the digital output configuration:

Parameter	Editable	Value	Default Value	Description
Used	No	True/False	False	Indicates whether the output channel is being used in a program or not.
Address	No	%Q0.x	–	Displays the address of the digital output on the controller, where x represents the channel number. If the controller has 8 digital output channels, x varies from 0...7. If the controller has 16 digital output channels, x varies from 0...15. For example, `%Q0.2` is the third digital output channel on the controller.
Used by	No	*any*	*empty*	Displays the name of the component that uses the output channel. For example, if the output channel is used as status alarm, it displays **Alarm**.
Status Alarm	Yes	True/False	False	Allows you to enable or disable the status alarm for the output (`%Q0.0`...`%Q0.7`). You can configure only one output channel for the status alarm. You cannot configure an output as status alarm if the output is used in a program.
Fallback value	Yes	1 or 0	0	Specify the value to apply to this output (fallback to 0 or fallback to 1) when the logic controller enters the `STOPPED` or an exception state. The default value is 0. If **Maintain values** fallback mode is configured, the output retains its current value when the logic controller enters the `STOPPED` or an exception state. This field is disabled for the output configured as **Status Alarm**. For more details, refer to the Fallback Behavior *(see SoMachine Basic, Operating Guide)*.

Additional configuration details are displayed in the **Programming** tab. For more information, refer to Digital Outputs (%Q) *(see page 111)*.

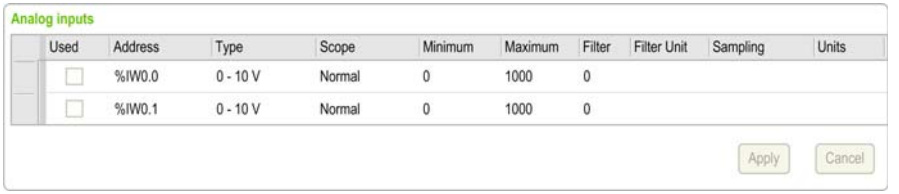
Configuring Analog Inputs

Introduction

The analog inputs do not have any configurable property in SoMachine Basic. By default, analog inputs are used as regular inputs.

Analog Inputs Configuration

This table describes how to configure the analog inputs:

Step	Action
1	Click the **Analog inputs** node in the hardware tree to display the analog input properties. This figure shows the properties of the analog inputs in the editor area: 
2	Edit the properties to configure the analog inputs. For detailed information on the analog input configuration parameters, refer to the table below.

This table describes each parameter of the analog input configuration:

Parameter	Editable	Value	Default Value	Description
Used	No	True/False	False	Indicates whether the input channel is being used in a program or not.
Address	No	%IW0.x	–	Displays the address of the analog input on the controller, where x represents the channel number. If the controller has 2 analog input channels, x is either 0 or 1. For example, `%IW0.1` is the second analog input channel on the controller.
Type	No	**0 - 10 V**	**0 - 10 V**	Indicates the channel mode. For example, **0 - 10 V** refers to the channel that can be used for an electrical input of voltage type in the range 0...10 V.
Scope	No	**Normal**	**Normal**	Indicates the range of values for a channel.
Minimum	No	0	0	Indicates the lower measurement limit.
Maximum	No	0...1023	1000	Indicates the upper measurement limit.
Filter	No	0	0	Indicates the filtering value. Multiply by the **Filter Unit** value to obtain the filtering time.

Parameter	Editable	Value	Default Value	Description
Filter Unit	No	100 ms	*empty*	Specifies the unit of time for the filtering value.
Sampling	No	–	*empty*	–
Units	No	*any*	*empty*	Indicates the unit of the analog input.

Additional configuration details are displayed in the **Programming** tab. For more information, refer to Analog Inputs (%IW) *(see page 112)*.

Configuring High Speed Counters

Introduction

You can configure high speed counters to perform any one of the following functions:

- Up/down counter
- Bi-phase counter
- Single counter
- Frequency meter

The high speed counter supports counting of digital inputs up to frequencies of 60 kHz in single word or double word computational mode.

Dedicated I/O Assignments

The `High Speed Counter` function blocks use dedicated inputs and auxiliary inputs and outputs. These inputs and outputs are not reserved for the exclusive use of `High Speed Counter` function blocks.

If `%I0.0` or `%I0.1` is in use by the program as a regular digital input, `%HSC0` is not available.

If `%I0.6` or `%I0.7` is in use by the program as a regular digital input, `%HSC1` is not available.

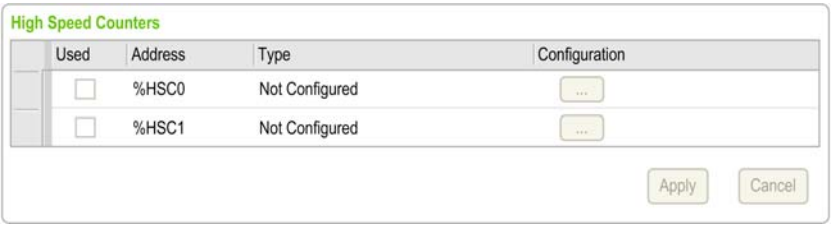
This table summarizes these assignments:

Counter Type	**Main Inputs**		**Auxiliary Inputs**		**Reflex Outputs**	
%HSC0	**%I0.0**	**%I0.1**	**%I0.2**	**%I0.3**	**%Q0.2**	**%Q0.3**
%HSC1	**%I0.6**	**%I0.7**	**%I0.5**	**%I0.4**	**%Q0.4**	**%Q0.5**
Up/Down counter	Pulse input	Direction input	Preset input*	Catch input*	Reflex output R*	Reflex output S*
Bi-phase counter	Pulse input Phase A	Pulse input Phase B	Preset input*	Catch input*	Reflex output R*	Reflex output S*
Single counter	Pulse input	Not used	Preset input*	Catch input*	Reflex output R*	Reflex output S*
Frequency meter	Pulse input	Not used	Not used	Not used	Not used	Not used

***** When not used, the input or output functions as a normal digital I/O available to be managed by the application in the main task cycle.

High Speed Counters Configuration

This table describes how to configure the high speed counters:

Step	Action
1	Click the **High Speed Counters** node in the hardware tree to display the high speed counter properties. This figure shows the properties of the high speed counters in the editor area: 
2	Edit the properties to configure the high speed counters. For detailed information on the high speed counter configuration, refer to the table below.

This table describes each parameter of the high speed counters configuration:

Parameter	Editable	Value	Default Value	Description
Used	No	True/False	False	Indicates whether the high speed counter is being used in a program or not.
Address	No	%HSCx		Displays the address of the high speed counter, where x is the object number. For example, `%HSC1` is the address of the second high speed counter on the controller.
Type	Yes	**Not Configured** **Up/Down Counter** **Bi-Phase Counter** **Single Counter** **Frequency Meter**	**Not Configured**	Allows you to select the counter operational mode from the drop-down list. For more information on counter types, refer to High Speed Counters Function Blocks (%HSC) *(see page 122)*.

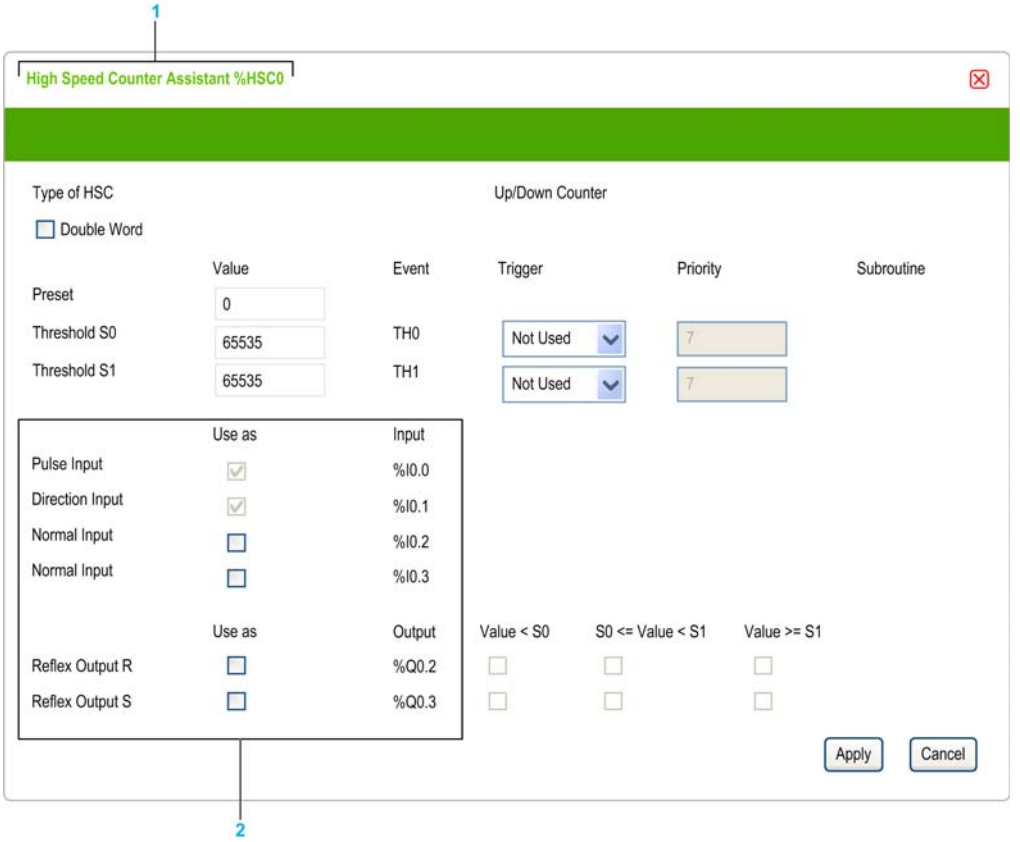
Parameter	Editable	Value	Default Value	Description
Configuration	Yes	**[...]** (Button)	Disabled	Allows you to configure the high speed counter parameters using an assistant window. This button is enabled only if the high speed counter function is selected from the list. When the counter type is **Not Configured**, the assistant configuration button is disabled. A window **High Speed Counter Assistant (%HSCx)** appears when you click the configuration button, where x is the counter number on the controller.

For details on the configuration of the **Up/Down Counter**, **Bi-Phase Counter**, and **Single Counter**, refer to Configuring High Speed Counters *(see page 75)*.

For details on the configuration of the **Frequency Meter**, refer to Configuring Frequency Meter *(see page 79)*.

Configuring High Speed Counters

This figure shows an instance of the assistant window for %HSC0 configured as the **Up/Down Counter**:



Item	Description
1	Displays the title of the assistant dialog window. If you are configuring the counter %HSC0, the window title appears as **High Speed Counter Assistant (%HSC0)** and for the counter %HSC1, the window title appears as **High Speed Counter Assistant (%HSC1)**.
2	Displays the dedicated inputs, auxiliary inputs, and reflex outputs. Properties in this area of the assistant window are different for each counter type. However, the other parameters of this window remain same in all the cases.parameters remain same. These properties depend upon the selected counter type for both %HSC0 and %HSC1. For more details, refer to the Dedicated I/O Assignments *(see page 72)* section.

This table describes each parameter of the assistant screen for the counters **Up/Down Counter**, **Bi-Phase Counter**, and **Single Counter**, for both %HSC0 and %HSC1:

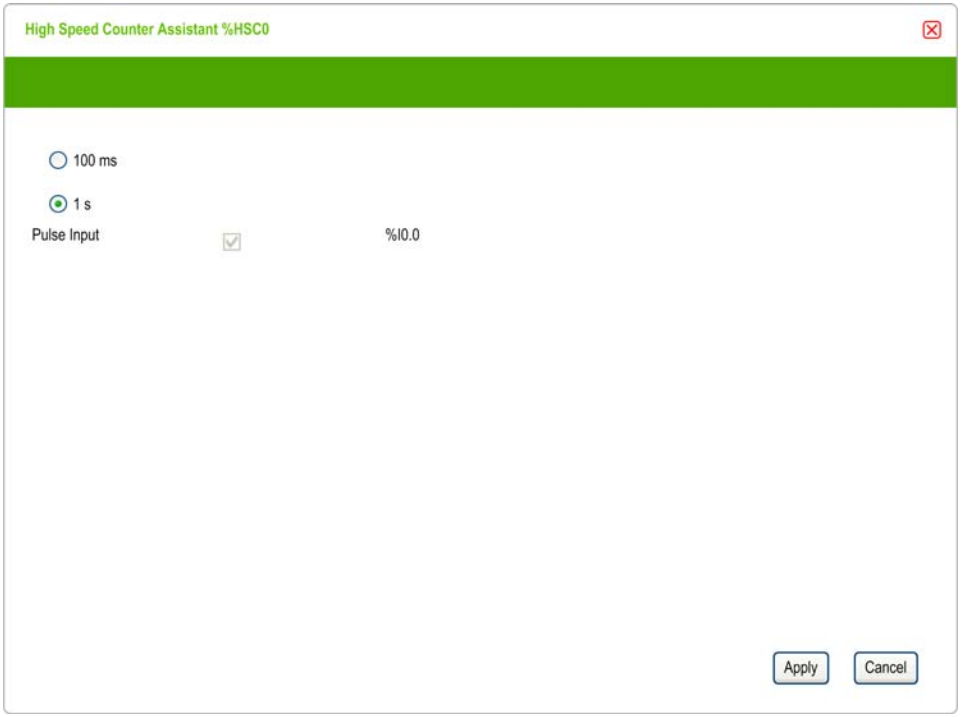
Parameter	Editable	Value	Default Value	Description
Double Word	Yes	True/False	False	Allows you to toggle between input data size of Word (16 bits) and Double Word (32 bits). By default, this parameter is disabled, which indicates that the current data size is Word (16 bits). Enabling this field changes the data size to Double Word (32 bits).
Preset	Yes	0...65535 (Word) 0...4294967295 (Double Word)	0 (Word) 0 (Double Word)	Allows you to specify the preset value for the counting functions.
Threshold S0	Yes	0...65535 (Word) 0...4294967295 (Double Word)	65535 (Word) 4294967295 (Double Word)	Allows you to specify the value of the HSC flag S0 that contains the value of the threshold TH0.
Threshold S1	Yes	0...65535 (Word) 0...4294967295 (Double Word)	0...65535 (Word) 0...4294967295 (Double Word)	Allows you to specify the value for the HSC flag S1 that contains the value of the threshold TH1.
Trigger	Yes	**Not Used** **Falling Edge** **Rising Edge** **Both edges**	**Not Used**	Allows you to select a triggering function for an event (for both threshold TH0 and TH1) from the drop-down list. If you select a triggering function from the drop-down list (other than **Not Used**), the **Priority** parameter enables for editing to set the priority of the event.
Priority	Yes	0...7	7	Allows you to set the priority of the triggering function of an event (for both threshold TH0 and TH1). This field enables only when you select a triggering function for the event.
Subroutine	No	*any*	*empty*	Displays the subroutine associated with an input configured as an event (for both threshold TH0 and TH1).

Parameter	Editable	Value	Default Value	Description	
When configuring for **Up/Down Counter**:				For **%HSC0**:	For **%HSC1**:
Pulse Input	No	True/False	True	%I0.0 is used as pulse input.	%I0.6 is used as pulse input.
Direction Input	No	True/False	True	%I0.1 is used as directional input.	%I0.7 is used as directional input.
When configuring for **Bi-phase Counter**:				For **%HSC0**:	For **%HSC1**:
Pulse Input Phase A	No	True/False	True	%I0.0 is used as pulse input for phase A.	%I0.6 is used as pulse input for phase A.
Pulse Input Phase B	No	True/False	True	%I0.1 is used as pulse input for phase B.	%I0.7 is used as pulse input for phase B.
When configuring for **Single Counter**:				For **%HSC0**:	For **%HSC1**:
Pulse Input	No	True/False	True	%I0.0 is used as pulse input.	%I0.6 is used as pulse input.
Normal Input	No	True/False	True	%I0.1 is used as normal input.	%I0.7 is used as normal input.
When configuring for **Up/Down Counter**, **Bi-phase Counter**, and **Single Counter**:				For **%HSC0**:	For **%HSC1**:
Normal Input	Yes	True/False	False	%I0.2 is used as normal input. Click the **Use as** check box to use this input as **Preset Input**.	%I0.5 is used as normal input. Click the **Use as** check box to use this input as **Preset Input**.
Normal Input	Yes	True/False	False	%I0.3 is used as normal input. Click the **Use as** check box to use this input as **Catch Input**.	%I0.4 is used as normal input. Click the **Use as** check box to use this input as **Catch Input**.
Reflex Output R	Yes	True/False	False	Allows you to enable or disable the reflex output at the address %Q0.2.	Allows you to enable or disable the reflex output at the address %Q0.4.

Parameter	Editable	Value	Default Value	Description	
Reflex Output S	Yes	True/False	False	Allows you to enable or disable the reflex output at the address %Q0.3.	Allows you to enable or disable the reflex output at the address %Q0.5.
Value < S0	Yes	True/False	False	Allows you to enable or disable the condition in which the counter is constantly compared to the output value to set the reflex output when the output value is less than the value of HSC flag S0.	
S0 <= Value < S1	Yes	True/False	False	Allows you to enable or disable the condition in which the counter is constantly compared to the output value to set the reflex output when the output value is greater than or equals to the value of the HSC flag S0 and the output value is less than the value of the HSC flag S1.	
Value >= S1	Yes	True/False	False	Allows you to enable or disable the condition in which the counter is constantly compared to the output value to set the reflex output when the output value is greater than or equals to the value of HSC flag S1.	

Configuring Frequency Meter

This figure shows the **High Speed Counter Assistant (%HSC0)** window for the counter type **Frequency Meter**:



This table describes each parameter of the **High Speed Counter Assistant (%HSCx)** window for the counter type **Frequency Meter**:

Parameter	Editable	Value	Default Value	Description
100 ms	Yes(1)	True/False	False	Allows you to select the time base of 100 ms to measure the frequency between 100 Hz to 60 kHz.
1 s	Yes(1)	True/False	True	Allows you to select the time base of 1 ms to measure the frequency between 100 Hz to 60 kHz.
Pulse Input	No	True/False	True	Indicates that the pulse input is used at the address `%I0.0` for `%HSC0` or `%I0.6` for `%HSC1`.
(1) By default, the time base value is set to 1 s. You can select only one time base value for the counter.				

Additional configuration details are displayed in the **Programming** tab.

For more details on the `High Speed Counter` function block, refer to High Speed Counter Function Block (%HSC) *(see page 122)*.

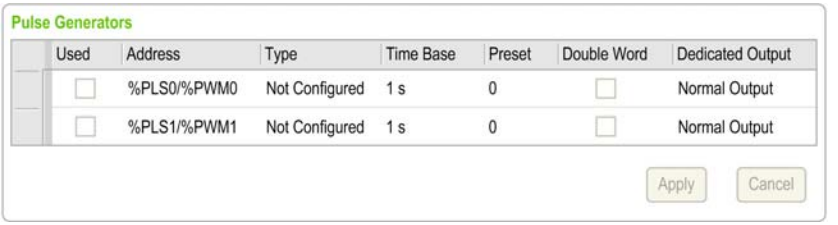
Configuring Pulse Generators

Introduction

The pulse generator function blocks, Pulse (PLS) and Pulse Width Modulation (PWM) are used to generate square wave signals on dedicated output channels `%Q0.0` or `%Q0.1`, with variable width and duty cycle.

Pulse Generators Configuration

This table describes how to configure the pulse generators:

Step	Action
1	Click the **Pulse Generators** node in the hardware tree to display the pulse generator properties. This figure shows the properties of the pulse generators in the editor area: 
2	Edit the properties to configure the pulse generators. For detailed information on the pulse generator configuration parameters, refer to the table below.

This table describes each parameter of the pulse generator configuration:

Parameter	Editable	Value	Default Value	Description
Used	No	True/False	False	Indicates whether the pulse generated output is being used in a program or not.
Address	No	%PLSx %PWMx	%PLSx/%PWMx	Displays the address of the `Pulse` output or `Pulse Width Modulation` output, where x is the output number. If the controller has 2 pulse generators, then the value for x is 0 and 1. For example, `%PLS0` is the address of the first pulse output on the controller. If you select the output type **PLS**, the **Address** field displays only %PLSx and if you select **PWM**, it displays only %PWMx.

Parameter	Editable	Value	Default Value	Description
Type	Yes	**Not Configured** **PLS** **PWM**	**Not Configured**	Allows you to select the type of the pulse train output. You can select **PLS** for pulse output or **PWM** for pulse width modulation output. If you select **PLS** or **PWM** for an input, the following parameters enables: ● **Time Base** ● **Preset** ● **Double Word**
Time Base	Yes	**0.1 ms** **1 ms** **10 ms** **1 s**	**1 s**	Allows you to select the time base for the frequency measurement.
Preset	Yes	Refer to the table below for complete range of PLS and PWM pulse generators.	0	Allows you to specify the preset value for the pulse train output.
Double Word	Yes	False	True/False	Allows you to toggle between the data size of Word (16 bits) and Double Word (32 bits). By default, this parameter is disabled, which indicates that the current data size is Word (16 bits). Enabling this field changes the data size to Double Word (32 bits).
Dedicated Output	No	**Normal Output** **Pulse Output**	**Normal Output**	Indicates the output type of the pulse generator. When the output channel is configured for the pulse or the pulse width modulation, it displays **Pulse Output** and when not used for pulse output, it displays **Normal Output**.

This table shows the range of values of the **Preset** parameter:

Type	Time Base	Preset Value Range
PLS	0.1 ms	1...20000
	1 ms	1...2000
	10 ms	1...200
	1 s	1 or 2
PWM	0.1 ms	1...10000
	1 ms	1...1000
	10 ms	1...100
	1 s	1

Additional configuration details are displayed in the **Programming** tab.

For more details on the `Pulse` function block, refer to Pulse (%PLS) *(see page 135)*.

For more details on the `Pulse Width Modulation` function block, refer to Pulse Width Modulation (%PWM) *(see page 142)*.

Chapter 5
I/O Bus Configuration

Overview

This chapter describes how to configure the I/O bus (expansion modules) of the M221 Logic Controller.

What Is in This Chapter?

This chapter contains the following topics:

I/O Configuration General Practices

Match Software and Hardware Configuration

The I/O that may be embedded in your controller is independent of the I/O that you may have added in the form of I/O expansion. It is crucial that the logical I/O configuration within your program matches the physical I/O configuration of your installation. If you add or remove any physical I/O to or from the I/O expansion bus, or, depending on the controller reference, to or from the controller (in the form of cartridges), it is imperative that you update your application configuration. This is also true for any field bus devices you may have in your installation. Otherwise, there is the possibility that the I/O expansions will no longer function while the embedded I/O that may be present in your controller will continue to operate.

⚠ WARNING

UNINTENDED EQUIPMENT OPERATION

Update the configuration of your program each time you add or delete any type of I/O expansions, or you add or delete any devices on your field bus.

Failure to follow these instructions can result in death, serious injury, or equipment damage.

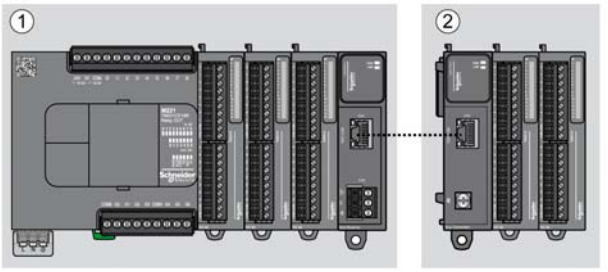
Maximum Hardware Configuration

Introduction

The M221 Logic Controller is a control system that offers an all-in-one solution with optimized configurations and an expandable architecture.

Local and Remote Configuration Principle

The following figure defines the local and remote configurations:



(1) Local configuration
(2) Remote configuration

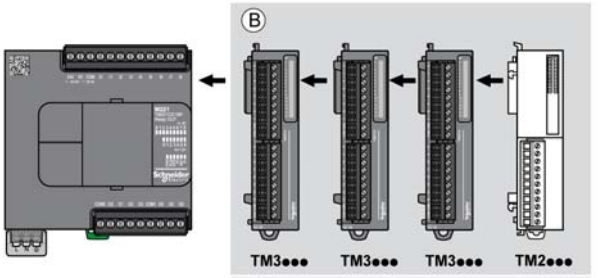
M221 Logic Controller Local Configuration Architecture

Optimized local configuration and flexibility are provided by the association of:

- M221 Logic Controller
- TM3 expansion modules
- TM2 expansion modules

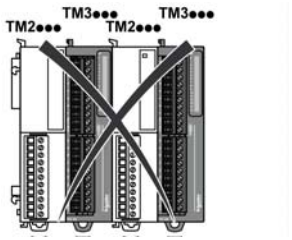
Application requirements determine the architecture of your M221 Logic Controller configuration.

The following figure represents the components of a local configuration:



(B) Expansion modules (see maximum number of modules)

NOTE: It is prohibited to mount a TM2 module before any TM3 module as indicated in the following figure:



M221 Logic Controller Remote Configuration Architecture

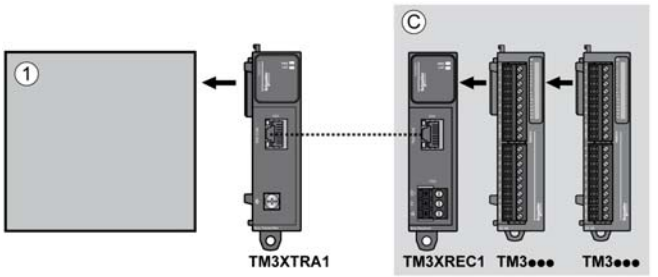
Optimized remote configuration and flexibility are provided by the association of:

- M221 Logic Controller
- TM3 expansion modules
- TM3 transmitter and receiver modules

Application requirements determine the architecture of your M221 Logic Controller configuration.

NOTE: You cannot use TM2 modules in configurations that include the TM3 transmitter and receiver modules.

The following figure represents the components of a remote configuration:



(1) Logic controller and modules
(C) Expansion modules (7 maximum)

Maximum Number of Modules

The following table shows the maximum configuration supported:

References	Maximum	Type of Configuration
TM221C16• TM221CE16•	4 TM3 / TM2 expansion modules	Local
TM221C24• TM221CE24• TM221C40• TM221CE40• TM221M16R• TM221ME16R• TM221M16T• TM221ME16T• TM221M32TK TM221ME32TK	7 TM3 / TM2 expansion modules	Local
TM3XREC1	7 TM3 expansion modules	Remote
NOTE: TM3 transmitter and receiver modules are not included in a count of the maximum number of expansion modules.		

NOTE: The configuration with its TM3 and TM2 expansion modules is validated by SoMachine Basic software in the **Configuration** window.

NOTE: In some environments, the maximum configuration populated by high consummation modules, coupled with the maximum distance allowable between the TM3 transmitter and receiver modules, may present bus communication issues although the SoMachine Basic software allows for the configuration. In such a case you will need to analyze the consummation of the modules chosen for your configuration, as well as the minimum cable distance required by your application, and possibly seek to optimize your choices.

Current Supplied to the I/O Bus

The following table shows the maximum current supplied by the controllers to the I/O Bus:

Reference	IO Bus 5 Vdc	IO Bus 24 Vdc
TM221C16R TM221CE16R	325 mA	120 mA
TM221C16T TM221CE16T	325 mA	148 mA
TM221C24R TM221CE24R	520 mA	160 mA
TM221C24T TM221CE24T	520 mA	200 mA
TM221C40R TM221CE40R	520 mA	240 mA

Reference	IO Bus 5 Vdc	IO Bus 24 Vdc
TM221C40T TM221CE40T	520 mA	304 mA
TM221M16R• TM221ME16R•	520 mA	460 mA
TM221M16T• TM221ME16T•	520 mA	492 mA
TM221M32TK TM221ME32TK	520 mA	484 mA

NOTE: Expansion modules consume current from the 5 Vdc and 24 Vdc supplied to the I/O Bus. Therefore, the current delivered by the logic controller to the I/O Bus defines the maximum number of expansion modules that can be connected to the I/O Bus (validated by SoMachine Basic software in the **Configuration** window).

Configuring Cartridges and Expansion Modules

Introduction

In your project, you can add the following devices to the controller:

- TMC2 Cartridges
- TM3 Digital I/O Modules
- TM3 Expert I/O Modules
- TM2 Digital I/O Modules
- TM2 Analog I/O Modules

TMC2 Cartridges

For more information about cartridge configuration, refer to the following programming and hardware guides:

Cartridge Type	Hardware Guide	Programming Guide
TMC2 Cartridges	TMC2 Cartridges Hardware Guide	TMC2 Cartridges Programming Guide

TM3 Expansion Modules

For more information about module configuration, refer to the following programming and hardware guides of each expansion module type:

Expansion Module Type	Hardware Guide	Programming Guide
TM3 Digital I/O Expansion Modules	TM3 Digital I/O Expansion Modules Hardware Guide	TM3 Expansion Modules Programming Guide
TM3 Analog I/O Expansion Modules	TM3 Analog Modules Hardware Guide	
TM3 Expert I/O Expansion Modules	TM3 Expert I/O Modules Hardware Guide	
TM3 Transmitter and Receiver Modules	TM3 Transmitter and Receiver Modules Hardware Guide	

TM2 Expansion Modules

For more information about module configuration, refer to the programming and hardware guides of each expansion module type:

Expansion Module Type	Hardware Guide	Programming Guide
TM2 Digital I/O Modules	TM2 Digital I/O Modules Hardware Guide	TM2 Expansion Modules Programming Guide
TM2 Analog I/O Modules	TM2 Analog I/O Modules Hardware Guide	

Chapter 6
Embedded Communication Configuration

Overview

This chapter describes how to configure the communication features of the M221 Logic Controller.

What Is in This Chapter?

This chapter contains the following sections:

Section 6.1
Ethernet Configuration

What Is in This Section?

This section contains the following topics:

Configuring Ethernet Network

Introduction

You can configure the TCP/IP connection to the logic controller by configuring the Ethernet network. The Ethernet establishes a local area network (LAN) between the logic controller and the other devices. The Ethernet configuration provides you the ability to configure the IP address of the network device.

NOTE: The controller-PC link uses the TCP/IP protocol. It is required for this protocol to be installed on the PC.

You can obtain the Ethernet IP address by the following protocols:
- Dynamic Host Configuration Protocol (DHCP)
- Bootstrap Protocol (BOOTP)

You can also specify the Ethernet IP address by specifying the following addresses:
- IP address
- Subnet mask
- Gateway address

Ethernet Configuration

This table describes how to configure the Ethernet:

Step	Action
1	Click the **ETH1** node in the hardware tree to display the Ethernet properties. This figure shows the Ethernet properties in the editor area: 
2	Edit the properties to configure the Ethernet. For detailed information on the Ethernet configuration parameters, refer to the table below.

This table describes each parameter of the Ethernet configuration:

Parameter	Editable	Value	Default Value	Description
Ethernet				
Device name	No	*any*	**M221** (if the controller used in the cofiguration is M221 Logic Controller)	Displays the name of the device that is connected with the Ethernet network.
IP address by DHCP	Yes(1)	True/False	False	Allows you to obtain the IP address from the DHCP server on the network.
IP address by BOOTP	Yes(1)	True/False	False	Allows you to obtain the IP address from the Boot PROM configuration server on the network.
Fixed IP address	Yes(1)	True/False	True	Allows you to specify the IP address manually for host or network interface identification.
IP address	Yes(2)	*w.x.y.z*(3)	0.0.0.0	Allows you to specify the IP address of the device in the Ethernet network. Assigning 0.0.0.0 as IP address for the M221 Logic Controller forces the firmware to generate an IP address from the MAC address.
Subnet mask	Yes(2)	*w.x.y.z*(3)	0.0.0.0	Allows you to specify the address of the subnetwork to authorize a group of devices for data exchange. It determines which bits in an IP address correspond to the network address and which bits correspond to the subnet portions of the address.
Gateway address	Yes(2)	*w.x.y.z*(3)	0.0.0.0	Allows you to specify the IP address of the node (a router) on a TCP/IP network that serves as an access point to another network.
Transfer Rate	No	–	**Auto**	Displays the transfer rate for obtaining the IP address.
Security Parameters				
Programming protocol enabled	Yes	True/False	True	Allows you to enable or disable programming protocol for communication with the other devices in the network.

(1) You can select any one option for IP addressing. Selecting any one option, disables the other options.
(2) These options are enabled only if you select the option **Fixed IP Address** for IP addressing.
(3) *w*, *x*, *y*, and *z* are the bytes that store the address and each byte can store a value in the range 0...255.

Parameter	Editable	Value	Default Value	Description
Ethernet/IP adapter enabled	Yes	True/False	True	Allows you to enable or disable Ethernet/IP adapter to connect to a network for data exchange.
Modbus server enabled	Yes	True/False	True	Allows you to enable or disable Modbus server for serial device connectivity.
Auto discovery protocol enabled	Yes	True/False	True	Allows you to enable or disable auto discovery protocol to automatically detect the devices in a network.
(1) You can select any one option for IP addressing. Selecting any one option, disables the other options. **(2)** These options are enabled only if you select the option **Fixed IP Address** for IP addressing. **(3)** *w*, *x*, *y*, and *z* are the bytes that store the address and each byte can store a value in the range 0...255.				

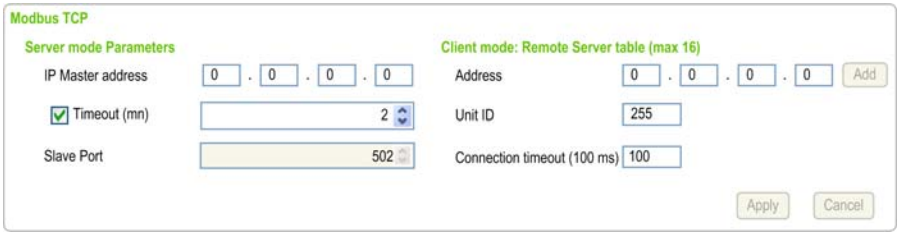
Configuring Modbus TCP

Introduction

You can configure the Ethernet port to enable the embedded Modbus TCP server giving the logic controller Modbus TCP abilities.

Modbus TCP Configuration

This table describes how to configure the Modbus TCP:

Step	Action
1	Click the **Modbus TCP** node that appears below the **ETH1** node in the hardware tree to display the Ethernet/IP adapter properties. This figure shows the properties of the Ethernet/IP adapter in the editor area: 
2	Edit the properties to configure the Modbus TCP. For detailed information on the Modbus TCP configuration parameters, refer to the table below.

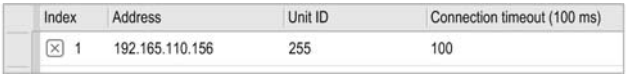
This table describes each parameter of the Modbus TCP configuration:

Parameter	Editable	Value	Default Value	Description
Server mode Parameters				
IP Master address	Yes	*w.x.y.z*(1)	0.0.0.0	Allows you to specify the IP address of the preferred remote server (master server).
Timeout (mn)	Yes	1...200	2	Allows you to specify the connection timeout duration of 1...200 min. You can enable or disable this option using the checkbox.
Slave Port	No	502	502	Indicates the port number of the server IP address.
Client mode: Remote Server table (max 16)				
(1) *w*, *x*, *y*, and *z* are the bytes that store the address and each byte can store a value in the range 0...255.				

Parameter	Editable	Value	Default Value	Description
Address	Yes	*w.x.y.z*(1)	0.0.0.0	Allows you to specify the IP address of the remote server. Also, refer to Adding Remote Servers *(see page 97)*.
Unit ID	Yes	0...255	255	Allows you to specify the unit ID of the remote server.
Connection timeout (100 ms)	Yes	0...65535	100	Allows you to specify the connection timeout duration.
(1) *w*, *x*, *y*, and *z* are the bytes that store the address and each byte can store a value in the range 0...255.				

Adding Remote Servers

This table describes how to add a remote server for Modbus TCP:

Step	Action
1	Enter the IP address in the **Address** field.
2	Enter the value for **Unit ID** and **Connection timeout (100 ms)**.
3	Click the **Add** button. **Result**: A list of remote servers that you have added, appears on the screen. This figure shows the table listing the remote servers: 

This table describes each column of the table listing the remote servers:

Parameter	Editable	Value	Default Value	Description
Index	No	0...16	–	Displays the index number of the servers which are remotely connected.
Address	Yes	*w.x.y.z*(1)	0.0.0.0	Displays the IP address of the remote server.
Unit ID	Yes	0...255	255	Displays the unit ID of the remote server.
Connection timeout (100 ms)	Yes	0...65535	100	Displays the connection timeout duration.
(1) *w*, *x*, *y*, and *z* are the bytes that store the address and each byte can store a value in the range 0...255.				

Click the close button in the row to remove a remote server.

Double-click the remote server entry in a row to edit the values.

Section 6.2
Serial Line Configuration

Configuring Serial Line

Introduction

The M221 Logic Controller references are equipped with at least 1 serial line. The controller references without the Ethernet feature support 2 serial lines:

- SL1 (serial line)
- SL2 (serial line)

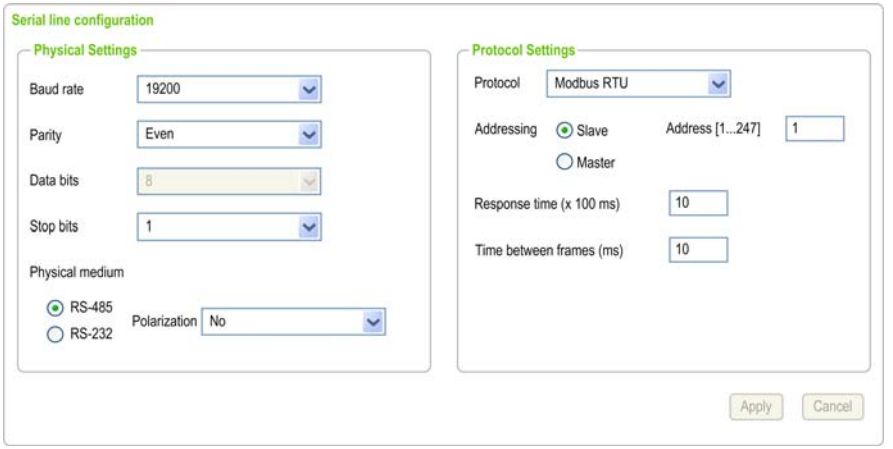
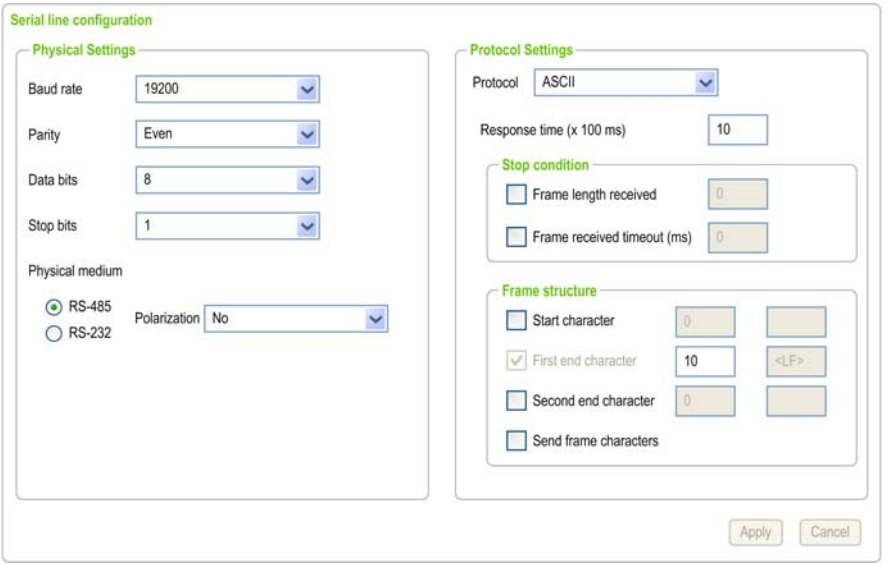
Each serial line can be configured for any one of the following protocols:

- Modbus RTU
- Modbus ASCII
- ASCII

You can configure both physical and protocol settings for the serial line. Serial lines are configured for the Modbus RTU protocol by default.

Serial Line Configuration

This table describes how to configure the serial line:

Step	Action
1	Click the **SL1 (Serial line)** or **SL2 (Serial line)** node in the hardware tree to display the serial line properties. This figure shows the properties of the serial line for **Modbus RTU** and **Modbus ASCII** protocols:  This figure shows the properties of the serial line for **ASCII** protocol: 

Step	Action
2	Edit the properties to configure the serial line. For detailed information on the serial line configuration parameters, refer to the table below.

This table describes each parameter of the serial line:

Parameter	Editable	Value	Default Value	Description
Physical settings				
Baud rate	Yes	**1200** **2400** **4800** **9600** **19200** **38400** **57600** **115200**	**19200**	Allows you to select the data transmission rate (bits per second) for the modem from the drop-down list.
Parity	Yes	**None** **Even** **Odd**	**Even**	Allows you to select the parity of the transmitted data for error detection. Parity is a method of error detection in transmission. When parity is used with a serial port, an extra data bit is sent with each data character, arranged so that the number of 1 bits in each character, including the parity bit, is always odd or always even. If a byte is received with the wrong number of 1 bits, the byte is corrupt. However, an even number of detected errors can pass the parity check.
Data bits	Yes (only for the **ASCII** protocol	**7** **8**	**8**	Allows you to select the data bit from the drop-down list. The number of data bits in each character can be 7 (for true ASCII) or 8 (for any kind of data, as this matches the size of a byte). 8 data bits are almost universally used in all applications.
Stop bits	Yes	**1** **2**	**1**	Allows you to select the stop bit from the drop-down list. Stop bit is a bit indicating the end of a byte of data. For electronic devices usually 1 stop bit is used. For slow devices like electromechanical teleprinters, 2 stop bits are used.

Parameter	Editable	Value	Default Value	Description
Physical medium	Yes	**RS-485** True/False **RS-232** True/False	**RS-485** True	Allows you to select the physical medium for communication. You can select either **RS485** or **RS232** medium for serial line 1. For serial line 2, only **RS485** medium is available. A physical medium in data communications is the transmission path over which a signal propagates. It is an interface for interconnection of devices with the logic controller.
Polarization	Yes (for cartridges only) No (for the controller)	**Yes** **No**	**No**	Polarization resistors are integrated in the cartridge module. For the controller, this parameter is disabled and for the cartridges, this parameter allows you to switch on or off polarization.
Protocol settings				
Protocol	Yes	**Modbus RTU** **Modbus ASCII** **ASCII**	**Modbus RTU**	Allows you to select the protocol transmission mode for communication from the drop-down list. Protocol advanced parameters are displayed based on the selected protocol. Refer to the following figures and tables.
Protocol settings for the **Modbus RTU** and **Modbus ASCII** protocols:				
Addressing	Yes	**Slave** True/False **Master** True/False	**Slave** True	Allows you to select the addressing mode. You can only select either of the **Slave** or **Master** addressing. Enabling any of the addressing mode, disables the other one.
Address [1...247]	Yes	1...247	1	Allows you to specify the address ID of the slave. **NOTE:** This field is displayed only for the addressing of the slave. For master, this field does not appear on the screen.
Response time (x 100 ms)	Yes	10...255 ms	10	Allows you to specify the response time of the protocol to the queries.
Time between frames (ms)	Yes	3...255 ms	10	Allows you to specify the time between frames of the protocol.
Protocol settings for the **ASCII** protocol:				
Response time (x 100 ms)	Yes	10...255 ms	10	Allows you to specify the response time of the protocol to the queries.

Parameter	Editable	Value	Default Value	Description
Stop condition				
Frame length received	Yes (only if the check box is selected)	1...255	0 (if check box is not selected) 1 (if check box is selected)	Allows you to specify the length of the received frame. **NOTE:** You can configure only one parameter for stop condition that is either **Frame length received** or **Frame received timeout (ms)**.
Frame received timeout (ms)	Yes (only if the check box is selected)	1...255	0 (if check box is not selected) 10 (if check box is selected)	Allows you to specify the timeout duration for the received frame.
Frame structure				
Start character	Yes (only if the check box is selected)	1...255	0 (if check box is not selected) 58 (if check box is selected)	Allows you to specify the start character of the frame. The ASCII character corresponding to the start character value is displayed on right-hand side of the value field.
First end character	Yes	1...255	0 (if check box is not selected) 10 (if check box is selected)	Allows you to specify the first end character of the frame. **NOTE:** To be able to enable or disable the **First end character**, configure at least one stop condition parameter. The ASCII character corresponding to the first end character value is displayed on right-hand side of the value field.
Second end character	Yes (only if the check box is selected)	1...255	0 (if check box is not selected) 10 (if check box is selected)	Allows you to specify the second end character of the frame. **NOTE:** This field is disabled with the disabled **First end character** parameter. The ASCII character corresponding to the second end character value is displayed on right-hand side of the value field.
Send frame characters	Yes	True/False	False	Allows you to enable or disable sending first end character of the frame to the ASCII protocol.

Part III
Programming the M221 Logic Controller

Overview

This part provides information about the system and I/O objects specific to the M221 Logic Controller. These objects are displayed in the **Programming** tab.

For descriptions of all other objects, refer to SoMachine Basic Generic Functions Library Guide.

What Is in This Part?

This part contains the following chapters:

Chapter 7
How to Use the Source Code Examples

How to Use the Source Code Examples

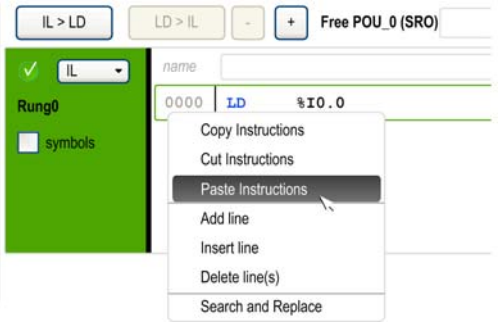
Overview

Except where explicitly mentioned, the source code examples contained in this book are valid for both the Ladder Diagram and Instruction List programming languages. A complete example may require more than one rung.

Reversibility Procedure

Only Instruction List source code is shown in this book.

To obtain the equivalent Ladder Diagram source code:

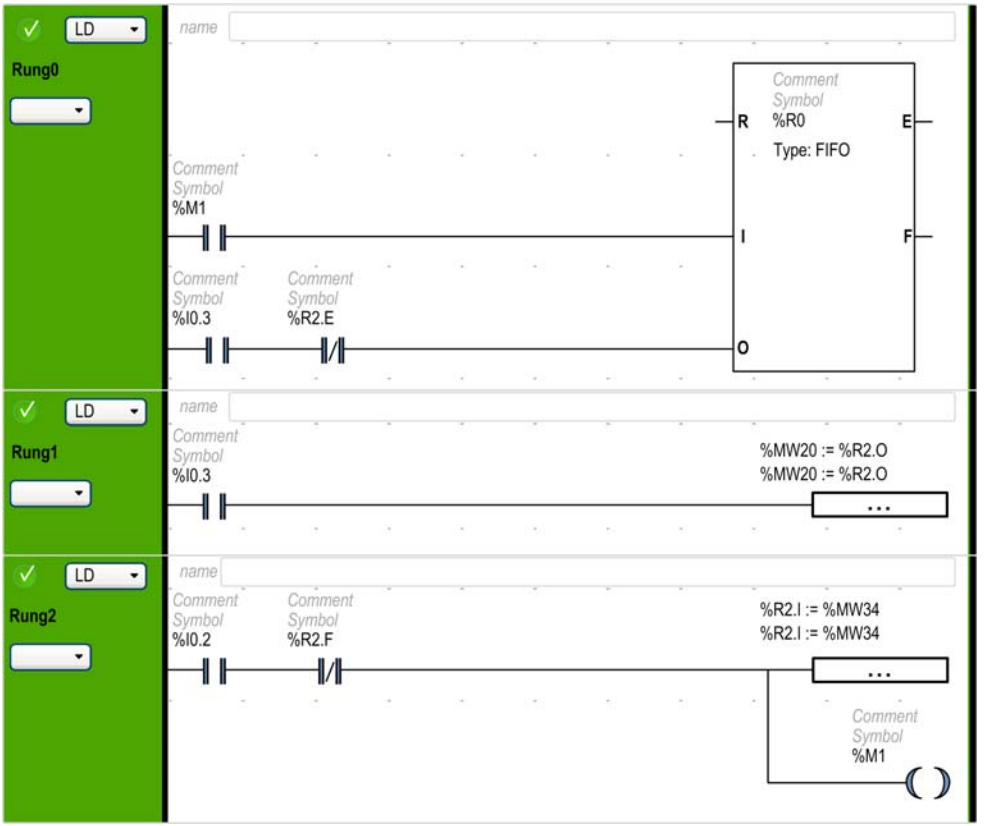
Step	Action
1	In SoMachine Basic, create a new POU containing an empty rung.
2	In this rung, click the **LD > IL** button to display Instruction List source code.
3	Select and copy (**Ctrl+C**) the source code for the first rung of the sample program.
4	Right-click on the line number **0000** of the first instruction and choose **Paste Instructions** to paste the source code into the rung:  **NOTE:** Remember to delete the **LD** instruction from the last line of the rung if you have pasted the instructions by inserting the line(s) before the default LD operator.
5	Click the **IL > LD** button to display the Ladder Diagram source code.
6	Repeat the previous steps for any additional rungs in the sample program. Click  on the toolbar to add new rungs.

Example

Instruction List program:

Rung	Source Code
0	BLK %R0 LD %M1 I LD %I0.3 ANDN %R2.E O END_BLK
1	LD %I0.3 [%MW20:=%R2.O]
2	LD %I0.2 ANDN %R2.F [%R2.I:=%MW34] ST %M1

Corresponding Ladder Diagram:



Chapter 8
I/O Objects

What Is in This Chapter?

This chapter contains the following topics:

Digital Inputs (%I)

Introduction

Digital input bit objects are the image of digital inputs on the logic controller.

Displaying Digital Input Properties

Follow these steps to display properties of the digital inputs:

Step	Action
1	Select the **Tools** tab in the left-hand area of the **Programming** window.
2	Click **I/O objects** →**Digital inputs**. **Result**: Digital input properties appear on the screen.

Digital Inputs Properties

This table describes each property of the digital input:

Parameter	Editable	Value	Default Value	Description
Used	No	True/False	False	Indicates whether the input channel is being referenced in a program.
Address	No	%I0.i	–	Displays the address of the digital input on the controller, where i represents the channel number. If the controller has n digital input channels, the value of i is given as 0...n-1. For example, `%I0.2` is the digital input at the digital input channel number 2 of the logic controller.
Symbol	Yes	–	–	The symbol associated with this address. Double-click in the **Symbol** column and type the name of the symbol to associate with this input. If a symbol already exists, you can right-click in the **Symbol** column and choose **Search and Replace** to find and replace occurrences of this symbol throughout the program and/or program comments.
Comment	Yes	–	–	A comment associated with this address. Double-click in the **Comment** column and type an optional comment to associate with this channel.

Digital Outputs (%Q)

Introduction

Digital output bit objects are the image of digital outputs on the logic controller.

Displaying Digital Output Properties

Follow these steps to display properties of the digital outputs:

Step	Action
1	Select the **Tools** tab in the left-hand area of the **Programming** window.
2	Click **I/O objects** →**Digital outputs**. **Result**: Digital output properties appear on the screen.

Digital Outputs Properties

This table describes each property of the digital output:

Parameter	Editable	Value	Default Value	Description
Used	No	True/False	False	Indicates whether the output channel is being referenced in a program.
Address	No	%Q0.i	–	Displays the address of the digital output on the controller, where i represents the channel number. If the controller has n digital output channels, the value of i is given as 0...n-1. For example, `%Q0.3` is the digital output at the digital output channel number 3 of the logic controller.
Symbol	Yes	–	–	The symbol associated with this address. Double-click in the **Symbol** column and type the name of the symbol to associate with this output. If a symbol already exists, you can right-click in the **Symbol** column and choose **Search and Replace** to find and replace occurrences of this symbol throughout the program and/or program comments.
Comment	Yes	–	–	The comment associated with this address. Double-click in the **Comment** column and type an optional comment to associate with this channel.

Analog Inputs (%IW)

Introduction

Analog input word objects are the digital values of an analog signal connected to the logic controller.

Two 0-10V analog inputs are embedded in the logic controller. The embedded analog inputs use a 10 bits resolution converter so that each increment is approximately 10 mV ($10V/2^{10}$-1). Once the system detects the value 1023, the channel is considered to be saturated.

Refer to M221 Hardware Guide and TMC2 Cartridges Hardware Guide used in the configuration for more details.

Displaying Analog Input Properties

Follow these steps to display properties of the analog inputs:

Step	Action
1	Select the **Tools** tab in the left-hand area of the **Programming** window.
2	Click **I/O objects** →**Analog inputs**. **Result**: Analog input properties appear on the screen.

Analog Inputs Properties

This table describes each property of the analog input:

Parameter	Editable	Value	Default Value	Description
Used	No	True/False	False	Indicates whether the input channel is being referenced in a program.
Address	No	%IW0.i	–	Displays the address of the embedded analog input on the controller, where i represents the channel number. If the controller has n analog input channels, the value of i is given as 0...n-1. For example, `%IW0.1` is the analog input at the analog input channel number 1 of the logic controller.
		%IW0.x0y	–	Displays the address of the analog output channel on the cartridge, where x is the cartridge number and and y is the channel number.

Parameter	Editable	Value	Default Value	Description
Symbol	Yes	–	–	The symbol associated with this address. Double-click in the **Symbol** column and type the name of the symbol to associate with this input. If a symbol already exists, you can right-click in the **Symbol** column and choose **Search and Replace** to find and replace occurrences of this symbol throughout the program and/or program comments.
Comment	Yes	–	–	The comment associated with this address. Double-click in the **Comment** column and type a comment to associate with this address.

Analog Outputs (%QW)

Introduction

Analog output word objects are the digital values of the analog signals recieved from the logic controller using cartridges.

Two 0-10 V analog outputs and two 4-20 mA analog outputs are embedded in the cartridges TMC2AQ2C and TMC2AQ2V respectively.

Refer to TMC2 Cartridges Hardware Guide used in the configuration for more details.

Displaying Analog Output Properties

Follow these steps to display properties of the analog outputs:

Step	Action
1	Select the **Tools** tab in the left-hand area of the **Programming** window.
2	Click **I/O objects** →**Analog outputs**. **Result**: Analog output properties appear on the screen.

Analog Outputs Properties

This table describes each property of the analog output:

Parameter	Editable	Value	Default Value	Description
Used	No	True/False	False	Indicates whether the output channel is being referenced in a program.
Address	No	%QW0.x0y	–	Displays the address of the analog output channel on the cartridge, where x is the cartridge number and and y is the channel number.
Symbol	Yes	–	–	The symbol associated with this address. Double-click in the **Symbol** column and type the name of the symbol to associate with this output. If a symbol already exists, you can right-click in the **Symbol** column and choose **Search and Replace** to find and replace occurrences of this symbol throughout the program and/or program comments.
Comment	Yes	–	–	The comment associated with this address. Double-click in the **Comment** column and type a comment to associate with this address.

Chapter 9
Function Blocks

What Is in This Chapter?

This chapter contains the following sections:

Section 9.1
Fast Counter (%FC)

Using Fast Counter Function Blocks

This section provides descriptions and programming guidelines for using `Fast Counter` function blocks.

What Is in This Section?

This section contains the following topics:

Description

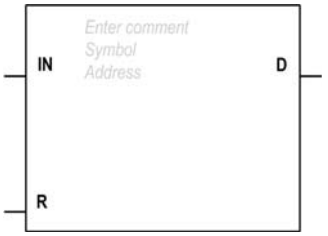
Introduction

The `Fast Counter` function block 1123 serves as either an up-counter or a down-counter. It can count the rising edge of digital inputs up to frequencies of 5 kHz in single word or double word computational mode. Because `Fast Counter` function blocks are managed by specific hardware interrupts, maintaining maximum frequency sampling rates may vary depending on your specific application and hardware configuration.

The `Fast Counter` function blocks `%FC0`, `%FC1`, `%FC2`, and `%FC3` use dedicated inputs `%I0.2`, `%I0.3`, `%I0.4` and `%I0.5` respectively. These bits are not reserved for their exclusive use. Their allocation must be considered with the use of other function blocks for these dedicated resources.

Illustration

This illustration is a `Fast Counter` function block in single-word mode:



Inputs

The `Fast Counter` function block has the following inputs:

Label	Description	Value
IN	Enable	At state 1, the current value is updated according to the pulses applied to the physical input. At state 0, the current value is held at its last value.
R	Reset (optional)	Used to initialize the block. At state 1, the current value is reset to 0 if configured as an up-counter, or set to `%FC.P` or `%FC.PD` if configured as a down-counter. The Done bit `%FC.D` is set back to its default value.

Outputs

The `Fast Counter` function block has the following output:

Label	Description	Value
D	Done (`%FCi.D`)	This bit is set to 1 when: ● `%FCi.V` or `%FCi.VD` reaches the preset value `%FCi.P` or `%FCi.PD` configured as an up-counter. ● or when `%FCi.V` or `%FCi.VD` reaches 0 when configured as a down-counter. This read-only bit is reset only by setting `%FCi.R` to 1.

Configuration

Parameters

To configure parameters, follow the Configuring a Function Block procedure *(see SoMachine Basic, Generic Functions Library Guide)* and read the description of Memory Allocation Modes in the SoMachine Basic Operating Guide *(see SoMachine Basic, Operating Guide)*.

The `Fast Counter` function block has the following parameters:

Parameter	Description	Value
Used	Address used	If selected, this address is currently in use in a program.
Address	`%FCi` `Fast Counter` address	The instance identifier, where it is from 0 to the number of objects available on this logic controller. Refer to Maximum Number of Objects table *(see page 32)* for the maximum number of `Fast Counters`.
Input	`%IO.i`	The dedicated input associated with this function block instance. `%IO.2...%IO.5`
Symbol	Symbol	The symbol associated with this object. Refer to the SoMachine Basic Operating Guide (Defining and Using Symbols) for details.
Configured	Whether to count up or down	Set to one of: • **Not used** • **Up Counter** • **Down Counter**
Preset	Preset value (`%FCi.P` or `%FCi.PD`)	Initial value may be set: • Using associated object `%FCi.P` from 1 to 65535 in single word mode, • Using associated object `%FCi.PD` from 1 to 4294967295 in double word mode.
Double Word	Double word mode	If selected, use double word mode. Otherwise, use single-word mode.
Comment	Comment	An optional comment can be associated with this object. Double-click in the **Comment** column and type a comment.

Objects

The `Fast Counter` function block is associated with the following objects:

Object	Description	Value
`%FCi.V` `%FCi.VD`	Current value	The current value increments or decrements according the up or down counting function selected. For up-counting, the current counting value is updated and can reach 65535 in single word mode (`%FCi.V`) and 4294967295 in double word mode (`%FCi.VD`). For down-counting, the current value is the preset value `%FC.P` or `%FC.PD` and can count down to 0.

Object	Description	Value
%FCi.P %FCi.PD	Preset value	See description in Parameters table above.
%FCi.D	Done	See description in Outputs table above.

Special Note

The application can change the preset value %FCi.P or %FCi.PD and the current value %FCi.V or %FCi.VD at any time. A new value is taken into account only if the R input is active or at the rising edge of the D output %FC.D. This allows for successive different counts without the loss of a single pulse.

Operation

This table describes the main stages of Fast Counter function block operations:

Operation	Action	Result
Count up	A rising edge appears at the Count up input.	The current value %FCi.V is incremented by 1 unit.
	When the preset value %FCi.P or %FCi.PD is reached.	The Done output bit %FCi.D is set to 1.
Count down	A rising edge appears at the down-counting input.	The current value %FCi.V is decremented by 1 unit.
	When the value is 0.	The Done output bit %FCi.D is set to 1.

Special Cases

This table contains a list of special operating cases for the Fast Counter function block:

Special Case	Description
Effect of cold restart (%S0=1)	Resets all the Fast Counter attributes with the values configured or user application *(see page 151)*.
Effect of warm restart (%S1=1)	No effect *(see page 151)*.
Effect of controller stops	The Fast Counter continues to count with the parameter settings enabled at the time the controller was stopped.

Programming Example

Introduction

In this example, the application counts a number of items up to 5000 while %I0.1 is set to 1. The input for %FC0 is the dedicated input %I0.2. When the preset value is reached, %FC0.D is set to 1 and retains the same value until %FC0.R is commanded by the result of AND on %I0.2 and %M0.

Programming

This example is a Fast Counter function block:

Rung	Instruction
0	BLK %FC1 LD %I0.1 IN LD %I0.2 AND %M0 R OUT_BLK LD D ST %Q0.0 END_BLK

NOTE: Refer to the reversibility procedure *(see page 105)* to obtain the equivalent Ladder Diagram.

NOTE: Refer to the reversibility procedure *(see page 105)* to obtain the equivalent Ladder Diagram.

Section 9.2
High Speed Counter (%HSC)

Using High Speed Counter Function Blocks

This section provides descriptions and programming guidelines for using `High Speed Counter` function blocks.

What Is in This Section?

This section contains the following topics:

Description

Introduction

The `High Speed Counter` function block **11123** can be configured by SoMachine Basic to perform any one of the following functions:

- Up/down counter
- Bi-phase counter
- Single counter
- Frequency meter

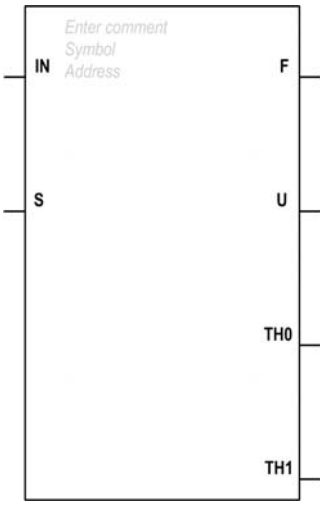
The `High Speed Counter` supports counting of digital inputs up to frequencies of 60 kHz in single word or double word computational mode.

The `High Speed Counter` function block uses dedicated inputs and auxiliary inputs and outputs. Refer to *hardware guide* of your controller for more information on inputs and outputs.

You must configure the `High Speed Counter` function in the **Configuration** tab (**High Speed Counters** →**High Speed Counter Assistant**) before using an instance of the function block, refer to Configuring High Speed Counters *(see page 72)*.

Illustration

This illustration is an `High Speed Counter` function block:



Inputs

The `High Speed Counter` function block has the following inputs:

Label	Description	Value
IN	Enable (required)	0 or 1
S	Preset input. Depending on the configuration, at state 1: ● Up/Down if down function in progress or Up/down bi-phase: initializes the current value with the preset value. ● Up/Down if up function in progress or Single Up Counting: resets the current value to 0. In addition, this also initializes the operation of the threshold outputs and takes into account any user modifications to the threshold values set in the properties window or the program.	0 or 1

Outputs

The `High Speed Counter` function block has the following outputs:

Label	Description	Value
F	Overflow	0 to 65535 or 65535 to 0 in single mode. 0 to 4294967295 or 4294967295 to 0 in double word mode
U	Counting direction Set by the system, this bit is used by the up/down counting function to indicate the direction of counting: ● As a single phase up or down counter, `%I0.1` decides the direction for `%HSC0` and `%I0.7` for `%HSC1`. ● For a bi-phase up/down counter, it is the phase difference between the 2 signals that determines the direction. For `%HSC0`, `%I0.0` is dedicated to IA (Pulse Input Phase A) and `%I0.1` to IB (Pulse Input Phase B). For `%HSC1`, `%I0.6` is dedicated to IA and `%I0.7` to IB.	0 (Down counting) 1 (Up counting)
TH0	Threshold bit 0 exceeded Set to 1 when the current value is greater than or equal to the threshold value S0 (`%HSCi.S0`). It is advisable to test this bit only once in the program because it is updated in real time. The user application is responsible for the validity of the value at its time of use.	0 or 1
TH1	Threshold bit 1 exceeded Set to 1 when the current value is greater than or equal to the threshold value S1 (`%HSCi.S1`). It is advisable to test this bit only once in the program because it is updated in real time.	0 or 1

Configuration

Parameters

To configure parameters, follow the Configuring a Function Block procedure *(see SoMachine Basic, Generic Functions Library Guide)* and read the description of Memory Allocation Modes in the SoMachine Basic Operating Guide *(see SoMachine Basic, Operating Guide)*.

The `High Speed Counter` function block has the following parameters:

Parameter	Description	Value
Used	Address used	If selected, this address is currently in use in a program.
Address	`%HSCi` `High Speed Counter` address	The instance identifier, where i is from 0 to the number of objects available on this logic controller. Refer to Maximum Number of Objects table *(see page 32)* for the maximum number of `High Speed Counter` objects.
Symbol	Symbol	The symbol associated with this object. Refer to the SoMachine Basic Operating Guide (Defining and Using Symbols) for details.
Preset	Preset input value (`%HSCi.P`, `%HSCi.PD`). Only used by the up/down counting function and single up counting.	Initial value may be set: ● Using associated object `%HSC.P` from 0 to 65535 ● Using associated object `%HSC.PD` from 0 to 4294967295
S0	This word contains the value of threshold 0. The meaning is defined during configuration of the function block. This value must be less than S1 (`%HSC.S1`).	Initial value may be set: ● Using associated object `%HSC.S0` from 0 to 65535 ● Using associated object `%HSC.S0D` from 0 to 4294967295
S1	This word contains the value of threshold 1. The meaning is defined during configuration of the function block. This value must be greater than S0 (`%HSC.S0`).	Initial value may be set: ● Using associated object `%HSC.S1` from 0 to 65535 ● Using associated object `%HSC.S1D` from 0 to 4294967295
Time Base	Frequency Measurement Time Base	`%HSC.T`: 100 ms or 1000 ms
Comment	Comment	An optional comment can be associated with this object. Double-click in the **Comment** column and type a comment.

Objects

The High Speed Counter function block is associated with the following objects:

Object	Description	Value
%HSCi.V %HSCi.VD	Current value	See description in Parameters table above.
%HSCi.P %HSCi.PD	Preset value	See description in Parameters table above.
%HSCi.S0 %HSCi.S0D	Threshold 0	See description in Parameters table above.
%HSCi.S1 %HSCi.S1D	Threshold 1	See description in Parameters table above.
%HSCi.C %HSCi.CD	Capture value	See description in Parameters table above.
%HSCi.U	Counting direction	See description in Outputs table above.
%HSCi.F	Overflow	See description in Outputs table above.
%HSCi.T	Time base	See description in Parameters table above.
%HSCi.R	Enable reflex output 0	See description in Parameters table above.
%HSCi.S	Enable reflex output 1	See description in Parameters table above.

Special Cases

This table shows a list of special operating of the High Speed Counter function block:

Special Case	Description
Effect of cold restart (%S0=1)	Resets all the High Speed Counter attributes with the values configured by the user program or user application *(see page 151)*.
Effect of warm restart (%S1=1)	Has no effect *(see page 151)*.
Effect of controller stop	The High Speed Counter stops its function and the outputs stay in their current state. **NOTE:** When the controller stops, the outputs stay in their current state only if the fallback behavior of the tasks is configured to maintain values of the outputs. For more information on configuring fallback behavior, refer to Fallback Behavior *(see SoMachine Basic, Operating Guide)*.

High Speed Counter in Counting Mode

Introduction

The `High Speed Counter` function block works at a maximum frequency of 60 kHz, with a range of 0 to 65535 in single word mode and 0 to 4294967295 in double word mode.

The pulses to be counted are applied in the following way:

Function	Description	%HSC0		%HSC1	
		IA	IB	IA	IB
Up/Down counter	The pulses are applied to the physical input, the current operation (upcount/downcount) is given by the state of the physical input IB.	%I0.0	%I0.1	%I0.6	%I0.7
Bi- phase counter	The 2 phases of the encoder are applied to physical inputs IA and IB.	%I0.0	%I0.1	%I0.6	%I0.7
Single up counter	The pulses are applied to the physical input IA. IB is not used.	%I0.0	-	%I0.6	–
IA Pulse Input Phase A **IB** Pulse Input Phase B					

Notes on Outputs

For all functions in counting mode, the current value is compared to 2 thresholds (`%HSC.S0` or `%HSC.S0D` and `%HSC.S1` or `%HSC.S1D`).

According to the result of these comparison 2-bit objects (`%HSC.TH0` and `%HSC.TH1`) are:

- set to 1 if the current value is greater or equal to the corresponding threshold
- or reset to 0 in the opposite case.

Reflex outputs (if configured) are set to 1 in accordance with these comparisons.

NOTE: None, 1 or 2 outputs can be configured.

`%HSC.U` is an output of the function block; it gives the direction of the associated counter variation (1 for UP, 0 for DOWN).

Special Note

Up-count or down-count operations are made on the rising edge of pulses, and only if the counting function block is enabled.

There are 2 optional inputs used in counting mode: Catch Input and Preset Input:

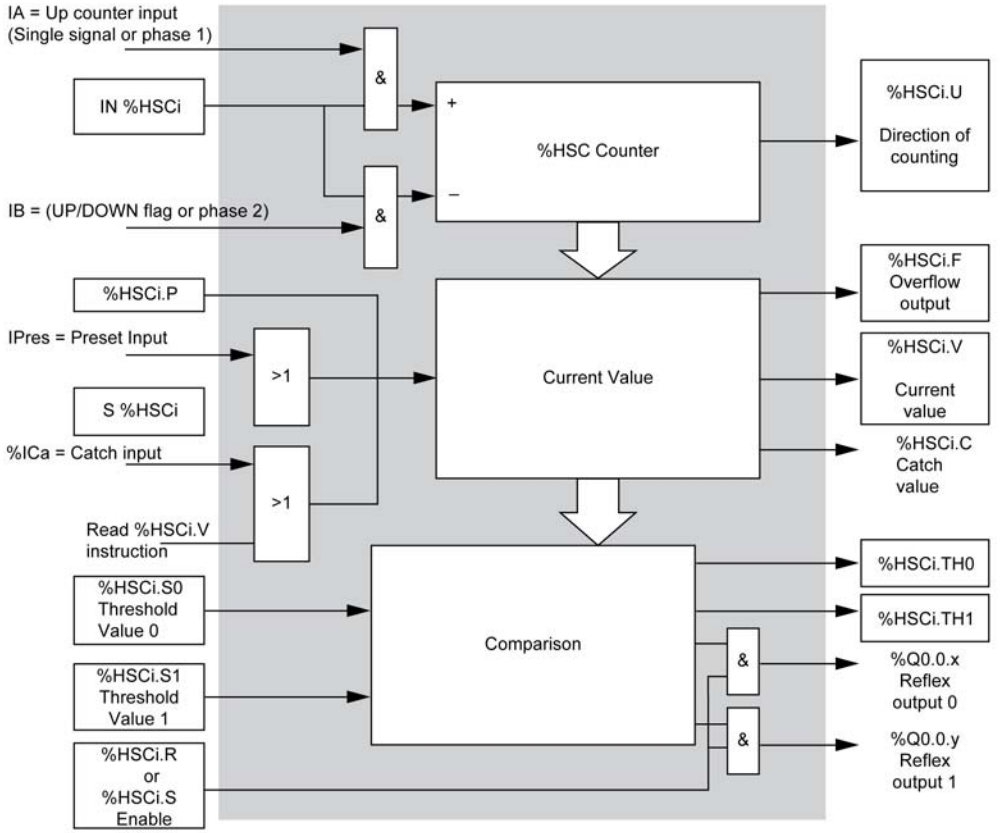
- Catch input is used to capture the current value (`%HSC.V` or `%HSC.VD`) and stored it in `%HSC.C` or `%HSC.CD`. The catch inputs are specified as `%I0.3` for `%HSC0` and `%I0.4` for `%HSC1` if available.

- When Preset input is active, the current value is affected in the following ways:
  - For up-counting, %HSC.V or %HSC.VD is reset to 0
  - For down-counting, %HSC.V or %HSC.VD is written with the content of %HSC.P or %HSC.PD, respectively.
  - For frequency counting, %HSC.V or %HSC.VD is set to 0

NOTE: %HSC.F is also set to 0. The preset inputs are specified as %I0.2 for %HSC0 and %I0.5 for %HSC1 if available.

Operation

This illustration is the operation diagram of the counting mode in single word mode (in double word mode, use the double word function variables, so):



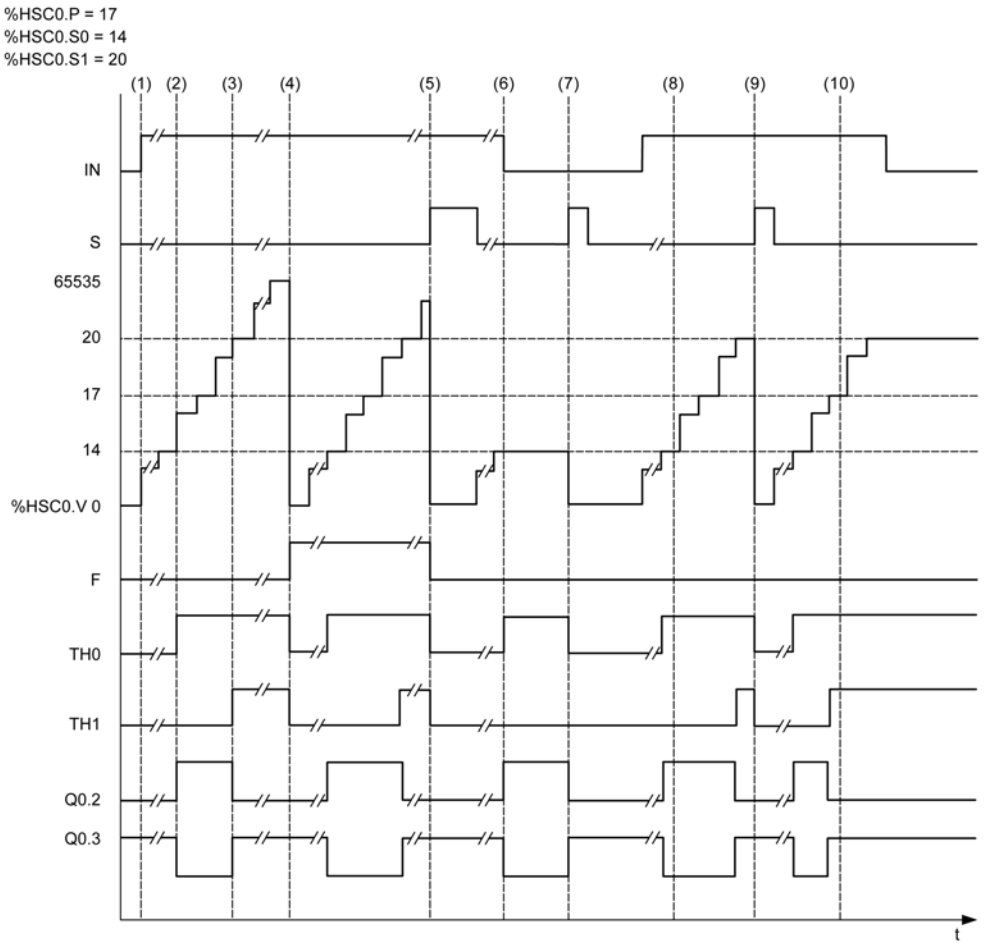
NOTE: Outputs are managed independently from the controller cycle time. The response time is 0...1 ms.

Single Up Counter Timing Diagram

Output value example:

Reflex Output	Value < %HSC.S0	%HSC0.S0 <= Value < %HSC0.S1	Value >= %HSC0.S1
%Q0.2		X	
%Q0.3	X		X

Timing diagram:



(1) `IN` is set to 1: the counting function is activated (`%HSC0.U = 1` because `%HSC0` is an up-counter)
(2) `%Q0.2` (Reflex Output) and `TH0` are set to 1
(3) `TH1` is set to 1

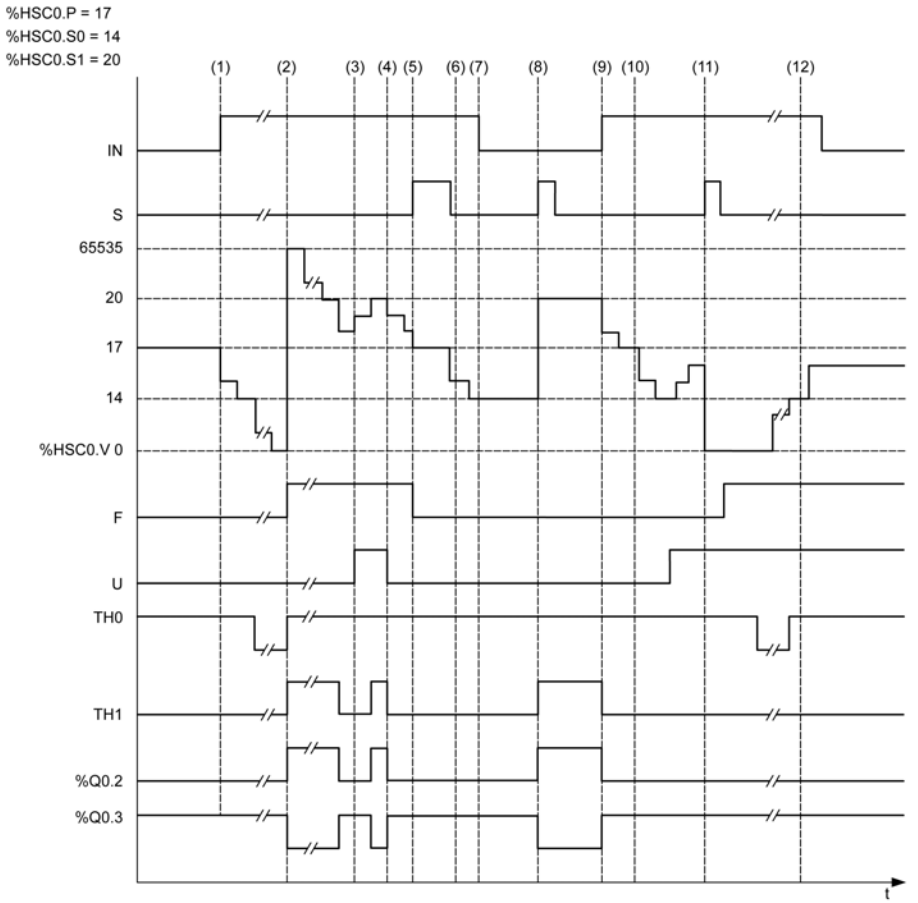
(4) The maximum value is reached so on the next count %HSC0.V is reset to 0 and F is set to 1
(5) S is set to 1, the current value, %HSC0.V, is set to 0
(6) The current function is inhibited while IN is set to 0
(7) While the function is inhibited, S is set to 1 so the current value is reset to 0
(8) Change of threshold value S1 to 17
(9) S is set to 1 so the new value of S1 will be granted at the next count
(10) Catch input is set to 1 so %HSC0.C = 17

Up-Down Counter Timing Diagram

Output value example:

Reflex Output	Value < %HSC.S0	%HSC0.S0 <= Value < %HSC0.S1	Value >= %HSC0.S1
%Q0.2			X
%Q0.3	X	X	

Timing diagram:



(1) Input IN is set to 1 so down-counting mode starts (`%HSC0.U = 0` that is, `IB = 0`)
(2) The current value reaches 0 so `F` output flag is set to 1 and `%HSC0.V` is set to 65535 at the next count
(3) Change at the `IB` input, the counter is now in up counting mode and `%HSC0.U = 1`
(4) `IB` input is set to 0 so the counter is in down counting mode and `%HSC0.U` is set to 0
(5) Input S is set to 1 while down counting is in progress, so `%HSC0.V` is initialized to the Preset value `%HSC0.P = 17`
(6) S is reset to 0 and the preset value `%HSC0.P` is changed to 20
(7) The input `IN` is set to 0 so the function is inhibited, `%HSC0.V` is held
(8) S is set to 1 so the new preset value (`%HSC0.P = 20`) is taken into account and the reflex outputs are updated
(9) `IN` input is set to 1 and the function restarts in down counting mode
(10) The threshold value `%HSC0.S1` is set to 17
(11) S input active makes threshold `S1` new value to be granted at the next count and resets `%HSC0.V` to 0
(12) A catch of the current value `%HSC0.V` is made so `%HSC0.C = 14`

High Speed Counter in Frequency Meter Mode

Introduction

The frequency meter mode of an `High Speed Counter` is used to measure the frequency of a periodic signal in Hz on input IA (pulse input phase A).

The frequency range which can be measured is 1 Hz to 60 kHz.

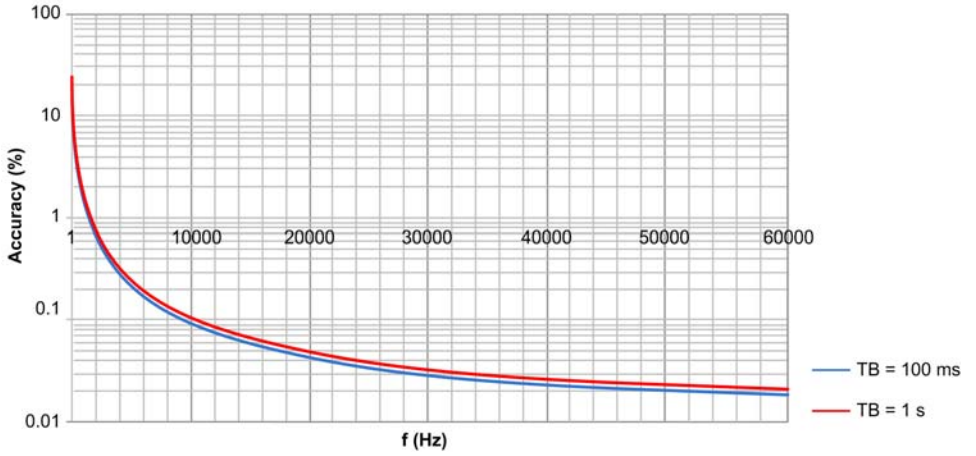
It is possible to choose between 2 time bases, the choice being made by the object `%HSC.T` (Time base):

Time Base	Accuracy	Update
100 ms	0.05% for 60 kHz 10% for 100 Hz	10 times per second
1 s	0.005% for 60 kHz 10% for 10 Hz	Once per second

Accuracy Measurement

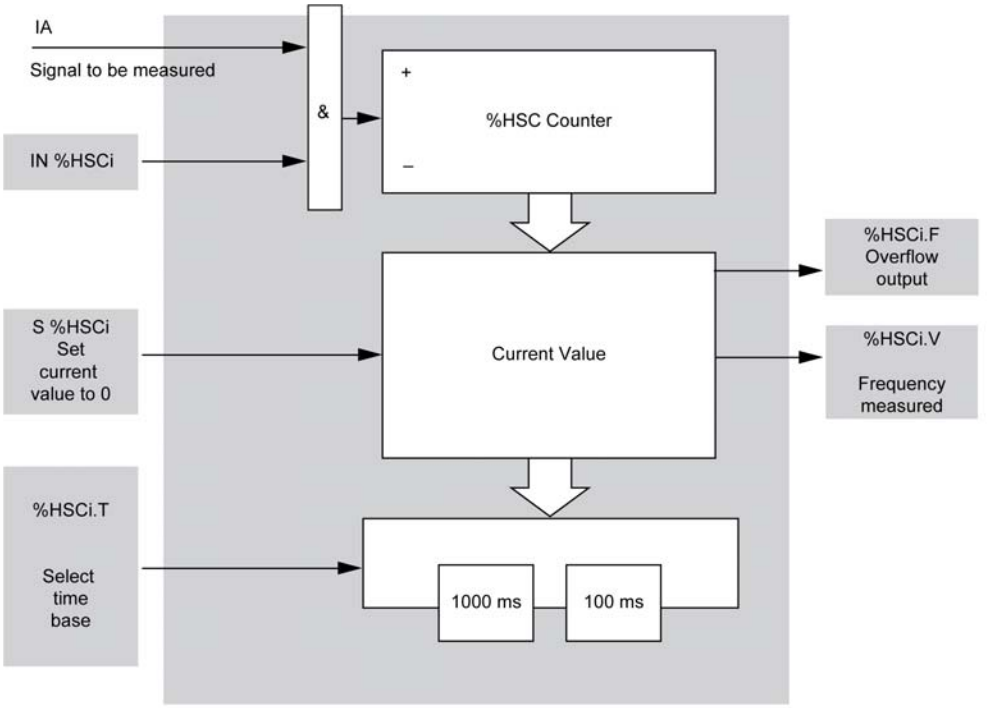
$$Accuracy(\%) = \frac{1}{f[Hz]} \times \frac{1}{TB[s]} \times 100$$

Semi-log curve:



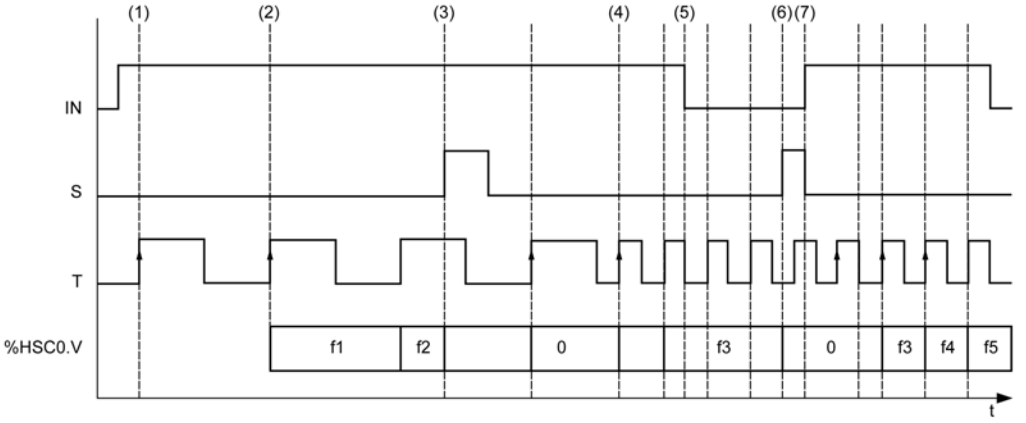
Operation

This illustration is the operation diagram of the frequency meter mode:



Timing Diagram

This timing diagram is an example of using a `High Speed Counter` in frequency meter mode:



(1) The first frequency measurement starts at a rising edge of the `TB` signal
(2) `%HSC0.V` is updated after one period of the `TB`
(3) Input `IN` and input S are set to 1 so `%HSC0.V` is set to 0
(4) `%HSC0.T` is set to 100 ms, so the current measurement is canceled and a new one starts
(5) Input `IN` is set to 0, so the frequency measurement function is inhibited and `%HSC0.V` is held
(6) S is set to 1, so the current value `%HSC0.V` is set to 0
(7) S is set to 0 and `IN` is set to 1, so the measurement will start at the next rising edge of the `TB` signal

Section 9.3
Pulse (%PLS)

Using Pulse Function Blocks

This section provides descriptions and programming guidelines for using `Pulse` function blocks.

What Is in This Section?

This section contains the following topics:

Description

Introduction

The Pulse function block  is used to generate square wave signals.

2 Pulse function blocks are available on the dedicated output channel %Q0.0 or %Q0.1. Logic controllers with relay outputs for these 2 channels do not support the Pulse function block. Refer to *hardware guide* of your controller for more information on inputs and outputs.

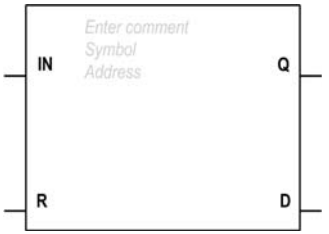
The Pulse function block allows only a single signal width, or duty cycle, of 50%.

You can choose to limit either the number of pulses or the period when the pulse train is executed. These factors can be determined at the time of configuration and/or updated by the program.

You must configure the Pulse function block in the **Configuration →Pulse Generators** before using an instance of the function block, refer to Configuring Pulse Generators *(see page 80)*.

Illustration

This illustration is a Pulse function block:



Inputs

The Pulse function block has the following inputs:

Label	Object	Description	Value
IN	%PLSi.IN	Enable	At state 1, the pulse is produced at the dedicated output channel. At state 0, the output channel is set to 0.
R	%PLSi.R	Reset to 0 (optional)	At state 1, outputs %PLSi.Q and %PLSi.D are set to 0. The number of pulses generated in period T is set to 0.

Outputs

The `Pulse` function block has the following outputs:

Label	Object	Description	Value
Q	`%PLSi.Q`	Generation in progress	At state 1, indicates that the `Pulse` signal is generated at the dedicated output channel configured.
D	`%PLSi.D`	Generation complete (optional)	At state 1, signal generation is complete. The number of desired pulses has been reached.

Configuration

Parameters

To configure parameters, follow the Configuring a Function Block procedure *(see SoMachine Basic, Generic Functions Library Guide)* and read the description of Memory Allocation Modes in the SoMachine Basic Operating Guide *(see SoMachine Basic, Operating Guide)*.

The `Pulse` function block has the following parameters:

Parameter	Description	Value
Used	Address used	If selected, this address is currently in use in a program.
Address	`%PLSi` `Pulse` address	The instance identifier, where i is from 0 to the number of objects available on this logic controller. Refer to Maximum Number of Objects table *(see page 32)* for the maximum number of `Pulse` objects.
Symbol	Symbol	The symbol associated with this object. Refer to the SoMachine Basic Operating Guide (Defining and Using Symbols) *(see SoMachine Basic, Operating Guide)* for details.
Preset	Preselection of the period (`%PLSi.P`)	• Time Base = 1 s, `%PLSi.P`=1 or 2 • Time Base = 10 ms, 1<=`%PLSi.P`<=200 • Time Base = 1 ms, 1<=`%PLSi.P`<=2000 • Time Base = 0.1 ms, 1<=`%PLSi.P`<=20000
Num. Pulse	Number of pulses (`%PLSi.N`, `%PLSi.ND`)	To produce an unlimited number of pulses, set `%PLS.N` or `%PLS.ND` to 0.
Current	Current output (`%PLSi.Q`)	0 or 1.
Done	Done pulse (`%PLSi.D`)	At state 1, signal generation is complete. The number of desired pulses has been reached. It is reset by either setting the IN or the R inputs to 1.
Duty Cycle	`%PLSi.R`	This value gives the percentage of the signal in state 1 in a period. The width Tp is thus equal to: `TP = T x (%PLSi.R:100)`. The user application writes the value for `%PLSi.R`. It is this word which controls the duty cycle of the period. The default value is 0 and values greater than 100 are considered to be equal to 100.
Comment	Comment	An optional comment can be associated with this object. Double-click in the **Comment** column and type a comment.

Objects

The `Pulse` function block is associated with the following objects:

Object	Description	Size (bit)	Defaut Value	Range	
`%PLSi.P`	Preset value	16	Preset (set on **Configuration →Pulse Generators**)	**Preset** `%PLSi.P`	**Time Base**
				1...20000	0.1 ms
				1...2000	1 ms
				1...200	10 ms
				1 or 2	1 s (default)
`%PLSi.N`	Number of pulses	16	0	0...32767	
`%PLSi.ND`		32	0	0...2147483647	

Rules of Use

The output signal period `T` is set with **Preset** and the **Time Base** parameters such as `T = %PLSi.P` x .**Time Base**.

This table shows the range of available periods:

Time Base	Frequency
0.1 ms	0.5 Hz...10000 Hz
1 ms	0.5 Hz...1000 Hz
10 ms	0.5 Hz...100 Hz
1 s	0.5 Hz...1 Hz

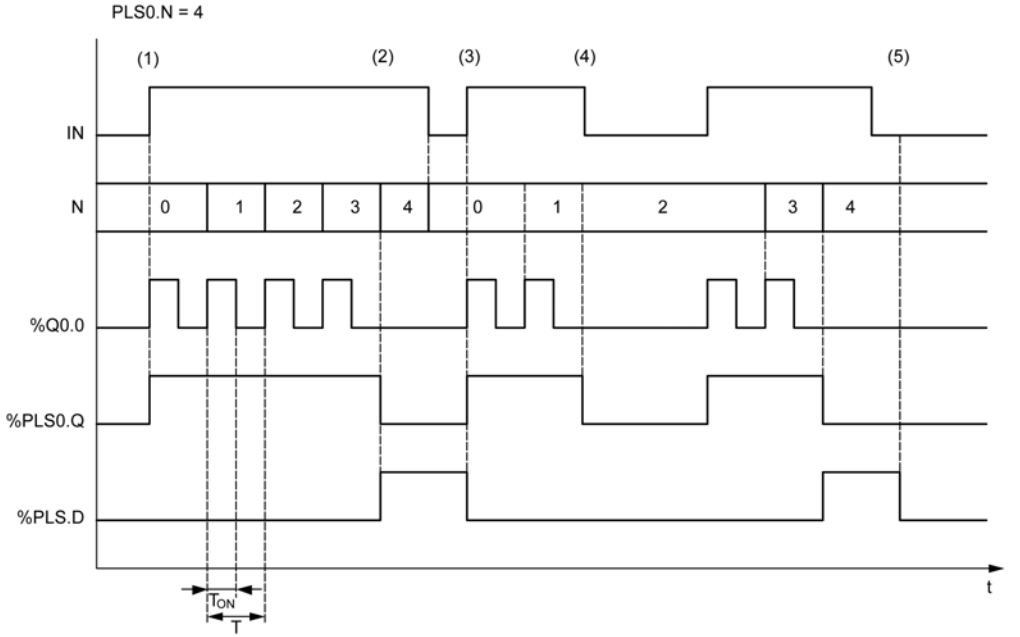
The **Time Base** is set on the **Configuration →Pulse Generators**, refer to Configuring Pulse Generators *(see page 80)*. and cannot be modified.

If `%PLSi.P` is:
- changed, the output signal period is changed at the end of the current period.
- set to 0, the pulse generation function is stopped.
- out of range, the parameter is forced to 0 and the pulse generation function is stopped.

If `%PLSi.N` (or `%PLSi.ND` in **Double Word** mode) is:
- changed, the number of pulse to be generated is used at the next execution of the pulse generation function (`%PLSi.D` = 1 or after `%PLSi.R` = 1).
- set to 0, unlimited number of pulse are generated.
- out of range, the parameter is forced to 0.

Timing Diagram

This diagram displays the timing for `Pulse` function block:



(1) `IN` input is set to 1, the pulse signal is generated at the dedicated output (`%Q0.0`) so `%PLSi.Q` is set to 1
(2) The number of pulses reaches `%PLS0.N` (=4) so the Done flag output (`%PLS0.D`) is set to 1 and the pulse generation is stopped (`%PLS0.Q = 0`)
(3) `IN` input is set to 1 so `%PLS0.D` is reset to 0
(4) `IN` input is set to 0 so the output channel is set to 0 and `%PLS0.Q` = 0 indicates that the signal generation is not active
(5) `%PLS0.D` is set to 0 by setting `R` input to 1

Special Cases

Special Case	Description
Effect of cold restart (`%S0=1`)	pulse generation is stopped output is reset to 0 configuration parameters is restored
Effect of warm restart (`%S1=1`)	none *(see page 151)*.
Effect at controller stop	pulse generation is stopped Output is set to: ● **Maintain values** ● or **Fallback value**
Effect of Online Modification)	none

Programming Example

Introduction

The `Pulse` function block can be configurated as in this programming example.

Programming

This example is a `Pulse` function block:

Rung	Instruction

0	`BLK %PLS0`
	`LD %M1`
	`IN`
	`LD %M0`
	`R`
	`OUT_BLK`
	`LD Q`
	`ST %Q0.5`
	`LD D`
	`ST %M10`
	`END_BLK`

NOTE: Refer to the reversibility procedure *(see page 105)* to obtain the equivalent Ladder Diagram.

NOTE: Refer to the reversibility procedure *(see page 105)* to obtain the equivalent Ladder Diagram.

Section 9.4
Pulse Width Modulation (%PWM)

Using Pulse Width Modulation Function Blocks

This section provides descriptions and programming guidelines for using `Pulse Width Modulation` function blocks.

What Is in This Section?

This section contains the following topics:

Description

Introduction

The `Pulse Width Modulation` function block  generates a square wave signal on dedicated output channel `%Q0.0` or `%Q0.1`, with variable width and, therefore, duty cycle.

Controllers with relay outputs for these 2 channels do not support this function due to a frequency limitation.

Refer to Maximum Number of Objects table *(see page 32)* for information on the number of `Pulse Width Modulation` function blocks available. `%PWM0` uses dedicated output `%Q0.0` and `%PMW1` uses dedicated output `%Q0.1`. The Pulse function blocks `%PLS` contend to use these same dedicated outputs so you must choose between the 2 functions.

You must configure the `Pulse Width Modulation` function block in the **Configuration →Pulse Generators** before using an instance of the function block, refer to Configuring Pulse Generators *(see page 80)*.

NOTE: Only some references of logic controller use the `Pulse Width Modulation` function block.

Illustration

This illustration is the `Pulse Width Modulation` function block:

Inputs

The `Pulse Width Modulation` function block has the following input:

Label	Object	Description	Value
IN	`%PWMi.IN`	Enable	At state 1, the `Pulse Width Modulation` signal is generated at the output channel. At state 0, the output channel is set to 0.

Configuration

Parameters

To configure parameters, follow the Configuring a Function Block procedure *(see SoMachine Basic, Generic Functions Library Guide)* and read the description of Memory Allocation Modes in the SoMachine Basic Operating Guide *(see SoMachine Basic, Operating Guide)*.

The `Pulse Width Modulation` function block has the following parameters:

Parameter	Description	Value
Used	Address used	If selected, this address is currently in use in a program.
Address	`%PWMi` `Pulse Width Modulation` address	The instance identifier, where i is from 0 to the number of objects available on this logic controller. Refer to Maximum Number of Objects table *(see page 32)* for the maximum number of `Pulse Width Modulation` objects.
Symbol	Symbol	The symbol associated with this object. Refer to the SoMachine Basic Operating Guide (Defining and Using Symbols) *(see SoMachine Basic, Operating Guide)* for details.
Preset	Preselection of the period	• Time Base = 1 s, `%PWMi.P=1` • Time Base = 10 ms, `1<=%PWMi.P<=100` • Time Base = 1 ms, `1<=%PWMi.P<=1000` • Time Base = 0.1 ms, `1<=%PWMi.P<=10000`
Num. Pulse	Number of pulses (`%PWMi.N`, `%PWMi.ND`)	To produce an unlimited number of pulses, set `%PWMi.N` or `%PWMi.ND` to 0.
Current	Current output (`%PWMi.Q`)	0 or 1.
Done	Done pulse (`%PWMi.D`)	At state 1, signal generation is complete. The number of desired pulses has been reached. It is reset by either setting the IN or the R inputs to 1.
Duty cycle	`%PWMi.R`	This value gives the percentage of the signal in state 1 in a period. The width Tp is thus equal to: `TP = T x (%PWMi.R:100)`. The user application writes the value for `%PWMi.R`. It is this word which controls the duty cycle of the period. The default value is 0 and values greater than 100 are considered to be equal to 100.
Comment	Comment	An optional comment can be associated with this object. Double-click in the **Comment** column and type a comment.

Objects

The `Pulse Width Modulation` function block is associated with the following objects:

Object	Description	Size (bit)	Defaut Value	Range	
`%PWMi.P`	Preset value	16	Preset (set on **Configuration →Pulse Generators**)	**Preset** `%PWMi.P`	**Time Base**
				1...10000	0.1 ms
				1...1000	1 ms
				1...100	10 ms
				1	1 s (default)
`%PWMi.R`	Ratio	16	0	0...100	

Rules of Use

The output signal period `T` is set with **Preset** and the **Time Base** parameters such as `T = %PWMi.P` x **Time Base**.

This table shows the range of available periods:

Time Base	Frequency
0.1 ms	1 Hz...10000 Hz
1 ms	1 Hz...1000 Hz
10 ms	1 Hz...100 Hz
1 s	1 Hz...1 Hz

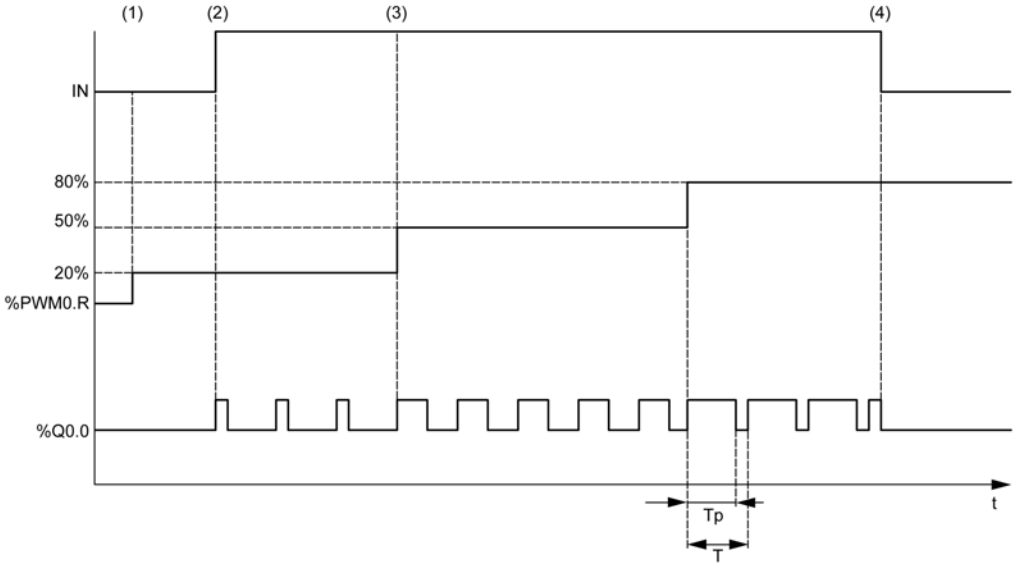
The **Time Base** is set on the **Configuration →Pulse Generators**, refer to Configuring Pulse Generators *(see page 80)*. and cannot be modified.

If `%PWMi.P` is:
- changed, the output signal period is changed at the end of the current period.
- set to 0, the pulse generation function is stopped.
- out of range, the parameter is forced to 0 and the pulse generation function is stopped.

If `%PWMi.R` is:
- set to 0, the pulse generation function is stopped (output set to 0).
- set to 100, the output signal is set to 1
- changed, the output signal ratio is changed at the end of the current period.
- out of range, the parameter is forced to 0.

Timing Diagram

This diagram displays the timing for the `Pulse Width Modulation` function block:



(1) The PWM ratio (`%PWMi.R`) is set to 20%, `IN = 0` so the pulse generation is not active
(2) `IN` is set to 1 so PWM output is activated
(3) The programmable width (`Tp`) changes with `%PWM.R`
(4) `IN` is set to 0 so the `PWM` function is inhibited

Special Cases

Special Case	Description
Effect of cold restart (`%S0=1`)	pulse generation is stopped output is reset to 0 configuration parameters is restored
Effect of warm restart (`%S1=1`)	none *(see page 151)*.
Effect at controller stop	pulse generation is stopped Output is set to: • **Maintain values** • or **Fallback value**
Effect of Online Modification)	none

Programming Example

Introduction

The `Pulse Width Modulation` function block can be configured as in this programming example.

Programming Example

In this example:

- The signal width is modified by the program according to the state of controller input `%I0.0` and `%I0.1`.
- The time base is set to 10 ms.
- The preset value `%PWM0.P` is set to 50 so the ratio step is equal to 2%.
- The configurable period T is equal to 500 ms.

The result is:

- If `%I0.0` and `%I0.1` are set to 0, the `%PWM0.R` ratio is set at 20%, the duration of the signal at state 1 is then: 20% x 500 ms = 100 ms.
- If `%I0.0` is set to 0 and `%I0.1` is set to 0, the `%PWM0.R` ratio is set at 50% (duration 250 ms).
- If `%I0.0` and `%I0.1` are set to 1, the `%PWM0.R` ratio is set at 80% (duration 400 ms).

Examples of `Pulse Width Modulation` instructions:

Rung	Instruction

0	`LDN  %I0.0`
	`ANDN  %I0.1`
	`[%PWM0.R:=20]`

1	`LD  %I0.0`
	`ANDN  %I0.1`
	`[%PWM0.R:=50]`

2	`LD  %I0.0`
	`AND  %I0.1`
	`[%PWM0.R:=80]`

3	`BLK  %PWM0`
	`LD  %I0.2`
	`IN`
	`END_BLK`

NOTE: Refer to the reversibility procedure *(see page 105)* to obtain the equivalent Ladder Diagram.

NOTE: Refer to the reversibility procedure *(see page 105)* to obtain the equivalent Ladder Diagram.

Chapter 10
System Objects

What Is in This Chapter?

This chapter contains the following topics:

System Bits (%S)

Introduction

This section provides information about the function of system bits.

Displaying System Bits Properties

Follow these steps to display properties of the system bits:

Step	Action
1	Select the **Tools** tab in the left-hand area of the **Programming** window.
2	Click **System objects** →**System Bits**. **Result**: System bit properties appear on the screen.

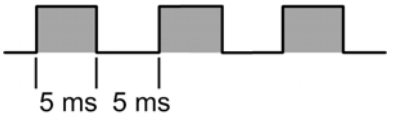
System Bits Properties

This table describes each property of the system bit:

Parameter	Editable	Value	Default Value	Description
Used	No	True/False	False	Indicates whether the system bit is being referenced in a program.
Address	No	%Si	–	Displays the system bit address, where i is the bit number that represents the sequential position of the system bit in the memory. If the controller has maximum n system bits, the value of i is given as 0...n-1. For example, %S4 is system bit 4.
Symbol	Yes	–	–	The symbol associated with the system bit. Double-click in the **Symbol** column and type the name of the symbol to associate with the system bit. If a symbol already exists, you can right-click in the **Symbol** column and choose **Search and Replace** to find and replace occurrences of the symbol throughout the program and/or program comments.
Comment	Yes	–	–	A comment associated with the system bit. Double-click in the **Comment** column and type an optional comment to associate with the system bit.

System Bits Description

This table presents the description of the system bits and how they are controlled:

System Bit	Function	Description	Init State	Control
%S0	Cold Start	Normally set to 0, it is set to 1 by: ● A power return with loss of data (battery malfunction), ● The user program or an animation table. This bit is set to 1 during the first complete scan. It is reset to 0 by the system before the next scan.	0	S or U–~~S~~, SIM
%S1	Warm Start Only Read operation is available	Normally set to 0. It is set to 1 by a power return with data backup. It is reset to 0 by the system at the end of the complete scan.	0	S
%S4 %S5 %S6 %S7	Time base: 10 ms Time base: 100 ms Time base: 1 s Time base: 1 min	The rate of status changes is measured by an internal clock. They are not synchronized with the controller scan. Example: %S4  	–	S, SIM (except %S4)
%S9	Reset outputs	Normally set to 0. It can be set to 1 by the program or by the terminal (in the Animation Table Editor): ● At state 1, outputs are forced to 0 when the controller is in RUN mode, ● At state 0, outputs are updated normally.	0	U
%S10	I/O communication status	Normally set to 1 (TRUE on control panel). This bit can be set to 0 (FALSE on control panel) by the system when an I/O communication interruption is detected.	1	S
%S11	Watchdog overflow	Normally set to 0. This bit can be set to 1 by the system when the program execution time (scan time) exceeds the maximum scan time (software watchdog). Watchdog overflow causes the controller state to change to HALT.	0	S

S Controlled by the system
U Controlled by the user
U–~~S~~ Set to 1 by the user, reset to 0 by the system
S–~~U~~ Set to 1 by the system, reset to 0 by the user
SIM Applied in the Simulator

System Bit	Function	Description	Init State	Control
%S12	PLC in RUN mode	This bit reflects the running state of the controller. The system sets the bit to: ● 1 when the controller is running, ● 0 for stop, init, or any other state.	0	S, SIM
%S13	First cycle in RUN	Normally at 0, this bit is set to 1 by the system during the first scan after the controller state has been changed to RUN.	1	S, SIM
%S17	Last ejected bit	Normally set to 0. It is set by the system according to the value of the last ejected bit. It indicates the value of the last ejected bit.	0	S→U, SIM
%S18	Arithmetic overflow or error	Normally set to 0. It is set to 1 in the case of an overflow when a 16-bits operation is performed, that is: ● A result greater than + 32767 or less than - 32768, in single length, ● A result greater than + 2147483647 or less than - 2147483648, in double length, ● A result greater than + 3.402824E+38 or less than - 3.402824E+38, in floating point, ● Division by 0, ● The square root of a negative number, ● BTI or ITB conversion not significant: BCD value out of limits. It must be tested by the user program after each operation where there is a risk of an overflow; then reset to 0 by the user program if an overflow occurs.	0	S→U, SIM
%S19	Scan period overrun (periodic scan)	Normally at 0, this bit is set to 1 by the system in the event of a scan period overrun (scan time greater than the period defined by the user program at configuration or programmed in %SW0). This bit is reset to 0 by the user program.	0	S→U
%S20	Index overflow	Normally at 0, it is set to 1 when the address of the indexed object becomes less than 0 or more than the maximum size of an object. It must be tested by the user program, after each operation where there is a risk of overflow; then reset to 0 if an overflow occurs.	0	S→U, SIM

S Controlled by the system
U Controlled by the user
U→S Set to 1 by the user, reset to 0 by the system
S→U Set to 1 by the system, reset to 0 by the user
SIM Applied in the Simulator

System Bit	Function	Description	Init State	Control
%S21	GRAFCET initialization	Normally set to 0, it is set to 1 by: ● A cold restart, %S0 = 1, ● The user program, in the preprocessing program part only, using a Set Instruction (S %S21) or a set coil –(S)– %S21, ● The terminal. At state 1, it causes GRAFCET initialization. Active steps are deactivated and initial steps are activated. It is reset to 0 by the system after GRAFCET initialization.	0	U→S, SIM
%S22	GRAFCET reset	Normally set to 0, it can only be set to 1 by the program in pre-processing. At state 1, it causes the active steps of the entire GRAFCET to be deactivated. It is reset to 0 by the system at the start of the execution of the sequential processing.	0	U→S, SIM
%S23	Preset and freeze GRAFCET	Normally set to 0, it can only be set to 1 by the program in the pre-processing program module. Set to 1, it validates the pre-positioning of GRAFCET. Maintaining this bit at 1 freezes the GRAFCET (freezes the chart). It is reset to 0 by the system at the start of the execution of the sequential processing to ensure that the GRAFCET chart moves on from the frozen situation.	0	U→S, SIM

S Controlled by the system
U Controlled by the user
U→S Set to 1 by the user, reset to 0 by the system
S→U Set to 1 by the system, reset to 0 by the user
SIM Applied in the Simulator

System Bit	Function	Description	Init State	Control
%S33	Read or Write selection for Ethernet server configuration read/change	Normally at 0. ● Set to 0, the %SW33 to %SW38 contains the Ethernet parameters in use (IP declared or IP assigned by BOOTP or automatic IP self assigned). These parameters are those configured in the application or those of the post configuration in SD card (in this case, %SW98 or %SW99 or %SW100 is different from 0). ● Set to 1 (if there is no post configuration in use), then the new configuration is given by %SW33 to %SW38. This bit can be set to its initial state 0 by the user program and the system (on cold restart). Then, the Ethernet is reset to apply the application configuration whatever the current configuration is. This bit cannot be set to 1 if a post configuration is in use.	0	U→S
%S34	Ethernet Autonegotiation	Set to 0 to allow the autonegotiation of the speed and half or full duplex mode. Set to 1 to force some specific configuration set in %S35 and %S36. **NOTE:** A change in the state of %S34, %S35, or %S36 will provoke a reinitialization of the Ethernet channel, so, after the change the Ethernet channel will be unavailable for few minutes.	0	U
%S35	Ethernet half/full duplex mode	In case of the %S34 = 0 (autonegotiation) this bit will be set by the system, and it will be read only for the user. But is the %S34 = 1, the mode will be forced based on the value of this bit set by the user: ● Set to 0 if Half Duplex, ● Set to 1 if Full Duplex. **NOTE:** A change in the state of %S34, %S35, or %S36 will provoke a reinitialization of the Ethernet channel, so, after the change the Ethernet channel will be unavailable for few minutes.	–	U or S

S Controlled by the system
U Controlled by the user
U–S Set to 1 by the user, reset to 0 by the system
S–U Set to 1 by the system, reset to 0 by the user
SIM Applied in the Simulator

System Bit	Function	Description	Init State	Control
%S36	Ethernet speed	In case of the %S34 = 0 (autonegotiation) this bit will be set by the system, and it will be read only for the user. But is the %S34 = 1, the mode will be forced based on the value of this bit set by the user: ● Set to 0 if 10 Mbps, ● Set to 1 if 100 Mbps. **NOTE:** A change in the state of %S34, %S35, or %S36 will provoke a reinitialization of the Ethernet channel, so, after the change the Ethernet channel will be unavailable for few minutes.	–	U or S
%S38	Permission for events to be placed in the events queue	Normally at 1. ● Set to 0, events cannot be placed in the events queue. ● Set to 1, events are placed in the events queue as soon as they are detected, This bit can be set to its initial state 1 by the user program and the system (on cold restart).	1	U→S
%S39	Saturation of the events queue	Normally at 0. ● Set to 0, all events are reported. ● Set to 1, at least one event is lost. This bit can be set to 0 by the user program and the system (on cold restart).	0	U→S
%S50	Updating the date and time using words %SW49 to %SW53	Normally on 0, this bit can be set to 1 or 0 by the program. ● Set to 0, the date and time can be read. ● Set to 1, the date and time can be updated. The internal RTC controller is updated on a falling edge of %S50.	0	U→S

S Controlled by the system
U Controlled by the user
U–S Set to 1 by the user, reset to 0 by the system
S–U Set to 1 by the system, reset to 0 by the user
SIM Applied in the Simulator

System Bit	Function	Description	Init State	Control
%S51	Time-of-day clock status	Normally on 0, this bit can be set to 1 or 0 by the program. ● Set to 0, the date and time are consistent. ● Set to 1, the date and time must be initialized by the user program. When this bit is set to 1, the time of day clock data is not valid. The date and time may never have been configured, the battery may be low, or the controller correction constant may be invalid (never configured, difference between the corrected clock value and the saved value, or value out of range). State 1 transitioning to state 0 forces a Write of the correction constant to the RTC.	0	U→S, SIM
%S52	RTC = detected error	This bit managed by the system indicates that the RTC correction has not been entered, and the date and time are false. ● Set to 0, the date and time are consistent, ● At state 1, the date and time must be initialized.	0	S, SIM
%S59	Updating the date and time using word %SW59	Normally on 0, this bit can be set to 1 or 0 by the program. ● Set to 0, the system word %SW59 is not managed, ● Set to 1, the date and time are incremented or decremented according to the rising edges on the control bits set in %SW59.	0	U
%S75	Battery status	This system bit is set by the system and can be read by the user. It indicates the battery status: ● Set to 0, the external battery is operating normally. ● Set to 1, external battery power is low, or no external battery is detected.	0	S
%S92	%MW variables saved on flash	Set to 1 if there is valid data saved on flash. Set to 0 if the data block is invalid or the write operation is in progress.	–	S
%S93	Backup %MW in flash	User set this bit to 1 to store the %MW variable in the flash (up to 1000).	–	U
%S94	Restore %MW	User set this bit to 1 to restore the saved data.	–	U

S Controlled by the system
U Controlled by the user
U→S Set to 1 by the user, reset to 0 by the system
S→U Set to 1 by the system, reset to 0 by the user
SIM Applied in the Simulator

System Bit	Function	Description	Init State	Control
%S96	Backup program OK	This bit can be read at any time (either by the program or while adjusting), in particular after a cold start or a warm restart. ● Set to 0, the backup program is invalid. ● Set to 1, the backup program is valid.	0	S, SIM
%S101	Changing a port address (Modbus protocol)	Used to change a port address using system words %SW101 (port 1) and %SW102 (port 2). To do this, %S101 must be set to 1. ● Set to 0, the address cannot be changed. The value of %SW101 and %SW102 matches the current port address, ● Set to 1, the address can be changed by changing the values of %SW101 (port 1) and %SW102 (port 2). Having modified the values of the system words, %S101 must be set back to 0.	0	U
%S103 %S104	Using the ASCII protocol	Enables the use of the ASCII protocol on Comm 1 (%S103) or Comm 2 (%S104). The ASCII protocol is configured using system words %SW103 and %SW105 for Comm 1, and %SW104 and %SW106 for Comm 2. ● Set to 0, the protocol used is the one configured in SoMachine Basic, ● Set to 1, the ASCII protocol is used on Comm 1 (%S103) or Comm 2 (%S104). In this case, the system words %SW103, %SW105, and %sw121 must be previously configured for COM 1, and %SW104, %SW106, and %SW122 for COM 2. Each change of those %SW will be tacking account after a rising edge to %S103 or %S104.	0	U
%S119	Local I/O detected error	Normally set to 1. This bit can be set to 0 when an I/O communication interruption is detected on the base controller. %SW118 determines the nature of the communication interruption. Resets to 1 when the communication interruption disappears.	1	S

S Controlled by the system
U Controlled by the user
U–S Set to 1 by the user, reset to 0 by the system
S–U Set to 1 by the system, reset to 0 by the user
SIM Applied in the Simulator

System Words (%SW)

Introduction

This section provides information about the function of system words.

Displaying System Words Properties

Follow these steps to display properties of the system words:

Step	Action
1	Select the **Tools** tab in the left-hand area of the **Programming** window.
2	Click **System objects** →**System Words**. **Result**: System word properties appear on the screen.

System Bits Properties

This table describes each property of the system word:

Parameter	Editable	Value	Default Value	Description
Used	No	True/False	False	Indicates whether the system word is being referenced in a program.
Address	No	%Si	–	Displays the system word address, where i is the word number that represents the sequential position of the system word in the memory. If the controller has maximum n system words, the value of i is given as 0...n-1. For example, `%SW50` is system bit 50.
Symbol	Yes	–	–	The symbol associated with the system word. Double-click in the **Symbol** column and type the name of the symbol to associate with the system word. If a symbol already exists, you can right-click in the **Symbol** column and choose **Search and Replace** to find and replace occurrences of the symbol throughout the program and/or program comments.
Comment	Yes	–	–	A comment associated with the system word. Double-click in the **Comment** column and type an optional comment to associate with the system word.

System Words Description

This table presents the description of the system words and how they are controlled:

System Words	Function	Description	Control
%SW0	Controller scan period (master task set to periodic scan mode)	Modifies controller scan period defined at configuration through the user program in an animation table.	U, SIM
%SW1	Periodic task period	Modifies the cycle time [5...255 ms] of the periodic task, without losing the **Period** value specified in the periodic task properties window. Allows you to recover the **Period** value saved in the periodic task properties window: ● in case of a cold start, or ● if the value you write in %SW1 is outside [5...255] range. The %SW1 value can be modified in the program at each end of a cycle, in the program or in an animation table without having to stop the program. Cycle times can be correctly observed while the program is running.	U, SIM
%SW6	Controller status %MW60012	Controller status: 0 = NO CONFIG 2 = STOP 3 = RUN 4 = HALT 5 = POWERLESS	S, SIM

(1) If a single expansion module is missing at power-on, then all expansion module bits are set to 1 (detected error).
S Controlled by the system
U Controlled by the user
SIM Applied in the simulator

System Words	Function	Description	Control

%SW7	Controller state	• Bit [0]: Backup/restore in progress:
		  • Set to 1 if backup/restore of the user program is in progress,
		  • Set to 0 if backup/restore of the user program is complete or disabled.
		• Bit [1]: Configuration of the controller is OK:
		  • Set to 1 if configuration ok.
		• Bit [2]: SD card status bits:
		  • Set to 1 if SD card is present.
		• Bit [3]: SD card status bits:
		  • Set to 1 if SD card is being accessed.
		• Bit [4]: Application in RAM different than EEPROM:
		  • Set to 1 if RAM application different to EEPROM.
		• Bit [6]: not used (status 0)
		• Bit [7]: Controller reserved:
		  • Set to 1 when the controller is in connected mode with SoMachine Basic.
		• Bit [8]: Application in Write mode:
		  • Set to 1 if application is protected.
		• Bit [9]: not used (status 0)
		• Bit [10]: Second serial port installed as cartridge (compact only):
		  • 0 = no serial cartridge
		  • 1 = serial cartridge installed
		• Bit [11]: Second serial port type:
		  • Set to 1 = EIA RS-485
		• Bit [12]: application valid in internal memory:
		  • Set to 1 if application valid.
		• Bit [14]: Valid application in RAM:
		  • Set to 1 if application valid.
		• Bit [15]: ready for execution:
		  • Set to 1 if ready for execution.
			S, SIM

%SW11	Software watchdog value	Contains the maximum value of the watchdog. The value (10...500 ms) is defined by the configuration.	U, SIM
%SW13	BOOT version Vxx.yy	For example, if %SW13=0010: • 8 MSB=00 in hexadecimal, then xx=0 in decimal • 8 LSB=10 in hexadecimal, then yy=16 in decimal As a result, BOOT version is V0.16.	U, SIM
(1) If a single expansion module is missing at power-on, then all expansion module bits are set to 1 (detected error). **S** Controlled by the system **U** Controlled by the user **SIM** Applied in the simulator			

System Words	Function	Description	Control
%SW14	Commercial version, Vxx.yy	For example, if %SW14=0232: ● 8 MSB=02 in hexadecimal, then xx=2 in decimal ● 8 LSB=32 in hexadecimal, then yy=50 in decimal As a result, commercial version is V2.50. **NOTE:** Firmware version must be 2.5 or higher.	S, SIM
%SW15	Firmware version, Vaa.bb.cc.dd	● 8 MSB is aa in hexadecimal ● 8 LSB is bb in hexadecimal	S, SIM
%SW16		● 8 MSB is cc in hexadecimal ● 8 LSB is dd in hexadecimal	S, SIM
%SW17	Default status for floating operation	When an error is detected in a floating arithmetic operation, bit %S18 is set to 1 and the default status of %SW17 is updated according to the following coding: ● Bit[0]: Invalid operation, result is not a number (1.#NAN or -1.#NAN), ● Bit[1]: Reserved, ● Bit[2]: Divided by 0, result is infinite (-1.#INF or 1.#INF), ● Bit[3]: Result greater in absolute value than +3.402824e+38, result is infinite (-1.#INF or 1.#INF).	S and U, SIM
%SW18-%SW19	100 ms absolute timer counter	The counter works using 2 words: ● %SW18 represents the least significant word, ● %SW19 represents the most significant word.	S and U, SIM
%SW30	Last scan time (master task)	Shows execution time of the last controller scan cycle (in ms). **NOTE:** This time corresponds to the time elapsed between the start (acquisition of inputs) and the end (update of outputs) of a master task scan cycle. If the scan time is 2,250 ms, the %SW30 will be 2 and the %SW70 will be 250.	S

(1) If a single expansion module is missing at power-on, then all expansion module bits are set to 1 (detected error).
S Controlled by the system
U Controlled by the user
SIM Applied in the simulator

System Words	Function	Description	Control
%SW31	Max. scan time (master task)	Shows execution time of the longest controller scan cycle since the last cold start (in ms). **NOTE:** ● This time corresponds to the time elapsed between the start (acquisition of inputs) and the end (update of outputs) of a scan cycle. If the maximum scan time is 2,250 ms, the %SW31 will be 2 and the %SW71 will be 250. ● To ensure proper detection of a pulse signal when the latching input option is selected, the pulse width (T_{ON}) and the period (P) must meet the following 2 requirements: ● $T_{ON} \geq 1$ ms ● The input signal period (P) must follow the Nyquist-Shannon sampling rule stating that the input signal period (P) must be at least twice the maximum program scan time (%SW31): $P \geq 2 \times$ %SW31. **Note:** If this condition is not fulfilled, some pulses may be missed.	S
%SW32	Min. scan time (master task)	Shows execution time of shortest controller scan cycle since the last cold start (in ms). **NOTE:** This time corresponds to the time elapsed between the start (acquisition of inputs) and the end (update of outputs) of a scan cycle. If the minimum scan time is 2,250 ms, the %SW32 will be 2 and the %SW72 will be 250.	S

(1) If a single expansion module is missing at power-on, then all expansion module bits are set to 1 (detected error).
S Controlled by the system
U Controlled by the user
SIM Applied in the simulator

System Words	Function	Description	Control
	IP address for Ethernet server configuration read/write	The current Ethernet/IP settings can be changed. The Read or Write selection is done using the system bit %S33.	U
%SW33		IP address: %SW33 and %SW34 For IP address AA.BB.CC.DD: %SW33 = CC.DD and %SW34 = AA.BB	
%SW34		IP address: %SW33 and %SW34 For IP address AA.BB.CC.DD: %SW33 = CC.DD and %SW34 = AA.BB	
%SW35		Subnetwork mask: %SW35 and %SW36 For subnetwork mask AA.BB.CC.DD: %SW35 = CC.DD and %SW36 = AA.BB	
%SW36		Subnetwork mask: %SW35 and %SW36 For subnetwork mask AA.BB.CC.DD: %SW35 = CC.DD and %SW36 = AA.BB	
%SW37		Gateway address: %SW37 and %SW38 For Gateway address AA.BB.CC.DD: %SW37 = CC.DD and %SW38 = AA.BB	
%SW38		Gateway address: %SW37 and %SW38 For Gateway address A.BB.CC.DD: %SW37 = CC.DD and %SW38 = AA.BB	
%SW39	Periodic average time	Shows the average execution time of the periodic task in µs (last 5 times)	–
%SW40	Event 0 average time	Shows the average execution time of the event 0 in µs (last 5 times)	–
%SW41	Event 1 average time	Shows the average execution time of the event 1 in µs (last 5 times)	–
%SW42	Event 2 average time	Shows the average execution time of the event 2 in µs (last 5 times)	–
%SW43	Event 3 average time	Shows the average execution time of the event 3 in µs (last 5 times)	–
%SW44	Event 4 average time	Shows the average execution time of the event 4 in µs (last 5 times)	–
%SW45	Event 5 average time	Shows the average execution time of the event 5 in µs (last 5 times)	–
%SW46	Event 6 average time	Shows the average execution time of the event 6 in µs (last 5 times)	–

(1) If a single expansion module is missing at power-on, then all expansion module bits are set to 1 (detected error).
S Controlled by the system
U Controlled by the user
SIM Applied in the simulator

System Words	Function	Description		Control
%SW47	Event 7 average time	Shows the average execution time of the event 7 in µs (last 5 times)		–
%SW48	Number of events	Shows how many events have been executed since the last cold start. (Counts all events except cyclic events.) **NOTE:** Set to 0 (after application loading and cold start), increments on each event execution.		S, SIM
%SW49 %SW50 %SW51 %SW52 %SW53	Real-Time Clock (RTC)	RTC functions: words containing current date and time values (in BCD):		S and U, SIM
		%SW49	xN Day of the week (N=1 for Monday)	
		%SW50	00SS Seconds	
		%SW51	HHMM: hour and minute	
		%SW52	MMDD: month and day	
		%SW53	CCYY: century and year	
		These words are controlled by the system when bit %S50 is at 0. These words can be written by the user program or by the terminal when bit %S50 is set to 1. On a falling edge of %S50 the internal RTC controller is updated from the values written in these words.		
%SW54 %SW55 %SW56 %SW57	Date and time of the last stop	System words containing the date and time of the last power outage or controller stop (in BCD):		S, SIM
		%SW54	SS Seconds	
		%SW55	HHMM: hour and minute	
		%SW56	MMDD: month and day	
		%SW57	CCYY: century and year	

(1) If a single expansion module is missing at power-on, then all expansion module bits are set to 1 (detected error).
S Controlled by the system
U Controlled by the user
SIM Applied in the simulator

System Words	Function	Description			Control
%SW58	Code of last stop	Displays code giving cause of last stop:			S, SIM
		1 =		Run/Stop input edge	
		2 =		Stop at detected software error (controller scan overshoot)	
		3 =		Stop command	
		4 =		Power outage	
		5 =		Stop at detected hardware error	
		6 =		Init in cold start	
		7 =		Start in stop	
		8 =		Low battery	
		9 =		Controller is not OK to run	
%SW59	Adjust current date	Adjusts the current date. Contains 2 sets of 8 bits to adjust current date. The operation is always performed on rising edge of the bit. This word is enabled by bit %S59.			U
		Increment	**Decrement**	**Parameter**	
		bit 0	bit 8	Day of week	Not used
		bit 1	bit 9	Seconds	
		bit 2	bit 10	Minutes	
		bit 3	bit 11	Hours	
		bit 4	bit 12	Days	
		bit 5	bit 13	Month	
		bit 6	bit 14	Years	
		bit 7	bit 15	Centuries	Not used

(1) If a single expansion module is missing at power-on, then all expansion module bits are set to 1 (detected error).
S Controlled by the system
U Controlled by the user
SIM Applied in the simulator

System Words	Function	Description	Control
%SW63	EXCH1 block error code	EXCH1 error code: 0 - operation was successful 1 - number of bytes to be transmitted is too great (> 250) 2 - transmission table too small 3 - word table too small 4 - receive table overflowed 5 - time-out elapsed 6 - transmission 7 - bad command within table 8 - selected port not configured/available 9 - reception error: This error code reflect a bad reception frame. It can be caused due to a wrong configuration in the physical parameters (for example, parity, data bits, baudrate, and so on) or a bad physical connection causing signal degradation. 10 - cannot use %KW if receiving 11 - transmission offset larger than transmission table 12 - reception offset larger than reception table 13 - controller stopped EXCH processing	S
%SW64	EXCH2 block error code	EXCH2 error code: See %SW63.	S
%SW65	EXCH3 block error code	1-4, 6-13: See %SW63. (Note that error code 5 is invalid and replaced by the Ethernet-specific error codes 109 and 122 described below.) The following are Ethernet-specific error codes: 101 - no such IP address 102 - the TCP connection is broken 103 - no socket available (all connection channels are busy) 104 - network is down 105 - network cannot be reached 106 - network dropped connection on reset 107 - connection aborted by peer device 108 - connection reset by peer device 109 - connection time-out elapsed 110 - rejection on connection attempt 111 - host is down 120 - unknown index (remote device is not indexed in configuration table) 121 - fatal (MAC, chip, duplicate IP) 122 - receiving process timed-out after data was sent 123 - Ethernet initialization in progress	S

(1) If a single expansion module is missing at power-on, then all expansion module bits are set to 1 (detected error).
S Controlled by the system
U Controlled by the user
SIM Applied in the simulator

System Words	Function	Description	Control
%SW67	Function and type of controller	Contains the logic controller code ID. For more information, refer to the M221 Logic Controller Code ID table *(see page 174)*.	S, SIM
%SW70	Scan time micro sec microseconds resolution	Shows execution time of the last controller scan cycle (in µs). **NOTE:** This time corresponds to the time elapsed between the start (acquisition of inputs) and the end (update of outputs) of a master task scan cycle. If the scan time is 2,250 ms, the %SW30 will be 2 and the %SW70 will be 250.	–
%SW71	Max. scan time microseconds resolution	Shows execution time of the longest controller scan cycle since the last cold start (in ms). **NOTE:** This time corresponds to the time elapsed between the start (acquisition of inputs) and the end (update of outputs) of a scan cycle. If the scan time is 2,250 ms, the %SW31 will be 2 and the %SW71 will be 250.	–
%SW72	Min. scan time microseconds resolution	Shows execution time of the shortest controller scan cycle since the last cold start (in ms). **NOTE:** This time corresponds to the time elapsed between the start (acquisition of inputs) and the end (update of outputs) of a scan cycle. If the scan time is 2,250 ms, the %SW32 will be 2 and the %SW72 will be 250.	–
%SW75	Load of processor	Shows percentage of processing load.	S
%SW76 to %SW79	Down counters 1-4	These 4 words serve as 1 ms timers. They are decremented individually by the system every ms if they have a positive value. This gives 4 down counters down counting in ms which is equal to an operating range of 1 ms to 32767 ms. Setting bit 15 to 1 can stop decrementation.	S and U, SIM
%SW80	Embedded Analog Input	Each bit indicate status of the embedded analog inputs.	S
%SW94 %SW95	Application's signature %MW60028-%MW60034	In case of an application change, in terms of configuration or programming data, the signature (sum of all checksums) changes so. If %SW94 = 91F3 in hexadecimal, the application's signature is 91F3 in hexadecimal.	S, SIM

(1) If a single expansion module is missing at power-on, then all expansion module bits are set to 1 (detected error).
S Controlled by the system
U Controlled by the user
SIM Applied in the simulator

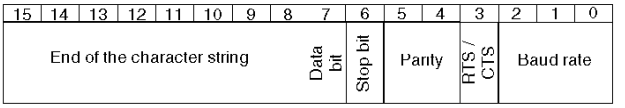
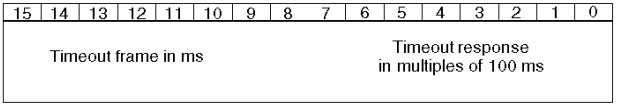
System Words	Function	Description	Control

%SW96	Command and/or diagnostics for save/restore function of application program	• Bit [1]: This bit is set by the firmware to indicate when the save is complete:
		  • Set to 1 if the backup is complete.
		  • Set to 0 if a new backup request is asked for.
		• Bit [2]: Back up detected error, refer to bits 8, 9, 10, 12 and 14 for further information:
		  • Set to 1 if an error appeared.
		  • Set to 0 if a new backup request is asked for.
		• Bit [6]: Set to 1 if the controller contains a valid application in RAM.
		• Bit [10]: Difference between internal RAM and Flash memory (1 = yes).
		  • Set to 1 if there is a difference.
		• Bit [12]: Indicates if a restore error has occurred:
		  • Set to 1 if an error is detected.
		• Bit [14]: Indicates if a Flash memory write error has occurred:
		  • Set to 1 if an error is detected.
			S, SIM

%SW98	Post configuration status (Serial Line 1)	The bits are set to 1 when the post configuration was applied for the parameter: • Bit[0]: Hardware option (RS485 or RS232) • Bit[1]: Baudrate • Bit[2]: Parity • Bit[3]: Data size • Bit[4]: Number of stop bits • Bit[5]: Modbus address • Bit[6]: Polarization (if available in the port)	
%SW99	Post configuration status (Serial Line 2)	The bits are set to 1 when the post configuration was applied for the parameter: • Bit[0]: Hardware option (RS485) • Bit[1]: Baudrate • Bit[2]: Parity • Bit[3]: Data size • Bit[4]: Number of stop bits • Bit[5]: Modbus address • Bit[6]: Polarization (if available in the port)	

(1) If a single expansion module is missing at power-on, then all expansion module bits are set to 1 (detected error).
S Controlled by the system
U Controlled by the user
SIM Applied in the simulator

System Words	Function	Description	Control
%SW100	Post configuration status (Ethernet)	The bits are set to 1 when the post configuration was applied for the parameter: ● Bit[0]: IP mode (fixed, DHCP, or BOOTP) ● Bit[1]: IP address ● Bit[2]: Network submask ● Bit[3]: Default gateway ● Bit[4]: Net name	
%SW101 %SW102	Value of the Modbus address port	When bit %S101 is set to 1, you can change the Modbus address of port 1 or port 2. The address of port 1 is %SW101, and that of port 2 is %SW102. **NOTE:** In online mode, the address of port 2 cannot be changed using system bit %S101 and system word %SW102.	S
(1) If a single expansion module is missing at power-on, then all expansion module bits are set to 1 (detected error). **S** Controlled by the system **U** Controlled by the user **SIM** Applied in the simulator			

System Words	Function	Description	Control
%SW103 %SW104	Configuration for use of the ASCII protocol	When bit %S103 (Comm 1) or %S104 (Comm 2) is set to 1, the ASCII protocol is used. System word %SW103 (Comm 1) or %SW104 (Comm 2) must be set according to the elements below:  ● Baud rate: ● 0: 1200 baud, ● 1: 2400 baud, ● 2: 4800 baud, ● 3: 9600 baud, ● 4: 19200 baud, ● 5: 38400 baud. ● RTS/CTS: ● 0: disabled, ● 1: enabled. ● Parity: ● 00: none, ● 10: odd, ● 11: even. ● Stop bit: ● 0: 1 stop bit, ● 1: 2 stop bits. ● Data bits: ● 0: 7 data bits, ● 1: 8 data bits.	S
%SW105 %SW106	Configuration for use of the ASCII protocol	When bit %S103 (Comm 1) or %S104 (Comm 2) is set to 1, the ASCII protocol is used. System word %SW105 (Comm 1) or %SW106 (Comm 2) must be set according to the elements below:  	S

(1) If a single expansion module is missing at power-on, then all expansion module bits are set to 1 (detected error).
S Controlled by the system
U Controlled by the user
SIM Applied in the simulator

System Words	Function	Description	Control
%SW107 %SW108 %SW109	MAC address	Shows the controller MAC address (only references with Ethernet channel).	S, P
%SW114	Enable schedule blocks	Enables or disables operation of schedule blocks by the user program. Bit 0: 1 = enables schedule block number 0 ... Bit 15: 1 = enables schedule block number 15 Initially all schedule blocks are enabled. If schedule blocks are configured the default value is FFFF If no schedule blocks are configured, the default value is 0.	S and U, SIM
%SW118	Base controller status word	Shows conditions on base controller. Bit 13: 0 = Configuration error (I/O extension configured but absent or inoperative). In this case, the bus is not started (the bus will be locked until the bit13 of %SW118 is equal to zero). Bit 14: 0 = Runtime error (last exchange with at least one TM2/TM3 failed). All the other bits of this word are set to 1 and are reserved. For a controller which operates properly, the value of this word is FFFFh.	S, SIM
%SW120	Expansion I/O module status (1)	One bit per module. Address 0 = bit 0 1 = detected error 0 = OK **NOTE:** If the value of %SW120 is not 0 just after the controller start, the bus TM2/TM3 will not start and the output of the extended modules cannot be managed. If the value of %SW120 changes, it indicates a status detected error on an expansion I/O module.	S, SIM
%SW121 %SW122	Configuration for use of ASCII protocol	When bit %S103 (Comm 1) or %S104 (Comm 2) is set to 1, the ASCII protocol is used. You can change the ASCII frame size of port 1 or port 2. The ASCII frame size of port 1 is %SW121, and that of port 2 is %SW122. The value is used only on EXCH instruction start. Then, if some bytes are already received, you cannot stop the reception until the last byte.	U
%SW130	Event execution time	Shows the last execution time of the event input %I0.2 in µs.	S
%SW131	Event execution time	Shows the last execution time of the event input %I0.3 in µs.	S

(1) If a single expansion module is missing at power-on, then all expansion module bits are set to 1 (detected error).
S Controlled by the system
U Controlled by the user
SIM Applied in the simulator

System Words	Function	Description	Control
%SW132	Event execution time	Shows the last execution time of the event input %I0.4 in µs.	S
%SW133	Event execution time	Shows the last execution time of the event input %I0.5 in µs.	S
%SW134	Event execution time	Shows the last execution time of the event Threshold 0 of HSC0 in µs.	S
%SW135	Event execution time	Shows the last execution time of the event Threshold 1 of HSC0 in µs.	S
%SW136	Event execution time	Shows the last execution time of the event Threshold 0 of HSC1 in µs.	S
%SW137	Event execution time	Shows the last execution time of the event Threshold 1 of HSC1 in µs.	S
%SW138	Periodic task execution time	Shows the last execution time of the periodic task in µs.	S
%SW139	Embedded digital output protection	Shows the protection error status of output blocks: Bit0 = 1 - O0 - O3 protect error - Block0 Bit1 = 1 - O4 - O7 protect error - Block1 Bit2 = 1 - O8 - O11 protect error - Block2 Bit3 = 1 - O12 - O15 protect error - Block3	S
%SW148	Number of persistent variables	Maximum 1,000 variables	U
%SW149	Event execution time	Shows the last execution time of the event input %I0.2 in ms.	S
%SW150	Event execution time	Shows the last execution time of the event input %I0.3 in ms.	S
%SW151	Event execution time	Shows the last execution time of the event input %I0.4 in ms.	S
%SW152	Event execution time	Shows the last execution time of the event input %I0.5 in ms.	S
%SW153	Event execution time	Shows the last execution time of the event threshold 0 of HSC0 in ms.	S
%SW154	Event execution time	Shows the last execution time of the event threshold 1 of HSC0 in ms.	S

(1) If a single expansion module is missing at power-on, then all expansion module bits are set to 1 (detected error).
S Controlled by the system
U Controlled by the user
SIM Applied in the simulator

System Words	Function	Description	Control
%SW155	Event execution time	Shows the last execution time of the event threshold 0 of HSC1 in ms.	S
%SW156	Event execution time	Shows the last execution time of the event threshold 1 of HSC1 in ms.	S
%SW157	FAST execution time	Shows the last execution time of the FAST task in ms.	S
%SW158	FAST average time	Shows the average execution time (last 5 times) of the FAST task in ms.	S
%SW159	Event 0 average time	Shows the average execution time (last 5 times) of event 0 in ms.	S
%SW160	Event 1 average time	Shows the average execution time (last 5 times) of event 1 in ms.	S
%SW161	Event 2 average time	Shows the average execution time (last 5 times) of event 2 in ms.	S
%SW162	Event 3 average time	Shows the average execution time (last 5 times) of event 3 in ms.	S
%SW163	Event 4 average time	Shows the average execution time (last 5 times) of event 4 in ms.	S
%SW164	Event 5 average time	Shows the average execution time (last 5 times) of event 5 in ms.	S
%SW165	Event 6 average time	Shows the average execution time (last 5 times) of event 6 in ms.	S
%SW166	Event 7 average time	Shows the average execution time (last 5 times) of event 7 in ms.	S

(1) If a single expansion module is missing at power-on, then all expansion module bits are set to 1 (detected error).
S Controlled by the system
U Controlled by the user
SIM Applied in the simulator

M221 Logic Controller Code ID

This table shows the code IDs of the M221 Logic Controller references:

Reference	Code ID
TM221M16R•	0x0780
TM221ME16R•	0x0781
TM221M16T•	0x0782
TM221ME16T•	0x0783
TM221M32TK	0x0784
TM221ME32TK	0x0785
TM221C16R	0x0786
TM221CE16R	0x0787
TM221C16T	0x0788
TM221CE16T	0x0789
TM221C24R	0x078A
TM221CE24R	0x078B
TM221C24T	0x078C
TM221CE24T	0x078D
TM221C40R	0x078E
TM221CE40R	0x078F
TM221C40T	0x0790
TM221CE40T	0x0791

Glossary

0-9

2-phase counter
Uses 2 input counter signals to count up and count down.

A

analog input
Converts received voltage or current levels into numerical values. You can store and process these values within the logic controller.

analog output
Converts numerical values within the logic controller and sends out proportional voltage or current levels.

application
A program including configuration data, symbols, and documentation.

B

BOOTP
(*bootstrap protocol*) A UDP network protocol that can be used by a network client to automatically obtain an IP address (and possibly other data) from a server. The client identifies itself to the server using the client MAC address. The server, which maintains a pre-configured table of client device MAC addresses and associated IP addresses, sends the client its pre-configured IP address. BOOTP was originally used as a method that enabled diskless hosts to be remotely booted over a network. The BOOTP process assigns an infinite lease of an IP address. The BOOTP service utilizes UDP ports 67 and 68.

C

configuration
The arrangement and interconnection of hardware components within a system and the hardware and software parameters that determine the operating characteristics of the system.

controller
Automates industrial processes (also known as programmable logic controller or programmable controller).

D

DHCP

(*dynamic host configuration protocol*) An advanced extension of BOOTP. DHCP is more advanced, but both DHCP and BOOTP are common. (DHCP can handle BOOTP client requests.)

digital I/O

(*digital input/output*) An individual circuit connection at the electronic module that corresponds directly to a data table bit. The data table bit holds the value of the signal at the I/O circuit. It gives the control logic digital access to I/O values.

DWORD

(*double word*) Encoded in 32-bit format.

E

expansion bus

An electronic communication bus between expansion I/O modules and a controller.

G

GRAFCET

The functioning of a sequential operation in a structured and graphic form.

This is an analytical method that divides any sequential control system into a series of steps, with which actions, transitions, and conditions are associated.

I

I/O

(*input/output*)

IEC 61131-3

Part 3 of a 3-part IEC standard for industrial automation equipment. IEC 61131-3 is concerned with controller programming languages and defines 2 graphical and 2 textual programming language standards. The graphical programming languages are ladder diagram and function block diagram. The textual programming languages include structured text and instruction list.

IL

(*instruction list*) A program written in the language that is composed of a series of text-based instructions executed sequentially by the controller. Each instruction includes a line number, an instruction code, and an operand (refer to IEC 61131-3).

instruction list language

A program written in the instruction list language that is composed of a series of text-based instructions executed sequentially by the controller. Each instruction includes a line number, an instruction code, and an operand (see IEC 61131-3).

L

ladder diagram language

A graphical representation of the instructions of a controller program with symbols for contacts, coils, and blocks in a series of rungs executed sequentially by a controller (see IEC 61131-3).

LAN

(*local area network*) A short-distance communications network that is implemented in a home, office, or institutional environment.

LD

(*ladder diagram*) A graphical representation of the instructions of a controller program with symbols for contacts, coils, and blocks in a series of rungs executed sequentially by a controller (refer to IEC 61131-3).

M

MAST

A processor task that is run through its programming software. The MAST task has 2 sections:

- **IN:** Inputs are copied to the IN section before execution of the MAST task.
- **OUT:** Outputs are copied to the OUT section after execution of the MAST task.

Modbus

The protocol that allows communications between many devices connected to the same network.

P

periodic execution

The task is executed either cyclically or periodically. In periodic mode, you determine a specific time (period) in which the task is executed. If it is executed under this time, a waiting time is generated before the next cycle. If it is executed over this time, a control system indicates the overrun. If the overrun is too high, the controller is stopped.

PID

(*proportional, integral, derivative*) A generic control loop feedback mechanism (controller) widely used in industrial control systems.

program

The component of an application that consists of compiled source code capable of being installed in the memory of a logic controller.

protocol

A convention or standard definition that controls or enables the connection, communication, and data transfer between 2 computing system and devices.

R

RTC

(*real-time clock*) A battery-backed time-of-day and calender clock that operates continuously, even when the controller is not powered for the life of the battery.

T

TCP

(*transmission control protocol*) A connection-based transport layer protocol that provides a simultaneous bi-directional transmission of data. TCP is part of the TCP/IP protocol suite.

Index

Symbols

A

B

C

D

E

F

H

I

M

O

P

R

S

U

W