Modicon M580

Hardware

Reference Manual

Original instructions
EIO0000001578.15
06/2024

Legal Information

The information provided in this document contains general descriptions, technical characteristics and/or recommendations related to products/solutions.

This document is not intended as a substitute for a detailed study or operational and site-specific development or schematic plan. It is not to be used for determining suitability or reliability of the products/solutions for specific user applications. It is the duty of any such user to perform or have any professional expert of its choice (integrator, specifier or the like) perform the appropriate and comprehensive risk analysis, evaluation and testing of the products/solutions with respect to the relevant specific application or use thereof.

The Schneider Electric brand and any trademarks of Schneider Electric SE and its subsidiaries referred to in this document are the property of Schneider Electric SE or its subsidiaries. All other brands may be trademarks of their respective owner.

Schneider Electric reserves the right to make changes or updates with respect to or in the content of this document or the format thereof, at any time without notice.

To the extent permitted by applicable law, no responsibility or liability is assumed by Schneider Electric and its subsidiaries for any errors or omissions in the informational content of this document, as well as any non-intended use or misuse of the content thereof.

Table of Contents

Safety Information

Important Information

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

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

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

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

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

CAUTION
CAUTION indicates a hazardous situation which, if not avoided, **could 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.

Before You Begin

Do not use this product on machinery lacking effective point-of-operation guarding. Lack of effective point-of-operation guarding on a machine can result in serious injury to the operator of that machine.

⚠WARNING

UNGUARDED EQUIPMENT

- Do not use this software and related automation equipment on equipment which does not have point-of-operation protection.
- Do not reach into machinery during operation.

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

This automation equipment and related software is used to control a variety of industrial processes. The type or model of automation equipment suitable for each application will vary depending on factors such as the control function required, degree of protection required, production methods, unusual conditions, government regulations, etc. In some applications, more than one processor may be required, as when backup redundancy is needed.

Only you, the user, machine builder or system integrator can be aware of all the conditions and factors present during setup, operation, and maintenance of the machine and, therefore, can determine the automation equipment and the related safeties and interlocks which can be properly used. When selecting automation and control equipment and related software for a particular application, you should refer to the applicable local and national standards and regulations. The National Safety Council's Accident Prevention Manual (nationally recognized in the United States of America) also provides much useful information.

In some applications, such as packaging machinery, additional operator protection such as point-of-operation guarding must be provided. This is necessary if the operator's hands and other parts of the body are free to enter the pinch points or other hazardous areas and serious injury can occur. Software products alone cannot protect an operator from injury. For this reason the software cannot be substituted for or take the place of point-of-operation protection.

Ensure that appropriate safeties and mechanical/electrical interlocks related to point-of-operation protection have been installed and are operational before placing the equipment into service. All interlocks and safeties related to point-of-operation protection must be coordinated with the related automation equipment and software programming.

NOTE: Coordination of safeties and mechanical/electrical interlocks for point-of-operation protection is outside the scope of the Function Block Library, System User Guide, or other implementation referenced in this documentation.

Start-up and Test

Before using electrical control and automation equipment for regular operation after installation, the system should be given a start-up test by qualified personnel to verify correct operation of the equipment. It is important that arrangements for such a check are made and that enough time is allowed to perform complete and satisfactory testing.

⚠WARNING

EQUIPMENT OPERATION HAZARD

- Verify that all installation and set up procedures have been completed.
- Before operational tests are performed, remove all blocks or other temporary holding means used for shipment from all component devices.
- Remove tools, meters, and debris from equipment.

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

Follow all start-up tests recommended in the equipment documentation. Store all equipment documentation for future references.

Software testing must be done in both simulated and real environments.

Verify that the completed system is free from all short circuits and temporary grounds that are not installed according to local regulations (according to the National Electrical Code in the U.S.A, for instance). If high-potential voltage testing is necessary, follow recommendations in equipment documentation to prevent accidental equipment damage.

Before energizing equipment:

- Remove tools, meters, and debris from equipment.

- Close the equipment enclosure door.
- Remove all temporary grounds from incoming power lines.
- Perform all start-up tests recommended by the manufacturer.

Operation and Adjustments

The following precautions are from the NEMA Standards Publication ICS 7.1-1995:

(In case of divergence or contradiction between any translation and the English original, the original text in the English language will prevail.)

- Regardless of the care exercised in the design and manufacture of equipment or in the selection and ratings of components, there are hazards that can be encountered if such equipment is improperly operated.
- It is sometimes possible to misadjust the equipment and thus produce unsatisfactory or unsafe operation. Always use the manufacturer's instructions as a guide for functional adjustments. Personnel who have access to these adjustments should be familiar with the equipment manufacturer's instructions and the machinery used with the electrical equipment.
- Only those operational adjustments required by the operator should be accessible to the operator. Access to other controls should be restricted to prevent unauthorized changes in operating characteristics.

About the Book

Document Scope

This document provides detailed information about the Modicon M580 programmable automation controller (PAC). These topics are also discussed:

- Install a local backplane in the M580 controller system.
- Configure the M580 PAC.
- The controller performs Ethernet I/O scanning of both RIO and DIO logic without affecting network determinism.

Validity Note

This document has been updated for the release of EcoStruxure™ Control Expert 16.0 with ControlExpert_V160_HF001 M580 Safety and BME•58•••• firmware version 4.21.

The characteristics of the products described in this document are intended to match the characteristics that are available on www.se.com. As part of our corporate strategy for constant improvement, we may revise the content over time to enhance clarity and accuracy. If you see a difference between the characteristics in this document and the characteristics on www.se.com, consider www.se.com to contain the latest information.

Related Documents

Title of Documentation	Reference Number
Control Panel Technical Guide, How to protect a machine from malfunctions due to electromagnetic disturbance	CPTG003_EN (ENG) CPTG003_FR (FRE)
Electrical installation guide	
Modicon M580 Standalone, System Planning Guide for Frequently Used Architectures	HRB62666 (ENG) HRB65318 (FRE) HRB65319 (GER) HRB65320 (ITA) HRB65321 (SPA) HRB65322 (CHS)
Modicon M580, System Planning Guide for Complex Topologies	NHA58892 (ENG) NHA58893 (FRE) NHA58894 (GER)

Title of Documentation	Reference Number
	NHA58895 (ITA) NHA58896 (SPA) NHA58897 (CHS)
Modicon M580 Hot Standby, System Planning Guide for Frequently Used Architectures	NHA58880 (ENG) NHA58881 (FRE) NHA58882 (GER) NHA58883 (ITA) NHA58884 (SPA) NHA58885 (CHS)
Modicon M580, Open Ethernet Network, System Planning Guide	EIO0000004111 (English)
Modicon M580 BMENOC0301/11, Ethernet Communication Module, Installation and Configuration Guide	HRB62665 (ENG) HRB65311 (FRE) HRB65313 (GER) HRB65314 (ITA) HRB65315 (SPA) HRB65316 (CHS)
Modicon M580, RIO Modules, Installation and Configuration Guide	EIO0000001584 (ENG) EIO0000001585 (FRE) EIO0000001586 (GER) EIO0000001587 (ITA) EIO0000001588 (SPA) EIO0000001589 (CHS)
Modicon M580, M340, and X80 I/O Platforms, Standards and Certifications	EIO0000002726 (ENG) EIO0000002727 (FRE) EIO0000002728 (GER) EIO0000002730 (ITA) EIO0000002729 (SPA) EIO0000002731 (CHS)
M580 BMENOS0300, Network Option Switch, Installation and Configuration Guide	NHA89117 (ENG) NHA89119 (FRE) NHA89120 (GER) NHA89121 (ITA) NHA89122 (SPA) NHA89123 (CHS)
Modicon eX80, BMEAHI0812 HART Analog Input Module & BMEAHO0412 HART Analog Output Module, User Guide	EAV16400 (ENG) EAV28404 (FRE) EAV28384 (GER) EAV28413 (ITA) EAV28360 (SPA) EAV28417 (CHS)
EcoStruxure™ Automation Device Maintenance, User Guide	EIO0000004033 (ENG) EIO0000004048 (FRE) EIO0000004046 (GER) EIO0000004049 (ITA) EIO0000004047 (SPA) EIO0000004050 (CHS)
Unity Loader, User Guide	33003805 (ENG) 33003806 (FRE)

Title of Documentation	Reference Number
	33003807 (GER) 33003809 (ITA) 33003808 (SPA) 33003810 (CHS)
EcoStruxure™ Control Expert, Operating Modes	33003101 (ENG) 33003102 (FRE) 33003103 (GER) 33003104 (SPA) 33003696 (ITA) 33003697 (CHS)
EcoStruxure™ Control Expert, Program Languages and Structure, Reference Manual	35006144 (ENG) 35006145 (FRE) 35006146 (GER) 35013361 (ITA) 35006147 (SPA) 35013362 (CHS)
Modicon X80 Racks and Power Supplies, Hardware, Reference Manual	EIO0000002626 (ENG) EIO0000002627 (FRE) EIO0000002628 (GER) EIO0000002630 (ITA) EIO0000002629 (SPA) EIO0000002631 (CHS)
Modicon Controllers Platform Cyber Security, Reference Manual	EIO0000001999 (ENG) EIO0000002001 (FRE) EIO0000002000 (GER) EIO0000002002 (ITA) EIO0000002003 (SPA) EIO0000002004 (CHS)

Product Related Information

⚠⚠ DANGER

HAZARD OF ELECTRIC SHOCK, EXPLOSION OR ARC FLASH

- Disconnect all power from all equipment including connected devices prior to removing any covers or doors, or installing or removing any accessories, hardware, cables, or wires except under the specific conditions specified in the appropriate hardware guide for this equipment.
- Always use a properly rated voltage sensing device to confirm the power is off where and when indicated.
- Replace and secure all covers, accessories, hardware, cables, and wires and confirm that a proper ground connection exists before applying power to the unit.
- Use only the specified voltage when operating this equipment and any associated products.

Failure to follow these instructions will result in death or serious injury.

⚠ WARNING

LOSS OF CONTROL

- Perform a Failure Mode and Effects Analysis (FMEA), or equivalent risk analysis, of your application, and apply preventive and detective controls before implementation.
- Provide a fallback state for undesired control events or sequences.
- Provide separate or redundant control paths wherever required.
- Supply appropriate parameters, particularly for limits.
- Review the implications of transmission delays and take actions to mitigate them.
- Review the implications of communication link interruptions and take actions to mitigate them.
- Provide independent paths for control functions (for example, emergency stop, over-limit conditions, and error conditions) according to your risk assessment, and applicable codes and regulations.
- Apply local accident prevention and safety regulations and guidelines.[1]
- Test each implementation of a system for proper operation before placing it 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.

Trademarks

QR Code is a registered trademark of DENSO WAVE INCORPORATED in Japan and other countries.

Terminology Derived from Standards

The technical terms, terminology, symbols and the corresponding descriptions in the information contained herein, or that appear in or on the products themselves, are generally derived from the terms or definitions of international standards.

In the area of functional safety systems, drives, and general automation, this may include, but is not limited to, terms such as *safety*, *safety function*, *safe state*, *fault*, *fault reset*, *malfunction*, *failure*, *error*, *error message*, *dangerous*, etc.

Among others, these standards include:

Standard	Description
IEC 61131-2:2007	Programmable controllers, part 2: Equipment requirements and tests.
ISO 13849-1:2023	Safety of machinery: Safety related parts of control systems. General principles for design.
EN 61496-1:2013	Safety of machinery: Electro-sensitive protective equipment. Part 1: General requirements and tests.
ISO 12100:2010	Safety of machinery - General principles for design - Risk assessment and risk reduction
EN 60204-1:2006	Safety of machinery - Electrical equipment of machines - Part 1: General requirements
ISO 14119:2013	Safety of machinery - Interlocking devices associated with guards - Principles for design and selection
ISO 13850:2015	Safety of machinery - Emergency stop - Principles for design
IEC 62061:2021	Safety of machinery - Functional safety of safety-related electrical, electronic, and electronic programmable control systems
IEC 61508-1:2010	Functional safety of electrical/electronic/programmable electronic safety-related systems: General requirements.
IEC 61508-2:2010	Functional safety of electrical/electronic/programmable electronic safety-related systems: Requirements for electrical/electronic/programmable electronic safety-related systems.
IEC 61508-3:2010	Functional safety of electrical/electronic/programmable electronic safety-related systems: Software requirements.
IEC 61784-3:2021	Industrial communication networks - Profiles - Part 3: Functional safety fieldbuses - General rules and profile definitions.
2006/42/EC	Machinery Directive
2014/30/EU	Electromagnetic Compatibility Directive
2014/35/EU	Low Voltage Directive

In addition, terms used in the information contained herein may tangentially be used as they are derived from other standards such as:

Standard	Description
IEC 60034 series	Rotating electrical machines
IEC 61800 series	Adjustable speed electrical power drive systems
IEC 61158 series	Digital data communications for measurement and control – Fieldbus for use in industrial control systems

Finally, the term *zone of operation* may be used in conjunction with the description of specific hazards, and is defined as it is for a hazard zone or danger zone in the Machinery Directive (2006/42/EC) and ISO 12100:2010.

NOTE: The aforementioned standards may or may not apply to the specific products cited in the present documentation. For more information concerning the individual standards applicable to the products described herein, see the characteristics tables for those product references.

Information on Non-Inclusive or Insensitive Terminology

As a responsible, inclusive company, Schneider Electric is constantly updating its communications and products that contain non-inclusive or insensitive terminology. However, despite these efforts, our content may still contain terms that are deemed inappropriate by some customers.

Modicon M580 PACs

What's in This Part

Introduction

This part provides information about the Modicon M580 Programmable Automation Controller (PAC), including physical and operational characteristics.

M580 PACs

What's in This Chapter

Introduction

This chapter introduces you to the physical and functional characteristics of the M580 PACs.

Functional Characteristics of M580 PACs

Introduction

This section describes the functional characteristics of M580 PACs. Performance, electrical characteristics, and memory capacities of the different controllers are detailed.

Introduction

Role of the Controller in a Control System

In a modular PAC system, the controller controls and processes the application. The local backplane identifies the controller. In addition to the controller, the local backplane contains a power supply module and may contain communication processing modules and input/output (I/O) modules.

The controller is in charge of:

- configuring the modules and devices present in the controller configuration
- processing the application
- reading the inputs at the beginning of tasks and applying the outputs at the end of tasks
- managing explicit and implicit communications

Modules may reside in the local backplane with the controller or they may be installed in remote drops at a distance from the local backplane. The controller has built-in capabilities to act as the RIO processor that manages communications between the controller and the Quantum and X80 EIO adapter modules that are installed in each remote drop.

Devices can be connected to the PAC network as either DIO clouds or DIO sub-rings.

For detailed information about the various architectures that the M580 network supports, refer to chapter *Planning and Designing a Typical M580 Network* (see Modicon M580 Standalone, System Planning Guide for Frequently Used Architectures). For a detailed description of the X80 EIO adapter modules and the options they provide for installing a remote drop, refer to Modicon M580, RIO Modules, Installation and Configuration Guide.

Functional Considerations

The controller solves control logic for the I/O modules and distributed equipment in the system. Choose a controller based on several operating characteristics:

- memory size
- processing power: the number of I/O points or channels that it can manage, page 27
- the speed at which the controller can execute the control logic, page 36
- communication capabilities: the types of Ethernet ports on the controller, page 73
- the number of local I/O modules and RIO drops that it can support, page 27
- the ability to function in harsh environments: (Three controllers are hardened to operate over extended temperature ranges and in dirty or corrosive environments.)
- network configuration (standalone or Hot Standby)

Standalone Controllers

This is a list of the available controllers. Some are available in both standard and industrially hardened modules. Industrially hardened modules have the letter H appended to the module name. The letter C at the end of the module name indicates a conformal coating for harsh environments:

- BMEP581020, BMEP581020H
- BMEP582020, BMEP582020H
- BMEP582040, BMEP582040H, BMEP582040S
- BMEP583020
- BMEP583040
- BMEP584020
- BMEP584040, BMEP584040S
- BMEP585040(C), BMEP585040(C)C

- BMEP586040, BMEP586040C, BMEP586040S

Controllers ending with “S” are safety-related. Refer to the Modicon M580 Safety System Planning Guide for a description of safety controllers.

Hot Standby Controllers

These controllers are compatible with M580 Hot Standby systems:

- BMEH582040, BMEH582040C, BMEH582040S
- BMEH584040, BMEH584040C, BMEH584040S
- BMEH586040(C), BMEH586040(C)C, BMEH586040S

NOTE: For detailed information about M580 Hot Standby configurations, refer to the *Modicon M580 Hot Standby System Planning Guide for Frequently Used Architectures*.

Altitude Operating Conditions

The characteristics apply to the controller for use at altitude up to 2000 m (6560 ft). When the controller operates above 2000 m (6560 ft), apply additional derating.

For detailed information, refer to chapter *Operating and Storage Conditions* (see Modicon M580, M340, and X80 I/O Platforms, Standards and Certifications).

Performance Characteristics

Introduction

M580 PACs have an embedded DIO scanner service to manage distributed equipment on the M580 device network. Some M580 PACs also have an embedded RIO scanner service to manage RIO drops.

To manage RIO drops on the device network, select one of these controllers with Ethernet I/O scanner service (both RIO and DIO scanner service):

- BMEP582040, BMEP582040H
- BMEP583040
- BMEP584040
- BMEP585040(C), BMEP585040(C)C
- BMEP586040, BMEP586040C
- BMEH582040, BMEH582040C

- BMEH584040, BMEH584040C
- BMEH586040(C), BMEH586040(C)C

Embedded Ethernet I/O scanner services are configured via the controller IP configuration, page 148.

NOTE: Some of this information applies to M580 Hot Standby configurations. For more information, refer to the *Modicon M580 Hot Standby System Planning Guide for Frequently Used Architectures* (see Modicon M580 Standalone, System Planning Guide for Frequently Used Architectures).

Controller Characteristics

These tables show the key characteristics of the M580 standalone and Hot Standby controllers. These characteristics represent the maximum values that a specific controller can manage in the M580 PAC system.

NOTE:

- The values in these tables may not be achieved depending on the I/O density and the number of available backplane slots.
- The following tables do not include safety controllers. Refer to the Modicon M580 Safety System Planning Guide (see Modicon M580, Safety System Planning Guide) for the performance characteristics of safety controllers.

Standalone Controllers:

Maximum number of ...	Reference (BMEP58 ...)								
	1020 (H)	**2020 (H)**	**2040 (H)**	**3020**	**3040**	**4020**	**4040**	**5040 (C)**	**6040 (C)**
discrete I/O channels	1024	2048	2048	3072	3072	4096	4096	5120	6144
analog I/O channels	256	512	512	768	768	1024	1024	1280	1536
expert channels	36	72	72	108	108	144	144	180	216
distributed devices[4]	64	128	64	128	64	128	64	64	64
Ethernet communication modules (including BMENOC0301/ BMENOC0311 modules, but not the controller)	2	2	2	3	3	4[(1)]	4[(1)]	6[(1)]	6[(1)]
local backplanes (main backplane + extended backplane)	4	4	4	8	8	8	8	8	8

Maximum number of ...	Reference (BMEP58 ...)								
	1020 (H)	**2020 (H)**	**2040 (H)**	**3020**	**3040**	**4020**	**4040**	**5040 (C)**	**6040 (C)**
RIO drops, page 29 (maximum of two backplanes per drop) (main backplane + extended backplane)	–	–	8(2)	–	16(2)	–	16(3)	31(3)	31(3)
Ethernet ports:									
• service	1	1	1	1	1	1	1	1	1
• RIO or distributed equipment	–	–	2	–	2	–	2	2	2
• distributed equipment	2	2	–	2	–	2	–	–	–

– (not available)
H (hardened)
C (coated version)
(1) Only three of these modules can be BMENOC0301/BMENOC0311 modules. All other are BMX Ethernet modules.

(2) Supports BM•CRA312•0 adapter modules.

(3) Supports BM•CRA312•0 and 140CRA31200 adapter modules.

(4) Of these connections: 3 are reserved for local slaves; the remainder are available for scanning distributed equipment.

Hot Standby Controllers:

Maximum number of ...	Reference (BMEH58 ...)		
	2040(C)	**4040(C)**	**6040(C)**
distributed devices	64	64	64
Ethernet communication modules (including BMENOC0301/ BMENOC0311 modules, but not the controller)	2	4(1)	6(1)
local backplanes (main backplane + extended backplane)	1	1	1
RIO drops, page 29 (maximum of two backplanes per drop) (main backplane + extended backplane)	8(2)	16(3)	31(3)
Ethernet ports:			
• service	1	1	1
• RIO or distributed equipment	2	2	2

Maximum number of ...	Reference (BMEH58 ...)		
	2040(C)	4040(C)	6040(C)
• distributed equipment	0	0	0
1. Only three of these communication modules can be BMENOC0301/BMENOC0311 modules. 2. Supports BM•CRA312•0 adapter modules. 3. Supports BM•CRA312•0 and 140CRA31200 adapter modules.			

RIO Drop Maximum Configuration

The maximum number of channels in an RIO drop depends on the eX80 EIO adapter module:

EIO adapter	Maximum number of Channels			
	Discrete	Analog	Expert	Sensor bus
BMXCRA31200	128	16	–	–
BMXCRA31210	1024	256	36	2
BMECRA31210	1024	256	36	2

NOTE: The number of available channels could differ from the maximum values shown because the values depend on the controller reference and the other modules in the same drop. More information is given in Modicon X80 I/O Modules (see Modicon M580, RIO Modules, Installation and Configuration Guide).

To configure Quantum RIO drops, refer to the Quantum EIO installation and configuration guide (see Quantum EIO, Remote I/O Modules, Installation and Configuration Guide).

Maximum Internal Memory Size

Program and Data Memory (Standalone). This table shows the program and data memory capacity for M580 standalone controllers:

Memory Size	Reference (BMEP58 ...)								
	1020(H)	2020(H)	2040(H)	3020	3040	4020	4040	5040(C)	6040(C)
internal memory size (KB)	4598	9048	9048	13558	13558	18678	18678	29174	65535(1)
(1) The sum of saved data, unsaved data, and program data is limited to 65535 KB.									

Program and Data Memory (Hot Standby). This table shows the program and data memory capacity for M580 Hot Standby controllers:

Memory Size	Reference (BMEH58 ...)		
	2040(C)	4040(C)	6040(C)
internal memory size (KB)	9462	18934	65536(1)
(1) The sum of saved data, unsaved data, and program data is limited to 65536 KB.			

Memory Areas (Standalone). This table shows the maximum memory size per area for M580 standalone controllers:

Maximum Memory Size	Reference (BMEP58 ...)								
	1020 (H)	2020 (H)	2040 (H)	3020	3040	4020	4040	5040 (C)	6040(C)
saved data (KB)(1)	384	768	768	1024	1024	2048	2048	4096	4096
program (KB)	4096	8162	8162	12288	12288	16384	16384	24576	65536(2)
(1) 10 KB are reserved for the system **(2)** The sum of saved data, unsaved data, and program data is limited to 65536 KB.									

Memory Areas (Hot Standby). This table shows the maximum memory size per area for M580 Hot Standby controllers:

Maximum Memory Size	Reference (BMEH58 ...)		
	2040(C)	4040(C)	6040(C)
saved data (KB)(1)	768	2048	4096
Hot Standby data exchanged (KB)	768	2048	4096
program (KB)	8162	16384	65536(2)
(1) 10 KB are reserved for the system **(2)** The sum of saved data, unsaved data, and program data is limited to 65536 KB.			

NOTE: Versions 2.30 and any subsequent supporting version(s) of M580 processor firmware provide a maximum of 64 K words of memory for State RAM. By contrast, the display for firmware versions 2.20 and earlier would appear to provide a maximum of 128 K words; however, the display is incorrect. As a result, if you upgrade controller firmware from version 2.20 or earlier to version 2.30 or any subsequent supporting version(s) for an existing project, the percentage of State RAM used by the application will appear to have doubled. In some cases, the percentage of State RAM used can exceed 100% and the application cannot be re-built. To re-build your application in this case, you will need to perform one or both of the following edits:

- Increase the amount of State RAM (the total of %M, %MW, %I, %IW), if possible.
- Re-define some located variables as unlocated (by removing the assigned address), until the total amount of State RAM used (the sum of %M, %MW, %I, % IW) no longer exceeds 100%.

Located Data (Standalone). This table shows the maximum and default size of located data (in KB) for each M580 standalone controller:

Object Types	Address	Reference (BMEP58 ...)								
		1020 (H)	2020 (H)	2040 (H)	3020	3040	4020	4040	5040 (C)	6040 (C)
internal bits	%Mi maximum	32634	32634	32634	32634	32634	32634	65280 (2)	65280 (2)	65280 (2)
	%Mi default	512	512	512	512	512	512	512	512	512
input/ output bits	%Ir.m.c %Qr.m.c	(1)	(1)	(1)	(1)	(1)	(1)	(1)	(1)	(1)
system bits	%Si	128	128	128	128	128	128	128	128	128
internal words	%MWi maximum	32464	32464	32464	65232	65232	65232	64896 (3)	64896 (3)	64896 (3)
	%MWi default	1024	1024	1024	2048	2048	2048	2048	2048	2048

(1) Memory size depends on the equipment configuration declared (I/O modules).
(2) 32624 for versions before 2.30.
(3) 65232 for versions before 2.30.

Located Data (Hot Standby). This table shows the maximum and default size of located data (in KB) for each M580 Hot Standby controller:

Object Types	Address	Reference (BMEH58 ...)		
		2040(C)	4040(C)	6040(C)
internal bits	%Mi maximum	32634	65280(2)	65280(2)
	%Mi default	512	512	512

Object Types	Address	Reference (BMEH58 ...)		
		2040(C)	4040(C)	6040(C)
input/output bits	%Ir.m.c %Qr.m.c	(1)	(1)	(1)
system bits	%Si	128	128	128
internal words	%MWi maximum	32464	64896(3)	64896(3)
	%MWi default	1024	1024	2048

(1) Memory size depends on the equipment configuration declared (I/O modules).
(2) 32624 for versions before 2.30.
(3) 65232 for versions before 2.30.

Size of Unlocated Data Memory

This list contains unlocated data types:

- elementary data type (EDT)
- derived data type (DDT)
- derived function block (DFB) and elementary function block (EFB)

The size limit of unlocated data is the global maximum memory size for data, page 29 minus the size consumed by located data.

Client and Server Requests per Scan

The communication performance of standalone (BMEP58•0•0) and Hot Standby (BMEH58•0•0) controllers is described in terms of the number of client and server requests per scan.

Modbus TCP and EtherNet/IP Server: The table below shows the maximum number of Modbus TCP, EtherNet/IP, or UMAS requests that can be served by the controller Modbus TCP server at each MAST scan.

When the incoming requests exceed these maximums, they are queued in a first-in/first out (FIFO) buffer. The size of the FIFO buffer is according to the selected controller:

Controller	Overall maximum		From USB	Maximum requests sent to IP address of the controller	Maximum requests sent to IP address of comm. modules
	Requests per Scan(1)	Request FIFO Size			
BMEP581020	8 (16)	32	4	8	16
BME•5820•0	16 (24)	32	4	12	16

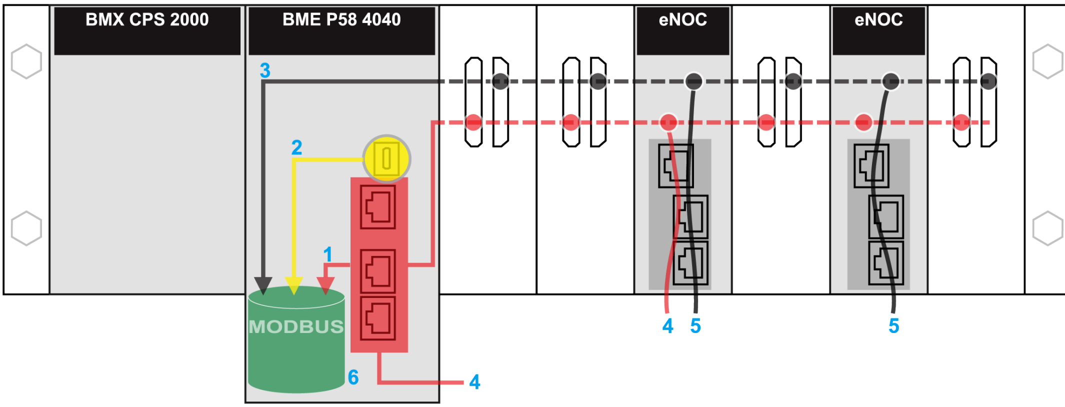
Controller	Overall maximum		From USB	Maximum requests sent to IP address of the controller	Maximum requests sent to IP address of comm. modules
	Requests per Scan(1)	Request FIFO Size			
BMEP5830•0	24 (32)	32	4	16	16
BME•5840•0	32 (40)	50	4	24	16
BMEP5850•0	40 (48)	50	4	32	16
BME•5860•0	56 (64)(2)	50	4	32	16

(1) This column shows the default limits for the number of requests served per cycle. The limit can be modified through %SW90, between 2 and the number indicated between brackets.

(2) The overall limit for the BME•5860•0 controller is higher than the sum of the limits for the USB, controller, and NOC modules. This is a provision for future evolutions.

The MAST task cycle time may increase by up to 0.5 ms per incoming request. When the communications load is high, you can limit the potential jitter of the MAST time by limiting the number of requests that are processed per cycle in %SW90.

Example: This example local backplane assembly includes a BMEP584040 controller and two BMENOC0301/BMENOC0311 Ethernet communication modules. Therefore, the maximum values in this example apply to the BMEP584040 controller (described above):



red: These requests are sent to the IP address of the controller.

yellow: These requests are from the USB port of the controller.

gray: These requests are sent to the IP address of a communications module (NOC).

1 The maximum number of requests to the IP address of the BMEP584040 controller (24).

2 The maximum number of requests from the USB port of the controller (4). (For example, a PC that runs Control Expert may be connected to the USB port.)

3 The maximum number of requests from all communications modules on the local backplane (16).

4 These requests are sent to the IP address of the BMEP584040 controller from devices that are connected to an Ethernet port on either the controller or a BMENOC0301/ BMENOC0311 module.

5 These requests are sent to the IP address of the BMENOC0301/BMENOC0311 from devices that are connected on the Ethernet port of either the BMENOC0301/BMENOC0311 or the controller. (In this case, enable the Ethernet backplane port of the BMENOC0301/ BMENOC0311.)

6 The Modbus server can manage in each request the maximum number of requests from the BMEP584040 controller (32). It also holds a maximum of 50 requests in a FIFO buffer.

Number of Connections: This table shows the maximum number of simultaneous Modbus TCP, EtherNet/IP, and UMAS connections for the embedded Ethernet port on these controllers:

Controller	Connections
BMEP581020	32
BME•5820•0	32
BMEP5830•0	48
BME•5840•0	64
BMEP5850•0	64
BME•5860•0	80

When an incoming connection request is accepted, the open connection that has been idle for the longest time is closed.

Modbus TCP and EtherNet/IP Client: This table shows the maximum number (per cycle) of communication EFs that support Modbus TCP and EtherNet/IP clients according to the selected controller:

Controller	EFs per Cycle
BMEP581020	16
BME•5820•0	32
BMEP5830•0	48
BME•5840•0	80
BMEP5850•0	80
BME•5860•0	96

OPC UA Performance

Each M580 PAC can support:

- Up to 64 connection in parallel using the UA_Connect function block.
- For each connection:
  - Up to 256 nodes (simple type) to read.
  - Up to 128 nodes (simple type) to write.

The following table presents the limits on the number of connections (sessions) and subscriptions supported by each M580 PAC:

Controller	Maximum Connections (Sessions)	Maximum Subscriptions
BMEP5810•0	4	8
BMEP5820•0	8	16

Controller	Maximum Connections (Sessions)	Maximum Subscriptions
BMEP5830•0	16	32
BMEP5840•0	32	64
BMEP5850•0	48	96
BMEP5860•0	64	128
BMEH5820•0	32	64
BMEH5840•0	48	96
BMEH5860•0	64	128

If these limits are exceeded, the OPC UA client detects the following errors:

- E_MaxConnectionsReached (ID 16#B000_0509) in the UA_Connect function block, and
- E_MaxSubscriptionsReached (ID 16#B000_0501) in the UA_SuscriptionCreate function block.

Application Code Execution Performance

This table shows the performance of the application code for each M580 standalone (BMEP58 ...) and Hot Standby (BMEH58...) controller:

	Reference BMEP58 .../BMEH58 ...								
	1020 (H)	2020 (H)	2040 (H)	3020	3040	4020	4040 (C)	5040 (C)	6040 (C)
boolean application execution (Kinst/ms(1))	10	10	10	20	20	40	40	50	50
typical execution (Kinst/ms (1))	7.5	7.5	7.5	15	15	30	30	40	40
(1) • Kist/ms: 1,024 instructions per millisecond • A typical execution holds 65% boolean instructions + 35% fixed arithmetic.									

Standards and Certifications

Download

Click the link that corresponds to your preferred language to download standards and certifications (PDF format) that apply to the modules in this product line:

Title	**Languages**
Modicon M580, M340, and X80 I/O Platforms, Standards and Certifications	• English: EIO0000002726 • French: EIO0000002727 • German: EIO0000002728 • Italian: EIO0000002730 • Spanish: EIO0000002729 • Chinese: EIO0000002731

States for M580 PACs

Introduction

This topic describes the operating states for M580 standalone and Hot Standby controllers.

Operating States for Standalone Controllers

All standalone M580 PACs have these operating states:

Operating State	Description
AUTOTEST	The controller is executing its internal self-tests. **NOTE:** If extended backplanes are connected to the main local backplane and line terminators are not inserted into the unused connectors on the backplane extender module, the controller remains in **AUTOTEST** after the self-tests have completed.
NOCONF	The application program is not valid.
STOP	The controller has a valid application, but it is stopped. The controller sets itself to predefined STOP state parameters, and can be restarted later.
HALT	The controller has an application, but it has stopped operating due to an error resulting in a blocking condition, which puts the controller in a HALT state, resulting in a recoverable, page 99 or nonrecoverable condition, page 96.
RUN	The controller is executing the application program.
WAIT	The controller is in a transitory state while it backs up data when a power down condition is detected. The controller starts again only when power is restored and the supply reserve is replenished. As it is a transitory state, it may not be viewed. The controller performs a warm restart, page 523 to exit the WAIT state.
ERROR	The controller is stopped because a hardware or system error is detected. When the system is ready to be restarted, the controller performs a cold start, page 520 to exit the ERROR state.
OS DOWNLOAD	A controller firmware download is in progress.

Monitoring the Controller Operating State

The LEDs on the controller front panel provide indications of its operating state, page 63.

Hot Standby System States

Controller State Versus Hot Standby System State

The state of the Hot Standby system depends on the operating state of the controller. These Hot Standby states are supported:

Controller Operating State	Hot Standby System State
INIT	INIT
STOP	STOP
RUN	PRIMARY with standby counterpart
	PRIMARY without standby counterpart
	STANDBY
	WAIT

This list describes the Hot Standby states:

- **Primary**: The controller controls the system processes and devices:
  - It executes program logic in a non-safety-related controller, and both process and safety-related program logic in a safety controller.
  - It receives input from, and controls output to, distributed equipment and RIO drops.
  - If connected to a controller in standby state, the primary controller verifies the status of, and exchanges data with, the standby controller.

  In a Hot Standby network, both controllers can be primary if both the Hot Standby and Ethernet RIO links are not functioning. When either of these two links is restored, the controller does one of the following:
  - Remains in the primary state.
  - Transitions to the standby state.
  - Transitions to the wait state.

- **Standby**: The standby controller maintains a state of readiness. It can take control of system processes and devices if the primary controller cannot continue to perform these functions:
  - It reads the data and the I/O states from the primary controller.
  - It does not scan distributed equipment, but receives this information from the primary controller.
  - It executes program logic. You can configure the standby controller to execute:

    - The first section of program logic (the default setting); or

    - Specified sections of program logic, including all MAST and FAST task sections.

      NOTE: You can specify if a section is to be executed in the **Condition** tab of the **Properties** dialog box for each section.
  - On each scan, it verifies the status of the primary controller.

  NOTE: When a controller is in Standby mode, both the module health status (MOD_HEALTH) and the channels health status (CH_HEALTH) of safety I/O modules are set to FALSE in the Standby controller DDDT. In this case, you can diagnose the health of safety I/O modules by monitoring their status in the Primary controller DDDT.
- **Wait**: The controller is in RUN mode, but cannot act as either primary or standby. The controller transitions from the wait state to either the primary or standby state, when the preconditions for that state exist, including:
  - The state of the Hot Standby link.
  - The state of the Ethernet RIO link.
  - The presence of at least one connection with an Ethernet RIO drop.
  - The position of the A/B rotary selection switch on the rear of the .
  - The state of the configuration. For example:

    - If a firmware mismatch exists, the `FW_MISMATCH_ALLOWED` flag is set.

    - If a logic mismatch exists, the `LOGIC_MISMATCH_ALLOWED` flag is set.

  In the wait state, the controller continues to communicate with other modules on the local backplane, and can execute program logic, if configured to do so. You can configure a controller in wait state to execute:
  - Specific sections of program logic in a non-safety-related controller (or process program logic in a safety controller), specified in the **Condition** tab of the **Properties** dialog box for each section.
  - The first section of program logic in a non-safety-related controller (or the first section of process program logic in a safety controller).
  - No program logic for a non-safety-related controller (or no process program logic for a safety controller).
- **INIT**: Both the controller and the Hot Standby system are initializing.

- **STOP**: The controller is in STOP mode. On the STOP to RUN transition, the controller may move to the wait, standby, or primary state. This transition depends on the state of the Ethernet RIO and Hot Standby links, and on the position of the A/B rotary selection switch on the rear of the controller.

NOTE: In addition to the controller operating states listed here, other operating states that are not related to the Hot Standby system, page 38 exist.

Controller Functions by Hot Standby System State

A controller performs these functions, depending on its Hot Standby state:

<table>
<tr><th rowspan="2">Controller functions</th><th colspan="3">Hot Standby system states</th></tr>
<tr><th>Primary</th><th>Standby</th><th>Wait</th></tr>
<tr><td>RIO drops</td><td>YES</td><td>NO</td><td>NO</td></tr>
<tr><td>Distributed equipment</td><td>YES</td><td>NO</td><td>NO</td></tr>
<tr><td>Execution of program logic (non-safety-related controller) or process task logic (safety controller)</td><td>YES</td><td>Depending on configuration, STANDBY controller can execute:
• First section (default)
• Specified sections (which can include all MAST and FAST sections)
• None</td><td>Depending on configuration, WAIT controller can execute:
• First section (default)
• Specified sections (which can include all MAST and FAST sections)
• None</td></tr>
<tr><td>Execution of safe logic (safety controller)</td><td>YES</td><td>NO</td><td>NO</td></tr>
<tr><td>Program Data Exchange (non-safety-related controller) or Process Data Exchange (safety controller)</td><td>YES</td><td>YES</td><td>NO</td></tr>
<tr><td>Safe Data Exchange (safety controller)</td><td>YES</td><td>YES</td><td>NO</td></tr>
<tr><td colspan="4">1. Data exchange is controlled by the Exchange on STBY attribute.</td></tr>
</table>

Controller Switchover in an M580 Hot Standby System

Introduction

The purpose of a Hot Standby system is to be ready to perform a switchover, if needed. A switchover is the immediate transfer of control of the network from the primary controller to the standby controller. The transfer needs to be swift and seamless.

The M580 Hot Standby system continuously monitors ongoing system operations, and determines if a condition requiring a switchover exists. On each scan, both the primary controller and the standby controller verify the health of the system.

The primary controller verifies the health of the following:

- the Ethernet RIO network link
- the Hot Standby link between the primary and standby controllers

The standby controller verifies the following:

- the health of the primary controller
- the identity of modules in both the primary and standby backplanes
- application versions running in the primary and standby controllers
- firmware versions of the primary and standby controllers
- the health of the Hot Standby link between the primary and standby controllers

Before each MAST task, the primary controller transfers to the standby controller system, status and I/O data, page 493, including date and time data. On switchover, the standby controller applies this time data and continues the same time stamping sequence. The maximum amount of transferable Hot Standby data depends on the controller.

> **NOTE:** Both the primary controller and the standby controller maintain independent event logs. If a switchover occurs, the events recorded in the log of the former primary controller will not be included in the event log of the new primary (formerly the standby) controller.

Switchover Causes

Any one of the following events will cause a switchover:

- The primary controller has encountered a blocking condition (see Modicon M580, Hardware, Reference Manual) and entered the HALT state.
- The primary controller has detected an unrecoverable hardware or system error.
- The primary controller has received a STOP command from Control Expert or the DDDT.
- An application program is being transferred to the primary controller.

- Primary controller power is turned off; a power cycle occurs.
- The following events simultaneously occur:
  - The primary controller loses communication to all RIO drops.
  - The Hot Standby link is healthy.
  - The standby controller maintains communication with at least one RIO drop.

Similar to a switchover, a swap is a controlled event that transfers control of the network from the primary controller to the standby controller. A swap can be caused by:

- Execution of the DDDT `CMD_SWAP` command by either program logic, or an animation table **Force** command.
- Manually clicking the **HSBY Swap** button in the **Task** tab of the controller **Animation** window in Control Expert.

Events that Do Not Cause Switchover

These events **DO NOT** cause a switchover:

- simultaneous interruption of communication with all RIO drops by both the primary and the standby controller
- partial interruption of communication with the RIO drops by the primary controller
- a Modbus connection break
- overload broadcast traffic generated by a peer (for example, SCADA, or another controller)
- a BMENOC0301/BMENOC0311 module that stops operating
- removal of an SD memory card, page 78
- for a Hot Standby safety system, if the primary controller is partially (either the SAFE program or the PROCESS program) in the HALT state, and not all of the tasks in the standby controller are in RUN

Switchover Execution Time

If both the primary controller and standby controller are operating normally, the Hot Standby system detects a switchover causal event within 15 ms.

For both a safety and non-safety-related controller system, the effect of the switchover on the application reaction time is:

- 15 ms for the I/O driven by the MAST task.
- 15 ms + T_{TASK} for the I/O driven by the FAST or the SAFE task, where T_{TASK} is the configured execution period for that task.

The application response time for a swap or a switchover can be calculated.

After the switchover, the former standby controller becomes the primary. In the worst case, the new primary controller operates with data of scan cycle N, while the outputs have received (from the former primary controller) data of scan cycle N+1. The new primary controller re-evaluates outputs beginning with scan N+1. As the Hot Standby switchover evaluation occurs during the MAST task, some FAST task program execution may be skipped.

Switchover Effect on Main IP Address Assignments

Distributed equipment uses the **Main IP address** setting, configured in the **IPConfig** tab, page 479, to communicate over an Ethernet network with the primary controller. On switchover, the **Main IP address** setting is automatically transferred from the former primary controller to the former standby – now the new primary – controller. Similarly, on switchover the **Main IP address + 1** setting is automatically transferred from the former standby controller to the new standby.

In this way, the configured links between the distributed equipment and the primary controller do not need to be edited in the event of a switchover.

NOTE:

- A switchover does not affect the assignment of **IP address A** or **IP address B**. These assignments are made exclusively by means of the A/B/Clear rotary switch on the back of the controller, and are not affected by a change in primary or standby Hot Standby status.
- When connecting Control Expert to the Hot Standby system, use **IP address A** or **IP address B** to maintain the connection on a switchover. Avoid using the **Main IP address**, because on switchover this becomes **Main IP address + 1** and will disconnect Control Expert.

Switchover Effect on Remote Outputs

For RIO drops, the switchover is transparent: the state of outputs is not affected by the switchover. During Hot Standby operations, each controller maintains an independent, redundant owner connection with each RIO drop. Each controller makes this connection via **IP address A** or **IP address B**, depending on the A/B/Clear rotary switch designation for its controller. When a switchover occurs, the new primary controller continues to communicate with I/O via its pre-existing redundant owner connection.

NOTE: The switchover may not be transparent with respect to distributed equipment outputs.

Switchover Effect on Communication Module State

In a high availability (Hot Standby) configuration that includes BMENOC0301/BMENOC0311/BMENOC0321(C) communication modules, set the **Watch Dog** of the appropriate task (MAST or FAST) to a value equal to or grater than the default setting of 250 ms. Smaller **Watch Dog** values may cause the communication modules to timeout and enter a non-configured (NOCONF) state.

Switchover Effect on Distributed Equipment Outputs

The behavior of distributed equipment outputs during a switchover depends on whether the equipment supports hold up time. If the device does not support hold up time, its outputs will most likely go to fallback when the connection with the primary controller is interrupted, and will recover their state after reconnecting with the new primary controller.

To achieve transparent behavior, the outputs need to support a sufficiently long hold up time, page 484.

Switchover Effect on CCOTF Changes

After the standby controller becomes the new primary, it operates using both the firmware and the application previously configured in it. If CCOTF, page 475 changes were previously made to the former primary controller that were not transferred to the former standby controller, these changes are not included in the configuration running in the new primary controller.

For example, assume that an I/O module was added to a remote I/O drop in the configuration running in the former primary controller. If the changed configuration was not transferred to the former standby controller, the added module will not be included in the configuration running in the former standby controller when it becomes the primary controller after switchover.

Switchover Effect on Program Logic Changes

A logic mismatch condition exists when changes have been made to the application in the primary controller, but not to the standby controller. If the `LOGIC_MISMATCH_ALLOWED`, page 498 flag is set, the standby controller can continue to operate as standby while a logic mismatch exists. In this case, if a switchover occurs, the new primary controller executes its own, different application using data received from the former primary controller.

Depending on the nature of the application modification, different results occur:

Modification to initial primary controller logic:	Effect on new primary controller program execution:
Only code is changed (no changes to variables).	All variable values exchanged between the controllers remain the same (EQUAL).
New variables were added.	The new variables are not used by the new primary controller.
Existing variables were deleted.	The new primary controller includes the deleted variables in program execution, and applies the most recent values to these variables.

Switchover Effects on Time Management

In an M580 Hot Standby system, the primary controller and the standby controller operate their own system timers, which are not automatically synchronized. Because both the primary controller and the standby controller share a common configuration, both can be configured to perform as NTP client or NTP server.

When the NTP client function is enabled in a Hot Standby system, the primary controller and the standby controller independently receive time settings from a designated NTP server.

When the NTP server is enabled in a Hot Standby system, only the primary controllers performs the role of server.

Before each scan, the primary controller transfers system data to the standby controller, including the following primary controller system time values:

- time of day
- application counters
- free running counter

On switchover, the former standby controller – now the new primary controller – applies the system time values sent by the former primary controller. Thereafter, the new primary controller continues to execute the application in the same time context as the former primary controller. If the NTP server function is enabled for the Hot Standby system, the new primary controller begins to perform the function of NTP server.

Switchover Effects on IPsec Connections

On switchover, the former primary BMENOC0301/BMENOC0311 module closes all connections that use its main IP address. These connections are re-opened on the new primary BMENOC0301/BMENOC0311 module using the main IP address after the two modules swap their main and main+1 IP addresses. As IPsec connections take a relatively long time to establish, it can take up to 5 minutes to re-establish an IPSEC connection that uses the main IP address.

Switchover Effect on Safety Operating Mode

When an M580 safety Hot Standby controller switches from standby controller to primary controller, the operating mode is automatically set to safety mode.

NOTE: The operating mode setting of a safety Hot Standby controller – either safety mode or maintenance mode – is not included in the transfer of an application from the primary controller to the standby controller.

Recovery of Former Primary Controller

The former primary controller may or may not become the standby controller, depending on cause of switchover.

<table>
<tr><th>If the switchover was caused by:</th><th>Make the former primary controller the standby by:</th></tr>
<tr><td>Primary halt (non-safety-related controller)</td><td>performing an <code>INIT</code> command and RUN the controller</td></tr>
<tr><td>Primary halt (safety controller - Process and/or SAFE task)</td><td>performing an <code>INIT</code> command (Process task) and/or an <code>INIT_SAFETY</code> command (SAFE task), and then RUN the controller</td></tr>
<tr><td>controller stop in a non-safety-related controller, or in both the Process and SAFE tasks of a safety controller</td><td>running the controller</td></tr>
<tr><td>Primary error detected</td><td>performing a controller <code>RESET</code> command</td></tr>
<tr><td>Application transfer on Primary</td><td>completing the transfer and RUN the application</td></tr>
<tr><td>Primary power off</td><td>powering up the controller</td></tr>
<tr><td>Loss of all RIO drops (if any) while HSBY link is still healthy and Standby controller has access to the drops</td><td>causing the controller to recover RIO drops</td></tr>
<tr><td>DDDT command</td><td rowspan="2">The former primary automatically becomes the standby, provided the necessary preconditions exist, for example:
Firmware mismatch is allowed, if a firmware mismatch exists.
Logic mismatch is allowed, if a logic mismatch exists.
Online modifications are allowed, if modifications have been made.</td></tr>
<tr><td>Control Expert HSBY Swap button</td></tr>
</table>

Electrical Characteristics

Introduction

The power supply module provides current to the modules installed on the local rack, including the controller. The controller current consumption contributes to the total rack consumption.

Controller Power Consumption

Typical controller consumption with a 24 Vdc power supply:

Controller	Typical Consumption
BMEP581020(H)	270 mA
BMEP5820•0(H)	270 mA
BMEP5830•0	295 mA
BMEP5840•0	295 mA
BMEP585040(C)(C)	300 mA
BMEP586040(C)	300 mA
BMEH582040(C)	335 mA (with a copper SFP)
BMEH584040(C)	360 mA (with a copper SFP)
BMEH586040(C)(C)	365 mA (with a copper SFP)

Mean Time Between Failures (MBTF)

For all controllers, the MTBF (measured at 30 °C continuous) is 600,000 hours.

Real-Time Clock

Introduction

Your controller has a real-time clock that:

- provides the date and time
- displays the date and time of the last application shut-down

Clock Accuracy

The resolution of the real-time clock is 1 ms. The clock accuracy is affected by the operating temperature of the application:

Operating Temperature	Maximum Daily Drift (Seconds/Day)	Maximum Yearly Drift (Minutes/Year)
25 °C (77 °F) stabilized	+/- 2.6	+/- 17.4
0...60 °C (32...140 °F)	+/- 5.2	+/-33.1

Clock Back-Up

The accuracy of the real-time clock is maintained for four weeks when the controller power is turned off if the temperature is below 45 °C (113 °F). If the temperature is higher, the back-up time is shorter. The real-time clock back-up does not need any maintenance.

If the back-up power is too low, system bit *%S51* is set to 1. This value indicates a loss of time when the power supply was OFF.

Date and Time

The controller updates the date and time in the system words *%SW49–%SW53* and *%SW70*. This data is in BCD.

> **NOTE:** For **M580** controllers, the time is in universal coordinated time (UTC). If local time is needed, use the `RRTC_DT` function.

Accessing the Date and Time

You can access the date and time:

- on the controller debug screen
- in the program
- from the DTM diagnostics screen

To read the date and time, read system words *%SW49* through *%SW53*. This operation sets system bit *%S50* to 0.

To write the date and time, write system words *%SW50* through *%SW53*. This operation sets system bit *%S50* to 1.

When system bit *%S59* is set to 1, you can increment or decrement the date and time values with system word *%SW59*.

The function performed by each bit in word *%SW59* is:

Bit	Function
0	increments the day of the week
1	increments the seconds
2	increments the minutes
3	increments the hours
4	increments the days
5	increments the months
6	increments the years
7	increments the centuries
8	decrements the day of the week
9	decrements the seconds
10	decrements the minutes
11	decrements the hours
12	decrements the days
13	decrements the months
14	decrements the years
15	decrements the centuries

NOTE: The preceeding functions are performed when system bit *%S59* is set to 1.

Determining the Date and Time of the Last Application Shutdown

The local date and time of the last application shutdown are displayed in system words *%SW54* through *%SW58*. They are displayed in BCD.

System Word	Most Significant Byte	Least Significant Byte
%SW54	seconds (0 to 59)	00
%SW55	hours (0 to 23)	minutes (0 to 59)
%SW56	month (1 to 12)	day in the month (1 to 31)
%SW57	century (0 to 99)	year (0 to 99)
%SW58	day of the week (1 to 7)	reason for the last application shutdown

The reason for the last application shutdown can be displayed by reading the least significant byte of system word *%SW58*, which can have these values (in BCD):

Word*%SW58* Value	Definition
1	application switched to STOP mode
2	application stopped by watchdog
4	power interruption
5	stop on detected hardware error
6	stop when errors such as these are detected: • software error (HALT instruction) • SFC error • application CRC checksum error • undefined system function call Details on the detected error are stored in *%SW125*.

Addressing Field Buses

Addressing Field Buses

The following field buses can be addressed by either configuring the appropriate protocol or using dedicated modules and devices.

Field Bus	Addressing Method
AS-i	AS-Interface bus is addressed with a Modicon X80 BMXEIA0100 module.
HART	HART communication protocol can be addressed using either the eX80 HART modules: • BMEAHI0812 HART analog input module • BMEAHO0412 HART analog output module **or** • a Modicon STB island with an STBNIP2311 EtherNet/IP network interface modue and an STBAHI8321 HART interface module.
Modbus TCP, EtherNet/IP	Modbus TCP devices are connected to the Ethernet DIO network.
Modbus Plus	Modbus Plus is supported using a gateway module like TCSEGDB23F24FA or TCSEGDB23F24FK.
PROFIBUS-DP	A PROFIBUS remote master is connected to the Ethernet DIO network. The process variables are exchanged via the DIO scanner service in the CPU. PROFIBUS gateway modules: TCSEGPA23F14F or TCSEGPA23F14FK
PROFIBUS-PA	A PROFIBUS remote master and a DP/PA interface are connected to an Ethernet DIO network. The process variables are exchanged via the DIO scanner service in the controller. PROFIBUS gateway modules: TCSEGPA23F14F or TCSEGPA23F14FK

BMEP58•••• Controller Physical Characteristics

Introduction

This section describes the physical elements that are displayed on the front panel of the Modicon M580 controllers. The various communication ports, LED diagnostic information, and several options available for industrial hardening and memory back-up are detailed.

Physical Description of Standalone Controllers

Position on the Local Rack

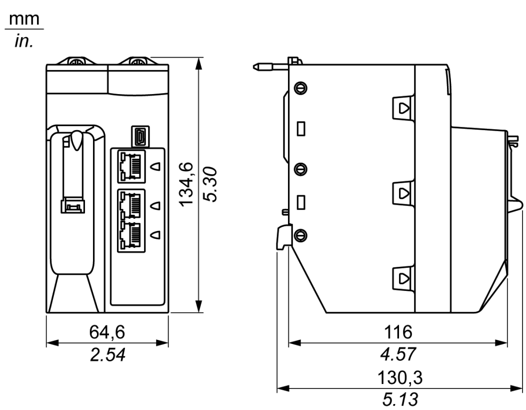
Every M580 standalone system requires one controller. The controller is installed in the two-module slot position directly to the right of the power supply in the main local rack. The controller cannot be put in any other slot location or any other rack. If there are extended racks in the local rack configuration, assign address 00 to the rack with the controller.

NOTE:

1. Refer to the list of M580 standalone controllers, page 25.
2. When the last two octets of the MAC address (*MAC5.MAC6*) correspond to *0.0* in the default address, make a point-to-point cable connection between your computer and the controller, communication module, or other module.

Dimensions

This graphic shows the front and side dimensions of the M580 standalone controllers:



NOTE: Consider the height of the controller when you are planning the installation of the local rack. The controller extends below the lower edge of the rack by:

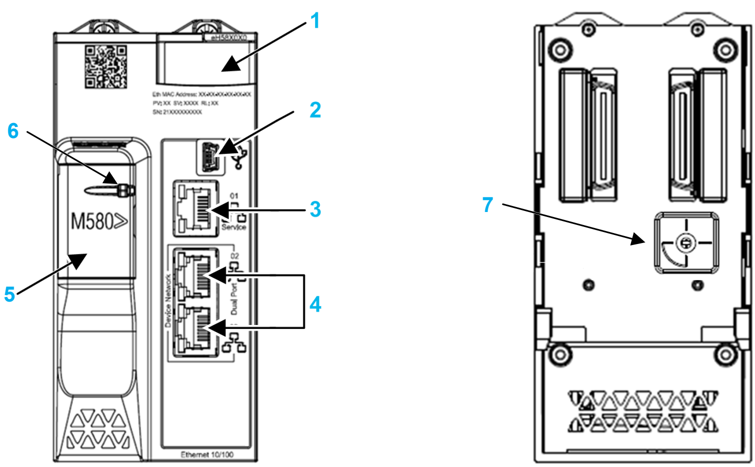
- 29.49 mm (1.161 in.) for an Ethernet rack
- 30.9 mm (1.217 in.) for an X Bus rack

Front and Rear Views

Standalone controllers have similar front panels. Depending on the standalone controller you choose, these differences apply:

- BMEP58•020: The embedded Ethernet I/O scanner service supports DIO only.
- BMEP58•040: The embedded Ethernet I/O scanner service supports both RIO and DIO.

Physical features:



1 LED diagnostic display panel for controller status and diagnostics

2 Mini-B USB port for module configuration via PC running Control Expert

3 RJ45 Ethernet service port connector

4 RJ45 connectors that together serve as a dual port to the Ethernet network

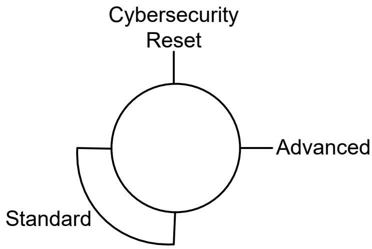
5 SD memory card slot (behind door)

6 SD memory card lockable door, page 61

7 Cybersecurity rotary selector switch, page 54

Cybersecurity Rotary Selector Switch

Use the rotary switch on the back of each M580 PAC to configure a cybersecurity operating mode for the module:



Switch positions are:

- **Standard**: the module supports basic cybersecurity features.
- **Advanced**: the module supports advanced cybersecurity features.
- **Reset**: the module returns to its out-of-the box cybersecurity setting.

The out-of-the-box switch position is **Standard**.

NOTE: The operating mode rotary selector switch will be made operational for future product releases. For this product release, operating mode is automatically set to **Standard**, regardless of the switch position.

Physical Description of Hot Standby Controllers

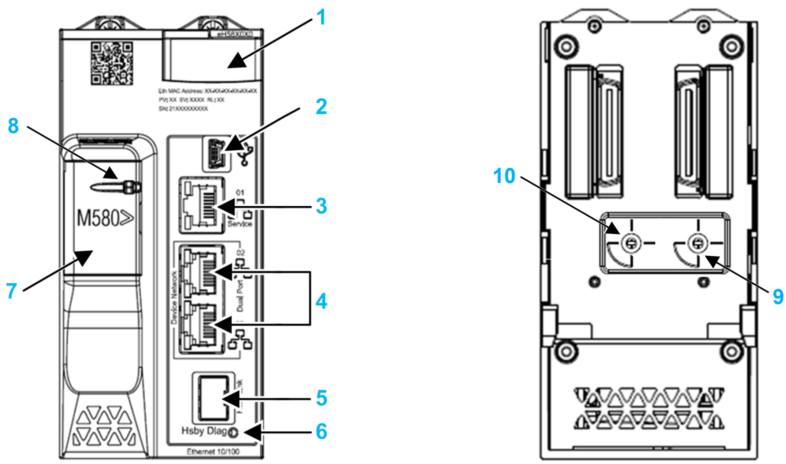
Hot Standby Controllers

These controllers support M580 Hot Standby systems:

- BMEH582040, BMEH582040C, BMEH582040S
- BMEH584040, BMEH584040C, BMEH584040S
- BMEH586040(C),BMEH586040(C)C, BMEH586040S

Controller Module Front and Back Views

The three Hot Standby controller modules have the same external hardware features. The front of the module is on the left. The back of the module is on the right:



1 LED diagnostic display panel

2 Mini-B USB port for module configuration via PC running Control Expert

3 RJ45 Ethernet service port connector

4 RJ45 connectors that together serve as a dual port to the Ethernet network

5 SFP socket for copper or fiber-optic Hot Standby link connection

6 Hot Standby status link LED

7 SD memory card slot (behind door)

8 SD memory card lockable door, page 61

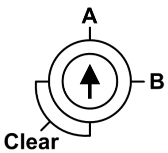
9 Cybersecurity rotary selector switch, page 54, with settings **Cybersecurity Reset**, **Advanced**, **Standard**

10 Hot Standby rotary selector, page 58, used to designate the controller as either controller **A** or controller **B**, or to **Clear** the existing Control Expert application

> **NOTE:** The only visible difference between safety and non-safety-related controllers is that safety controllers are colored red.

Hot Standby Rotary Selector Switch

Use the rotary switch on the back of each M580 Hot Standby controller to designate the role that the controller plays in the M580 Hot Standby configuration:



Use the small, plastic screwdriver provided with the controller to set the rotary switch according to its role in a Hot Standby system.

NOTE: A plastic screwdriver is provided for your convenience; use it, or an equivalent, to change the position of the rotary switch. Avoid using metal screwdrivers.

Rotary switch settings include:

Position	Result
A	• Designates the controller as controller A (see Modicon M580 Hot Standby, System Planning Guide for, Frequently Used Architectures), as referenced in Control Expert and the `T_M_ECPU_HSBY, page 304` DDDT. • Assigns the controller **IP address A** on Ethernet RIO network.
B	• Designates the controller as controller B (see Modicon M580 Hot Standby, System Planning Guide for, Frequently Used Architectures), as referenced in Control Expert and the `T_M_ECPU_HSBY` DDDT. • Assigns the controller **IP address B** on Ethernet RIO network.
Clear	• Clears the application in the controller, and places the controller into the `NO_CONF` operational state. • If an SD memory card is inserted in the controller, the application in the card is also cleared, **NOTE:** Setting the switch for each Hot Standby controller to the same A/B position can cause a a conflict of controller roles (see Modicon M580 Hot Standby, System Planning Guide for Frequently Used Architectures).

Clearing Controller Memory

To clear a controller memory, follow these steps:

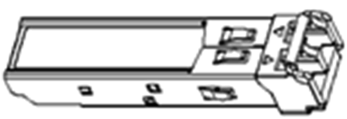
Step	Action
1	Set the rotary switch to **Clear**.
2	Power up the controller.

Step	Action
3	Power down the controller.
4	Set the rotary switch to **A** or **B**.

When you next power up the controller, if the remote controller is primary, the primary controller transfers the application to the local controller.

SFP Socket

Each controller module includes one Small Form-factor Pluggable (SFP) socket, to which you can connect either a fiber optic or a copper transceiver:



To insert a transceiver:

Step	Action
1	Check that the controller is powered off.
2	Position the transceiver so that its label is oriented to the left.
3	Press the SFP transceiver firmly into the socket until you feel it snap into place. **NOTE:** If the SFP transceiver resists, check the orientation of the transceiver and repeat these steps.

To remove a transceiver:

Step	Action
1	Check that the controller is powered off.
2	Pull out the latch to unlock the transceiver.
3	Pull on the transceiver to remove it.

NOTICE

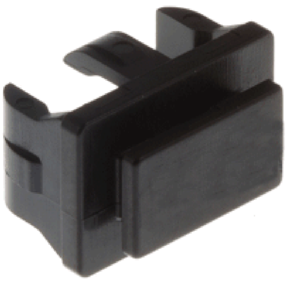
POTENTIAL EQUIPMENT DAMAGE

- Do not Hot Swap the SFP transceiver.
- Insert or remove the transceiver only when there is no power to the controller.

Failure to follow these instructions can result in equipment damage.

NOTE: For part numbers and other information regarding the available transceivers, refer to the description of controller Hot Standby link transceivers (see Modicon M580 Hot Standby, System Planning Guide for Frequently Used Architectures).

Each module comes with a stopper. When the SFP socket is not connected to a transceiver, cover the unused socket with the cover to keep out dust.



Grounding Considerations

Follow all local and national safety codes and standards.

DANGER

ELECTRIC SHOCK

Wear personal protective equipment (PPE) when working with shielded cables.

Failure to follow these instructions will result in death or serious injury.

The backplane for your M580 PAC is common with the functional ground (FE) plane and must be mounted and connected to a grounded, conductive backplane.

WARNING

UNINTENDED EQUIPMENT OPERATION

Connect the backplane to the functional ground (FE) of your installation.

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

Anti-Tampering Seals and Lockable SD Card Door

Anti-Tampering Seals

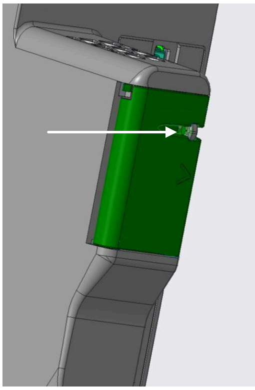
Two anti-tampering seals are placed on the right side of both the standalone and Hot Standby M580 CPUs, where the bezel (i.e. the front section of the module container) connects to the housing (i.e. the rear section of the module container). These seals indicates if the module has been opened and possibly tampered with.

The module container has not been opened when the anti-tampering seal looks like this:

The module container has been opened when the anti-tampering seal looks like this:

Lockable SD Card Door

The door that covers the SD card slot can be locked or sealed.



To do this:

1. Close the SD card door.
2. Insert the wire end of a lead seal (or the cable of a padlock) through the hole in the piece that protrudes through the SD card door.

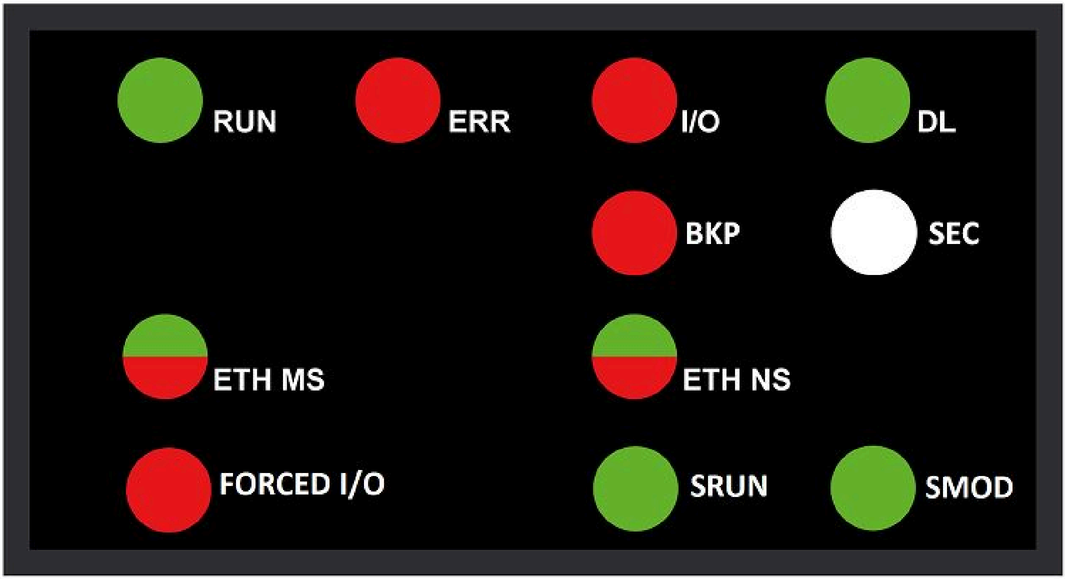
   NOTE: You can use a wire or cable with a maximum diameter of 1.50 mm.
3. Close the lead seal (or lock the padlock).

NOTE: The seal or padlock are not supplied with the module.

LED Diagnostics for Standalone Controllers

LED Display

An LED display is located on the front panel of the CPU:

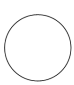
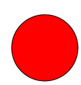
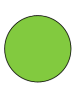
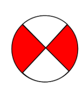
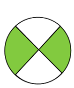
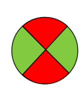


LED Descriptions

LED Indicator	Description
RUN	**ON**: The controller is in RUN state.
ERR	**ON**: The controller or system has detected an error.
I/O	**ON**: The controller or system has detected an error in one or more I/O modules.
DL (*download*)	• **Flashing**: Firmware update in progress. • **OFF**: No firmware update in progress.

LED Indicator	Description
BKP	**ON**: • The memory card or controller flash memory is missing or inoperable. • The memory card is not usable (incorrect format, page 78, unrecognized type). • The memory card or controller flash memory content is inconsistent with the application. • The memory card has been removed and reinserted. • A **PLC > Project Backup... > Backup Clear** command has been performed when no memory card was present. The **BKP** LED remains **ON** until the project is successfully backed up.
	OFF: The memory card or controller flash memory content is valid, and the application in the execution memory is identical.
SEC	Not used.
ETH MS	Module Status (green/red): Indicates the Ethernet port configuration status.
ETH NS	Network Status (green/red): Indicates the Ethernet connection status.
FORCED I/O	**ON**: At least one input or output on a digital I/O module is forced.
SRUN	Apply only to safety controllers.
SMOD	Apply only to safety controllers.

This table describes the LED indicator patterns used in the LED diagnostic indications table thereafter:

Symbol	Description	Symbol	Description
	off		steady red
	steady green		flashing red
	flashing green		flashing red/green

LED Diagnostic Indications

In a Hot Standby system, specific IP addresses (Main IP Address, Main IP Address + 1, IP Address A, IP Address B) are assigned (see Modicon M580 Hot Standby, System Planning

Guide for, Frequently Used Architectures) and these addresses must not be used by other devices in the system.

Duplicate IP addresses can cause errors in communication with the other modules.

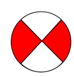
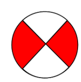
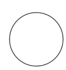
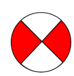
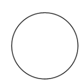
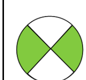
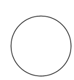
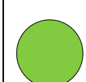
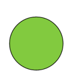
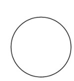
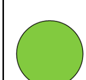
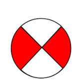
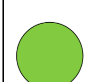
⚠ WARNING

UNINTENDED EQUIPMENT OPERATION

- Confirm that each module has a unique IP address.
- Do not assign an IP address equal to the Main IP Address, the Main IP Address + 1, IP Address A, or IP Address B to any Ethernet device that potentially communicates with the Hot Standby system.

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

The LEDs provide detailed diagnostic information when you observe their pattern in combination:

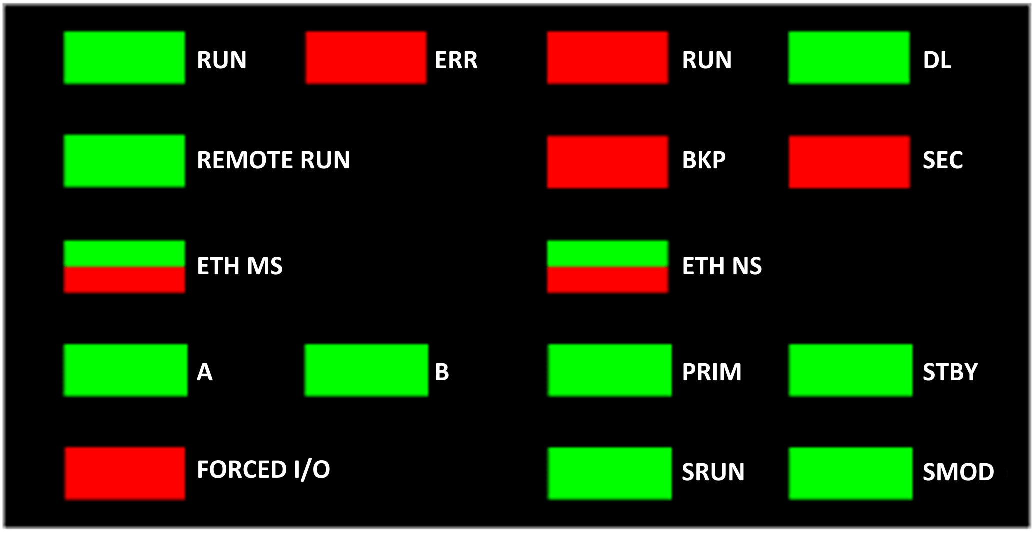
Condition	Controller State	RUN	ERR	I/O	ETH MS	ETH NS
power on	Autotest					
not configured (before getting a valid IP address or configuration is invalid)	NO-CONF					–
configured	Stop			• **off**: no error detected • **steady red**: error detected in a module or a channel		• **off**: invalid IP address • **flashing green**: valid IP address but no EtherNet/IP connection • **steady green**: EtherNet/IP connection established
	RUN					
recoverable error detected	HALT			–		• **flashing red**: At least one exclusive owner CIP connection (for which the BMENOC0301/ BMENOC0311 is the originator) is timed out. The LED flashes until the connection is reestablished or the module is reset.

Condition	Controller State	RUN	ERR	I/O	ETH MS	ETH NS
duplicate IP address	–	–	–	–		
unrecoverable error detected	–					
power off	–					
–: any pattern						

LED Diagnostics for Hot Standby Controllers

LED Panel

The front face of a BMEH58•040 Hot Standby controller presents the following LED panel, which you can use to diagnose the state of the M580 Hot Standby system:



NOTE: The **SRUN** and **SMOD** LEDs apply only to safety controllers. The **SEC** LED is not used.

- For a description of the safety controller LEDs **SRUN** and **SMOD**, refer to the topic *LED Displays for the M580 Safety Controler and Copro* (see Modicon M580, Safety System Planning Guide) in the *Modicon M580, Safety System Planning Guide*.
- For a presentation of LED diagnostics for safety-related controllers, refer to the topic *M580 Safety Controller LED Diagnostics* (see Modicon M580, Safety Manual) in the *Modicon M580, Safety Manual*.

Hot Standby Panel LEDs

Use the BMEH58•040 Hot Standby controller A and B LEDs to identify the controller configurations, as set by the rotary switch on each controller:

A/B/Clear Rotary Switch Position, page 58	LED	
	A	B
Local controller is A, remote controller is B	ON	OFF
Local controller is B, remote controller is A	OFF	ON
Both controller configured as A	Flashing	OFF
Both controller configured as B	OFF	Flashing
Local rotary switch on CLEAR	Flashing	Flashing

In the Hot Standby Panel LED diagnostic presentation, above:

- The local controller is the controller whose LEDs you are observing, which could be either A or B.
- The remote controller is the controller whose LEDs you are not observing, typically located in a remote location.

For example, consider the design where the two controllers are physically distant but communicate via a tunnel, with a controller located at each tunnel terminus. In this case, the local controller is the one in front of you; the remote controller is the one at the distant end of the tunnel. But, if you move to the other end of the tunnel, the formerly remote controller becomes the local controller and the original local controller becomes the remote controller. By contrast, the designations of controller A and controller B do not change.

Use the BMEH58•040 REMOTE RUN LED on the local controller to identify the operational status of the remote controller:

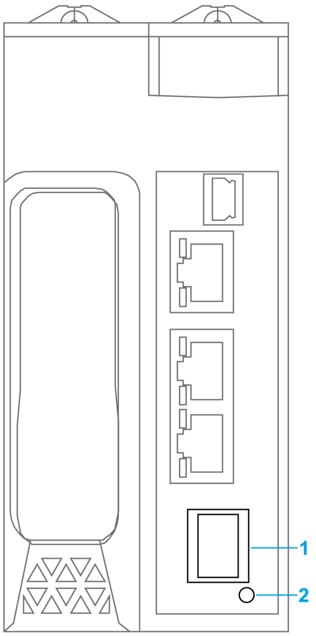
REMOTE RUN LED	Remote controller State
ON	RUN
Flashing	STOP
OFF	Indeterminate

Use the BMEH58•040 **PRIM**, and **STBY** LEDs to identify the operational status of the local and remote controller:

LED		Controller State	
PRIM	STBY	Local controller	Remote controller
ON	OFF	Primary	Standby
ON	Flashing	Primary	Wait
Flashing	Flashing	Wait	Indeterminate
OFF	OFF	Wait	Indeterminate
OFF	ON	Standby	Primary

Hot Standby Link LED

A Hot Standby link LED is located on the front of the BMEH58•040 controller:



1 SFP socket for copper or fiber-optic Hot Standby link connection

2 Hot Standby link LED

Use this LED to diagnose the state of the Hot Standby link:

Status	Color	Description
on	green	The port is communicating with the remote controller.
flashing	green	The port is configured and operational, but a Hot Standby link is not made.
off	—	The Hot Standby link is not configured or is not operational.

Ethernet Port Connector LEDs

Each Ethernet RJ45 connector presents a pair of LED indicators:

The Ethernet connector LEDs indicate the following states:

LED	Color	State	Description
ACT	Green	Flashing	Data is being transmitted over the link.
		Off	No transmission activity is occurring.
LNK	Green	On	Link speed = 100 Mbit/s.
	Yellow	On	Link speed = 10 Mbit/s.
	Green / Yellow	Off	No link is established.

Non-Hot Standby Panel LEDs

Refer to the following topics for additional information regarding non-Hot Standby LEDs:

- *LED Diagnostics for M580 Standalone Controllers* in the Modicon M580 Hardware Reference Manual, page 63 for standalone, non-safety–related LEDs.
- *M580 Safety Controllers LED Diagnostics* in the M580 M580 Safety Manual (see Modicon M580, Safety Manual), for safety–related LEDs.

USB Port

Introduction

The USB port is a high-speed, mini-B USB connector, version 2.0 (480 Mbps) that can be used for a Control Expert program or human-machine interface (HMI) panel. The USB port can connect to another USB port, version 1.1 or later.

NOTE: Install M580 USB drivers before connecting the USB cable between the CPU and the PC.

Transparency

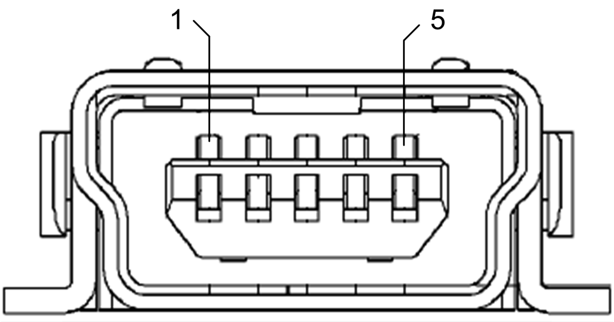
If your system requires transparency between the device connected to the USB port and the M580 device network, add a persistent static route in the device's routing table.

Example of a command to address a device network with IP address `X.X.0.0` (for a Windows PC): `route add X.X.0.0 mask 255.255.0.0 90.0.0.1 -p`

(In this case, `X.X.0.0` is the network address used by the M580 device network, and `255.255.0.0` is the corresponding subnet mask.)

Pin Assignments

The USB port has the following pin positions and pinouts:



Legend:

Pin	Description
1	VBus
2	D-

Pin	Description
3	D+
4	not connected
5	ground
shell	chassis ground

Cables

Use a BMX XCA USB H018 (1.8 m/5.91 ft) or BMX XCA USB H045 (4.5 m/14.764 ft) cable to connect the panel to the CPU. (These cables have a type A connector on one side and the mini-B USB on the other side.)

In a fixed assembly with an XBT-type console connected to the CPU, connect the USB cable to a protection bar (see Modicon X80, Racks and Power Supplies, Hardware Reference Manual). Use the exposed part of the shield or the metal lug on the BMX XCA cable to make the connection.

Ethernet Ports

Introduction

There are three RJ45 Ethernet ports on the front of the controller: one service port, and two device network ports. The ports share the characteristics described below.

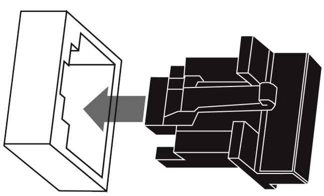
Common Characteristics

The three ports have the same RJ45 connector and use the same type of Ethernet cables.

NOTE: The three Ethernet ports are connected to chassis ground, and the equipment requires an equipotential ground (see Modicon X80, Backplanes and Power Supplies, Hardware Reference Manual).

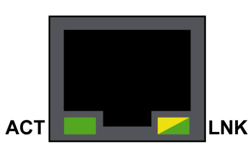
Dust Cover

To keep dust from entering the unused Ethernet ports, cover the unused ports with the stopper:

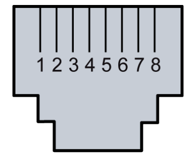


Ethernet Ports

Each RJ45 connector has a pair of LED indicators:



The pin positions, pinouts, and cable connections are the same on the three RJ45 Ethernet ports:

<table>
<tr><th>Pin</th><th colspan="2">Description</th></tr>
<tr><td>1</td><td>TD+</td><td rowspan="9">Pinout:

</td></tr>
<tr><td>2</td><td>TD-</td></tr>
<tr><td>3</td><td>RD+</td></tr>
<tr><td>4</td><td>not connected</td></tr>
<tr><td>5</td><td>not connected</td></tr>
<tr><td>6</td><td>RD-</td></tr>
<tr><td>7</td><td>not connected</td></tr>
<tr><td>8</td><td>not connected</td></tr>
<tr><td>—</td><td>shell/chassis ground</td></tr>
</table>

NOTE: The TD pins (pins 1 and 2) and the RD pins (pins 3 and 6) can be reversed to allow the exclusive use of straight-through cables.

The ports have an auto MDIX capability that automatically detects the direction of the transmission.

It is required to use one of these Ethernet cables to connect to the Ethernet ports:

- TCSECN3M3M••••: Cat 5E Ethernet straight-through shielded cable, rated for industrial use, CE- or UL-compliant
- TCSECE3M3M••••: Cat 5E Ethernet straight-through shielded cable, rated for industrial use, CE-compliant
- TCSECU3M3M••••: Cat 5E Ethernet straight-through shielded cable, rated for industrial use, UL-compliant

The maximum length for a copper cable is 100 m. For distances greater than 100 m, use fiber optic cable. The controller does not have fiber ports. You may use dual ring switches (DRSs) or BMX NRP •••• fiber converter modules (see Modicon M580 Standalone, System Planning Guide for, Frequently Used Architectures) to handle the copper-fiber conversion.

Ethernet Ports on Standalone Controllers

On standalone controllers, the **ACTIVE** LED is green. The **LNK** LED is either green or yellow, depending on the status:

LED	LED Status	Description
ACTIVE	OFF	No activity is indicated on the Ethernet connection.
	ON / flashing	Data is being transmitted and received on the Ethernet connection.
LNK	OFF	No link is established at this connection.
	ON green	A 100 Mbps link* is established at this connection.
	ON yellow	A 10 Mbps link* is established at this connection.
* The 10/100 Mbps links support both half-duplex and full-duplex data transfer and autonegotiation.		

Service Port

The service port is the uppermost of the three Ethernet ports on the front panel of the controller. This port can be used:

- to provide an access point that other devices or systems can use to monitor or communicate with the M580 PAC
- as a standalone DIO port that can support a star, daisy chain, or mesh topology of distributed equipment
- to mirror the controller ports for Ethernet diagnostics. The service tool that views activity on the mirrored port may be a PC or an HMI device.

NOTE: Do not used the service port to connect to the device network unless in some specific conditions described in *Modicon M580, Open Ethernet Network, System Planning Guide*.

The service port does not support the RSTP network protocol. Connecting the service port to the device network, either directly or through a switch/hub, can result in the creation of logical loops in the network, which can adversely affect network performance.

The service port does not support either VLANs or QoS tagging of Ethernet packets. The service port is inherently non-deterministic.

⚠WARNING

UNINTENDED EQUIPMENT OPERATION

Do not connect together the service ports of the Hot Standby controllers.

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

Device Network Dual Ports

When a controller does not support RIO scanning, the two ports below the service port marked **Device Network** are DIO ports.

These controllers do not support RIO scanning:

- BMEP581020 and BMEP581020H
- BMEP582020 and BMEP582020H
- BMEP583020
- BMEP584020

You may use a **Device Network** port to support a star, daisy chain, or mesh topology of distributed equipment. You may use both **Device Network** ports to support a ring topology.

When a controller supports RIO scanning, the two ports below the service port marked **Device Network** are RIO ports. These controllers support RIO scanning:

- BMEP582040, BMEP582040H
- BMEP583040
- BMEP584040
- BMEP585040, BMEP585040C
- BMEP586040, BMEP586040C
- BMEH582040, BMEH582040C
- BMEH584040, BMEH584040C
- BMEH586040, BMEH586040C

When used as RIO ports, both ports connect the controller to the main ring in an Ethernet daisy-chain loop or ring.

For more information about RIO/DIO architectures, refer to the chapter *Modicon M580 System*.

Grounding Considerations

Follow all local and national safety codes and standards.

DANGER

ELECTRIC SHOCK

Wear personal protective equipment (PPE) when working with shielded cables.

Failure to follow these instructions will result in death or serious injury.

The backplane for your M580 PAC is common with the functional ground (FE) plane and must be mounted and connected to a grounded, conductive backplane.

⚠ WARNING

UNINTENDED EQUIPMENT OPERATION

Connect the backplane to the functional ground (FE) of your installation.

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

SD Memory Card

BMXRMS004GPF SD Memory Card

The SD memory card is an option that can be used for application and data storage. The SD memory card slot in the M580 PAC housing is behind a door.

Use a BMXRMS004GPF memory card in your controller. It is a 4 GB, Class 6 card rated for industrial use. Other memory cards, including those used in M340 controllers, are not compatible with M580 PACs.

NOTE: If you insert an incompatible SD memory card in the controller:

- The controller remains in NOCONF state, page 38.
- The controller **BKP** LED turns ON.
- The memory card access LED flashes.

BMXRMS004GPF SD Memory Card Format

The BMXRMS004GPF memory card is formatted specifically for the M580 PAC.

- If you use this card with another controller or tool, the card may not be recognized.
- If you re-format the card in another device – e.g., a camera – the card becomes incompatible for use by an M580 PAC. In this case, you need to return the card to Schneider Electric for re-formatting.

Memory Card Characteristics

These memory card characteristics apply to M580 PACs:

Characteristic	**Value**
global memory size	4 GB
application backup size	200 MB
data storage size	3.8 GB
write/erase cycles (typical)	100,000
operating temperature range	–40...+85 °C (–40...+185 °F)
file retention time	10 years
memory zone for FTP access	data storage directory only

NOTE: Due to formatting, wearout, and other internal mechanisms, the actual available capacity of the memory card is slightly lower than its global size.

Supported Functions

The SD memory card supports read-only data storage functions, page 505.

NOTE: In addition to these read-only data storage functions, you can also read and write to the SD memory card using the following Control Expert project management (see Modicon M580, Hardware, Reference Manual) commands located in the **PLC > Project Backup** menu:

- **Backup Compare**
- **Backup Restore**
- **Backup Save**

Formatting the Memory Card is Unnecessary

The SD memory card comes pre-formatted from the factory. There is no need to manually format the SD memory card using your PC. If you attempt to format the SD memory card, you may alter the formatted structure of the card, thereby rendering the card unusable.

Memory Card Access LED

Introduction

The green memory card access LED underneath the SD memory card door indicates the controller access to the memory when a card is inserted. This LED can be seen when the door is open.

Dedicated LED States

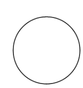
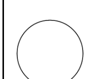
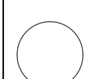
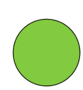
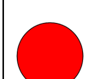
By itself, the **memory card access** LEDs indicate these states:

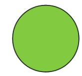
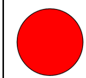
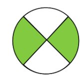
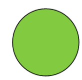
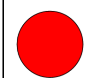
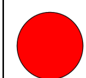
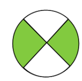
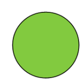
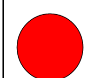
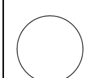
LED Status	Description
ON	The memory card is recognized, but the controller is not accessing it.
flashing	The controller is accessing the memory card.
OFF	The memory card can be removed from the controller slot or the controller does not recognize the memory card.

NOTE: Confirm that the LED is OFF before you remove the card from the slot.

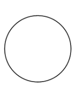
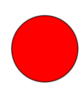
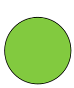
Combined LED Meanings

The access card LED operates together with the **BKP** LED, page 63. Their combined patterns indicate the following diagnostic information:

Memory Card Status	Conditions	Controller State	Memory Card Access LED	BKP LED
no memory card in the slot	—	no configuration		
memory card not OK	—	no configuration		
memory card without project	—	no configuration		

Memory Card Status	Conditions	Controller State	Memory Card Access LED	BKP LED
memory card with a non-compatible project	—	no configuration		
memory card with a compatible project	An error is detected when the project is restored from the memory card to the controller RAM.	no configuration	during transfer:  end of transfer: 	during transfer:  end of transfer: 
	No error is detected when the project is restored from the memory card to the controller RAM.	—	during transfer:  end of transfer: 	during transfer:  end of transfer: 
– no specific circumstances or controller state				

This legend shows the different LED patterns:

Symbol	Meaning	Symbol	Meaning
	off		steady red
	steady green		flashing green

Data Storage Elementary Functions

Data Storage Elementary Functions

These `DataStorage_EF` elementary functions are supported in Control Expert for the M580 controllers:

EF	Controller		Description
	BMEP58•0•0	BMEH58•040	
`CLOSE_FILE`	X	X	The `CLOSE_FILE` function closes the file identified by the file descriptor attribute. If another user is working on the same file via a different descriptor, the file remains open.
`CREATE_FILE` (see EcoStruxure™ Control Expert, System, Block Library)	X	—	The `CREATE_FILE` function creates a new file, assigns it the specified file name, and indicates the purposes for which the file is opened: read-only, write-only, read-write.
`DELETE_FILE` (see EcoStruxure™ Control Expert, System, Block Library)	X	—	The `DELETE_FILE` function deletes the specified file.
`GET_FILE_INFO` (see EcoStruxure™ Control Expert, System, Block Library)	X	X	The `GET_FILE_INFO` function retrieves information about a specified target file. Execute the `OPEN_FILE` function for the target file before executing the `GET_FILE_INFO` function, because the identity of the target file comes from the output parameter of the `OPEN_FILE` block.
`GET_FREESIZE` (see EcoStruxure™ Control Expert, System, Block Library)	X	X	The `GET_FREESIZE` function displays the amount of available space on the SD memory card.
`OPEN_FILE` (see EcoStruxure™ Control Expert, System, Block Library)	X	X (read only)	The `OPEN_FILE` function opens a specified existing file.
`RD_FILE_TO_DATA` (see EcoStruxure™ Control Expert, System, Block Library)	X	X	The `RD_FILE_TO_DATA` function enables reading data from a file, at the current position in the file, and copies the data to a direct address variable, a located variable, or an unlocated variable.
`SEEK_FILE` (see EcoStruxure™ Control Expert, System, Block Library)	X	X	The `SEEK_FILE` function sets the current byte offset in the file to a new specified offset position, which can be: the offset, the current position plus the offset, the file size plus the offset.
`SET_FILE_ATTRIBUTES` (see EcoStruxure™ Control Expert, System, Block Library)	X	—	The `SET_FILE_ATTRIBUTES` function sets the read-only status of a file attribute. Read-only status can be set or cleared. This function can be applies only to a file that is already open via the `CREATE_FILE` or `OPEN_FILE` function.

EF	Controller		Description
	BMEP58•0•0	BMEH58•040	
`WR_DATA_TO_FILE` (see EcoStruxure™ Control Expert, System, Block Library)	X	X	The `WR_DATA_TO_FILE` function enables the writing of the value of a direct address variable, a located variable, or an unlocated variable to a file. The value is written to the current position in the file. After the write, the current position in the file is updated.
X (supported) — (not supported)			

For additional information on each function, refer to the chapter *Implementing File Management* (see EcoStruxure™ Control Expert, System, Block Library).

Firmware Update

Depending on the initial version and the targeted version of the controller, the procedure is different. A new boot loader was introduced at version 4.x. Thus, the procedures to update from an earlier version (V3.22 or earlier) to version V4.x, or to downgrade from a V4.x version to an earlier version, require specific procedures.

For detailed procedures for firmware update, refer to *Modicon M580 Controller Firmware Installation Guide*.

Installing and Diagnosing Modules on the Local Rack

What's in This Part

Introduction

This part provides instructions for installing and assembling M580 CPUs.

Installing Modules in an M580 Rack

What's in This Chapter

Overview

This chapter explains how to install a CPU module in an M580 rack.

Module Guidelines

Guidelines

Rack Position	Rack Type	Slots Marking			
		00	**01**	**02**	**...n (1)**
local	main rack	controller		module	module
	X80 extended rack	module	module	module	module
	Premium extended rack	module	module	module	module
remote drop	main rack	(e)X80 EIO adapter module	module	module	module
	extended rack	module	module	module	module

1 slots from number **03** to last numbered slot of the rack

NOTE: When your installation has more than one rack in the local rack or at a remote drop, the BMX XBE 1000 rack exender module goes in the slot marked **XBE** of the X80 racks.

Check that the controller is installed in the two slots marked **00** and **01** on the local rack before powering up the system. If the controller is not installed in these two slots, the CPU starts in NOCONF state, page 38 and uses the configured IP address (not the default IP address, which starts with 10.10 and uses the last two bytes of the MAC address).

NOTE: When the last two octets of the MAC address (*MAC5.MAC6*) correspond to *0.0* in the default address, make a point-to-point cable connection between your computer and the controller, communication module, or other module.

Services and Addresses

IP addresses: This table shows the availability of network services regarding the relationship between the controller's IP addresses and its ports.

NOTE: When the Ethernet IP address is assigned in the same network range as the USB port (90.0.0.x), the USB port does not work.

Service	BMEP58•040 (DIO, ERIO)	BMEP58•020 Controller (DIO)
EtherNet/IP scanner	• IP A (RIO) • IP main (DIO)	• IP A (DI•R supports redundant owner) • IP main (DIO)
Modbus	IP main	IP main
FDR server and DHCP	• IP A (RIO) • IP main (DIO)	IP main
SNTP server	IP A	IP main
other services*	IP main	IP main
SNMP source IP address	IP A or IP main	IP A or IP main
SNTP client source IP address	IP A or IP main	IP A or IP main
LLDP	IP main	IP main
RSTP	IP main	IP main
*Web server. EtherNet/IP adapter, Modbus server/FTP		

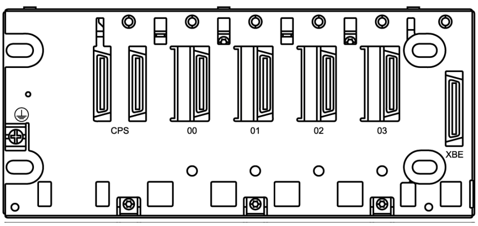
MAC addresses: This table shows the availability of network services in terms of the relationship between the controller's MAC addresses and its ports:

Service	BMEP58•040 (DIO, ERIO)	BMEP58•020 Controller (DIO)
EtherNet/IP scanner	module MAC	module MAC
Modbus	module MAC	module MAC
FDR server and DHCP	module MAC	module MAC
SNTP server	module MAC	module MAC
other services*	module MAC	module MAC
SNMP source IP address	module MAC	module MAC

Service	BMEP58•040 (DIO, ERIO)	BMEP58•020 Controller (DIO)
SNTP client source IP address	module MAC	module MAC
LLDP	port MAC = (module MAC + 1, 2, 3, or 4)**	port MAC = (module MAC + 1, 2, 3, or 4)**
RSTP	port MAC = (module MAC + 1, 2, or 3)**	port MAC = (module MAC + 1, 2, or 3)**
*Web server. EtherNet/IP adapter, Modbus server/FTP **Ports: • **port 1:** module MAC + 1 (service port) • **port 2:** module MAC + 2 • **port 3:** module MAC + 3 • **port 4:** module MAC + 4 (Ethernet backplane)		

Rack Markings

Example of BMXXBP•••• (PV:02 and any subsequent supporting versions) rack with slot markings:



Installing the Controller

Introduction

You can install any standard controller (BMEP58•0•0) or any Hot Standby controller (BMEH58•0•0) in these racks:

- BMXXBP•••• (PV:02 and any subsequent supporting versions) X Bus rack
- BMEXBP••00 or BMEXBP••02 Ethernet rack

Exception: You can install the BMXCPS4002 only on these dual-bus (Ethernet and X Bus) racks:

- BMEXBP0602
- BMEXBP1002

Installation Precautions

An M580 controller is powered by the rack bus. Confirm that the rack power supply is turned off before installing the controller.

DANGER

HAZARD OF ELECTRIC SHOCK, EXPLOSION OR ARC FLASH

- Disconnect all power from all equipment including connected devices prior to removing any covers or doors, or installing or removing any accessories, hardware, cables, or wires except under the specific conditions specified in the appropriate hardware guide for this equipment.
- Always use a properly rated voltage sensing device to confirm the power is off where and when indicated.
- Replace and secure all covers, accessories, hardware, cables, and wires and confirm that a proper ground connection exists before applying power to the unit.
- Use only the specified voltage when operating this equipment and any associated products.

Failure to follow these instructions will result in death or serious injury.

Remove the protective cover from the rack slot connectors before plugging the module in the rack.

⚠ WARNING

UNINTENDED EQUIPMENT OPERATION

Ensure that the controller does not contain an unsupported SD memory card before powering up the controller.

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

NOTE:

- Check that the memory card slot door is closed after a memory card is inserted in the controller, and remains closed during operations.
- Refer to `%SW97` to check the status of the SD card.

Grounding Considerations

Follow all local and national safety codes and standards.

⚠ ⚠ DANGER

ELECTRIC SHOCK

Wear personal protective equipment (PPE) when working with shielded cables.

Failure to follow these instructions will result in death or serious injury.

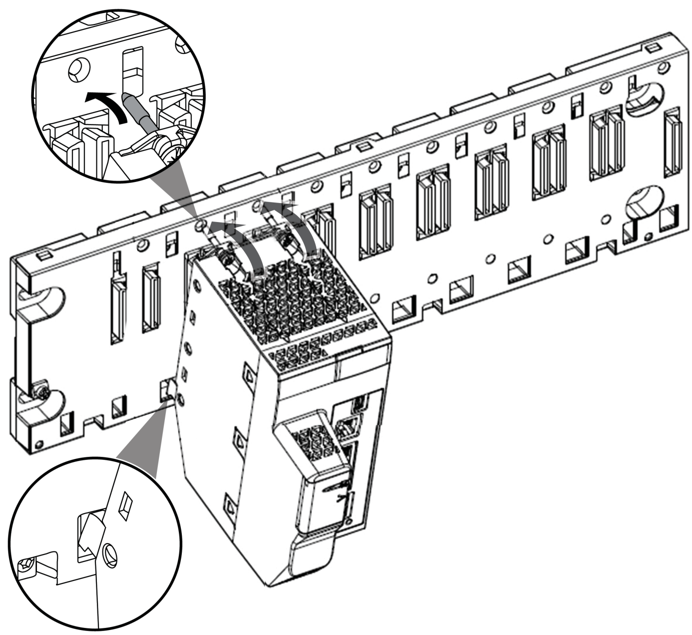
NOTE: Refer to the ground protection information provided in the *Electrical installation guide* and *Control Panel Technical Guide, How to protect a machine from malfunctions due to electromagnetic disturbance*, page 15.

Installing the Controller

Install the controller in the rack slots marked **00** and **01**. If you do not install the controller in these two slots, it starts in NOCONF state, page 38 state and uses the default IP address, which starts with `10.10` and uses the last two bytes of the MAC address.

NOTE: When the last two octets of the MAC address (*MAC5.MAC6*) correspond to *0.0* in the default address, make a point-to-point cable connection between your computer and the controller, communication module, or other module.

Install a controller in a rack:

Step	Action	Illustration
1	Verify that the power supply is turned off.	–
2	If you are installing a Hot Standby controller, on the back of the controller, set the A/B/Clear selector switch, page 58 to the appropriate selection, “A” or “B”. **NOTE:** When you later install the companion Hot Standby controller, set its rotary switch to the other A/B position.	–
3	Verify that: • if an SD memory card is used, it is supported by the controller • the connectors’ protective covers are removed • the controller is placed on the slots marked **00** and **01**	
4	Position the locating pins situated at the rear of the module (on the bottom part) in the corresponding slots in the rack.	
5	Swivel the module towards the top of the rack so that the module sits flush with the back of the rack. The module is now set in position.	
6	Tighten the 2 screws on top of the controller to maintain the module in place on the rack. tightening torque: 0.7...1.5 N•m (0.52...1.10 lbf-ft).	–

Installing Modules in the Second Local Rack

If you are installing a Hot Standby system, you need to install the same collection of modules, with the same versions of firmware, that were installed on the first rack. Install each module in the same slot that its counterpart occupies on the first rack. Follow the same procedure described above, except set the A/B/Clear selector switch, page 58 on the back of the standby controller to other A/B position.

Connecting the Hot Standby Local Racks

If you are installing a Hot Standby system, you need to connect the communication link to controller A and controller B before applying power to either local rack. If you start up the controllers before they are connected via the Hot Standby link, both controllers attempt to assume the role of primary controller in your Hot Standby system.

DANGER

HAZARD OF ELECTRIC SHOCK

- Connect the functional ground (FG) terminal of the power supply module directly to the protective earth screw of the rack.
- Do not chain the function ground (FG) terminals of redundant power supply modules together.
- Do not connect anything else to the functional ground (FG) terminal of the power supply module.

Failure to follow these instructions will result in death or serious injury.

DANGER

HAZARD OF ELECTRIC SHOCK

- Use only cables with ring or spade lugs and ensure that there is a ground connection.
- Make sure that grounding hardware is tightened properly.

Failure to follow these instructions will result in death or serious injury.

Before you connect the two Hot Standby local racks, verify that an equipotential grounding system (see Modicon X80, Racks and Power Supplies, Hardware Reference Manual) is in place that includes the two racks (plus any other equipment you intend to connect to the two Hot Standby local racks).

When installing modules with fiber optic transceivers, do the following to help prevent dust and pollution from disrupting light production into the fiber optic cable.

NOTICE

EQUIPMENT DAMAGE

- Keep caps on jumpers and transceivers when not in use.
- Insert the optical cable into the transceivers carefully, respecting the longitudinal axis of the transceiver.
- Do not use force when inserting the cable into the optical transceivers.

Failure to follow these instructions can result in equipment damage.

Each Hot Standby controller includes on its front face an SFP socket, page 57. This socket can accept an SFP transceiver module (see Modicon M580 Hot Standby, System Planning Guide for, Frequently Used Architectures) for either copper or single mode fiber optic cabling of the Hot Standby link. Your choice of SFP transceiver and cabling is determined by the distance between the two Hot Standby local racks (see Modicon M580 Hot Standby, System Planning Guide for, Frequently Used Architectures).

Installing an SD Memory Card in a CPU

Introduction

The BME•58•••• CPUs support the use of the BMXRMS004GPF 4GB SD memory card.

Memory Card Maintenance

To keep the memory card in normal working order:

- Avoid removing the memory card from its slot when the CPU accesses the card (memory card access green LED ON or blinking).
- Avoid touching the memory card connectors.
- Keep the memory card away from electrostatic and electromagnetic sources as well as heat, sunlight, water, and moisture.
- Avoid impact on the memory card.
- Before sending a memory card by post (mail), check the postal service security policy. In some countries, the postal service exposes mail to high levels of radiation as a security measure. These high levels of radiation may erase the contents of the memory card and render it unusable.
- If a card is extracted without generating a rising edge of the bit %S65 and without checking that the memory card access green LED is OFF, the data (files, application, and so on) may be lost or become unreliable.

Memory Card Insertion Procedure

Procedure for inserting a memory card into a BME•58•••• CPU:

Step	Description
1	Open the SD memory card protective door.
2	Insert the card in its slot.
3	Push the memory card until you hear a click. **Result:** The card should now be clipped into its slot. **Note:** Insertion of the memory card does not force an application restore.
4	Close the memory card protective door.

Memory Card Removal Procedure

NOTE: Before removing a memory card, a rising edge on bit %S65 needs to be generated. If a card is extracted without generating a rising edge of the bit `%S65` and without checking that the memory card access green LED is OFF, the data may be lost.

Procedure for removing a memory card from a BME•58•••• CPU:

Step	Description
1	Generate a rising edge on bit `%S65`.
2	Check that the memory card access green LED is OFF.
3	Open the SD memory card protective door.
4	Push the memory card until you hear a click, then release the pressure on the card. **Result:** The card should unclip from its slot.
5	Remove the card from its slot. **Note:** The memory card access green LED is ON when the memory card is removed from the CPU.
6	Close the memory card protective door.

M580 Diagnostics

What's in This Chapter

Introduction

This chapter provides information on diagnostics that can be performed via hardware indications (based on LED status) and system bits or words when necessary. The entire M580 system diagnostics is explained in the *Modicon M580 System Planning Guide*.

The CPU manages different types of detected error:

- detected errors that can be recovered and do not change the PAC behavior unless specific options are used
- detected errors that cannot be recovered and lead the CPU to the halt state
- CPU or system detected errors that lead the CPU to an error state

Blocking Conditions

Introduction

Blocking conditions caused during the execution of the application program do not cause system errors, but they stop the CPU. The CPU goes into the HALT state, page 38.

NOTE:

- When a BMEH58•040 CPU is in the HALT state, the RIO and DIO outputs behave the same way as they do when the CPU is in STOP state, page 454.
- For information about Hot Standby diagnostics, refer to the diagnostics chapter (see Modicon M580 Hot Standby, System Planning Guide for, Frequently Used Architectures) in the M580 Hot Standby installation guide.

Diagnostics

Visual indications of a blocking condition are the **ERR** LED on the CPU front panel, page 63.

A description of the error is provided in system word `%SW125`.

The address of the instruction that was executing when the blocking condition occurred is provided by system words `%SW126` through `%SW127`.

`%SW125` system word values and corresponding blocking condition description:

`%SW125` Value (hex)	**Blocking Condition Description**
0•••	execution of an unknown function
0002	SD card signature feature (used with *SIG_CHECK* and *SIG_WRITE* functions)
2258	execution of the HALT instruction
2259	execution flow different than the reference flow
23••	execution of a CALL function towards an undefined subroutine
81F4	SFC node incorrect
82F4	SFC code inaccessible
83F4	SFC work space inaccessible
84F4	too many initial SFC steps
85F4	too many active SFC steps
86F4	SFC sequence code incorrect
87F4	SFC code description incorrect
88F4	SFC reference table incorrect
89F4	SFC internal index calculation detected error
8AF4	SFC step status not available
8BF4	SFC memory too small after a change due to a download
8CF4	transition/action section inaccessible
8DF4	SFC work space too small
8EF4	version of the SFC code older than the interpreter
8FF4	version of the SFC code more recent than the interpreter
90F4	poor description of an SFC object: NULL pointer
91F4	action identifier not authorized
92F4	poor definition of the time for an action identifier

`%SW125` **Value (hex)**	**Blocking Condition Description**
93F4	macro step cannot be found in the list of active steps for deactivation
94F4	overflow in the action table
95F4	overflow in the step activation/deactivation table
9690	error detected in the application CRC check (checksum)
DE87	calculation detected error on numbers with decimal points
DEB0	watchdog overrun
DEF0	division by 0
DEF1	character string transfer detected error
DEF2	capacity exceeded
DEF3	index overrun
DEF7	SFC execution detected error
DEFE	SFC steps undefined

Restarting the Application

After a blocking condition has occurred, the halted CPU needs to be initialized. The CPU can also be initialized by setting the `%S0` bit to 1.

When initialized, the application behaves as follows:

- the data resume their initial value
- tasks are stopped at end of cycle
- the input image is refreshed
- outputs are controlled in fallback position

The RUN command then allows the application to be restarted.

Non-blocking Conditions

Introduction

The system enters a non-blocking condition when it detects an input/output error on the backplane bus (X Bus or Ethernet) or through execution of an instruction, which can be processed by the user program and does not modify the CPU status.

Conditions Linked to I/O Diagnostics

A non-blocking condition linked to the I/O is diagnosed with the following indications:

- CPU **I/O** LED pattern: steady ON
- module **I/O** LED pattern: steady ON
- system bits (type of error):
  - `%S10` set to 0: I/O error detected on one of the modules on the rack (channel power supply detected error, or broken channel, or module not compliant with the configuration, or inoperative module, or module power supply detected error)
  - `%S16` set to 0: I/O error detected in the task in progress
  - `%S40`–`%S47` set to 0: I/O error detected on rack address 0 to 7
- system bits and words combined with the channel having an error detected (I/O channel number and type of detected error) or I/O module Device DDT information (for modules configured in Device DDT addressing mode):
  - bit `%Ir.m.c.ERR` set to 1: channel error detected (implicit exchanges)
  - word `%MWr.m.c.2`: the word value indicates the type of error detected on the specified channel and depends on the I/O module (implicit exchanges)

Conditions Linked to Execution of the Program Diagnostics

A non-blocking condition linked to execution of the program is diagnosed with the following system bits and words:

- system bits (type of error detected):
  - %S15 set to 1: character string manipulation error detected
  - %S18 set to 1: capacity overrun, error detected on a floating point, or division by 0 (see EcoStruxure™ Control Expert, Operating Modes)
  - %S20 set to 1: index overrun
- system word (nature of the error detected):
  - `%SW125`, page 97 (always updated)

NOTE: The CPU can be forced to the HALT state, page 38 on program execution recoverable condition.

There are 2 ways to force a CPU to stop when non-blocking errors linked to the execution of the program are detected:

- Use the diagnostic program function accessible through Control Expert programming software.
- set the system bit `%S78` (HALTIFERROR) to 1.

CPU or System Errors

Introduction

CPU or system errors are related either to the CPU (equipment or software) or to the rack internal bus wiring. The system can no longer operate correctly when these errors occur.

A CPU or system error causes the CPU to stop in ERROR mode and requires a cold restart. Before applying a cold restart, set the CPU to STOP mode to keep the PAC from returning to ERROR mode.

Diagnostics

A CPU or system error is diagnosed with the following indications:

- CPU **I/O** LED pattern: steady on
- system word `%SW124` value defines the detected error source:
  - 80 hex: system watchdog error or rack internal bus wiring error
  - 81 hex: rack internal bus wiring error
  - 90 hex: interruption not foreseen, or system task pile overrun

CPU Application Compatibility

Application Compatibility

These tables show the standalone (BMEP58•0•0) and Hot Standby (BMEH58•0•0) CPUs that can download and execute applications that are built on a different CPU.

These applications are built on standalone CPUs and transferred to standalone CPUs:

Standalone CPUs	Download and execute the application here (BMEP58...								
Build the application here (↓).	**1020**	**2020**	**2040**	**3020**	**3040**	**4020**	**4040**	**5040**	**6040**
BMEP581020	X	X	–	X	–	X	–	–	–
BMEP582020	–	X	–	X	–	X	–	–	–
BMEP582040	–	–	X	–	X	–	X	X	X
BMEP583020	–	–	–	X	–	X	–	–	–
BMEP583040	–	–	–	–	X	–	X	X	X
BMEP584020	–	–	–	–	–	X	–	–	–
BMEP584040	–	–	–	–	–	–	X	X	X
BMEP585040	–	–	–	–	–	–	–	X	X
BMEP586040	–	–	–	–	–	–	–	–	X
X yes **–** no									

These applications are built on Hot Standby CPUs and transferred to Hot Standby CPUs:

Hot Standby CPUs	Download and execute the application here (BMEH58...		
Build the application here (↓).	**2040**	**4040**	**6040**
BMEH582040	X	X	X
BMEH584040	–	X	X
BMEH586040	–	–	X
X yes **–** no			

Example: An application built on a BMEP583020 CPU can only be downloaded or executed on a BMEP583020 or a BMEP584020 CPU.

NOTE: For all M580 CPUs, versions 1.10 and 2.00 are not compatible. You cannot configure a CPU V2.00, and download the application to a CPU V1.10.

Processor Performance

What's in This Chapter

Introduction

This section describes BMEP58•0•0 processor performance.

Execution of Tasks

General

BME P58 •0•0 processors can execute single-task and multi-task applications. Unlike a single-task application, which only executes master tasks, a multi-task application.defines the task execution priorities.

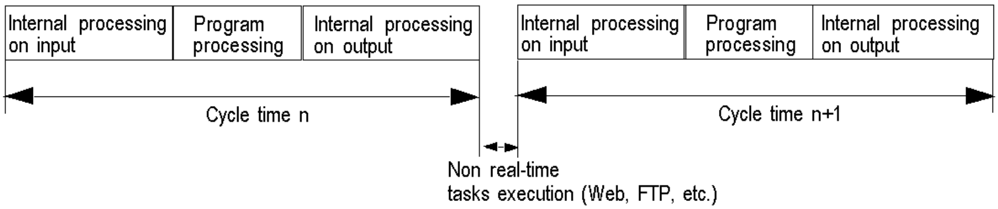
Master Task

The master task represents the application program's main task. You can choose from the following MAST task execution modes:

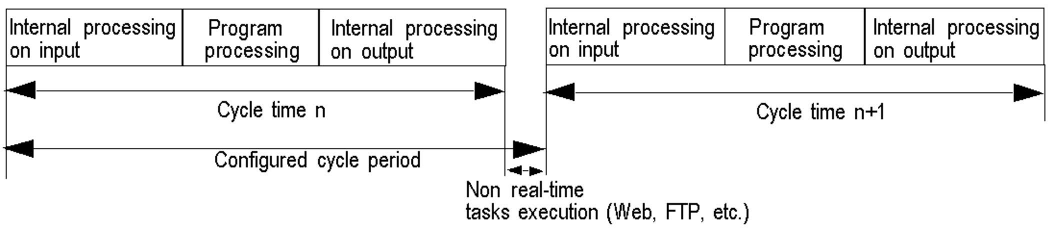
- Cyclical (default setup): Execution cycles are performed in sequence, one after the other.
- Periodical: A new cycle is started periodically, according to a user-defined time period (1 - 255 ms).

  If the execution time is longer than the period configured by the user, the bit `%S19` is set to 1, and a new cycle is launched.

The following illustration shows the cyclical execution of the MAST task:



The following illustration shows the periodical execution of the MAST task:



Both MAST task cycle modes are controlled by a watchdog.

The watchdog is triggered if the MAST task execution time is longer than the maximum period defined in the configuration, and causes a software error. The application then goes into HALT status, and the bit `%S11` is set to 1 (the user must reset it to 0).

The watchdog value (`%SW11`) may be configured between 10 ms and 1,500 ms (default value: 250 ms).

NOTE: Configuring the watchdog to a value that is less than the period is not allowed.

In periodical operating mode, an additional check detects when a period has been exceeded. The PLC will not switch off if the period overrun remains less than the watchdog value.

Bit `%S19` signals a period overrun. It is set to 1 by the system when the cycle time becomes longer than the task period. Cyclical execution then replaces periodical execution.

The MAST task can be checked with the following system bits and system words:

System Object	Description
`%SW0`	MAST task period
`%S30`	Activation of the master task
`%S11`	Watchdog default
`%S19`	Period exceeded

System Object	Description
%SW27	Last cycle overhead time (in ms)
%SW28	Longest overhead time (in ms)
%SW29	Shortest overhead time (in ms)
%SW30	Last cycle execution time (in ms)
%SW31	Longest cycle execution time (in ms)
%SW32	Shortest cycle execution time (in ms)

Fast Task

The FAST task is for periodical processing and processing over short durations.

FAST task execution is periodical and must be quick so that no lower priority tasks overrun. The FAST task period can be configured (1 - 255 ms). The FAST task execution principle is the same as for periodical execution of the master task.

The FAST task can be checked with the following system bits and system words:

System Object	Description
%SW1	FAST task period
%S31	Activation of the fast task
%S11	Watchdog default
%S19	Period exceeded
%SW33	Last cycle execution time (in ms)
%SW34	Longest cycle execution time (in ms)
%SW35	Shortest cycle execution time (in ms)

Event Tasks

With event processing, the application program’s reaction time can be reduced for events originating from:

- input/output modules (EVTi blocks)
- events timers (TIMERi blocks)

Event processing execution is asynchronous. The occurrence of an event reroutes the application program towards the process associated with the input/output channel, or to the event timer that caused the event.

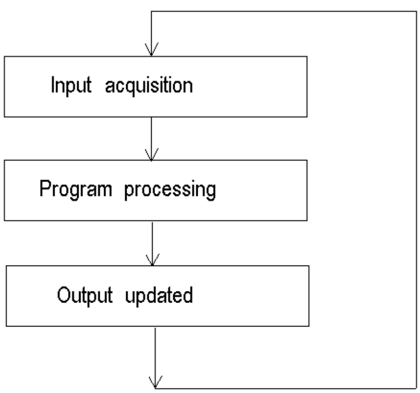
Event tasks can be checked with the following system bits and system words:

System Object	Description
%S38	Activation of events processing
%S39	Saturation of the event signal management stack.
%SW48	Number of IO events and telegram processes executed **NOTE:** TELEGRAM is available only for PREMIUM (not on Quantum neither M340)

Single Task Execution

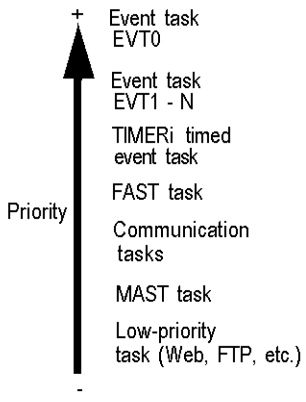
A single-task application program is associated with one task; the MAST task.

The following diagram shows a single-task application's execution cycle:

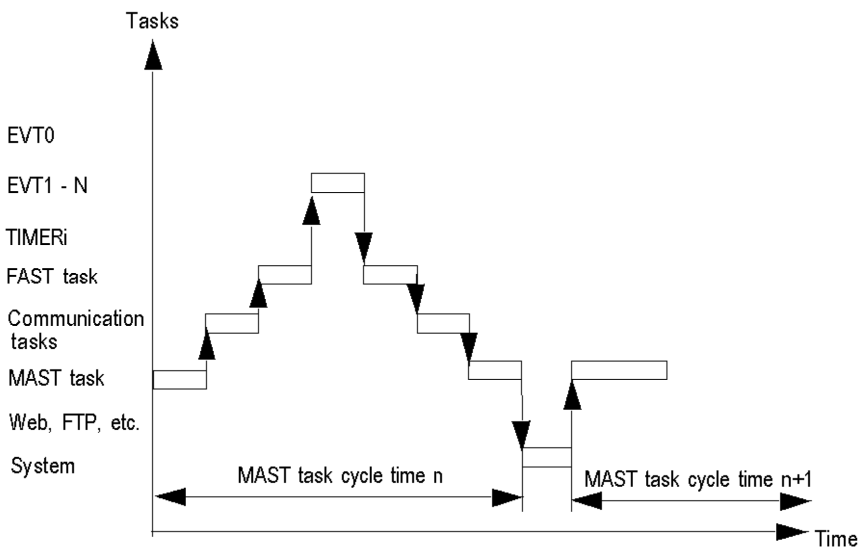


Multi-Task Execution

The following diagram shows the level of priority of the tasks in a multi-task structure:



The following diagram shows the execution of tasks in a multi-task structure:



MAST Task Cycle Time: Introduction

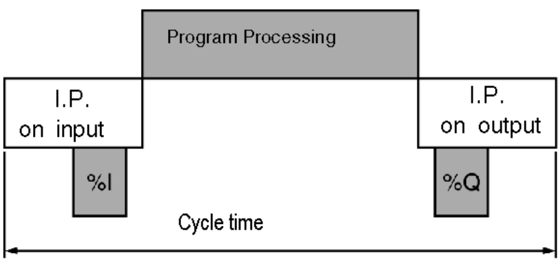
General

The MAST task cycle time is the sum of the following:

- internal processing time on input,
- master task program processing time,
- internal processing time on output.

Illustration

The following diagram defines the MAST task cycle time:



I.P. Internal Processing.

MAST Task Cycle Time: Program Processing

Definition of Program Processing Time

Program processing time is equivalent to the time needed to execute application code.

Application Code Execution Time

The application code execution time is the sum of the times needed for the application program to execute each instruction, at each PLC cycle.

The table below gives the execution time for 1 K of instructions (i.e. 1024 instructions).

Processors	**Application Code Execution Time (1)**	
	100 % Boolean Program	65 % Boolean + 35 % Digital Program
BMEP581020, BMEP581020H BMEP582020, BMEP582020H BMEP582040, BMEP52040H BMEP583020 BMEP583040 BMEP584020 BMEP584040 BMEP585040, BMEP585040C BMEP586040, BMEP586040C	0.12 milliseconds	0.15 milliseconds

(1) All instructions are executed at each PLC cycle.

MAST Task Cycle Time: Internal Processing on Input and Output

General

The internal processing time for inputs and outputs is the sum of the following:

- MAST task system overhead time
- maximum communication system reception time and input management time for implicit inputs/outputs
- maximum communication system transmission time and output management time for implicit inputs/outputs

MAST Task System Overhead Time

For BMEP58•0•0 processors, the MAST task system overhead time is 700 μs.

NOTE: Three system words give information on the MAST task system overhead times:

- `%SW27`: last cycle overhead time
- `%SW28`: longest overhead time
- `%SW29`: shortest overhead time

Implicit Input/Output Management Time

The implicit input management time is the sum of the following:

- Fixed base of 25 μs
- Sum of the input management times for each module (in the following table, IN)

The implicit output management time is the sum of the following:

- Fixed base of 25 μs (FAST), 73 μs (MAST)
- Sum of the output management times for each module (in the following table, OUT)

The table below shows the input (IN) and output (OUT) topological **(T)** and DDT **(DDT)** management times for each module.

Module		Input Management Time (IN) (µs)	Output Management Time (OUT) (µs)	Total Management Time (IN+OUT) (µs)
BMXDDI1602, 16 discrete inputs module	*T:*	60	40	100
	DDT:	30	29	60
BMXDDI3202K, 32 discrete inputs module	*T:*	67	44	111
	DDT:	34	31	64
BMXDDI6402K, 64 discrete inputs module	*T:*	87	63	150
	DDT:	40	43	83
BMXDDO1602, 16 discrete outputs module	*T:*	60	45	105
	DDT:	31	34	64
BMXDDO1612, 16 discrete outputs module	*T:*	60	45	105
	DDT:	30	33	63
BMXDDO3202 BMXDDO3202H	*T:*			
	DDT:			
BMXDDO3202K, 32 discrete outputs module	*T:*	67	51	118
	DDT:	33	35	69
BMXDDO6402K, 64 discrete outputs module	T:	87	75	162
	DDT:	40	50	89
BMXDDM16022, 8 discrete inputs and 8 discrete outputs module	T:	68	59	127
	DDT:	44	51	95
BMXDDM3202K, 16 discrete inputs and 16 discrete outputs module	*T:*	75	63	138
	DDT:	48	54	102
BMXDDM16025, 8 discrete inputs and 8 discrete outputs module	*T:*	68	59	127
	DDT:	44	51	95
BMXDAI0805, 8 discrete inputs module	*T:*	60	40	100
	DDT:	28	28	56
BMXDAI1602, 16 discrete inputs module	*T:*	60	40	100
	DDT:	29	29	59
BMXDAI1603, 16 discrete inputs module	*T:*	60	40	100
	DDT:	30	29	59

Module		Input Management Time (IN) (µs)	Output Management Time (OUT) (µs)	Total Management Time (IN+OUT) (µs)
BMXDAI1604, 16 discrete inputs module	*T:*	60	40	100
	DDT:	30	29	58
BMXDAO1605, 16 discrete outputs module	*T:*	60	45	105
	DDT:	30	33	64
BMXAMI0410 analog module	*T:*	103	69	172
	DDT:	43	42	85
BMXAMI0800 analog module	*T:*	103	69	172
	DDT:	63	65	129
BMXAMI0810 analog module	*T:*	103	69	172
	DDT:	63	65	128
BMXAMO0210 analog module	*T:*	65	47	112
	DDT:	30	35	65
BMXAMO802 analog module	*T:*	110	110	220
	DDT:	47	74	121
BMXAMM0600 analog module	*T:*	115	88	203
	DDT:	82	80	162
BMXDRA0804, 8 discrete outputs module	*T:*	56	43	99
	DDT:	27	31	58
BMXDRA0805, 8 discrete outputs module	*T:*	56	43	99
	DDT:	28	31	59
BMXEHC0200 dual-channel counting module	*T:*	102	93	195
	DDT:	101	108	208
BMXEHC0800 eight-channel counting module	*T:*	228	282	510
	DDT:	261	317	578

Communication System Time

Communication (excluding telegrams) is managed during the MAST task internal processing phases:

- on input for receiving messages
- on output for sending messages

The MAST task cycle time is, therefore, affected by the communication traffic. The communication time spent per cycle varies considerably, based on the following elements:

- traffic generated by the processor: number of communication EFs active simultaneously
- traffic generated by other devices to the processor, or for which the processor ensures the routing function as master

This time is only spent in the cycles where there is a new message to be managed.

NOTE: These times may not all occur in the same cycle. Messages are sent in the same PLC cycle as instruction execution when communication traffic is low. However, responses are never received in the same cycle as instruction execution.

MAST Task Cycle Time Calculation

General

The MAST task cycle time can be calculated before the implementation phase, if the desired PLC configuration is already known. The cycle time may also be determined during the implementation phase, using the system words `%SW30` - `%SW32`.

Calculation Method

The following table shows how to calculate the MAST task cycle time.

Step	Action
1	Calculate the input and output internal processing time by adding the following times: • MAST task system overhead time (see Modicon M340, Processors, Setup Manual) • maximum communication system reception time and input management time for implicit inputs/outputs (see Modicon M340, Processors, Setup Manual) • maximum communication system transmission time and output management time for implicit inputs/outputs (see Modicon M340, Processors, Setup Manual)
2	Calculate the program processing time (see Modicon M340, Processors, Setup Manual) according to the number of instructions and the type (Boolean, digital) of program.
3	Add together the program processing time, and the input and output internal processing time.

FAST Task Cycle Time

Definition

The FAST task cycle time is the sum of the following:

- program processing time
- internal processing time on input and output

Definition of Internal Processing Time on Input and Output

The internal processing time on input and output is the sum of the following:

- FAST task system overhead time
- implicit input/output management time on input/output (see Modicon M340, Processors, Setup Manual)

For the BMEP58•0•0 processors, the FAST task system overhead time is 130 µs.

Event Response Time

General

The response time is the time between an edge on an event input and the corresponding edge on an output positioned by the program in an event task.

Response Time

The following table gives the response time for the BMEP58•0•0 processors with an application program of 100 Boolean instructions and the module.

Processors	Minimum	Typical	Maximum
BMEP58•0•0	1625 μs	2575 μs	3675 μs

Configuring the Controller in Control Expert

What's in This Part

Introduction

This part describes how to configure an M580 PAC system with Control Expert.

NOTE: The device configuration procedure is valid when configuring a project with Control Expert Classic. When you configure your device from a system project, some commands are disabled in the Control Expert editor. In this case, you need to configure these parameters at the system level by using the Topology Manager.

M580 CPU Configuration

What’s in This Chapter

Introduction

The chapter describes the configuration of the M580 CPU.

Control Expert Projects

Overview

Use this section to add an M580 CPU to your Control Expert application.

Creating a Project in Control Expert

Introduction

If you have not created a project in Control Expert and installed a power supply and an M580 CPU, use the following steps to create a new Control Expert project containing these components:

- M580 CPU, page 24
- power supply

Creating and Saving a Control Expert Project

Follow these steps to create a Control Expert project:

Step	Action
1	Open Control Expert.
2	Click **File > New...** to open the **New Project** window.
3	In the **PLC** window, expand the **Modicon M580** node, and select a CPU. **NOTE:** Refer to the CPU Scanner Service, page 26 topic to select the appropriate CPU, depending upon your DIO and RIO needs. In the **Rack** window, expand the **Modicon M580 local drop** node, and select a rack.
4	Click **OK**. **Result**: The **Security enforcement** dialog opens. You can use this dialog to: • Create an Application password: to help prevent both theft of and unauthorized access to the new application. • Create also a File encryption password: to help prevent malicious file corruption and intellectual property theft. • Elect not to create either an Application password or a File encryption password.
5	(Optional) To create an Application password, use the **Entry** and **Confirmation** fields to input and confirm the password. The Application password needs to: • be a minimum 8 characters long. • contain at lease one uppercase character, at least one lowercase character, one number, and one non-alphanumeric character.
6	(Optional) To create an File encryption password, use the **Entry** and **Confirmation** fields to input and confirm the password. The File encryption password needs to: • be a minimum 8 characters long. • contain at lease one uppercase character, at least one lowercase character, one number, and one non-alphanumeric character.

Step	Action
	• be different from the Application password.
7	Click **OK** to save your new password(s) or click **Cancel** to proceed without Application and File encryption passwords. **Result**: The **Project Browser** dialog opens.
8	Click **File > Save** to open the **Save As** dialog.
9	Enter a **File name** for your Control Expert project and click **Save**. **Result**: Control Expert saves your project to the specified path location.

Changing the Default Storage Location (Optional)

You can change the default location that Control Expert uses to store project files before you click **Save**:

Step	Action
1	Click **Tools > Options** to open the **Options Management** window.
2	In the left pane, navigate to **Options > General > Paths**.
3	In the right pane, type in a new path location for the **Project path**. You can also edit these items: • **Import/Export file path** • **XVM path** • **Project settings templates path**
4	Click **OK** to close the window and save your changes.

Selecting a Power Supply

A default power supply is automatically added to the rack in a new Control Expert project. To use a different power supply, follow these steps:

Step	Action
1	In the **Project Browser**, double-click **PLC Bus** to display a graphical representation of the hardware rack: • The selected M580 CPU is in the second position. • A default power supply appears in the first position. • Control Expert automatically opens the **Hardware Catalog** that corresponds to the **PLC bus** tab.
2	Select the power supply automatically added to the **PLC bus**.

Step	Action
3	Press the **Delete** key to remove the power supply.
4	Double-click the first slot of the **PLC bus** to open the **New Device** list.
5	Double-click the preferred power supply to make it appear in the **PLC bus**.
6	**File > Save**Click to save your project.

Improving the Security of a Project in Control Expert

Creating an Application Password

In Control Expert, create a password to help protect your application from unwanted modifications. The password is stored encrypted in the application. Any time the application is modified, the password is required.

In addition to the password protection, you can encrypt the application files (`.STU`, `.STA` and `.ZEF`).

The file encryption option is protected by a password mechanism:

Step	Action
1	In the **Project Browser** window, right-click **Project > Properties**.
2	In the **Properties of Project** window, click the **Project & Controller Protection** tab.
3	In the **Application** field, click **Change password**.
4	In the **Modify Password** window, enter a password in the **Entry** and **Confirmation** fields.
5	Click **OK**.
6	Select the **Auto-lock** check box if you want to require the password to resume the application display. You may also click the up/down arrows to set the number of minutes after which time the application auto-locks.
7	In addition, you can select the **File encryption active** check box if you want to encrypt the application files. **Result**: The **Create Password** window appears.
8	Enter a password in the **Entry** and **Confirmation** fields. Click **OK** to confirm.
9	To validate the changes: • Click **Apply** so that the **Properties of Project** window remains open. – or – • Click **OK** to close the window.
10	Click **File > Save** to save your application.

NOTE: If you forget your password, contact your local Schneider Electric service representative.

More information about application password is given in Application Protection (see EcoStruxure™ Control Expert, Operating Modes).

NOTE: When you export an unencrypted project to an `.XEF` or `.ZEF` file, the application password is removed.

NOTE: As of controller firmware version 4.10, you can no longer access controller functionality in any mode without the appropriate password.

You can help limit remote access to your application and data, regardless of password authentication, by following the **Memory Protect** instructions (detailed hereafter).

Using Memory Protect

In Control Expert, select the **Memory Protect** option to help protect your application from remote modifications, even if the remote user has the correct password. You accomplish this by configuring a dedicated, physical input that, when TRUE, restricts any remote access.

Step	Action
1	In the **Project Browser** window, expand the **Configuration** folder to display the controller.
2	To open the controller configuration window: • Double-click the controller. – or – • Right-click **BMEP58•0•0 > Open**.
3	In the controller window, click the **Configuration** tab.
4	Select the **Memory protect** check box, and enter an input address of your choice - but not from a safety module.
5	Click **File > Save** to save your application.

NOTE: Memory protect is not available for Hot Standby controllers.

Configuring the Size and Location of Inputs and Outputs

Introduction

In the Control Expert **Project Browser**, double-click **PLC Bus** to display the main rack. Then click on the CPU (but not on the Ethernet connectors) to open the CPU configuration window.

Setting Global Addresses and Operating Mode Parameters

Click on the **Configuration** tab to edit the size and starting positions of inputs and outputs:

Step	Action	
1	Double-click the image of the M580 CPU in the **PLC Bus** to view its properties.	
2	Select the **Configuration** tab.	
3	In the **Operating mode** area, select the boxes to enable the following parameters in your application:	
4	**Run/Stop input** **Run/Stop by input only**	Select **Run/Stop input** then enter a discrete input address of your choice – but not from a Safety module. Use these two parameters to place the PAC into Run or Stop mode. For more information regarding the effect of these parameters, refer to the topic Managing Run/Stop Input, page 515. (default = de-selected)
	Memory protect	Select **Memory protect** then enter a discrete input address of your choice – but not from a Safety module. This function is activated by an input bit. It prohibits the transfer of a project into the PAC and modifications in online mode, regardless of the communication channel. The Run and Stop commands are authorized. (default = de-selected)
	Maintenance authorization	Select **Maintenance authorization** then enter a discrete input address of your choice – but not from a Safety module. This selection is available only for Safety controllers and disallows setting the Maintenance mode of the Safety controller from Control Expert if not allowed by the discrete input.
	Automatic start in Run	The enabling of this option automatically places the PAC into RUN mode in the event of a cold start. (default = de-selected)
	Initialize %MWi on cold start	On a cold start, page 519 or on download if you select this parameter (default state): • The %MWi and %SWi are handled like other global variables (initialized to 0 or initial value, according to the application) in all cold start cases. On cold start or on download if you de-select this parameter:

Step	Action	
		• For %MWi: ◦ If %MW were previously saved in internal flash memory (using the %SW96 word) they are restored from internal flash memory, ◦ If not: – If cold start is linked to a power-off or of a push on the reset button, the %MW are initialized. – If not, the values of %MW are maintained. **NOTE:** if the new (or restored) application has more %MW than the previous one, the added %MW are set to 0 (non-zero initial values are not applied) • For %SWi, you will not be able to use %SW139 and %SW141 to create a Modbus mapping offset. Any offset values input to these system words will not be effective without initializing a value.
	Cold Start Only	If selected, this option forces the cold start , page 520of the application, instead of the normal warm start. By default, the Cold Start Only option is unchecked. An application using this function is not: • Downloadable to a PAC with a previous version. • Executable on a PAC with a previous version.
5	The option **Support Quantum remote drops** is only available for BMEP584040, BMEP585040(C), BMEP586040, BMEH584040, and BMEH586040. By default, this option is checked (allowing usage of Quantum remote drops) and the percentage of memory usage is displayed (bar graph). **NOTE:** The limitation of state ram depends on the Quantum memory structure. When unchecked, adding Quantum drops in the configuration is not allowed. Also, unchecking this option is not possible, if there is at least one Quantum drop in your configuration.	
6	Configure the size of the memory locations in the **Size of global address fields**. **NOTE:** High end standalone and Hot Standby CPUs (BMEP584040, BMEP585040(C), BMEP586040, BMEH584040 and BMEH586040) include State RAM memory management for Quantum Ethernet RIO drops. The State RAM feature supports LL984 logic sections for converted LL984 applications. The following memory management options are presented in the **Configuration** tab:	
	Mem usage	The percentage of CPU memory usage (bar graph), based on the cumulative values input into the %M, %MW, %I, and %IW fields, below. (Supported only by high end standalone and Hot Standby CPUs that support State RAM. For these CPUs, the option **Support Quantum remote drops** has to be checked previously). **NOTE:** Input values so that the percentage of CPU memory usage does not exceed 100%.
	%M-0x **%MW-4x** **%I-1x** **%IW-3x** **%KW**	Enter the appropriate value for each address field type. (%I and %IW are supported only by high end standalone and Hot Standby CPUs that support State RAM.) **NOTE:** The values for %IW and %MW, have to be divisible by 8 for version before 2.30 and divisible by 128 for other versions. The value for %KW have to be divisible by 8 for all versions.
	Viewer	Opens the **State RAM Viewer**, which displays the allocation of used memory.

Step	Action
	NOTE: To input: • Maximum values: Click the **Maximum values** button, select the appropriate boxes in the **Max** column, then click **OK**. • Default values: Click the **Default values** button, select the appropriate boxes in the **Default** column, then click **OK**. **NOTE:** M580 / S908 applications: In M580 CPUs that are compatible with Quantum S908 network adapter (see Modicon Quantum 140CRA31908, Adapter Module, Installation and Configuration Guide) and an OS version ≥ 02.30: (number of %I + number of %M) ≤ 65535. The maximum number of %I is 65280. The maximum number of %M is 65280.
7	Select the **Online modification in RUN or STOP** check box (in the **Configuration Online Modification** field) to use the change configuration on the fly (CCOTF) feature.
8	Select **Edit > Validate** (or click the  toolbar button) to save the configuration.

NOTE:

- After you validate module settings for the first time, you cannot edit the module name. If you subsequently decide to change the module name, delete the existing module from the configuration, then add and rename a replacement module.
- In addition to the Configuration tab, described above, the CPU configuration window presents an **I/O Objects** tab, and an **Animation** tab with three sub-tabs: Task, Real-time Clock, and Information.

M580 State RAM without Quantum Remote Drop Configured

These tables gives the default and maximum values of memory objects for M580 CPU that do not support Quantum drops or if the option **Support Quantum remote drops** is not checked.

Reference	%M		%I		Limit for %M + %I
	Default	Maximum	Default	Maximum	
BMEP581020(H)	512	32634	512	32634	≤32634
BMEP582020(H)	512	32634	512	32634	≤32634
BMEP582040(H)	512	32634	512	32634	≤32634
BMEH582040(C)	512	32634	512	32634	≤32634
BMEP583020	512	32634	512	32634	≤32634
BMEP583040	512	32634	512	32634	≤32634
BMEP584020	512	32634	512	32634	≤32634

Reference	%M		%I		Limit for %M + %I
	Default	Maximum	Default	Maximum	
BMEP584040	512	65280	512	65280	≤65280
BMEH584040(C)	512	65280	512	65280	≤65280
BMEP585040(C)	512	65280	512	65280	≤65280
BMEP586040(C)	512	65280	512	65280	≤65280
BMEH586040(C)	512	65280	512	65280	≤65280

Reference	%MW		%IW		Limit for %MW + %IW
	Default	Maximum	Default	Maximum	
BMEP581020(H)	1024	32464	1024	32464	≤32464
BMEP582020(H)	1024	32464	1024	32464	≤32464
BMEP582040(H)	1024	32464	1024	32464	≤32464
BMEH582040(C)	1024	32464	1024	32464	≤32464
BMEP583020	2048	65232	2048	65232	≤65232
BMEP583040	2048	65232	2048	65232	≤65232
BMEP584020	2048	65232	2048	65232	≤65232
BMEP584040	2048	65232	2048	65232	≤65232
BMEH584040(C)	2048	65232	2048	65232	≤65232
BMEP585040(C)	2048	65232	2048	65232	≤65232
BMEP586040(C)	2048	65232	2048	65232	≤65232
BMEH586040(C)	2048	65232	2048	65232	≤65232

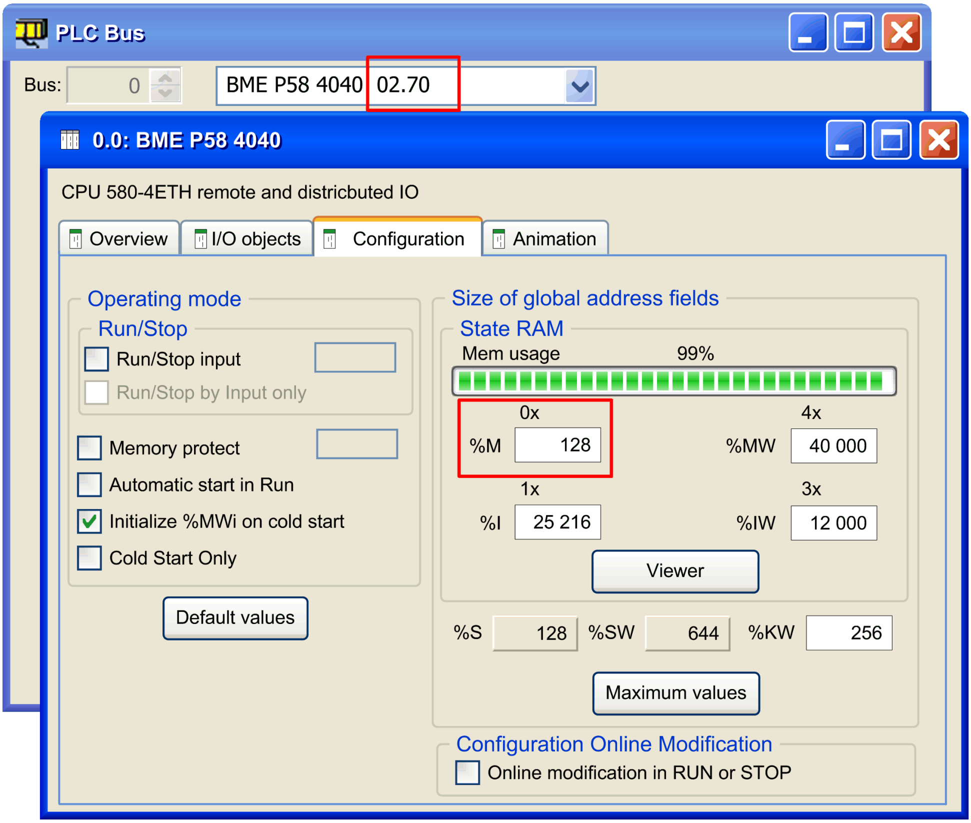
M580 State RAM with Quantum Remote Drops Configured

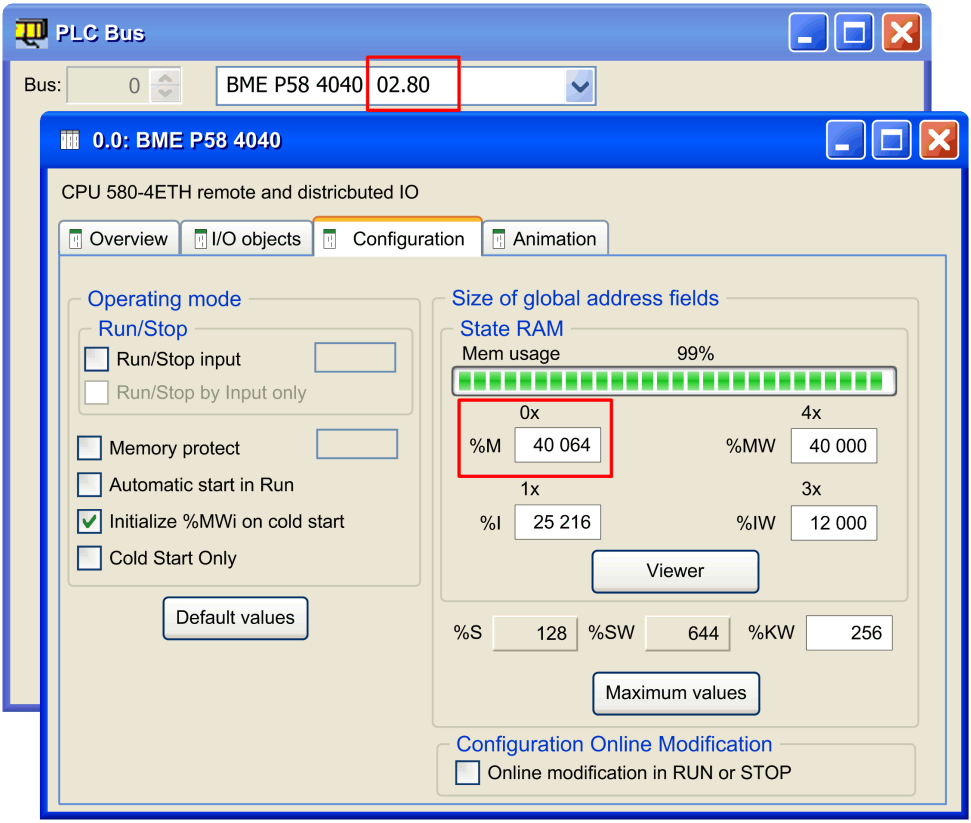
On M580 CPU SV 2.70 (or earlier), each %I and %M objects takes around 1 byte.

On M580 CPU SV 2.80 (or any subsequent supporting version(s)) the space taken by each %I or %M is optimized and the state RAM can now be filled with a larger number of objects.

When Quantum Ethernet Remote drops are configured on M580 CPU SV 2.80 (or any subsequent supporting version(s)), the total size of the state RAM is unchanged (128 Kbytes), but you can assign a larger number of %M and %I.

Example: with numbers of %IW = 12 000, %MW = 40 000, and %I = 25216, the maximum number of %M is 128 on CPU SV 2.70 while it is 40 064 on CPU SV 2.80.


PLC Bus
Bus:
0
BME P58 4040
02.70
0.0: BME P58 4040
CPU 580-4ETH remote and districbuted IO
Overview
I/O objects
Configuration
Animation
Operating mode
Run/Stop
Run/Stop input
Run/Stop by Input only
Memory protect
Automatic start in Run
Initialize %MWi on cold start
Cold Start Only
Default values
Size of global address fields
State RAM
Mem usage
99%
0x
%M
128
4x
%MW
40 000
1x
%I
25 216
3x
%IW
12 000
Viewer
%S
128
%SW
644
%KW
256
Maximum values
Configuration Online Modification
Online modification in RUN or STOP



Completing the Ethernet Network Configuration

After you configure these settings, configure the CPU settings beginning with its Channel Properties. Then configure the Ethernet network devices.

Protecting Located Data in Monitoring Mode

Introduction

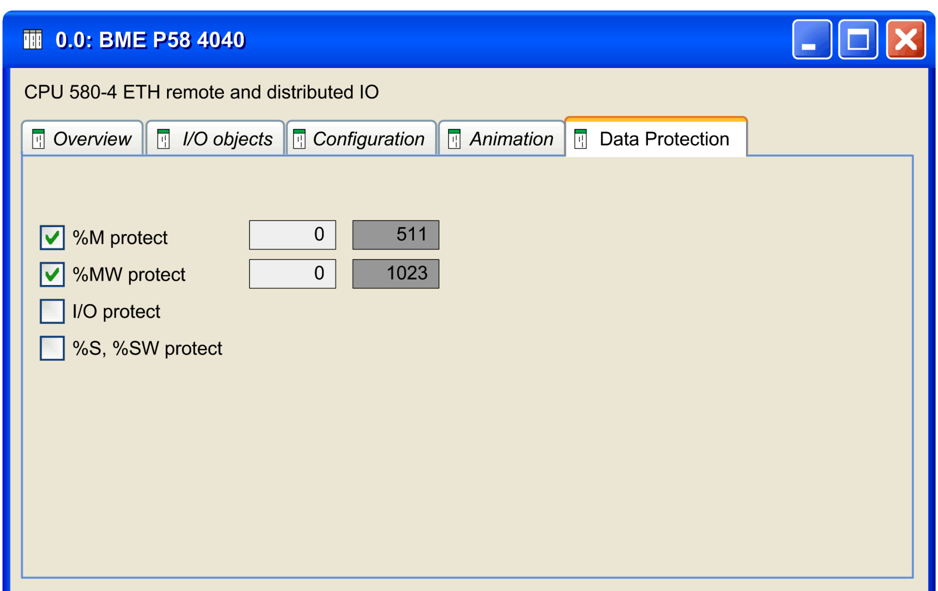
Before any action on the data memory protection, you must activate this feature in your project settings.

In the Control Expert main window, click **Tools > Project Setting > PLC embedded data**. Then select the **Data memory protect** box and click **Apply**.

The data memory protection feature is supported by M580 CPU with the firmware V3.20 or any subsequent supporting version(s). For details, refer to the chapter *Data Memory Protection* (see EcoStruxure™ Control Expert, Operating Modes).

Procedure of Protecting Located Data

Follow the procedure below to define the located data to protect:

Step	Action
1	In the Control Expert **Project Browser**, double-click **PLC Bus** to display the main rack. Then double-click on the M580 CPU (but not on the Ethernet connectors) to view its properties.
2	Select the **Data Protection** tab. 
3	Select the boxes to enable the data protection:

Step	Action	
	%M protect	The protected area is always located at the end of the %M area. Only the starting address of the protected area can be set. The end address of the protected area is not configurable (grayed). The end address of the protected area equals to n-1 where n is the number of available %M defined by the PLC abilities and set in the **Configuration** tab. If **%M protect** is selected, you can enter the starting address or the %M data to protect. By default, the starting address is 0. Unchecking the %M protection reset the starting address.
	%MW protect	The protected area is always located at the end of the %M area. Only the starting address of the protected area can be set. The end address of the protected area is not configurable (grayed). The end address of the protected area equals to n-1 where n is the number of available %MW defined by the PLC abilities and set in the **Configuration** tab. If **%MW protect** is selected, you can enter the starting address or the %M data to protect. By default, the starting address is 0. Unchecking the %MW protection reset the starting address. **NOTE:** Array variables which are mapped on a %MW range must be entirely inside or entirely outside of the protected %MW range.
	I/O protect	If selected, all I/O objects (including DTM-objects) are protected. **NOTE:** except state Ram objects.
	%S, %SW protect	If selected, all system bits and system words are protected.
4	Select **Edit > Validate** (or click the toolbar button) to save the configuration.	

Project Management

Downloading the Application to the CPU

Download the Control Expert application to the CPU through one of its ports or through a connection to an Ethernet communication module:

Method	Connection
USB port	If the CPU and the PC that are running Control Expert both have USB ports, you can download the application to the CPU directly through the USB ports, page 71 (version 1.1 or later).
Ethernet port	If the CPU and the PC that are running Control Expert both have Ethernet ports, you can download the application to the CPU directly through the Ethernet ports.
communication module	You can download the application to the CPU by connecting Control Expert to the IP address of a communication module.

NOTE: For details, refer to *Downloading CPU Applications* (see Modicon M580 Standalone, System Planning Guide for, Frequently Used Architectures) in the *Modicon M580 Hot Standby System Planning Guide for Frequently Used Architectures*.

Converting Legacy Applications to M580

For details on this conversion process, contact your Schneider Electric customer support.

Restoring and Backing Up Projects

The CPU application RAM, page 513 and the CPU flash memory automatically and manually perform the following:

- Restore a project in the CPU from the flash memory (and the memory card if inserted):
  - Automatically after a power cycle
  - Automatically on a warm restart
  - Automatically on a cold start
  - Manually with a Control Expert command: **PLC > Project Backup > Backup Restore**

    NOTE: If a memory card is inserted with a different application than the application in the CPU, the application is transferred from the memory card to the CPU application RAM when the restore function is carried out. If this is done unintentionally, the previous settings – including IP address and FDR obtained settings – will be overwritten and lost.

- Save the CPU project in the flash memory (and the memory card if inserted):
  - Automatically after an online modification is performed in the application RAM
  - Automatically after a download
  - Automatically on detection of `%S66` system bit rising edge
  - Manually with a Control Expert command: **PLC > Project Backup > Backup Save**

    NOTE: Backup begins after the completion of the current MAST cycle and before the start of the next MAST cycle.

    If MAST is configured as periodic, set the MAST period to a value larger than the actual MAST execution time. This lets the processor complete an entire backup without interruption.

    If the MAST period is set to a value less than the actual MAST execution time, backup processing is fragmented and requires a longer time to finish.
- Compare the CPU project and the flash memory project:
  - Manually with a Control Expert command: **PLC > Project Backup > Backup Compare**

    NOTE: When a valid memory card is inserted, page 78 with a valid application, the application backup and restore operations are performed as follows:

    - The application backup is performed on the memory card first and then on the flash memory.
    - The application restore is performed from the memory card to the CPU application RAM first and then copied from the application RAM to the flash memory.

DIO Scanner Functionality

Introduction

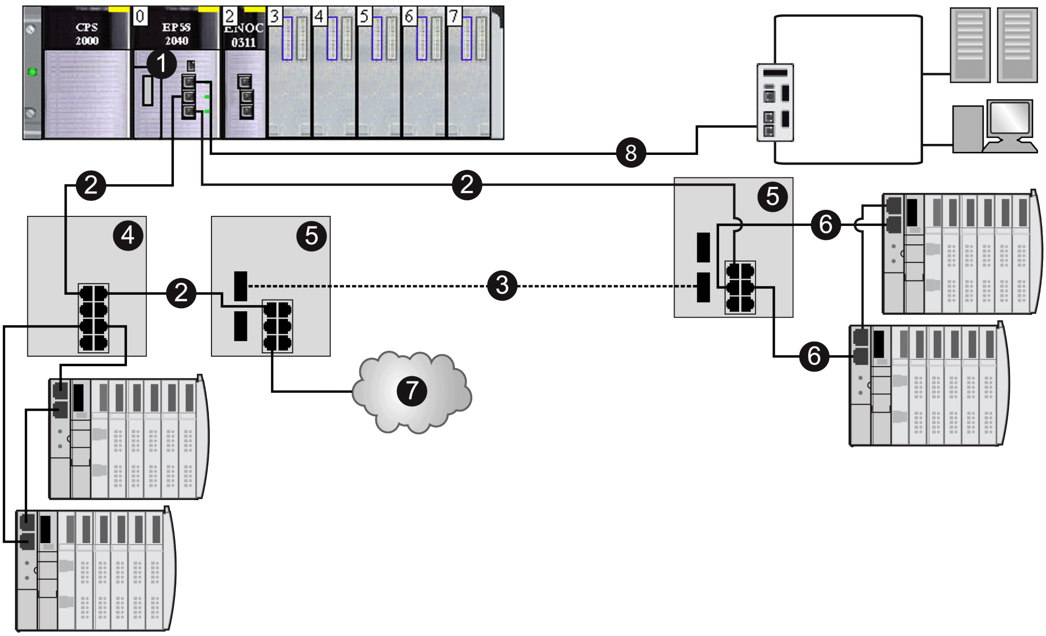
An embedded DIO scanner service in a standalone (BMEP58•0•0) or Hot Standby (BMEH58•0•0) M580 CPU can manage distributed equipment. Through this service, Ethernet gateway devices (like Profibus and CANopen masters) can operate as distributed equipment.

All DIO scanning communications occur over the Ethernet backplane or through an Ethernet port.

NOTE: The BMEP58•040 CPUs also manage RIO modules through the RIO scanner service, but this discussion applies to the DIO scanner service.

DIO Scanner Service Overview

In this network example, the CPU is connected to the DIO network (2) and the control network (8).



1 a CPU with an embedded DIO scanner service

2 copper portion of the main ring

3 fiber portion of the main ring

4 DRS connecting a DIO sub-ring to the main ring

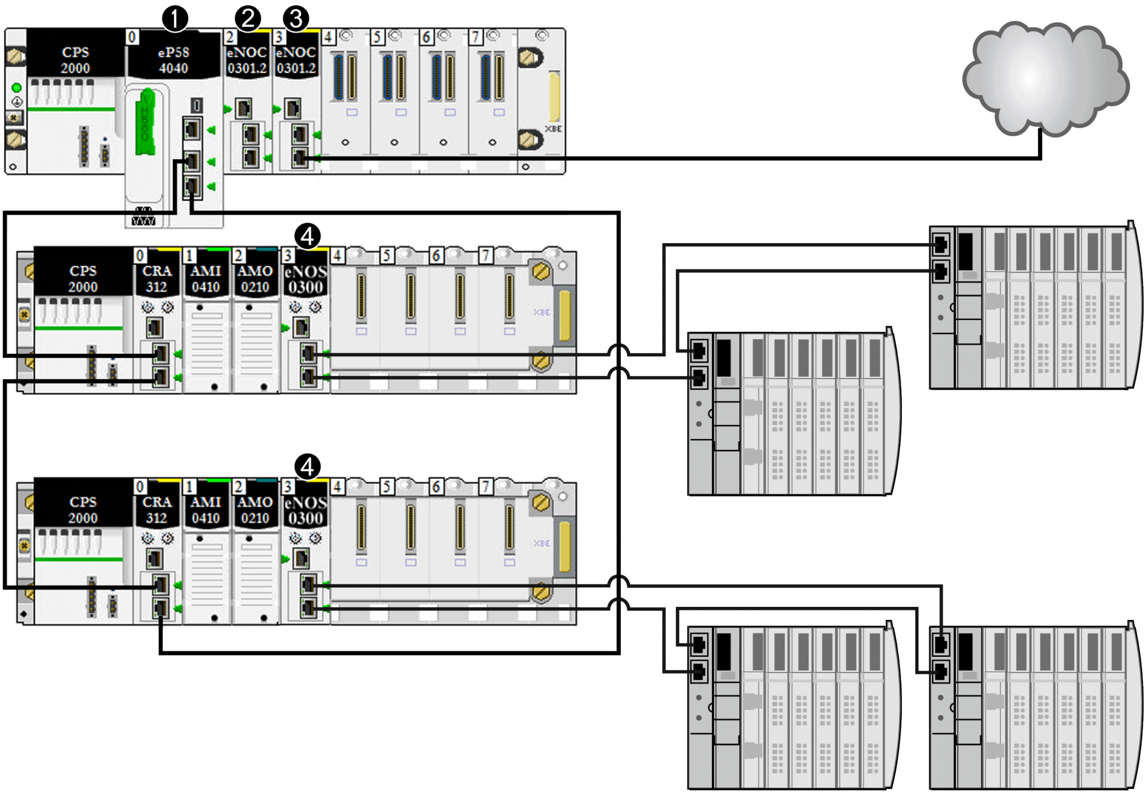
5 DRS configured for copper-to-fiber and fiber-to-copper transition connecting a DIO sub-ring to the main ring

6 DIO sub-ring

7 DIO cloud

8 CPU connecting the control network to the M580 system

This illustration shows direct connections to distributed equipment:



1 A CPU on the main rack runs the Ethernet I/O communication server service.

2 A BMENOC0301/BMENOC0311 Ethernet communication module (Ethernet backplane connection disabled) manages distributed equipment on the device network.

3 A BMENOC0301/BMENOC0311 Ethernet communication module (Ethernet backplane connection enabled) is connected to a DIO cloud.

4 A BMENOS0300 network option switch module is connected to a DIO sub-ring.

Configuring the CPU with Control Expert

Introduction

Use the instructions in this section to configure the M580 CPU in Control Expert.

NOTE: Some configuration features for the M580 CPU are accessed through the Control Expert **DTM Browser**. Those configuration instructions appear elsewhere in this document, page 167.

Control Expert Configuration Tabs

Accessing the Control Expert Configuration Tabs

Access the controller configuration parameters for RIO and distributed equipment:

Step	Action
1	Open a project that includes an M580 controller that supports RIO and DIO networks.
2	In the **Project Browser**, double-click **Project > Configuration > PLC bus**.
3	In the **PLC bus** dialog box, double-click the drawing with 3 Ethernet ports in the middle of the controller.
4	In the **Security** tab, check to see that the services that you require are enabled.(See the Note below.)
5	In the **IPConfig** tab, you may change the IP address of the controller or you may configure the default address, which starts with 10.10 and uses the last two bytes of the MAC address. **NOTE:** When the last two octets of the MAC address (*MAC5.MAC6*) correspond to *0.0* in the default address, make a point-to-point cable connection between your computer and the controller, communication module, or other module..

NOTE: For improved security, some of the communication services (FTP, TFTP, and HTTPS) are disabled by default. You may wish to perform some actions (such as a firmware update, web access, or remote I/O) that require the availability of one or more of these services. Before configuring Ethernet parameters, set the security levels, page 142 to meet your requirements. When these services are not needed, you should disable them.

Control Expert Configuration Tabs

This table indicates the Control Expert configuration tabs that are available (X) and unavailable (—) for M580 controllers:

Control Expert Tab	Services	
	Controllers with Embedded RIO Scanning (BME•58•040)	Controllers without Embedded RIO Scanning (BME•58•020)
Security	X	X
IPConfig	X	X
RSTP	X	X
SNMP	X	X
NTP	X	X
Switch	—	X
QoS	—	X
Service Port	X	X
Advanced Settings	—	X
Safety	—[1]	—
1. The Safety tab applies only to M580 safety standalone controllers.		

NOTE: To maintain RIO performance, you cannot access these tabs for BME•58•040 controllers.

About Control Expert Configuration

Accessing Configuration Settings

Access the configuration settings for the M580 CPU in Control Expert:

Step	Action
1	Open Control Expert.
2	Open a Control Expert project that includes an M580 CPU in the configuration.
3	Open the **Project Browser** (**Tools > Project Browser**).
4	Double-click **PLC bus** in the **Project Browser**.
5	In the virtual rack, double-click the Ethernet ports of the M580 CPU to see these configuration tabs: • **Security** • **IPConfig** • **RSTP** • **SNMP** • **NTP** • **Switch** (See note 1.) • **QoS** (See note 1.) • **Service Port** • **Advanced Settings** (See note 1.) • **Safety** (See note 2.) These configuration tabs are described in detail in the pages that follow. **NOTE:** 1. This tab is not available for CPUs that provide the RIO Ethernet scanning services. 2. This tab applies only to standalone M580 safety CPUs.

Security Tab

Introduction

Control Expert provides security services for the controller. Enable and disable these services on the **Security** tab in Control Expert.

Accessing the Security Tab

View the **Security** configuration options:

Step	Action
1	Open your Control Expert project.
2	Double-click the Ethernet ports on the controller in the local backplane or right-click the Ethernet ports and select **Open Submodule**.
3	Select the **Security** tab in the **RIO DIO Communicator Head** window to enable/disable Ethernet services.

Available Ethernet Services

You can enable or disable these Ethernet services:

Field	Comment
Enforce Security	Click the **Enforce Security** button to execute these functions: • Enable **Access Control**. • Disable **FTP**, **TFTP**, **HTTP**, **EIP**, **SNMP**, and **DHCP/BOOTP**. **NOTE:** From version 4.10, HTTPS replaces HTTP. HTTPS is not affected when the **Enforce Security** button is selected. **NOTE:** You can set each field individually once the global setting is applied.
Unlock Security	Click the **Unlock Security** button to execute these functions: • Enable **TFTP**, **HTTP**, **EIP**, **SNMP**, and **DHCP/BOOTP**. • Disable **Access Control**. **NOTE:** From version 4.10, HTTPS replaces HTTP. HTTPS is not affected when the **Unlock Security** button is selected. **NOTE:** You can set each field individually once the global setting is applied.
FTP	Enable or disable (default) firmware upgrade, SD memory card data remote access, data storage remote access, and device configuration management using the FDR service. **NOTE:** Local data storage remains operational, but remote access to data storage is disabled.

Field	Comment
TFTP	Enable or disable (default) the ability to read RIO drop configuration and device configuration management using the FDR service. **NOTE:** Enable this service to use eX80 Ethernet adapter modules.
HTTPS	Enable or disable (default) the web access service.
DHCP / **BOOTP**	Enable or disable (default) the automatic assignment of IP addressing settings. For DHCP, also enable/disable automatic assignment of subnet mask, gateway IP address, and DNS server names.
SNMP	Enable or disable (default) the protocol used to monitor the device.
EIP	Enable or disable (default) access to the EtherNet/IP server.
Engineering Link Mode	Depending on the level of targeted cybersecurity, you can select one of the following three **Engineering Link Modes**: • **Full Access** The controller behaves as in previous firmware versions. Secure and non-secure communications are accepted. For Control Expert communication, the controller accepts the **Modbus TCP** and **Modbus TCP via USB** non-secure drivers or the **HTTPS** and **HTTPS via USB** secure drivers. For SCADA or controller-to-controller communication, **Modbus TCP** (port 502) is accepted. • **Filtered** (default) Use this hybrid mode to apply cybersecurity on the engineering link and non-secure connectivity on links to SCADA or other controllers. For Control Expert communication, the controller accepts **HTTPS** and **HTTPS via USB** secure drivers. For SCADA or controller-to-controller communication, **Modbus TCP** (port 502) or **UMAS** (OFS) are accepted. **NOTE:** In **Filtered** mode, the controller accepts the **Modbus TCP** and **Modbus TCP via USB** non-secure drivers but only with **Connection mode** set to **monitoring** in the options of the project. Monitoring mode is a read-only mode, in which it is not possible to download an application to the controller or stop the controller. • **Enforced** This mode provides the highest level of security. Only secure protocols are accepted by the controller. For Control Expert communication, the controller accepts only the **HTTPS** and **HTTPS via USB** secure drivers. For SCADA or controller-to-controller communication, **Modbus TCP** (port 502) or **UMAS** (OFS) are **NOT** accepted. **NOTE:** The **Engineering Link Mode** is available only for M580 controllers with firmware as of version 4.20 (or subsequent supporting versions) when the **HTTPS** service is enabled. Refer to the detailed description of **Engineering Link Mode**, page 147.
Access Control	Enable (default) or disable Ethernet access to the multiple servers in the controller from unauthorized network devices.

Field	Comment
Authorized addresses(1)	• **Subnet** (**Yes** or **No**) • **IP Address:** 0.0.0.0 ... 223.255.255.255 • **Subnet mask:** 224.0.0.0 ... 255.255.255.252 • **FTP:** Grant access to the FTP server in the controller. • **TFTP:** Grant access to the TFTP server in the controller. • **HTTPS:** Grant access to the HTTP secured server in the controller. • **Port 502:** Grant access to port 502 (typically used for Modbus messaging) of the controller. • **EIP:** Grant access to the EtherNet/IP server in the controller. • **SNMP:** Grant access to the SNMP agent resident in the controller.
1 Set **Access Control** to **Enabled** to modify this field.	

NOTE: Refer to the ETH_PORT_CTRL topic, page 564 for information regarding using this function block to control the FTP, TFTP, HTTPS, and DHCP/BOOTP protocols.

Enable/Disable Ethernet Services

You can enable or disable Ethernet services on the **Security** tab:

- Enable/disable FTP, TFTP, HTTPS, EIP, SNMP, and DHCP/BOOTP for all IP addresses. (You can use this feature offline only. The configuration screen is grayed out in online mode.)

  – or –
- Enable/disable FTP, TFTP, HTTPS, Port 502, EIP, and SNMP for each authorized IP address. (You can use this feature online.)

Set the **Security** tab parameters before you download the application to the controller. The default settings (maximum security level) reduce the communication capacities and port access.

NOTE: Disable services that are not being used.

Using Access Control for Authorized Addresses

Use the **Access Control** area to restrict device access to the controller in its role as a server. After you enable access control in the **Security** dialog box, you can add the IP addresses of the devices that you want to communicate with the controller to the list of **Authorized Addresses**:

- By default, the IP address of the controller embedded Ethernet I/O scanner service with **Subnet** set to **Yes** allows any device in the subnet to communicate with the controller through EtherNet/IP or Modbus TCP.

- Add the IP address of any client device that may send a request to the controller Ethernet I/O scanner service, which, in this case, acts as a Modbus TCP or EtherNet/IP server.
- Add the IP address of your maintenance PC to communicate with the controller through the controller Ethernet I/O scanner service via Control Expert to configure and diagnose your application.
- If the controller is configured as a network time service client in the NTP tab, page 155, add the IP address of the network time server (or servers, if more than one server). This is the same IP address that was added to the list of **Server IP addresses** in the NTP tab.

  NOTE: The subnet in the **IP Address** column can be the subnet itself or any IP address inside the subnet. If you select **Yes** for a subnet that does not have a subnet mask, a pop-up window states that the screen cannot be validated because of a detected error.

You can enter a maximum of 127 authorized IP addresses or subnets.

Adding Devices to the Authorized Addresses List

To add devices to the **Authorized Addresses** list:

Step	Action
1	Set **Access Control** to **Enabled**.
2	In the **IP Address** column of the **Authorized Addresses** list, enter an IP address. Enter the address of the device to access the controller Ethernet I/O scanner service with either of these methods: • *Add a single IP address*: Enter the IP address of the device and select **No** in the **Subnet** column. • *Add a subnet*: Enter a subnet address in the **IP Address** column. Select **Yes** in the **Subnet** column. Enter a subnet mask in the **Subnet Mask** column. **NOTE:** • The subnet in the **IP Address** column can be the subnet itself or any IP address in the subnet. If you enter a subnet without a subnet mask, an on-screen message states that the screen cannot be validated. • A red exclamation point indicates a detected error in the entry. You can save the configuration only after the detected error is corrected.
3	Select one or more of the following methods of access you are granting the device or subnet: **FTP**, **TFTP**, **HTTP** , **HTTPS** if available, **Port 502**, **EIP**, **SNMP**.
4	Repeat steps 2 and 3 for each additional device or subnet to which you want to grant access to the controller Ethernet I/O scanner service. **NOTE:** You can enter up to 127 authorized IP addresses or subnets.
5	Click **Apply**.

Removing Devices from the Authorized Addresses List

To remove devices from the **Authorized Addresses** list:

Step	Action
1	In the **Authorized Addresses** list, select the IP address of the device to delete.
2	Press the **Delete** button.
3	Click **Apply**.

Engineering Link Mode

Overview

The engineering link mode lets you restrict the connection to M580 controllers to certain protocols to help secure communications with Control Expert, Control Expert Classic, and HMI panels/software. It is available for M580 controllers with firmware version 4.20 and subsequent supporting versions.

M580 Controller Connection Settings

The engineering link mode that is selected in the **Security** tab defines the protocols and modes to access the control project in online mode as described in the following table.

<table>
<tr><td></td><td></td><td>Secure engineering link(1)</td><td colspan="2">Engineering link(1)</td><td>HMI/SCADA(2)</td></tr>
<tr><td></td><td>Protocols</td><td>HTTPS or HTTPS via USB</td><td colspan="2">Modbus TCP or Modbus TCP via USB</td><td>Modbus TCP</td></tr>
<tr><td></td><td>Connection mode</td><td>Monitoring and programming modes</td><td>Programming mode</td><td>Monitoring mode</td><td>–</td></tr>
<tr><td rowspan="3">Engineering Link Mode</td><td>Full Access</td><td>Yes</td><td>Yes</td><td>Yes</td><td>Yes</td></tr>
<tr><td>Filtered</td><td>Yes</td><td>No</td><td>Yes</td><td>Yes</td></tr>
<tr><td>Enforced</td><td>Yes</td><td>No</td><td>No</td><td>No</td></tr>
<tr><td colspan="6">(1) Connection with Control Expert or Control Expert Classic.
(2) Connection with HMI panels/software.</td></tr>
</table>

NOTE: For M580 Safety controllers, the SAFE peer-to-peer communication does not work if the Engineering Link Mode is set to "Enforced" on the receiver CPU.

NOTE: For more information on drivers/protocols, refer to the topic describing the types of connections with controllers (see *EcoStruxure Control Expert, Operating Modes*)

NOTE: For more information on the programming and monitoring modes, refer to Services in Online Mode (see *EcoStruxure Control Expert, Operating Modes*).

IPConfig Tab

IPConfig Parameters

IP address configuration field on the **IPConfig** tab:

Parameter	Default Value	Description
Main IP address	192.168.10.1	The IP address of the controller and DIO scanner. This address can be used: • By Control Expert, an HMI, or SCADA to communicate with the controller. • To access the controller web pages. • By the controller to perform I/O scanning of DIO devices.
IP address A	192.168.11.1	This address applies to the RIO scanner service in the controller designated as **A**. (See the note below.)
IP address B	–	For M580 Hot Standby controllers only, this address applies to the RIO scanner service in the controller designated as **B**. (See the note below.)
Subnetwork mask	255.255.0.0	This bit mask identifies or determines the IP address bits that correspond to the network address and the subnetwork portion of the address. (The value can be changed to any valid value in the subnetwork.)
Gateway address	192.168.10.1	This is the IP address of the default gateway to which messages for other networks are transmitted.
NOTE: • If you change **IP address A**, the system may recalculate all IP addresses (including those of the drops) to keep all devices in the same subnetwork. • In M580 Hot Standby systems, both controller **A** and controller **B** maintain a redundant owner connection with each RIO device (BM•CRA312•0 adapter). For this reason, when a Hot Standby switchover occurs, the state of RIO outputs is not affected – the Hot Standby switchover transition is transparent.		

Viewing and Editing the IP Address and Device Name of Network Devices

The **CRA IP address configuration** area on the **IPConfig** tab is provided for controllers with Ethernet I/O scanner service (controllers with commercial references that end *40*). Use this area to display a list of RIO/DIO scanners and BM•CRA312•0 adapters, and view or edit the device IP address and device Identifier:

Step	Action
1	Click the **CRA IP address configuration** link to open the **Ethernet Network** window.
2	In the **Subtype** header, filter the device list by selecting: • **Scanner RIO/DIO** • **CRA** • **...** (select both) This list applies the selected filter, and displays all detected network devices of the selected type(s).
3	The **IP Address** field displays the address that was automatically assigned when the device was added to the network. **NOTE:** Although the IP address is editable, accept the automatically assigned IP address.
4	The **Identifier** field displays the identifier for the module, which is also the **Device Name**. To edit the **Identifier** setting: 1. Double-click on the **Identifier** value. The value becomes editable. 2. Type in a new value. 3. Click the Control Expert **Validate** button. The new **Identifier** setting is applied.

NOTE: The other fields in the **Ethernet Network** window are read-only.

Advanced Configuration

To configure DHCP and FDR services in the DTM browser, click the **Services configuration link** in the **Advanced configuration** section of the window.

RSTP Tab

Introduction

The Ethernet DEVICE NETWORK ports on the front of the M580 CPU support *rapid spanning tree protocol* (RSTP). RSTP is an OSI layer 2 protocol defined by IEEE 802.1D 2004. RSTP performs these services:

- RSTP creates a loop-free logical network path for Ethernet devices that are part of a topology that includes redundant physical paths. When either DEVICE NETWORK port (ETH 2 or ETH 3) on the CPU is disconnected, the RSTP service directs traffic to the other port.
- RSTP automatically restores network communication by activating redundant links when a network event causes a loss of service.

NOTE: When an RSTP link is disconnected, the RSTP service acts on an event and forwards traffic through the correct port. During this re-connect time (50ms max), some packets may be lost.

The RSTP service creates a loop-free logical network path for Ethernet devices that are part of a topology that includes redundant physical paths. When the network experiences a loss of service, the RSTP-enabled module automatically restores network communication by activating redundant links.

NOTE: RSTP can be implemented only when all network switches are configured to support RSTP.

Changing these parameters can affect sub-ring diagnostics, RIO determinism, and network recovery times.

Assign the Bridge Priority for RIO/DIO Scanner Service

A **bridge priority** value is used to establish the relative position of a switch in the RSTP hierarchy. Bridge priority is a 2-byte value for the switch. The valid range is 0 ... 65535, with a default of 32768 (the midpoint).

Assign the **Bridge Priority** on the **RSTP** page:

Step	Action
1	Select **RSTP** to see the **RSTP Operational State**.
2	Select a **Bridge Priority** from the drop-down list in the **RSTP Operational State** area: • **Root (0)** (default) • **Backup Root (4096)** • **Participant (32768)**
3	Finish the configuration: • **OK**: Assign the **Bridge Priority**, and close the window. • **Apply**: Assign the **Bridge Priority**, and keep the window open.

RSTP Parameters for CPUs with RIO and DIO Scanner Service

RSTP tab:

Field	Parameter	Value	Comment
RSTP Operational State	**Bridge Priority**	**Root (0)**	default
		Backup Root (4096)	–
		Participant (32768)	–

RSTP Parameters for CPUs without RIO Scanner Service (DIO Scanner Service Only)

RSTP tab:

Field	Parameter	Value	Comment
RSTP Operational State	**Bridge Priority**	**Root(0)**	–
		Backup Root(4096)	–
		Participant(32768)	default
Bridge parameters	**Force version**	2	You cannot edit this value.
	Forward delay (ms)	21000	
	Maximum Age Time (ms)	40000	
	Transmit Hold Count	40	
	Hello Time (ms)	2000	

Field	Parameter	Value	Comment
Port 2 Parameters	–	–	You cannot edit these field parameters.
Port 3 Parameters	–	–	You cannot edit these field parameters.

SNMP Tab

Use the **SNMP** tab in Control Expert to configure individual SNMP parameters for these modules:

- M580 CPU modules
- (e)X80 EIO adapter modules on RIO drops
- 140CRA3120• RIO adapter modules in Quantum EIO systems

An SNMP agent is a software component of the SNMP service that runs on these modules to allow access to diagnostic and management information for the modules, as defined by the supported MIBs: MIB2, Bridge MIB, and LLDP MIB.

You can use SNMP browsers, network management software, and other tools to access this data. In addition, the SNMP agent can be configured with the IP addresses of one or two devices (typically PCs that run network management software) to be the targets of event-driven trap messages. Traps can inform the management device of the following events: Link up, Link down, Cold start, Warm start, and Authentication failure.

Use the **SNMP** tab to configure the SNMP agents for communication modules in the local rack and RIO drops. The SNMP agent can connect to and communicate with one or two SNMP managers as part of an SNMP service. The SNMP service includes:

- authentication checking by the Ethernet communication module, of any SNMP manager that sends SNMP requests
- management of events or traps

SNMP V1 and SNMP V3

M580 CPU modules with firmware version ≥ 4.01 and higher support both:

- SNMP V1.
- SNMP V3, with the *SNMPSecurityLevel* of *NoAuthNoPriv*.

M580 CPU modules with firmware version < 4.01 support only SNMP V1.

SNMP Parameters

View and edit these properties on the **SNMP** page:

Property		Description
SNMP Version	SNMP V1	SNMP V1 and SNMP V3 present different formats and configurable parameters, as indicated below.
	SNMP V3	

Property		Description
IP Address Managers [1, 3]	IP Address Manager 1	The IP address of the first SNMP manager to which the SNMP agent sends notices of traps.
	IP Address Manager 2	The IP address of the second SNMP manager to which the SNMP agent sends notices of traps.
Agent [1, 3]	Location	The device location (32 characters maximum).
	Contact	Information describing the person to contact for device maintenance (32 characters maximum)
	SNMP Manager	Select one: • **Disabled**: You can edit the Location and Contact settings on this page. • **Enabled**: You cannot edit the Location and Contact settings on this page. (Those settings are managed by the SNMP Manager.)
Community Names [1]	Get	Password required by the SNMP agent before executing read commands from an SNMP manager (default = **public**).
	Set	Password required by the SNMP agent before executing write commands from an SNMP manager (default = **private**).
	Trap	Password an SNMP manager requires from the SNMP agent before the manager accepts trap notices from the agent (default = **alert**).
Security [1]	Enable Authentication Failure Trap	**TRUE** causes the SNMP agent to send a trap notice to the SNMP manager if an unauthorized manager sends a Get or Set command to the agent (default = **Disabled**).
Username [3]		The username value required for SNMP V3 communication.
1. Supported by SNMP V1. 3. Supported by SNMP V3.		

Apply the configuration by clicking a button:

- **Apply**: Save changes.
- **OK**: Save changes and close the window.

NTP Tab

You can configure an M580 controller as an NTP server or an NTP client in the Control Expert NTP tab.

When the controller firmware version is:

- Earlier than V4.01, the SNTP protocol is employed and you can configure the controller as:
  - NTP client
  - NTP server
  - Both NTP client and server
- V4.01 or any subsequent supporting version(s), the NTPv4 protocol is employed and you can configure the controller as:
  - NTP server only
  - NTP server and client

To begin, open the controller configuration tabs in Control Expert, page 139.

NTP Service Features

The NTP service has these features:

- A periodic time correction is obtained from the reference-standard time server.
- There is an automatic switchover to a backup (secondary) time server if an error is detected with the normal time server system.
- Controller projects use a function block to read the accurate clock, allowing project events or variables to be time stamped. (Refer to the *System Time Stamping User Guide* (see System Time Stamping, User Guide) for detailed information about timestamping performance.)

  NOTE:

  When the M580 controller is configured as either an NTP server or as an NTP client, the BM•CRA312•0 (e)X80 EIO adapter modules are NTP clients of the controller:

  - When only BM•CRA31200 modules are configured as NTP clients, the accuracy of this server allows time discrimination of 20 ms.
  - All BM•CRA31200 modules in the network have the same client configuration.

NTP Client Mode

When the controller is configured as an NTP client, the network time service (SNTP or NTPv4) synchronizes the clock in the M580 controller to that of the time server. The synchronized value is used to update the clock in the controller. Typical time service configurations utilize redundant servers and diverse network paths to achieve high accuracy and reliability.

When the controller firmware version is:

- Earlier than V4.01, you can specify a primary and secondary NTP server.
- V4.01 and any subsequent supporting version(s), you can identify up to 8 NTP servers, and specify the preferred server.

  NOTE: When the controller operates as an network time service client, if you have enabled **Access Control** in the Security tab, page 142 you need to enter the network time server IP address in the access control list. Otherwise, the controller cannot reach the server.

To establish the accurate Ethernet system network time, the system performs the following at power up:

- requires the controller to boot
- uses the controller to obtain the time from the NTP server
- requires a predefined interval until time is accurate; your configuration determines how long before time is accurate
- may require several updates to achieve peak accuracy

Once an accurate time is received, the service sets the status in the associated time service register.

The time service clock value starts at 0 until fully updated from the controller.

Model	Starting Date
Modicon M580 with Control Expert	January 1st 1980 00:00:00.00

Stop or run controller:

- Stop and run have no effect on the accuracy of the clock.
- Stop and run have no effect on the update of the clock.
- A transition from one mode to the other has no effect on the accuracy of the Ethernet system network time.

Download application:

- The status clock value associated with the time service register in the M580 controller is reinitialized after an application is downloaded or after an NTP server swap. The time is accurate after two polling periods.

NOTE: For NTP diagnostics, refer to the NTP web page.

NTP Server Mode

When the controller is configured as an NTP server, it can synchronize client clocks (such as a BM•CRA31200 (e)X80 EIO adapter module). The controller's internal clock is then used as reference clock for NTP services.

NTP Parameters for a Controller with Firmware earlier than V4.01

Use the pull-down menu in the **NTP** field to configure the controller as an **NTP Client** or an **NTP Server**:

Value	Comment
Disabled	default: Both the NTP server and the NTP client services of the controller are disabled.
NTP Client	The controller functions as the NTP client. In this case, configure the **NTP Server Configuration** parameters. **NOTE:** Enable the NTP client here to automatically enable the NTP client service on all BM•CRA312•0 adapter modules.
NTP Server	The Ethernet I/O scanner controller acts as an NTP server. **NOTE:** Enable the NTP client here to automatically enable the NTP client service on all BM•CRA312•0 adapter modules and to configure the BM•CRA312•0 to use the controller as the NTP server.

Assign values to these parameters in the **NTP Server Configuration** field:

Parameter	Comment
Primary NTP Server IP address	the IP address of the NTP server, from which the controller first requests a time value
Secondary NTP Server IP address	the IP address of the backup NTP server, from which the controller requests a time value after not receiving a response from the primary NTP server
Polling Period	The time (in seconds) between updates from the NTP server. Smaller values typically result in better accuracy. **NOTE:** This parameter applies only to the SNTP protocol and to controllers using a firmware version earlier than V4.01.

NTP Parameters for a Controller with Firmware V4.01

Use the following settings to configure the NTP protocol for controller with firmware V4.01 or any subsequent supporting version(s):

Parameter	Description
Server Only / **Client Server**	Specify the NTP role of the controller: server only, or both client and server.
Stratum	The relative position of the server in the NTP network. This represents the distance of the controller (in its role as NTP server) from the reference clock. • 0 is lowest (directly connected) • 15 is most distant (hence less reliable) When the controller is operating as: • Client and server: this parameter is auto-configured. It is equal to the stratum value of the system peer +1. • Server only or in orphan mode (i.e., when the controller's subnet becomes isolated from other NTP servers and assumes the role as interim server): you can configure this parameter.
Server IPv4 address[1]	The IP addresses of reference NTP servers used by the controller. Minimum of 4; maximum of 8.
Used as preferred[1]	Indicates the NTP server in the list to be used by the controller.
Quality threshold (ms)[1]	Threshold for NTP accuracy. Setting range 0...1000. • 0 = not used. • Default value = 50 ms. The **Quality threshold** setting is compared to the DDT value NTP_WITHIN. If the **Quality threshold** is ≥ NTP_WITHIN, the NTP_QUALITY_WARNING DDT item is set to true (1) and the event is recorded in syslog.
1. If **Server Only** is selected, these parameters are disabled.	

Switch Tab

Description

The **Switch** tab is only available for CPUs without RIO scanner service. It contains these fields:

Field	Parameter	Value	Comment
ETH1	–	–	You cannot edit these field parameters here. Configuration can be modified in the **Service Port** tab, page 162.
ETH2	**Enabled**	**Yes**	default
		No	–
	Baud Rate	**Auto 10/100 Mbits/sec**	default
		100 Mbits/sec Half duplex	–
		100 Mbits/sec Full duplex	–
		10 Mbits/sec Half duplex	–
		10 Mbits/sec Full duplex	–
ETH3	**Enabled**	**Yes**	default
		No	–
	Baud Rate	**Auto 10/100 Mbits/sec**	default
		100 Mbits/sec Half duplex	–
		100 Mbits/sec Full duplex	–
		10 Mbits/sec Half duplex	–
		10 Mbits/sec Full duplex	–
Backplane	–	–	You cannot edit these field parameters.

NOTE: ETH1 port is a dedicated service port and the Ethernet backplane network is dedicated to the communication between modules on the rack. The switch parameters for those two ports cannot be configured in the **Switch** tab.

QoS Tab

Description

The M580 CPU can be configured to perform Ethernet packet tagging. The CPU supports the OSI layer 3 quality of service (QoS) standard defined in RFC-2475. When you enable QoS, the CPU adds a *differentiated services code point* (DSCP) tag to each Ethernet packet that it transmits to indicate the priority of that packet.

QoS Tab

The **QoS** tab is available only on CPUs that do not support the RIO scanner service (only on CPUs with commercial references that end with *20*).

Field	Parameter	Value	Comment
DSCP Tagging	–	**Enabled**	default
		Disabled	–
PTP	**DSCP PTP Event Priority**	59	–
	DSCP PTP General Priority	47	–
EtherNet/IP Traffic	**DSCP Value For I/O Data Schedule Priority Messages**	47	–
	DSCP Value For Explicit Message	27	–
	DSCP Value For I/O Data Urgent Priority Messages	55	–
	DSCP Value For I/O Data High Priority Messages	43	–
	DSCP Value For I/O Data Low Priority Messages	31	–
Modbus TCP Traffic	**DSCP Value For I/O Messages**	43	–
	DSCP Value For Explicit Message	27	–
Network Time Protocol Traffic	**DSCP Value For Network Time Protocol Messages**	59	–

DSCP tagging lets you prioritize the Ethernet packet streams based on the type of traffic in that stream.

To implement QoS settings in your Ethernet network:

- Use network switches that support QoS.
- Consistently apply DSCP values to network devices and switches that support DSCP.

- Confirm that switches apply a consistent set of rules for sorting DSCP tags, when transmitting and receiving Ethernet packets.

Service Port Tab

Service Port Parameters

These parameters are on the Control Expert **Service Port** tab:

Field	Parameter	Value	Comment
Service Port	–	**Enabled** (default)	Enable the port and edit port parameters.
	–	**Disabled**	Disable the port (no access to parameters).
Service Port Mode	–	**Access** (default)	This mode supports communications to Ethernet devices.
	–	**Mirroring**	In port mirroring mode, data traffic from one or more of the other ports is copied to this port. Connect a packet sniffing tool to this port to monitor and analyze port traffic. **NOTE:** In this mode, the **Service** port acts like a read-only port. That is, you cannot access devices (ping, connection to Control Expert, and so on) through the **Service** port.
Access Port Configuration	**Service Port Number**	**ETH1**	You cannot edit the value in the **Service Port Number** field.
Port Mirroring Configuration	**Source Port (s)**	**Internal Port**	Ethernet traffic to and from the internal processor sent to the Service Port
		ETH2	Ethernet traffic to and from ETH2 sent to the Service Port
		ETH3	Ethernet traffic to and from ETH3 sent to the Service Port
		Backplane Port	Ethernet traffic to and from the backplane sent to the Service Port
Automatic blocking of service port on Standby CPU (*in Hot Standby system only*)	–	**Deselected** (default)	Automatically enables the service port of the standby BMENOC0301.4, or any subsequent supporting version(s)of the module, to allow an RIO main ring, with or without distributed equipment, to communicate with the control network.
		Selected	Automatically blocks the service port to help avoid an unintentional loop.

Hot Standby Configuration

In an M580 Hot Standby configuration, some topologies may unintentionally create a loop that interferes with network communication. These topologies are essentially related to the management of flat networks, i.e., topologies in which the control network, remote I/O network, and/or the device network belong to the same subnet.

To help avoid creating an unintentional loop caused by connection to the service port, select the **Automatic blocking of service port on Standby CPU** check box that appears in the ServicePort tab of the configuration dialog. This check box is available only in Unity Pro 13.1 or any subsequent supporting version(s).

NOTE:

Unity Pro is the former name of Control Expert for version 13.1 or earlier.

To configure, select the **ServicePort** tab.

- Select the **Automatic blocking of service port on Standby CPU** check box so that the service port of the standby CPU is automatically blocked.
- Deselect the check box so that the service port is not automatically blocked.

The check box is deselected (unblocked) by default.

NOTE: These features are implemented in a Hot Standby system using a CPU with firmware version 2.7 or any subsequent supporting version(s), and a BMENOC0301.4 or any subsequent supporting version(s) of the module.

Refer to the **ServicePort** configuration topic (see Modicon M580 Hot Standby, System Planning Guide for, Frequently Used Architectures) to see topology examples in which this issue exists.

On-line Behavior

The **Service Port** parameters are stored in the application, but you can reconfigure the parameters in connected mode. Values that you reconfigure in connected mode are sent to the PAC through explicit messaging.

The changed values are not stored, so a mismatch can exist between the parameters that are being used and those that are in the stored application.

Advanced Settings Tab

Introduction

The **Advanced Settings** tab is only available for CPUs that do not support RIO scanning (DIO scanner service only). The **Advanced Settings** contains these fields:

- **EtherNet/IP Timeout Settings**
- **EtherNet/IP Scanner Behavior**

Timeout Settings

These parameters are in the **EtherNet/IP Timeout Settings** field:

Parameter	Value	Comment
FW_Open I/O Connection Timeout (msec)	4960	Specifies the amount of time the scanner waits for FW_Open response of an I/O connection.
FW_Open EM Connection Timeout (msec)	3000	Specifies the amount of time the scanner waits for FW_Open response of an EM connection.
EM Connection RPI (msec)	10000	Sets T->O and O->T RPI for all EM connections.
EM Request Timeout (sec)	10	Specifies the amount of time the scanner will wait between the request and the response of an explicit message.

Scanner Behavior

These parameters are in the **EtherNet/IP Scanner Behavior** field:

Parameter	Value	Comment
Allow RESET via explicit message	**Disabled**	(Default.) The scanner ignores the Identity object reset service request.
	Enabled	The scanner will reset if an Identity object reset service request is received.
Behavior when CPU state is STOP	**Idle**	(Default.) The EtherNet/IP I/O connection stays open, but the **Run/Idle** flag is set to Idle.
	STOP	The EtherNet/IP IO connection is closed.

Safety Tab

Introduction

A CIP Safety CPU is the originator of CIP Safety communications, and is identified by its originator unique identifier (OUNID). Use this tab to configure an OUNID for the CIP Safety CPU. Each OUNID is a 10 byte concatenated value, consisting of a:

- Safety Network Number (6 bytes)
- IP Address (4 bytes)

NOTE: Changes to the OUNID can be made only offline. After the changed configuration is built, the application can be downloaded to the PAC.

Safety Network Number

The Safety Network Number component of the OUNID can be auto-generated by Control Expert, or user-generated by manual input. If this number is:

- Auto-generated (the default), it is based on the current timestamp (date and time).
- Manually generated, it can be any 6 byte hexadecimal character string.

You can update the OUNID by updating the auto-generated value, or changing the manual value.

IP Address

This is automatically set to the CPU Main IP address, page 148. The OUNID is updated if the IP address changes.

CIP Safety OUNID Parameters

This tab page presents the following parameters:

Parameter	Description
Safety Network Number	Click **Advanced...** to open the **Safety Network Number** dialog, where you can enter this setting: • Automatically, by selecting **Time-based**, then clicking the **Generate** button. The auto-generated value appears in the **Number** field. • Manually, by selecting **Manual**, then a 6 byte hexadecimal character string in the **Number** field. Click **OK** to close the dialog and save the Safety Network Number.
IP Address	This read-only setting is automatically input, based on the configured **Main IP address** CPU setting.
OUNID	The auto-generated hexadecimal identifier: a concatenation of the Safety Network Number and the IP Address.

Configuring the M580 CPU with DTMs in Control Expert

Introduction

Some configuration features for the M580 CPU are accessed through its corresponding M580 DTM in the Control Expert **DTM Browser**.

Use the instructions in this section to configure the M580 CPU through the DTM.

About DTM Configuration in Control Expert

Introduction

The configuration of the M580 controller through standard Control Expert features is described elsewhere in this guide, page 139.

Some configuration that is specific to a particular device (like the M580 controller) is done through a corresponding device type manager (DTM) in Control Expert. This section describes that configuration.

Accessing Configuration Settings

Follow these steps to access the configuration settings in the DTM for the M580 controller in Control Expert:

Step	Action
1	Open Control Expert.
2	Open a Control Expert project that includes a M580 controller in the configuration.
3	Open the **DTM Browser** (**Tools > DTM Browser**).
4	Double-click the DTM that corresponds to the M580 controller in the **DTM Browser** to open the device editor of the DTM.
5	These headings appear in the configuration tree of the M580 DTM: • **Channel Properties** • **Services** • **EtherNet/IP Local Slaves** • **Device List** • **Logging**

Implicit Connections

You can use routed communications to make implicit EtherNet/IP or Modbus TCP connections to these devices in a different subnet:

- controller modules
- BMENOC0301/BMENOC0311 communications modules
- BMENOC0321(C) high-end control module

Accessing Channel Properties

Introduction

On the Control Expert **Channel Properties** page, you can select a **Source IP Address** (PC) from a pull-down menu.

The **Source IP Address** (PC) menu is a list of IP addresses that are configured for a PC that has the Control Expert DTM installed.

To make the connection, choose a **Source IP Address** (PC) that is in the same network as the controller and the device network.

You can execute these tasks through this connection:

- Perform fieldbus discovery.
- Execute Online Actions.
- Send an explicit message to an EtherNet/IP device.
- Send an explicit message to a Modbus TCP device.
- Diagnose modules.

Open the Page

View the **Channel Properties** for the controller:

Step	Action
1	Open a Control Expert project that includes a M580 controller.
2	Open the **DTM Browser** (**Tools > DTM Browser**).
3	In the **DTM Browser**, find the name that you assigned to the controller.
4	Double-click (or right-click **Open**) the name of the controller to open the configuration window.
5	Select **Channel Properties** in the navigation pane.

Property Descriptions

This table describes the parameters for the **Channel Properties**:

Field	Parameter	Description
Source Address	Source IP Address (PC)	A list of IP addresses assigned to network interface cards installed on your PC. **NOTE:** If the configured main IP address of the controller is not in the subnet of any of the IP configured on the interface cards of the PC, then the first interface card IP is suggested by default.
	Sub-Network Mask (read-only)	The subnet mask that is associated with the selected source IP address (PC).
EtherNet/IP Network Detection	Begin detection range address	The first IP address in the address range for automatic field bus discovery of EtherNet/IP devices.
	End detection range address	The last IP address in the address range for automatic field bus discovery of EtherNet/IP devices.
Modbus Network Detection	Begin detection range address	The first IP address in the address range for automatic field bus discovery of Modbus TCP devices.
	End detection range address	The last IP address in the address range for automatic field bus discovery of Modbus TCP devices.

Make the Connection

Connect to the **Source IP Address** (PC):

Step	Action
1	Select an IP address from the **Source IP Address** (PC) pull-down menu.
2	Press the **Apply** button.
3	In the **DTM Browser**, find the name that you assigned to the controller.
4	Right-click the name of the controller and scroll to **Connect**.

TCP/IP Monitoring

Expand (+) the **Channel Properties** heading in the configuration tree and select the **TCP/IP** item at level 1.

The read-only information on this page monitors the IP parameters that were configured in Control Expert.

Managing Source IP Addresses for Multiple PCs

When you connect a PC to a DTM-based Control Expert application, Control Expert requires that you define the IP address of the PC connected to the PLC, which is referred to as the *source IP address (PC)*. Rather than having to perform a **Build** in Control Expert each time you connect a PC to the PLC, the source IP address (PC) is selected automatically when you import the Control Expert application. During application import, the DTM retrieves all available configured NIC addresses of a connected PC and matches the subnet mask of the master with the available NIC list.

- If a match between the subnet mask of the master and the NIC list exists, Control Expert automatically selects the matched IP address as the *source IP address (PC)* in the **Channel Properties** page.
- If multiple matches exist, Control Expert automatically selects the IP address nearest to the subnet mask.
- If no match exists, Control Expert automatically selects the IP address to the nearest available subnet mask.

Configuring DHCP and FDR Address Servers

DHCP and FDR Address Servers

The M580 CPU includes both a dynamic host communication protocol (DHCP) and a fast device replacement (FDR) server. The DHCP server provides IP address settings to networked up to devices. The FDR server provides operating parameter settings to replacement Ethernet devices that are equipped with FDR client functionality.

Accessing the Address Server

Access the address server for the M580 CPU in Control Expert:

Step	Action
1	Open Control Expert.
2	Open a Control Expert project that includes a M580 CPU in the configuration.
3	Open the **DTM Browser** (**Tools > DTM Browser**).
4	Double-click the DTM that corresponds to the M580 CPU in the **DTM Browser** to open the device editor of the DTM.
5	Expand (+) the **Services** heading in the configuration tree.
6	Select the **Address Server** item in the configuration tree to see the address server configuration.

Configuration

Configure the address server to perform these tasks:

- Enable and disable the CPU FDR service.
- View an automatically generated list of all devices included in the CPU configuration, displaying for each device:
    - IP addressing parameters
    - whether the device IP addressing parameters are provided by the CPU embedded DHCP server

Manually add remote devices that are not part of the CPU configuration to the CPU DHCP client list.

NOTE: Remote devices added in this way are equipped with DHCP client software and are configured to subscribe to the CPU IP addressing service.

Enabling the FDR Service

To enable the FDR service, set the **FDR Server** field to **Enabled**. To disable the service, toggle the same field to **Disabled**.

You can disable the FDR service for CPUs that do not support RIO scanning (commercial references that end in *20*). The FDR service is always enabled for CPUs that support RIO scanning (commercial references that end in *40*).

Any networked Ethernet device equipped with FDR client functionality can subscribe to the CPU FDR service.

The maximum size of the FDR client operating parameter files depends on the CPU reference. When this capacity is reached, the CPU cannot store additional client FDR files

CPU Reference	PRM File Size	Concurrent Connections
BMEP581020	8 MB	64
BMEP582020	16 MB	128
BMEP582040	17 MB	136
BMEP583020	16 MB	128
BMEP583040	25 MB	208
BMEP584020	16 MB	128
BMEP584040	25 MB	208
BMEP585040	25 MB	208
BMEP586040	25 MB	208
BMEH582040	25 MB	208
BMEH584040	25 MB	208
BMEH586040	25 MB	208

NOTE: The FDR usage percentage is monitored by the FDR_USAGE variable in the DDDT, page 295.

Viewing the Auto-Generated DHCP Client List

The list of **Automatically Added Devices** includes a row for each remote device that is:

- part of the CPU configuration
- configured to subscribe to the CPU DHCP addressing service

NOTE: You cannot add devices to this list in this page. Instead, use the configuration pages for the remote device to subscribe to this service.

This table describes the available properties:

Property	Description
Device No	The number assigned to the device in the Control Expert configuration.
IP Address	The client device IP address.
DHCP	TRUE indicates that the device subscribes to the DHCP service.
Identifier Type	Indicates the mechanism used by the server to recognize the client (MAC address or DHCP device name).
Identifier	The actual MAC address or DHCP device name.
Netmask	The client device subnet mask.
Gateway	A DHCP client device uses the gateway IP address to access other devices that are not located on the local subnet. A value of 0.0.0.0 constrains the DHCP client device by allowing it to communicate only with devices on the local subnet.

Manually Adding Remote Modules to the DHCP Service

Remote modules that are part of the CPU configuration – and which have subscribed to the CPU IP addressing service – automatically appear in the **Automatically Added Devices** list.

Other remote modules that are not part of the CPU configuration can be manually added to the CPU DHCP IP addressing service.

Manually add networked Ethernet modules that are not part of the CPU configuration to the CPU IP addressing service:

<table>
<tr><th>Step</th><th colspan="2">Description</th></tr>
<tr><td>1</td><td colspan="2">In the Address Server page, click the Add button in the Manually Added Devices field to instruct Control Expert to add an empty row to the list.</td></tr>
<tr><td rowspan="6">2</td><td colspan="2">In the new row, configure these parameters for the client device:</td></tr>
<tr><td>IP Address</td><td>Type in the IP address of the client device.</td></tr>
<tr><td>Identifier Type</td><td>Select the type of value the client device uses to identify itself to the FDR server:
• MAC address
• device Name</td></tr>
<tr><td>Identifier</td><td>Depending upon the identifier type, type in the client device setting for the MAC address or name.</td></tr>
<tr><td>Netmask</td><td>Type in the client device subnet mask.</td></tr>
<tr><td>Gateway</td><td>Type in the gateway address that remote devices can use to communicate with devices located on other networks. Use 0.0.0.0 if remote devices do not communicate with devices located on other networks.</td></tr>
<tr><td>3</td><td colspan="2">Refer to the topic Configuring Properties in the Device Editor (see Modicon M580, BMENOC0301/0311 Ethernet Communications Module, Installation and Configuration Guide) for instructions on how to apply edited properties to networked devices.</td></tr>
</table>

Configuring Generic Device DTMs

The following topics describe how to use Control Expert to select and configure a generic device DTM for a remote device, including properties that define:

- the connection between the remote device and the controller
- the degree to which the actual remote device must match the remote device described in the Control Expert project configuration

NOTE: If using a vendor-specific DTM, consult the documentation the vendor provides for that device.

For an example of how to configure a Schneider Electric DTM for the STBNIC2212 communication module, refer to the chapter Implicit Messaging, page 367.

Generic DTM Types

For an M580 project, the following generic DTMs are available:

- Advanced Generic DTM
- Generic Device DTM
- Generic Device Explicit Message DTM
- Generic Safety DTM

NOTE: The following topics address non-safety generic DTMs. For more information about M580 Safety DTMs, refer to the *Modicon M580 Safety Manual* topic *Configuring Safety Device DTMs*,

Displaying Remote Device and DTM Properties

Use this page to view properties that describe:

- the remote device, and
- its DTM

To display this page, select a remote device in the **DTM Browser** to open its DTM. Then, in the left pane of the **Device Editor**, select the node that displays the assigned device name.

NOTE: When this page is displayed, if this device is capable of supporting an additional connection, you can use the **Add Connection** command to create a new connection for this device, page 178.

Properties

The properties displayed in this page are read-only. The source of the displayed property values is the device DTM. The following list presents an example of the self-explanatory properties you may see displayed in a generic DTM:

- File Name
- File:
  - Description
  - File Creation Date
  - File Creation Time
  - Last Modification Date
  - Last Modification Time
  - EDS Revision
- Device:
  - Vendor Name
  - Device Type
  - Vendor Code
  - Product Type
  - Product Code
  - Major Revision
  - Minor Revision
  - Product Name
  - Catalog Number

Adding a Generic Device DTM to an M580 Project

As a prerequisite, create a project in control expert with an M580 controller.

To add a generic device DTM:

Step	Action
1	Select **Tools > DTM Browser**.
2	Right click on the controller node, for example, BMEP58_ECPU_EXT and select **Add...**.
3	For **Protocol** select **EtherNet IP**.
4	Select one of the following generic DTMs from the list:

Step	Action
	• **Advanced Generic EDS** • **Generic Device** • **Generic Device Explicit Msg**
5	Click **Add DTM**.
6	In the **Properties of device** dialog **General** tab, accept the default **Name** or enter a new name for the DTM, then click **OK**.

The new DTM appears in the **DTM Browser** as a node on the **Distributed Bus**.

Adding and Removing Connections

Use the **Device Editor** to access the DTM for a remote device, where you can add and remove device connections.

For Advanced Generic DTMs and Generic Device DTMs, one Exclusive Owner connection is added by default.

No connection can be added to a Generic Device Explicit Msg DTM.

Adding a Connection

To add a connection for a remote device:

Step	Action
1	In the **DTM Browser**, double-click a remote device. Its DTM opens in the **Device Editor**.
2	In the left pane of the **Device Editor**, select the node displaying the name of the remote device. **NOTE:** • For Advanced Generic DTMs and Generic Device DTMs, one **Exclusive Owner** connection is added by default. • If the device is capable of supporting additional connections, the **Add Connection** button becomes enabled. For example: ◦ a Generic Device DTM can support only a single **Status (Optional Connection)** connection. ◦ an Advanced Generic DTM can support multiple **Exclusive Owner, Listen Only**, and **Input Only** connections. • If the **Add Connection** button remains disabled, the device is presently supporting its maximum number of connections. In this case, a new connection can be added only after an existing connection is removed.
3	Click the **Add Connection** button. The **Select the connection to add** dialog opens.

Step	Action
4	In the **Connection to add** list, select a connection type. **NOTE:** The types of connections available in the list depends upon the connection types supported by the specific remote device.
5	Click **OK** to close the dialog. The new connection appears in the tree control in the left pane.
6	Click the following tabbed pages, and configure the properties in each page (as necessary): • Connection, page 179 • Identity Check, page 181 • Configuration Settings, page 182
7	Do one of the following: • click **Apply** to save your edits and leave the window open, or • click **OK** to save your edits and close the window

Removing a Connection

To remove a connection between a remote device and the communication module:

Step	Action
1	In the **DTM Browser**, double-click a remote device. Its DTM opens in the **Device Editor**.
2	In the left pane of the **Device Editor**, beneath the remote device name, select the connection node you wish to remove.
3	Click the **Remove Connection** button. The dialog opens. The connection disappears from the tree control.
4	Do one of the following: • click **Apply** to save your edits and leave the window open, or • click **OK** to save your edits and close the window

Configuring Generic DTM EtherNet/IP Connections

Use this tab to configure connection properties that are required by the remote device DTM. An EtherNet/IP connection provides a communication link between two or more devices. Properties for a single connection must be configured in the DTMs for each of the connected devices.

To open this page:

Step	Action
1	Double-click on the remote device in the **DTM Browser** to open its DTM in the **Device Editor**.
2	In the navigation tree in the left pane of the **Device Editor**, select the connection node you want to configure.
3	In the right pane of the **Device Editor**, click the **Connection** tab.

NOTE: When this page is open, you can use the **Remove Connection** command to delete the selected connection.

Remote Device Connection Properties

A connection to a remote Schneider Electric device can present these properties:

Property	Description
Output RPI (O->T)	Output RPI or Input RPI indicates the refresh period for the respective connection in milliseconds. (These parameters can also be set in the DTM for the communication module device.)
Input RPI (T->O)	
Input size	This is the number of bytes (0 ... 505) that are reserved for input data.
Input instance	The input instance of the connection (0 ... 255).
Input mode	This mode is the input transmission type: • Multicast • Point to Point
Input type (read only)	This is the Ethernet packet type (fixed or variable length) for transmission. **NOTE:** The Ethernet communication module supports only **Fixed** length packets.
Input priority	This transmission priority value depends upon the device DTM. These are the available values: • Low • High • Scheduled
Input trigger	These are the available values for the transmission trigger: • Cyclic • Change of state or application
Output size	This is the number of bytes (0 ... 509) that are reserved for output data.
Output instance	The output instance of the connection (0 ... 255).
Output mode	This mode is the output transmission type: • Multicast • Point to Point

Property	Description
Output type (read only)	This is the Ethernet packet type (fixed or variable length) for transmission. **NOTE:** The Ethernet communication module supports only **Fixed** length packets.
Output priority	This transmission priority value depends upon the device DTM. These are the available values: • Low • High • Scheduled
Configuration Instance	The configuration instance of the connection (0 ... 255).

Checking Remote Device Identity

Use this tab to specify the degree to which a remote device (detected on the network) conforms to the configuration settings for the same remote device in the Control Expert application project. Control Expert does not maintain connections to a remote device that does not pass this identity check.

NOTE: This page appears only for generic DTM types that support connections, for example, Generic Device DTM, Advanced Generic DTM, and Generic Safety DTM.

The Generic Device Explicit Msg DTM does not support connections.

To open this page:

Step	Action
1	Double-click on the remote device in the **DTM Browser** to open its DTM in the **Device Editor**.
2	In the navigation tree in the left pane of the **Device Editor** select the connection node you want to configure.
3	In the right pane of the **Device Editor**, click the **Identity Check** tab.

NOTE: When this page is open, you can use the **Remove Connection** command to delete the selected connection.

Remote Device Identity Properties

A connection to a remote Schneider Electric device can present these properties:

Property	Description
Check Identity	This property defines the rule that Control Expert uses to compare the configured versus the actual remote device. These are the available settings: • **Must match exactly**: The DTM or EDS file exactly matches the remote device. • **Disable**: The checking function does not run. The identity portion of the connection is filled with zero values (the default setting). • **Must be compatible**: When the remote device is not the same as defined by the DTM/EDS, it emulates the DTM/EDS definitions. • **None**—no checking occurs; the identity portion of the connection is omitted • **Custom**: Enable the following parameter settings individually.
When **Check identity** is set to **Custom**, complete these fields:	
Compatibility Mode	• **True**: For each of the following selected tests, the DTM/EDS and remote device are compatible. • **False**: For each of the following selected tests, the DTM/EDS and remote device match exactly.
Minor Version	For each of these, select a setting: • **Compatible**: Include the parameter in the test. • **Not checked**: Do not include the parameter in the test.
Major Version	
Product Code	
Product Type	
Product Vendor	

Generic DTM Configuration Settings

Use the **Configuration Settings** tab to complete the configuration of the connection to this remote device. The information added in this page extends the address path to the remote device.

To open this page:

Step	Action
1	Double-click on the remote device in the **DTM Browser** to open its DTM in the **Device Editor**.
2	In the navigation tree in the left pane of the **Device Editor** select the connection node you want to configure.
3	In the right pane of the **Device Editor**, click the **Configuration Settings** tab.

NOTE: When this page is open, you can use the **Remove Connection** command to delete the selected connection.

Configuration Settings

The content of this page can vary, depending upon the DTM – selected in the **Add** dialog – that defines this device. Examples of DTM properties that may be configured in this page include:

This DTM type...	**Can require this content...**	
	Property	**Description**
Generic Device	Configuration[1]:	A hexadecimal extension to the addressing path.
Advanced Generic Device	Input Instance[1]:	The device specific assembly number associated with input (T -> O) transmissions.
	Output Instance[1]:	The device specific assembly number associated with output (O -> T) transmissions.
	Configuration Instance[1]:	The device specific assembly number associated with device configuration settings.
	Configuration[1]:	A hexadecimal extension to the addressing path.
1. The value, or range of values, that can be used to configure this property must be obtained from the manufacturer of the specific device and device DTM.		

Diagnostics through the Control Expert DTM Browser

Introducing Diagnostics in the Control Expert DTM

Introduction

The Control Expert DTM provides diagnostics information that is collected at configured polling intervals. Use this information to diagnose the operation of the embedded Ethernet scanner service in the CPU.

Connect the DTM

Before you can open the diagnostics page, make the connection between the DTM for the CPU's embedded scanner service:

Step	Action
1	Open a Control Expert project.
2	Open the Control Expert **DTM Browser** (**Tools > DTM Browser**).
3	Right-click the name that is assigned to your CPU in the **DTM Browser**.
4	Select **Connect**.

Open the Page

Access the **Diagnosis** information:

Step	Action
1	Right-click the name that is assigned to your CPU in the **DTM Browser**.
2	Select **Device Menu > Diagnosis** to view the available diagnostics pages.

Diagnostics Information

The diagnostics window has two distinct areas:

- left pane: LED icons indicate the operating status of modules, devices, and connections.

- right pane: These pages show diagnostics data for these items:
  - CPU’s embedded scanner service
  - local slave nodes that are activated for the CPU’s embedded scanner service
  - EtherNet/IP connections between the CPU’s embedded scanner service and a remote EtherNet/IP device

When the appropriate DTM is connected to the CPU, Control Expert sends an explicit message request once per second to detect the state of the CPU’s embedded scanner service and of all the remote devices and EtherNet/IP connections linked to the CPU.

Control Expert places one of these status icons over the module, device, or connection in the left pane of the **Diagnostic** window to indicate its current status:

Icon	Communication module	Connection to a remote device
	Run state is indicated.	The health bit for every EtherNet/IP connection and Modbus TCP request (to a remote device, sub-device, or module) is set to active (1).
	One of these states is indicated: • unknown • stopped • not connected	The health bit for at least one EtherNet/IP connection or Modbus TCP request (to a remote device, sub-device, or module) is set to inactive (0).

Bandwidth Diagnostics

Introduction

Use the **Bandwidth** page to view the dynamic and static data for the bandwidth use by the embedded Ethernet scanner service in the CPU.

NOTE: Before you can open the diagnostics page, make the connection between the DTM for the CPU's embedded scanner service and the physical module.

Open the Page

Access the **Bandwidth** information:

Step	Action
1	In the **DTM Browser**, right-click the name that is assigned to your CPU.
2	Select **Device menu > Diagnosis**.
3	In the left pane of the **Diagnosis** window, select the CPU node.
4	Select the **Bandwidth** tab to open that page.

Data Display

Use the **Refresh Every 500ms** checkbox to display the static or dynamic data:

Checkbox	Description
Selected	• Display data that is dynamically updated every 500 ms. • Increment the number at the top of the table each time data is refreshed.
De-selected	• Display static data. • Do not increment the number at the top of the table. That number now represents a constant value.

Bandwidth Diagnostic Parameters

The **Bandwidth** page displays the following parameters for the communication module:

Parameter	Description
I/O - Scanner:	
EtherNet/IP Sent	The number of EtherNet/IP packets the module has sent in packets/second.
EtherNet/IP Received	The number of EtherNet/IP packets the module has received in packets/second.
Modbus TCP Received	The number of Modbus TCP requests the module has sent in packets/second.
Modbus TCP Responses	The number of Modbus TCP responses that the CPU’s embedded scanner service has received in packets/second.
I/O - Adapter:	
EtherNet/IP Sent	The number of EtherNet/IP packets (per second) that the CPU’s embedded scanner service has sent in the role of a local slave.
EtherNet/IP Received	The number of EtherNet/IP packets (per second) that the CPU’s embedded scanner service has received in the role of a local slave.
I/O - Module	
Module Capacity	The maximum number of packets (per second) that the CPU’s embedded scanner service can process.
Module Utilization	The percentage of the CPU’s embedded scanner service capacity being used by the application.
Messaging - Client:	
EtherNet/IP Activity	The number of explicit messages (packets per second) sent by the CPU’s embedded scanner service using the EtherNet/IP protocol.
Modbus TCP Activity	The number of explicit messages (packets per second) sent by the CPU’s embedded scanner service using the Modbus TCP protocol.
Messaging - Server:	
EtherNet/IP Activity	The number of server messages (packets per second) received by the CPU’s embedded scanner service using the EtherNet/IP protocol.
Modbus TCP Activity	The number of server messages (packets per second) received by the CPU’s embedded scanner service using the Modbus TCP protocol.
Module:	
Processor Utilization	The percentage of the CPU’s embedded scanner service processing capacity used by the present level of communication activity.

RSTP Diagnostics

Introduction

Use the **RSTP Diagnostic** page to view the status of the RSTP service of the embedded Ethernet scanner service in the controller. The page displays dynamically generated and static data for the module.

NOTE: Before you can open the diagnostics page, make the connection between the DTM for the controller's embedded scanner service and the physical module.

Open the Page

Access the **RSTP Diagnosis** information:

Step	Action
1	In the **DTM Browser**, right-click the name that is assigned to your controller.
2	Select **Device menu > Diagnosis**.
3	In the left pane of the **Diagnosis** window, select the controller node.
4	Select **RSTP Diagnostic** tab to open that page.

Data Display

Select the **Refresh Every 500ms** check box to display the static or dynamic data:

Checkbox	Description
Selected	• Display data that is dynamically updated every 500 ms. • Increment the number at the top of the table each time data is refreshed.
De-selected	• Display static data. • Do not increment the number at the top of the table. That number now represents a constant value.

RSTP Diagnostic Parameters

The **RSTP Diagnostic** page displays the following parameters for each controller port:

Parameter	Description
Bridge RSTP Diagnostic:	
Bridge Priority	This 8-byte field contains the two-byte value that is assigned to the controller's embedded Ethernet switch.
MAC Address	The Ethernet address of the controller, found on the front of the controller.
Designated Root ID	The Bridge ID of the root device.
Root Path Cost	The aggregate cost of port costs from this switch back to the root device.
Default Hello Time	The interval at which Configuration BPDU messages are transmitted during a network convergence. For RSTP this is a fixed value of 2 seconds.
Learned Hello Time	The current Hello Time value learned from the root switch.
Configured Max Age	The value (6 ... 40) that other switches use for MaxAge when this switch is acting as the root.
Learned Max Age	The maximum age learned from the root switch. This is the actual value currently used by this switch.
Total Topology Changes	The total number of topology changes detected by this switch since the management entity was last reset or initialized.
Ports ETH 2 and ETH 3 RSTP Statistics:	
Status	The port's current state as defined by RSTP protocol. This state controls the action the port takes when it receives a frame. Possible values are: disabled, discarding, learning, forwarding.
Role:	The port's current role per RSTP protocol. Possible values are: root port, designated port, alternate port, backup port, disabled port.
Cost	The logical cost of this port as a path to the root switch. If this port is configured for AUTO then the cost is determined based on the connection speed of the port.
STP Packets	A value in this field indicates that a device on the network has the STP protocol enabled. **NOTE:** • Other devices that are enabled for STP can severely affect the network convergence times. Disable the STP protocol (but not the RSTP protocol) on every network device that supports STP. • The controller does not support the STP protocol. The controller's embedded switch ignores STP packets.

Network Time Service Diagnostics

Introduction

Use the **Network Time Service Diagnostic** page to display dynamically generated data describing the operation of the simple network time protocol – either SNTP or NTPv4 (depending on your CPU firmware) – service that you configured in the network time server page, page 155 in Control Expert.

NOTE: Before you can open the diagnostics page, make the connection between the DTM for the target communication module and the CPU.

Refer to the *System Time Stamping User Guide* (see System Time Stamping, User Guide) for detailed diagnostic information.

Open the Page

Access the **NTP Diagnostic** information:

Step	Action
1	In the **DTM Browser**, find the name that is assigned to the CPU.
2	Right-click the CPU DTM, and select **Device menu > Diagnosis**.
3	In the left pane of the **Diagnosis** window, select the CPU node.
4	Select the **NTP Diagnostic** tab to open that page.

Click the **Reset Counter** button to reset the counting statistics on this page to 0.

SNTP Service Diagnostic Parameters

This table describes the SNTP time synchronization service parameters:

Parameter	Description
Refresh Every 500ms	Check this box to dynamically update the page every 500ms. The number of times this page has been refreshed appears immediately to the right.
Network Time Service	Monitor the operational status of the service in the module: • *green*: operational • *orange*: disabled
Network Time Server Status	Monitor the communication status of the NTP server: • *green*: The NTP server is reachable.

<table>
<tr><th>Parameter</th><th>Description</th></tr>
<tr><td></td><td>red: The NTP server is not reachable.</td></tr>
<tr><td>Last Update</td><td>Elapsed time, in seconds, since the most recent NTP server update.</td></tr>
<tr><td>Current Date</td><td>System date</td></tr>
<tr><td>Current Time</td><td>The system time is presented in the hh:mm:ss format.</td></tr>
<tr><td>DST Status</td><td>Set the status of the automatic daylight savings service:
ON: The automatic adjustment of daylight savings is enabled. The current date and time reflect the daylight savings time adjustment.
OFF: The automatic adjustment of daylight savings is disabled. (The current date and time may not reflect the daylight savings time adjustment.)</td></tr>
<tr><td>Quality</td><td>This correction (in seconds) applies to the local counter at every NTP server update. Numbers greater than 0 indicate increasingly excessive traffic condition or an NTP server overload.</td></tr>
<tr><td>Requests</td><td>This value represents the total number of client requests sent to the NTP server.</td></tr>
<tr><td>Responses</td><td>This value represents the total number of server responses sent from the NTP server.</td></tr>
<tr><td>Errors</td><td>This value represents the total number of unanswered NTP requests.</td></tr>
<tr><td>Last Error</td><td>This value indicates the last detected error code received from the NTP client:
0: good NTP configuration
1: late NTP server response (can be caused by excessive network traffic or server overload)
2: NTP not configured
3: invalid NTP parameter setting
4: NTP component disabled
5: NTP server is not synchronized (NTP server needs to be synchronized so that the NTP accesses behave as defined in the client NTP settings)
7: unrecoverable NTP transmission
9: invalid NTP server IP address
15: invalid syntax in the custom time zone rules file</td></tr>
<tr><td>Primary / Secondary NTP Server IP</td><td>The IP addresses correspond to the primary and secondary NTP servers.
NOTE: A green LED to the right of the primary or secondary NTP server IP address indicates the active server.</td></tr>
<tr><td>Auto Adjust Clock for Daylight Savings</td><td>Configure the daylight savings adjustment service:
enabled
disabled</td></tr>
<tr><td>DST Start / DST End</td><td>Specify the day on which daylight savings time begins and ends:<table><tr><td>Month</td><td>Set the month in which daylight savings time starts or ends.</td></tr><tr><td>Day of Week</td><td>Set the day of the week on which daylight savings time starts or ends.</td></tr><tr><td>Week#</td><td>Set the occurrence of the specified day within the specified month.</td></tr></table></td></tr>
</table>

Parameter	Description
Time Zone	Select the time zone plus or minus Universal Time, Coordinated (UTC)
Offset	Configure the time (in minutes) to be combined with the time zone selection (above) to produce the system time.
Polling Period	Set the frequency with which the NTP client requests an updated time from the NTP server

NTPv4 Service Diagnostic Parameters

This table describes the NTP time synchronization service parameters:

Parameter		Description
Refresh Every 500ms		Check this box to dynamically update the page every 500ms. The number of times this page has been refreshed appears immediately to the right.
NTP V4 Service	Service state	The operational status of the service in the module: • *green*: operational • *orange*: disabled
	Sync	The status of the module: • *green*: synchronized • *orange*: not synchronized
	Accuracy	NTP clients only: The estimated difference between local (client) time and server time.
	Mode	• Server / Client • Server only
System clock	Date	Local date.
	Time	Local time.
	Time Zone	The local time zone, by reference to coordinated universal time (UTC).
	DST	The status of the automatic daylight savings service: • *ON*: The automatic adjustment of daylight savings is enabled. The current date and time reflect the daylight savings time adjustment. • *OFF*: The automatic adjustment of daylight savings is disabled. (The current date and time may not reflect the daylight savings time adjustment.)
<NTP System Status>	UTC-Date	The date at the UTC time source.
	UTC-Time	The time at the UTC time source.

Parameter		Description
	Stratum	The relative position in the hierarchy between this client and the original time source (stratum 1) reference. If the mode is: • Server/Client: the value equals the system peer stratum value + 1. • Server only (or orphan): a user-defined value.
	Root delay	NTP clients only: The round trip request delay, in milliseconds, from a client to a stratum 1 server.
	Root dispersion	NTP clients only: The additional delay contributed by other factors.
	Polling time	NTP clients only: Polling interval, in seconds.
	RefID	IPv4 address of the time source.
<NTP Peers Statuses> (NTP clients only)	NTP client CPU can be configured with up to 8 time source peers, each a potential server to the CPU NTP client.	
	IP	Peer IPv4 address of the peer.
	RefID	IP address of the time source used by the peer.
	Select	Indicates the peer used as the time source (Current) and other viable peer time sources (Candidate).
	Reach count	Percentage of NTP messages successfully sent to and received from the peer.
	Stratum	The relative position in the hierarchy between this client and the original time source (stratum 1) reference.
	Poll	Polling interval, in seconds.
	Delay	Time to send request / receive response.
	Offset	The value to subtracted from received time value to obtain time value to be applied.
	Jitter	Variability in delay.

Local Slave / Connection Diagnostics

Introduction

Use the **Local Slave Diagnostic** page and the **Connection Diagnostic** page to display the I/O status and production/consumption information for a selected local slave or connection.

NOTE:

- Before you can open the diagnostics page, make the connection between the DTM for the target communication module and the CPU.
- To get data from the primary CPU, make the connection to the Main IP address of the CPU (see Modicon M580 Hot Standby, System Planning Guide for, Frequently Used Architectures).

Open the Page

Access the diagnostics information:

Step	Action
1	In the **DTM Browser**, find the name that is assigned to the CPU.
2	Right-click the CPU DTM, and select **Device menu > Diagnosis**.
3	In the left pane of the **Diagnosis** window, select the CPU node.
4	Select the **Local Slave Diagnostic** tab or the **Connection Diagnostic** tab to open that page.

Data Display

Use the **Refresh Every 500ms** checkbox to display the static or dynamic data:

Checkbox	Description
Selected	• Display data that is dynamically updated every 500 ms. • Increment the number at the top of the table each time data is refreshed.
De-selected	• Display static data. • Do not increment the number at the top of the table. That number now represents a constant value.

Local Slave / Connection Diagnostic Parameters

This following tables display the diagnostic parameters for the selected local slave or scanner connection.

This table shows the **Status** diagnostic parameters for the selected connection:

Parameter	Description
Input	An integer representing input status.
Output	An integer representing output status.
General	An integer representing basic connection status.
Extended	An integer representing extended connection status.

The **Input** and **Output** status diagnostic parameters can present these values:

Input/Output Status (dec)	Description
0	OK
33	Time-out
53	IDLE
54	Connection established
58	Not connected (TCP)
65	Not connected (CIP)
68	Connection establishing
70	Not connected (EPIC)
77	Scanner stopped

This table shows the **Counter** diagnostic parameters for the selected connection:

Parameter	Description
Frame Error	Increments each time a frame is not sent by missing resources or is impossible to send.
Time-Out	Increments each time a connection times out.
Refused	Increments when connection is refused by the remote station.
Production	Increments each time a message is produced.
Consumption	Increments each time a message is consumed.
Production Byte	Total of produced messages, in bytes, since the communication module was last reset.

Parameter	Description
Consumption Byte	Total of consumed messages, in bytes, since the communication module was last reset.
Theoretical Packets per second	Packets per second calculated using current configuration value.
Real Packets per second	Actual number of packets per second generated by this connection.

This table shows the **Diagnostic** parameters for the selected connection:

Parameter	Description
CIP Status	An integer representing CIP status.
Extended Status	An integer representing extended CIP status.
Production Connection ID	The connection ID for the data produced by the local slave.
Consumption Connection ID	The connection ID for the data produced by the local slave.
O -> T API	Actual packet interval (API) of the production connection.
T -> O API	Actual packet interval (API) of the consumption connection.
O -> T RPI	Requested packet interval (RPI) of the production connection.
T -> O RPI	Requested packet interval (RPI) of the consumption connection.

This table shows the **Socket Diagnostics** diagnostic parameters for the selected connection:

Parameter	Description
Socket ID	Internal identification of the socket.
Remote IP Address	IP address of the remote station for this connection.
Remote Port	UDP port number of the remote station for this connection.
Local IP Address	IP address of the communication module for this connection.
Local Port	UDP port number of the communication module for this connection.

This table shows the **Production** diagnostic parameters for the selected connection:

Parameter	Description
Sequence Number	The number of the sequence in the production.
Max Time	Maximum time between two produced messages.

Parameter	Description
Min Time	Minimum time between two produced messages.
RPI	Current production time.
Overrun	Increments each time a produced message exceeds RPI.
Underrun	Increments each time a produced message is less than RPI.

This table shows the **Consumption** diagnostic parameters for the selected connection:

Parameter	Description
Sequence Number	The number of the sequence in the consumption.
Max Time	Maximum time between two consumption messages.
Min Time	Minimum time between two consumption messages.
RPI	Current consumption time.
Over Run	Increments each time a consumed message exceeds RPI.
Under Run	Increments each time a consumed message is less than RPI.

Local Slave or Connection I/O Value Diagnostics

Introduction

Use the **I/O Values** page to display both the input data image and output data image for the selected local slave or scanner connection.

NOTE: Before you can open the diagnostics page, make the connection, page 401 between the DTM and the target communication module.

Open the Page

Access the **I/O Values** information:

Step	Action
1	In the **DTM Browser**, find the name that is assigned to the CPU DTM.
2	Right-click the CPU DTM , and select **Device menu > Diagnosis**.
3	In the left pane of the **Diagnosis** window, select the CPU.
4	Select the **I/O Values** tab.

Data Display

Use the **Refresh Every 500ms** checkbox to display the static or dynamic data:

Checkbox	Description
Selected	• Display data that is dynamically updated every 500 ms. • Increment the number at the top of the table each time data is refreshed.
De-selected	• Display static data. • Do not increment the number at the top of the table. That number now represents a constant value.

Local Slave / Scanner Connection I/O Values

This page displays theses parameters for either a local slave or a remote device connection input and output values:

Parameter	Description
Input/Output data display	A display of the local slave or remote device input or output data image.
Length	The number of bytes in the input or output data image.
Status	The Scanner Diagnostic object’s status, with respect to the read of the input or output data image.

Logging DTM Events to a Control Expert Logging Screen

Description

Control Expert maintains a log of events for:

- the Control Expert embedded FDT container
- each Ethernet communication module DTM
- each EtherNet/IP remote device DTM

Events relating to the Control Expert FDT container are displayed in the **FDT log event** page of the **Output Window**.

Events relating to a communication module or remote EtherNet/IP device are displayed:

- in configuration mode: in the **Device Editor**, by selecting the **Logging** node in the left pane
- in diagnostic mode: in the **Diagnostics** window, by selecting the **Logging** node in the left pane

Logging Attributes

The **Logging** window displays the result of an operation or function performed by Control Expert. Each log entry includes the following attributes:

Attribute	Description	
Date/Time	The time the event occurred, displayed in the format: yyyy-mm–dd hh:mm:ss	
Log Level	The level of event importance. Values include:	
	Information	A successfully completed operation.
	Warning	An operation that Control Expert completed, but which may lead to a subsequent error.
	Error	An operation that Control Expert was unable to complete.
Message	A brief description of the core meaning of the event.	
Detail Message	A more detailed description of the event, which may include parameter names, location paths, etc.	

Accessing the Logging Screen

In Control Expert:

Step	Action
1	Open a project that includes a BME •58 •0•0 Ethernet CPU.
2	Clock **Tools > DTM Browser** to open the **DTM Browser**.
3	In the **DTM Browser**, double-click the CPU (or right-click **Open**) to open the configuration window.
4	Select **Logging** in the navigation tree in the left pane of the window.

Logging DTM and Module Events to the SYSLOG Server

Configuring the SYSLOG Server

Configure the SYSLOG server address for logging DTM and module events:

Step	Action
1	In Control Expert, select **Tools** > **Project Settings**.
2	In the left pane of the **Project Settings** window, select **Project Settings** > **General** > **PLC diagnostics**.
3	In the right pane: • Select the **PLC event logging** check box. • In the **SYSLOG server address** field enter the IP address of the SYSLOG server. • In the **SYSLOG server port number** field, enter the port number. **NOTE:** The SYSLOG server protocol is not configurable, and is set to **tcp** by default.

NOTE: Refer to the *Modicon Controllers Platform Cyber Security Reference Manual* for information on setting up a SYSLOG server in your system architecture (see Modicon Controllers Platform, Cyber Security, Reference Manual).

Enable Tracking for SYSLOG Events

Perform these tasks in the Security Editor tool to enable the Syslog service to track the Syslog events in the Syslog server :

Tab	Task
Profiles	Create a new profile with the applicable audit cases.
Policies	Enable at least the minimum security (Security on, no login).
	Select the **Audit box** to implement the audit for the new profiles you want to monitor.

DTM Events Logged to the SYSLOG Server

These DTM events are logged to the SYSLOG server:

- Configuration parameter change
- Add/Delete device
- Rebuild All

- Build Changes
- Renaming of I/O variables
- Add/Modify tasks

BME•58•0•0 CPU Events Logged to the SYSLOG Server

These BME•58•0•0 CPU events are logged to the SYSLOG server:

- TCP connection error due to Access Control List
- Enable/Disable of communication services outside configuration
- Ethernet port link up/down events
- RSTP topology change
- Program operating mode change of COMs (RUN, STOP, INIT)
- Successful and unsuccessful FTP login

Online Action

Online Action

Introduction

You can view and configure the settings in the **Online Action** menu when the M580 CPU is connected through the Control Expert **DTM Browser**.

Accessing Online Action

Follow these directions to access the **Online Action** settings for the M580 CPU:

Step	Action
1	Open the **DTM Browser** in Control Expert (**Tools > DTM Browser**).
2	Select the M580 DTM in the **DTM Browser**.
3	Connect the DTM to the Control Expert application (**Edit > Connect**).
4	Right-click the M580 DTM.
5	Scroll to the **Online Action** menu (**Device menu > Additional functions > Online Action**).
6	3 tabs appear: • **Ethernet/IP Objects** • **Port Configuration** • **Ping**

EtherNet/IP Objects

Displays object parameters value when available.

Click **Refresh** to update the displayed values.

Port Configuration

Configure and read the service port mode:

Field	Description
Service Port Mode	• **Access** (default) • **Mirroring** **NOTE:** This mode can also be set in the CPU configuration tabs, page 162.
Access Port Configuration	Displays the access port configuration information (refer to CPU configuration tabs, page 162).
Port Mirroring Configuration	Displays the port mirroring configuration (refer to CPU configuration tabs, page 162).

Ping

Field	Parameter	Description
Address	**IP Address**	Type the IP address to ping.
Ping	**Ping**	Click to ping the address set.
	Ping Result	Displays the ping result.
	Repeat (100ms)	Select this parameter to repeat ping if no reply is received.
	Stop on Error	Select this parameter to stop repeating ping if an error is detected when **Repeat (100ms)** is selected.
	Clear	Click to clear the **Ping Result** display.

EtherNet/IP Objects Tab

Introduction

Use the **EtherNet/IP Objects** tab in the **Online Action** window:

- Retrieve and display current data describing the state of CIP objects for the selected CPU or remote EtherNet/IP device.
- Reset the selected CPU or remote EtherNet/IP device.

Access the Page

Open the **EtherNet/IP Objects** tab:

Step	Action
1	Connect the DTM to the module (see Modicon M580, BMENOC0301/0311 Ethernet Communications Module, Installation and Configuration Guide).
2	Open the **Online Action** page (see Modicon M580, BMENOC0301/0311 Ethernet Communications Module, Installation and Configuration Guide).
3	Select the **EtherNet/IP Objects** tab.

Available CIP Objects

You can retrieve CIP objects according to the Control Expert operating mode:

Mode	Available CIP Objects
Standard	Identity object, page 219
Advanced	Identity object, page 219
	Connection Manager object, page 225
	TCP/IP Interface object, page 237
	Ethernet Link object (see Modicon M580, BMENOC0301/0311 Ethernet Communications Module, Installation and Configuration Guide)
	QoS object, page 230

Service Port Tab

Introduction

Use the **Service Port** tab in the **Online Action** window to view and edit communication port properties for a distributed EtherNet/IP device. Use this tab to execute these commands:

- *Refresh*: Use a Get command to retrieve port configuration settings from a distributed EtherNet/IP device.
- *Update*: Use a Set command to write all or selected edited values to the same distributed EtherNet/IP device

The configuration information on the **Service Port** tab is sent in EtherNet/IP explicit messages that employ the address and messaging settings configured for Ethernet/IP explicit messaging (below).

Access the Page

Open the **EtherNet/IP Objects** tab:

Step	Action
1	Connect the DTM to the module (see Modicon M580, BMENOC0301/0311 Ethernet Communications Module, Installation and Configuration Guide).
2	Open the **Online Action** page (see Modicon M580, BMENOC0301/0311 Ethernet Communications Module, Installation and Configuration Guide).
3	Select the **EtherNet/IP Objects** tab.
4	Configure the Service port with the instructions from the offline configuration (see Modicon M580, BMENOC0301/0311 Ethernet Communications Module, Installation and Configuration Guide).
5	Click the **Update** button to apply the new configuration.

Pinging a Network Device

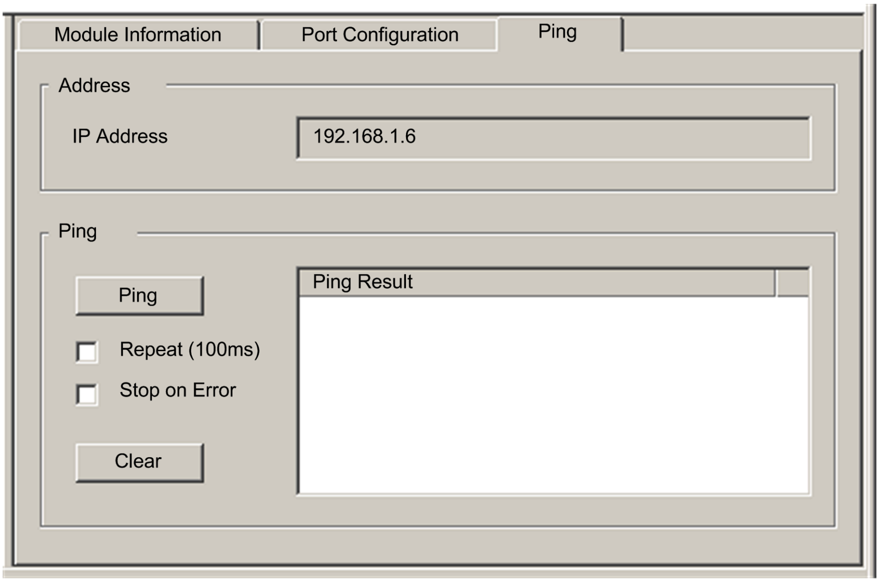
Overview

Use the Control Expert ping function to send an ICMP echo request to a target Ethernet device to determine:

- if the target device is present, and if so
- the elapsed time to receive an echo response from the target device

The target device is identified by its IP address setting. Enter only valid IP addresses in the **IP Address** field.

The ping function can be performed in the **Ping** page of the **Online Action** window:



Pinging a Network Device

Ping a network device:

Step	Action
1	In the **DTM Browser**, select the CPU upstream of the remote EtherNet/IP device you want to ping.
2	Right-click and select **Device Menu > Online Action**. **Result**: The **Online Action** window opens.

Step	Action
3	In the **Online Action** window, select the device you want to ping. **Result**: The window displays pages containing online information for the selected device. **NOTE:** The specific collection of displayed pages depends on the type of device selected: • the CPU • a remote EtherNet/IP device • a remote Modbus TCP device
4	Select the **Ping** page. To send... • a single ping: Deselect the **Repeat** checkbox. • a series of pings (1 every 100 ms): Select the **Repeat** checkbox.
5	(Optional) Select **Stop on Error** to stop pinging an unsuccessful communication.
6	Click **Ping** once to begin pinging.
7	Click **Ping** a second time to stop repeated pinging, where no error has been detected.
8	The **Ping Result** box displays the ping outcome. Click **Clear** to empty the **Ping Result** box.

Diagnostics Available through Modbus/TCP

Modbus Diagnostic Codes

Introduction

CPUs and BMENOC0301/BMENOC0311 communication modules in M580 systems support the diagnostic codes in these tables.

Function Code 3

Some module diagnostics (I/O connection, extended health, redundancy status, FDR server, etc.) are available to Modbus clients that read the local Modbus server area. Use Modbus function code 3 with the unit ID set to 100 for register mapping:

Type	Offset Modbus Address	Size (Words)
Basic Networks Diagnostic Data	0	39
Ethernet Port Diagnostics Data (Internal port)	39	103
Ethernet Port Diagnostics Data (**ETH 1**)	142	103
Ethernet Port Diagnostics Data (**ETH 2**)	245	103
Ethernet Port Diagnostics Data (**ETH 3**)	348	103
Ethernet Port Diagnostics Data (backplane)	451	103
Modbus TCP/Port 502 Diagnostic Data	554	114
Modbus TCP/Port 502 Connection Table Data	668	515
SNTP Diagnostics	1218	57
QoS Diagnostics	1275	11
Identify	2001	24

For a description of available function codes refer to the list of supported Modbus diagnostic codes in the topic *Modbus Diagnostic Codes* (see Quantum IEC61850, 140 NOP 850 00, Installation and Configuration Guide) in the *Quantum EIO Control Network Installation and Configuration Guide*.

Function Code 8, Subcode 21

Function Code 8, subcode 21 (decimal – 15 hex), provides information regarding the NTPv4 service and peers.

Operation Code (hex)	Description
0x77	Get NTP Service Status
0x78	Get NTP Peer Status

The structure of these operation codes are as follows:

Get NTP Service Status

Field	Length [bytes]	Value (hex)
Request and Response fields:		
Function Code	1	08
Sub Function Code Hi	1	00
Sub Function Code Low	1	15
Operation Code Hi	1	00
Operation Code Low	1	77
Response only fields:		
Byte Count	1	49
NTP Service • NTP Mode: Bits 0-3 • Status: Bits 4-7	1	NTP Mode: • 0x1: Client/Server • 0x2: Server Only NTP Status: • 0x1: Enable • 0x2:Disable
Sync	1	UINT (Leap Byte)
Stratum	1	UINT • Value = 16 Indicates KISS Code represented in the Reference ID field is ASCII. • Else, Reference ID field to be parsed as Hex IP address.
Precision	1	INT
Alarm	1	When Accuracy exceeds the user configured NTPv4 Threshold

Get NTP Service Status (Continued)

Field	Length [bytes]	Value (hex)
Accuracy	4	FLOAT (TIME_WITHIN)
Root Delay	4	FLOAT
Root Dispersion	4	FLOAT
Reference ID	4	UINT
Reference DATE_TIME-MICRO_ SEC	4	UINT
Clock DATE_TIME-MICRO_SEC	4	UINT
Peer	2	
DST Status	1	
Time Zone	4	
Time Zone Offset (minutes)	2	
Daylight Saving Time Bias (minutes)	1	
Daylight Saving Start Date - Month	1	
Daylight Saving Start Date - Week #, Day of Week **MS 4-Bits: Occurrence #** **(1 = 1ST Ooccurrence, 2 = 2ND Occurrence…, 5 = FIFTH OR LAST OCCURRENCE)** **LS 4-Bits: Day of the Week:** **(0 = Sunday…, 6 = Saturday)**	1	
Daylight Saving Start Time (Seconds elapsed from midnight)	4	
Daylight Saving End Date – Month	1	
Daylight Saving End Date – Week #, Day of Week	1	
Daylight Saving End Time (Seconds elapsed from midnight)	4	

Get NTP Peer Status

Field	Length [bytes]	Value (hex)
Request and Response fields:		
Function Code	1	08

Get NTP Peer Status (Continued)

Field	Length [bytes]	Value (hex)
Sub Function Code Hi	1	00
Sub Function Code Low	1	15
Operation Code Hi	1	00
Operation Code Low	1	75
Byte Count	1	F9
Peer Count	1	Default - 8
FLOAT Precision	1	For the FLOAT values, below
Response only fields (The following fields repeat, with the suffix # incarnated, for each system peer):		
Remote IP 1	4	Remote IP Address
Reference ID 1	4	• If Stratum = 16, this field is interpreted as 4 Bytes ASCII. • Else, the field is parsed as an IPv4 Address.
Select 1	1	The currently selected server: • 0X0: Default • 0X1: Current • 0X2: Candidate
Reach Percentage 1	1	Percentage Representation (0-100%)
Stratum 1	1	Least value determines current/candidate Server IP. If value = 16, then Ref ID field is parsed as 4 bytes ASCII.
Poll 1	2	INT
Delay 1	4	FLOAT
Offset 1	4	FLOAT
Jitter 1	4	FLOAT
When	6	6 byte ASCII. sec/min/hr since last received packet

Function Code 8, Subcode 22

Modbus function code 08, subcode 22, provides a variety of diagnostic functions:

Operation Code	Diag. Control	Description
0x01	0x0100	network diagnostic data
	0x0200	Read the Ethernet port diagnostic data from the switch manager.
	0x0300	Read the Modbus TCP/port 502 diagnostic data from the Modbus server.
	0x0400	Read the Modbus TCP/port 502 connection table from the Modbus server.
	0x07F0	Read the data structure offset data from the Modbus server.
0x02	0x0100	Clear the basic network diagnostic data. **NOTE:** Only specific parameters of basic network diagnostic data are used to clear requests.
	0x0200	Clear the Ethernet port diagnostic data. **NOTE:** Only specific parameters of basic network diagnostic data are used to clear requests.
	0x0300	Clear the Modbus TCP/port 502 diagnostic data. **NOTE:** Only specific parameters of Modbus port 502 diagnostic data are used to clear requests.
	0x0400	Clear the Modbus TCP/port 502 connection table. **NOTE:** Only specific parameters of Modbus port 502 connection data are use to clear requests.
0x03	0	Clear all diagnostic data. **NOTE:** Only specific parameters of each diagnostic data are used to clear requests.

Read Device Identification

Modbus function code 43, subcode 14: A Modbus request associated with function code 43 (Read Device Identification) asks a Modbus server to return the vendor name, product name, version number, and other optional fields:

Category	Object ID	Object Name	Type	Requirement
Basic	0x00	VendorName (vendor name)	ASCII string	mandatory
	0x01	ProductCode (product code)	ASCII string	mandatory
	0x02	MajorMinorRevision (version number)	ASCII string	mandatory
Regular	0x03	VendorUrl (vendor URL)	ASCII string	optional
	0x04	ProductName (product name)	ASCII string	optional
	0x05	ModelName (model name)	ASCII string	optional
	0x06	UserApplicationName (user application name)	ASCII string	optional
	0x07...0x7F	(reserved)	ASCII string	optional
Extended	0x80...0xFF	device-dependent		optional

This table provides sample responses to the Modbus request (function code 43, subcode 14):

Module	0x00 Vendor ID	0x01 Part Number	0x02 Version
BMEP584020 CPU	Schneider Electric	BMEP584020	v02.10
BMENOC0301 module	Schneider Electric	BMENOC0301	V02.04 build 0009
BMENOC0311 module	Schneider Electric	BMENOC0311	V02.04 build 0009
BMENOC0321 module	Schneider Electric	BMENOC0321	V01.01 build 0004

Diagnostics Available through EtherNet/IP CIP Objects

Introduction

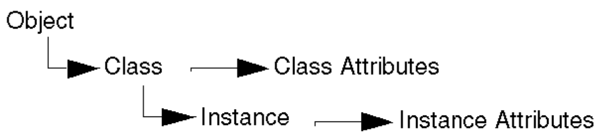
Modicon M580 applications use CIP within a producer/consumer model to provide communication services in an industrial environment. This section describes the available CIP objects for diagnostics of Modicon M580 CPU modules.

About CIP Objects

Overview

The Ethernet communication module can access CIP data and services located in connected devices. The CIP objects and their content depend on the design of each device.

CIP object data and content are exposed—and accessed—hierarchically in the following nested levels:



NOTE: You can use explicit messaging to access these items:

- Access a collection of instance attributes by including only the class and instance values for the object in the explicit message.
- Access a single attribute by adding a specific attribute value to the explicit message with the class and instance values for the object.

This chapter describes the CIP objects that the Ethernet communication module exposes to remote devices.

Identity Object

Overview

The Identity object presents the instances, attributes and services described below.

Class ID

01

Instance IDs

The Identity object presents two instances:

- 0: class
- 1: instance

Attributes

Identity object attributes are associated with each instance, as follows:

Instance ID = 0 (class attributes):

Attribute ID	Description	GET	SET
01	Revision	X	—
02	Max Instance	X	—
X = supported — = not supported			

Instance ID = 1 (instance attributes):

Attribute ID		Description	Type	GET	SET
hex	dec				
01	01	Vendor ID	UINT	X	—
02	02	Device Type	UINT	X	—
03	03	Product Code	UINT	X	—
04	04	Revision	STRUCT	X	—

Attribute ID		Description	Type	GET	SET
hex	dec				
		Major	USINT		
		Minor	USINT		
05	05	Status bit 2: 0x01=the module is configured bits 4-7: 0x03=no I/O connections established 0x06=at least 1 I/O connection in run mode 0x07=at least 1 I/O connection established, all in IDLE mode	Word	X	—
06	06	Serial Number	UDINT	X	—
07	07	Product Name	STRING	X	—
18	24	Modbus Identity	STRUCT	X	—
X = supported — = not supported					

Services

The Identity object performs the following services upon the listed object types:

Service ID		Description	Class	Instance	Notes
hex	dec				
01	01	Get_Attributes_All	X	X	Returns: • all class attributes (instance = 0) • instance attributes 1 to 7 (instance = 1)
0E	14	Get_Attribute_Single	X	X	Returns the value of the specified attribute.
X = supported — = not supported					

Message Router Object

Overview

The Message Router object provides a messaging connection point through which a client may address a service to any object class or instance residing in the physical device.

Class ID

02 (hex and decimal)

Instance IDs

The Message Router object presents two instances:

- 0: class
- 1: instance

Attributes

Message Router object attributes are associated with each instance, as follows:

Instance ID = 0 (class attributes):

Attribute ID (hex and dec)	Description	GET	SET
01	Revision	X	—
02	Maximum Instance	X	—
03	Number of Instances	X	—
04	Optional Attribute List	X	—
05	Optional Service List	X	—
06	Maximum Number of Class Attributes	X	—
07	Maximum Number of Instance Attributes	X	—
X = supported — = not supported			

Instance ID = 1 (instance attributes):

Attribute ID		Description	Type	GET	SET	Value
hex	dec					
01	01	Object list	STRUCT of	X	—	A list of supported objects (i.e. a structure with an array of object class codes supported by the device)
		Number	UINT	X	—	The number of supported classes (i.e. class codes) in the classes array
		Classes	Array of UINT	X	—	List of supported class codes supported by the device
02	02	Number Available	UINT	X	—	Maximum number of connections supported
03	03	Number Active	UINT	X	—	Number of connections allocated to system communication
04	04	Active Connections	Array of UINT	X	—	A list of the system connection IDs of the active connections
X = supported — = not supported						

Services

The Message Router object performs the following services upon the listed object types:

Service ID		Description	Class	Instance	Notes
hex	dec				
01	01	Get_Attributes_All	X	X	Returns: • all class attributes (instance = 0) • instance attributes 1 to 7 (instance = 1)
0E	14	Get_Attribute_Single	X	X	Returns the value of the specified attribute.
X = supported — = not supported					

Assembly Object

Overview

The assembly object consists of the attributes and services. Assembly instances exist only when you configure local slaves, page 396 for the M580 CPU modules.

You can send an explicit message to the assembly object only when no other connections have been established that read from or write to this object. For example, you can send an explicit message to the assembly object if a local slave instance is enabled, but no other module is scanning that local slave.

Class ID

04

Instance IDs

The assembly object presents these instance identifiers:

- 0: class
- 101, 102, 111, 112, 121, 122: instance

Attributes

The assembly object consists of these attributes:

Instance ID = 0 (class attributes):

Attribute ID	Description	GET	SET
01	Revision	X	—
02	Max Instance	X	—
03	Number of Instances	X	—
X = supported — = not supported			

Instance attributes:

Instance ID	Attribute ID	Description	Type	GET	SET
101	03	Local slave 1: T->O (output data)	Array of BYTE	X	—
102		Local slave 1: O>T (input data)	Array of BYTE	X	—
111	03	Local slave 2: T->O (output data)	Array of BYTE	X	—
112		Local slave 2: O>T (input data)	Array of BYTE	X	—
X = supported — = not supported					

Services

The CIP assembly object performs these services upon the listed object types:

Service ID		Description	Class	Instance	Notes
hex	dec				
0E	14	Get_Attribute_Single	X	X	Returns the value of the specified attribute
X = supported — = not supported					
1. When valid, the size of the data written to the assembly object using the Set_Attribute_Single service equals the size of the assembly object as configured in the target module.					

Connection Manager Object

Overview

The Connection Manager object presents the instances, attributes and services described below.

Class ID

06

Instance IDs

The Connection Manager object presents two instance values:

- 0: class
- 1: instance

Attributes

Connection Manager object attributes are associated with each instance, as follows:

Instance ID = 0 (class attributes):

Attribute ID	Description	GET	SET
01	Revision	X	—
02	Max Instance	X	—
X = supported — = not supported			

Instance ID = 1 (instance attributes):

Attribute ID		Description	Type	GET	SET	Value
hex	dec					
01	01	Open Requests	UINT	X	X	Number of Forward Open service requests received
02	02	Open Format Rejects	UINT	X	X	Number of Forward Open service requests that were

Attribute ID		Description	Type	GET	SET	Value
hex	dec					
						rejected due to incorrect format
03	03	Open Resource Rejects	UINT	X	X	Number of Forward Open service requests that were rejected due to lack of resources
04	04	Open Other Rejects	UINT	X	X	Number of Forward Open service requests that were rejected for reasons other than incorrect format or lack of resources
05	05	Close Requests	UINT	X	X	Number of Forward Close service requests received
06	06	Close Format Requests	UINT	X	X	Number of Forward Close service requests that were rejected due to incorrect format
07	07	Close Other Requests	UINT	X	X	Number of Forward Close service requests that were rejected for reasons other than incorrect format
08	08	Connection Timeouts	UINT	X	X	Total number of connection timeouts that occurred in connections controlled by this connections manager
09	09	Connection Entry List	STRUCT	X	—	0 (Unsupported optional item
0B	11	CPU_Utilization	UINT	X	—	0 (Unsupported optional item
0C	12	MaxBuffSize	UDINT	X	—	0 (Unsupported optional item
0D	13	BufSize Remaining	UDINT	X	—	0 (Unsupported optional item
X = supported — = not supported						

Services

The Connection Manager object performs the following services on the listed object types:

Service ID		Description	Class	Instance	Notes
hex	dec				
01	01	Get_Attributes_All	X	X	Returns the value of all attributes.
0E	14	Get_Attribute_Single	X	X	Returns the value of the specified attribute.
X = supported — = not supported					

Modbus Object

Overview

The Modbus object converts EtherNet/IP service requests to Modbus functions, and Modbus exception codes to CIP General Status codes. It presents the instances, attributes and services described below.

Class ID

44 (hex), 68 (decimal)

Instance IDs

The Modbus object presents two instance values:

- 0: class
- 1: instance

Attributes

The Modbus object consists of the following attributes:

Instance ID = 0 (class attributes):

Attribute ID	Description	GET	SET
01	Revision	X	—
02	Max Instance	X	—
X = supported — = not supported			

Instance ID = 1 (instance attributes):

Attribute ID	Description	Type	GET	SET
—	No instance attributes are supported	—	—	—

Services

The Modbus object performs the following services upon the listed object types:

Service ID		Description	Class	Instance
hex	dec			
0E	14	Get_Attribute_Single	X	X
4B	75	Read_Discrete_Inputs	—	X
4C	76	Read_Coils	—	X
4D	77	Read_Input_Registers	—	X
4E	78	Read_Holding_Registers	—	X
4F	79	Write_Coils	—	X
50	80	Write_Holding_Registers	—	X
51	81	Modbus_Passthrough	—	X
X = supported — = not supported				

Quality Of Service (QoS) Object

Overview

The QoS object implements Differentiated Services Code Point (DSCP or *DiffServe*) values for the purpose of providing a method of prioritizing Ethernet messages. The QoS object presents the instances, attributes and services described below.

Class ID

48 (hex), 72 (decimal)

Instance IDs

The QoS object presents two instance values:

- 0: class
- 1: instance

Attributes

The QoS object consists of the following attributes:

Instance ID = 0 (class attributes):

Attribute ID	Description	GET	SET
01	Revision	X	—
02	Max Instance	X	—
X = supported — = not supported			

Instance ID = 1 (instance attributes):

Attribute ID	Description	Type	GET	SET	Value
04	DSCP Urgent	USINT	X	X	For CIP transport class 0/1 Urgent priority messages.
05	DSCP Scheduled	USINT	X	X	For CIP transport class 0/1 Urgent priority messages.

Attribute ID	Description	Type	GET	SET	Value
06	DSCP High	USINT	X	X	For CIP transport class 0/1 Urgent priority messages.
07	DSCP Low	USINT	X	X	For CIP transport class 0/1 Urgent priority messages.
08	DSCP Explicit	USINT	X	X	For CIP explicit messages (transport class 2/3 and UCMM).
X = supported — = not supported					

NOTE: A change in the instance attribute value takes effect on device re-start, for configurations made from flash memory.

Services

The QoS object performs the following services upon the listed object types:

Service ID		Description	Class	Instance
hex	dec			
0E	14	Get_Attribute_Single	X	X
10	16	Set_Attribute_Single	—	X
X = supported — = not supported				

Port Object

Overview

The Port object describes the communication interfaces that exist on the device and that are visible to CIP.

Class ID

F4 (hex), 244 (decimal)

Instance IDs

The Port object presents two instances:

- 0: class
- 1: instance

Attributes

Port object attributes are associated with each instance, as follows:

Instance ID = 0 (class attributes):

Attribute ID (hex and dec)	Description	GET	SET
01	Revision	X	—
02	Maximum Instance	X	—
03	Number of Instances	X	—
04	Optional Attribute List	X	—
05	Optional Service List	X	—
06	Optional Maximum Number of Class Attributes	X	—
07	Optional Maximum Number of Instance Attributes	X	—
08	Entry Port Returns the instance of the Port object that describes the port through which this request entered the device	X	—

Attribute ID (hex and dec)	Description	GET	SET
09	Port Instance Information Array of structures containing instance attributes 1 and 2 (see below) from each port instance	X	—
	Port Type (see Instance attribute 01)	X	—
	Port Number (see Instance attribute 02)	X	—
X = supported — = not supported			

Instance ID = 1 (instance attributes):

Attribute ID		Description	Type	GET	SET	Value
hex	dec					
01	01	Port Type.	UINT	X	—	• 0: Routing not supported • 1: Vendor specific • 2: ControlNet • 3: ControlNet Redundant • 4: EtherNet/IP (formerly TCP/IP) • 5: DeviceNet • 6-199: Vendor specific • 200: CompoNet • 201: Modbus/TCP • 202: Modbus/SL • 203: SERCOS III • 204: HART • 205: IO-Link • 206-65535: Reserved
02	02	Port Number		X	—	The CIP number
03	03	Logical Link Object	STRUCT of	X	—	A list of supported objects (i.e. a structure with an array of object class codes supported by the device)
		Path Length	UINT	X	—	The number of 16-bit words in the following path.
		Link Path	Padded EPATH	X	—	Logical path segments that identify the object for this port.

Attribute ID		Description	Type	GET	SET	Value
hex	dec					
04	04	Port Name	SHORT_ STRING	X	—	String name of port interface name, up to 64 characters
05	05	Port Type Name	SHORT_ STRING	X	—	String name of port interface type, up to 64 characters
06	06	Port Description	SHORT_ STRING	X	—	String that describes the port
07	07	Port Number and Node Address	Padded EPATH	X	—	A single port segment containing the Port Number of this port and the Link Address of this device on this port.
08	08	Port Node Range	STRUCT of	X	—	
		Minimum Node Number	UINT	X	—	For example, on port.
		Maximum Node Number	UINT	X	—	For example, on port.
09	09	Chassis Identity	Packed EPATH	X	—	Electronic key of the chassis to which this port is attached. This attribute is a single Logical Electronic Key Segment with Format 4 of the Logical Electronic Key segment.
A	10	Port Routing Capabilities	DWORD	X	—	Bit string defining the routing capabilities of this port, where 0= not-supported, 1=supported: • bit 0: Incoming unconnected messages • bit 1: Outgoing unconnected messages • bit 2: Incoming transport class 0/1 connections • bit 3: Outgoing transport class 0/1 connections • bit 4: Incoming transport class 2/3 connections • bit 5: Outgoing transport class 2/3 connections • bit 6: Outgoing DeviceNet CIP Safety connections (only for DeviceNet ports) • bits 7-31: Reserved

Attribute ID		Description	Type	GET	SET	Value
hex	dec					
B	11	Associated Communication Objects	STRUCT of	X	—	List of communication object instances associated with this instantiated Port Object (see list, below)
		Number of entries in following Array:	USINT	X	—	
			Array of STRUCT of	X	—	
		Number of 16 bit words in the following path	USINT	X	—	
		Logical path segments that identify an associated communication object instance	Padded EPATH	X	—	
X = supported — = not supported						

The list of Associated Communication Objects in Attribute 11 (dec) / B (hex) includes:

DeviceNet Object – 0x03	RSTP Port Object – 0x55	TCP/IP Interface Object – 0xF5
Modbus Object – 0x44	Parallel Redundancy Protocol Object – 0x56	Ethernet Link Object – 0xF6
Modbus Serial Link Object – 0x46	PRP Nodes Table Object – 0x57	0xF6 • CompoNet Link Object – 0xF7
Device Level Ring Object – 0x47	EtherNet/IP Security Object – 0x5E	CompoNet Repeater Object – 0xF8
QoS Object – 0x48	ControlNet Object – 0xF0	CompoNet Repeater Object – 0xF8
SERCOS III Link Object – 0x4C	ControlNet Keeper Object – 0xF1	IO-Link Master PHY Object – 0x10C
RSTP Bridge Object – 0x54	ControlNet Scheduling Object – 0xF2	—

Services

The port object performs the following services upon the listed object types:

Service ID		Description	Class	Instance	Notes
hex	dec				
01	01	Get_Attributes_All	X	X	Returns: • all class attributes (instance = 0) • instance attributes 1 to 7 (instance = 1)
10	10	Set_Attribute_Single	—	X	Modifies an attribute..
0E	14	Get_Attribute_Single	X	X	Returns the value of the specified attribute.
X = supported — = not supported					

TCP/IP Interface Object

Overview

The TCP/IP interface object presents the instances (per network), attributes and services described below.

Class ID

F5 (hex), 245 (decimal)

Instance IDs

The TCP/IP interface object presents 2 instance values:

- 0: class
- 1: instance

Attributes

TCP/IP interface object attributes are associated with each instance, as follows:

Instance ID = 0 (class attributes):

Attribute ID	Description	GET	SET
01	Revision	X	—
02	Max Instance	X	—
X = supported — = not supported			

Instance ID = 1 (instance attributes):

Attribute ID	Description	Type	GET	SET	Value
01	Status	DWORD	X	—	0x01
02	Configuration Capability	DWORD	X	—	0x01 = from BootP 0x11 = from flash 0x00 = other

Attribute ID	Description	Type	GET	SET	Value
03	Configuration Control	DWORD	X	X	0x01 = out-of-box default
04	Physical Link Object	STRUCT	X	—	
	Path Size	UINT			
	Path	Padded EPATH			
05	Interface Configuration	STRUCT	X	X	0x00 = out-of-box default
	IP Address	UDINT			
	Network Mask	UDINT			
	Gateway Address	UDINT			
	Name Server	UDINT			
	Name Server 2	UDINT			
	Domain Name	STRING			
06	Host Name	STRING	X	—	
X = supported — = not supported					

Services

The TCP/IP interface object performs the following services upon the listed object types:

Service ID		Description	Class	Instance	Notes
hex	dec				
01	01	Get_Attributes_All	X	X	Returns the value of all attributes.
0E	14	Get_Attribute_Single	X	X	Returns the value of the specified attribute.
10	16	Set_Attribute_Single[1]	—	X	Sets the value of the specified attribute.
X = supported — = not supported					
1. The Set_Attribute_Single service can execute only when these preconditions are satisfied: • Configure the Ethernet communication module to obtain its IP address from flash memory. • Confirm that the PLC is in stop mode.					

Ethernet Link Object

Overview

The Ethernet Link object consists of the instances, attributes, and services described below.

Class ID

F6 (hex), 246 (decimal)

Instance IDs

The Ethernet Link object presents these instance values:

- 101: backplane slot 1
- 102: backplane slot 2
- 103: backplane slot 3
- ...
- 112: backplane slot 12
- 255: internal port

Attributes

The Ethernet Link object presents the following attributes:

Instance ID = 0 (class attributes):

Attribute ID	Description	GET	SET
01	Revision	X	—
02	Max Instance	X	—
03	Number of Instances	X	—
X = supported — = not supported			

Instance ID = 1 (instance attributes):

Attribute ID		Description	Type	GET	SET	Value
hex	dec					
01	01	Interface Speed	UDINT	X	—	Valid values: 0, 10, 100.
02	02	Interface Flags	DWORD	X	—	Bit 0: link status 0 = Inactive 1 = Active
						Bit 1: duplex mode 0 = half duplex 1 = full duplex
						Bits 2...4: negotiation status 3 = successfully negotiated speed and duplex 4 = forced speed and link
						Bit 5: manual setting requires reset 0 = automatic 1 = device need reset
						Bit 6: local hardware detected error 0 = no event 1 = event detected
03	03	Physical Address	ARRAY of 6 USINT	X	—	module MAC address
04	04	Interface Counters	STRUCT	X	—	
		In octets	UDINT			octets received on the interface
		In Ucast Packets	UDINT			unicast packets received on the interface
		In NUcast Packets	UDINT			non-unicast packets received on the interface
		In Discards	UDINT			inbound packets received on the interface, but discarded
		In Errors	UDINT			inbound packets with detected errors (does not include in discards)
		In Unknown Protos	UDINT			inbound packets with unknown protocol
		Out Octets	UDINT			octets sent on the interface
		Out Ucast Packets	UDINT			unicast packets sent on the interface
		Out NUcast Packets	UDINT			non-unicast packets sent on the interface

Attribute ID		Description	Type	GET	SET	Value
hex	dec					
		Out Discards	UDINT			outbound packets discarded
		Out Errors	UDINT			outbound packets with detected errors
05	05	Media Counters	STRUCT	X	—	
		Alignment Errors	UDINT			frames that are not an integral number of octets in length
		FCS Errors	UDINT			CRC error — frames received do not pass the FCS check
		Single Collisions	UDINT			successfully transmitted frames that experienced exactly 1 collision
		Multiple Collisions	UDINT			successfully transmitted frames that experienced more than 1 collision
		SQE Test Errors	UDINT			number of times the detected SQE test error is generated
		Deferred Transmissions	UDINT			frames for which first transmission attempt is delayed because the medium is busy
		Late Collisions	UDINT			number of times a collision is detected later than 512 bit times into the transmission of a packet
		Excessive Collisions	UDINT			frames that do not transmit due to excessive collisions
		MAC Transmit Errors	UDINT			frames that do not transmit due to a detected internal MAC sublayer transmit error
		Carrier Sense Errors	UDINT			times that the carrier sense condition was lost or not asserted when attempting to transmit a frame
		Frame Too Long	UDINT			frames received that exceed the maximum permitted frame size
		MAC Receive Errors	UDINT			frames not received on an interface due to a detected internal MAC sublayer receive error
06	06	Interface Control	STRUCT	X	—	API of the connection
		Control Bits	WORD			Bit 0: Auto-negotiation disabled (0) or enabled (1). **NOTE:** When auto-negotiation is enabled, 0x0C (object state conflict) is returned when attempting to set either: • forced interface speed • forced duplex mode

Attribute ID		Description	Type	GET	SET	Value
hex	dec					
						Bit 1: forced duplex mode (if auto-negotiation bit = 0) 0 = half duplex 1 = full duplex
		Forced Interface Speed	UINT			Valid values include 10000000 and 100000000. **NOTE:** Attempting to set any other value returns the detected error 0x09 (invalid attribute value).
10	16	Interface Label	SHORT_ STRING	X	—	A fixed textual string identifying the interface, that should include 'internal' for internal interfaces. Maximum number of characters is 64.
X = supported — = not supported						

Services

The Ethernet Link object performs the following services upon the listed object types:

Service ID		Description	Class	Instance
hex	dec			
01	01	Get_Attributes_All	X	X
10	16	Set_Attribute_Single	—	X
0E	14	Get_Attribute_Single	X	X
4C	76	Get_and_Clear	—	X
X = supported — = not supported				

Module Diagnostic Object

Overview

The Module Diagnostic object presents the instances, attributes and services described below.

Class ID

300 (hex), 768 (decimal)

Instance IDs

The Module Diagnostic object presents two instances:

- 0: class
- 1: instance

Attributes

Module Diagnostic object attributes are associated with each instance, as follows:

Instance ID = 0 (class attributes):

Attribute ID	Description	GET	SET
01	Revision	X	—
02	Maximum Instance	X	—
X = supported — = not supported			

Instance ID = 1 (instance attributes):

Attribute ID		Description	Type	GET	SET	Value
hex	dec					
01	01	Module Status	WORD	X	—	• 0x01 = STARTED • 0x02 = STOPPED • 0x03 = RUNNING
02	02	CNF Version	WORD	X	—	
03	03	CRC	UDINT	X	—	
04	04	Connection Status	STRUCT of	X	—	
		Size Table	WORD			In bytes -16 bytes
		Table	WORD[ ]			Padded on word • Describes I/O connections. • Each bit describes one I/O connection – the first bit is the first I/O connection. • Value 1 indicates that INPUT and OUTPUT status of an I/O connection are OK (status equal to 0). • Value 0 indicates that INPUT and OUTPUT status of an I/O connection are not OK (status not equal to 0). • The table consists of 8 words (128 I/O connections).
05	05	CCO Mode	WORD	X	—	• 0x00 = Block access to connection configuration object (CCO) • 0x01 = STOPPED
X = supported — = not supported						

Services

The Module Diagnostic object performs the following services upon the listed object types:

Service ID		Description	Class	Instance	Notes
hex	dec				
01	01	Get_Attributes_All	X	X	Returns the value of all attributes.
10	16	Set_Attribute_Single	—	X	Sets the value of the specified attribute.
X = supported — = not supported					

Scanner Diagnostic Object

Overview

The Scanner Diagnostic object presents the instances, attributes and services described below.

Class ID

301 (hex), 769 (decimal)

Instance IDs

The Scanner Diagnostic object presents two instances:

- 0: class
- 1: instance

Attributes

Scanner Diagnostic object attributes are associated with each instance, as follows:

Instance ID = 0 (class attributes):

Attribute ID	Description	GET	SET
01	Revision	X	—
02	Maximum Instance	X	—
X = supported — = not supported			

Instance ID = 1 (instance attributes):

Attribute ID		Description	Type	GET	SET	Value
hex	dec					
01	01	Control Bits	WORD	X	X	• TRUE = Activate checking time for production and consumption • FALSE = Inactive (default)
02	02	ST_DIAG_CNT	STRUCT of	X	X	
		wErrFrameCnt	UINT			Incremented each time a frame is not sent for lack of resources or was impossible to send.
		wErrTimeOutCnt	UINT			Incremented when one connection is timed out.
		wErrRefusedCnt	UINT			Incremented when one connection is refused by the remote station.
		dwProdCnt	UDINT			Incremented at each production.
		dwConsCnt	UDINT			Incremented at each consumption.
		dwProdByteCnt	UDINT			Total bytes produced.
		dwConsByteCnt	UDINT			Total bytes consumed.
03	03	Input Status	WORD	X	—	See below.
04	04	Output Status	WORD	X	—	See below.
05	05	ST_LINK	STRUCT of	X	—	
		CIP Status	UINT			See below.
		Extended Status	UINT			See below.
		Production Connection ID	DWORD			
		Consumed Connection ID	DWORD			
		OtoT API	UDINT			API of the Connection
		TtoO API (API of the Connection)	UDINT			API of the Connection
		OtoT RPI (RPI of the Connection)	UDINT			RPI of the Connection
		TtoO RPI (RPI of the Connection)	UDINT			RPI of the Connection

Attribute ID		Description	Type	GET	SET	Value
hex	dec					
06	06	ST_SOCK_PARAM	STRUCT of	X	—	
		IpSockId	DWORD			Internal identifier
		IpForeign	DWORD			Remote station IP
		wPortForeign	UINT			Remote station port number
		IpLocal	DWORD			Local station IP
		wPortLocal	UINT			Local station port number
07	07	ST_PRODUCTION	STRUCT of	X	—	
		bValid	WORD			• 0 = STRUCT production data is not valid • 1 = STRUCT production data is valid
		dwCurrentTime	UDINT			Internal: number of ticks before next production
		dwProductionTime	UDINT			Internal: number of ticks between production
		SequenceNumber	UDINT			Number of the sequence in the production
		stCheckTime	STRUCT of			
		dwLastTime	UDINT			Internal use
		dwMaxTime	UDINT			Maximum time between productions
		dwMinTime	UDINT			Minimum time between productions
		dwRPI	UDINT			Connection API
		wOverRun	UINT			Number of times the production was too long
		wUnderRun	UINT			Number of times the production was too fast
		dwCurrentTime	UDINT			Internal use

Attribute ID		Description	Type	GET	SET	Value
hex	dec					
08	08	ST_ CONSUMPTION	STRUCT of	X	—	
		bValid	WORD			• 0 = STRUCT consumption data is not valid • 1 = STRUCT consumption data is valid
		dwCurrentTime	UDINT			Internal: number of ticks before timeout
		dwConsumption-Time	UDINT			Internal: number of ticks of the timeout
		SequenceNumber	UDINT			Number of the sequence in the consumption
		stCheckTime	STRUCT of			
		dwLastTime	UDINT			Internal use
		dwMaxTime	UDINT			Maximum time between consumptions
		dwMinTime	UDINT			Minimum time between consumptions
		dwRPI	UDINT			Connection API
		wOverRun	UINT			Number of times the consumption was too long
		wUnderRun	UINT			Number of times the consumption was too fast
		dwCurrentTime	UDINT			Internal use
09	09	CCO Status	STRUCT of	X	—	Status of the Connection Configuration Object – see below
		byGeneralStatus	BYTE			
		byReserved	BYTE			
		Extended	WORD			
X = supported — = not supported						

Status values for the Scanner Diagnostic object:

Status	Description	CIP Status	Extended	Context
0	OK	0	0	The IO data are correctly exchanged
33	Time-Out	0xFB	0xFB0B	Timeout detected on consumption

Status	Description	CIP Status	Extended	Context
53	IDLE	0	0	An IDLE notification is received
54	Connection established	0	0	The connection is established, but the IO data are not consumed yet
		0xFB	0xFB08	Impossible to start the production
		0xFB	0xFB09	Impossible to start the consumption
		0xFB	0xFB0A	Not enough resources to manage the connection
58	Not connected (TCP)	0xFE	TCP Error	Error on TCP connection
65	Not connected (CIP)	status	extended	The Fw_Open response indicates a detected error.
		0xFB	0xFB01	Timeout for Fw_Open response
		0xFB	0xFB02	Incorrect format of the Fw_Open response (so addr)
		0xFB	0xFB03	Incorrect parameters in the response (OT Net Par)
		0xFB	0xFB04	Incorrect parameters in the response (TO Net Par)
		0xFB	0xFB05	Asking port number different than 2222
		0xFB	0xFB06	Error in joining the UDP multicast group
		0xFB	0xFB07	Optimization error / indeterminable MAC address
68	Connection establishing	0xD0	0x0001	Connection is closed
		0xD0	0x0002	Connection is pending
70	Not connected (EPIC)	0xFD	Status	Error code in register session response
		0xFD	Status	Error code in the frame
		0xFD	Status	Encapsulation session unregistered
77	Scanner stopped	0	0	Connection is stopped

Services

The Scanner Diagnostic object performs the following services upon the listed object types:

Service ID		Description	Class	Instance	Notes
hex	dec				
01	01	Get_Attributes_All	X	X	Returns the value of all attributes.
61	97	Get_Output	—	X	Returns the status and value the output: • Offset 0 / UINT / Status • Offset 2 / USINT [0...409] / Output data
62	98	Get_Input	—	X	Returns the status and value the input: • Offset 0 / UINT / Status • Offset 2 / USINT [0...409] / Intput data
63	99	Set_DiagCounters	—	X	Sets the value of ST_Diag_CNT to 0..
X = supported — = not supported					

NOTE: If a service is addressed on an instance that does not exist or is not an I/O connection for the scanner, the service detects the following error: 0x05 – Path destination unknown.

Adapter Diagnostic Object

Overview

The Adapter Diagnostic object presents the instances, attributes and services described below.

Class ID

302 (hex), 770 (decimal)

Instance IDs

The Adapter Diagnostic object presents two instances:

- 0: class
- 1: instance

Attributes

Adapter Diagnostic object attributes are associated with each instance, as follows:

Instance ID = 0 (class attributes):

Attribute ID	Description	GET	SET
01	Revision	X	—
02	Maximum Instance	X	—
X = supported — = not supported			

Instance ID = 1 (instance attributes):

Attribute ID		Description	Type	GET	SET	Value
hex	dec					
01	01	Control Bits	WORD	X	—	• 0 = Deactivate (default) • 1 = Activate checking time for production and consumption.
02	02	ST_DIAG_CNT	STRUCT of	X	—	
		wErrFrameCnt	UINT			Incremented each time a frame is not sent for lack of resources or was impossible to send.
		wErrTimeOutCnt	UINT			Incremented when one connection is timed out.
		wErrRefusedCnt	UINT			Incremented when one connection is refused by the remote station.
		dwProdCnt	UDINT			Incremented at each production
		dwConsCnt	UDINT			Incremented at each consumption
		dwProdByteCnt	UDINT			Total bytes produced
		dwConsByteCnt	UDINT			Total bytes consumed
03	03	Input Status	WORD	X	—	See below.
04	04	Output Status	WORD	X	—	See below.
05	05	ST_LINK	STRUCT of	X	—	
		CIP Status	UINT			See below.
		Extended Status	UINT			See below.
		Production Connection ID	DWORD			
		Consumed Connection ID	DWORD			
		OtoT API	UDINT			API of the connection
		TtoO API	UDINT			API of the connection
		OtoT RPI	UDINT			RPI of the connection
		TtoO RPI	UDINT			RPI of the connection

Attribute ID		Description	Type	GET	SET	Value
hex	dec					
06	06	ST_SOCK_ PARAM	STRUCT of	X	—	
		IpSockId	DWORD			Internal Identifier
		IpForeign	DWORD			Remote station IP
		wPortForeign	UINT			Remote station port number
		IpLocal	DWORD			Local station IP
		wPortLocal	UINT			Local station port number
07	07	ST_ PRODUCTION	STRUCT of	X	—	
		bValid	WORD			• 0 = STRUCT production data is not valid. • 1 = STRUCT production data is valid
		dwCurrentTime	UDINT			Internal – Number of ticks before next production
		dwProduction-Time	UDINT			Internal – Number of ticks between production
		SequenceNum-ber	UDINT			Number of the sequence in the production
		stCheckTime	STRUCT of			
		dwLastTime	UDINT			Internal use
		dwMaxTime	UDINT			Maximum time between two productions
		dwMinTime	UDINT			Minimum time between two productions
		dwRPI	UDINT			API of the connection
		wOverRun	UINT			Number of times the production was too long
		wUnderRun	UINT			Number of times the production was too fast
		dwCurrentTime	UDINT			Internal use

Attribute ID		Description	Type	GET	SET	Value
hex	dec					
08	08	ST_ CONSUMPTION	STRUCT	X	—	
		bValid	WORD			• 0 = STRUCT consumption data is not valid. • 1 = STRUCT consumption data is valid
		dwCurrentTime	UDINT			Internal – Number of ticks before timeout
		dwconsumption-Time	UDINT			Internal – Number of ticks of the timeout
		SequenceNum-ber	UDINT			Number of the sequence in the consumption
		stCheckTime	STRUCT			
		dwLastTime	UDINT			Internal use
		dwMaxTime	UDINT			Maximum time between two consumptions
		dwMinTime	UDINT			Minimum time between two consumptions
		dwRPI	UDINT			API of the connection
		wOverRun	UINT			Number of times the consumption was too long
		wUnderRun	UINT			Number of times the consumption was too fast
		dwCurrentTime	UDINT			Internal use
09	09	ASM Status	STRUCT of			See below.
		byGeneralStatus	BYTE			
		byReserved	BYTE			
		Extended Status	WORD			
X = supported — = not supported						

Adapter Diagnostic status values include the following:

Status	Description	CIP Status	Extended	Context
0	OK	0	0	The IO data are correctly exchanged
54	Connection in progress	0	0	The connection is in progress, but the IO data are not consumed yet.

Status	Description	CIP Status	Extended	Context
33	No connection	0	0	No connection
		0xFB	0xFB01	Connection in timeout
		0xFB	0xFB07	Optimization error / indeterminable MAC address
		0xFB	0xFB0B	Timeout on consumption
		0xFB	0xFB0C	Connection closed by a forward close
		0xFB	0xFB0E	Module in STOP
		0xFD	Status	Error from Encapsulation layer
		0xFE	TCP Error	Error on TCP connection
		0x02	0	No more resource to handle the connections
		0x20	0	Connections refused because of incorrect format or parameters
53	IDLE	0	0	A notification of IDLE is received

Services

The Adapter Diagnostic object performs the following services upon the listed object types:

Service ID		Description	Class	Instance	Notes
hex	dec				
01	01	Get_Attributes_All	X	X	Returns: • all class attributes (instance = 0) • instance attributes 1 to 7 (instance = 1)
61	97	Get_Output	—	X	Returns the status and value the output: • Offset 0 / UINT / Status • Offset 2 / USINT [0...409] / Output data
62	98	Get_Input	—	X	Returns the status and value the input: • Offset 0 / UINT / Status • Offset 2 / USINT [0...409] / Intput data

Service ID		Description	Class	Instance	Notes
hex	dec				
63	99	Set_DiagCounters	—	X	Sets the values of: • ST_Diag_CNT to 0. ...and... • ST_CHECK_TIME – both production and consumption –to 0 (but not the fields dwLastTime and dwCurrentTime)
X = supported — = not supported					

NOTE: If a service is addressed on an instance that does not exist, the service detects the following error: 0x05 – Path destination unknown.

EtherNet/IP Interface Diagnostics Object

Overview

The EtherNet/IP Interface Diagnostics object presents the instances, attributes and services described below.

Class ID

350 (hex), 848 (decimal)

Instance IDs

The EtherNet/IP Interface object presents two instance values:

- 0: class
- 1: instance

Attributes

EtherNet/IP Interface Diagnostics object attributes are associated with each instance, as follows:

Instance ID = 0 (class attributes):

Attribute ID	Description	GET	SET
01	Revision	X	—
02	Max Instance	X	—
X = supported — = not supported			

Instance ID = 1 (instance attributes):

Attribute ID	Description	Type	GET	SET	Value
01	Protocols Supported	UINT	X	—	
02	Connection Diagnostics	STRUCT	X	—	
	Max CIP IO Connections opened	UINT			Number of Class 1 connections opened since the last reset

Attribute ID	Description	Type	GET	SET	Value
	Current CIP IO Connections	UINT			Number of Class 1 connections currently opened
	Max CIP Explicit Connections opened	UINT			Number of Class 3 connections opened since the last reset
	Current CIP Explicit Connections	UINT			Number of Class 3 connections currently opened
	CIP Connections Opening Errors	UINT			Increments each time a Forward Open is not successful (Originator and Target)
	CIP Connections Timeout Errors	UINT			Increments when a connection times out (Originator and Target)
	Max EIP TCP Connections opened	UINT			Number of TCP connections (used for EIP, as client or server) opened since the last reset
	Current EIP TCP Connections	UINT			Number of TCP connections (used for EIP, as client or server) currently open
03	IO Messaging Diagnostics	STRUCT	X	X	
	IO Production Counter	UDINT			Increments each time a Class 0/1 message is sent
	IO Consumption Counter	UDINT			Increments each time a Class 0/1 message is received
	IO Production Send Errors Counter	UINT			Increments each time a Class 0/1 message is not sent
	IO Consumption Receive Errors Counter	UINT			Increments each time a consumption is received with a detected error
04	Explicit Messaging Diagnostics	STRUCT	X	X	
	Class 3 Msg Send Counter	UDINT			Increments each time a Class 3 message is sent (client and server)
	Class 3 Msg Receive Counter	UDINT			Increments each time a Class 3 message is received (client and server)
	UCMM Msg Receive Counter	UDINT			Increments each time a UCMM message is sent (client and server)
	UCMM Msg Receive Counter	UDINT			Increments each time a UCMM message is received (client and server)
X = supported — = not supported					

Services

The EtherNet/IP Interface Diagnostics object performs the following services upon the listed object types:

Service ID		Description	Class	Instance	Notes
hex	dec				
01	01	Get_Attributes_All	X	X	Returns the value of all attributes.
0E	14	Get_Attribute_Single	—	X	Returns the value of the specified attribute.
4C	76	Get_and_Clear	—	X	Returns and clears the values of all instance attributes.
X = supported — = not supported					

EtherNet/IP IO Scanner Diagnostics Object

Overview

The EtherNet/IP IO Scanner Diagnostics object presents the instances, attributes and services described below.

Class ID

351 (hex), 849 (decimal)

Instance IDs

The EtherNet/IP IO Scanner Diagnostics object presents two instances:

- 0: class
- 1: instance

Attributes

EtherNet/IP IO Scanner Diagnostics object attributes are associated with each instance, as follows:

Instance ID = 0 (class attributes):

Attribute ID	Description	GET	SET
01	Revision	X	—
02	Max Instance	X	—
X = supported — = not supported			

Instance ID = 1 (instance attributes):

Attribute ID	Description	Type	GET	SET
01	IO Status Table	STRUCT	X	—
	Size	UINT		
	Status	ARRAY of UNINT		
X = supported — = not supported				

Services

The EtherNet/IP IO Scanner Diagnostics object performs the following services upon the listed object types:

Service ID		Description	Class	Instance	Notes
hex	dec				
01	01	Get_Attributes_All	X	X	Returns the value of all attributes.
0E	14	Get_Attribute_Single	X	X	Returns the value of the specified attribute.
X = supported — = not supported					

IO Connection Diagnostics Object

Overview

The IO Connection Diagnostics object presents the instances, attributes and services described below.

Class ID

352 (hex), 850 (decimal)

Instance IDs

The IO Connection Diagnostics object presents two instance values:

- 0 (class)
- 257 ... 643 (instance): The instance number matches the connection number in the **Connection Settings** configuration (see Modicon M580, BMENOC0301/0311 Ethernet Communications Module, Installation and Configuration Guide).

NOTE: The Instance ID number = the Connection ID. For *M580* specifically, you can look up the Connection ID on the DTM Device List screen.

Attributes

IO Connection Diagnostics object attributes are associated with each instance, as follows:

Instance ID = 0 (class attributes):

Attribute ID	Description	GET	SET
01	Revision	X	—
02	Max Instance	X	—
X = supported — = not supported			

Instance ID = 1 to 256 (instance attributes):

Attribute ID	Description	Type	GET	SET	Value
01	IO Communication Diagnostics	STRUCT	X	X	

Attribute ID	Description	Type	GET	SET	Value
	IO Production Counter	UDINT			Increments at each production
	IO Consumption Counter	UDINT			Increments at each consumption
	IO Production Send Errors Counter	UINT			Increments each time a production is not sent
	IO Consumption Receive Errors Counter	UINT			Increments each time a consumption is received with a detected error
	CIP Connection Timeout Errors	UINT			Increments when a connection times out
	CIP Connection Opening Errors	UINT			Increments each time a connection is unable to open
	CIP Connection State	UINT			State of the Connection Bit
	CIP Last Error General Status	UINT			General status of the last error detected on the connection
	CIP Last Error Extended Status	UINT			Extended status of the last error detected on the connection
	Input Communication Status	UINT			Communication status of the inputs (see table, below)
	Output Communication Status	UINT			Communication status of the outputs (see table, below)
02	Connection Diagnostics	STRUCT	X	X	
	Production Connection ID	UDINT			Connection ID for production
	Consumption Connection ID	UDINT			Connection ID for consumption
	Production RPI	UDINT			RPI for production
	Production API	UDINT			API for production
	Consumption RPI	UDINT			RPI for consumption
	Consumption API	UDINT			API for consumption
	Production Connection Parameters	UDINT			Connection parameters for production
	Consumption Connection Parameters	UDINT			Connection parameters for consumption
	Local IP	UDINT			—
	Local UDP Port	UINT			—
	Remote IP	UDINT			—
	Remote UDP Port	UINT			—
	Production Multicast IP	UDINT			Multicast IP used for production (or 0)

Attribute ID	Description	Type	GET	SET	Value
	Consumption Multicast IP	UDINT			Multicast IP used for consumption (or 0)
	Protocols Supported	UDINT			Protocol supported on the connection: 1 = EtherNet/IP
X = supported — = not supported					

The following values describe the structure of the instance attributes: *CIP Connection State*, *Input Communication Status*, and *Output Communication Status*:

Bit Number	Description	Values
15...3	*Reserved*	0
2	Idle	0 = no idle notification 1 = idle notification
1	Consumption inhibited	0 = consumption started 1 = no consumption
0	Production inhibited	0 = production started 1 = no production

Services

The EtherNet/IP Interface Diagnostics object performs the following services upon the listed object types:

Service ID		Description	Class	Instance	Notes
hex	dec				
01	01	Get_Attributes_All	X	X	Returns the value of all attributes.
0E	14	Get_Attribute_Single	—	X	Returns the value of the specified attribute.
4C	76	Get_and_Clear	—	X	Returns and clears the values of all instance attributes.
X = supported — = not supported					

EtherNet/IP Explicit Connection Diagnostics Object

Overview

The EtherNet/IP Explicit Connection Diagnostics object presents the instances, attributes and services described below.

Class ID

353 (hex), 851 (decimal)

Instance IDs

The EtherNet/IP Explicit Connection Diagnostics object presents two instance values:

- 0: class
- 1...*N*: instance (*N* = maximum concurrent number of explicit connections)

Attributes

EtherNet/IP Explicit Connection Diagnostics object attributes are associated with each instance, as follows:

Instance ID = 0 (class attributes):

Attribute ID hex	Description	Value	GET	SET
01	Revision	1	X	—
02	Max Instance	0...N	X	—
X = supported — = not supported				

Instance ID = 1 to *N* (instance attributes):

Attribute ID hex	Description	Type	GET	SET	Value
01	Originator connection ID	UDINT	X	—	Originator to target connection ID
02	Originator IP	UINT	X	—	

Attribute ID hex	Description	Type	GET	SET	Value
03	Originator TCP Port	UDINT	X	—	
04	Target connection ID	UDINT	X	—	Target to originator connection ID
05	Target IP	UDINT	X	—	
06	Target TCP Port	UDINT	X	—	
07	Msg Send Counter	UDINT	X	—	Incremented each time a Class 3 CIP message is sent on the connection
08	Msg Receive counter	UDINT	X	—	Increments each time a Class 3 CIP message is received on the connection
X = supported — = not supported					

Services

The EtherNet/IP Explicit Connection Diagnostics object performs the following services upon the listed object type:

Service ID		Description	Class	Instance	Notes
hex	dec				
01	01	Get_Attributes_All	X	X	Returns the value of all attributes.
X = supported — = not supported					

EtherNet/IP Explicit Connection Diagnostics List Object

Overview

The EtherNet/IP Explicit Connection Diagnostics List object presents the instances, attributes and services described below.

Class ID

354 (hex), 852 (decimal)

Instance IDs

The EtherNet/IP Explicit Connection Diagnostics List object presents two instance values:

- 0: class
- 1: instance

Attributes

EtherNet/IP Explicit Connection Diagnostics List object attributes are associated with each instance, as follows:

Instance ID = 0 (class attributes):

Attribute ID	Description	GET	SET
01	Revision	X	—
02	Max Instance	X	—
X = supported — = not supported			

Instance ID = 1 to 2 (instance attributes):

Attribute ID	Description	Type	GET	SET	Value
01	Number of connections	UINT	X	—	Total number of opened explicit connections
02	Explicit Messaging Connections Diagnostic List	ARRAY of STRUCT	X	—	

Attribute ID	Description	Type	GET	SET	Value
	Originator connection ID	UDINT			O->T connection ID
	Originator IP	UINT			—
	Originator TCP port	UDINT			—
	Target connection ID	UDINT			T->O connection ID
	Target IP	UDINT			—
	Target TCP port	UDINT			—
	Msg Send counter	UDINT			Increments each time a Class 3 CIP message is sent on the connection
	Msg Receive counter	UDINT			Increments each time a Class 3 CIP message is received on the connection
X = supported — = not supported					

Services

The EtherNet/IP Explicit Connection Diagnostics object performs the following services upon the listed object types:

Service ID		Description	Class	Instance	Notes
hex	dec				
01	01	Get_Attributes_All	X	—	Returns the value of all attributes.
08	08	Create	X	—	—
09	09	Delete	—	X	—
4B	75	Explicit_Connections_Diagnostic_Read	—	X	—
X = supported — = not supported					

RSTP Diagnostics Object

Overview

The RSTP Diagnostics object presents the instances, attributes and services described below.

Class ID

355 (hex), 853 (decimal)

Instance IDs

The RSTP Diagnostics object presents these instance values:

- 0: class
- 1: instance

Attributes

RSTP Diagnostics object attributes are associated with each instance.

Instance ID = 0 (class attributes):

Attribute ID	Description	Type	GET	SET
01	Revision: This attribute specifies the current revision of the RSTP Diagnostic Object. The revision is increased by 1 at each new update of the object.	UINT	X	—
02	Max Instance: This attribute specifies the maximum number of instances that may be created for this object on a per device basis (for example, an RSTP Bridge). There is 1 instance for each RSTP port on a device.	UINT	X	—
X = supported — = not supported				

Instance ID = 1 to *N* (instance attributes):

Attribute ID	Description	Type	GET	CLEAR	Value
01	**Switch Status**	STRUCT	X	—	—

Attribute ID	Description	Type	GET	CLEAR	Value
	Protocol Specification	UINT	X	—	Refer to RFC-4188 for attribute definitions and value range. In addition, the following value is defined: [4]: the protocol is IEEE 802.1D-2004 and IEEE 802.1W
	Bridge Priority	UDINT	X	—	Refer to RFC-4188 for attribute definitions and value range.
	Time Since Topology Change	UDINT	X	—	
	Topology Change Count	UDINT	X	—	Refer to RFC-4188 for attribute definitions and value range.
	Designated Root	String	X	—	Refer to RFC-4188 for attribute definitions and value range.
	Root Cost	UDINT	X	—	
	Root Port	UDINT	X	—	
	Max Age	UINT	X	—	
	Hello Time	UINT	X	—	
	Hold Time	UDINT	X	—	
	Forward Delay	UINT	X	—	
	Bridge Max Age	UINT	X	—	
	Bridge Hello Time	UINT	X	—	
	Bridge Forward Delay	UINT	X	—	
02	**Port Status**	STRUCT	X	X	—
	Port	UDINT	X	X	Refer to RFC-4188 for attribute definitions and value range.
	Priority	UDINT	X	X	
	State	UINT	X	X	
	Enable	UINT	X	X	
	Path Cost	UDINT	X	X	
	Designated Root	String	X	X	
	Designated Cost	UDINT	X	X	
	Designated Bridge	String	X	X	
	Designated Port	String	X	X	
	Forward Transitions Count	UDINT	X	X	Refer to RFC-4188 for attribute definitions and value range. Services:

Attribute ID	Description	Type	GET	CLEAR	Value
					• Get_and_Clear: The current value of this parameter is returned with the response message. • other services: The current value of this parameter is returned without being cleared.
03	**Port Mode**	STRUCT	X	—	—
	Port Number	UINT	X	—	This attribute indicates the port number for a data query. The value range is configuration dependent. For a 4-port Ethernet device, as an instance, the valid range is 1...4.
	Admin Edge Port	UINT	X	—	This attribute indicates if this is a user-configured edge port: • 0: Force False • 1: Force True • 2: Auto Other values are not valid.
	Oper Edge Port	UINT	X	—	This attribute indicates if this port is currently an edge port: • 1: true • 2: false Other values are not valid.
	Auto Edge Port	UINT	X	—	This attribute indicates if this port is a dynamically determined edge port: • 1: true • 2: false Other values are not valid.
X = supported — = not supported					

Services

The RSTP Diagnostics object performs these services:

<table>
<tr><th colspan="2">Service ID</th><th rowspan="2">Description</th><th rowspan="2">Class</th><th rowspan="2">Instance</th><th rowspan="2">Notes</th></tr>
<tr><th>hex</th><th>dec</th></tr>
<tr><td>01</td><td>01</td><td>Get_Attributes_All</td><td>X</td><td>X</td><td>This service returns:
all attributes of the class
all attributes of the instance of the object</td></tr>
<tr><td>0E</td><td>14</td><td>Get_Attribute_Single</td><td>X</td><td>X</td><td>This service returns:
the contents of a single attribute of the class
the contents of the instance of the object as specified
Specify the attribute ID in the request for this service.</td></tr>
<tr><td>4C</td><td>76</td><td>Get_and_Clear</td><td>—</td><td>X</td><td>This service returns the contents of a single attribute of the instance of the object as specified. Then the relevant counter-like parameter(s) within the specified attribute are cleared. (Specify the attribute ID in the request for this service.)</td></tr>
<tr><td colspan="6">X = supported
— = not supported</td></tr>
</table>

Service Port Control Object

Overview

The Service Port Control object is defined for port control purposes.

Class ID

400 (hex), 1024 (decimal)

Instance IDs

The Service Port Control object presents these instance Values:

- 0: class
- 1: instance

Attributes

Service Port Control object attributes are associated with each instance.

Required class attributes (instance 0):

Attribute ID	Description	Type	Get	Set
01	Revision	UINT	X	—
02	Max Instance	UINT	X	—
X = supported — = not supported				

Required instance attributes (instance 1):

Attribute ID		Description	Type	Get	Set	Value
hex	dec					
01	01	Port Control	UINT	X	X	0 (default): disabled 1: access port 2: port mirroring
02	02	Mirror	UINT	X	X	bit 0 (default): ETH 2 port bit 1: ETH 3 port bit 2: backplane port bit 3: internal port
X = supported — = not supported						

NOTE:

- If the SERVICE port is not configured for port mirroring, the mirror attribute is ignored. If the value of a parameter request is outside the valid range, the service request is ignored.
- In port mirroring mode, the SERVICE port acts like a read-only port. That is, you cannot access devices (ping, connection to Control Expert, etc.) through the SERVICE port.

Services

The Service Port Control object performs these services for these object types:

Service ID		Name	Class	Instance	Description
hex	dec				
01	01	Get_Attributes_All	X	X	Get all attributes in a single message.
02	02	Set_Attributes_All	—	X	Set all attributes in a single message.
0E	14	Get_Attribute_Single	X	X	Get a single specified attribute.
10	16	Set_Attribute_Single	—	X	Set a single specified attribute.
X = supported — = not supported					

SNTP Diagnostics Object

Overview

The SNTP Diagnostics object presents the instances, attributes and services described below.

Class ID

405 (hex), 1029 (decimal)

Instance IDs

The SNTP Diagnostics object presents two instances:

- 0: class
- 1: instance

Attributes

SNTP Diagnostics object attributes are associated with each instance, as follows:

Instance ID = 0 (class attributes):

Attribute ID	Description	GET	SET
01	Revision	X	—
02	Maximum Instance	X	—
X = supported — = not supported			

Instance ID = 1 (instance attributes):

Attribute ID		Description	Type	GET	SET	Value
hex	dec					
01	01	Network Time Service Configuration	STRUCT of	X	—	
		Primary NTP Server IP Address	UDINT			
		Secondary NTP Server IP Address	UDINT			
		Polling Period	USINT			In seconds
		Update controller with Module Time	USINT			• 0 = do not update • 1 = update
		Time Zone	UDINT			Depends on the operating system of the configuration software.
		Time Zone Offset	INT			In minutes
		Daylight saving time bias	USINT			
		Daylight Saving Start Date - Month	USINT			
		Daylight Saving Start Date - week #, day of week	USINT			• MSB (4 bits) : week # • LSB (4 bits) : 0=Sunday…6= Saturday
		Daylight Saving Start Time	UDINT			Seconds elapsed from midnight
		Daylight Saving End Date - Month	USINT			
		Daylight Saving End Date - week #, day of week	USINT			• MSB (4 bits) : week # • LSB (4 bits) : 0=Sunday…6= Saturday
		Daylight Saving End Time	UDINT			Seconds elapsed from midnight
		Reserved	USINT[15]			
02	02	Network Time Service Status	UDINT	X	—	• 1 = idle • 2 = operational
03	03	Link to NTP Server Status	UDINT	X	—	• 1 = NTP server not reachable • 2 = NTP server is reachable
04	04	Current NTP Server IP Address	UDINT	X	—	

Attribute ID		Description	Type	GET	SET	Value
hex	dec					
05	05	NTP Server Type	UDINT	X	—	Re: the server identified in attribute 03: • 0 = primary • 1= secondary
06	06	NTP Server Time Quality	UDINT	X	—	Jitter of the clock/time in microseconds/ second
07	07	Number of NTP Requests Sent	UDINT	X	—	
08	08	Number of Communication Errors	UDINT	X	—	
09	09	Number of NTP Responses Received	UDINT			
A	10	Last Error	UINT			• 0 = no error • 1 = NTP_ERROR_CONF_BAD_ PARAM • 2 = NTP_ERROR_CONF_BAD_ CONF • 3 = NTP_ERROR_CREATE_ SERVICE • 4 = NTP_ERROR_WRONG_STATE • 5 = NTP_ERROR_NO_RESPONSE

B	11	Current Date and Time	DATE_ AND_ TIME			`{`
						`time_of_day UDINT,`
						`date UINT`
						`}`
						Refer to CIP specification.

C	12	Daylight Savings Status	UDINT			• 1 = Daylight savings is enabled and the date/time is within the applicable period • 2 = Daylight savings is not enabled or enabled but not within the applicable period
D	13	Time Since Last Update	DINT			Amount of time elapsed since a valid response from the NTP server in 100ms increments. -1 = not updated
X = supported — = not supported						

Services

The SNTP Diagnostics object performs the following services upon the listed object types:

Service ID		Description	Class	Instance	Notes
hex	dec				
01	01	Get_Attributes_All	X	X	Returns: • all class attributes (instance = 0) • instance attributes 1 to 7 (instance = 1)
0E	14	Get_Attribute_Single	X	X	Returns the value of the specified attribute.
32	50	Clear_All	—	X	Clears data in attributes 6, 7, 8, 9, 10, 13 (all attributes defined in decimal notation).
X = supported — = not supported					

Hot Standby FDR Sync Object

Overview

The Hot Standby FDR Sync object presents the instances, attributes and services described below.

Class ID

406 (hex), 1030 (decimal)

Instance IDs

The Hot Standby FDR Sync object presents two instances:

- 0: class
- 1: instance

Attributes

Hot Standby FDR Sync object attributes are associated with each instance, as follows:

Instance ID = 0 (class attributes):

Attribute ID	Description	GET	SET
01	Revision	X	—
02	Maximum Instance	X	—
X = supported — = not supported			

Instance ID = 1 (instance attributes):

Attribute ID		Description	Type	GET	SET	Value
hex	dec					
01	01	Status	UDINT	X	—	• bit 0: 0 = service not running; 1 = service is running • bit 1:0 = service has no detected error; 1 = service has detected an error
02	02	Checksum of the parameter (.prm) files	UDINT	X	—	
X = supported — = not supported						

Services

The Hot Standby FDR Sync object performs the following services upon the listed object types:

Service ID		Description	Class	Instance	Notes
hex	dec				
01	01	Get_Attributes_All	X	X	Returns: • all class attributes (instance = 0) • instance attributes 1 to 7 (instance = 1)
07	07	Stop	—	X	In Standby state, start the synchronization service. In Primary state, no action.
0E	14	Get_Attribute_Single	X	X	Returns the value of the specified attribute.
4B	75	Copy_Primary_to_Standby	X	X	Applicable only if the device is in Standby state. Otherwise, an error is detected.
4C	76	Copy_Standby_to_Primary	X	X	Applicable only if the device is in Standby state. Otherwise, an error is detected.
4D	77	Clear_Files_in_Primary	X	X	Applicable only if the device is in Primary state. Otherwise, an error is detected.
X = supported — = not supported					

Ethernet Backplane Diagnostics Object

Overview

The Ethernet Backplane Diagnostics object presents the instances, attributes and services described below.

Class ID

407 (hex), 1031 (decimal)

Instance IDs

The Ethernet Backplane Diagnostics object presents two instances:

- 0: class
- 1: instance

Attributes

Ethernet Backplane Diagnostics object attributes are associated with each instance, as follows:

Instance ID = 0 (class attributes):

Attribute ID	Description	GET	SET
01	Revision	X	—
02	Maximum Instance	X	—
03	Number of Instances	X	—
X = supported — = not supported			

Instance ID = 1 (instance attributes):

Attribute ID		Description	Type	GET	SET	Value
hex	dec					
01	01	Backplane Ethernet Port Status	UINT	X	—	Link status/health of each module on the backplane: • bit 0-14: 0 = link is up, 1 = link is down • bit 15: 0 = backplane is in normal operating state • bit 15: 1= backplane is not in normal operating state
02	02	Extended Health of Ethernet Backplane	UINT	X	—	For all bits, below, 0 = no error detected, 1 = error detected: • Bit 0: SMI error detected • Bit 1: HUBIX error detected • Bit 2: Undervoltage detected • Bit 3: Overvoltage detected • Bit 4: Backplane head did not respond • Bit 14: Backplane firmware is not compatible • Bit 15: Backplane did not respond • Other bits: reserved
X = supported — = not supported						

Services

The Ethernet Backplane Diagnostics object performs the following services upon the listed object types:

Service ID		Description	Class	Instance	Notes
hex	dec				
01	01	Get_Attributes_All	X	X	Returns: • all class attributes (instance = 0) • instance attributes 1 to 7 (instance = 1)
0E	14	Get_Attribute_Single	X	X	Returns the value of the specified attribute.
X = supported — = not supported					

DTM Device Lists

Introduction

This section describes the connection of an M580 CPU to other network nodes through the Control Expert **DTM Browser**.

Device List Configuration and Connection Summary

Introduction

The Device List contains read-only properties that summarize these items:

- configuration data:
  - input data image
  - output data image
  - maximum and actual numbers of devices, connections, and packets
- Modbus request and EtherNet/IP connection summary

Open the Page

View the read-only properties of the M580 CPU in the Control Expert **Device List**:

Step	Action
1	Open your Control Expert project.
2	Open the DTM Browser (**Tools > DTM Browser)**.
3	Double-click the CPU DTM in the **DTM Browser** to open the configuration window. **NOTE:** You can also right-click the CPU DTM and select **Open**.
4	Select **Device List** in the navigation tree.

Configuration Summary Data

Select **Device List** and view the **Configuration Summary** table on the **Summary** tab to see values for these items:

- **Input**

- **Output**
- **Configuration Size**

Expand (+) the **Input** row to view the **Input Current Size** values:

Description	Source
This value is the sum of Modbus requests and EtherNet/IP connection sizes.	This value is configured in the **General** page for a selected distributed device and connection.

Expand (+) the **Output** row to view the **Output Current Size** values:

Description	Source
This value is the sum of Modbus requests and EtherNet/IP connection sizes.	This value is configured in the **General** page for a selected distributed device and connection.

The maximum size of the X Bus input or output memory variable is 4 KB (2048 words). The variable contains a 16-byte descriptor followed by a value that represents the number of input or output data objects. Each data object contains a 3-byte object header followed by the input or output data. The number of data objects and the size of the input or output data depend on the configuration. The maximum overhead in the variable is 403 bytes (16 + 387), where 16 is the number of bytes in the descriptor and 387 is the product of 3 x 129, where 3 is the number of bytes in the header and 129 is the number of input or output objects (128 maximum scanned devices or local slaves that the BMENOC03•1 module supports plus one input or output object for the scanner DDDT). Therefore, at least 3.6 KB of the 4-KB variable is available for the input or output current size.

NOTE: The input current size also includes 28 words of scanner DDT input data. The output current size also includes 24 words of scanner DDT output data.

Expand (+) the **Configuration Size** row in the **Connection Summary** table to view these values:

Name	Description	Source
Maximum Number of DIO Devices	the maximum number of distributed devices that can be added to the configuration	predefined
Current Number of DIO Devices	the number of distributed devices in the current configuration	network design in the Control Expert device editor
Maximum Number of DIO Connections	the maximum number of connections to distributed devices that can be managed by the CPU	predefined
Current Number of DIO Connections	the number of connections to distributed devices in the current configuration	network design in the Control Expert device editor
Maximum Number of CSIO Devices	the maximum number of CIP Safety devices that can be added to the configuration	capability of the module
Current Number of CSIO Devices	the number of active and inactive CIP Safety devices in the current configuration	number of CIP Safety devices in the **Device List > Safe Bus**

Name	Description	Source
Maximum Number of CSIO Connections	the maximum number of CIP Safety connections to distributed devices that can be managed by the Ethernet communications module	capability of the module
Current Number of CSIO Connections	the number of connections by active devices in the current configuration	device configuration in the Control Expert **Device Editor**
Maximum Number of Packets	the maximum number of packets per second the module is able to manage	predefined
Current Number of Input Packets	total number of input packets (traffic) per second, based on the current number of modules and its configured input data	network design in the Control Expert device editor
Current Number of Output Packets	total number of output packets (traffic) per second, based on the current number of modules and its configured output data	network design in the Control Expert device editor
Current Number of Total Packets	total number of packets (traffic in both directions) per second, based on the current number of modules and its configured I/O data	network design in the Control Expert device editor

Request / Connection Summary Data

Select **Device List** and view the **Request / Connection Summary** table on the **Summary** tab. The Control Expert DTM uses this information to calculate the total bandwidth that distributed equipment consumes:

Column	Description
Connection Bit	• Connection health bits display the status of each device with one or more connections. • Connection control bits can be toggled on and off using object IDs.
Task	The task that is associated with this connection.
Input Object	The ID of the input object associated with the connection (see the note following the table).
Output Object	The ID of the output object associated with the connection (see the note following the table).
Device	The device **Number** is used for the health and control bit index.
Device Name	A unique name associated with the device that owns the connection.
Type	The target device type: • EtherNet/IP • Local Slave • Modbus TCP

Column	Description
Address	The target device IP address for remote devices (does not apply to local slaves).
Rate (msec)	The RPI (for EtherNet/IP) or the repetitive rate (for Modbus TCP), in ms.
Input Packets per Second	The number of input (T->O) packets per second exchanged over this connection.
Output Packets per Second	The number of output (O->T) packets per second exchanged over this connection.
Packets per Second	The total number of packets per second exchanged over this connection in both Input and output directions.
Bandwidth Usage	The total bandwidth used by this connection (total bytes per second traffic).
Size In	The number of input words configured for this remote device.
Size Out	The number of output words configured for this remote device.

NOTE: The numeric identifiers in the **Input Object** and **Output Object** columns represent the objects associated with a single device connection (scan line). For example, if an EtherNet/IP connection has an input object of 260 and an output object of 261, the corresponding control bits for this connection are in the DIO_CTRL field in the M580 CPU device DDT. Object 260 is the fifth bit and object 261 is the sixth bit in this field. There can be multiple connections for a device. Set the corresponding bits to control the input and output objects for these connections.

Device List Parameters

Introduction

Configure parameters for devices in the **Device List** on these tabs:

- **Properties**
- **Address Setting**
- **Request Setting** (Modbus devices only)

View the Configuration Tabs

Navigate to the **Device List** configuration tabs

Step	Action
1	In the **DTM Browser** (**Tools > DTM Browser**), double-click the DTM that corresponds to the CPU.
2	In the navigation pane, expand (+) the **Device List**, page 285 to see the associated Modbus TCP and EtherNet/IP devices.
3	Select a device from the **Device List** to view the **Properties**, **Address Setting**, and **Request Setting** tabs tabs. **NOTE:** These tabs are described in detail below.

Properties Tab

Configure the **Properties** tab to perform these tasks:

- Add the device to the configuration.
- Remove the device from the configuration.
- Edit the base name for variables and data structures used by the device.
- Indicate how input and output items are created and edited.

Configure the **Properties** tab:

Field	Parameter	Description
Properties	**Number**	The relative position of the device in the list.
	Active Configuration	**Enabled:** Add this device to the Control Expert project configuration.
		Disabled: Remove this device from the Control Expert project configuration.

Field	Parameter	Description
IO Structure Name	**Structure Name**	Control Expert automatically assigns a structure name based on the variable name.
	Variable Name	**Variable Name**: An auto-generated variable name is based on the alias name.
	Default Name	Press this button to restore the default variable and structure names.
Items Management	**Import Mode**	**Manual**: I/O items are manually added in the **Device Editor**. The I/O items list is not affected by changes to the device DTM.
		Automatic: I/O items are taken from the device DTM and updated if the items list in the device DTM changes. Items cannot be edited in the **Device Editor**.
	Reimport Items	Press this buttom to import the I/O items list from the device DTM, overwriting any manual I/O item edits. Enabled only when **Import mode** is set to **Manual**.

Click **Apply** to save your edits and leave the window open for further edits.

Address Setting Tab

Configure the **Address Setting** page to perform these tasks:

- Configure the IP address for a device.
- Enable or disable DHCP client software for a device.

  NOTE: When the DHCP client software is enabled in a Modbus device, it obtains its IP address from the DHCP server in the CPU.

In the **Address Setting** page, edit these parameters to conform to your application's design and functionality:

Field	Parameter	Description
IP Configuration	**IP Address**	By default: • The first three octet values equal the first three octet values of the CPU. • The fourth octet value equals this device Number setting. In this case, the default value is 004. In our continuing example, type in the address **192.168.1.17**.
	Subnet Mask	The device subnet mask. **NOTE:** For this example, accept the default value (255.255.255.0).
	Gateway	The gateway address used to reach this device. The default of 0.0.0.0 indicates this device is located on the same subnet as the CPU. **NOTE:** For this example, accept the default value.

Field	Parameter	Description
Address Server	**DHCP for this Device**	**Enabled**: Activate the DHCP client in this device. The device obtains its IP address from the DHCP service provided by the CPU appears on the auto-generated DHCP client list (see Modicon M580, BMENOC0321 Control Network Module, Installation and Configuration Guide).
		Disabled (default): Deactivates the DHCP client in this device.
		NOTE: For this example, select **Enabled**.
	Identified by	If **DHCP for this Device** is **Enabled**, it indicates the device identifier type: • **MAC Address** • **Device Name** **NOTE:** For this example, select **Device Name**.
	Identifier	If DHCP for this Device is Enabled, the specific device MAC Address or Name value. **NOTE:** For this example, accept the default setting of **NIP2212_01** (based on the **Alias name**).

Click **Apply** to save your edits, and leave the window open for further edits.

Request Setting Tab

Configure the **Request Setting** tab to add, configure, and remove Modbus requests for the Modbus device. Each request represents a separate link between the CPU and the Modbus device.

NOTE: The **Request Setting** tab is available only when a Modbus TCP device is selected in the **Device List**.

Create a request:

Step	Action
1	Press the **Add Request** button to see a new request in the table. Press the **Add Request** button: • The new request appears in the table. • The corresponding request items appear in the **Device List**. **NOTE:** The **Add Request** function is enabled only when **Import Mode** on the **Properties** tab is set to **Manual**.
2	Configure the request settings according to the table below.
3	Repeat these steps to create additional requests.
4	Press the **Apply** to save the request.

This table describes the **Request Settings** parameters for Modbus devices:

Setting	Description
Connection Bit	This bit indicates the read-only offset for the health bit for this connection. Offset values (starting at 0) are auto-generated by the Control Expert DTM based on the connection type.
Unit ID	The Unit ID is the number used to identify the target of the connection. **NOTE:** Consult the manufacturer's user manual for the specific target device to find its Unit ID.
Health Time Out (ms)	This value represents the maximum allowed interval between device responses before a time out is detected: • valid range: 5 ... 65535 ms • interval: 5 ms • default: 1500 ms
Repetitive Rate (ms)	This value represents the data scan rate in intervals of 5 ms. (The valid range is 0...60000 ms. The default is 60 ms.)
RD Address	This is the address of the input data image in the Modbus device.
RD Length	This value represents the number of words (0...125) in the Modbus device that the CPU reads.
Last Value	This value represents the behavior of input data in the application if communications are lost: • **Hold Value** (default) • **Set To Zero**
WR Address	This is the address of the output data image in the Modbus device.
WR Length	This value represents the number of words (0...120) in the Modbus device to which the CPU writes.

Remove a request:

Step	Action
1	Click a row in the table.
2	Press the **Remove** button to remove the request. **NOTE:** The corresponding request items disappear from the **Device List**.
3	Press the **Apply** to save the configuration.

The next step is to connect the Control Expert project to the Modbus device.

Standalone DDT Data Structure for M580 PACs

Introduction

This topic describes the Control Expert **Device DDT** tab for an M580 PAC in a local backplane. A derived data type (DDT) is a set of elements with the same type (ARRAY) or with different types (structure).

NOTE: The device DDT type supported by a standalone M580 PAC depends on its firmware version, and can be T_BMEP58_ECPU, T_BMEP58_ECPU_EXT, T_BMEP58_ECPU_EXT2, or T_BMEP58_ECPUPRP_EXT.

Access the Device DDT Tab

Access the device DDT for the controller in Control Expert:

Step	Action
1	Open a Control Expert project that includes an M580 PAC in the configuration.
2	Rebuild the project (**Build > Rebuild All Project**.)
3	Open the **Data Editor** in the Control Expert **Project Browser** (**Tools > Data Editor**).
4	Select the **Device DDT** checkbox.
5	Expand (+) the Device DDT in the **Name** column.

You can add this variable to an Animation Table, page 323 to read the status and set the object control bit.

NOTE: The red arrow and lock icons in the **Device DDT** table indicate that the variable name was auto-generated by Control Expert based on the configuration of the communication module, local slave, or distributed device. The variable name cannot be edited.

Input and Output Freshness

This table describes the inputs and outputs that are associated with EtherNet/IP or Modbus devices:

Name	Description
Freshness	This is a global bit: • **1:** All input objects below (**Freshness_1**, **Freshness_2**, etc.) for the associated device are true (**1**) and provide up-to-date data. • **0:** One or more inputs (below) is not connected and does not provide up-to-date data.
Freshness_1	This bit represents individual input objects for the connection: • **1:** The input object is connected and provides up-to-date data. • **0:** The input object is not connected and does not provide up-to-date data.
Freshness_2 **Freshness_3** ...	This bit represents an individual input object for the device: • **1:** The input object is true (**1**) and provides up-to-date data. • **0:** The input object is not connected (**0**) and does not provide up-to-date data.
(available)	The rows after the **Freshness** data are organized in groups of **Inputs** and **Outputs** that have user-defined names. The number of input and output rows depends on the number of input and output requests configured for a particular device.

Parameters

Use the Control Expert **Device DDT** tab to configure parameters for the controller RIO head on the local backplane:

Parameter		Description
Implicit device DDT	Name	the default name of the device DDT
	Type	module type (uneditable)
Goto details		link to the DDT data editor screen

Standalone Configuration

These tables describe the fields in the implicit device DDT type that is used with the controller RIO communication server in standalone configurations using Unity Pro 10.0 or any subsequent supporting version(s), and M580 PAC version 2.01 or any subsequent supporting version(s).

NOTE:

Unity Pro is the former name of Control Expert for version 13.1 or earlier.

Input Parameters

The following tables describe the input parameters in the device DDT for the controller.

ETH_STATUS (WORD):

Name	Type	Bit	Description
PORT1_LINK	BOOL	0	0 = ETH 1 link is down.
			1 = ETH 1 link is up.
PORT2_LINK	BOOL	1	0 = ETH 2 link is down.
			1 = ETH 2 link is up.
PORT3_LINK	BOOL	2	0 = ETH 3 link is down.
			1 = ETH 3 link is up.
ETH_BKP_PORT_LINK	BOOL	3	0 = Ethernet backplane link is down.
			1 = Ethernet backplane link is up.
REDUNDANCY_STATUS (see the note below.)	BOOL	5	0 = Redundant path is not available.
			1 = Redundant path is available.
SCANNER_OK	BOOL	6	0 = Scanner is not present.
			1 = Scanner is present.
GLOBAL_STATUS	BOOL	7	0 = At least one service is not operating normally. **NOTE:** Refer to the footnotes for `SERVICE_STATUS` and `SERVICE_STATUS2`, below, to identify the services that set `GLOBAL STATUS` to 0.
			1 = All services are operating normally.
NETWORK_HEALTH	BOOL	8	0 = A potential network broadcast storm is detected. **NOTE:** Check your wiring and your controller and BMENOC0301/BMENOC0311 configurations.
			1 = A network broadcast storm is not detected.

NOTE:

- You can monitor interruptions in the RIO main ring by diagnosing the *REDUNDANCY_STATUS* bits in the controller DDT. The system detects and reports in this bit a main ring cable interruption that persists for at least 5 seconds.

  REDUNDANCY_STATUS bit value:

  0: The cable is broken, disconnected, or the device is stopped.

  1: The loop is present and healthy.
- For RIO main rings using BMECRA31310 redundant adapters, the *REDUNDANCY_STATUS* bit is not supported and will be set to **0**.

Duplicate IP addresses can cause errors in communication with the other modules.

⚠ WARNING

UNINTENDED EQUIPMENT OPERATION

Confirm that each module has a unique IP address.

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

SERVICE_STATUS (WORD):

Name	Type	Bit	Description
RSTP_SERVICE[1]	BOOL	0	0 = RSTP service is not operating normally.
			1 = RSTP service is operating normally or disabled.
PORT502_SERVICE[1]	BOOL	2	0 = Port 502 service is not operating normally.
			1 = Port 502 service is operating normally or disabled.
SNMP_SERVICE[1]	BOOL	3	0 = SNMP service is not operating normally.
			1 = SNMP service is operating normally or disabled.
MAIN_IP_ADDRESS_STATUS	BOOL	4	0 = The main IP address is a duplicate or unassigned.
			1 = The main IP address is unique and valid.
ETH_BKP_FAILURE	BOOL	5	0 = Ethernet backplane hardware is not operating properly.
			1 = Ethernet backplane hardware is operating properly.
ETH_BKP_ERROR	BOOL	6	0 = Ethernet backplane error detected.
			1 = Ethernet backplane is operating properly.
EIP_SCANNER[1]	BOOL	7	0 = Service not operating normally.
			1 = Service operating normally.
MODBUS_SCANNER[1]	BOOL	8	0 = Service not operating normally.
			1 = Service operating normally.
NTP_SERVER[1, 2]	BOOL	9	0 = SNTP server not operating normally.
			1 = SNTP server operating normally.
SNTP_CLIENT[1, 2]	BOOL	10	0 = Service not operating normally.

Name	Type	Bit	Description
			1 = Service operating normally.
WEB_SERVER[1]	BOOL	11	0 = Service not operating normally.
			1 = Service operating normally.
FIRMWARE_UPGRADE	BOOL	12	0 = Service not operating normally.
			1 = Service operating normally.
FTP	BOOL	13	0 = Service not operating normally.
			1 = Service operating normally.
FDR_SERVER[1]	BOOL	14	0 = Service not operating normally.
			1 = Service operating normally.
EIP_ADAPTER[1]	BOOL	15	0 = EIP adapter (server) service not operating normally.
			1 = EIP adapter (server) service operating normally.
1. When this service is set to 0, `GLOBAL_STATUS` is also set to 0. 2. Only for firmware earlier than version 4.01.			

SERVICE_STATUS2 (WORD):

Name	Type	Bit	Description
A_B_IP_ADDRESS_STATUS	BOOL	0	0 = Duplicate IP or no IP address assigned.
			1 = IP addresses (A/B status) correctly assigned.
LLDP_SERVICE[1]	BOOL	1	0 = LLDP service is not operating normally.
			1 = LLDP service is operating normally or disabled.
EVENT_LOG_STATUS	BOOL	2	0 = Event log service is not operating normally.
			1 = Event log service is operating normally or is disabled.
LOG_SERVER_NOT_ REACHABLE	BOOL	3	1 = No acknowledgment received from the syslog server.
			0 = Acknowledgment received from the syslog server
CSIO_SCANNER (CIP Safety PAC)	BOOL	4	0 = At least one CIP Safety connection is not operating normally.
			1 = All CIP Safety I/O devices are operating normally.
NTP_SYNC	BOOL	5	1= Server Only mode..

Name	Type	Bit	Description
			0 = Not Server Only mode.
NTP_SERVICE	BOOL	6	0 = NTP Daemon status = down.
			1 = NTP Daemon status = active.
NTP_QUALITY_WARNING	BOOL	7	1= Quality of the clock out of the range defined in the configuration.
			0 = Clock quality within defined configuration range.
(reserved)	–	8–15	(reserved)
1. When this service is set to 0, `GLOBAL_STATUS` is also set to 0.			

ETH_PORT_1_2_STATUS (BYTE):

Name	Type	Description
Ethernet ports function and RSTP role coded on 2 bits	Bits 1...0	0: ETH 1 disabled
		1: ETH 1 access port
		2: ETH 1 port mirroring
		3: ETH 1 device network port
	Bits 3...2	reserved (0)
	Bits 5...4	0: ETH 2 disabled
		1: ETH 2 access port
		2: ETH 2 port mirroring
		3: ETH 2 device network port
	Bits 7...6	0: ETH 2 alternate RSTP port
		1: ETH 2 backup RSTP port
		2: ETH 2 designated RSTP port
		3: ETH 2 root RSTP port

ETH_PORT_3_BKP_STATUS (BYTE):

Name	Bit	Description
Ethernet ports function and RSTP role coded on 2 bits	Bits 1...0	0: ETH 3 disabled
		1: ETH 3 access port
		2: ETH 3 port mirroring
		3: ETH 3 device network port

Name	Bit	Description
	Bits 3...2	0: ETH 3 alternate RSTP port
		1: ETH 3 backup RSTP port
		2: ETH 3 designated RSTP port
		3: ETH 3 root RSTP port
	Bits 5...4	0: The Ethernet backplane port is disabled.
		1: The Ethernet backplane port is enabled to support Ethernet communications.
	Bits 7...6	reserved (0)

FDR_USAGE:

Type	Type	Description
FDR_USAGE	BYTE	% of FDR server usage

NTP_WITHIN:

Type	Type	Description
NTP_WITHIN	UINT	Estimated accuracy of the clock in milliseconds.

NTP_NB_SERVER_CONNECTED:

Type	Type	Description
NTP_SERVER_CONNECTED	UINT	Number of servers connected.

IN_PACKETS (UINT):

Type	Bit	Description
UINT	0-7	number of packets received on the interface (internal ports)

IN_ERRORS (UINT):

Type	Bit	Description
UINT	0-7	number of inbound packets that contain detected errors

OUT_PACKETS (UINT):

Type	Bit	Description
UINT	0-7	number of packets sent on the interface (internal ports)

OUT_ERRORS (UINT):

Type	Bit	Description
UINT	0-7	number of outbound packets that contain detected errors

CONF_SIG (UDINT):

Type	Bit	Description
UDINT	0-15	Signatures of all files on local module FDR server

Output Parameters

Although the complete Hot Standby Device DDT is not exchanged from the primary controller to the standby controller, these fields are transferred: *DROP_CTRL*; *RIO_CTRL*; *DIO_CTRL*

These tables describe those output parameters:

DROP_CTRL:

Name	Type	Rank	Description
DROP_CTRL	BOOL	1...32 or 1...64	1 bit per RIO drop (up to 32 or 64 depending on the controller firmware version)

RIO_CTRL:

Name	Type	Rank	Description
RIO_CTRL	BOOL	257...384	1 bit per RIO (up to 128)

DIO_CTRL:

Name	Type	Rank	Description
DIO_CTRL	BOOL	513...640	1 bit per DIO (up to 128)

CSIO_HEALTH:

Name	Type	Rank	Description
CSIO_HEALTH (safety)	BOOL	769...896	CSIO health bits (1 bit per DIO up to 68 CSIOs)

SERVICE_CMD (WORD):

Name	Bit	Rank	Description
NTP_STOP	BOOL	0	0: to start the service
			1: to stop the service

RED_PRP_DROP_SWAP:

Name	Type	Rank	Description
RED_PRP_DROP_SWAP	BOOL	1...64	1 bit per PRP drop (up to 64). A swap is only possible for the PRP drop managed by BMECRA31310(H) adapter modules in redundant mode.

Device Health Status

Although the complete Hot Standby Device DDT is not exchanged from the primary controller to the standby controller, these fields are transferred: DROP_HEALTH; RIO_HEALTH; LS_HEALTH; DIO_HEALTH

This table describes the health of the devices that are scanned by the module. The data is presented as an array of boolean:

Parameter	Type	Health status of ...
DROP_HEALTH	ARRAY [1...32] of BOOL or ARRAY [1...64] of BOOL	One array element corresponds to one X80 drop managed by a BMXCRA••••• or BMECRA••••• adapter module (up to a maximum of 32 or 64 depending on the controller firmware version).
RIO_HEALTH	ARRAY [257...384] of BOOL	RIO devices: One array element corresponds to one RIO device (up to a maximum of 128 RIO devices).
LS_HEALTH	ARRAY [1...3] of BOOL	local slaves: One array element corresponds to one local slave (up to a maximum of three local slaves).
DIO_HEALTH	ARRAY [513...640] of BOOL	DIO devices: One array element corresponds to one DIO device (up to a maximum of 128 DIO devices).
CSIO_HEALTH (CIP Safety PAC)	ARRAY [769...896] of BOOL	CSIO devices: One array element corresponds to one CSIO device (up to a maximum of 128 CSIO devices).

Values:

- 1 (true): A device is healthy. The input data from the device is received within the pre-configured health timeout.
- 0 (false): A device is not healthy. The input data from the device is not received within the pre-configured health timeout.

Hot Standby DDT Data Structure

Introduction

The T_M_ECPU_HSBY DDT is the exclusive interface between the M580 Hot Standby system and the application running in a BMEH58•040 or BMEH58•040S controller. The DDT instance should appear as: ECPU_HSBY_1.

NOTE: For firmware version 2.80 and later, the T_M_ECPU_HSBY DDT is named T_M_ECPU_HSBY_EXT.

NOTICE

UNMONITORED LOSS OF REDUNDANCY IN HOTSTANDBY SYSTEM

Review and manage the T_M_ECPU_HSBY DDT for proper operation of the system.

Failure to follow these instructions can result in equipment damage.

The T_M_ECPU_HSBY DDT presents three distinct sections:

- LOCAL_HSBY_STS: Provides information about the local controller. Data is both auto-generated by the Hot Standby system, and provided by the application. This data is exchanged with the remote controller.
- REMOTE_HSBY_STS: Provides information about the remote controller, and contains the image of the last received exchange from the counterpart controller. The validity of this information is represented by the REMOTE_STS_VALID flag in the common part of this DDT . When set to 1, both controllers are communicating.

  NOTE: The structure of both the LOCAL_HSBY_STS and Remote_HSBY_STS sections are determined by the HSBY_STS_T data type, and are therefore identical. Each is used to describe data relating to one of the two Hot Standby controllers.
- A common part of the DDT: Consists of several objects, including status data, system control objects, and command objects:
  - Status data is provided by the Hot Standby system as a result of diagnostic checking.
  - System control objects enable you to define and control system behavior.
  - Command data objects include executable commands you can use to modify the system state.

Local Controller versus Remote Controller

The T_M_ECPU_HSBY DDT employs the terms *local* and *remote*:

- *Local* refers to the Hot Standby controller to which your PC is connected.

- *Remote* refers to the other Hot Standby controller.

Data Boundary Alignment

M580 BMEH58•040 and BMEH58•040S controllers feature a 32-bit data design. For this reason, stored data objects are placed on a four-byte boundary.

T_M_ECPU_HSBY DDT

You must confirm that the standby controller is ready to assume the primary role before executing a swap command.

Verify that the value of the REMOTE_HSBY_STS.EIO_ERROR bit of the standby controller is 0 before you execute a swap command (either by application logic or in Control Expert).

The `T_M_ECPU_HSBY` / `T_M_ECPU_HSBY_EXT` DDT consists of these objects:

Element	Type	Description	Written by
REMOTE_STS_VALID	BOOL	• True: At least one of the HSBY_LINK_ERROR or HSBY_SUPPLEMENTARY_LINK_ERROR is set to 0. • False (default): Both HSBY_LINK_ERROR and HSBY_SUPPLEMENTARY_LINK_ERROR are set to 1.	System
APP_MISMATCH	BOOL	The original application in the two controllers is different. (Default = false)	System
LOGIC_MISMATCH_ ALLOWED	BOOL	• True: The standby remains standby in case of logic mismatch. • False (default): The standby goes into wait state in case of logic mismatch.	Application
LOGIC_MISMATCH	BOOL	Different revisions of the same application exist in the two controllers. (Default = false)	System
SFC_MISMATCH	BOOL	• True: The applications in the primary controller and the standby controller are different in at least one SFC section. In the event of a switchover, the graphs that are different are reset to their initial state. • False (default): All SFC sections are identical.	System
OFFLINE_BUILD_ MISMATCH	BOOL	The two controllers are running different revisions of the same application.In this condition: • A data exchange between the two controllers may not be possible. • A swap or switchover may not be transparent. • Neither controller can be standby	System

Element	Type	Description	Written by
		(Default = false)	
APP_BUILDCHANGE_DIFF	UINT	The number of build change differences between the applications in the primary controller versus the standby controller. Evaluated by the primary.	System
MAX_APP_ BUILDCHANGE_DIFF	UINT	Maximum number of build change differences permitted by the Hot Standby system, from 0...50 (default = 20). Set in the **Hot Standby** tab as **Number of modifications**.	Application
FW_MISMATCH_ALLOWED	BOOL	Allows mismatched firmware between primary and standby controllers: • True: the standby remains standby in case of FW mismatch. • False (default): the standby goes into wait state in case of FW mismatch. (Default = false)	Application
FW_MISMATCH	BOOL	The OS are different in the two controllers. (Default = false)	System
DATA_LAYOUT_MISMATCH	BOOL	The Data layout are different on the two controllers. The data transfer is partially performed. (Default = false)	System
DATA_DISCARDED	UINT	Number of KB sent by the primary and discarded by the standby (rounded up to the next KB). Represents data for variables added to primary, but not to standby. (Default = 0)	System
DATA_NOT_UPDATED	UINT	Number of KB not updated by the standby (rounded up to the next KB). Represents variables deleted from the primary that remain in the standby. (Default = 0)	System
BACKUP_APP_MISMATCH	BOOL	• False (default): The backup application In the 2 Hot Standby controllers are equal. **NOTE:** The backup application resides in flash memory or on the SD memory card of the controller. It is created either by the **PLC > Project Backup... > Backup Save** command, or by setting the %S66 system bit (Application Backup) to 1. • True: All other cases.	System
PLCA_ONLINE	BOOL	Controller A is configured to enter the primary or standby state. (Default = true) **NOTE:** Executable only on controller A.	Configuration
PLCB_ONLINE	BOOL	Controller B is configured to enter the primary or standby state. (Default = true) **NOTE:** Executable only on controller B.	Configuration

Element	Type	Description	Written by
CMD_SWAP	BOOL	• Set to 1 by program logic or animation table to initiate a switchover. The primary goes into wait, then the standby goes primary, finally the wait goes standby. The command is ignored if there is no standby. **NOTE:** Executable on both primary and standby. • Reset to 0 (default) by the system on switchover completion or if there is no standby. **NOTE:** • This command is designed to be used by the application in response to detected errors. It is not intended to be used for periodic switchovers. • If the application has to switchover periodically, the period between switchovers must not be less than 120 seconds.	Application / System
CMD_APP_TRANSFER	BOOL	• Set to 1 by program logic or animation table to start an application transfer from the primary to the standby. Executable only on the primary. **NOTE:** The application transferred is the backup application, stored in flash memory or on the SD card. If the application running does not match the backup application, perform an application backup (**PLC > Project Backup... > Backup Save** or set the %S66 system bit to 1) before performing the transfer. • Reset to 0 (default) by the system on transfer completion.	Application / System
CMD_RUN_AFTER_ TRANSFER	BOOL[0...2]	• Set to 1 by program logic or animation table to automatically start in Run after a transfer. **NOTE:** Executable only on the primary. • Reset to 0 (default) by the system after transfer completion and: ◦ remote controller is in Run ◦ Controller is not primary ◦ by animation table or logic command	Application / System
CMD_RUN_REMOTE	BOOL	• Set to 1 by program logic or animation table to run the remote controller. This command is ignored if the CMD_STOP_REMOTE is true. **NOTE:** Executable only on the primary. • Reset to 0 (default) by the system when the remote controller enters standby or wait state.	Application / System

Element	Type	Description	Written by
CMD_STOP_REMOTE	BOOL	• Set to 1 by program logic or animation table to stop the remote controller. **NOTE:** Executable on the primary, the standby, or a stopped controller. • Reset to 0 (default) by the application to end the stop command.	Application
CMD_COMPARE_INITIAL_ VALUE	BOOL	• Set to 1 by program logic or animation table to begin a comparison of the initial values of variables exchanged by the two Hot Standby controllers. **NOTE:** Executable on both primary and standby only in Run mode. • Reset to 0 (default) by the system when the comparison is complete, or if the comparison is not possible.	Application / System
INITIAL_VALUE_MISMATCH	BOOL	• True: if the initial values for exchanged variables are different or if the comparison is not possible. • False (false): if the initial values for exchanged variables are identical.	System
MAST_SYNCHRONIZED (1)	BOOL	• True: if the exchanged data from the previous MAST cycle was received by the standby. • False (default): if the exchanged data from at least the previous MAST cycle was not received by the standby. **NOTE:** Closely monitor the MAST_ SYNCHRONIZED and FAST_SYNCHRONIZED variables related to the MAST and FAST tasks as indicated at the end of this table.	System
FAST_SYNCHRONIZED (1)	BOOL	• True: if the exchanged data from the previous FAST cycle was received by the standby. • False (default): if the exchanged data from at least the previous FAST cycle was not received by the standby. **NOTE:** Closely monitor the MAST_ SYNCHRONIZED and FAST_SYNCHRONIZED variables related to the MAST and FAST tasks as indicated at the end of this table.	System
SAFE_SYNCHRONIZED	BOOL	• True: if the exchanged data from the last SAFE cycle was received by the standby. • False (default): if, at least, the exchanged data from the last SAFE cycle was not received by the standby.	System
SAFETY_LOGIC_ MISMATCH	BOOL	• True: the SAFE logic part of the application is different in the two controllers. • False (default): the SAFE logic part of the application is identical in the two controllers.	–

Element	Type	Description	Written by
		NOTE: • The content for this element is determined by comparing system word %SW169 for each controller. • This element is included in T_M_ECPU_HSBY_EXT DDT version 2.80 and later.	
LOCAL_HSBY_STS	T_M_ECPU_HSBY_STS	Hot Standby status for the local controller	(see below)
REMOTE_HSBY_STS	T_M_ECPU_HSBY_STS	Hot Standby status for the remote controller	(see below)
(1): • Closely monitor the MAST_SYNCHRONIZED, FAST_SYNCHRONIZED, and SAFE_SYNCHRONIZED variables related to the MAST, FAST and SAFE tasks. If its value is zero (False), then the database exchanged between the primary and the standby controllers is not transmitted at each cycle. In this situation, change the configured period of this task with a higher value than its current execution time (for the MAST task: %SW0 > %SW30; for the FAST task %SW1 > %SW33; for the SAFE task %SW4 > %SW42. More details on %SW0 + %SW1 and %SW30 + %SW31 in EcoStruxure™ Control Expert, System Bits and Words, Reference Manual). • Example of consequence: upon an Application Program Transfer (APT) command, the primary controller might not be able to transfer the program to the standby controller.			

T_M_ECPU_HSBY_STS Data Type

The `T_M_ECPU_HSBY_STS` / `T_M_ECPU_HSBY_STS_EXT` data type presents the following elements.

NOTE: For firmware version 2.80 and later, the T_M_ECPU_HSBY_STS DDT is named T_M_ECPU_HSBY_STS_EXT.

Element	Type	Description	Written by
HSBY_LINK_ERROR	BOOL	• True: No connection on the Hot Standby link. • False: The Hot Standby link is operational.	System
HSBY_SUPPLEMENTARY_LINK_ERROR	BOOL	• True: No connection on the Ethernet RIO link. • False: The Ethernet RIO link is operational.	System
WAIT	BOOL	• True: The controller is in Run state but waiting to go primary or standby. • False: The controller is in standby, primary or stop state.	System
RUN_PRIMARY	BOOL	• True: The controller is in primary state. • False: The controller is in standby, wait or stop state.	System

Element	Type	Description	Written by
RUN_STANDBY	BOOL	• True: The controller is in standby state. • False: The controller is in primary, wait or stop state.	System
STOP	BOOL	• True: The controller is in stop state. • False: The controller is in primary, standby or wait state.	System
PLC_A	BOOL	• True: the controller A/B/Clear switch, page 58 is in “A” position. • False: the controller switch is not in “A” position.	System
PLC_B	BOOL	• True: the controller A/B/Clear switch, page 58 is in “B” position. • False: the controller switch is not in “B” position.	System
EIO_ERROR	BOOL	• True: The controller does not detect any of the configured Ethernet RIO drops. • False: The controller detects at least one configured Ethernet RIO drop. **NOTE:** This bit is always false when no drop is configured.	System
SD_CARD_PRESENT	BOOL	• True: A valid SD card is inserted. • False: No SD card, or an invalid SD card is inserted.	System
LOCAL_RACK_STS	BOOL]	• True: The local rack configuration is OK. • False: The local rack configuration is not OK (for example, modules missing or in incorrect slots, etc.)	Application
MAST_TASK_STATE	BYTE	State of the MAST task: • 0: Not existent • 1: Stop • 2: Run • 3: Breakpoint • 4: Halt	System
FAST_TASK_STATE	BYTE	State of the FAST task: • 0: Not existent • 1: Stop • 2: Run • 3: Breakpoint • 4: Halt	System

Element	Type	Description	Written by
SAFE_TASK_STATE	BYTE	State of the SAFE task: • 0: Not existent • 1: Stop • 2: Run • 3: Breakpoint • 4: Halt **NOTE:** This element is included in T_M_ECPU_HSBY_STS_EXT DDT version 2.80 and later.	System
REGISTER	WORD[0...63]	Unmanaged data added to the application via the **Exchange on STBY** attribute.	Application

Explicit Messaging

Introduction

You can configure EtherNet/IP and Modbus TCP explicit messages for the M580 CPU in the following ways:

- Connect the CPU to a Control Expert project (see Modicon M580 Standalone, System Planning Guide for, Frequently Used Architectures).
- Use the DATA_EXCH function block in application logic to transmit EtherNet/IP or Modbus TCP explicit messages.
- Use a WRITE_VAR or a READ_VAR function block to exchange Modbus TCP explicit messages, for example, service data objects (SDOs).

NOTE: A single Control Expert application can contain more than 16 explicit messaging blocks, but only 16 explicit messaging blocks can be active at the same time.

Configuring Explicit Messaging Using DATA_EXCH

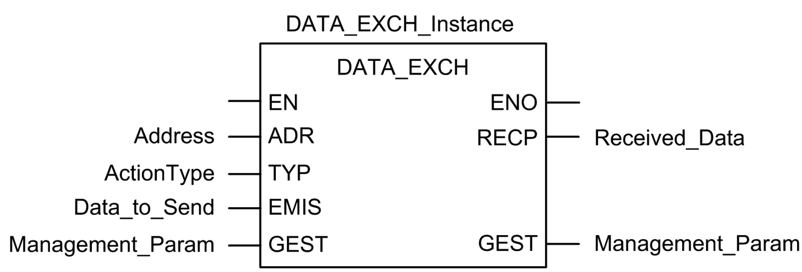
Overview

Use the `DATA_EXCH` function block to configure both Modbus TCP explicit messages and connected and unconnected EtherNet/IP explicit messages.

The `Management_Param`, the `Data_to_Send`, and the `Received_Data` parameters define the operation.

`EN` and `ENO` can be configured as additional parameters.

FBD Representation



Input Parameters

Parameter	Data type	Description
EN	BOOL	This parameter is optional. When this input is set to one, the block is activated and can solve the function blocks algorithm. When this input is set to zero, the block is deactivated and won't solve the function block algorithm.
Address	Array [0...7] of INT	The path to the destination device, the content of which can vary depending on the message protocol. Use the Address function as an is input to the block parameter ADR.. Refer to a description of the Address parameter for: • EtherNet/IP messages, page 318 • Modbus TCP messages (see Modicon M340, BMX NOC 0401 Ethernet Communication Module, User Manual)
ActionType	INT	The type of action to perform. For both the EtherNet/IP and Modbus TCP protocols, this setting = 1 (transmission followed by await reception).
Data_to_Send	Array [n...m] of INT	The content of this parameter is specific to the protocol, either EtherNet/IP or Modbus TCP. For EtherNet/IP explicit messaging, refer to the topic Configuring the Data_To_Send Parameter, page 318. For Modbus TCP explicit messaging, refer to Control Expert online help.

Input/Output Parameters

The Management_Param array is local:

Parameter	Data type	Description
Management_Param	Array [0...3] of INT	The management parameter, page 314, consisting of four words.

Do not copy this array during a switchover from a primary to a standby CPU in a Hot Standby system. Uncheck the **Exchange On STBY** variable in Control Expert when you configure a Hot Standby system.

NOTE: Refer to the description of Hot Standby system data management and the T_M_ ECPU_HSBY DDT (see Modicon M580 Hot Standby, System Planning Guide for, Frequently Used Architectures) in the M580 Hot Standby System Planning Guide (see Modicon M580 Hot Standby, System Planning Guide for, Frequently Used Architectures).

Output Parameters

Parameter	Data type	Description
ENO	BOOL	This parameter is optional. When you select this output you also get the EN input. ENO output is activated upon successful execution of the function block.
Received_Data	Array [n...m] of INT	The EtherNet/IP (CIP) response, page 319 or the Modbus TCP response (see Modicon M340, BMX NOC 0401 Ethernet Communication Module, User Manual). The structure and content depends upon the specific protocol.

Configuring the DATA_EXCH Management Parameter

Introduction

The structure and content of the management parameter of the `DATA_EXCH` block is common to both EtherNet/IP and Modbus TCP explicit messaging.

Configuring the Management Parameter

The management parameter consists of four contiguous words:

<table>
<tr><th rowspan="2">Data source</th><th rowspan="2">Register</th><th colspan="2">Description</th></tr>
<tr><th>High Byte (MSB)</th><th>Low Byte (LSB)</th></tr>
<tr><td rowspan="2">Data managed by the system</td><td><code>Management_Param[0]</code></td><td>Exchange number</td><td>Two read-only bits:
• Bit 0 = Activity bit, page 314
• Bit 1 = Cancel bit</td></tr>
<tr><td><code>Management_Param[1]</code></td><td>Operation report (see Modicon M580 Standalone, System Planning Guide for, Frequently Used Architectures)</td><td>Communication report (see Modicon M580 Standalone, System Planning Guide for, Frequently Used Architectures)</td></tr>
<tr><td rowspan="2">Data managed by the user</td><td><code>Management_Param[2]</code></td><td colspan="2">Block timeout. Values include:
• 0 = infinite wait
• other values = timeout x 100 ms, for example:
  ◦ 1 = 100 ms
  ◦ 2 = 200 ms</td></tr>
<tr><td><code>Management_Param[3]</code></td><td colspan="2">Length of data sent or received:
• Input (before sending the request): length of data in the <code>Data_to_Send</code> parameter, in bytes
• Output (after response): length of data in the <code>Received_Data</code> parameter, in bytes</td></tr>
</table>

Activity Bit

The activity bit is the first bit of the first element in the table. The value of this bit indicates the execution status of the communication function:

- **1**: The bit is set to 1 when the function launches.

- **0**: The bit returns to 0 upon the completion of the execution. (The transition from 1 to 0 increments the exchange number. If an error is detected during the execution, search for the corresponding code in the operation and communication report (see Modicon M580 Standalone, System Planning Guide for, Frequently Used Architectures).)

For example, you can make this declaration in the management table:

`Management_Param[0] ARRAY [0..3] OF INT`

For that declaration, the activity bit corresponds to this notation:

`Management_Param[0].0`

NOTE: The notation previously used requires configuration of the project properties in such a way as to authorize the extraction of bits on integer types. If this is not the case, `Management_Param[0].0` cannot be accessed in this manner.

Explicit Messaging Services

Overview

Every explicit message performs a service. Each service is associated with a service code. Identify the explicit messaging service by its name, decimal number, or hexadecimal number.

You can execute explicit messages using the `DATA_EXCH` function block in the Control Expert DTM.

Services

The services available in Control Expert include, but are not limited to, these service codes:

Service Code		Description	Available in...	
Hex	Dec		DATA_EXCH block	Control Expert GUI
1	1	Get_Attributes_All	X	X
2	2	Set_Attributes_All	X	X
3	3	Get_Attribute_List	X	—
4	4	Set_Attribute_List	X	—
5	5	Reset	X	X
6	6	Start	X	X
7	7	Stop	X	X
8	8	Create	X	X
9	9	Delete	X	X
A	10	Multiple_Service_Packet	X	—
B-C	11-12	(*Reserved*)	—	—
D	13	Apply_Attributes	X	X
E	14	Get_Attribute_Single	X	X
10	16	Set_Attribute_Single	X	X
11	17	Find_Next_Object_Instance	X	X
14	20	Error Response (DeviceNet only)	—	—
15	21	Restore	X	X
16	22	Save	X	X

Service Code		Description	Available in...	
Hex	Dec		DATA_EXCH block	Control Expert GUI
17	23	No Operation (NOP)	X	X
18	24	Get_Member	X	X
19	25	Set_Member	X	X
1A	26	Insert_Member	X	X
1B	27	Remove_Member	X	X
1C	28	GroupSync	X	—
1D-31	29-49	(*Reserved*)	—	—
"X" indicates the service is available. "—" indicates the service is not available.				

Configuring EtherNet/IP Explicit Messaging Using DATA_EXCH

Configuring the Address Parameter

To configure the Address parameter, use the `ADDM` function to convert the character string, described below, to an address that is input into the ADR parameter of the DATA_EXCH block:

ADDM('rack.slot.channel{ip_address}message_type.protocol'), where:

This field...	Represents...
rack	the number assigned to the rack containing the communication module
slot	the position of the communication module in the rack
channel	the communication channel—set to a value of **0**
ip_address	the IP address of the remote device, for example 193.168.1.6
message_type	the type of message, presented as a three character string—either: • **UNC** (indicating an unconnected message), or • **CON** (indicating a connected message)
protocol	the protocol type—the three character string **CIP**

Configuring the Data_to_Send Parameter

The `Data_to_Send` parameter varies in size. It consists of contiguous registers that include —in sequence—both the message type and the CIP request:

Offset (words)	Length (bytes)	Data Type	Description
0	2 bytes	Bytes	Message type: • High byte = size of the request in words • Low byte = EtherNet/IP service code
1	`Management_Param[3]` (size of `Data_to_Send`) minus 2	Bytes	The CIP request[1]. **NOTE:** The structure and size of the CIP request depends on the EtherNet/IP service.
1 Structure the CIP request in little endian order.			

Contents of the Received_Data Parameter

The `Received_Data` parameter contains only the CIP response. The length of the CIP response varies, and is reported by `Management_Param[3]` after the response is received. The format of the CIP response is described, below:

Offset (words)	Length (bytes)	Data Type	Description
0	2	Byte	• High byte (MSB) = reserved • Low byte (LSB): reply service
1	2	Byte	• High byte (MSB): length of additional status • Low byte (LSB): EtherNet/IP general status (see Modicon M340, BMX NOC 0401 Ethernet Communication Module, User Manual)
2	length of additional status	Byte array	Additional Status[1]
...	`Management_Param[3]` (size of `Received_Data`) minus 4, and minus the additional status length	Byte array	Response data
1. Refer to *The CIP Networks Library, Volume 1, Common Industrial Protocol* at section 3-5.6 *Connection Manager Object Instance Error Codes*.			

NOTE: The response is structured in little endian order.

Checking the Received_Data Response for System and CIP Status

Use the contents of the Received_Data parameter to check both the system status and the CIP status of the Ethernet communication module when handling the explicit message.

First:	Check the value of the high byte (MSB) of the first response word, positioned at offset 0. If the value of this byte is: • equal to 0: the system properly handled the explicit message • not equal to 0: a system-based event occurred Refer to the list of EtherNet/IP Explicit Messaging Event Codes (see Modicon M580 Standalone, System Planning Guide for, Frequently Used Architectures) for an explanation of the system-based event code contained in the second response word, positioned at offset 1.
Next:	If the system properly handled the explicit message, and the high byte of the first response word equals 0, check the value of the second response word, positioned at offset 1. If the value of this word is: • equal to 0: the explicit message was properly handled by the CIP protocol • not equal to 0: a CIP protocol-based event occurred Refer to your CIP documentation for an explanation of the CIP status displayed in this word.

EtherNet/IP Explicit Message Example: Get_Attribute_Single

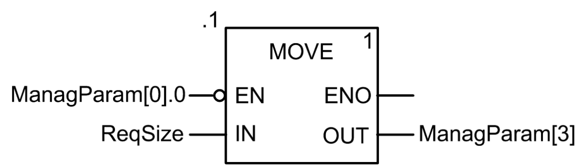
Overview

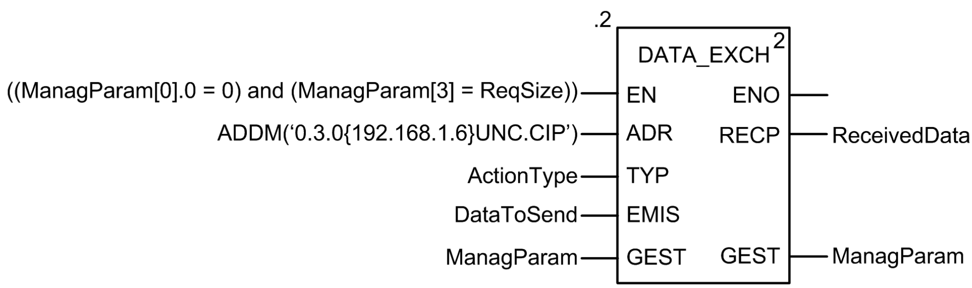
The following unconnected explicit messaging example shows you how to use the DATA_EXCH function block to retrieve diagnostic data from a remote device (at IP address 192.168.1.6). This example is executing a Get_Attribute_Single of assembly instance 100, attribute 3.

You can perform the same explicit messaging service using the **EtherNet/IP Explicit Message** window (see Modicon M580, BMENOC0301/0311 Ethernet Communications Module, Installation and Configuration Guide).

Implementing the DATA_EXCH Function Block

To implement the DATA_EXCH function block, create and assign variables for the following blocks:





Configuring the Address Variable

The Address variable identifies the explicit message originating device (in this example, the communication module) and the target device. Note that the Address variable does not include the Xway address elements {Network.Station} because we are not bridging through another PLC station. As an example, use the `ADDM` function to convert the following character string to an address:

`ADDM`('0.1.0{192.168.1.6}UNC.CIP'), where:

- rack = 0
- module (slot number) = 1
- channel = 0
- remote device IP address = 192.168.1.6
- message type = unconnected
- protocol = CIP

Configuring the ActionType Variable

The ActionType variable identifies the function type for the `DATA_EXCH` function block:

Variable	Description	Value (hex)
ActionType	Transmission followed by wait for response	16#01

Configuring the DataToSend Variable

The DataToSend variable identifies the details of the CIP explicit message request:

Variable	Description	Value (hex)
DataToSend[0]	CIP request service information: • High byte = request size in words: 16#03 (3 decimal) • Low byte = service code: 16#0E (14 decimal)	16#030E
DataToSend[1]	CIP request class information: • High byte = class: 16#04 (4 decimal) • Low byte = class segment: 16#20 (32 decimal)	16#0420
DataToSend[2]	CIP request instance information: • High byte = instance: 16#64 (100 decimal) • Low byte = instance segment: 16#24 (36 decimal)	16#6424
DataToSend[3]	CIP request attribute information: • High byte = attribute: 16#03 (3 decimal) • Low byte = attribute segment: 16#30 (48 decimal)	16#0330

Viewing the Response

Use a Control Expert Animation table to display the ReceivedData variable array. Note that the ReceivedData variable array consists of the entire data buffer.

To display the CIP response, follow these steps:

<table>
<tr><th>Step</th><th colspan="2">Action</th></tr>
<tr><td>1</td><td colspan="2">In Control Expert, select Tools → Project Browser to open the Project Browser.</td></tr>
<tr><td>2</td><td colspan="2">In the Project Browser, select the Animation Tables folder, then click the right mouse button. A pop-up menu appears.</td></tr>
<tr><td>3</td><td colspan="2">Select New Animation Table in the pop-up menu. A new animation table and its properties dialog both open.</td></tr>
<tr><td rowspan="5">4</td><td colspan="2">In the Properties dialog, edit the following values:</td></tr>
<tr><td>Name</td><td>Type in a table name. For this example: ReceivedData.</td></tr>
<tr><td>Functional module</td><td>Accept the default <None>.</td></tr>
<tr><td>Comment</td><td>(Optional) Type your comment here.</td></tr>
<tr><td>Number of animated characters</td><td>Type in 100, representing the size of the data buffer in words.</td></tr>
<tr><td>5</td><td colspan="2">Click OK to close the dialog.</td></tr>
<tr><td>6</td><td colspan="2">In the animation table’s Name column, type the name of the variable assigned to the RECP pin: ReceivedData and press Enter. The animation table displays the ReceivedData variable.</td></tr>
<tr><td>7</td><td colspan="2">Expand the ReceivedData variable to display its word array, where you can view the CIP response contained in the ReceivedData variable.

NOTE: Each array entry presents 2 bytes of data in little endian format, where the least significant byte is stored in the smallest memory address. For example, ‘8E’ in word[0] is the lower byte, and ‘00’ is the upper byte.</td></tr>
</table>

EtherNet/IP Explicit Message Example: Read Modbus Object

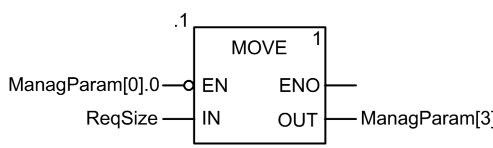
Overview

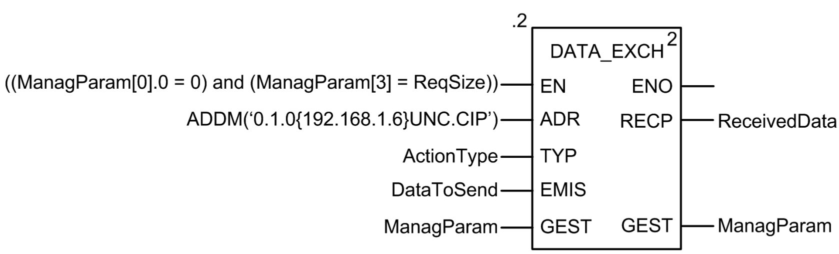
The following unconnected explicit messaging example shows you how to use the DATA_EXCH function block to read data from a remote device (for example, the STB NIP 2212 network interface module at IP address 192.168.1.6) using the Read_Holding_Registers service of the Modbus Object.

You can perform the same explicit messaging service using the **EtherNet/IP Explicit Message** window (see Modicon M580, BMENOC0301/0311 Ethernet Communications Module, Installation and Configuration Guide).

Implementing the DATA_EXCH Function Block

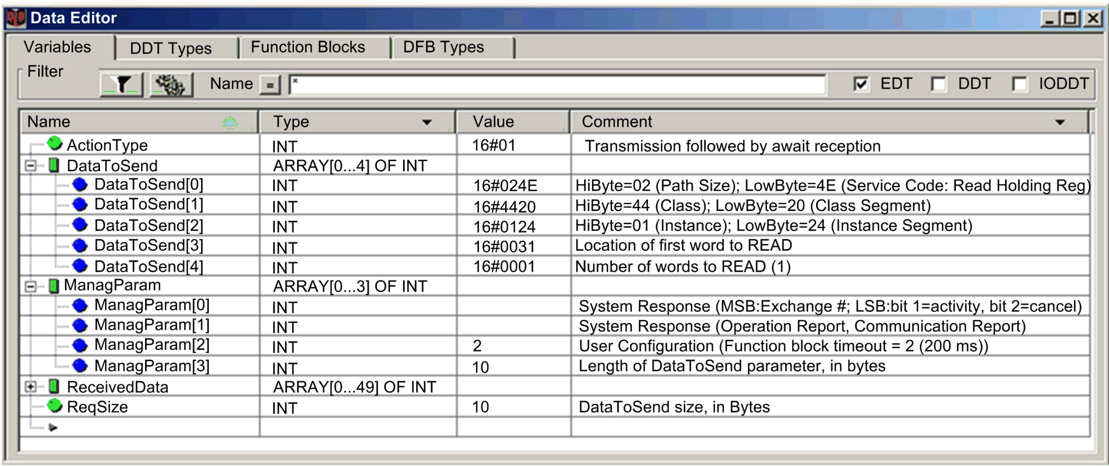
To implement the DATA_EXCH function block, you need to create and assign variables for the following blocks:





Declaring Variables

In this example, the following variables were defined. You can, of course, use different variable names in your explicit messaging configurations.



Name	Type	Value	Comment
ActionType	INT	16#01	Transmission followed by await reception
DataToSend	ARRAY[0...4] OF INT		
DataToSend[0]	INT	16#024E	HiByte=02 (Path Size); LowByte=4E (Service Code: Read Holding Reg)
DataToSend[1]	INT	16#4420	HiByte=44 (Class); LowByte=20 (Class Segment)
DataToSend[2]	INT	16#0124	HiByte=01 (Instance); LowByte=24 (Instance Segment)
DataToSend[3]	INT	16#0031	Location of first word to READ
DataToSend[4]	INT	16#0001	Number of words to READ (1)
ManagParam	ARRAY[0...3] OF INT		
ManagParam[0]	INT		System Response (MSB:Exchange #; LSB:bit 1=activity, bit 2=cancel)
ManagParam[1]	INT		System Response (Operation Report, Communication Report)
ManagParam[2]	INT	2	User Configuration (Function block timeout = 2 (200 ms))
ManagParam[3]	INT	10	Length of DataToSend parameter, in bytes
ReceivedData	ARRAY[0...49] OF INT		
ReqSize	INT	10	DataToSend size, in Bytes

Configuring the Address Variable

The Address variable identifies the explicit message originating device (in this example, the Ethernet communication module) and the target device. Note that the Address variable does not include the Xway address elements {Network.Station} because we are not bridging through another PLC station. Use the `ADDM` function to convert the following character string to an address:

`ADDM`('0.1.0{192.168.1.6}UNC.CIP'), where:

- rack = 0
- module (slot number) = 1
- channel = 0
- remote device IP address = 192.168.1.6
- message type = unconnected
- protocol = CIP

Configuring the ActionType Variable

The ActionType variable identifies the function type for the `DATA_EXCH` function block:

Variable	Description	Value (hex)
ActionType	Transmission followed by wait for response	16#01

Configuring the DataToSend Variable

The DataToSend variable identifies the type of explicit message and the CIP request:

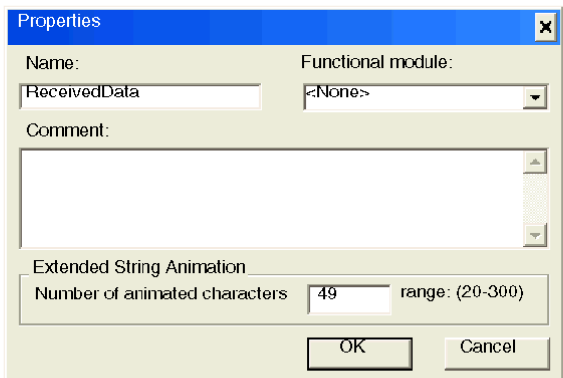
Variable	Description	Value (hex)
DataToSend[0]	CIP request service information: • High byte = request size in words: 16#02 (2 decimal) • Low byte = service code: 16#4E (78 decimal)	16#024E
DataToSend[1]	CIP request class information: • High byte = class: 16#44 (68 decimal) • Low byte = class segment: 16#20 (32 decimal)	16#4420
DataToSend[2]	CIP request instance information: • High byte = instance: 16#01 (1 decimal) • Low byte = instance segment: 16#24 (36 decimal)	16#0124
DataToSend[3]	Location of first word to be read): • High byte = 16#00 (0 decimal) • Low byte = 16#31 (49 decimal)	16#0031
DataToSend[4]	Number of words to read: • High byte = attribute: 16#00 (0 decimal) • Low byte = attribute segment: 16#01 (1 decimal)	16#0001

Viewing the Response

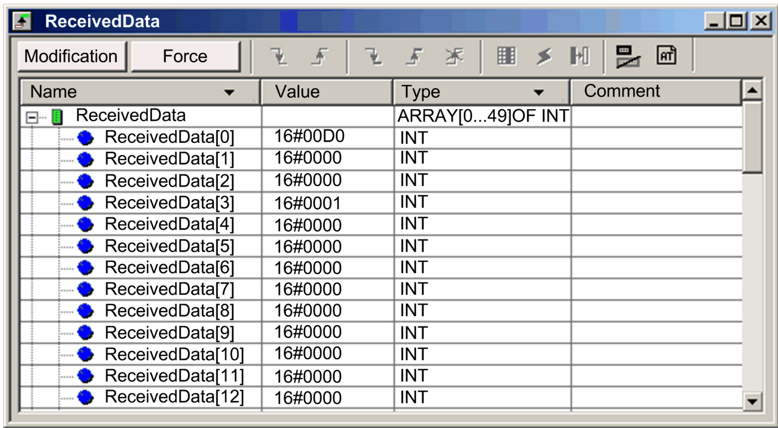
Use a Control Expert Animation table to display the ReceivedData variable array. Note that the ReceivedData variable array consists of the entire data buffer.

To display the CIP response, follow these steps:

Step	Action
1	In Control Expert, select **Tools** → **Project Browser** to open the Project Browser.
2	In the Project Browser, select the **Animation Tables** folder, then click the right mouse button. A pop-up menu appears.
3	Select **New Animation Table** in the pop-up menu. A new animation table and its properties dialog both open.

<table>
<tr><th>Step</th><th colspan="2">Action</th></tr>
<tr><td rowspan="5">4</td><td colspan="2">In the Properties dialog, edit the following values:</td></tr>
<tr><td>Name</td><td>Type in a table name. For this example: ReceivedData.</td></tr>
<tr><td>Functional module</td><td>Accept the default <None>.</td></tr>
<tr><td>Comment</td><td>(Optional) Type your comment here.</td></tr>
<tr><td>Number of animated characters</td><td>Type in 49, representing the size of the data buffer in words.</td></tr>
<tr><td>5</td><td colspan="2">The completed Properties dialog looks like this:


Click OK to close the dialog.</td></tr>
</table>

Step	Action
6	In the animation table's **Name** column, type in the name of the variable assigned to the RECP pin: **ReceivedData** and hit **Enter**. The animation table displays the ReceivedData variable.
7	Expand the ReceivedData variable to display its word array, where you can view the CIP response contained in the ReceivedData variable:  **Note:** Each array entry presents 2 bytes of data in little endian format, where the least significant byte is stored in the smallest memory address. For example, 'CE' in word[0] is the lower byte, and '00' is the upper byte.

EtherNet/IP Explicit Message Example: Write Modbus Object

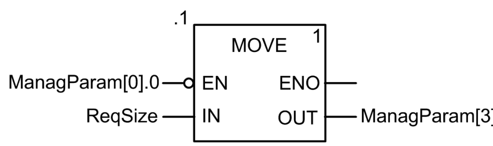
Overview

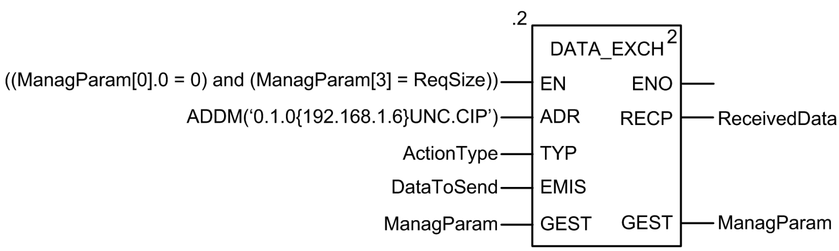
The following unconnected explicit messaging example shows you how to use the DATA_EXCH function block to write data to a remote device at IP address 192.168.1.6 using the Write_Holding_Registers service of the Modbus object.

You can perform the same explicit messaging service using the **EtherNet/IP Explicit Message** window (see Modicon M580, BMENOC0301/0311 Ethernet Communications Module, Installation and Configuration Guide) in the Control Expert DTM.

Implementing the DATA_EXCH Function Block

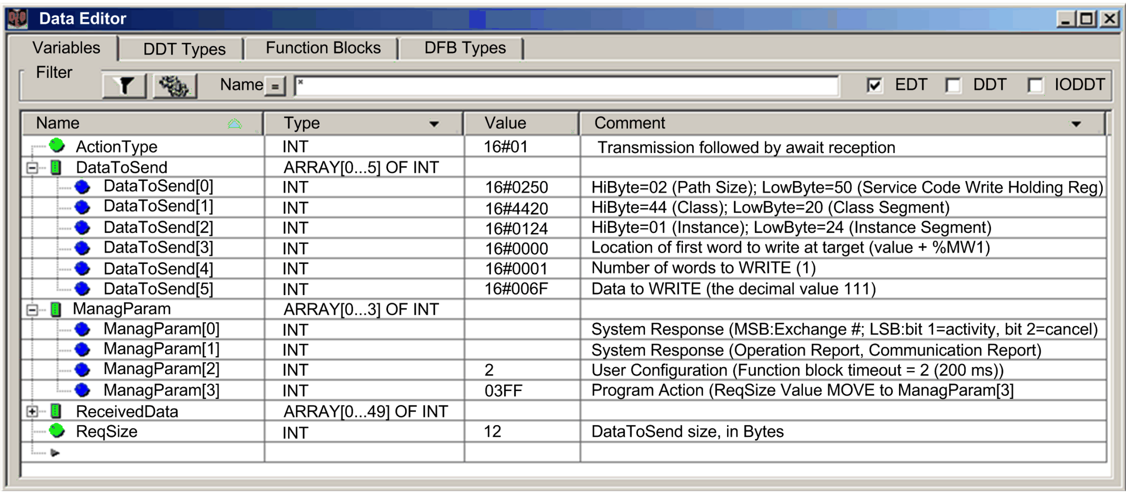
To implement the DATA_EXCH function block, you need to create and assign variables for the following blocks:





Declaring Variables

In this example, the following variables were defined. You can, of course, use different variable names in your explicit messaging configurations.



Name	Type	Value	Comment
ActionType	INT	16#01	Transmission followed by await reception
DataToSend	ARRAY[0...5] OF INT		
DataToSend[0]	INT	16#0250	HiByte=02 (Path Size); LowByte=50 (Service Code Write Holding Reg)
DataToSend[1]	INT	16#4420	HiByte=44 (Class); LowByte=20 (Class Segment)
DataToSend[2]	INT	16#0124	HiByte=01 (Instance); LowByte=24 (Instance Segment)
DataToSend[3]	INT	16#0000	Location of first word to write at target (value + %MW1)
DataToSend[4]	INT	16#0001	Number of words to WRITE (1)
DataToSend[5]	INT	16#006F	Data to WRITE (the decimal value 111)
ManagParam	ARRAY[0...3] OF INT		
ManagParam[0]	INT		System Response (MSB:Exchange #; LSB:bit 1=activity, bit 2=cancel)
ManagParam[1]	INT		System Response (Operation Report, Communication Report)
ManagParam[2]	INT	2	User Configuration (Function block timeout = 2 (200 ms))
ManagParam[3]	INT	03FF	Program Action (ReqSize Value MOVE to ManagParam[3]
ReceivedData	ARRAY[0...49] OF INT		
ReqSize	INT	12	DataToSend size, in Bytes

Configuring the Address Variable

The Address variable identifies the explicit message originating device (in this example, the communication module) and the target device. Note that the Address variable does not include the Xway address elements {Network.Station} because we are not bridging through another PLC station. Use the `ADDM` function to convert the following character string to an address:

`ADDM`('0.1.0{192.168.1.6}UNC.CIP'), where:

- rack = 0
- module (slot number) = 1
- channel = 0
- remote device IP address = 192.168.1.6
- message type = unconnected
- protocol = CIP

Configuring the ActionType Variable

The ActionType variable identifies the function type for the `DATA_EXCH` function block:

Variable	Description	Value (hex)
ActionType	Transmission followed by wait for response	16#01

Configuring the DataToSend Variable

The DataToSend variable identifies the type of explicit message and the CIP request:

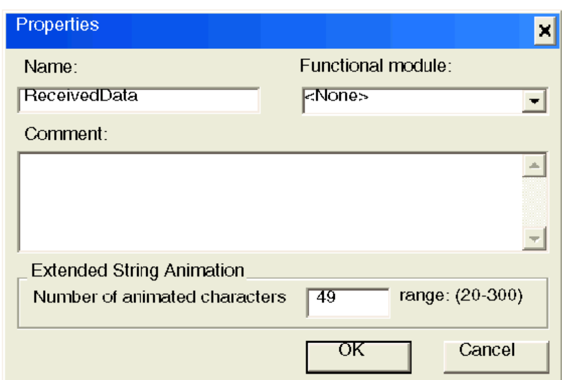
Variable	Description	Value (hex)
DataToSend[0]	CIP request service information: • High byte = request size in words: 16#02 (2 decimal) • Low byte = service code: 16#50 (80 decimal)	16#0250
DataToSend[1]	CIP request class information: • High byte = class: 16#44 (68 decimal) • Low byte = class segment: 16#20 (32 decimal)	16#4420
DataToSend[2]	CIP request instance information: • High byte = instance: 16#01 (1 decimal) • Low byte = instance segment: 16#24 (36 decimal)	16#0124
DataToSend[3]	Location of first word to write (+ %MW1): • High byte = 16#00 (0 decimal) • Low byte = 16#00 (0 decimal)	16#0000
DataToSend[4]	Number of words to write: • High byte = attribute: 16#00 (0 decimal) • Low byte = attribute segment: 16#01 (1 decimal)	16#0001
DataToSend[5]	Data to write: • High byte = attribute: 16#00 (0 decimal) • Low byte = attribute segment: 16#6F (111 decimal)	16#006F

Viewing the Response

Use a Control Expert Animation table to display the ReceivedData variable array. Note that the ReceivedData variable array consists of the entire data buffer.

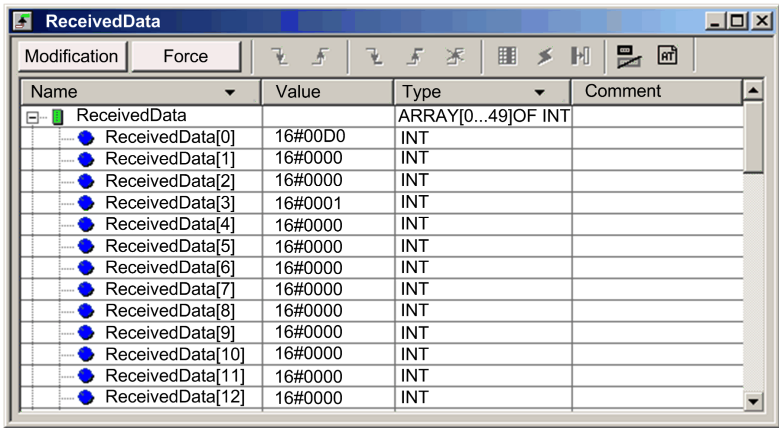
To display the CIP response, follow these steps:

<table>
<tr><th>Step</th><th colspan="2">Action</th></tr>
<tr><td>1</td><td colspan="2">In Control Expert, select Tools → Project Browser to open the Project Browser.</td></tr>
<tr><td>2</td><td colspan="2">In the Project Browser, select the Animation Tables folder, then click the right mouse button. A pop-up menu appears.</td></tr>
<tr><td>3</td><td colspan="2">Select New Animation Table in the pop-up menu. A new animation table and its properties dialog both open.</td></tr>
<tr><td rowspan="5">4</td><td colspan="2">In the Properties dialog, edit the following values:</td></tr>
<tr><td>Name</td><td>Type in a table name. For this example: ReceivedData.</td></tr>
<tr><td>Functional module</td><td>Accept the default <None>.</td></tr>
<tr><td>Comment</td><td>(Optional) Type your comment here.</td></tr>
<tr><td>Number of animated characters</td><td>Type in 49, representing the size of the data buffer in words.</td></tr>
<tr><td>5</td><td colspan="2">The completed Properties dialog looks like this:



Click OK to close the dialog.</td></tr>
</table>

<table>
<tr><th>Step</th><th>Action</th></tr>
<tr><td>6</td><td>In the animation table’s Name column, type in the name of the variable assigned to the RECP pin: ReceivedData and hit Enter. The animation table displays the ReceivedData variable.</td></tr>
<tr><td>7</td><td>Expand the ReceivedData variable to display its word array, where you can view the CIP response contained in the ReceivedData variable:



Note: Each array entry presents 2 bytes of data in little endian format, where the least significant byte is stored in the smallest memory address. For example, ‘D0’ in word[0] is the lower byte, and ‘00’ is the upper byte.</td></tr>
</table>

Modbus TCP Explicit Messaging Function Codes

Overview

You can execute Modbus TCP explicit messages using either a Control Expert `DATA_EXCH` function block or the Modbus Explicit Message Window.

NOTE: Configuration edits made to an Ethernet module are not saved to the operating parameters stored in the CPU and, therefore, are not sent by the CPU to the module on startup.

Function Codes

The function codes supported by the Control Expert graphical user interface include the following standard explicit messaging functions:

Function Code (dec)	Description
1	Read bits (%M)
2	Read input bits (%I)
3	Read words (%MW)
4	Read input words (%IW)
15	Write bits (%M)
16	Write words (%MW)

NOTE: You can use the `DATA_EXCH` function block to execute any Modbus function, via program logic. Because the available function codes are too numerous to list here, refer instead to the Modbus IDA website for more information about these Modbus functions, at http://www.Modbus.org.

Configuring Modbus TCP Explicit Messaging Using DATA_EXCH

Introduction

When you use the `DATA_EXCH` block to create an explicit message for a Modbus TCP device, configure this block the same way you would configure it for any other Modbus communication. Refer to the Control Expert online help for instructions on how to configure the `DATA_EXCH` block.

Configuring ADDM Block Unit ID Settings

When you configure the `DATA_EXCH` block, use the `ADDM` block to set the `DATA_EXCH` block's Address parameter. The `ADDM` block presents the configuration format ADDM('rack.slot.channel[ip_address]UnitID.message_type.protocol') where:

Parameter	Description
rack	the number assigned to the rack containing the communication module
slot	the position of the communication module in the rack
channel	the communication channel (set to a value of 0)
ip_address	the IP address of the remote device (for example, 192.168.1.7)
Unit ID	the destination node address, also known as the Modbus Plus on Ethernet Transporter (MET) mapping index value
message_type	the three-character string **TCP**
protocol	the three-character string **MBS**

The Unit ID value in a Modbus message indicates the destination of the message.

Refer to the Modbus diagnostic codes.

Contents of the Received_Data Parameter

The `Received_Data` parameter contains the Modbus response. The length of the response varies, and is reported by `Management_Param[3]` after the response is received. The format of the Modbus response is described, below:

Offset (words)	Length (bytes)	Description

0	2	First word of the Modbus response:
		• High byte (MSB):
		  ◦ if successful: Modbus Function Code
		  ◦ if not: Modbus function code + 16#80
		• Low byte (LSB):
		  ◦ if successful: depends on the request
		  ◦ if not: Modbus exception code

1	Length of the `Received_Data` parameter – 2	Remainder of the Modbus response: depends on the specific Modbus request)

NOTE:

- Structure the response in little endian order.
- In some cases of detected errors, Received_Data is also used to judge the type of detected error along with Management_Param.

Modbus TCP Explicit Message Example: Read Register Request

Introduction

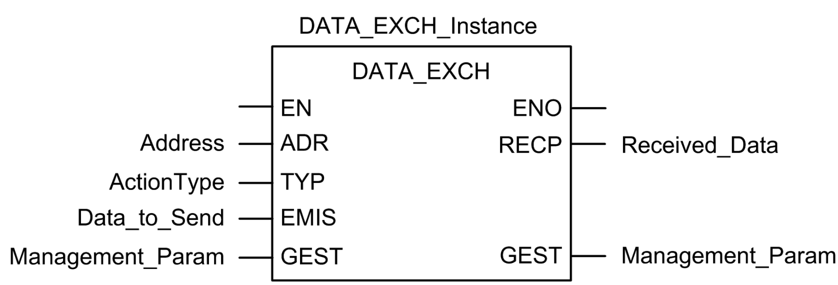
Use the DATA_EXCH function block to send a Modbus TCP explicit message to a remote device at a specific IP address to read a single word located in the remote device.

The Management_Param, the Data_to_Send, and the Received_Data parameters define the operation.

EN and ENO can be configured as additional parameters.

Implementing the DATA_EXCH Function Block

To implement the DATA_EXCH function block, create and assign variables for the for following:



Configuring the Address Variable

The Address variable identifies the explicit message originating device and the target device. Note that the Address variable does not include the Xway address elements {Network.Station} because you are not bridging through another PAC station. Use the ADDM function to convert the following character string to an address:

`ADDM`(‘0.1.0{192.168.1.7}TCP.MBS’), where:

- rack = 0
- module (slot number) = 1
- channel = 0
- remote device IP address = 192.168.1.7
- message type = TCP
- protocol = Modbus

Configuring the ActionType Variable

The ActionType variable identifies the function type for the `DATA_EXCH` function block:

Variable	Description	Value (hex)
ActionType	Transmission followed by wait for response	16#01

Configuring the DataToSend Variable

The DataToSend variable contains the target register address and the number of registers to read:

Variable	Description	Value (hex)
DataToSend[0]	• High byte = Most significant byte (MSB) of register address 16#15 (21 decimal) • Low byte = function code: 16#03 (03 decimal)	16#1503
DataToSend[1]	• High byte = Most significant byte (MSB) of the number of registers to read: 16#00 (0 decimal) • Low byte = Least significant byte (LSB) of register address: 16#0F (15 decimal)	16#000F
DataToSend[2]	CIP request instance information: • High byte = not used: 16#00 (0 decimal) • Low byte = Least significant byte (LSB) of the number of registers to read: 16#01 (1 decimal)	16#0001

NOTE: For detailed information about M580 network topologies, refer to the *Modicon M580 Standalone System Planning Guide for Frequently Used Architectures* and *Modicon M580 System Planning Guide for Complex Topologies*.

Viewing the Response

Use a Control Expert Animation table to display the ReceivedData variable array. Note that the ReceivedData variable array consists of the entire data buffer.

To display the Modbus TCP response, follow these steps:

<table>
<tr><th>Step</th><th colspan="2">Action</th></tr>
<tr><td>1</td><td colspan="2">In Control Expert, select Tools > Project Browser.</td></tr>
<tr><td>2</td><td colspan="2">In the Project Browser, select the Animation Tables folder, and click the right mouse button.

Result: A pop-up menu appears.</td></tr>
<tr><td>3</td><td colspan="2">Select New Animation Table in the pop-up menu.

Result: A new animation table and its properties dialog open.</td></tr>
<tr><td rowspan="5">4</td><td colspan="2">In the Properties dialog, edit the following values:</td></tr>
<tr><td>Name</td><td>Type in a table name. For this example: ReceivedData.</td></tr>
<tr><td>Functional module</td><td>Accept the default <None>.</td></tr>
<tr><td>Comment</td><td>(Optional) Type your comment here.</td></tr>
<tr><td>Number of animated characters</td><td>Type in 100, representing the size of the data buffer in words.</td></tr>
<tr><td>5</td><td colspan="2">Click OK to close the dialog.</td></tr>
<tr><td>6</td><td colspan="2">In the animation table's Name column, type in the name of the variable assigned to the databuffer: ReceivedData and press Enter.

Result: The animation table displays the ReceivedData variable.</td></tr>
<tr><td>7</td><td colspan="2">Expand the ReceivedData variable to display its word array, where you can view the CIP response contained in the ReceivedData variable.

NOTE: Each array entry presents 2 bytes of data in little endian format. For example, ‘03’ in word [0] is the low byte, and ‘02’ is the high byte.</td></tr>
</table>

Sending Explicit Messages to EtherNet/IP Devices

Introduction

Use the **EtherNet/IP Explicit Message** window to send an explicit message from Control Expert to the M580 CPU.

An explicit message can be connected or unconnected:

- **connected:** A connected explicit message contains both path information and a connection identifier to the target device.
- **unconnected:** An unconnected message requires path (addressing) information that identifies the destination device (and, optionally, device attributes).

You can use explicit messaging to perform many different services. Not every EtherNet/IP device supports every service.

Accessing the Page

Before you can perform explicit messaging, connect the DTM for the M580 CPU to the CPU itself:

Step	Action
1	Open the **DTM Browser** in Control Expert (**Tools > DTM Browser**).
2	Select the M580 DTM in the **DTM Browser**.
3	Right-click the M580 DTM.
4	Scroll to the EtherNet/IP explicit messaging page (**Device menu > Additional functions > EtherNet/IP Explicit Message**).

Configuring Settings

Configure the explicit message using these settings on the **EtherNet/IP Explicit Messaging** page:

Field	Setting
Address	**IP Address:** The IP address of the target device that is used to identify the target of the explicit message.
	Class: The **Class** integer (1 ... 65535) is the identifier of the target device that is used in the construction of the message path.
	Instance: The **Instance** integer (0 ... 65535) is the class instance of the target device that is used in the construction of the message path.
	Attribute: Check this box to enable the **Attribute** integer (0 ... 65535), which is the specific device property that is the target of the explicit message that is used in the construction of the message path.
Service	**Number:** The **Number** is the integer (1 ... 127) associated with the service to be performed by the explicit message. **NOTE:** If you select **Custom Service** as the named service, type in a service number. This field is read-only for all other services.
	Name: Select the service that the explicit message is intended to perform.

Field	Setting
	Enter Path(hex): Check this box to enable the message path field, where you can manually enter the entire path to the target device.
Data(hex)	**Data(hex):** This value represents the data to be sent to the target device for services that send data.
Messaging	**Connected:** Select this radial button to make the connection.
	Unconnected: Select this radial button to end the connection.
Response(hex)	The **Response** area contains the data sent to the configuration tool by the target device in hexadecimal format.
Status	The **Status** area displays messages that indicate whether or not the explicit message has succeeded. (See the *CIP General Status Codes* topic in the *Modicon M580 Hardware Reference Manual*.).
Button	**Send to Device**: When your explicit message is configured, click **Send to Device**.

Click the **Close** button to save the changes and close the window.

Sending Explicit Messages to Modbus Devices

Introduction

Use the Modbus explicit messaging window to send an explicit message from Control Expert to the M580 CPU.

You can use explicit messaging to perform many different services. Not every Modbus TCP device supports every service.

Accessing the Page

Before you can perform explicit messaging, connect the DTM for the M580 CPU to the CPU itself:

Step	Action
1	Open the **DTM Browser** in Control Expert (**Tools > DTM Browser**).
2	Select the M580 DTM in the **DTM Browser**.
3	Right-click the M580 DTM.
4	Scroll to the EtherNet/IP explicit messaging page (**Device menu > Additional functions > Modbus Explicit Message**).

Configuring Settings

Configure the explicit message using these settings on the **Modbus Explicit Messaging** page:

Field	Setting
Address	**IP Address:** The IP address of the target device that is used to identify the target of the explicit message.
	Start Address: This setting is a component of the addressing path.
	Quantity: This setting is a component of the addressing path.
	Read Device Id Code: This read-only code represents the service that the explicit message is intended to perform.
	Object Id: This read-only identifier specifies the object that the explicit message is intended to access.
	Unit Id: This integer represents the device or module that is the target of the connection: • **255:** (default)**:** Use this value to access the M580 CPU itself.

Field	Setting
	• **0 ... 254:** Use these values to identify the device number of the target device behind a Modbus TCP to Modbus gateway.
Service	**Number:** This integer (0 ... 255) represents the service to be performed by the explicit message.
	Name: Select the integer (0 ... 255) that represents the service that the explicit message is intended to perform.
Data	**Data(hex):** This value represents the data to be sent to the target device for services that send data.
Response	The **Response** area displays any data sent to the configuration tool by the target device in hexadecimal format.
Status	The **Status** area displays messages indicating whether or not the explicit message has succeeded.
Button	**Send to Device:** After your explicit message is configured, click **Send to Device**.

Click the **Close** button to save the changes and close the window.

Explicit Messaging Using the MBP_MSTR Block in Quantum RIO Drops

Introduction

This section shows you how to configure both EtherNet/IP and Modbus TCP explicit messages in Quantum RIO drops by including the `MBP_MSTR` function block in the logic of your Control Expert project.

Configuring Explicit Messaging Using MBP_MSTR

Overview

You can use the `MBP_MSTR` function block to configure both Modbus TCP and EtherNet/IP connected and unconnected explicit messages.

The operation begins when the input to the `EN` pin is turned ON. The operation ends if the `ABORT` pin is turned ON, or if the `EN` pin is turned OFF.

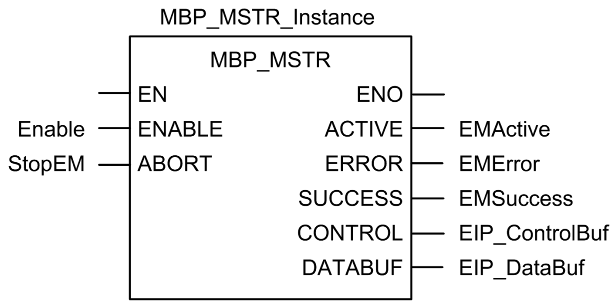
The `CONTROL` and `DATABUF` output parameters define the operation.

> **NOTE:** The structure and content of the `CONTROL` and `DATABUF` output parameters differ for explicit messages configured using the EtherNet/IP and Modbus TCP protocols. Refer to the topics Configuring the Control Parameter for EtherNet/IP and Configuring the Control Parameter for Modbus TCP for instructions on how to configure these parameters for each protocol.

The `ACTIVE` output turns ON during operation; the `ERROR` output turns ON if the operation aborts without success; the `SUCCESS` output turns ON at the successful completion of the operation.

`EN` and `ENO` can be configured as additional parameters.

Representation in FBD



Input Parameters

Parameter	Data type	Description
`ENABLE`	`BOOL`	When ON, the explicit message operation (specified in the first element of the `CONTROL` pin) is executing.
`ABORT`	`BOOL`	When ON, the operation is aborted.

Output Parameters

Parameter	Data type	Description
`ACTIVE`	`BOOL`	ON when the operation is active. OFF at all other times.
`ERROR`	`BOOL`	ON when the operation is aborted without success. OFF before operation, during operation, and if operation succeeds.
`SUCCESS`	`BOOL`	ON when the operation concludes successfully. OFF before operation, during operation, and if operation does not conclude successfully.
`CONTROL`[1]	`WORD`	This parameter contains the control block. The first element contains a code describing the operation to be performed. The content of the control block depends on the operation. The structure of the control block depends on the protocol (EtherNet/IP or Modbus TCP). **Note:** Assign this parameter to a located variable.

Parameter	Data type	Description
DATABUF[1]	WORD	This parameter contains the data buffer. For operations that: • provide data — e.g., a write operation — this parameter is the data source • receive data — e.g., a read operation — this parameter is the data destination **Note:** Assign this parameter to a located variable.
1. Refer to the topics Configuring the Control Block for EtherNet/IP and Configuring the Control Block for Modbus TCP for instructions on how to configure these parameters for the EtherNet/IP and Modbus TCP communication protocols.		

EtherNet/IP Explicit Messaging Services

Overview

Every EtherNet/IP explicit message performs a service. Each service is associated with a service code (or number). You will need to identify the explicit messaging service by its name, decimal number, or hexadecimal number.

You can execute EtherNet/IP explicit messages using either a Control Expert `MBP_MSTR` function block or the Control Expert Ethernet Configuration Tool's **EtherNet/IP Explicit Message Window**.

> **NOTE:** Configuration edits made to an Ethernet communication module from the Control Expert Ethernet Configuration Tool's EtherNet/IP Explicit Message Window are not saved to the operating parameters stored in the CPU and, therefore, are not sent by the CPU to the module on startup.

You can use Control Expert to construct a request that executes any service supported by the target device that is compliant with the EtherNet/IP protocol.

Services

The services supported by Control Expert include the following standard explicit messaging services:

Service Code		Description	Available in...	
Hex	Dec		MBP_MSTR block	Control Expert GUI
1	1	Get_Attributes_All	X	X
2	2	Set_Attributes_All	X	X
3	3	Get_Attribute_List	X	—
4	4	Set_Attribute_List	X	—
5	5	Reset	X	X
6	6	Start	X	X
7	7	Stop	X	X
8	8	Create	X	X
9	9	Delete	X	X
A	10	Multiple_Service_Packet	X	—
D	13	Apply_Attributes	X	X
E	14	Get_Attribute_Single	X	X

Service Code		Description	Available in...	
Hex	Dec		MBP_MSTR block	Control Expert GUI
10	16	Set_Attribute_Single	X	X
11	17	Find_Next_Object_Instance	X	X
14	20	Detected Error Response (DeviceNet only)	—	—
15	21	Restore	X	X
16	22	Save	X	X
17	23	No Operation (NOP)	X	X
18	24	Get_Member	X	X
19	25	Set_Member	X	X
1A	26	Insert_Member	X	X
1B	27	Remove_Member	X	X
1C	28	GroupSync	X	—
"X" = the service is available. "—" = the service is not available.				

Configuring the CONTROL and DATABUF Parameters

Overview

The `CONTROL` and `DATABUF` output parameters define the operation performed by the `MBP_MSTR` function block. For the EtherNet/IP protocol, the structure of the `CONTROL` and `DATABUF` output parameters remains the same for every explicit messaging service, page 346.

Configuring the Control Parameter

The Control parameter consists of 9 contiguous words, as described below:

Register	Function	Description
`CONTROL[0]`	Operation	• 14 = unconnected • 270 = connected
`CONTROL[1]`	Detected error status	Holds the event code (see Modicon M580 Standalone, System Planning Guide for, Frequently Used Architectures) (read-only).
`CONTROL[2]`	Data buffer length	Data buffer length, in words
`CONTROL[3]`	Response offset	Offset for the beginning of the response in the data buffer, in 16-bit words **Note:** To avoid overwriting the request, confirm that the response offset value is greater than the request length `CONTROL[7]`.
`CONTROL[4]`	Slot	High byte = slot location on backplane
		Low byte = 0 (not used)
`CONTROL[5]`[1]	IP address	High byte = byte 4 of the IP address (MSB)
		Low byte = byte 3 of the IP address
`CONTROL[6]`[1]		High byte = byte 2 of the IP address
		Low byte = byte 1 of the IP address (LSB)
`CONTROL[7]`	Request length	Length of the CIP request, in bytes
`CONTROL[8]`	Response length	Length of the response received, in bytes Read only—set after completion
1. For example, the Control parameter handles the IP address 192.168.1.6 in the following order: Byte 4 = 192, Byte 3 = 168, Byte 2 = 1, Byte 1 = 6.		

Configuring the Data Buffer

The data buffer varies in size. It consists of contiguous registers that include—in sequence—both the CIP request and the CIP response. To avoid overwriting the request, confirm that the data buffer is large enough to simultaneously contain both the request and response data.

<table>
<tr><td rowspan="2">Data Buffer:

Variable size: set in <code>CONTROL[2]</code></td><td>CIP Request:

Request size: set in <code>CONTROL[7]</code></td></tr>
<tr><td>CIP Response:

Starting position: set in <code>CONTROL[3]</code>

Response size: reported in <code>CONTROL[8]</code>

NOTE: If the response offset is smaller than the request size, the response data overwrites part of the request.</td></tr>
</table>

The format of the data buffer's CIP request and CIP response is described, below.

NOTE: Structure both the request and response in little endian order.

Request:

Byte offset	Field	Data type	Description
0	Service	Byte	Service of the explicit message
1	Request_Path_Size	Byte	The number of words in the Request_Path field
2	Request_Path	Padded EPATH	This byte array describes the path of the request—including class ID, instance ID, etc.—for this transaction
...	Request_Data	Byte array	Service specific data to be delivered in the explicit message request—if none, this field is empty

Response:

Byte offset	Field	Data type	Description
0	Reply Service	Byte	Service of the explicit message + 16#80
1	Reserved	Byte	0
2	General Status	Byte	EtherNet/IP General Status (see Modicon M340, BMX NOC 0401 Ethernet Communication Module, User Manual)
3	Size of Additional Status	Byte	Additional Status array size—in words

Byte offset	Field	Data type	Description
4	Additional Status	Word array	Additional status[1]
...	Response Data	Byte array	Response data from request, or additional detected error data if General Status indicates a detected error

1. Refer to *The CIP Networks Library, Volume 1, Common Industrial Protocol* at section 3-5.6 *Connection Manager Object Instance Detected Error Codes*;

MBP_MSTR Example: Get_Attributes_Single

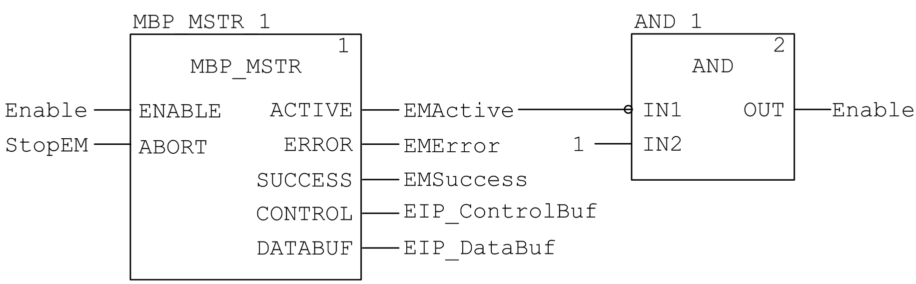
Overview

The following unconnected explicit messaging example shows you how to use the MBP_MSTR function block to retrieve diagnostic information for an STB island from an STB NIC 2212 network interface module, by using the Get_Attributes_Single service.

You can perform the same explicit messaging service using the **EtherNet/IP Explicit Message Window** of the Control Expert Ethernet Configuration Tool (see Quantum EIO, Control Network, Installation and Configuration Guide).

Implementing the MBP_MSTR Function Block

To implement the MBP_MSTR function block, you need to create and assign variables, then connect it to an AND block. In the following example, the logic will continuously send an explicit message upon receiving notice of success:



Input Variables

Variables need to be created and assigned to input pins. For the purpose of this example, variables have been created — and named — as described below. (You can use different variable names in your explicit messaging configurations.)

Input Pin	Variable	Data Type
ENABLE	Enable	BOOL
ABORT	StopEM	BOOL

Output Variables

Variables also need to be created and assigned to output pins. (The names assigned to output variables apply only to this example, and can be changed in your explicit messaging configurations.)

Output Pin	Variable	Data Type
ACTIVE	EMActive	BOOL
ERROR	EMError	BOOL
SUCCESS	EMSuccess	BOOL
CONTROL	EIP_ControlBuf	Array of 10 WORDs
DATABUF	EIP_DataBuf	Array of 100 WORDs

NOTE: To simplify configuration, you can assign the CONTROL and DATABUF output pins to a byte array consisting of located variables. When configured in this manner, you will not need to be aware of the location of data within a word (for example, high versus low byte, and big or little endian format).

Control Array

The control array parameter (EIP_ControlBuf) consists of 9 contiguous words. You need to configure only some control words; other control words are read-only and are written to by the operation. In this example, the control array defines the operation as an unconnected explicit message, and identifies the target device:

Register	Description	Configure	Setting (hex)
CONTROL[0]	Operation: High byte = • 00 (unconnected), or • 01 (connected) Low byte = 0E (CIP explicit message)	Yes	16#000E (unconnected)
CONTROL[1]	Detected error status: read-only (written by operation)	No	16#0000
CONTROL[2]	Data buffer length = 100 words	Yes	16#0064
CONTROL[3]	Response offset: offset — in words — for the beginning of the explicit message response in the databuffer	Yes	16#0004
CONTROL[4]	High byte = slot location of the communication module in the backplane	Yes	16#0400

Register	Description	Configure	Setting (hex)
	Low byte = 0 (not used)		
CONTROL[5][1]	IP address of the Ethernet communication module: High byte = byte 4 of the IP address Low byte = byte 3 of the IP address	Yes	16#C0A8
CONTROL[6][1]	IP address of the Ethernet communication module: High byte = byte 2 of the IP address Low byte = byte 1 of the IP address	Yes	16#0106
CONTROL[7]	CIP request length (in bytes)	Yes	16#0008
CONTROL[8]	Length of received response (written by operation)	No	16#0000
1. In this example, the control parameter handles the IP address 192.168.1.6 in the following order: Byte 4 = 192, Byte 3 = 168, Byte 2 = 1, Byte 1 - 6.			

CIP Request

The CIP request is located at the beginning of the databuffer and is followed by the CIP response. In this example, the CIP request calls for the return of a single attribute value (diagnostic data), and describes the request path through the target device's object structure leading to the target attribute:

Request word	High byte		Low byte	
	Description	Value (hex)	Description	Value (hex)
1	Request path size (in words)	16#03	EM Service: Get_Attributes_Single	16#0E
2	Request path: class assembly object	16#04	Request path: logical class segment	16#20
3	Request path: instance	16#64	Request path: logical instance segment	16#24
4	Request path: attribute	16#03	Request path: logical attribute segment	16#30

Combining the high and low bytes, above, the CIP request would look like this:

Request word	Value
1	16#030E
2	16#0420
3	16#6424
4	16#0330

Viewing the Response

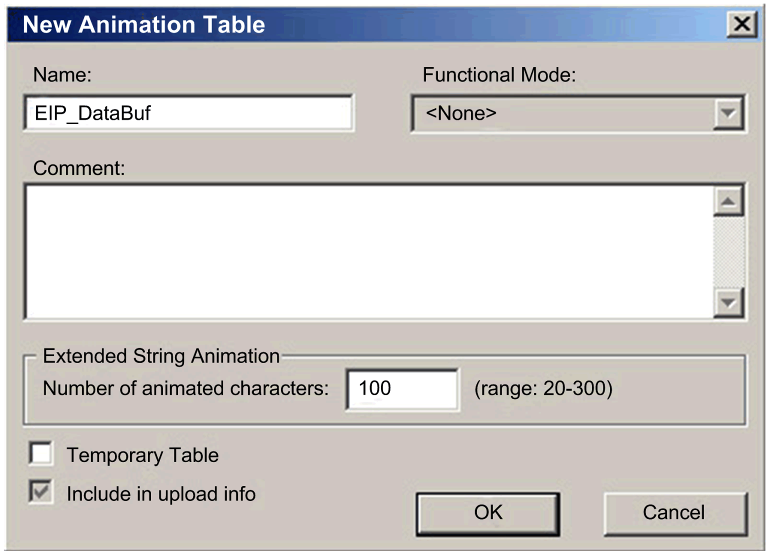
Use a Control Expert Animation table to display the EIP_DataBuf variable array. Note that the EIP_DataBuf variable array consists of the entire data buffer, which includes the:

- CIP request (4 words) located in EIP_DataBuf(1-4)
- CIP service type (1 word) located in EIP_DataBuf(5)
- CIP request status (1 word) located in EIP_DataBuf(6)
- CIP response (in this case, 10 words) located in EIP_DataBuf(7-16)

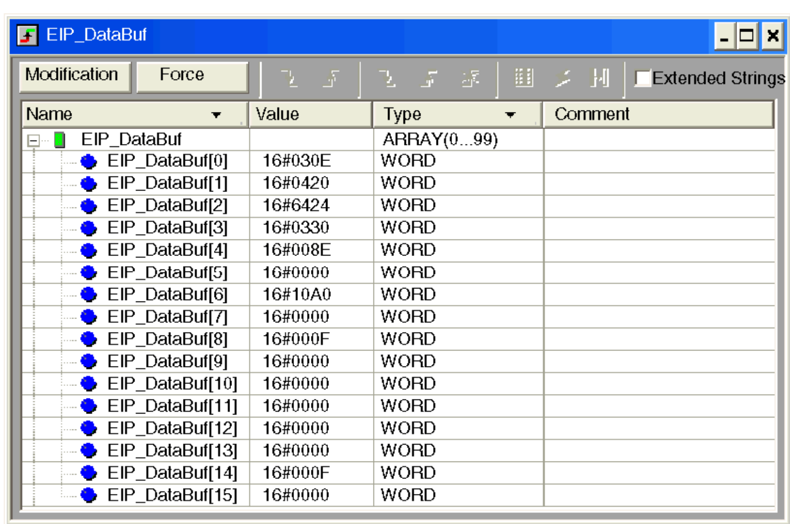
To display the CIP response, follow these steps:

<table>
<tr><th>Step</th><th colspan="2">Action</th></tr>
<tr><td>1</td><td colspan="2">In Control Expert, select Tools → Project Browser to open the Project Browser.</td></tr>
<tr><td>2</td><td colspan="2">In the Project Browser, right-click Animation Tables > New Animation Table.
Result: A new animation table opens.</td></tr>
<tr><td rowspan="5">3</td><td colspan="2">In the New Animation Table dialog, edit the following values:</td></tr>
<tr><td>Name</td><td>Type in a table name. For this example: EIP_DataBuf.</td></tr>
<tr><td>Functional Mode</td><td>Accept the default <None>.</td></tr>
<tr><td>Comment</td><td>Leave blank.</td></tr>
<tr><td>Number of animated characters</td><td>Type 100, representing the size of the data buffer in words.</td></tr>
</table>

<table>
<tr><th>Step</th><th>Action</th></tr>
<tr><td>4</td><td>
The completed dialog looks like this:



Click **OK** to close the dialog.
</td></tr>
<tr><td>5</td><td>In the animation table’s Name column, type in the name of the variable assigned to the databuffer: EIP_DataBuf and press Enter. The animation table displays the EIP_DataBuf variable.</td></tr>
<tr><td>6</td><td>
Expand the EIP_DataBuf variable to display its word array, where you can view the CIP response at words EIP_DataBuf(7-16):



</td></tr>
</table>

Step	Action
	Note: Each word presents 2 bytes of data in little endian format, where the least significant byte is stored in the smallest memory address. For example, ‘0E’ in EIP_DataBuf[0] is the low byte, and ‘03’ is the high byte.

Modbus TCP Explicit Messaging Function Codes

Overview

Every Modbus TCP explicit message performs a function. Each function is associated with a code (or number). You will need to identify the explicit messaging function by its name, decimal number, or hexadecimal number.

You can execute Modbus TCP explicit messages using either a Control Expert `MBP_MSTR` function block or the Control Expert Ethernet Configuration Tool's **Modbus Explicit Message Window**.

NOTE: Configuration edits made to an Ethernet communication module from the Control Expert Ethernet Configuration Tool are not saved to the operating parameters stored in the CPU and, therefore, are not sent by the CPU to the module on startup.

Services

The function codes supported by Control Expert include the following standard explicit messaging functions:

Function Code		Description	Available in...	
Hex	**Dec**		**MBP_MSTR block**	**Control Expert GUI**
1	1	Write data	X	X
2	2	Read data	X	X
3	3	Get local statistics	X	X
4	4	Clear local statistics	X	X
7	7	Get remote statistics	X	X
8	8	Clear remote statistics	X	X
A	10	Reset module	X	X
17	23	Read / write data	X	X
FFF0	65520	Enable / disable HTTPS and FTP/TFTP services	X	-
"X" = the service is available. "—" = the service is not available.				

Configuring the Control Parameter for Modbus TCP Explicit Messaging

Overview

The `CONTROL` and `DATABUF` output parameters define the operation performed by the `MBP_MSTR, page 343` function block. For the Modbus TCP protocol, both the structure and the content of the `CONTROL` output parameter vary, depending upon the function code, page 357.

The structure of the `CONTROL` parameter is described, below, for each supported function code.

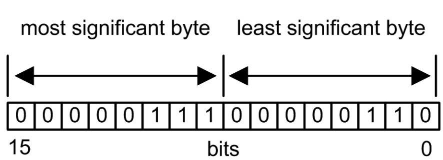
Refer to the *Quantum Ethernet I/O System Planning Guide* for an example of an MSTR block created in a Control Expert application to read the ports of a dual-ring switch (DRS) to diagnose a sub-ring break.

Control Parameter Routing Register

The `CONTROL[5]` routing register specifies the source and destination node addresses for network data transfer, and consists of the following 2 bytes:

- Most Significant Byte (MSB): contains the source node address, for example, the slot number of the 140 NOC 78• 00
- Least Significant Byte (LSB): contains the destination node address — a value representing either a direct or a bridge address. The LSB is required for devices that are reached through a bridge, for example, an Ethernet to Modbus bridge or an Ethernet to Modbus Plus bridge. The values of the LSB are as follows:
  - If no bridge is used: LSB is set to zero(0).
  - If a bridge is used: LSB contains the Modbus Plus on Ethernet Transporter (MET) mapping index value. This value, also known as the Unit ID, indicates the device to which the message is directed.

The `CONTROL[5]` routing register:



When the Ethernet communication module acts as a server, the LSB indicates the destination of a message received by the communication module:

- messages with an LSB value from 0 to 254 are forwarded to and processed by the CPU

- messages with an LSB value of 255 are retained and processed by the Ethernet communication module

NOTE: Unit ID 255 should be used when requesting diagnostic data from the Ethernet communication module.

Write Data

The control parameter consists of 9 contiguous words, as described below:

Register	Function	Description
`CONTROL[1]`	Operation	1 = write data
`CONTROL[2]`	Detected error status	Holds the event code (see Modicon M580 Standalone, System Planning Guide for, Frequently Used Architectures) (read-only)
`CONTROL[3]`	Data buffer length	Number of addresses sent to the slave
`CONTROL[4]`	Starting register	Start address of the slave to which the data is written, in 16-bit words
`CONTROL[5]`	Routing register	High byte = Ethernet communication module slot
		Low byte = MBP on Ethernet transporter (MET) mapping index
`CONTROL[6]`[1]	IP address	Byte 4 of the IP address (MSB)
`CONTROL[7]`[1]		Byte 3 of the IP address
`CONTROL[8]`[1]		Byte 2 of the IP address
`CONTROL[9]`[1]		Byte 1 of the IP address (LSB)
1. For example, the control parameter handles the IP address 192.168.1.7 in the following order: Byte 4 = 192, Byte 3 = 168, Byte 2 = 1, Byte 1 = 7.		

Read Data

The control parameter consists of 9 contiguous words, as described below:

Register	Function	Description
`CONTROL[1]`	Operation	2 = read data
`CONTROL[2]`	Detected error status	Holds the event code (see Modicon M580 Standalone, System Planning Guide for, Frequently Used Architectures) (read-only)
`CONTROL[3]`	Data buffer length	Number of addresses to be read from the slave

Register	Function	Description
CONTROL[4]	Starting register	Determines the %MW starting register in the slave from which the data is read. For example: 1 = %MW1, 49 = %MW49)
CONTROL[5]	Routing register	High byte = Ethernet communication module slot
		Low byte = MBP on Ethernet transporter (MET) mapping index
CONTROL[6][1]	IP address	Byte 4 of the IP address (MSB)
CONTROL[7][1]		Byte 3 of the IP address
CONTROL[8][1]		Byte 2 of the IP address
CONTROL[9][1]		Byte 1 of the IP address (LSB)
1. For example, the control parameter handles the IP address 192.168.1.7 in the following order: Byte 4 = 192, Byte 3 = 168, Byte 2 = 1, Byte 1 = 7.		

Get Local Statistics

The control parameter consists of 9 contiguous words, as described below:

Register	Function	Description
CONTROL[1]	Operation	3 = read local statistics
CONTROL[2]	Detected error status	Holds the event code (see Modicon M580 Standalone, System Planning Guide for, Frequently Used Architectures) (read-only)
CONTROL[3]	Data buffer length	Number of addresses to be read from local statistics (0...37)
CONTROL[4]	Starting register	First address from which the statistics table is read (Reg1= 0)
CONTROL[5]	Routing register	High byte = Ethernet communication module slot
		Low byte = MBP on Ethernet transporter (MET) mapping index
CONTROL[6]	(not used)	—
CONTROL[7]		
CONTROL[8]		
CONTROL[9]		

Module Response: A TCP/IP Ethernet module responds to the `Get Local Statistics` command with the following information:

<table>
<tr><th>Word</th><th colspan="4">Description</th></tr>
<tr><td>00...02</td><td colspan="4">MAC Address</td></tr>
<tr><td rowspan="8">03</td><td colspan="4">Board Status — this word contains the following bits:</td></tr>
<tr><td>Bit 15</td><td>0 = Link LED off; 1 = Link LED ON</td><td>Bit 3</td><td>Reserved</td></tr>
<tr><td>Bits 14...13</td><td>Reserved</td><td>Bit 2</td><td>0 = half duplex; 1 = full duplex</td></tr>
<tr><td>Bit 12</td><td>0 = 10 Mbit; 1 = 100 Mbit</td><td>Bit 1</td><td>0 = not configured; 1 = configured</td></tr>
<tr><td>Bits 11...9</td><td>Reserved</td><td>Bit 0</td><td>0 = PLC not running; 1 = PLC or NOC running</td></tr>
<tr><td rowspan="2">Bits 8...4</td><td colspan="3">Module Type — this bit presents the following values:</td></tr>
<tr><td colspan="2">• 0 = NOE 2x1
• 1 = ENT
• 2 = M1E
• 3 = NOE 771 00
• 4 = ETY
• 5 = CIP
• 6 = (reserved)
• 7 = 140 CPU 651 x0
• 8 = 140 CRP 312 00
• 9 = (reserved)
• 10 = 140 NOE 771 10</td><td>• 11 = 140 NOE 771 01
• 12 = 140 NOE 771 11
• 13 = (reserved)
• 14 = 140 NOC 78• 00
• 15...16 = (reserved)
• 17 = M340 CPU
• 18 = M340 NOE
• 19 = BMX NOC 0401
• 20 = TSX ETC 101
• 21 = 140 NOC 771 01</td></tr>
<tr><td>04 and 05</td><td colspan="4">Number of receiver interrupts</td></tr>
<tr><td>06 and 07</td><td colspan="4">Number of transmitter interrupts</td></tr>
<tr><td>08 and 09</td><td colspan="4">Transmit_timeout detected error count</td></tr>
<tr><td>10 and 11</td><td colspan="4">Collision_detect error count</td></tr>
<tr><td>12 and 13</td><td colspan="4">Missed packets</td></tr>
<tr><td>14 and 15</td><td colspan="4">(reserved)</td></tr>
<tr><td>16 and 17</td><td colspan="4">Number of times driver has restarted</td></tr>
<tr><td>18 and 19</td><td colspan="4">Receive framing detected error</td></tr>
<tr><td>20 and 21</td><td colspan="4">Receiver overflow detected error</td></tr>
<tr><td>22 and 23</td><td colspan="4">Receive CRC detected error</td></tr>
<tr><td>24 and 25</td><td colspan="4">Receive buffer detected error</td></tr>
<tr><td>26 and 27</td><td colspan="4">Transmit buffer detected error</td></tr>
<tr><td>28 and 29</td><td colspan="4">Transmit silo underflow</td></tr>
</table>

Word	Description
30 and 31	Late collision
32 and 33	Lost carrier
34 and 35	Number of retries
36 and 37	IP address

Clear Local Statistics

The control parameter consists of 9 contiguous words, as described below:

Register	Function	Description
`CONTROL[1]`	Operation	4 = clear local statistics
`CONTROL[2]`	Detected error status	Holds the event code (see Modicon M580 Standalone, System Planning Guide for, Frequently Used Architectures) (read-only)
`CONTROL[3]`	(not used)	—
`CONTROL[4]`	(not used)	—
`CONTROL[5]`	Routing register	High byte = Ethernet communication module slot
		Low byte = MBP on Ethernet transporter (MET) mapping index
`CONTROL[6]`	(not used)	—
`CONTROL[7]`		
`CONTROL[8]`		
`CONTROL[9]`		

Get Remote Statistics

The control parameter consists of 9 contiguous words, as described below:

Register	Function	Description
`CONTROL[1]`	Operation	7 = get remote statistics
`CONTROL[2]`	Detected error status	Holds the event code (see Modicon M580 Standalone, System Planning Guide for, Frequently Used Architectures) (read-only)
`CONTROL[3]`	Data buffer length	Number of addresses to be read from the statistics data field (0...37)

Register	Function	Description
CONTROL[4]	Starting register	First address from which the node statistics table is read
CONTROL[5]	Routing register	High byte = Ethernet communication module slot
		Low byte = MBP on Ethernet transporter (MET) mapping index
CONTROL[6][1]	IP address	Byte 4 of the IP address (MSB)
CONTROL[7][1]		Byte 3 of the IP address
CONTROL[8][1]		Byte 2 of the IP address
CONTROL[9][1]		Byte 1 of the IP address (LSB)
1. For example, the control parameter handles the IP address 192.168.1.7 in the following order: Byte 4 = 192, Byte 3 = 168, Byte 2 = 1, Byte 1 = 7.		

Clear Remote Statistics

The control parameter consists of 9 contiguous words, as described below:

Register	Function	Description
CONTROL[1]	Operation	8 = clear remote statistics
CONTROL[2]	Detected error status	Holds the event code (see Modicon M580 Standalone, System Planning Guide for, Frequently Used Architectures) (read-only)
CONTROL[3]	(not used)	—
CONTROL[4]	(not used)	—
CONTROL[5]	Routing register	High byte = Ethernet communication module slot
		Low byte = MBP on Ethernet transporter (MET) mapping index
CONTROL[6][1]	IP address	Byte 4 of the IP address (MSB)
CONTROL[7][1]		Byte 3 of the IP address
CONTROL[8][1]		Byte 2 of the IP address
CONTROL[9][1]		Byte 1 of the IP address (LSB)
1. For example, the control parameter handles the IP address 192.168.1.7 in the following order: Byte 4 = 192, Byte 3 = 168, Byte 2 = 1, Byte 1 = 7.		

Reset Module

The control parameter consists of 9 contiguous words, as described below:

Register	Function	Description
CONTROL[1]	Operation	10 = reset module
CONTROL[2]	Detected error status	Holds the event code (see Modicon M580 Standalone, System Planning Guide for, Frequently Used Architectures) (read-only)
CONTROL[3]	(not used)	—
CONTROL[4]	(not used)	—
CONTROL[5]	Routing register	High byte = Ethernet communication module slot
		Low byte = MBP on Ethernet transporter (MET) mapping index
CONTROL[6]	(not used)	—
CONTROL[7]		
CONTROL[8]		
CONTROL[9]		

Read/Write Data

The control parameter consists of 11 contiguous words, as described below:

Register	Function	Description
CONTROL[1]	Operation	23 = read / write data
CONTROL[2]	Detected error status	Holds the event code (see Modicon M580 Standalone, System Planning Guide for, Frequently Used Architectures) (read-only)
CONTROL[3]	Data buffer length	Number of addresses sent to the slave
CONTROL[4]	Starting register	Determines the %MW starting register in the slave to which the data will be written. For example: 1 = %MW1, 49 = %MW49)
CONTROL[5]	Routing register	High byte = Ethernet communication module slot
		Low byte = MBP on Ethernet transporter (MET) mapping index
CONTROL[6][1]	IP address	Byte 4 of the IP address (MSB)
CONTROL[7][1]		Byte 3 of the IP address

Register	Function	Description
CONTROL[8][1]		Byte 2 of the IP address
CONTROL[9][1]		Byte 1 of the IP address (LSB)
CONTROL[10]	Data buffer length	Number of addresses to be read from the slave
CONTROL[11]	Starting register	Determines the %MW starting register in the slave from which the data is read. For example: 1 = %MW1, 49 = %MW49)
1. For example, the control parameter handles the IP address 192.168.1.7 in the following order: Byte 4 = 192, Byte 3 = 168, Byte 2 = 1, Byte 1 = 7.		

Enable/Disable HTTP or FTP/TFTP Services

When HTTPS or FTP/TFTP has been enabled using Control Expert configuration tools (see Quantum EIO, Control Network, Installation and Configuration Guide), an MSTR block can be used to change the enabled state of the service while the application is running. The MSTR block cannot change the state of the HTTPS or FTP/TFTP services if the service was disabled using one of the configuration tools.

The control parameter consists of 9 contiguous words, as described below:

Register	Function	Description
CONTROL[1]	Operation	FFF0 (hex) 65520 (dec) = enable / disable HTTPS or FTP/TFTP
CONTROL[2]	Detected error status	Holds the event code (read-only). Codes returned include: 0x000 (Success): MSTR block with operational code 0xFFF0 was called and the enabled state of HTTPS or FTP/TFTP was changed. 0x5068 (Busy): MSTR block with operational code 0xFFF0 was called within 2 seconds of the previous call (regardless of return code from previous call). 0x4001 (Same state): MSTR block with operational code 0xFFF0 was called to change the enabled state of HTTPS and FTP/TFTP to the states they were already in. 0x2004 (Invalid data): MSTR block with operational code 0xFFF0 was called and the data in the control block did not match the specifications. 0x5069 (Disabled): If the HTTPS or FTP/TFTP service was already disabled via the Control Expert interface when the MSTR block with operational code 0xFFF0 was called to change the state of the disabled service.
CONTROL[3]		Set this register to 1.

Register	Function	Description
CONTROL[4]		
CONTROL[5]	Module slot number and destination ID	High byte = Module slot number communication module slot
		Low byte = Destination ID
CONTROL[6]	Request mode	Bit 0 (LSB) = 1: Enable FTP/TFTP Bit 0 (LSB) = 0: Disable FTP/TFTP Bit 1 = 1: Enable HTTP Bit 1 = 0: Disable HTTP
CONTROL[7]		Set this register to 0.
CONTROL[8]		
CONTROL[9]		

HTTPS, FTP, and TFTP service state changes made by MSTR with operation code FFF0 (hex) are overridden by the configured value when the module is power-cycled or reset and when a new application is downloaded to the module.

Here are some examples:

State Configured By Control Expert	Action attempted using MSTR with operation code FFF0 (hex)	Result
Disabled	Any	MSTR returns detected error code 0x5069 (service was already disabled by configuration)
Enabled	Disable	MSTR returns code 0x000 (success). • Another MSTR block action enables the service –OR– • The module is reset or power-cycled –OR– • A new application is downloaded with the service disabled by configuration
	Enable	MSTR returns detected error code 0x4001 (same state). No change made.

Implicit Messaging

Introduction

This section extends the sample Control Expert application and contains these instructions:

- Add an STB NIC 2212 EtherNet/IP network interface module to your Control Expert application.
- Configure the STB NIC 2212 module.
- Configure EtherNet/IP connections to link the Ethernet communications module and the STB NIC 2212 network interface module.
- Configure I/O items for the Advantys island.

NOTE: The instructions in this section describe an example of a single, specific device configuration. For other configuration choices, refer to the Control Expert help files.

Setting Up Your Network

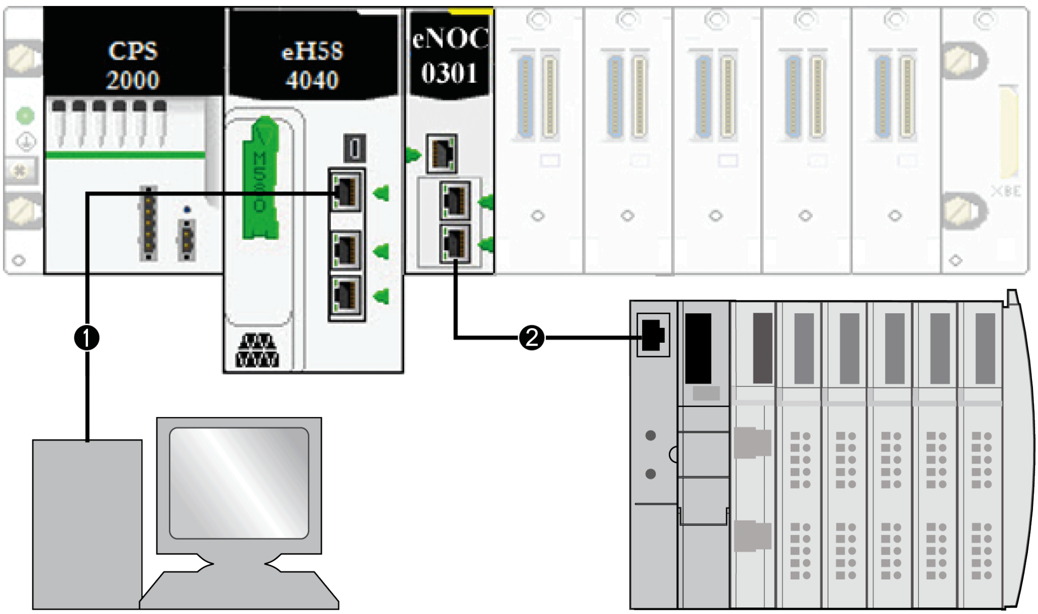
Introduction

Use this example to establish communications between the M580 rack and an Advantys STB NIC 2212 network interface module (NIM).

The STB NIC 2212 is Schneider Electric's EtherNet/IP network interface module for Advantys islands.

Network Topology

This sample network shows the Ethernet network devices used in this configuration:



1 The M580 CPU (with DIO scanner service) on the local rack is connected to a PC that runs the Control Expert software.

2 The BMENOC0301/BMENOC0311 Ethernet communications module on the local rack is connected to an STB NIC 2212 NIM on an Advantys island.

To re-create this example, use the IP addresses from your own configuration for these items:

- M580 CPU
- PC
- BMENOC0301/BMENOC0311 Ethernet communication module
- STB NIC 2212 network interface module

Adding an STB NIC 2212 Device

Overview

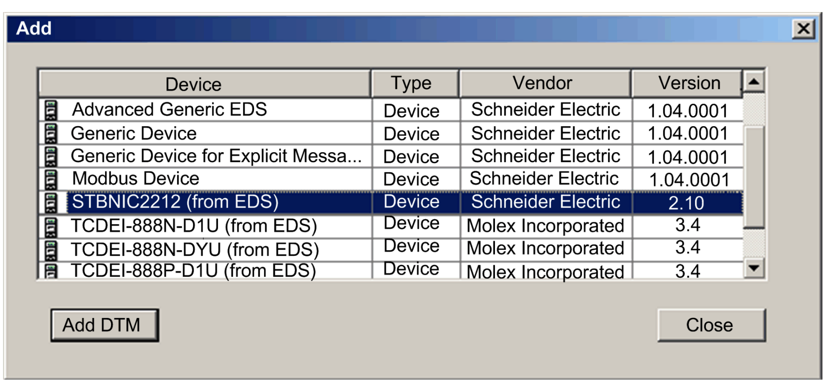
You can use the Control Expert device library to add a remote device—in this example the STB NIC 2212 module—to your project. Only a remote device that is part of your Control Expert device library can be added to your project.

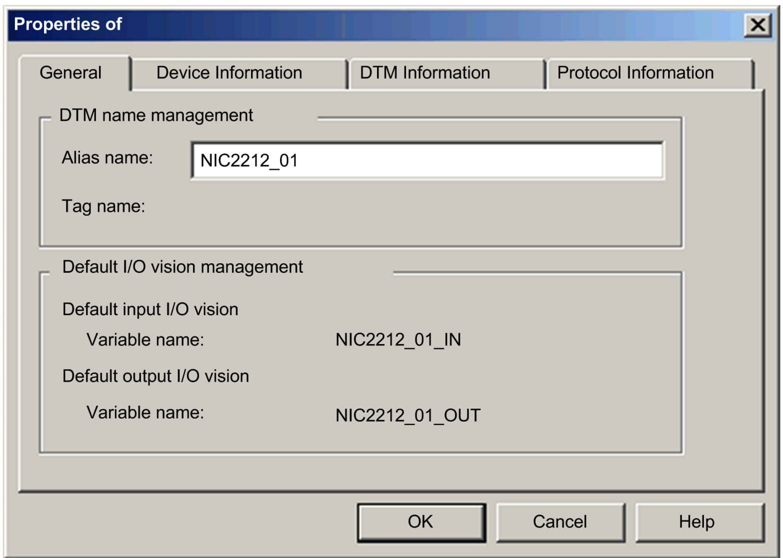
Alternatively, with a remote device already added to your device library, you can use automatic device discovery to populate your project. Perform automatic device discovery by using the **Field bus discovery** command with a communication module selected in the **DTM Browser**.

Adding an STB NIC 2212 Remote Device

NOTE: This example uses a device-specific DTM. If you do not have a device-specific DTM, Control Expert provides a generic device DTM.

Add the STB NIC 2212 to your project:

Step	Action
1	In the **DTM Browser**, right-click the DTM that corresponds to the Ethernet communication module.
2	Scroll to **Add**.
3	Select **STBNIC2212 (from EDS)**:  **NOTE:** Click a column name to sort the list of available devices. (For example, click **Device** to view the items in the first column in alphabetical order.)
4	Click the **Add DTM** button to see the association between the Ethernet communication module and the STB NIC 2212 in the **DTM Browser**.

Step	Action
5	In the **DTM Browser**, right-click the STB NIC 2212 node that is associated with the Ethernet communication module DTM.
6	Scroll to **Properties**.
7	On the **General** tab, create a unique **Alias name**. (Using similar devices that use the same DTM can result in duplicate module names.) In this example, type in the name **NIC2212_01**:  Control Expert uses the **Alias name** as the base for both structure and variable names. **NOTE:** The **Alias name** is the only editable parameter on this tab. The other parameters are read-only.
8	Click **OK** to add the STB NIC 2212 network interface module to the **DTM Browser**, beneath the communication module.

The next step is to configure the device you have just added to the project.

Configuring STB NIC 2212 Properties

Introduction

Use Control Expert to edit the settings for STB NIC 2212 device.

NOTE: To edit these settings, disconnect the DTM from a device.

Accessing the Device Properties

View the **Properties** tab:

Step	Action
1	Double-click the DTM that corresponds to the BMENOC0301/BMENOC0311 module to access the configuration.
2	In the navigation tree, expand the **Device List**, page 285 to see the associated local slave instances.
3	Select the device that corresponds to the name **NIC2212_01**.
4	Select the **Properties** tab.

These configuration tabs are available for the device:

- **Properties**
- **Address Setting**

Properties

Configure the **Properties** tab to perform these tasks:

- Add the STB NIC 2212 to the configuration.
- Remove the STB NIC 2212 from the configuration.
- Edit the base name for variables and data structures used by the STB NIC 2212.
- Indicate how input and output items are created and edited.

The descriptions for parameters (see Modicon M580, BMENOC0301/0311 Ethernet Communications Module, Installation and Configuration Guide) in the **Properties** tab are described in the configuration chapter. Use these values and names from the sample configuration:

Field	Parameter	Description
Properties	**Number**	Accept the default.

Field	Parameter	Description
	Active Configuration	Accept the default (**Enabled**).
IO Structure Name	**Structure Name**	Control Expert automatically assigns a structure name based on the variable name.
	Variable Name	**Variable Name**: Accept the auto-generated variable name (based on the alias name).
	Default Name	Press this button to restore the default variable and structure names. For this example, custom names are used.
Items Management	**Import Mode**	Select **Manual**.
	Reimport Items	Press this buttom to import the I/O items list from the device DTM, overwriting any manual I/O item edits. Enabled only when **Import mode** is set to **Manual**.

Click **Apply** to save your edits and leave the window open.

Address Setting

Use the **Address Setting** tab to enable the DHCP client in the STB NIC 2212 network interface module. When the DHCP client is enabled in the remote device, it obtains its IP address from the DHCP server in the Ethernet communication module.

Configure the **Address Setting** page to perform these tasks:

- Configure the IP address for a device.
- Enable or disable DHCP client software for a device.

The descriptions for parameters in the **Address Setting** tab are described in the configuration chapter. Use these values and names from the sample configuration:

Field	Parameter	Description
Change Address	**IP Address**	In our continuing example, type in the address **192.168.1.6**.
Address Server	**DHCP for this Device**	Select **Enabled**.
	Identified by	Select **Device Name**.
	Identifier	Accept the default setting of the STB NIC 2212 device (based on the **Alias name**).
	Mask	Accept the default value (255.255.0.0).
	Gateway	Configure the default value (192.168.10.1).

The next step is to configure the connection between the communication module and the remote device.

Configuring EtherNet/IP Connections

Overview

An EtherNet/IP connection provides a communication link between 2 or more devices. Properties for a single connection can be configured in the DTMs for the connected devices.

The following example presents settings for a connection between the CPU's DIO scanner service and a remote STB NIC 2212 network interface module. Configuration edits are made to the DTMs for each device.

When making DTM edits, disconnect the selected DTM from the actual module or device (see Modicon M580, BMENOC0301/0311 Ethernet Communications Module, Installation and Configuration Guide).

Accessing the Connection Information

View the connection information tabs:

Step	Action
1	In Control Expert, double-click the DTM for the CPU's DIO scanner service to access the configuration.
2	In the navigation tree, expand the **Device List** (see Modicon M580, BMENOC0301/0311 Ethernet Communications Module, Installation and Configuration Guide) to see the associated local slave instances.
3	Expand (+) the device that corresponds to the STB NIC 2212 module.
4	Select **Read Input/ Write Output Data** to see the **Connection Settings** and **Connection Information** tabs.

Connection Settings

Control Expert automatically creates a connection between a communication module and remote device when the remote device is added to the Control Expert project. Thereafter, many edits to the connection can be made in the DTM for the remote device. However, some of the connection parameters can also be configured in the DTM for the communication module, as demonstrated below.

Edit these parameters on the **Connection Settings** tab. Use settings that are appropriate to your application:

Parameter	Description
Connection Bit	The (read-only) offset for both the health bit and the control bit for this connection. Offset values are auto-generated by the Control Expert DTM.
Request Packet Interval (RPI)	The refresh period for this connection , from 2 to 65535 ms. Default = 12 ms. Type **30** ms. **NOTE:** This parameter can be set in the DTM for the communication module or the remote device.
Time-out Multiplier	This setting, multiplied against the RPI, produces a value that triggers an inactivity timeout. Setting selections include: x4, x8, x16, x32, x64, x128, x256 and x512. For this example, accept the default (**x4**).
Input Fallback Mode	This parameter describes the behavior of inputs in the application in the event communication is lost. Select **Set to Zero**.

Click **OK** to save your settings.

NOTE: The connection information page is read-only when the DTM is selected. This information needs to be set in the DTM for the remote device.

Configuring Connection Settings in the Remote Device DTM

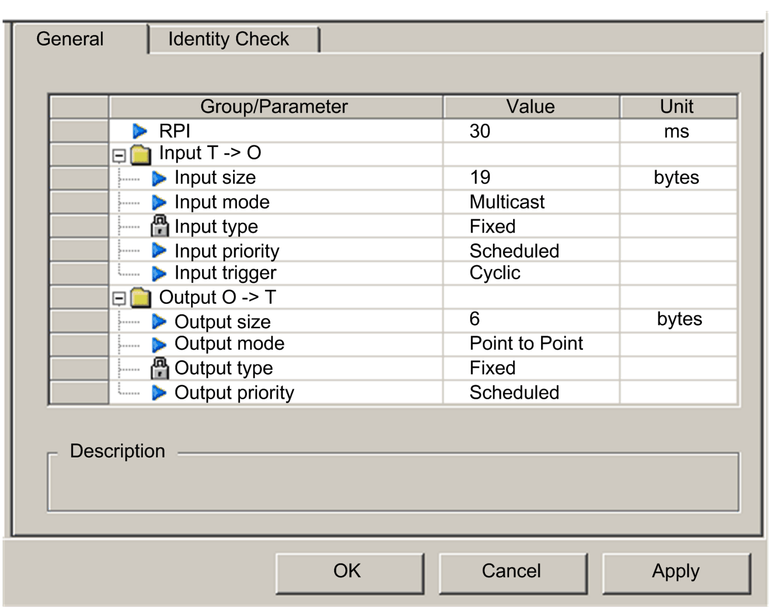
Connections between the CPU's DIO scanner service and a remote device can be created and edited in the DTM for the remote device.

In this example, the following configuration edits are made to the connection that Control Expert automatically created when the remote device was added to the project. Use settings that are appropriate for your actual application:

Step	Action
1	Open the DTM for the remote device by selecting it in the **Device Editor**.
2	Open the **Device Editor**: • Use the main menu (**Edit > Open**) ... *or* ... • Right-click and scroll to **Open**.
3	In the navigation pane (on the left side of the **Device Editor**), confirm that the remote device connection is of the type **Read Input / Write Output Data**. To view the connection type, select the STB NIC 2212 module in the left pane of the **Device Editor**. If the connection type is not of the type **Read Input / Write Output Data**, delete the existing connection and add a new one, as follows: 1. With the connection selected in the left pane, click the **Remove Connection** button. **Result**:The existing connection is removed. 2. Click the **Add Connection** button. **Result**:The **Select the connection to add** dialog opens. 3. Use the scroll buttons on the drop down list to display and select the **Read Input / Write Output Data** connection type. 4. Click **OK** to close the **Select the connection to add** dialog. **Result**:The new connection node appears. 5. Click **Apply** to save the new connection, leaving the **Device Editor** open for additional edits.

General Tab

This is the **General** tab of the DTM for the STB NIC 2212:



Edit the settings in the **General** tab:

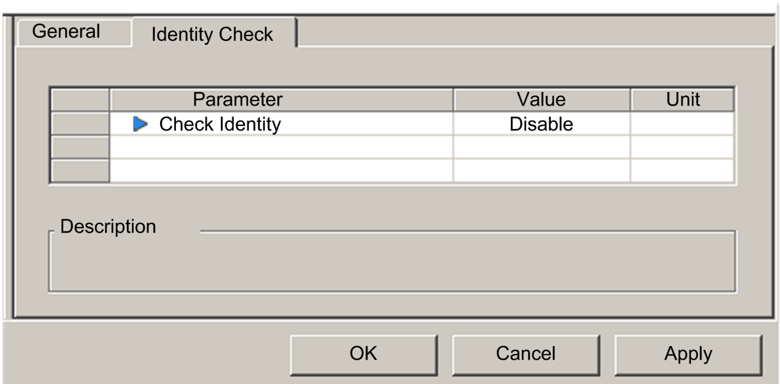
Parameter	Description
RPI	The refresh period for this connection. Accept the value of **30** ms. (This parameter can be set in the DTM for the communication module or the remote device.)
Input size	The number of bytes (0 ... 509) configured in the STB NIC 2212 module.
Input mode	Transmission type: • Multicast • Point to Point For this example, accept the default (Multicast).
Input type	Ethernet packet type (fixed or variable length) to be transmitted. (Only **Fixed** length packets are supported.)
Input priority	The transmission priority value depends upon the device DTM. These are the available values: • Low • High • Scheduled For this example, accept the default selection (Scheduled). **NOTE:** For remote modules that support more than one priority value, you can use this setting to specify the order in which the Ethernet communication module handles packets. For more information, refer to the topic describing QoS packet prioritization (see Modicon M580, BMENOC0301/0311 Ethernet Communications Module, Installation and Configuration Guide).
Input trigger	These are the available transmission trigger values: • Cyclic • Change of state or application For input I/O data, select **Cyclic**.
Output size	The number of bytes configured in the STB NIC 2212 module in increments of 4 bytes (2 words).
Output mode	Accept the default (**Point to Point**).
Output type	(Read-only). Only **Fixed** length packets are supported.
Output priority	Accept the default (**Scheduled**).

Click **Apply** to save your settings and leave the window open.

Identity Check Tab

Configure the **Identity Check** page to set rules for comparing the identity of the network devices (as defined by their DTM or EDS files) against the identity of the actual network device.

This is the **Identity Check** tab:



Use the **Check Identity** parameter to set the rules that the CPU’s DIO scanner service uses to compare the configured versus the actual remote device:

- **Must match exactly**: The DTM or EDS file exactly matches the remote device.
- **Disable**: No checking occurs. The identity portion of the connection is filled with zero values (the default setting).
- **Must be compatible**: If the remote device is not the same as defined by the DTM/EDS, it emulates the DTM/EDS definitions.
- **None**: No checking occurs. The identity portion of the connection is omitted.
- **Custom**: Enable the following parameter settings, to be set individually.

Edit the settings in the **Identity Check** tab:

Parameter	Description
Compatibility Mode	**True**: For each of the following selected tests, the DTM/EDS and remote device need only be compatible.
	False: For each of the following selected tests, the DTM/EDS and remote device need to match exactly.
Compatibility Mode	Make a selection for each of these parameters: • **Compatible**: Include the parameter in the test. • **Not checked**: The parameter is not included in the test.
Minor Version	
Major Version	
Product Code	
Product Type	
Product Vendor	

Click **OK** to save your settings and close the window.

The next step is to configure I/O settings.

Configuring I/O Items

Overview

The final task in this example is to add I/O items to the configuration of the STB NIC 2212 and its eitht I/O modules:

- Use the Advantys configuration software to identify the relative position of each I/O module's inputs and outputs.
- Use the Control Expert **Device Editor** to create input and output items, defining each item's:
  - name
  - data type

I/O Item Types and Sizes

The goal is to create a collection of input items and output items that equal the input size and output size specified for the STB NIC 2212. In this example, items need to be created for:

- 19 bytes of inputs
- 6 bytes of outputs

The Control Expert **Device Editor** provides great flexibility in creating input and output items. You can create input and output items in groups of 1 or more single bits, 8-bit bytes, 16-bit words, 32-bit dwords, or 32-bit IEEE floating values. The number of items you create depends upon the data type and size of each item.

In the sample project, the following items were created:

- discrete bits for digital inputs and outputs
- 8-bit bytes or 16-bit words for analog inputs and outputs

Mapping Input and Output Items

Use the **Fieldbus Image** page of the **I/O Image Overview** window in the Advantys configuration software to identify the number and type of I/O items you need to create, as follows:

Step	Action
1	In the Advantys configuration software, select **Island → I/O Image Overview**. The **I/O Image** window opens to the **Fieldbus Image** page.
2	Select the first cell (word 1, cell 0) in the **Input Data** table to display (in the middle of the page) a description of the cell data and its source module.
3	Make a note of the word, bit(s), module and item information for that cell.
4	Repeat steps 2 and 3 for each cell containing either an S or an integer.

NOTE: The Fieldbus Image presents input and output data in the form of 16-bit words (starting with word 1). You need to rearrange this data for the Control Expert Ethernet Configuration Tool, which presents the same data in the form of 8-bit bytes (starting with byte 0).

NOTE: When you create items, align items of data type `WORD` and `DWORD`:

- `WORD` items: align these items on a 16-bit boundary
- `DWORD` items: align these items on a 32-bit boundary.

This process yields the following tables of input and output data:

Input Data:

Advantys Fieldbus Image		Control Expert EIP Items		STB Module	Description
Word	**Bit(s)**	**Byte**	**Bit(s)**		
1	0-15	0	0-7	NIC 2212	low byte status
		1	0-7		high byte status
2	0-1	2	0-1	DDI 3230	input data
	2-3		2-3	DDI 3230	input status
	4-5		4-5	DDO 3200	output data echo
	6-7		6-7	DDO 3200	output status
	8-11	3	0-3	DDI 3420	input data
	12-15		4-7	DDI 3420	input status
3	0-3	4	0-3	DDO 3410	output data echo
	4-7		4-7	DDO 3410	output status
	8-13	5	0-5	DDI 3610	input data
	14-15		6-7	NA	not used
4	0-5	6	0-5	DDI 3610	input status
	6-7		6-7	NA	not used

Advantys Fieldbus Image		Control Expert EIP Items		STB Module	Description
Word	Bit(s)	Byte	Bit(s)		
	8-13	7	0-5	DDO 3600	output data echo
	14-15		6-7	NA	not used
5	0-5	8	0-5	DDO 3600	output status
	6-15	8	6-7	NA	not used
		9	0-7		
6	0-15	10	0-7	AVI 1270	input data ch 1
		11	0-7		
7	0-7	12	0-7	AVI 1270	input status ch 1
	8-15	13	0-7	NA	not used
8	0-15	14	0-7	AVI 1270	input data ch 2
		15	0-7		
9	0-7	16	0-7	AVI 1270	input status ch 2
	8-15	17	0-7	AVO 1250	output status ch 1
10	0-7	18	0-7	AVO 1250	output status ch 2
	8-15	NA	NA	NA	not used

Output Data:

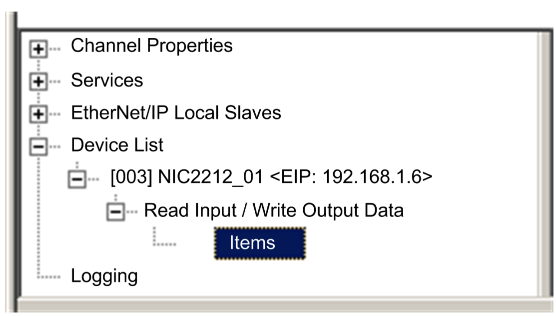
Advantys Fieldbus Image		Control Expert EIP Items		Module	Description
Word	Bit(s)	Byte	Bit(s)		
1	0-1	0	0-1	DDO 3200	output data
	2-5		2-5	DDO 3410	output data
	6-7		6-7	NA	not used
	8-13	1	0-5	DDO 3600	output data
	14-15		6-7	NA	not used
2	0-15	2	0-7	AVO 1250	output data ch 1
		3	0-7		
3	0-15	4	0-7	AVO 1250	output data ch 2
		5	0-7		

This example shows you how to create 19 bytes of inputs and 6 bytes of outputs. To efficiently use space, this example creates items in the following sequence:

- input bit items
- input byte and word items
- output bit items
- output byte and word items

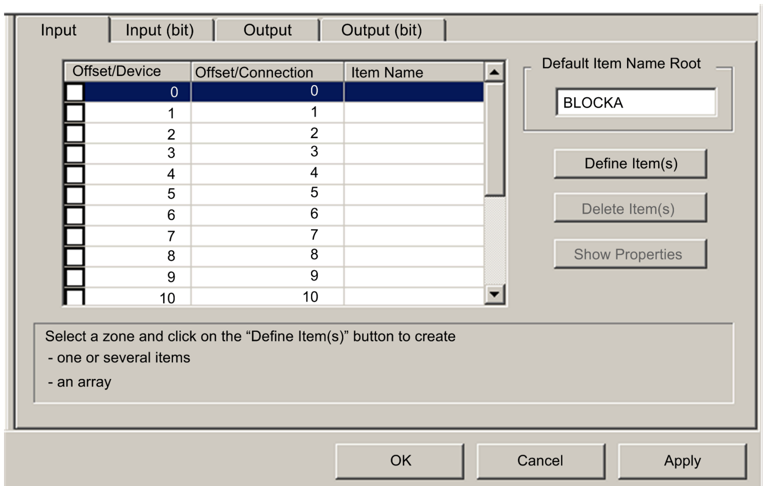
Creating Input Bit Items

To create input bit items for the STB NIC 2212 example, beginning with 16 discrete inputs for NIC 2212 status:

Step	Action
1	In the **DTM Browser**, select the DTM for the BMENOC0301/BMENOC0311.
2	Do one of the following: • in the main menu, select **Edit > Open**. — or — • Right-click and select **Open** in the pop-up menu. **Result**: The **Device Editor** opens, displaying the controller DTM.
3	In the left pane of the **Device Editor**, navigate to and select the **Items** node for the STB NIC 2212 network interface module: 

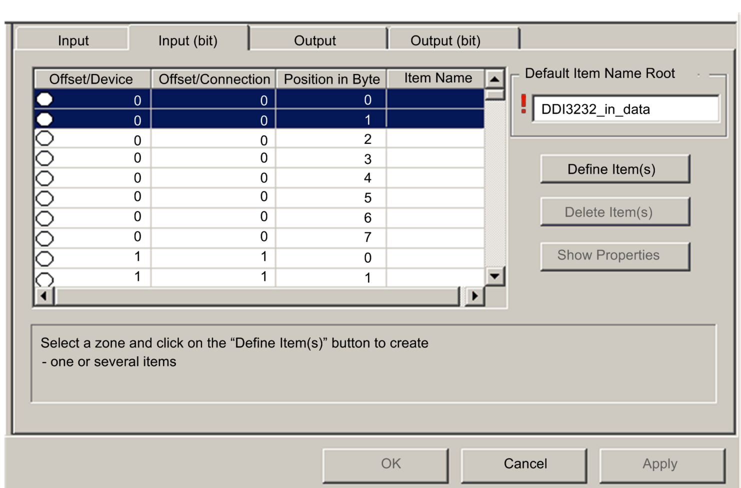
<table>
<tr><th>Step</th><th>Action</th></tr>
<tr><td>4</td><td>

The **Items** window opens:



</td></tr>
<tr><td>5</td><td>Select the Input (bit) tab to display that page.</td></tr>
<tr><td>6</td><td>In the Input (bit) page, type the following default root name (representing device status) into the Default Items Name Root input box type: DDI3232_in_data.</td></tr>
<tr><td>7</td><td>

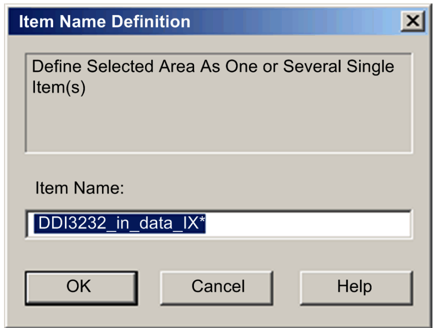
In the **Items List**, select the first 2 rows in the table. (These rows represent bits 0-1 in byte.)



</td></tr>
<tr><td>8</td><td>Click the Define Item(s) button.</td></tr>
</table>

<table>
<tr><th>Step</th><th>Action</th></tr>
<tr><td></td><td>

Result: The **Item Name Definition** dialog opens:

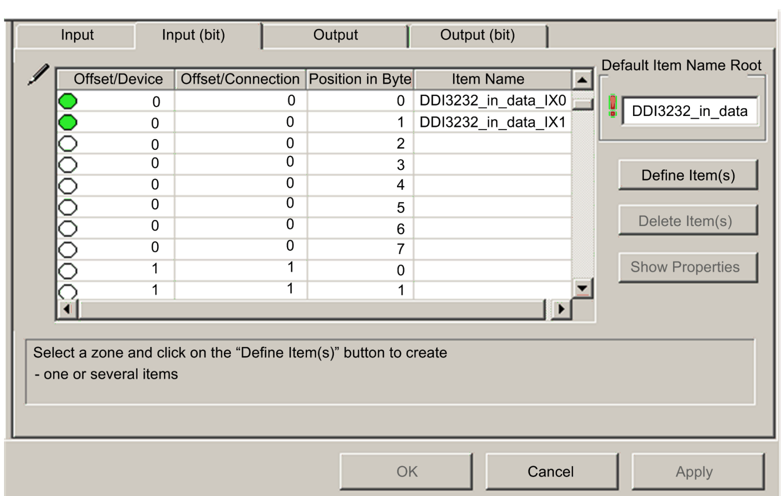


NOTE: The asterisk (*) indicates that a series of discrete items with the same root name will be created.

</td></tr>
<tr><td>9</td><td>

Accept the default **Item Name**, and click **OK**.

Result: 2 discrete input items are created:



</td></tr>
<tr><td>10</td><td>

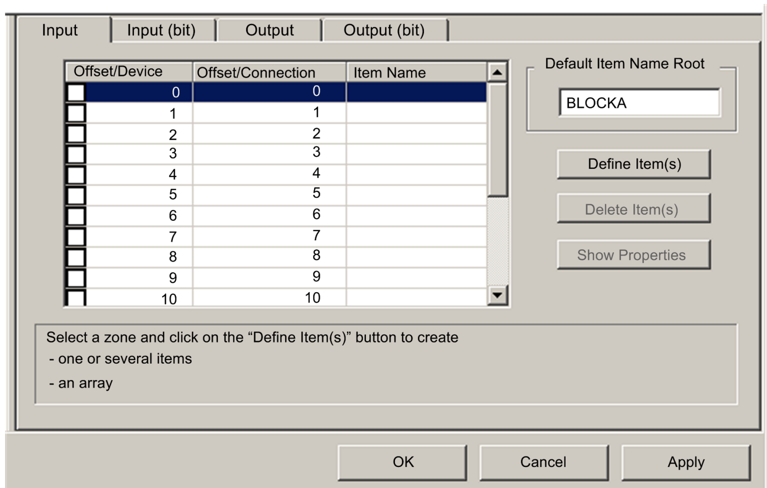
Click **Apply** to save the items and leave the page open.

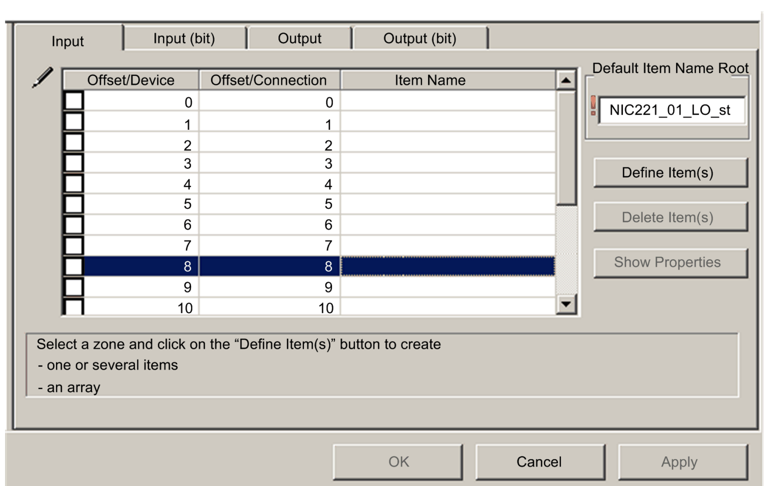
</td></tr>
</table>

Step	Action
11	Repeat steps 6 - 10 for each group of discrete input items you need to create. In this example, that includes items for each of the following groups: • Byte: 0, Bits: 2-3, Default Items Name Root: DDI3230_in_st • Byte: 0, Bits: 4-5, Default Items Name Root: DDO3200_out_echo • Byte: 0, Bits: 6-7, Default Items Name Root: DDO3200_out_st • Byte: 1, Bits: 0-3, Default Items Name Root: DDI3420_in_data • Byte: 1, Bits: 4-7, Default Items Name Root: DDI3420_in_st • Byte: 2, Bits: 0-3, Default Items Name Root: DDO3410_out_echo • Byte: 2, Bits: 4-7, Default Items Name Root: DDO3410_out_st • Byte: 3, Bits: 0-5, Default Items Name Root: DDI3610_in_data • Byte: 4, Bits: 0-5, Default Items Name Root: DDI3610_in_st • Byte: 5, Bits: 0-5, Default Items Name Root: DDO3600_out_echo • Byte: 6, Bits: 0-5, Default Items Name Root: DDO3600_out_st
12	The next task is to create input bytes and words.

Creating Input Items

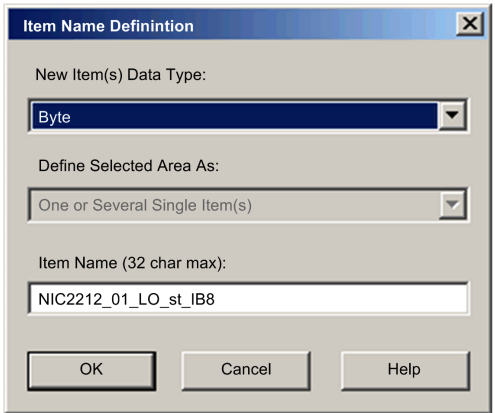
To create input items for the STB NIC 2212 example, begin with an input data byte containing low byte status for the STB NIC 2212 module:

<table>
<tr><th>Step</th><th>Action</th></tr>
<tr><td>1</td><td>Select the Input tab to return to that page:


NOTE: In this example, both the Offset/Device and Offset/Connection columns represent the byte address. The items you create will be either an 8-bit byte or a 16-bit word</td></tr>
<tr><td>2</td><td>In the Default Item Name Root input box type: NIC22212_01_LO_st.</td></tr>
<tr><td>3</td><td>Starting at the first available whole input word, select the single row at byte 8:

</td></tr>
<tr><td>4</td><td>Click the Define Item(s) button.</td></tr>
</table>

<table>
<tr><th>Step</th><th>Action</th></tr>
<tr><td></td><td>

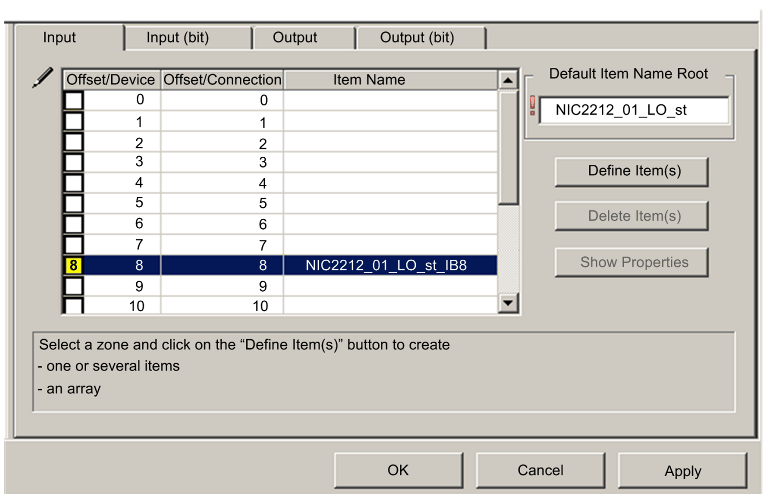
Result: The **Item Name Definition** dialog opens:



</td></tr>
<tr><td>5</td><td>

Select **Byte** as the **New Item(s) Data Type**, then click **OK**.

Result: A new byte item is created:



</td></tr>
<tr><td>6</td><td>Click Apply to save the new items and leave the page open.</td></tr>
</table>

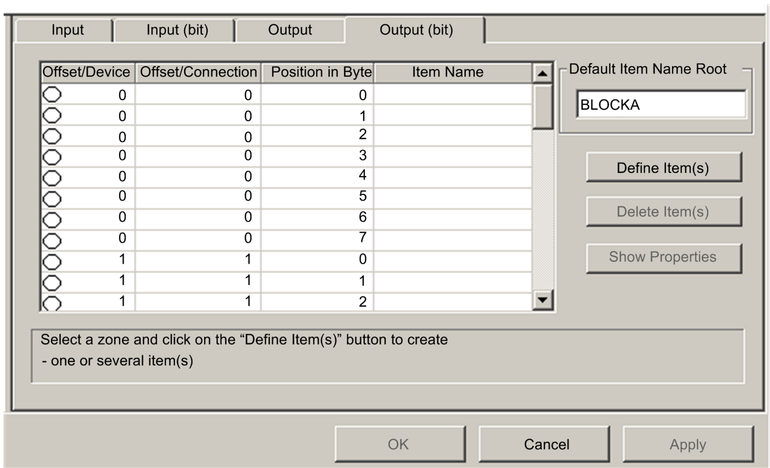
Step	Action
7	Repeat steps 2 - 6 for each byte or word input item you need to create. **NOTE:** The number of rows you select for a new item depends upon the item type. If the item is a: • byte: select a single row • word: select two rows, beginning at the next available whole word In this example, you will create items for each of the following: • Byte: 9, Default Items Name Root: NIC2212_01_HI_st • Word: 10, Default Items Name Root: AVI1270_CH1_in_data • Byte: 12, Default Items Name Root: AVI1270_CH1_in_st • Word: 14-15, Default Items Name Root: AVI1270_CH2_in_data • Byte: 16, Default Items Name Root: AVI1270_CH2_in_st • Byte: 17, Default Items Name Root: AVO1250_CH1_out_st • Byte: 18, Default Items Name Root: AVO1250_CH2_out_st
8	The next task is to create output bits.

Creating Output Bit Items

To create output bit items for the STB NIC 2212 example, beginning with 2 output bits for the STB DDO3200 module:

<table>
<tr><th>Step</th><th>Action</th></tr>
<tr><td>1</td><td>

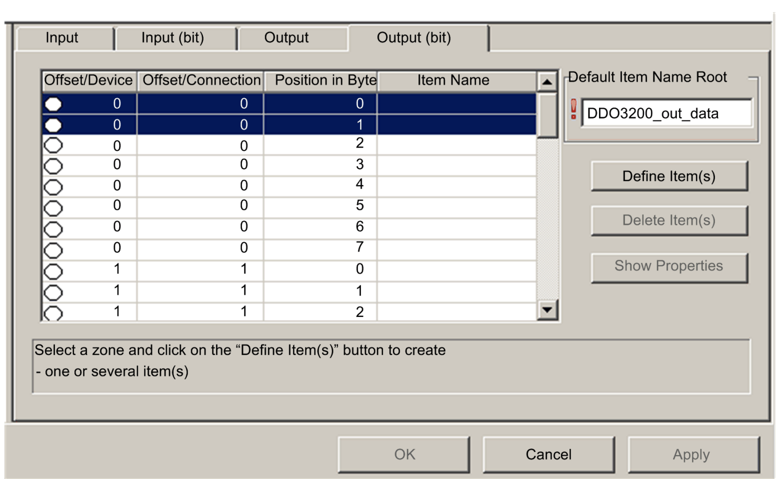
Select the **Output (bit)** tab to open the following page:



NOTE: Both the **Offset/Device** and **Offset/Connection** columns represent the byte address of an output, while the **Position in Byte** column indicates the bit position (within the byte) of each discrete output item.

</td></tr>
<tr><td>2</td><td>In the Default Items Name Root input box type: DDO3200_out_data.</td></tr>
<tr><td>3</td><td>

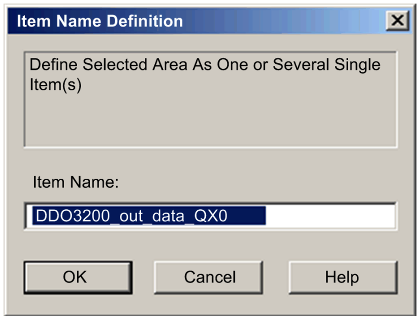
In the **Items List**, select the rows that correspond to bits 0-1 in byte 0—i.e., the first 2 rows:



</td></tr>
<tr><td>4</td><td>Click the Define Item(s) button.</td></tr>
</table>

<table>
<tr><th>Step</th><th>Action</th></tr>
<tr><td></td><td>

Result: The **Item Name Definition** dialog opens:

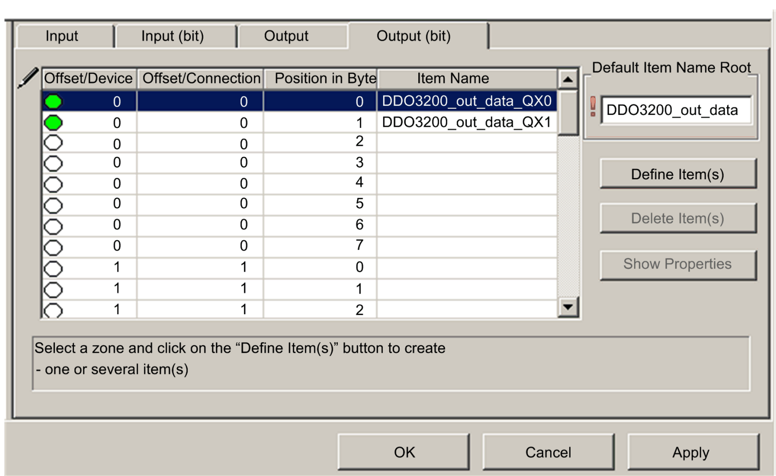


NOTE: The asterisk (*) indicates that a series of discrete items with the same root name will be created.

</td></tr>
<tr><td>5</td><td>

Accept the default output name and click **OK**.

Result: 2 discrete output items are created:



</td></tr>
<tr><td>6</td><td>Click Apply to save the new items and leave the page open.</td></tr>
<tr><td>7</td><td>

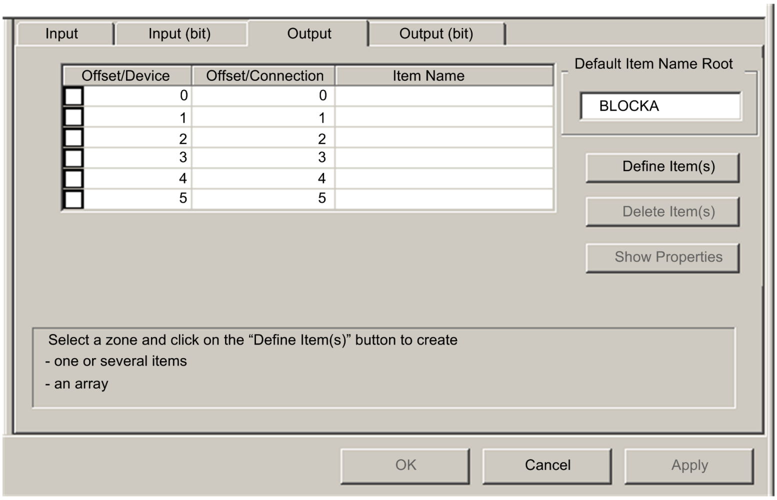
Repeat steps 2 - 6 for each group of discrete output items you need to create. In this example, that includes items for each of the following groups:

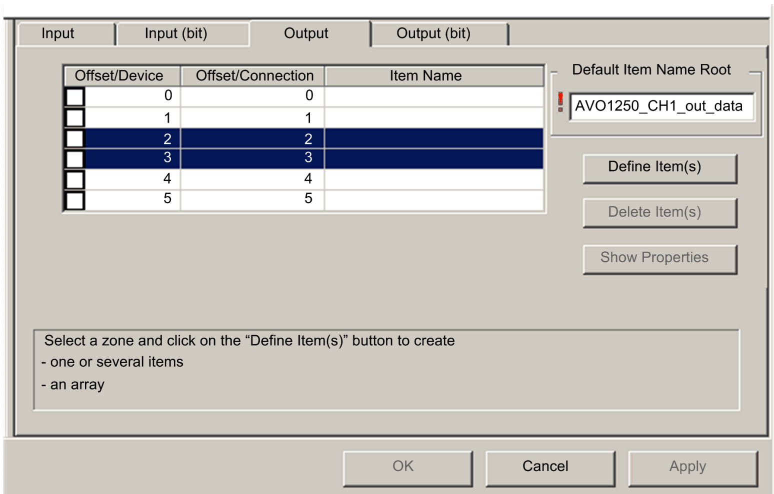
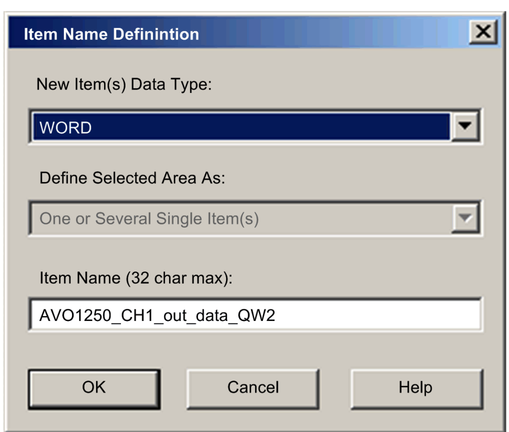
- Byte: 0, Bits: 2-5, Default Items Name Root: DDO3410_out_data
- Byte: 1, Bits: 0-5, Default Items Name Root: DDO3600_out_data

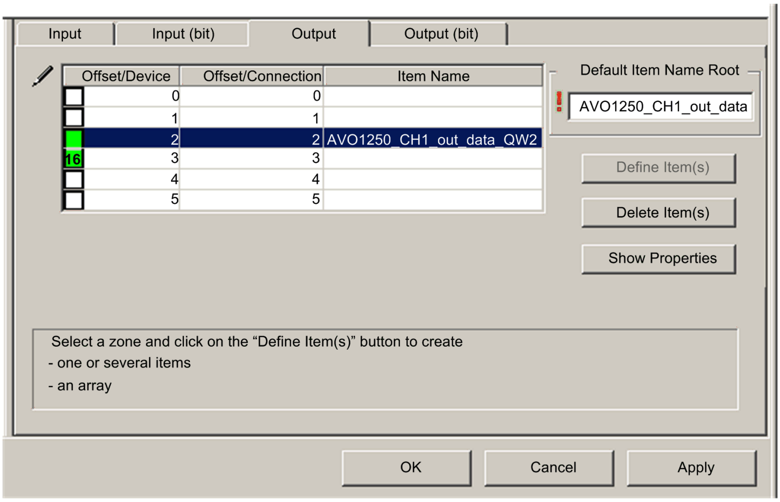
</td></tr>
<tr><td>8</td><td>The next task is to create output bytes and words.</td></tr>
</table>

Creating Numeric Output Items

To create output items for the STB NIC 2212, example, beginning with an output data word for the STB AVO 1250 module:

Step	Action
1	Click on the **Output** tab to open the following page:  **NOTE:** In this example, both the **Offset/Device** and **Offset/Connection** columns represent the byte address. The items you create will be 16-bit words comprising 2 bytes.
2	In the **Default Item Name Root** input box type: **AVO1250_CH1_out_data**.

<table>
<tr><th>Step</th><th>Action</th></tr>
<tr><td>3</td><td>Starting at the next available whole word, select 2 rows: 2 and 3:

</td></tr>
<tr><td>4</td><td>Click the Define Item(s) button.
Result: The Item Name Definition dialog opens:

</td></tr>
<tr><td>5</td><td>Accept the default output name and click OK.</td></tr>
</table>

Step	Action
	Result: The following output word item is created: 
6	Click **Apply** to save the new item and leave the page open.
7	Repeat steps 2 - 6 for the AVO 1250 channel 2 output data at bytes 4 and 5.
8	Click **OK** to close the **Items** window.
9	Select **File > Save** to save your edits.

EtherNet/IP Implicit Messaging

Overview

As a best practice, the RPI for EtherNet/IP implicit message connections are 1/2 of MAST cycle time. If the resulting RPI is less than 25 ms, the implicit message connections may be adversely affected when the diagnostic features of the controller Ethernet I/O scanner service are accessed through explicit messages or the DTM.

In this situation, use these timeout multiplier, page 314 settings:

RPI (ms)	Timeout Multiplier	Connection Timeout (ms)
2	64	128
5	32	160
10	16	160
20	8	160
25	4	100

NOTE: If you use values that are lower than those in the table, the network can consume unnecessary bandwidth, which can affect the performance of the module within the system.

Configuring the M580 CPU as an EtherNet/IP Adapter

Introduction

This section describes the configuration of an M580 CPU as an EtherNet/IP adapter using *local slave* functionality.

Introducing the Adapter

Introduction

The embedded Ethernet I/O scanner service in the M580 PAC scans network modules.

However, you can enable the controller scanner service as an EtherNet/IP adapter. When the adapter functionality is enabled, network scanners can access controller data that is mapped to adapter assembly objects in the controller program.

NOTE:

- The controller scanner service continues to function as a scanner when it is enabled as an EtherNet/IP adapter.
- To get data from the primary controller, make the connection to the Main IP address of the Controller (see Modicon M580 Hot Standby, System Planning Guide for Frequently Used Architectures).

The controller scanner service supports up to 16 instances of adapters (adapter 1 ... adapter 3). Each enabled adapter instance supports these connections:

- one exclusive owner connection
- one listen-only connection

Process Overview

These are the steps in the adapter configuration process:

Stage	Description
1	Enable and configure the controller scanner service as an adapter.
2	Configure adapter instances in the scanner service. (Adapter instances correspond to each enabled adapter that is scanned.)
3	Specify the size of adapter input and output assemblies in the scanner service. (Use sizes that match the input and output sizes of the enabled adapter, page 126.)

Implicit and Explicit Messaging

In its role as an EtherNet/IP adapter, the controller scanner services responds to these requests from network scanners:

- **implicit messages:** Implicit messaging requests are sent from a network scanner device to the controller. When the adapter functionality is enabled, network scanners can perform these tasks:
  - read messages from the controller scanner service
  - write messages to the controller scanner service

  Implicit messaging is especially suited to the exchange of peer-to-peer data at a repetitive rate.
- **explicit messages:** The controller scanner service responds to explicit messaging requests that are directed to CIP objects. When adapters are enabled by the controller, explicit messaging requests can access the controller scanner service CIP assembly instances. (This is a read-only function.)

Third-Party Devices

If the controller scanner service that communicates with the adapter can be configured using Control Expert, use DTMs that correspond to the controller to add those modules to your configuration.

Third-party EtherNet/IP scanners that access the adapter assembly instances through the controller's scanner service do so with respect to the assembly mapping table. The controller scanner service is delivered with its corresponding EDS file. Third-party scanners can use the contents of the EDS file to map inputs and outputs to the appropriate assembly instances of the controller scanner service.

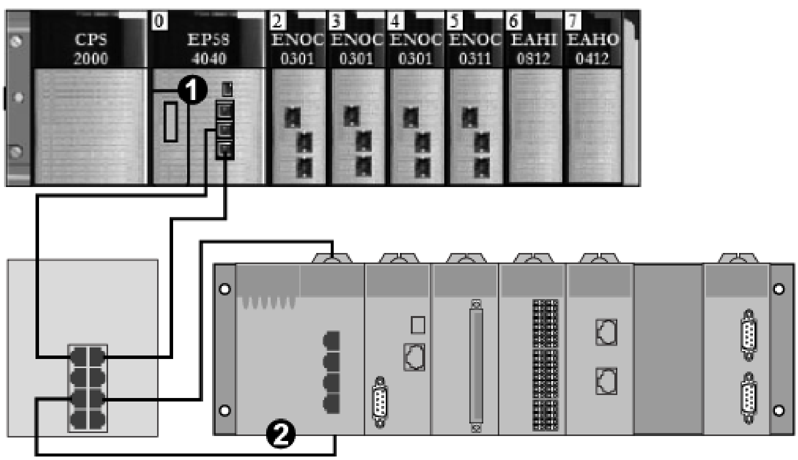
Local Slave Configuration Example

Introduction

Use these instructions to create a simple local slave configuration that includes a network scanner (originator, **O**) and an M580 CPU that is enabled as a local slave (target, **T**).

Originator and Target Devices

This figure, which is a subset of the sample network, shows the enabled local slave (1) and the master device (2):



1 M580 CPU: The CPU on the M580 local rack. In this example, you will enable this CPU’s embedded scanner service as a local slave device (or target, **T**).

2 Modicon M340 rack: In this example, the scanner (or originator, **O**) on this rack scans the CPU data on the M580 rack through the enabled local slave (M580 CPU’s scanner service).

Enabling Local Slaves

Introduction

In a sample configuration, you will enable **Local Slave 1** and **Local Slave 2**.

First, use these instructions to enable **Local Slave 1** in the CPU's embedded scanner service configuration. At the end of this exercise, repeat these instructions to enable **Local Slave 2**.

Enabling a Local Slave

Enable the CPU in the M580 local rack as a target device (local slave):

Step	Action
1	Open your M580 Control Expert project.
2	On the **General** tab, assign this **Alias name** to the CPU: BMEP58_ECPU_EXT.
3	In the **DTM Browser** (**Tools > DTM Browser**), double-click the DTM that corresponds to the alias name of the BMENOC0301.2 module to open the configuration window.
4	In the navigation pane, expand (+) **EtherNet/IP Local Slaves** to see the 3 available local slaves.
5	Select a local slave to see its properties. (For this example, select **Local Slave 1**.)
6	In the drop-down list (**Properties > Active Configuration**), scroll to **Enabled**.
7	Click **Apply** to enable **Local Slave 1**.
8	Click **OK** to apply the changes and close the configuration window.

You now have enabled **Local Slave 1** for the CPU's scanner service at IP address 192.168.20.10.

EtherNet/IP scanners that scan the network for the CPU's scanner service at that IP address can use implicit messages to read from and write to the assembly instances that are associated with the local slave instance.

Enabling Another Local Slave

This example uses two local slave connections. Make a second connection for **Local Slave 2**:

Step	Action
1	Repeat the steps above to enable a second local slave (**Local Slave 2**). **NOTE:** The appropriate IP address for this example (192.168.20.10) was already assigned to the CPU's scanner service in the assignment of **Local Slave 1**.
2	Continue to the next procedure to configure the network scanner (originator, **O**).

Accessing Local Slaves with a Scanner

Introduction

Use these instructions to map local slave instances in a network scanner to the enabled local slaves in the CPU's embedded scanner service (**Local Slave 1**, **Local Slave 2**, **Local Slave 3**).

This example uses a BMENOC0301 Ethernet communication module as a network scanner (originator, **O**) that scans the CPU scanner service when it is enabled as a local slave (target, **T**).

Configure the BMENOC0301 module in an M580 Control Expert project.

Adding the Device DTM

Create a local slave instance that corresponds to an enabled local slave by name:

Step	Action
1	Open your M580 Control Expert project.
2	Right-click the BMENOC0301 module in the **DTM Browser** (**Tools > DTM Browser**) and select **Add**.
3	Select the DTM that corresponds to the CPU. **NOTE:** • The DTM used in this example corresponds to the CPU's scanner service. For other target devices, use the DTM from the manufacturer that corresponds to your scanner device. • The corresponding input I/O vision and output I/O vision variables are automatically created with the respective suffixes **_IN** and **_OUT**.
4	Press the **Add DTM** button to open the **Properties of device** dialog window.
5	Assign a context-sensitive **Alias name** that corresponds to **Local Slave 1** for the CPU. **Example**: BMEP58_ECPU_from_EDS_LS1
6	Click **OK** to see the local slave instance in the **DTM Browser**.

Mapping Local Slave Numbers

In the M580 Control Expert project, associate the local slave instances in the BMENOC0301 scanner with specific local slaves that are enabled for the CPU's scanner service:

Step	Action
1	In the **DTM Browser**, double-click the local slave instance that corresponds to **Local Slave 1** in the CPU target device (BMEP58_ECPU_from_EDS_LS1). **NOTE:** The default connection is **Local Slave 1 - Exclusive Owner**, which is most applicable to **Local Slave 1** in the target device.
2	Select **Local Slave 1 - Exclusive Owner**.
3	Click **Remove Connection** to delete the connection to **Local Slave 1**.
4	Click **Add Connection** to open the dialog box (**Select connection to add**).
5	Select **Local Slave 4 - Exclusive Owner**.
6	Click **Apply**.

The local slave (**Local Slave 1**) is now the target of a local slave instance with a context-sensitive connection name (**Local Slave 1 - Exclusive Owner**).

Mapping IP Addresses

Associate the IP address of the local slave (target, **T**) with the local slave instances in the scanner (originator, **O**) configuration:

Step	Action
1	Double-click the BMENOC0301 module in the **DTM Browser**.
2	In the navigation pane, expand the **Device List** (see Modicon M580, BMENOC0301/0311 Ethernet Communications Module, Installation and Configuration Guide).
3	Select a local slave instance (BMEP58_ECPU_from_EDS_LS1).
4	Select the **Address Setting** tab.
5	In the **IP Address** field, enter the IP address of the local slave device (192.168.20.10).
6	Click inside the navigation pane to make the **Apply** button active. **NOTE:** You may have to select **Disabled** in the drop-down menu (**DHCP for this device**) to activate the **OK** and **Apply** buttons.
7	Configure the data size.
8	Click **Apply**.

Configuring an Additional Connection

You have created one local slave instance that corresponds by name and IP address to an enabled local slave. This example uses two local slave connections, so make another connection for **Local Slave 2**.

Step	Action
1	Repeat the preceeding steps, page 402 to create a second local slave instance that corresponds to **Local Slave 2**.
2	Build the Control Expert project.

Accessing the Device DDT Variables

Step	Actiom
1	In the Project Browser (**Tools > Project Browser**), expand **Variables & FB instances**.
2	Double-click **Device DDT Variables** to see the device DDTs that correspond with the CPU's scanner service.

Local Slave Parameters

Accessing the Configuration

Open the **EtherNet/IP Local Slaves** configuration page:

Step	Action
1	Open the Control Expert project.
2	Open the **DTM Browser** (**Tools > DTM Browser**).
3	In the **DTM Browser**, double-click the CPU DTM to open the configuration window. **NOTE:** You can also right-click the CPU DTM and select **Open**.
4	Expand (+) **Device List** in the navigation tree to see the local slave instances.
5	Select the local slave instance to view the **Properties** and **Assembly** configuration tabs.

Properties

Identify and enable (or disable) the local slave on the **Properties** tab:

Parameter	Description	
Number	The Control Expert DTM assigns a unique identifier (number) to the device. These are the default values: • *local slave 1*: 129 • *local slave 2*: 130 • *local slave 3*: 131	
Active Configuration	Enabled	Enable the local slave with the configuration information in the **Assembly** fields when the CPU scanner service is an adapter for the local slave node.
	Disabled	Disable and deactivate the local slave. Retain the current local slave settings.
Comment	Enter an optional comment (maximum: 80 characters).	
Connection Bit	The connection bit is represented by an integer (769 ... 896). **NOTE:** • This setting is auto-generated after the local slave settings are input and the network configuration is saved. • The connection bit is represented by an integer: ◦ 385...387 (firmware v1.0) ◦ 769...896 (firmware v.2.10)	

Assembly

Use the **Assembly** area of the **Local Slave** page to configure the size of the local slave inputs and outputs. Each device is associated with these assembly instances:

- Outputs
- Inputs
- Configuration
- Heartbeat (The heartbeat assembly instance is for listen-only connections only.)

The Control Expert assembly numbers are fixed according to this table, where **O** indicates the originator (scanner) device and **T** indicates the target device:

Local Slave	Number		Connection
	Device	Assembly	
1	129	101	Outputs (T->O)
		102	Inputs (O->T)
		103	Configuration
		199	Heartbeat
2	130	111	Outputs (T->O)
		112	Inputs (O->T)
		113	Configuration
		200	Heartbeat
3	131	121	Outputs (T->O)
		122	Inputs (O->T)
		123	Configuration
		201	Heartbeat

NOTE: When using explicit messaging to read the CPU's scanner service assembly instance, allocate sufficient room for the response. The size of the response equals the sum of: assembly size + 1 byte (Reply service) + 1 byte (General Status).

Limitations (from the perspective of the local slave):

- *maximum RPI value*: 65535 ms
- *maximum timeout value*: 512 * RPI
- *outputs (T->O)*: 509 bytes maximum
- *inputs (O->T)*: 505 bytes maximum
- *configuration for the CPU scanner service*: 0 (fixed)

Working with Device DDTs

Introduction

Use Control Expert to create a collection of device derived data types (DDDTs) and variables that support communications and the transfer of data between the PAC and the various local slaves, distributed devices, and corresponding I/O modules.

You can create DDDTs and corresponding variables in the Control Expert DTM. Those program objects support your network design.

NOTE: The default device name depends on the firmware version installed in the selected CPU, and may be one of the following:

- T_BMEP58_ECPU
- T_BMEP58_ECPU_EXT
- T_M_ECPU_HSBY

Use the DDDTs for these tasks:

- Read status information from the Ethernet communication module.
- Write control instructions to the Ethernet communication module.

You can double-click the name of the DDDT in the **Project Browser** at any time to view its properties and open the corresponding EDS file.

NOTE: For applications that require multiple DDDTs, create an **Alias name** that logically identifies the DDDT with the configuration (module, slot, local slave number, etc.).

DDDT Variables

You can access the DDDTs and the corresponding variables in Control Expert and add them to a user-defined **Animation Table**. Use that table to monitor read-only variables and edit read-write variables.

Use these data types and variables to perform these tasks:

- Read the status of connections and communications between the Ethernet communication module and distributed EtherNet/IP and Modbus TCP devices:
  - The status is displayed in the form of a HEALTH_BITS array consisting of 32 bytes.
  - A bit value of 0 indicates the connection is lost or the communication module can no longer communicate with the distributed device.
- Toggle a connection ON (1) or OFF (0) by writing to a selected bit in a 16-word DIO_CTRL array
- Monitor the value of local slave and distributed device input and output items that you created in Control Expert.

NOTE: The HEALTH_BITS array is not copied to the standby CPU in a Hot Standby switchover. The DIO_CTRL array is copied to the standby CPU in a Hot Standby switchover.

Displaying the Order of Input and Output Items

View the DDDTs in Control Expert (**Project Browser > Variables & FB instances > Device DDT Variables**). The **Data Editor** is now open. Click the **DDT Types** tab.

The **Data Editor** displays each input and output variable. When you open the first input and output variables, you can see both the connection health bits, page 301 and the connection control bits, page 300.

This table shows the rule assignment for connection numbers:

Input Variables	Order	Output Variables
Modbus TCP input variables (note 1)	1	Modbus TCP output variables (note 1)
ERIO drop input variables	2	
local slave input variables (note 2)	3	local slave output variables (note 3)
EtherNet/IP input variables(note 1)	4	EtherNet/IP output variables (note 1)
NOTE 1: DDDTs are in this format: • **i.** by device number • **ii.** within a device (by connection number) • **iii.** within a connection (by item offset) **NOTE 2:** Local slave variables are in this format: • **i.** by local slave number • **ii.** within each local slave (by item offset)		

Hardware Catalog

Introduction

The Control Expert **Hardware Catalog** displays the modules and devices that you can add to a Control Expert project. Each module or device in the catalog is represented by a DTM that defines its parameters.

Introduction to the Hardware Catalog

Introduction

The Control Expert **Hardware Catalog** contains a list of modules and devices that you can add to a Control Expert project. EtherNet/IP and Modbus TCP devices are located in the **DTM Catalog** tab at the bottom of the **Hardware Catalog**. Each module or device in the catalog is represented by a DTM that defines its parameters.

EDS Files

Not all devices in today's market offer device-specific DTMs. Some devices are defined by device-specific EDS files. Control Expert displays EDS files in the form of a DTM. In this way, you can use Control Expert to configure devices that are defined by an EDS file in the same way you would configure a device defined by its DTM.

Other devices lack both a DTM and an EDS file. Configure those devices by using the generic DTM on the **DTM Catalog** page.

View the Hardware Catalog

Open the Control Expert **Hardware Catalog**:

Step	Action
1	Open Control Expert.
2	Find the **PLC bus** in the **Project Browser**.
3	Use one method to open the catalog: • Use the pull-down menu (**Tools > Hardware Catalog**). • Double-click an empty slot in the **PLC bus**.

Adding a DTM to the Control Expert Hardware Catalog

A Manufacturer-Defined Process

Before a DTM can be used by the Control Expert **Hardware Catalog**, install the DTM on the host PC (the PC that is running Control Expert).

The installation process for the DTM is defined by the device manufacturer. Consult the documentation from the device manufacturer to install a device DTM on your PC.

NOTE: After a device DTM is successfully installed on your PC, update the Control Expert Hardware Catalog to see the new DTM in the catalog. The DTM can then be added to a Control Expert project.

Adding an EDS File to the Hardware Catalog

Introduction

You may want to use an EtherNet/IP device for which no DTM is in the catalog. In that case, use these instructions to import the EDS files into the catalog to create a corresponding DTM.

Control Expert includes a wizard you can use to add one or more EDS files to the Control Expert **Hardware Catalog**. The wizard presents instruction screens to execute these commands:

- Simplify the addition of EDS files to the **Hardware Catalog**.
- Provide a redundancy check when you add duplicate EDS files to the **Hardware Catalog**.

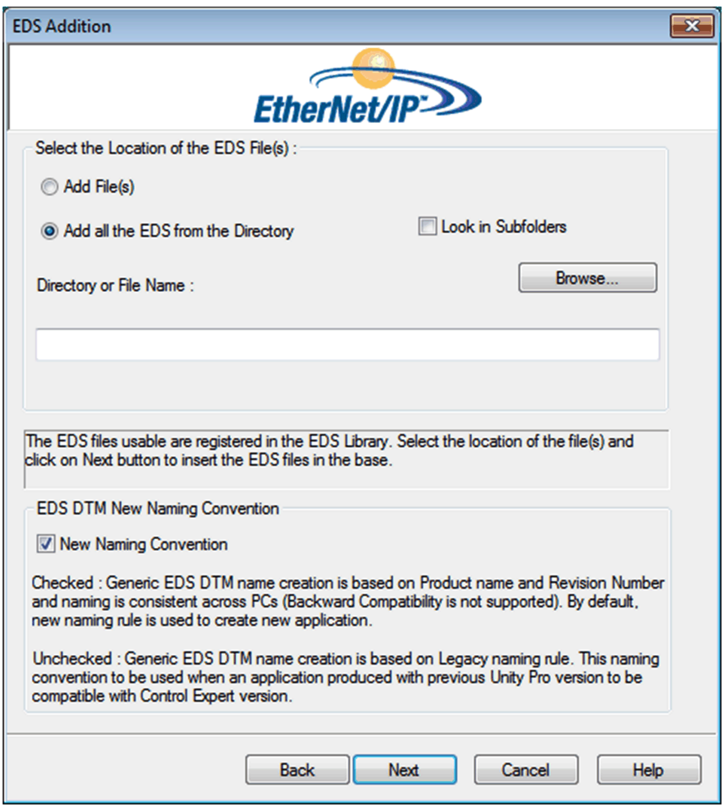
NOTE: The Control Expert **Hardware Catalog** displays a partial collection of DTMs and EDS files that are registered with the ODVA. This library includes DTMs and EDS files for products that are not manufactured or sold by Schneider Electric. The non-Schneider Electric EDS files are identified by vendor in the catalog. Contact the identified device's manufacturer for inquiries regarding the corresponding non-Schneider Electric EDS files.

Adding EDS Files

Open the **EDS Addition** dialog box:

Step	Action
1	Open a Control Expert project that includes an Ethernet communication module.
2	Open the **DTM Browser** (**Tools > DTM Browser**).
3	In the **DTM Browser**, select a communication module.
4	Right-click on the communication module and scroll to **Device menu > Additional functions > Add EDS to library**.
5	In the **EDS Addition** window, click **Next**.

You can now see this page:



Add one or more EDS files to the library:

Step	Action
1	Use these commands in the **Select the Location of the EDS File(s)** area of the **EDS Addition** dialog box to identify the location of the EDS files: • **Add File(s)**: Add one or more EDS files that are individually selected. • **Add all the EDS from the Directory**: Add all files from a selected folder. (Check **Look in Subfolders** to add EDS files from the folders within the selected folder.)
2	Click **Browse** to open a navigation dialog box.
3	Select the location of the EDS file(s): • Navigate to at least one EDS file. • Navigate to a folder that contains EDS files. **NOTE:** Keep the location selected (highlighted).
4	Click **Select** to close the navigation window. **NOTE:** Your selection appears in the **Directory or File Name** field.
5	Choose the naming convention rule for the EDS DTM name creation. The new naming convention is based on Model Name / Product Name and Revision. A random character is automatically suffixed when Model Name / Product Name and Revision of an EDS file in the library is identical. The new naming convention is irrespective of the order in which EDS files are added to device library. By default, the **New Naming Convention** check box is selected and the new naming rule applies. **NOTE:** To keep backward compatibility with Unity Pro/Control Expert versions, unchecked the **New Naming Convention** check box and the naming rule is based on Model Name / Product Name.
6	Click **Next** to compare the selected EDS files to the files in the library. **NOTE:** If one or more selected EDS files is a duplicate, a **File Already Exists** message appears. Click **Close** to hide the message.
7	The next page of the **EDS Addition** wizard opens. It indicates the status of each device you attempted to add: • check mark (green): The EDS file can be added. • informational icon (blue): There is a redundant file. • exclamation point (red): There is an invalid EDS file. **NOTE:** You can click **View Selected File** to open and view the selected file.
8	Click **Next** to add the non-duplicate files. **Result**: The next page of the **EDS Addition** wizard opens to indicate that the action is complete.
9	Click **Finish** to close the wizard. **Result**: The hardware catalog automatically updates.

Removing an EDS File from the Hardware Catalog

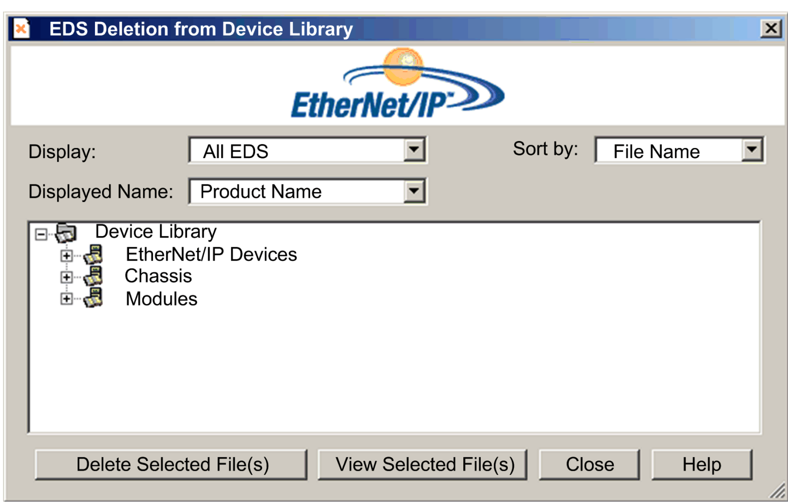
Introduction

You can remove a module or device from the list of available devices in the Control Expert **Hardware Catalog** by removing its **EDS** file from the library.

When you remove an EDS file from the library, the device or module disappears from the **DTM Catalog**. However, removing the file from the library does not delete the file from its stored location, so you can import the file again later.

Removing an EDS File from the Catalog

Use these steps to remove an EDS file from the catalog:

<table>
<tr><th>Step</th><th colspan="2">Action</th></tr>
<tr><td>1</td><td colspan="2">Open the Control Expert DTM Browser (Tools > DTM Browser).</td></tr>
<tr><td>2</td><td colspan="2">In the DTM Browser, select an Ethernet communication module.</td></tr>
<tr><td>3</td><td colspan="2">Right-click the module and scroll to Device menu > Additional functions > Remove EDS from library to open the EDS Deletion from Device Library window:

</td></tr>
<tr><td rowspan="2">4</td><td colspan="2">Use the selection lists in the heading of this window to specify how EDS files are displayed:</td></tr>
<tr><td>Display</td><td>Choose criteria to filter the list of EDS files:
All EDS (no filtering)
Only Devices</td></tr>
</table>

Step	Action	
		• **Only Chassis** • **Only Modules**
	Sort by	Choose criteria to sort the list of displayed EDS files: • **File Name** • **Manufacturer** • **Category** • **Device Name**
	Displayed Name	Choose the identifier for each device: • **Catalog Name** • **Product Name**
5	Expand (+) the **Device Library** navigation tree and select the EDS file you want to remove. **NOTE:** Click **View Selected File** to see the read-only contents of the selected EDS file.	
6	Click the **Delete Selected File(s)** button to open the **DeleteEDS** dialog box.	
7	Click **Yes** to remove the selected EDS file from the list.	
8	Repeat these steps for each EDS file you want to delete.	
9	Click **Finish** to close the wizard. **Result**: The hardware catalog automatically updates.	

Export / Import EDS Library

Introduction

To use the same project on two Control Expert installations (for example a source, and a target Host PCs), you may have to update the DTM **Hardware Catalog** of the target Host PC.

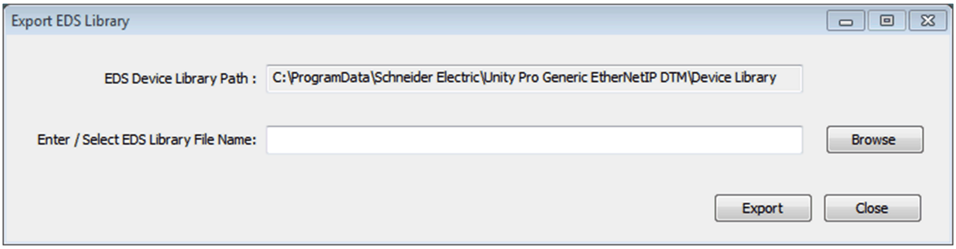
Instead of adding one by one the missing EDS files in the target Host PC, you can update the DTM **Hardware Catalog** in two steps:

- Exporting the EDS library from the source Host PC.
- Importing the EDS library in the target Host PC.

NOTE: When you export the EDS library, the software generates an **.DLB** file which contains all the DTM created form EDS files.

Exporting EDS Library

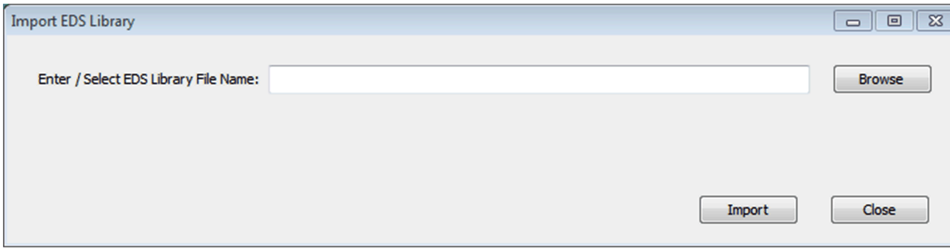
Open the **Export EDS Library** dialog box:

Step	Action
1	Open a Control Expert project that includes an Ethernet communication module.
2	Open the **DTM Browser** (**Tools > DTM Browser**).
3	In the **DTM Browser**, select a communication module.
4	Right-click on the communication module and scroll to **Device menu > Additional functions > Export EDS library** to open the **Export EDS library** window: 
5	For the archived EDS library you want to create: • Enter the full folder path along with the file name in the **Enter / Select EDS Library File Name** field, or • Click **Browse** to open a navigation dialog box: ◦ Select the location, and ◦ Enter the file name, and ◦ Click **Save** to close the navigation window and your selection appears in the **Enter / Select EDS Library File Name** field.
6	Click **Export** to create the archived EDS library. **Result:** A new wizard opens to indicate that the export is complete. Click **Ok** to close the wizard.
7	In the **Export EDS library** window, click **Close**.

Importing EDS Library

Use these steps to import an archived EDS library:

Step	Action
1	Open the Control Expert **DTM Browser** (**Tools > DTM Browser**).
2	In the **DTM Browser**, select an Ethernet communication module.

Step	Action
3	Right-click the module and scroll to **Device menu > Additional functions > Import EDS library** to open the **Import EDS library** window: 
4	For the archived EDS library you want to import: • Enter the full folder path along with the file name in the **Enter / Select EDS Library File Name** field, or • Click **Browse** to open a navigation dialog box: ◦ Select the location, and ◦ Enter the file name, and ◦ Click **Save** to close the navigation window and your selection appears in the **Enter / Select EDS Library File Name** field.
5	Click **Import**. **Result:** A new wizard opens to indicate that the export is complete. Click **Ok** to close the wizard.
6	In the **Import EDS library** window, click **Close**.

M580 CPU Embedded Web Pages

Introduction

The M580 CPU includes a Hypertext Transfer Protocol Secure (HTTPS). The server transmits web pages for the purpose of monitoring, diagnosing, and controlling remote access to the communication module. The server provides secure access to the CPU from standard internet browsers.

Introducing the Standalone Embedded Web Pages

Introduction

Use the embedded web server pages to display real-time diagnostics data for the M580 controller and other networked devices.

Browser Requirements

The embedded web server in the M580 controller displays data in standard HTML web pages. Access the embedded web pages on a PC, iPad, or Android tablet with these browser versions:

Browser	Application	Minimum Version
Internet Explorer	Windows	v8 or any subsequent supporting version
	Windows Phone OS	v10 or any subsequent supporting version
Google Chrome	Windows	v11 or any subsequent supporting version
	Android OS (minimum version 4)	v35 or any subsequent supporting version
Mozilla Firefox	Windows	v4 or any subsequent supporting version
Safari	Apple Macintosh	v6.0 (See note below.)
	Windows	*(none)*

Access the Web Pages

Open the **Home** page:

Step	Action
1	Open an Internet browser.
2	In the address bar, enter the IP address of the M580 controller, page 148.
3	Press **Enter** and wait for the page to open. Two submenus are available: • **Home** • **Diagnostics**

Click the **Home** submenu to access the **Status Summary**, page 420 page.

Click the **Diagnostics** submenu to expand and access the following pages:

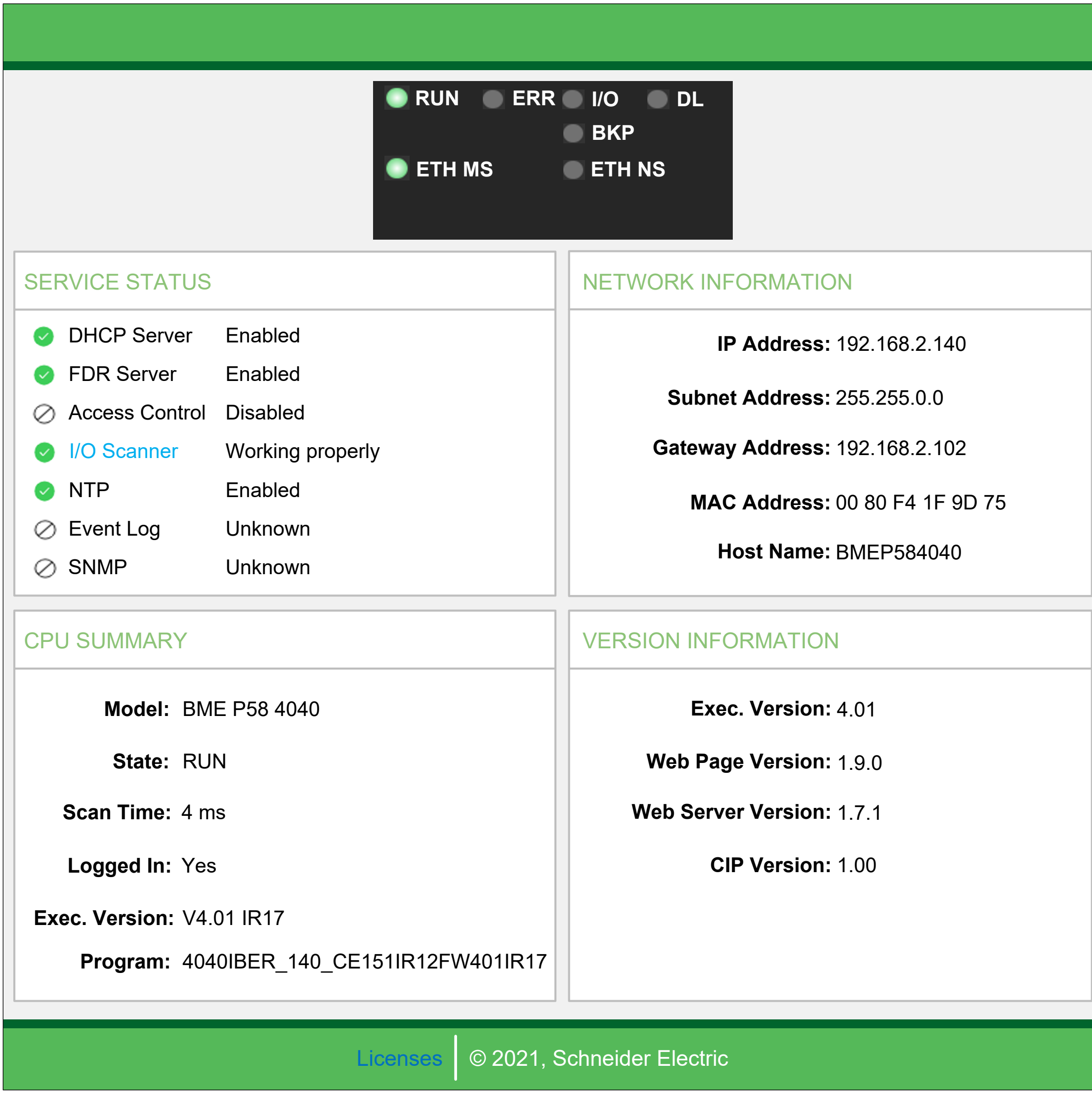
- **Module**:
  - **Status Summary**, page 420
  - **Performance**, page 423
  - **Port Statistics**, page 425
- **Connected Devices**:
  - **I/O Scanner**, page 427
  - **Messaging**, page 430
- **Services**:
  - **QoS**, page 431
  - **Network Time Service**, page 433
  - **Redundancy**, page 438
- **System**:
  - **Alarm Viewer**, page 440
  - **Rack Viewer**, page 442
- **File Manager**:
  - **Data Storage**, page 446
  - **Event Log**, page 449

Status Summary (Standalone CPUs)

Open the Page

Access the **Status Summary** page from the **Diagnostics** tab (**Module > Status Summary**):

Status Summary (Standalone CPU) page:



NOTE:

- This page is updated every 5 seconds.
- For Hot Standby CPUs refer to the **Status Summary** page for Hot Standby CPUs, page 451.

Diagnostic Information

The objects on this page provide status information:

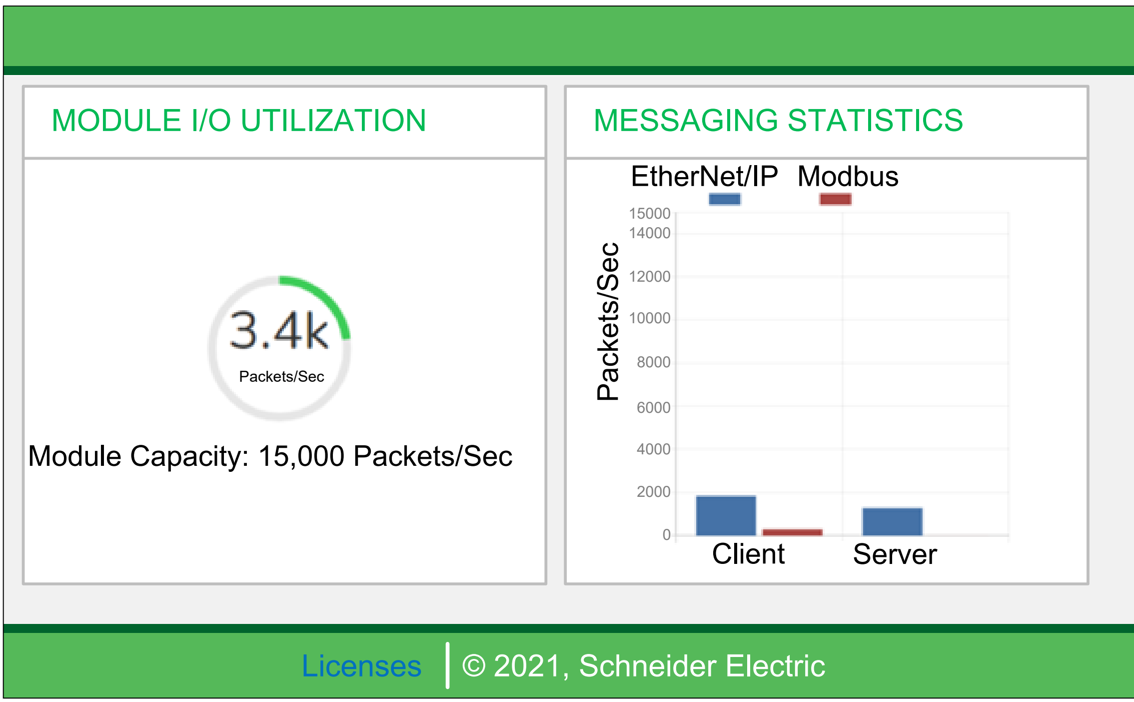
Parameters	Description	
LEDs	The black field contains LED indicators (**RUN**, **ERR**, etc.). **NOTE:** The diagnostics information is explained in the description of LED activity and indications, page 63.	
Service Status	green	The available service is operational and running.
	red	An error is detected in an available service.
	black	The available service is not present or not configured.
Version Information	This field describes the software versions that are running on the CPU.	
CPU Summary	This field describes the CPU hardware and the applications that are running on the CPU.	
Network Information	This field contains network and hardware address information and connectivity that corresponds to the CPU.	

Performance

Open the Page

Access the **Performance** page from the **Diagnostics** tab (**Module > Performance**):

Performance page:



NOTE:

- Move the mouse over the dynamic graphs to see the current numeric values.
- This page is updated every 5 seconds.

Diagnostic Information

This table describes the performance statistics:

Field	Description
Module I/O Utilization	This graph shows the total number of packets (per second) the CPU can handle at once.
Messaging Statistics	This graph shows the number of Modbus/TCP or EtherNet/IP messages per second for the client or server.

Port Statistics

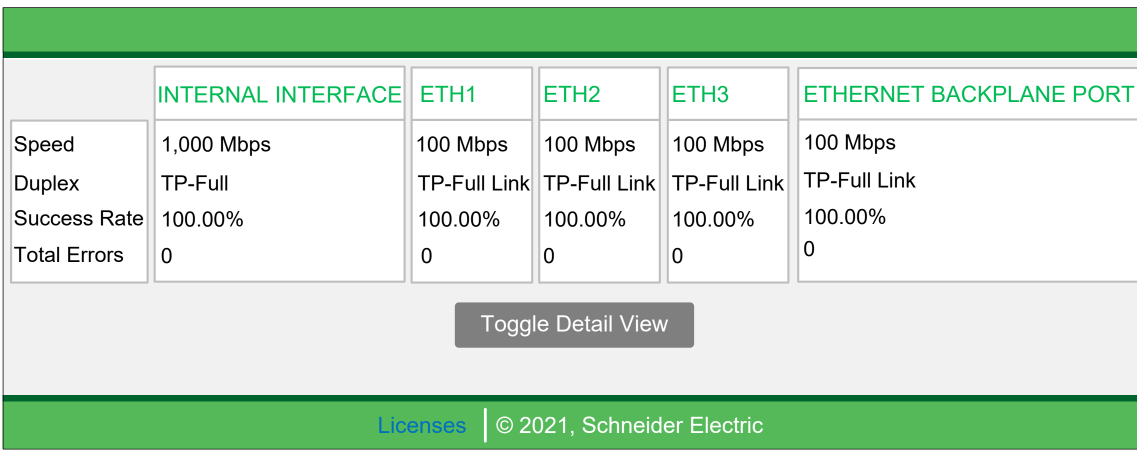
Open the Page

Access the **Port Statistics** page from the **Diagnostics** tab (**Module > Port Statistics**):

NOTE: This page is updated every 5 seconds.

Click **Toggle Detail View** to change between the detail and non-detail view of the page.

Port Statistics page (non–detail view):



	INTERNAL INTERFACE	ETH1	ETH2	ETH3	ETHERNET BACKPLANE PORT
Speed	1,000 Mbps	100 Mbps	100 Mbps	100 Mbps	100 Mbps
Duplex	TP-Full	TP-Full Link	TP-Full Link	TP-Full Link	TP-Full Link
Success Rate	100.00%	100.00%	100.00%	100.00%	100.00%
Total Errors	0	0	0	0	0

Port Statistics page (detail view):

	INTERNAL INTERFACE	ETH1	ETH2	ETH3	ETHERNET BACKPLANE PORT
Speed	1,000 Mbps	100 Mbps	100 Mbps	100 Mbps	100 Mbps
Duplex	TP-Full	TP-Full Link	TP-Full Link	TP-Full Link	TP-Full Link
Frames Transmitted	126,405,904	22,172,504	1,387,779	128,125,148	22,631,961
Frames Received	251,592, 440	16,285	17,717,591	252,189,875	4,763,558
Bytes Transmitted	824,012,094	1,650,722,909	111,228,318	1,464,271,580	1,710,176,669
Bytes Received	-1,064,465,543	6,937,846	1,325,197,655	-82,010,691	448,453,655
Inbound Packet Errors	0	0	0	0	0
Inbound Packets Discarded	0	0	0	0	0
Outbound Packet Errors	0	0	0	0	0
Outbound Packets Discarded	0	0	0	0	0
Excessive Collisions	0	0	0	0	0
Late Collisions	0	0	0	0	0
CRC Errors	0	0	0	0	0
Carrier Sense Errors	0	0	0	0	0
FCS Errors	0	0	0	0	0
Alignment Errors	0	0	0	0	0
Internal MAC Trans. Errors	0	0	0	0	0
Internal MAC Rec. Errors	0	0	0	0	0
SQE Total Errors	0	0	0	0	0

Toggle Detail View

Licenses |

NOTE: This page is updated every 5 seconds.

Diagnostic Information

This page shows the statistics for each port on the CPU. This information is associated with the configuration of the Ethernet ports, page 73 and the configuration of the service/extended port, page 162.

The frame color indicates the port activity:

- *green*: active
- *gray*: inactive
- *yellow*: error detection
- *red*: error detection

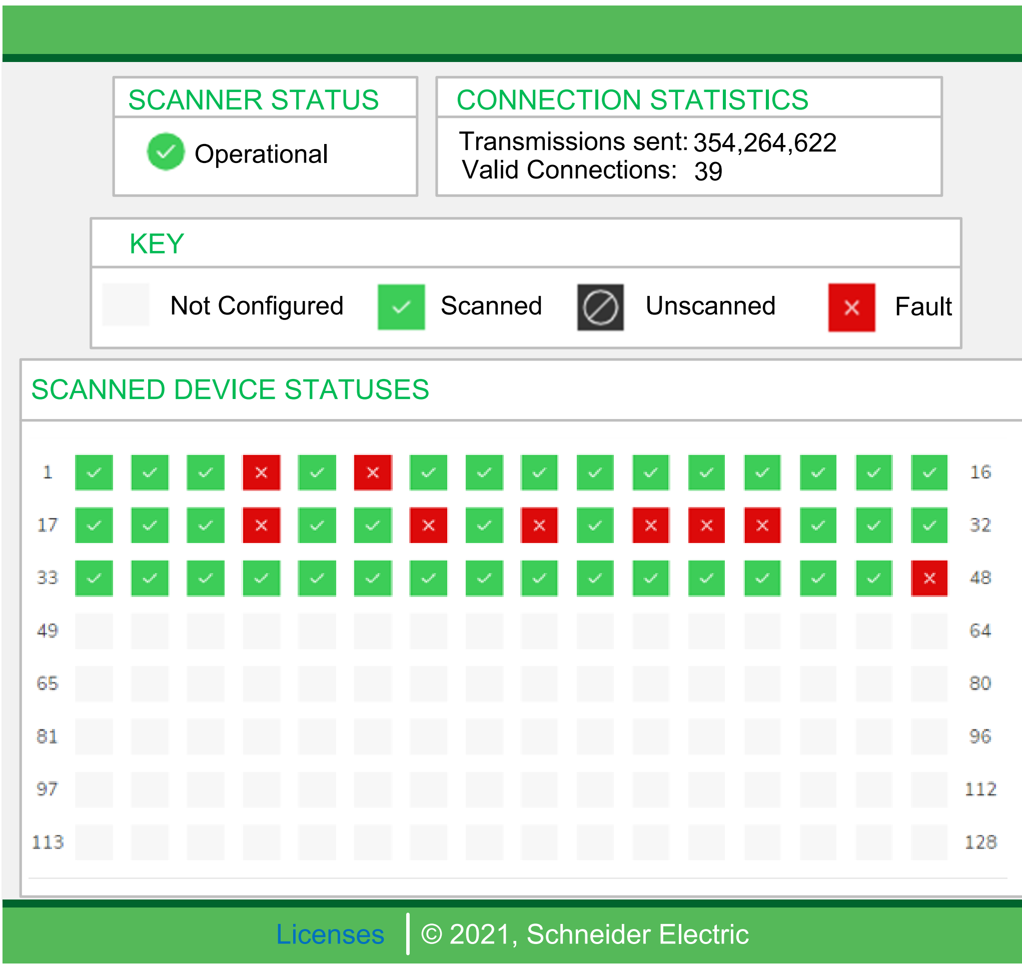
I/O Scanner

Open the Page

Access the **I/O Scanner** page from the **Diagnostics** tab—

(**Connected Devices > I/O Scanner**):

I/O Scanner page:



NOTE: This page is updated every 5 seconds.

Toggling Between Scanners

Some M580 safety CPUs include both a Modbus TCP (Ethernet I/O) scanner and a CIP Safety (IEC 61784-3) scanner. For these safety CPUs, this page includes a **Toggle Scanner** button. Use this to change the display from one scanner to the other. When the CIP Safety scanner is displayed, the web page banner reads **I/O Scanner - CIP Safety**.

Diagnostic Information

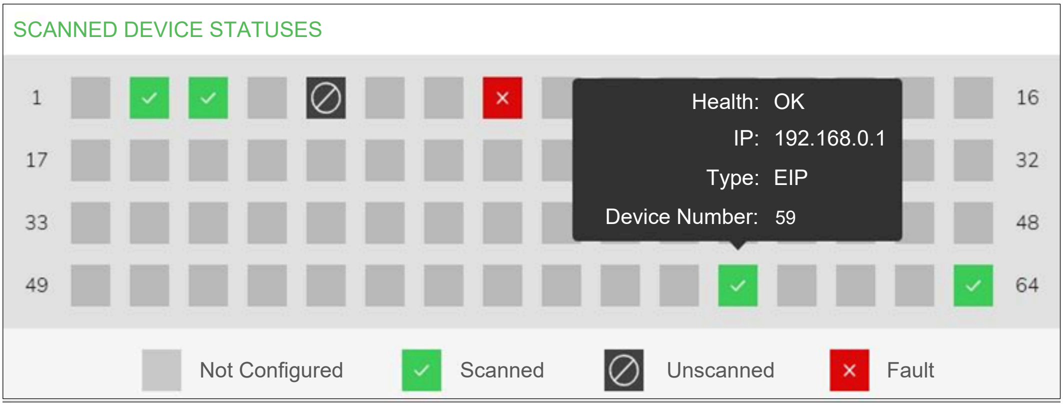
This table describes the scanner status and connection statistics:

Scanner Status	**Operational**	The I/O scanner is enabled.
	Stopped	The I/O scanner is disabled.
	Idle	The I/O scanner is enabled but not running.
	Unknown	The I/O scanner returns unexpected values from the device.
Connection Statistics	**Transactions per Second**	
	Number of Connections	

In the **Scanned Device Status** display, the colors that appear in each block indicate these states for specific remote devices:

Color	Indication	Status
gray	**Not Configured**	There is an unconfigured device.
black	**Unscanned**	The scanning of the specific device has been intentionally disabled.
green	**Scanned**	A device is being scanned successfully.
red	**Fault**	A device that is being scanned is returning detected errors.

Hold the cursor over any block to get information for a specific device:

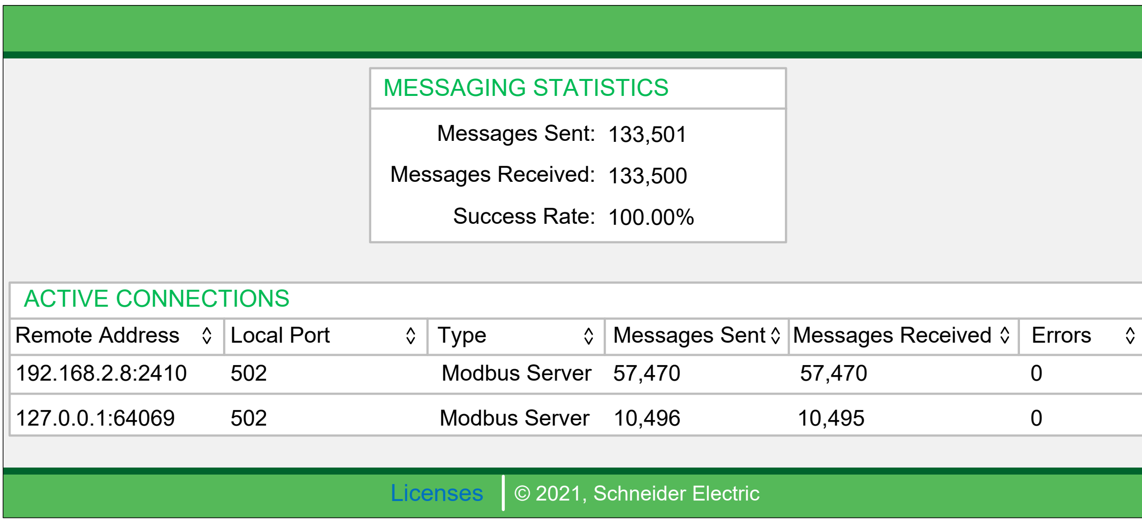


Messaging

Open the Page

Access the **Messaging** page from the **Diagnostics** tab (**Connected Devices > Messaging**):

Messaging page:



NOTE: This page is updated every 5 seconds.

Diagnostic Information

This page shows current information for open Modbus TCP connections on port 502:

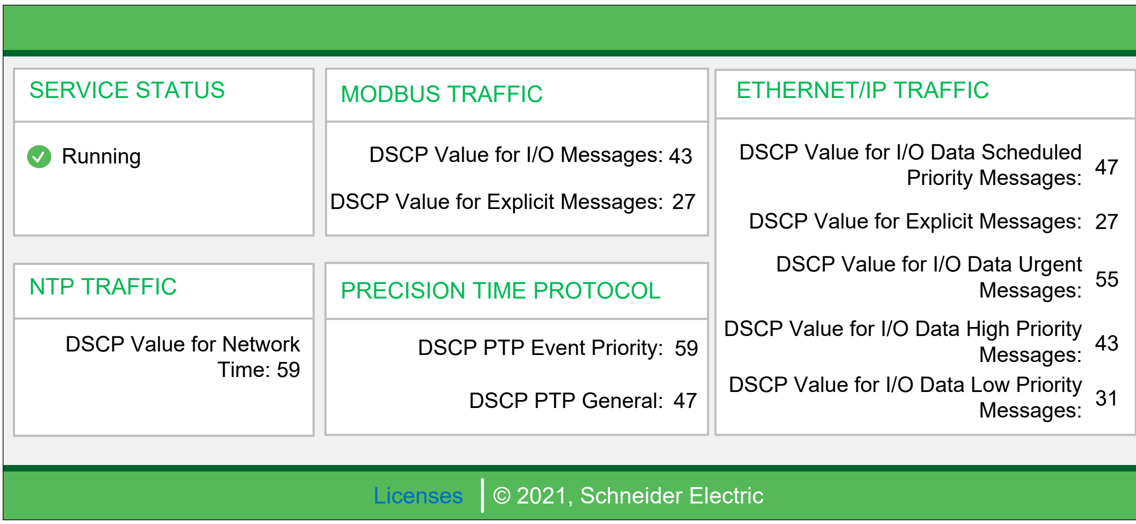
Field	Description
Messaging Statistics	This field contains the total number of sent and received messages on port 502. These values are not reset when the port 502 connection is closed. Therefore, the values indicate the number of messages that have been sent or received since the module was started.
Active Connections	This field shows the connections that are active when the **Messaging** page is refreshed.

QoS

Open the Page

Access the **QoS** (quality of service) page from the **Diagnostics** tab (**Services > QoS**):

QoS page:



NOTE:

- Configure the QoS in Control Expert, page 160.
- Click **Detail View** to expand the list of parameters.
- This page is updated every 5 seconds.

Service Status

This table shows the possible states for the **Service Status**:

Status	Description
Running	The service is correctly configured and running.
Disabled	The service is disabled.
Unknown	The status of the service is not known.

Diagnostic Information

This page displays information about the QoS service that you configure in Control Expert, page 160.

When you enable QoS, the module adds a differentiated services code point (DSCP) tag to each Ethernet packet it transmits, thereby indicating the priority of that packet:

Field	Parameter	Description
Precision Time Protocol	DSCP PTP Event Priority	Point-to-point time synchronization.
	DSCP PTP General	Point-to-point general.
EtherNet/IP Traffic	DSCP Value for I/O Data Scheduled Priority Messages	Configure the priority levels to prioritize the management of data packets.
	DSCP Value for Explicit Messages	
Modbus/TCP Traffic	DSCP Value for I/O Messages	**NOTE:** Use a larger timeout value for explicit messaging connections and a smaller timeout value for implicit messaging connections. The specific values that you employ depend on your application requirements.
	DSCP Value for Explicit Messages	
Network Time Protocol Traffic	DSCP Value for Network Time	—

Considerations

Take measures to effectively implement QoS settings in your Ethernet network:

- Use only network switches that support QoS.
- Apply the same DSCP values to all network devices and switches.
- Use switches that apply a consistent set of rules for handling the different DSCP values when transmitting and receiving Ethernet packets.

NTP

Introduction

The **NTP** page displays information about the network time service. There are three versions of this page, depending on the CPU firmware version and NTP mode:

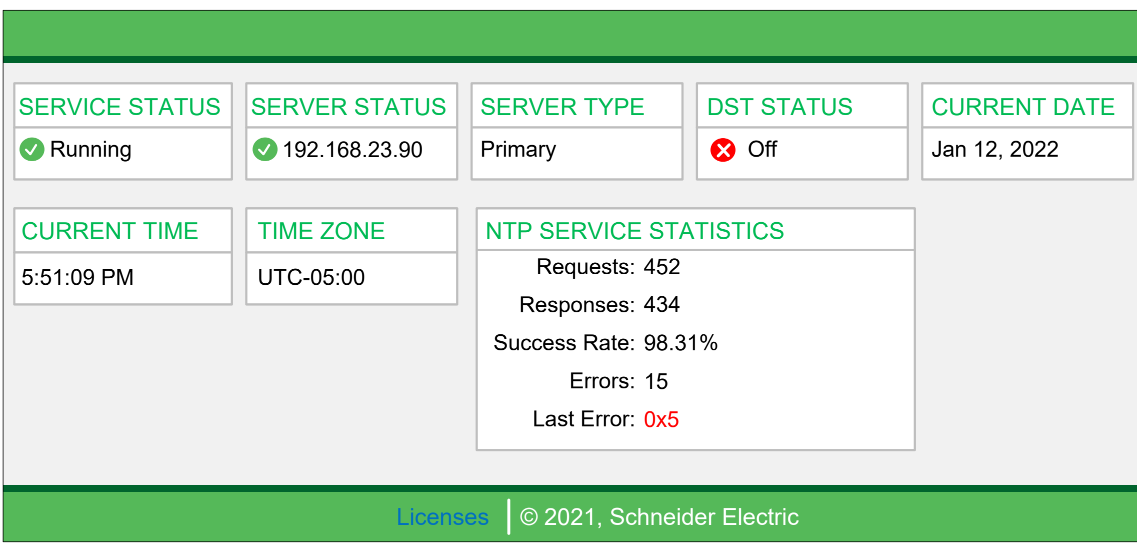
- Versions earlier than V4.01 display SNTP content.
- Version V4.01 and any subsequent supporting version(s) display NTPv4 content, either:
  - **Client / Server**mode
  - **Server only** mode

Configure this service in Control Expert, page 155.

Open the Page

Access the **NTP** page from the **Diagnostics** tab (**Services > NTP**):

SNTP content



NOTE:

- Click **Reset Counters** to reset all dynamic counters to 0.
- This page is updated every 5 seconds.

NTP content – Client/Server mode

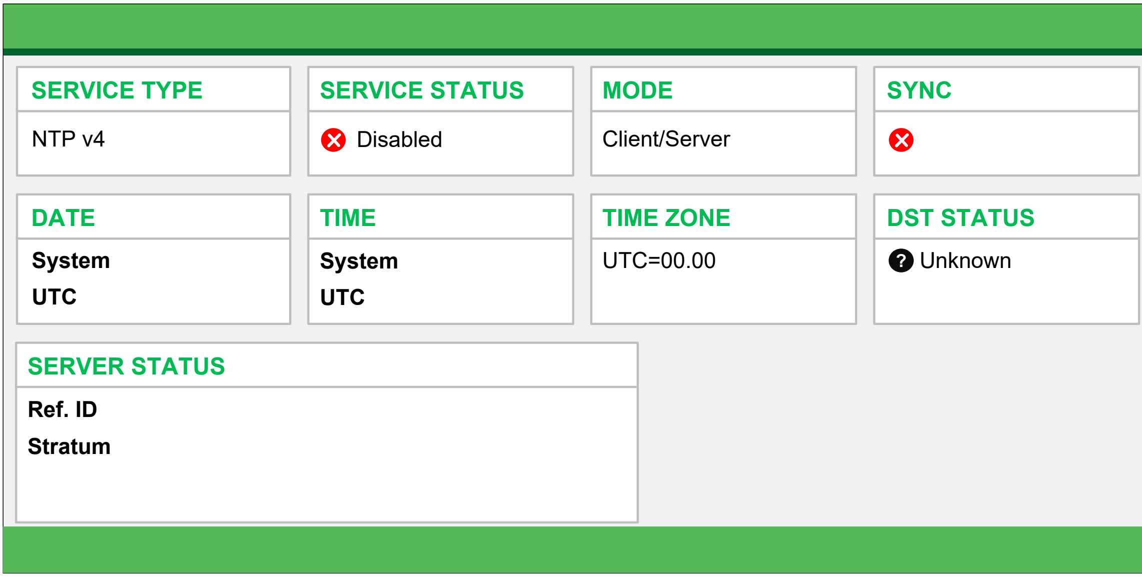
SERVICE TYPE	SERVICE STATUS	MODE	SYNC
NTP v4	Disabled	Client/Server	

DATE	TIME	TIME ZONE	DST STATUS
System **UTC**	**System** **UTC**	UTC=00.00	Unknown

SERVICE STATISTICS	SERVER STATUS
Root Delay **Root Dispersion** **Accuracy**	**Ref. ID** **Stratum** **Polling Time**

IP Address	Ref. ID	Select	Reach %	Stratum	Poll	Delay	Offset	Jitter
192.168.10.10	.LOCL.		0 %	16	32	15,25	20,526	25,000
192.168.10.11	172.16.10.45	**Current**	100 %	5	512	-1,258	-0,358	5,259
192.168.10.12	172.16.10.46	Candidate	100 %	5	512	-1,258	-0,358	5,259
192.168.10.13	172.16.10.48	Candidate	100 %	5	1024	-1,258	-0,358	5,259
192.168.10.14	172.16.11.145		12 %	7	32	-1,258	-0,358	5,259
192.168.10.15	.INIT.		0 %	16	32	100,000	25000	100,000
192.168.10.16	.STEP.		0 %	16	32	-1,258	-0,358	5,259
192.168.10.17	10.10.25.65		25 %	7	128	-1,258	-0,358	5,259

NTP content – Server only mode



Diagnostic Information

The Network Time Service synchronizes computer clocks over the Internet for the purposes of event recording (sequence events), event synchronization (trigger simultaneous events), or alarm and I/O synchronization (time stamp alarms):

SNTP and NTPv4 Common Data:

Field	Description	
Service Status	**Running**	The NTP service is correctly configured and running.
	Disabled	The NTP service is disabled.
	Unknown	The NTP service status is unknown.
Current Date (SNTP), **Date** (NTPv4)	SNTP: the current date in the selected time zone. NTPv4: • System: the local CPU date. • UTC: the same date in UTC.	
Current Time (SNTP), **Time** (NTPv4)	SNTP: the current time in the selected time zone. NTPv4: • System: the local CPU time. • UTC: the same time in UTC.	

SNTP and NTPv4 Common Data: (Continued)

Field	Description	
Time Zone	The time zone in terms of plus or minus Universal Time, Coordinated (UTC).	
DST Status	**Running**	DST (daylight saving time) is configured and running.
	Disabled	DST is disabled.
	Unknown	The DST status is unknown.

SNTP Data Only:

Field	Description	
Server Status	green	The server is connected and running.
	red	An incorrect server connection is detected.
	gray	The server status is unknown.
Server Type	**Primary**	A primary server polls a master time server for the current time.
	Secondary	A secondary server requests the current time only from a primary server.
NTP Service Statistics	These fields show the current values for service statistics.	
	Number of Requests	This field shows the total number of requests sent to the NTP server.
	Success Rate	This field shows the percentage of successful requests out of the total number of requests.
	Number of Responses	This field shows the total number of responses received from the NTP server.
	Last Error	This field contains the error code of the last error that was detected during the transmission of an e-mail message to the network.
	Number of Errors	This field contains the total number of e-mail messages that could not be sent to the network or that have been sent but not acknowledged by the server.

NTPv4 Data Only:

Field	Description
Service Type	Always **NTP v4**
Mode	The CPU's NTP role or roles: • **Server only**: The CPU provides time data to local NTP client devices. • **Client / Serve**r: The CPU receives time data from a remote NTP server, and also provides time data to local NTP client devices.
Sync	The CPU time is synchronized: • In **Client / Server** mode: to an external NTP server.

NTPv4 Data Only: (Continued)

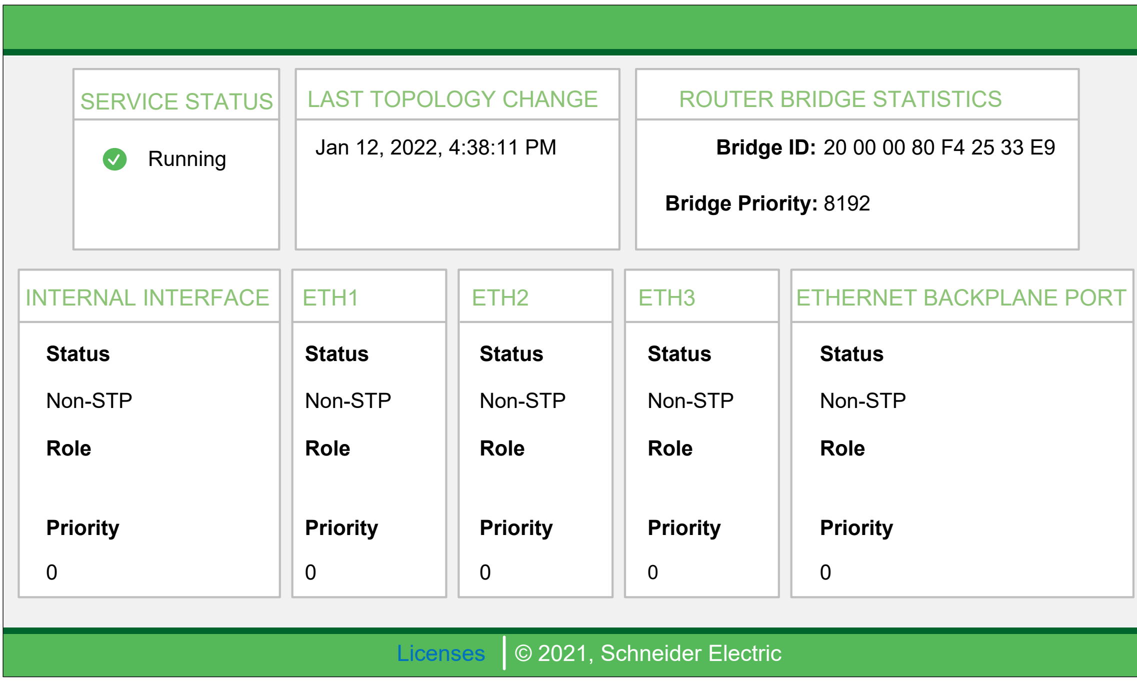
Field	Description	
	• In **Server only** mode: in the CPU configuration.	
Service Statistics	In **Client / Server** mode:	
	Root delay	As NTP client, the round trip request delay, in milliseconds, from a client to a stratum 1 server.
	Root dispersion	A NTP client, the additional delay contributed by other factors.
	Accuracy	As NTP client, the estimated difference between local (client) time and server time.
Server Status	**Ref. ID**	IPv4 address of the time source.
	Stratum	The relative position in the hierarchy between this client and the original time source (stratum 1) reference. If the mode is: • Server/Client: the value equals the system peer stratum value + 1. • Server only (or orphan): a user-defined value.
	Polling Time	As NTP client only: the polling interval, in seconds.
<NTP Peers Statuses> (NTP clients only)	NTP client CPU can be configured with up to 8 time source peers, each a potential server to the CPU NTP client.	
	IP	Peer IPv4 address of the peer.
	Ref. ID	IP address of the time source used by the peer.
	Select	Indicates the peer used as the time source (Current) and other viable peer time sources (Candidate).
	Reach count	Percentage of NTP messages successfully sent to and received from the peer.
	Stratum	The relative position in the hierarchy between this client and the original time source (stratum 1) reference.
	Poll	Polling interval, in seconds.
	Delay	Time to send request / receive response.
	Offset	The value to subtracted from received time value to obtain time value to be applied.
	Jitter	Variability in delay.

Redundancy

Open the Page

Access the **Redundancy** page on the **Diagnostic** tab (**Services > Redundancy**):

Redundancy page:



NOTE: This page is updated every 5 seconds.

Diagnostic Information

This page displays values from the RSTP configuration in Control Expert, page 150:

Field	Description	
Service Status	**Running**	The RSTP bridge on the corresponding CPU is properly configured and running.
	Disabled	The RSTP bridge on the corresponding CPU is disabled.
	Uknown	The status of the RSTP bridge on the corresponding CPU is not known.

Field	Description	
Last Topology Change	These values represent the date and time that the last topology change was received for the corresponding **Bridge ID**.	
Redundancy Status	**Status**	If an RSTP port: Discarding, learning, or forwarding. If not: Non-STP
	Role	If an RSTP port: Root, designated, alternate, backup, or disabled. If not: blank
	Priority	The RSTP priority assigned to the port..
Router Bridge Statistics	**Bridge ID**	This unique bridge identifier is the concatenation of the bridge RSTP priority and the MAC address.
	Bridge Priority	In Control Expert, configure the RSTP operating state, page 150 of the **Bridge ID**.

Alarm Viewer

Open the Page

Access the **Alarm Viewer** page from the **Diagnostics** tab (**System > Alarm Viewer**):

Alarm Viewer page:

ALARM LOG

Type	Status	Message	Occurrence	Acknowledged	Zone
System	Error	Character string fault	Nov 21, 2021, 5:52:30 PM	Not Required	0
System	Error	Character string fault	Nov 21, 2021, 5:52:30 PM	Not Required	0
System	Error	Character string fault	Nov 21, 2021, 5:52:30 PM	Not Required	0
System	Error	Arithmetic error	Nov 21, 2021, 5:52:30 PM	Not Required	0
System	Error	Character string fault	Nov 21, 2021, 5:52:30 PM	Not Required	0
System	Error	Character string fault	Nov 21, 2021, 5:52:30 PM	Not Required	0
System	Error	Character string fault	Nov 21, 2021, 5:52:30 PM	Not Required	0
System	Error	Character string fault	Nov 21, 2021, 5:52:30 PM	Not Required	0
System	Error	Task period Overshoot	Nov 21, 2021, 5:52:39 PM	Not Required	0

Licenses | © 2021, Schneider Electric

NOTE: This page is updated every 5 seconds.

Diagnostic Information

The **Alarm Viewer** page reports detected application errors. You can read, filter, and sort information about alarm objects on this page. Adjust the type of information displayed by the **Alarm Viewer** in the **Filter Alarms** box.

Each alarm has a timestamp, a description, and an acknowledgement status:

- critical (red)

- acknowledged (green)
- information (blue) (These alarms do not require acknowledgement.)

This table describes the components of the page:

Column	Description	
Type	This column describes the alarm type.	
Status	**STOP**	You need to acknowledge the alarm.
	ACK	An alarm has been acknowledged.
	OK	An alarm does not require acknowledgment.
Message	This column contains the text of the alarm message.	
Occurance	This column contains the date and time that the alarm occurred.	
Acknowledged	This column reports the acknowledged status of the alarm.	
Zone	This column contains the area or geographical zone from which the alarm comes (0: common area).	

Rack Viewer

Open the Page

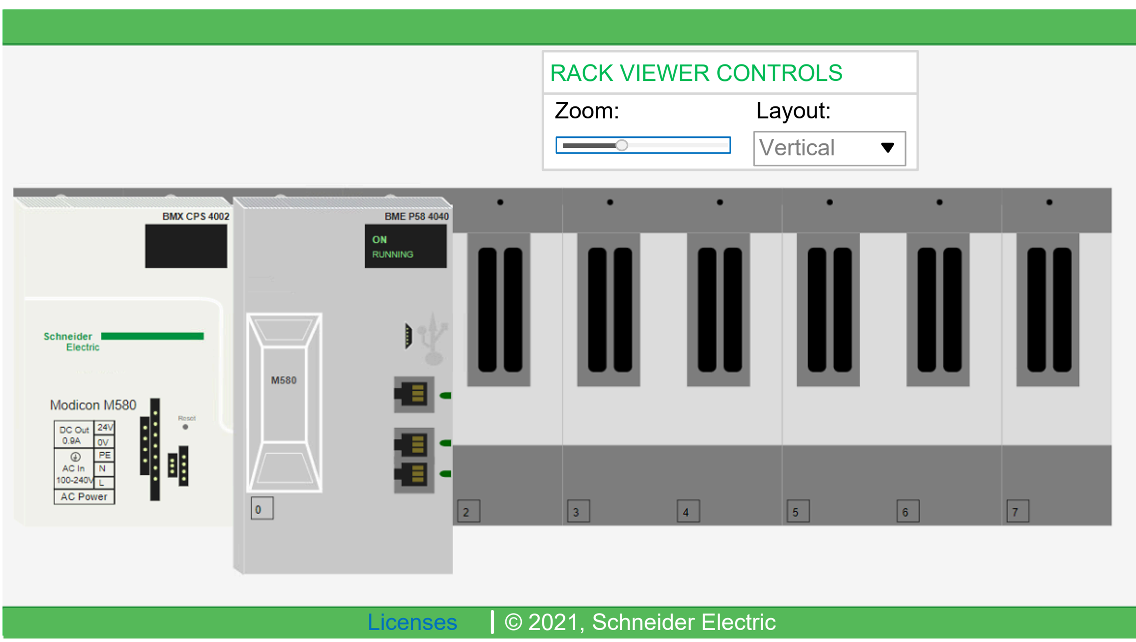
The BMEP584040, BMEP585040(C), and BMEP586040 standalone CPUs include a **Rack Viewer** web page. Access this page from the **Diagnostics** tab (**System > Rack Viewer**).

NOTE: You may have to wait a few seconds for the **Rack Viewer** to replicate your configuration.

Example

This example of a **Rack Viewer** page shows a standalone CPU on its rack with a power supply:

Rack Viewer page (Standalone CPU):



See also the example of the Hot Standby Rack Viewer page, page 455.

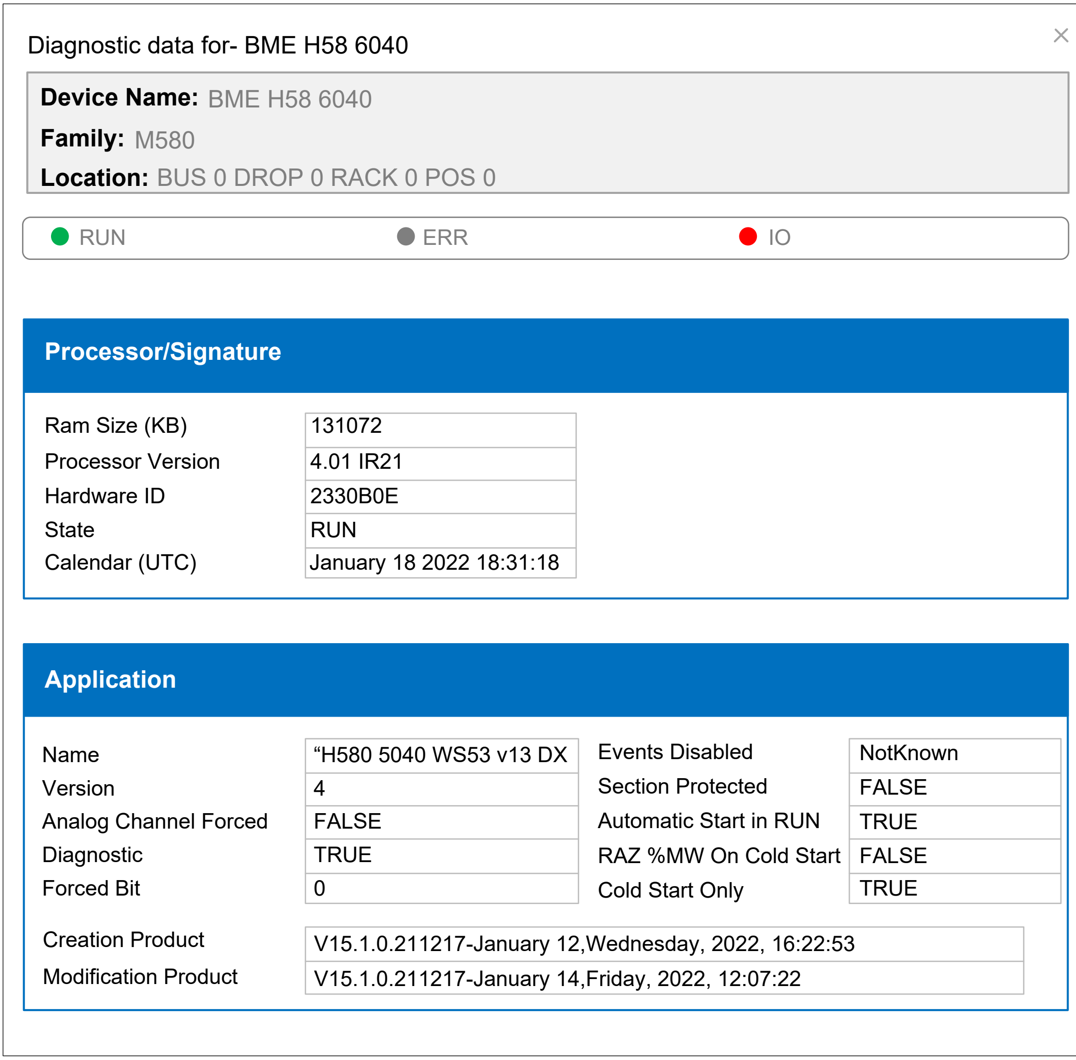
Information from This Page

The rack that appears in the top left of the **Rack Viewer** represents the local rack that contains the CPU.

Select navigation and view options in the **Rack Viewer** page:

Control	Selection	Description
Layout (menu)	**Horizontal**	Each RIO drop is shown in a top-to-bottom order beneath the primary bus. The lowest number RIO drop is at the top.
	Vertical	Each RIO drop is shown in a left-to-right order beneath the primary bus. The lowest number RIO drop is at the left.
Zoom (menu)	**Zooming**	Zoom in by sliding the control right. Zoom out by sliding the control left.

Double -click on any CPU in the **Rack Viewer** to see this information:



Refer to the Hot Standby Rack Viewer page, page 455 for a description of the fields shown above.

You can read this CPU data:

- CPU reference name
- bus, drop, rack, and slot location
- CPU state (**RUN**, **ERR**, and **I/O**)
- processor and network card information

- application name (on the CPU)

Data Storage

Open the Page

Access the **Data Storage** page from the **Diagnostics** tab (**Module > Data Storage**):

Use the **Data Storage** page to:

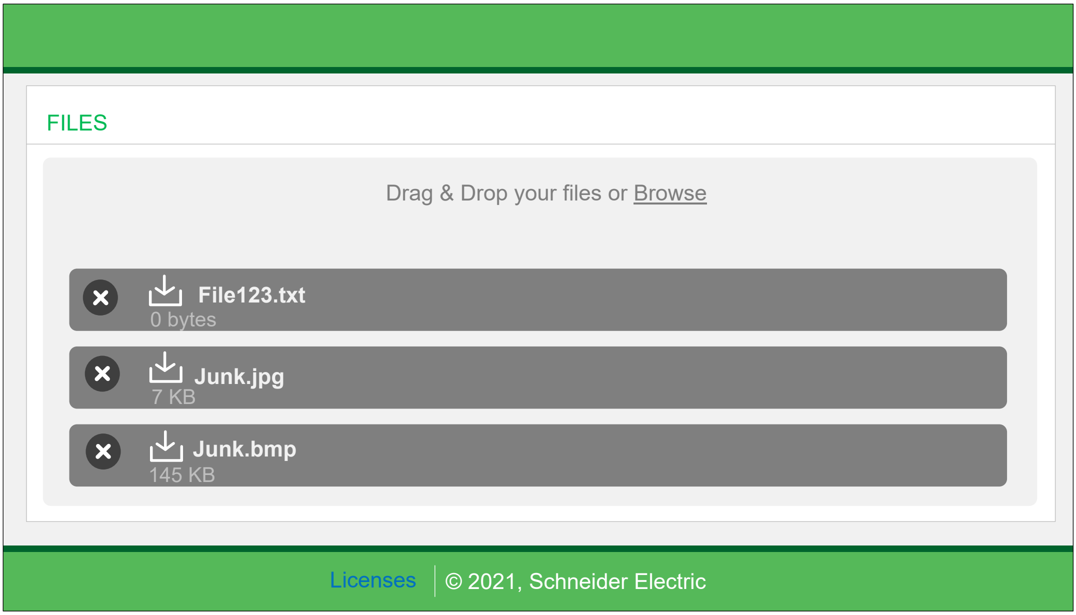
- Add (upload) files to an SD card inserted into the CPU.
- Transfer (download) files from an SD card inserted in the CPU to a specified location.
- Delete files that had been stored on an SD card inserted in the CPU.

NOTE:

- The maximum file size you can upload or download is 50 MB.
- This page is updated every 5 seconds.

Data Storage page:

When an SD card is inserted in the CPU, the **Data Storage** web page displays the files that are present on the SD card.



BMXRMS004GPF SD Memory Card

The Data Storage page supports the use of the BMXRMS004GPF SD memory card, page 78, which is specially formatted for use by the M580 CPUs:

- If you use this card with another CPU or tool, the card may not be recognized.
- If you re-format the card in another device – e.g., a camera – the card becomes incompatible for use by an M580 CPU. In this case, you need to return the card to Schneider Electric for re-formatting.

Adding, Transferring and Deleting SD Card Files

Adding a File to the SD Card

You can add (upload) files to the SD card in either of two ways:

- Drag and drop a file onto the Data Storage web page.

  Or...

- Click **Browse**, then in the **Open** dialog, navigate to and select a file, then click **Open**.

Transferring a File from the SD Card

To transfer (download) a file from the SD card, select the file to download, then click the downward pointing arrow next to the file name. The file is copied to the host PC **Downloads** folder.

Deleting a File from the SD Card

To delete a file from the SD card, select the file to delete, then click the button marked with an “X” next to the file name. The file is deleted from the SD card.

Supported File Types

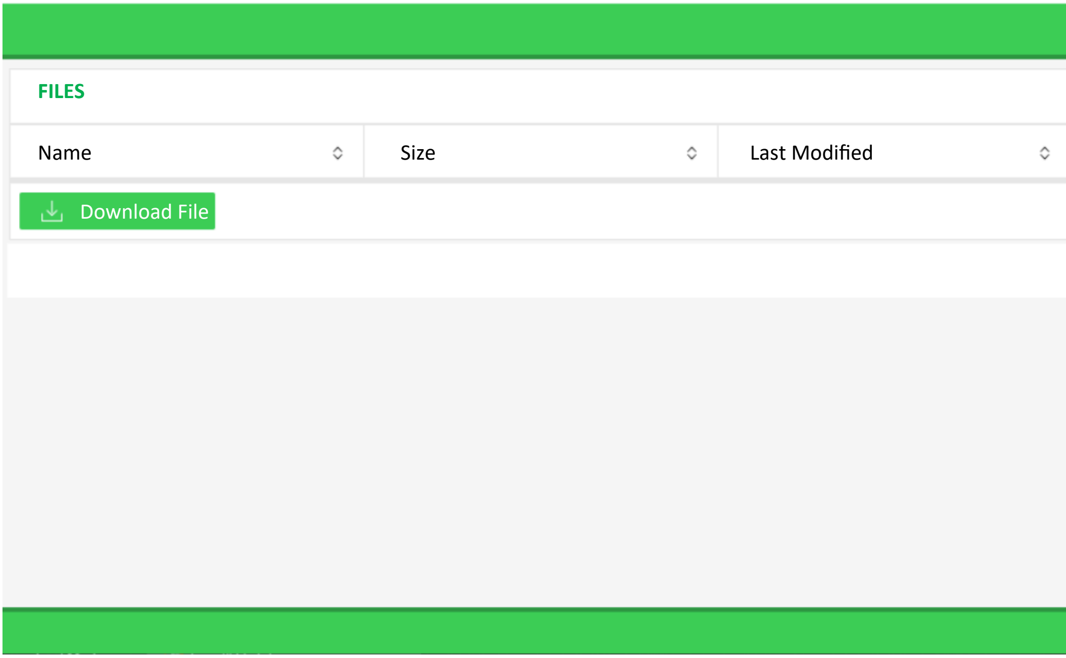
The Data Storage web page supports files of the following types (extensions):

Application File Types:

- Application File Types:
  - .eot
  - .js
  - .ttf
  - .woff
  - .wsdl
  - .xml
  - .xsd
- Image File Types:
  - .gif
  - .jpeg/.jpg
  - .png
  - .svg
- Text File Types:
  - .css
  - .htm/.html

Event Log

Use the event log page to save a log file of captured events:



To save an event log file:

1. Click **Download File**.
2. Enter the **File Name.**
3. Click **Start File Preparation**.

The file is prepared automatically. Upon completion:

- The new file is created in the host PC **Downloads** folder.
- The web page displays the **Name**, **Size** and **Last Modified** date of the new event log file.

M580 Hot Standby CPU Web Pages

Overview

This section describes the diagnostic web pages for the M580 BMEH58•040(S) Hot Standby CPU modules.

Introducing the M580 Hot Standby Controller Web Pages

Introduction

The M580 BMEH58•040(S) Hot Standby controllers includes an embedded web server that provide monitoring, diagnostic and file transfer functions.

These following web pages are common to both standalone and Hot Standby controllers:

- **Module**:
  - **Status Summary (Hot Standby)**, page 451
  - **HSBY Status**, page 454
  - **Performance**, page 423
  - **Port Statistics**, page 425
- **Connected Devices**:
  - **I/O Scanner**, page 427
  - Messaging, page 430
- **Services**:
  - **QoS**, page 431
  - **NTP**, page 433
  - **Redundancy**, page 438
- **System**:
  - **Alarm Viewer**, page 440
  - **Rack Viewer**, page 455
- **File Manager**:
  - **Data Storage**, page 446
  - **Event Log**, page 449

Browser Access Requirements

The embedded web pages are accessible using the following operating system and browser combinations:

Operating system	Browser
Android OS v4 mini	Chrome mobile minimum version 35.0.1916.141
iOS6	Safari v6
iOS7	
Windows 7	Internet Explorer v8.0.7601.17514
Windows 8	
Windows 8.1	
Windows 8.1 RT	Internet Explorer minimum v8
Windows Phone OS	Internet Explorer Mobile v10

The embedded web site is accessible via WiFi, using a smartphone or tablet equipped with a:

- Schneider Electric WiFi dongle, called the *wifer*, part number TCSEGWB13FA0.
- PMXNOW0300 wireless module.

Status Summary (Hot Standby CPUs)

Introduction

The **Status Summary** web page provides this information about the CPU:

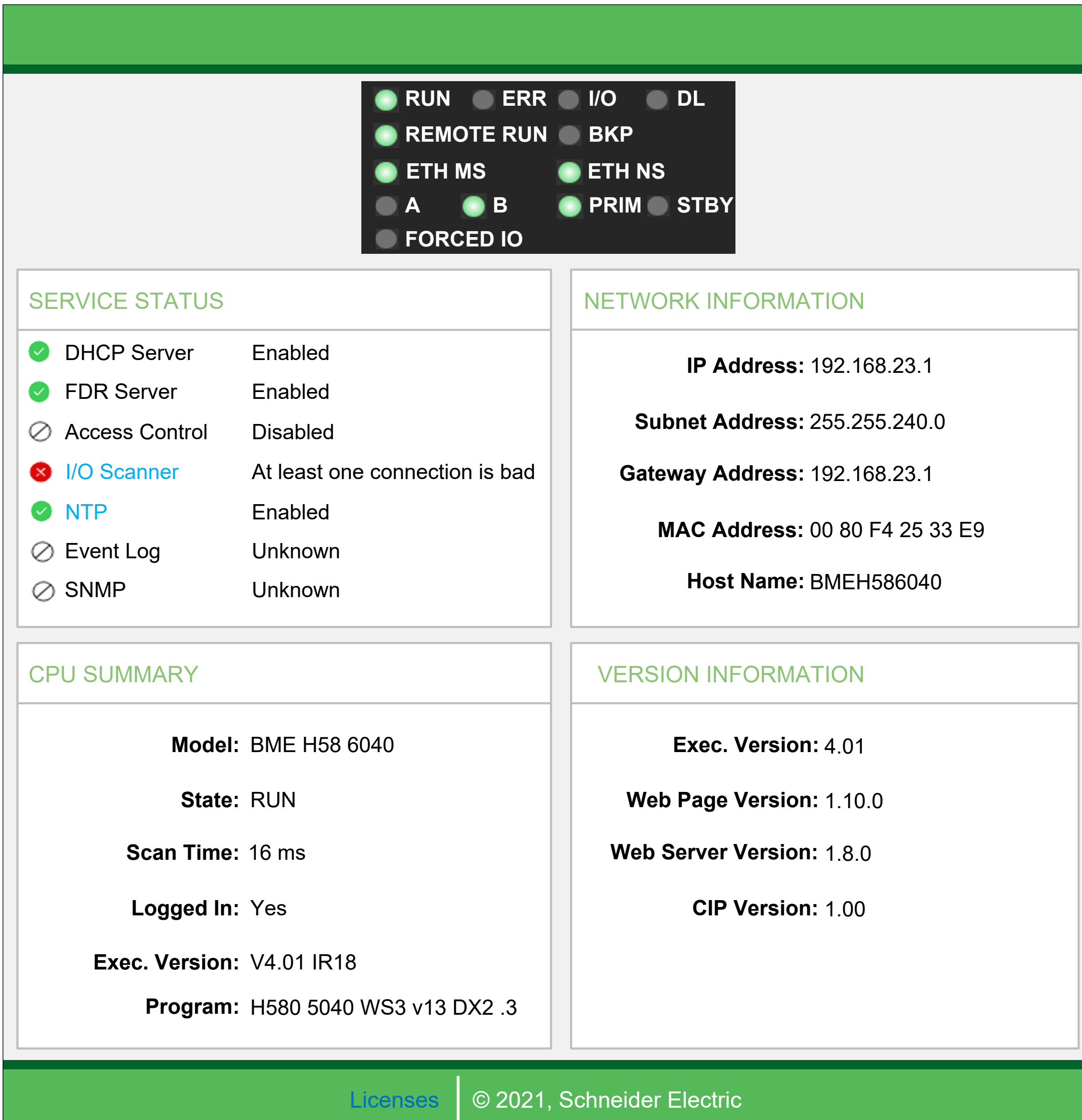
- Ethernet service diagnostic information
- Version descriptions for installed firmware and software
- CPU description and operating state
- IP addressing settings

NOTE: The **Status Summary** web page is refreshed every 5 seconds.

Open the Page

Access the **Status Summary** page on the **Diagnostics** tab (**Module > Status Summary**):

Status Summary page (Hot Standby CPU):



Diagnostic and Status Information

The **Status Summary** web page provides this information:

<table>
<tr><th>Parameters</th><th colspan="2">Description</th></tr>
<tr><td rowspan="3">LEDs</td><td colspan="2">The web page displays the state of these LEDs:</td></tr>
<tr><td>RUN
ERR
I/O
DL
REMOTE RUN
BKP
BKP
ETH NS</td><td>A
B
PRIM
STBY
FORCED_IO
SRUN (safety PAC)
SMOD (safety PAC)</td></tr>
<tr><td colspan="2">NOTE: The LEDs on the web page behave the same as the LEDs on the CPU, page 67.</td></tr>
<tr><td rowspan="6">Service Status</td><td colspan="2">This area presents information describing the status of CPU Ethernet services. The colored icons appearing to the left of some items indicate the following status:</td></tr>
<tr><td>green</td><td>The available service is operational and running.</td></tr>
<tr><td>red</td><td>An error is detected in an available service.</td></tr>
<tr><td>black</td><td>The available service is not present or not configured.</td></tr>
<tr><td colspan="2">The status of these Ethernet services is included:</td></tr>
<tr><td>DHCP Server
FDR Server
Access Control</td><td>Scanner Status
NTP Status
FDR Usage</td></tr>
<tr><td rowspan="2">Version Info.</td><td colspan="2">This area describes the software versions that are running on the CPU, including:</td></tr>
<tr><td>Executable Version
Web Server Version</td><td>Web Site Version
CIP Version</td></tr>
<tr><td>CPU Summary</td><td colspan="2">This area describes the CPU hardware and the applications that are running on the CPU, including:
Model
State
Scan Time</td></tr>
<tr><td>Network Info.</td><td colspan="2">This field contains IP addressing settings for the CPU, including:
IP Address
Subnet Address
Gateway Address</td></tr>
</table>

HSBY Status

Introduction

The **HSBY Status** web page provides this information about the Hot Standby system:

- Hot Standby role and status of the **Local** CPU
- Hot Standby role and status of the **Remote** CPU
- General errors detected for the Hot Standby system

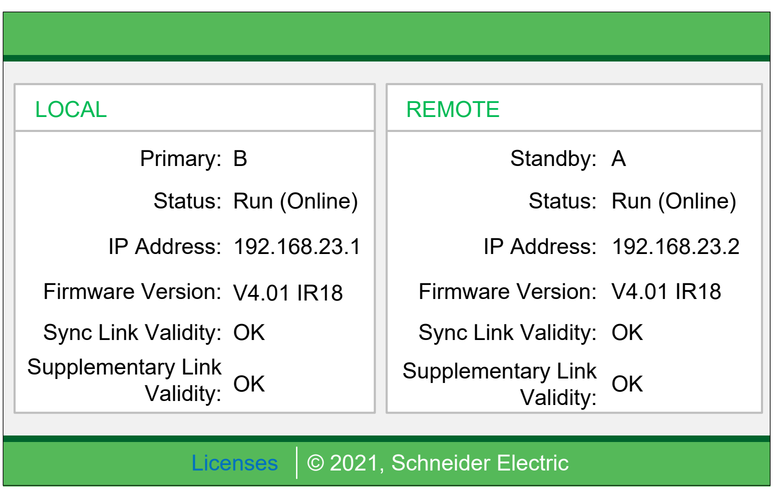
NOTE:

- The local CPU is the CPU configured with the **Main IP Address** (primary) or **Main IP Address + 1** (standby) used to access this web page.
- The **HSBY Status** web page is refreshed every 5 seconds.

Open the Page

Access the **HSBY Status** page from the **Diagnostics** tab (**Module > HSBY Status**):

HSBY Status page:



Diagnostic and Status Information

The **HSBY Status** web page provides this information:

Area	Description	
Local/Remote	This area displays the state of Hot Standby settings for the local and remote CPUs:	
	<Hot Standby Role>	The Hot Standby system role of the CPU. Valid values include: • Primary • Standby • Wait
	<A/B switch setting>	The designation of the CPU, defined by the rotary switch, page 58 on the back of the CPU. Valid values include: • A • B
	Status	The operating state of the CPU. Valid values include: • RUN • STOP • NoConf • HALT
	IP Address	The IP address used to communicate with the CPU for web page access: • For the primary Hot Standby CPU, this is the **Main IP Address** setting. • For the standby Hot Standby CPU, this is the **Main IP Address** setting + 1.
	Firmware Version	Firmware version of the CPU operating system.
	Sync Link Validity	The status of the Hot Standby link (see Modicon M580 Hot Standby, System Planning Guide for, Frequently Used Architectures): • OK: the link is operational. • NOK: the link is not operational.
	Supplementary Link Validity	The status of the Ethernet RIO link (see Modicon M580 Hot Standby, System Planning Guide for, Frequently Used Architectures): • OK: the link is operational. • NOK: the link is not operational.

Rack Viewer

Introducing the CPU Status Page

The BMEH584040(S) and BMEH586040(C)(S) Hot Standby CPUs include a **Rack Viewer** web page. Use this page to view CPU information, including:

- LEDs status
- processor identification
- application signature identification
- select application configuration settings

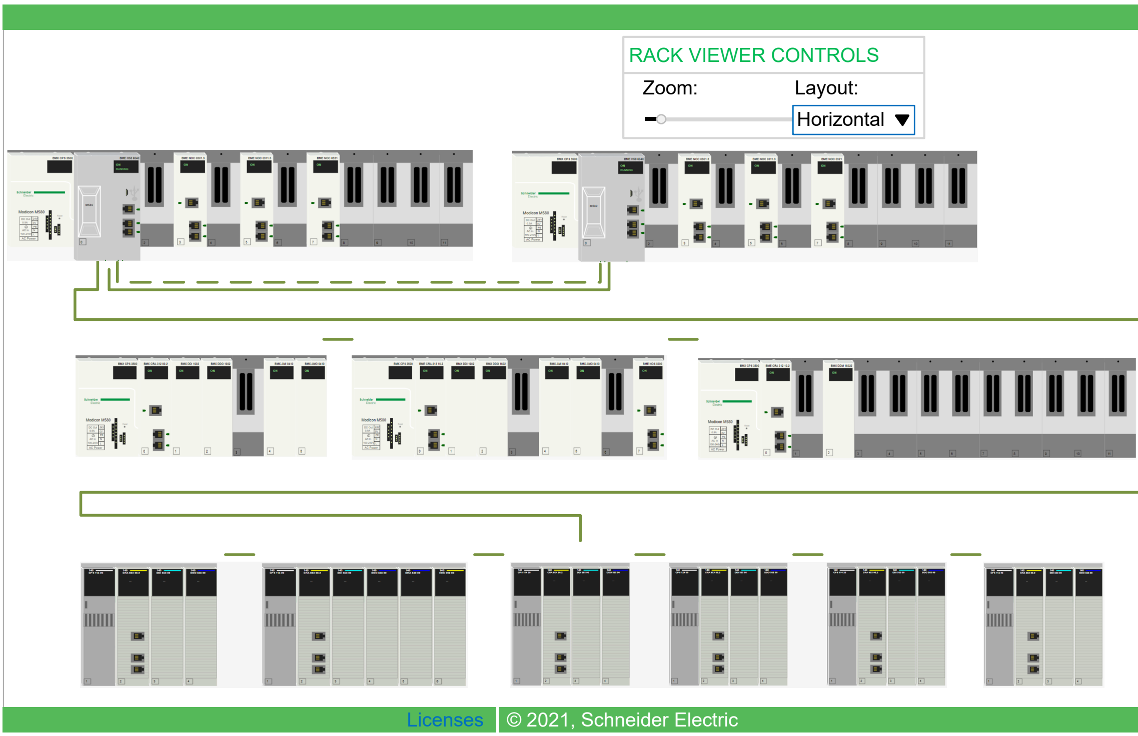
Access this page from the **Diagnostics** tab (**System > Rack Viewer**).

This example of a **Rack Viewer** page shows a Hot Standby CPU on its rack with a power supply:

Accessing the Rack Viewer Page

Access the **Rack Viewer** page from the **Diagnostics** menu. In the navigation menu at the left side of the page, select **Menu > System > Rack Viewer**:

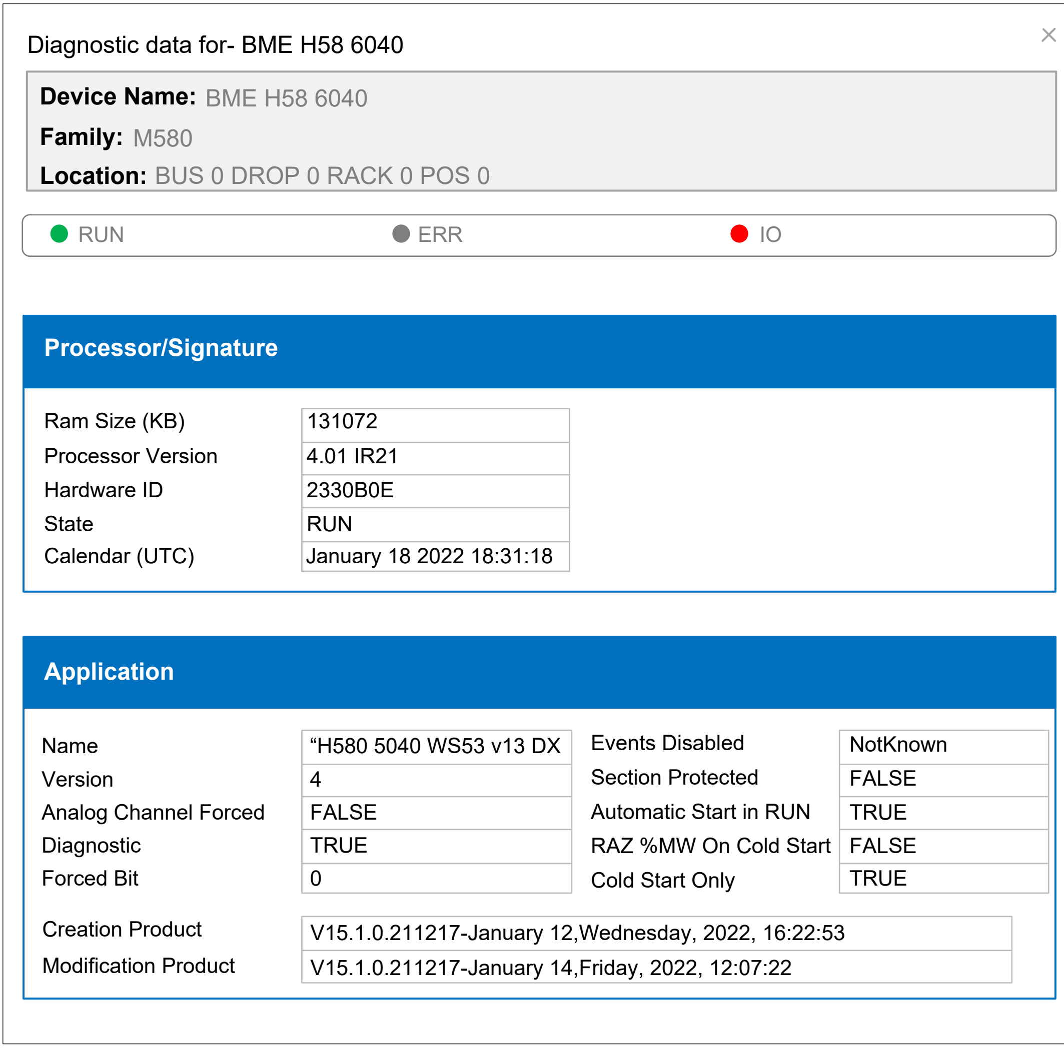
Rack Viewer page (HSBY CPU):



This example of a Rack Viewer page shows the Hot Standby connection between a primary CPU rack and a standby CPU rack. The Hot Standby connection (dashed line) is green when the Hot Standby link is healthy. If the Hot Standby link is not healthy, the dashed line is red.

Rack Viewer Data

Double-click on the **Rack Viewer** page to display Hot Standby CPU data.



Data Field	Description
Processor/Signature	
RAM size (kb)	The size of processor RAM in KB
Processor Version	Firmware version
Hardware ID	An identifier for the module hardware. OS Loader checks this value to determine compatibility between the hardware and the operating system.
State	The operating state of the processor: • NO CONFIGURATION

Data Field	Description
	• IDLE • STOP • RUN • HALT • INITIALIZING • ERROR • OS LOADER
Error	The identity of the last detected error
Calendar (UTC)	Date and time of last detected error
Application	
Name	Name of the Control Expert project
Version	Project version
Analog channel forced:	Indicates if one or more inputs or outputs for an analog channel have been forced: • **True** indicates the an analog input or output has been forced. • **False** indicates no analog input or output has been forced.
Diagnostic	Indicates if the diagnostic buffer has been activated for the project: • **True** indicates that **Application diagnostics** and/or **System diagnostics** has been selected in the **General > PAC Diagnostics** tab of the **Project Settings** dialog for the application. • **False** indicates **Application diagnostics** and **System diagnostics** have not been selected.
Forced bit	The number of forced bits in the application.
Creation Product	Includes both: • Version and build of Control Expert used to create the project. • Date and time the project was created.
Modification Product	Includes both: • Version and build of Control Expert used to edit the project. • Date and time the project was last edited.
Events Disabled	Indicates if all event processing has been disabled: • **True** indicates all event processing has been disabled. • **False** indicates event processing has not been disabled. **NOTE:** Events can be enabled/disabled by using: • The **Enable or Disable all** command (see EcoStruxure™ Control Expert, Operating Modes) in the **Task** tab of the CPU. • The `MASKEVT` and `UNMASKEVT` functions. • System bit %S38.

Data Field	Description
Section protected	Indicates if password access is required to edit one or more sections of the application: • **True** indicates that a password is required to edit specified sections of the application. • **False** indicates that no password is required for application editing.
Automatic Start in Run	Indicates if the application is automatically set to start when the PAC goes into RUN operational mode: • **True** indicates the application automatically starts. • **False** indicates the application does not automatically start.
RAZ %MW on cold start	Indicates if %MW registers are reset to their initial values on a cold start: • **True** indicates that values are reset. • **False** indicates that values are not reset.
Cold Start only	Indicates if a cold start is forced on a system re-start: • **True** indicates that a reset forces a cold start of the application. • **False** indicates that a warm start will occur on application reset.
Creation Product	Includes both: • Version and build of Control Expert used to edit the project. • Date and time the project was created.
Modification Product	Includes both: • Version and build of Control Expert used to edit the project. • Date and time the project was last edited.

Working with M580 Hot Standby Applications

What's in This Chapter

Overview

This chapter shows you how to configure and work with Hot Standby applications.

Configuration Compatibility

Control Expert Version Requirement

An M580 non-safety-related Hot Standby system can be configured using Control Expert L or XL version 11.0 or any subsequent supporting version(s). By contrast, an M580 safety Hot Standby system can be configured using only Control Expert XL Safety version 14.0 or any subsequent supporting version(s).

PAC Hardware

Confirm that the primary PAC and the standby PAC consist of compatible hardware, including:

- controller

- backplane
- Power supply
- Some communication modules

  NOTE: No I/O modules can be mounted onto the local backplane. Refer to the topic The *Modicon M580 Hot Standby Local Backplane* in the *M580 High Availability System Planning Guide* for a description of modules that can be added to the local backplane.

Controller Compatibility

An application created for a specific controller may not be compatible with other controllers. The M580 Hot Standby system compares the applications in the primary controller against the application in the standby controller to determine if the applications are compatible.

NOTE: An application created for a non-safety-related controller cannot be run on a safety controller, and an application created for a safety controller cannot be run on a non-safety-related controller.

For example:

- A Quantum 140CPU67•6• controller Hot Standby application is not downloadable to M580 BMEH58•040 Hot Standby controllers.
- An M580 BMEP58•0•0 controller application is not downloadable to M580 BMEH58•040 Hot Standby controllers.
- As described in the following table, an application designed for one M580 BMEH58•040 Hot Standby controller may not be downloadable to other M580 Hot Standby controllers.

The following table depicts the compatibility of applications among non-safety-related M580 Hot Standby controllers:

An application built for:	**Can be downloaded to and executed by the following controllers:**		
	BMEH582040	**BMEH584040**	**BMEH586040**
BMEH582040	X	X	X
BMEH584040	–	X	X
BMEH586040	–	–	X
X: Can receive and execute the application. –: Cannot receive and execute the application.			

The following table depicts the compatibility of applications among M580 safety controllers:

An application built for:	Can be downloaded to and executed by the following controllers:				
	BMEP582040S	BMEP584040S	BMEH582040S	BMEH584040S	BMEH586040S
BMEP582040S	1	2	2	4	4
BMEP584040S	3	1	3	4	4
BMEH582040S	2	2	1	2	2
BMEH584040S	3	2	3	1	2
BMEH586040S	3	2	3	3	1

1. Fully compatible.
2. Compatible, if controller is upgraded in Control Expert and the application is fully rebuilt.
3. Compatible, if controller is upgraded in Control Expert and the application is fully rebuilt, and there is no limitation as to memory size.
4. Compatible only for application with no CIP Safety devices, if controller is upgraded in Control Expert and the application is fully rebuilt.

Controller Firmware Mismatch

An M580 Hot Standby system can continue operating when there is a mismatch of firmware versions in the primary and standby controllers, if each controller firmware can execute the application. This makes it possible to upgrade (or downgrade) controller firmware without having to stop the operation of the Hot Standby system. To permit Hot Standby operations to continue in this case, use an animation table or program logic to set the `FW_Mismatch_Allowed` attribute of the `T_M_ECPU_HSBY`, page 498 to **True**.

Application Mismatch

An M580 Hot Standby system cannot operate if the primary and standby controllers are equipped with fundamentally different applications. In this case, the primary PAC operates as a standalone PAC, and the standby PAC enters the stop state.

To restore Hot Standby system operations, confirm that the same application is installed in both the primary and standby PACs.

Logic Mismatch

An M580 Hot Standby system can continue operating if the primary and standby controllers are running different revisions of the same application. In this case, both controllers were initially configured with the same application, but the logic in one controller – usually the primary controller – was subsequently revised.

For Hot Standby operations to continue when a logic mismatch exists, use an animation table or program logic to set the `Logic_Mismatch_Allowed` attribute of the `T_M_ECPU_HSBY`, page 498 DDT to **True**.

For Hot Standby operations to continue when a logic mismatch exists, do both of the following:

- Select **Online modification in RUN or STOP** in the **Configuration** tab of the controller.
- Set the **Number of modifications** in the **Configuration** tab of the controller.
- Use an animation table or program logic to set the `Logic_Mismatch_Allowed` attribute of the `T_M_ECPU_HSBY`, page 498 DDT to **True**.

NOTE: If the **Number of modifications** is set to 0, setting the `Logic_Mismatch_Allowed` attribute has no effect.

SFC Mismatch

A sequential function chart (SFC) mismatch occurs when the applications in the primary and standby controllers include graphic symbols that define sequential program steps, where differences exists in at least one SFC section.

Refer to the topic *Modifying an SFC Section Online*, page 478 for the procedure for making online modifications to an SFC section.

Modicon M580 Hot Standby Programming Rules

At a Glance

For Modicon M580 Hot Standby applications, some of the programming functionality you may have used does not apply to redundant operations. This section summarizes some of the code features and programming rules of a Modicon M580 Hot Standby application.

Error Correcting Code (ECC) Feature

M580 Hot Standby controllers with firmware version 2.50 and higher include an error correcting code (ECC) feature. ECC enhances reliability by reducing the likelihood of memory random access errors, when a Hot Standby controller accesses its internal memory, as part of a memory transfer event. The ECC function is enabled by default.

When ECC is enabled, it may impact the MAST cycle time of Hot Standby M580 PAC applications. This can be the case where a relatively small amount of code is transferred, but a large amount of data is transferred. If the impact on MAST cycle time is not suitable for your application, you can:

- Reduce the amount of exchanged data from the primary to the standby controller.
- For a non-safety-related controller application, disable the ECC feature using %SW150 (see EcoStruxure™ Control Expert, System Bits and Words, Reference Manual).

Changing Declared Variables

Using the save operation, which is invoked with the %S94 system bit, on the primary controller does not also apply to the standby controller.

If a swap or switchover occurs after a CCOTF has been performed on the primary controller and the application has not been transferred to the standby controller, then the behavior of the application is unpredictable.

The changes to declared variable values are not part of the database transfer, and can lead to unintended consequences at switchover.

⚠ WARNING

UNINTENDED EQUIPMENT OPERATION

In a Hot Standby system, do not overwrite the initial values for declared variables using the save operation invoked with the %S94 system bit.

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

Section Executed on Standby Restrictions

The following restrictions apply to sections executed on the Standby PLC, first section, or all sections depending on the configuration, page 471:

- Derived Function Blocks (DFB) may not be executed on Standby PLC sections.
- **R_TRIG, F_TRIG, TRIGGER, TON, TOF, TP** functions blocks may not be executed on Standby PLC sections.
- Asynchronous communication procedures may not be executed on Standby PLC sections.
- Asynchronous communication function blocks may not be executed on Standby PLC sections.

Asynchronous Communication Procedures

During a switchover event, asynchronous communication procedures: **READ_VAR, WRITE_VAR, DATA_EXCH, INPUT_CHAR, INPUT_BYTE, PRINT_CHAR,** do not automatically resume operation on the new Primary PLC without special care.

The following procedure should be used to allow asynchronous communication EFs to automatically resume operation after a switchover:

- Program your application so that all EFs management parameters are not exchanged with Standby PLC. To do this, de-select the **Exchange on STBY** attribute for the management parameter.
- Initialize the Length parameter each time the function is called.
- Set the Timeout parameter accordingly to your application:
  - If the communication function is send through the controller, the typical timeout value is 500 ms.
  - If the communication function is send through a NOC module, the typical timeout value is 2 s.

NOTE: If for some reason you are unable to follow this procedure, and a switchover renders your communication function inoperative, write your application program so that it sets the function activity bit to 0 before restarting the function in the new Primary controller.

Asynchronous Communication Function Blocks

During a switchover event, asynchronous communication function blocks, which use internal management parameters: **GET_TS_EVT_M, READ_DDT, READ_PARAM_MX, READ_STS_MX, RESTORE_PARAM_MX, SAVE_PARAM_MX, WRITE_CMD_MX, WRITE_PARAM_MX, MBP_MSTR, READ_SDO, WRITE_SDO, ETH_PORT_CTRL, PWS_DIAG, PWS_CMD, L9_MSTR,** do not automatically resume operation on the new Primary PLC without special care.

The following procedure should be used to allow asynchronous communication EFBs to automatically resume operation after a switchover:

- Program your application so that all EFBs instances are not exchanged with Standby PLC. To do this, de-select the **Exchange on STBY** attribute for the EFB instance.

Other Functions

While the use of the functions listed above is restricted, you are advised to use care even when employing permitted functions that are capable of writing to memory areas that are not part of the Hot Standby database transfer, such as Data Storage, page 505 function blocks for instance.

Debugging

Debugging your Hot Standby application program is now a two-stage process:

- First, you debug the application on a single Hot Standby PLC as if it was a standalone application. This allows you to use all of the powerful debugging features available in Control Expert, such as watchpoints, and so on.
- Next, you debug your application when it has been uploaded to two Hot Standby PLCs in a working redundant system, but in a non-production environment. On this platform, you evaluate performance specific to Hot Standby redundancy. Only a subset of Control Expert debug features can be used during this stage.

NOTE: See M580 Hot Standby Diagnostics, page 548 for further details on debugging your Hot Standby application program.

PME UCM 0202 Universal Communication Module

Do not use a **PME UCM 0202** Universal Communication Module in a Drop of a Modicon M580 Hot Standby configuration.

M580 Hot Standby System Configuration

Control Expert Configuration Tool

The exclusive configuration tool for an M580(S) Hot Standby system is:

- Version 11.0 and any subsequent supporting version(s) of Unity Pro L (for the BMEH582040 module).

  NOTE:

  Unity Pro is the former name of Control Expert for version 13.1 or earlier.

- Version 11.0 and any subsequent supporting version(s) of Unity Pro XL (for the BMEH584040 and BMEH586040 modules).
- Version 14.0 and any subsequent supporting version(s) of Control Expert XL Safety (for the BMEH582040S, BMEH584040S, and BMEH586040S.

Programming Application Languages and Libraries

Control Expert supports the following application languages and libraries for the M580 Hot Standby controllers:

Application language / library	Non-safety-related controllers		Safety controllers			
	BMEH58...		BMEH58...			
	2040	4040, 6040	2040S		4040S, 6040S	
			SAFE task	FAST, MAST tasks	SAFE task	FAST, MAST tasks
Function Block Diagram (FBD)	X	X	X	X	X	X
Ladder Diagram (LD)	X	X	X	X	X	X
Structured Text (ST)	X	X	–	X	–	X
Instruction List (IL)	X	X	–	X	–	X
Sequential Function Chart (SFC)	X	X	–	X	–	X
Derived Function Block (DFB)	X	X	X	X	X	X
Elementary Function (EF)	X	X	X[1]	X	X[1]	X
Elementary Function Block (EFB)	X	X	X[1]	X	X[1]	X
Ladder Logic 984 (LL984)	–	X	–	–	–	X

Application language / library	Non-safety-related controllers		Safety controllers			
	BMEH58...		BMEH58...			
	2040	4040, 6040	2040S		4040S, 6040S	
			SAFE task	FAST, MAST tasks	SAFE task	FAST, MAST tasks
PL7 - Standard Function Block (SFB)	–	–	–	–	–	–
X: Supported –: Not supported 1: EF/EFB prefixed with "S_"						

Configuring an M580 Hot Standby Controller

Introduction

This topic shows you how to configure the Hot Standby functionality of an M580 BMEH58•040 controller. For information on how to configure the non-Hot Standby functions for the controller, refer to Introducing the M580 Hot Standby Controller Web Pages, page 450

NOTE: The same procedure, as described below, can also be applied to the configuration of an M580 BMEH58•040S safety controller.

Accessing the M580 Controller Hot Standby Configuration Tab

Use the **Hot Standby** tab of an M580 BMEH58•040 controller to configure its Hot Standby function. To access this tab:

Step	Action
1	Add a BMEH58•040 controller to your project.
2	In the **Project Browser**, select **Configuration > PLC Bus > <rack> > <controller>**.
3	Right-click the controller and select **Open**.
4	Click the **Hot Standby** tab.

Configuring the Hot Standby Function

The **Hot Standby** tab presents the following configurable settings:

<table>
<tr><th colspan="2">Setting</th><th>Description</th></tr>
<tr><td rowspan="2">Run Mode</td><td>Controller A Online</td><td rowspan="2">Specify if controller A and controller B operate online at the next start-up:
TRUE (default): The controller attempts to operate online at next start-up. Depending on the other conditions, the controller may act as the primary or standby.
FALSE: The controller transitions to either the Wait or Stop state at next start-up.</td></tr>
<tr><td>Controller B Online</td></tr>
<tr><td>Standby On Logic Mismatch</td><td>Number of modifications</td><td>The maximum number of online build changes from 1...50 that can be performed on the primary controller. When this number of online build changes has been reached, you need to transfer the application from the primary to the standby to be able to make additional online build changes. Default = 20.
NOTE:
If this setting is set to 0, the <code>Logic Mismatch Allowed, page 498</code> flag has no effect.
This setting cannot be edited via CCOTF.</td></tr>
<tr><td>Behavior of the Controller in Wait and Standby mode</td><td>controller executes</td><td>Specify the sections of the MAST task the standby controller executes in Wait state:
All sections (default)
First section
No section at all
When Control Expert is connected to the standby controller, all Sections in the Project Browser are preceded by:
a green light for sections without condition or with a TRUE condition even if not executed
a red light for sections with a FALSE condition
NOTE:
You can also individually specify the sections of the MAST task the standby controller executes while in Wait state. Do this by adding a condition of execution in the Condition tab of the Properties window for a MAST task section.
For a safety controller, sections of the SAFE task are not executed when the PAC is in WAIT or STANDBY state.
You can also individually specify the sections of the MAST task the standby controller executes while in Wait state. Do this by adding a condition of execution in the Condition tab of the Properties window for a MAST task section.</td></tr>
<tr><td>Data Exchanged</td><td>–</td><td>A bar graph displays the percentage of controller memory used by Hot Standby data. The value depends on the M580 Hot Standby configuration.

The total data exchanged is displayed in KB as well as:
data exchange by MAST
data exchange by FAST
data exchanged by SAFE (for a safety controller)</td></tr>
</table>

Configuring Controller Online State

Controller A is the controller with the A/B/Clear rotary selector switch, page 58 (located on the back of the controller) set to A. Controller B is the controller with the A/B/Clear rotary selector switch set to B.

You can use the **Controller A Online** and **Controller B Online** settings, for the following purposes:

- To specify the controller that will be primary on a cold start. For example, set **Controller A Online** to **True** and **Controller B Online** to **False**. Controller A powers up as primary, and controller B powers up in wait state. After power up, you can manually set **Controller B Online** to **True**.
- To avoid an unintended switchover. For example, if controller A is primary and controller B is standby, set **Controller B Online** to **False**. Controller B enters wait state, and no switchover can occur.

These settings can be modified during runtime, or when the Hot Standby system is not operating.

Settings entered when the Hot Standby system is not running take effect after the next project build, when the Hot Standby system next starts-up.

If the Change Configuration on The Fly (CCOTF) function is enabled, settings entered when the Hot Standby system is running take effect on the next project build (or re-build).

No Local I/O Configuration

Because the local rack of a Hot Standby controller cannot include I/O modules, the following settings in a BMEH58•040 or BMEH58•040S controller **Configuration** tab are disabled:

- **Run/Stop input**
- **Run/Stop by input only**
- **Memory protect**
- **Maintenance Input** (safety PAC)

  NOTE: Instead of using the **Run/Stop input**, consider using the following approach to controlling the RUN/STOP operating state of a safety controller:

  - Use a BMENOC0301 or BMENOC0311 communication module and the IPsec protocol to help provide a secure connection to the controller.
  - Then use the `CMD_RUN_REMOTE` or `CMD_STOP_REMOTE` commands of the `T_M_ECPU_HSBY` DDT to change a remote controller operating state.

Enabling FDR Server Synchronization in a Hot Standby System

In an M580 Hot Standby system, a BMEH58•040 controller or a BMENOC0311 or a BMENOC0301 Ethernet communication module can perform the role of an FDR server. To permit the synchronization of the FDR server in the primary controller with the FDR server in the standby controller, you need to enable the TFTP service for the Hot Standby system.

To enable the TFTP service, follow these steps:

Step	Action
1	In the **Project Browser** double-click on the following: **Project > Configuration > 0:PLC bus > <rack> > <controller> > EIO**. The **RIO DIO Communicator Head** window opens.
2	Click the **Security** tab.
3	For the **TFTP** service, select **Enabled**.
4	If **Access Control** is enabled, create an entry for each device or subnet that you want to have TFTP access to the controller. **NOTE:** Select the **TFTP** column for each entry.
5	**Validate** and **Save** your edits.

NOTE: The FDR server cannot synchronize the primary and standby controllers when the TFTP service is disabled. The TFTP service is enabled and disabled by the execution of the `EthPort_Control_MX` function in the application.

If you want to programmatically enable or disable TFTP, include the `EthPort_Control_MX` function in a section of the application that is executed by the standby controller, so that this function is executed by both the primary and standby controllers.

Change Configuration On The Fly (CCOTF)

CCOTF Rules for Hot Standby

All M580 BMEH58•040 and BMEH58•040S controllers support CCOTF. CCOTF is enabled in the **Configuration** tab of the controller, in the **Configuration Online Modification** area, by selecting **Online modification in RUN or STOP**.

For information about CCOTF for M580 safety controllers, refer to the *Modicon M580 Safety Manual* (see Modicon M580, Safety Manual).

If a swap or switchover occurs after a CCOTF has been performed on the primary controller, and the application has not been transferred to the standby controller, then the behavior of the application is unpredictable.

⚠ WARNING

UNINTENDED EQUIPMENT OPERATION

- Before starting a CCOTF operation, verify that the application running in the Hot Standby system does not trigger a swap and that no condition exists that could forseeably cause a switchover.
- Always apply a CCOTF transaction on the primary controller.

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

NOTE: To download CCOTF changes to a Hot Standby system:

- Always apply a CCOTF transaction to the primary controller.
- Confirm the Hot Standby system is operational with a healthy Hot Standby link between the two controllers.
- Confirm that the impacted Ethernet RIO drop is operational, with a healthy Ethernet RIO link.

CCOTF allows modifications of a Hot Standby primary controller configuration in RUN mode. The changes that can be made in the primary controller are as follows:

- Add a discrete or analog module in a free slot.
- Delete a discrete or analog module.
- Modify the configuration and adjustment parameters of a module.

The changes that can be made in an Ethernet RIO drop are as follows:

- Add an (e)X80 or Quantum RIO drop.
- Add a discrete or analog module in a free slot.

- Delete a discrete or analog module.
- Modify the configuration and adjustment parameters of a module.

Any CCOTF changes made to the primary controller configuration are not automatically transmitted to the standby controller. Instead, the standby controller continues to be configured with its original application program.

CCOTF does not support all changes to the configuration. The following rules apply to CCOTF changes made to the primary Hot Standby controller configuration:

- A single CCOTF change can include multiple edits to multiple configuration objects.
- Edits to configuration objects are atomic: only one change can be made to a single configuration object. For example, you cannot add then delete the same I/O module in a single CCOTF change.
- CCOTF edits cannot be made to distributed equipment.
- For an (e)X80 or Quantum RIO drop, the following limits apply to changes made in the same CCOTF session:
  - Up to four modifications to the same RIO drop can be included in a single CCOTF change. For example:

    - Up to four I/O modules can be added to the same RIO drop.

    - Up to four I/O modules can be removed from the same RIO drop.

    - Up to four parameters can be edited for one I/O module in the same RIO drop.
  - No edits can be made to an adapter module.
  - No edits can be made to BMXERT1604 modules (time stamp).
  - The RPI setting for the RIO drop cannot be changed.
- IP addresses cannot be changed.
- Only one CCOTF change may be made to a single RIO drop. Before an additional CCOTF change can be made to the same RIO drop, transfer the application program from the primary controller to the standby controller.

  NOTE: You can set Control Expert to **Virtual connected mode** to test whether a proposed change to the configuration is a CCOTF event (see Modicon M580, Change Configuration on the Fly, User Guide).

When CCOTF changes are made to the primary controller, the `Logic_Mismatch_Allowed` flag in the `T_M_ECPU_HSBY` DDT determines if the standby controller can continue to operate online. If logic mismatches are not allowed, the standby controller transitions to wait state.

CCOTF changes can be made to the primary controller if the **Number of modifications** setting in Control Expert is not reached. When the number of allowed modifications is reached:

- No additional CCOTF changes can be made to the primary controller. The **Build > Build Changes** command in Control Expert is disabled.

- You need to transfer the application program in the primary controller to the standby controller, page 485.

Modifying an SFC Section Online

Precautions for Modifying an SFC Section Online

When the M580 Hot Standby system executes a switchover or a swap, the new primary controller tests the `SFC_MISMATCH` bit. The `SFC_MISMATCH` bit is set when the structure of at least one SFC section in the primary controller differs from that section in the standby controller. If this bit is set, the controller re-initializes the state-machine of all the modified SFC sections to help prevent any unpredictable behavior of the user application.

⚠ WARNING

UNINTENDED EQUIPMENT OPERATION

- Transfer the application from primary controller to the standby controller after each online modification of a MAST task section that is programmed using the sequential function chart (SFC) programming language.
- Do not execute a switchover or trigger a swap before this transfer is successfully completed.

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

To avoid the re-initialization of the SFC state-machines when you modify an SFC section, follow these steps:

Step	Action
1	Confirm that the `LOGIC_MISMATCH_ALLOWED` bit is set to 1. **NOTE:** If logic mismatch is not allowed, the standby controller enters wait after step 3.
2	Make the online edit to the SFC section in Control Expert.
3	Build the online change in Control Expert by selecting **Build > Build Changes**. The modification is made to the program running in the primary controller.

4	Transfer the application from the primary controller to the standby controller. Use a Control Expert animation table to set the `CMD_BACKUP_APPLI_TRANSFER` bit to 1.
	NOTE: Alternatively, you can automate the transfer in program logic using a code sequence like the following:
	`if (ECPU_HSBY_1->SFC_MISMATCH = 1)`
	`then ECPU_HSBY_1-->CMD_BACKUP_APPLI_TRANSFER = 1`

Configuring IP Addresses for an M580 Hot Standby System

Introduction

This topic shows you how to assign IP addresses to an M580 Hot Standby system. For information on how to configure other Ethernet communication settings for the controller, refer to the *M580 Hardware Reference Manual* (see Modicon M580, Hardware, Reference Manual).

Accessing the M580 Controller Hot Standby Animation task Tab

Use the **IPConfig** tab of the **EIO** configuration window for an M580 BMEH58•040 or BMEH58•040S controller to assign IP addresses. To access this tab:

Step	Action
1	Add a BMEH58•040 or BMEH58•040S controller to your project.
2	In the **Project Browser**, navigate to and select **Configuration > PLC Bus > <rack> > <CPU> > EIO**.
3	Click the right mouse button, then select **Open**.
4	Click the **IPConfig** tab.

Assigning IP Addresses to Modicon M580 BMEH58•040 or BMEH58•040S CPUs

An M580 Hot Standby system requires the assignment of three IP addresses. In addition, Control Expert automatically creates and assigns a fourth IP address. IP address settings include:

IP address name	Description
Main IP address	The configurable IPv4 IP address used by the primary controller for communication with distributed equipment. **NOTE:** Because this setting is always assigned to the primary controller, it can be associated with either the A or B controller. When a switchover occurs (for example, when controller B becomes primary) the main IP address assignment is transferred from controller A to controller B.
Main IP address + 1	The Control Expert auto-generated IPv4 IP address used by the standby controller for communication with distributed equipment. This auto-generated IP address equals the Main IP address plus 1 in the fourth octet. For example, if the Main IP address is 192.168.10.1, this auto-generated IP address is 192.168.10.2. **NOTE:** • This IP address is not editable in Control Expert. Its sole purpose is to provide seamless communication transitions on Hot Standby controller switchovers. • Avoid assigning this IP address (the Main IP address + 1) to any device that may communicate with the Hot Standby system. If you do assign this IP address to another device, a duplicate IP assignment condition may occur.
IP address A	The configurable IPv4 IP address for the controller with its A/B/Clear rotary selector switch, page 58 set to “A”. controller A uses this IP address for communication on the Ethernet RIO network.
IP address B	The configurable IPv4 IP address for the controller with its A/B/Clear rotary selector switch, page 58 set to “B”. controller B uses this IP address for communication on the Ethernet RIO network.
Subnetwork mask	The configurable 32-bit value used to identify both the network address and the subnetwork portion of the IP address.
Gateway address	The configurable IP address of the default gateway to which messages for other networks are transmitted.

⚠WARNING

UNINTENDED EQUIPMENT OPERATION

- Confirm that each module has a unique IP address.
- Do not assign an IP address equal to the Main IP Address, the Main IP Address + 1, IP Address A, or IP Address B to any Ethernet device that potentially communicates with the Hot Standby system.

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

Editing IP Address Settings for Adapter Modules

From the **IPConfig** tab, you can access IP address settings for (e)X80 EIO adapter modules. Click on the **Update CRA IP address configuration** link to open the **Ethernet Network Manager**, which lists adapter modules on connected Ethernet networks.

In the **Ethernet Network Manager**, you can edit the following settings for each adapter module:

- **IP address**: The configurable IPv4 IP address the adapter module uses for communication on the Ethernet network.
- **Identifier**: The text string used by the module to identify itself to other devices, for Ethernet services including DHCP and FDR. The value depends on the module you are using:
    - for 140CRA32100: 140CRA_*XXX*
    - for BMECRA31210: BMECRA_*XXX*
    - for BMXCRA312•0: BMXCRA_*XXX*

  Where *XXX* represents the concatenation of the two rotary switch settings on the (e) X80 EIO adapter module.

Configuring Data Variables for an M580 BMEH58•040(S) Hot Standby Application

Introduction

BMEH58•040 Hot Standby and BMEH58•040S safety Hot Standby controllers support the following data attributes:

Attribute	Controller	
	BMEH582040, BMEH582040S, BMEH584040, BMEH584040S	**BMEH586040, BMEH586040S**
Exchange On STBY	X	X
Retain	–	X
X: Supports the attribute. –: Does not include the attribute, because all data is retained.		

For a safety controller, each variable set to **Exchange On STBY** is associated with a task (MAST, FAST, or SAFE). The amount of data that can be exchanged from the primary to the standby safety controller depends on the task:

- MAST & FAST: up to 4 MB of data can be exchanged.
- SAFE: up to 1 MB of data can be exchanged.

For information on how to use the Control Expert **Data Editor**, and display the **Retain** and **Exchange On STBY** attributes, refer to the *Unity Pro Operating Modes* (see EcoStruxure™ Control Expert, Operating Modes) manual.

Retain

BME•586040 controllers present the **Retain** variable attribute. This attribute determines whether the variable value will persist after a warm start of the controller. If the attribute is:

- Selected: Variable data persists and is applied to the variable after a warm start.
- De-selected: Variable data is lost after a warm start; the variable value is reset.

For non-safety-related standalone Modicon M580 controllers, this attribute is read-only. It is selected by default and cannot be de-selected.

For both standalone and Hot Standby safety controllers, the **Retain** variable attribute is not included for variables created in the safe area. All safe data is not retained, because the SAFE task executes a cold start.

NOTE: In the event of a cold start of the controller, both retained and non-retained data is reset.

The amount of Refer to the *Modicon M580 High Availability System Planning Guide* varies, depending on the controller.

For the BME•586040 controllers, you cannot edit the **Retain** attribute for a variable that existed at controller start-up. When a variable is created online as part of a CCOTF change, you can edit the **Retain** attribute which remains modifiable until the first build change is performed.

NOTE: The amount of retained data is presented as saved data in the **Memory Usage** window.

Exchange On STBY

Before each scan in a Hot Standby system, the primary Hot Standby controller exchanges data with the standby controller. It exchanges only that data with the **Exchange On STBY** attribute set to **YES**.

NOTE:

- When a reference is initialized inside the **Data Editor**, the initialization variable needs to be part of the same task as the reference. Otherwise, a detected error message is included in the **Output Window** when the project is analyzed.
- The **Exchange On STBY** attribute is not editable for all variables.
- In a Hot Standby system, if you have configured explicit messaging using a communication function, exclude the communication function block `Management_Param` from the data to be transferred from primary to standby. To do this, de-select the **Exchange on STBY** attribute for the `Management_Param` parameter in Control Expert.

You cannot edit the **Exchange On STBY** attribute for a variable that existed at controller start-up. When a variable is created online as part of a CCOTF change, you can edit the **Exchange On STBY** attribute which remains modifiable until the first build change is performed.

The amount of Refer to the *Modicon M580 High Availability System Planning Guide* varies, depending on the controller.

Each variable that is included in the Hot Standby exchange also presents a read-only **Task** attribute. The setting of the **Task** attribute is auto-generated by Control Expert for each variable included in the Hot Standby exchange.

Configuring Hold Up Time for Drops and Devices

Hold Up Time

Hold up time is part of each configuration. It represents the time (in milliseconds) that device outputs are maintained in their current states after a communication disruption before reverting to their fallback values.

Hold up time settings can range from 50...65530 ms. By default, Control Expert sets hold-up time to 4 times the MAST **Watch Dog** setting. Because the default watchdog setting is 250 ms, Control Expert applies a default drop hold up time setting of 1000 ms.

Setting Hold Up Time for RIO Drops

When configuring MAST **Hold up time**, consider both of the following:

- The maximum time between controller requests.
- MAST task watchdog time.

If **Hold up time** is not set to a sufficiently large value, the outputs of a drop may enter fallback during a switchover. This can cause a disruption in the behavior of outputs that have a fallback setting other than *hold last value*.

To accommodate both MAST and FAST tasks for (e)X80 RIO drops, set drop **Hold up time** to a value not less than 4.4 times the MAST period.

M580 Hot Standby supports the following tasks:

Task	Type	Period	Watchdog time	Remote I/O platform:	
				Quantum RIO	**M580 (e)X80**
MAST[1]	Periodic	1...255 ms	10...1500 ms[2]	X	X
FAST	Periodic	1...255 ms	10...500 ms[2]	–	X
SAFE	Periodic	10...255 ms	10...500 ms[2]	–	X

X: Supported

–: Not supported

1. MAST task is mandatory and cannot be deactivated for both (e)X80 and Quantum RIO drops.

2. If CCOTF is activated, the minimum watchdog value is 64 ms.

Setting Hold Up Time for Distributed Equipment

The hold up time represents the time that device outputs are maintained in their current states after a communication disruption and before taking their fallback values. Because distributed devices are not connected to the primary controller during a switch-over, set the hold up time to a value greater than the expected duration of the communication interruption.

For Modbus TCP devices:

- Set the hold up time to exceed: 4.4 x (MAST period) + 600 ms.

For EtherNet/IP devices:

- Set the hold up time to exceed: 4.4 x (MAST period) + 5000 ms.

Transferring M580 Hot Standby Projects

Introduction

In an M580 Hot Standby system, both the primary controller and the standby controller begin by operating the same application. CCOTF changes that are made to the application running in the primary controller are not also made to the standby controller. This causes a logic mismatch to exist between the two controllers.

After modifications, it is necessary to transfer the application from the primary controller to the standby controller, so that both controllers are once again operating the same application. There are many ways to make this transfer.

NOTE: The operating mode setting of a safety PAC – either safety mode or maintenance mode – is not included in the transfer of an application from the primary PAC to the standby PAC. On a switchover, when a safety PAC switches from standby PAC to primary PAC, the operating mode is automatically set to safety mode.

For additional information on safety controller operating modes, refer to the *Modicon M580 Safety Manual* (see Modicon M580, Safety Manual).

Transferring the Application from the Primary to the Standby Controller

The Control Expert application can be transferred from the primary controller to the standby controller in many ways, including the following:

- **Automatic transfer**: If the non-primary controller is in a non-configured state, the primary controller automatically transfers the application program and data to the non-primary controller when it powers up. There are several ways a controller can be put into in a non-configured state, including:
  - It is a new device that is being deployed for the first time.
  - Its A/B/Clear rotary selector switch, page 58 was set to “Clear”, powered-up, then re-set to “A” or “B” (depending on the A/B designation of the primary controller).

    NOTE: To place the standby controller into run mode on restart, set the `CMD_RUN_AFTER_TRANSFER`, page 498 DDDT command to true before power-up.

- **Transfer from PC to the standby controller**: If your PC with Control Expert has open the same application as the one running in the primary controller, you can transfer the application from your PC to the standby controller. To do this, connect your PC to either the Ethernet service port or USB port of the standby controller, then use the **PLC > Transfer Project to PLC** command to make the transfer.

  NOTE: If the standby PAC is connected to a configuration tool, such as Control Expert, only the connected configuration tool can transfer an application to the standby PAC. In this case, the primary PAC cannot transfer an application to the standby.

- **Transfer from primary controller to standby controller**: With Control Expert connected to the primary controller, and with both the primary and standby controllers running, use one of the following methods to make the transfer:
  - Use the Control Expert **PLC > Transfer Project from Primary to StandBy PLC** GUI command.

    or
  - Use the `CMD_APP_TRANSFER` command of the `T_M_ECPU_HSBY` DDT.

    NOTE:

    - The application transferred is the backup application, stored in flash memory or on the SD card. If the application running does not match the backup application, perform an application backup (**PLC > Project Backup... > Backup Save** or set the %S66 system bit to 1) before performing the transfer.
    - If the `CMD_RUN_AFTER_TRANSFER`, page 498 flag is set, the standby controller automatically starts to run after completion of the transfer, reducing down time for the standby controller.

  In each case, if both the primary and standby controllers are equipped with SD memory cards, the application is transferred to both the standby controller and its SD memory card.

- **SD memory card**: If the primary controller includes an SD memory card with the current application, take the SD card from the primary controller, place it into the standby controller, then reboot the standby.

In each case:

- The application is transferred only if the application in the standby controller is different from the application being transferred to it.
- If the application running in the primary controller is different from the application stored in flash memory or on the SD memory card, perform a backup of the running application (**PLC > Project Backup > Backup Save**) before making the transfer.

NOTE:

- You cannot transfer the application from the standby controller to the primary controller.
- If the `Logic_Mismatch_Allowed` command is set, and if the **Number of modifications** has not been reached, you can connect Control Expert to the standby controller, then use the `CMD_SWAP` DDT command to make the standby controller the primary controller. Thereafter, you can transfer the application from new primary controller (formerly the standby) to the standby controller (formerly the primary).

Run After Transfer

If you use program logic or an animation table to set the `T_M_ECPU_HSBY` DDT command `CMD_RUN_AFTER_TRANSFER`, page 542, the primary PAC automatically begins to run immediately upon completion of the transfer.

Offline Application Modification with Allowed Application Mismatch

Procedure

⚠WARNING

UNINTENDED EQUIPMENT OPERATION

- Examine the impacts of the modifications on the application before transferring a modified application to the Standby controller.
- Ensure that the modified application does not have adverse effects on the process.

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

To make offline modifications to an application program in either CPU, follow these steps:

Step	Action
1	Verify the following: • The *HSBY_BUILD_OFFLINE (see EcoStruxure™ Control Expert, System, Block Library)* function block is implemented in the application program in both Primary and Standby CPU. • Application program is equal in Primary and Standby CPU. • The **Standby On Logic Mismatch** parameter is set in the **Hot Standby** configuration tab, page 471.
2	Connect Control Expert to the Primary CPU.
3	Set to 1 the *ALLOW_MISMATCH* bit of the *HSBY_BUILD_OFFLINE* function block. This setting authorizes the CPU to remain synchronized with its pair if a program is modified offline. **NOTE:** Verify that the section where the function block resides is executed by the Primary and the Standby CPU (verify the CPU section execution settings in the **Hot Standby** tab).
4	Confirm that the logic mismatch, page 463 is enabled.
5	Disconnect Control Expert from the CPU.
6	Modify the application program offline. **NOTE:** Only modifications within the scope of the application code and/or some items under the DTM browser modifications are working as valid offline build modification for the Standby CPU. Any other modifications (configuration changes for example) are not taken into account by the *HSBY_BUILD_OFFLINE* function block.
7	Perform a **Build Changes** and save. **NOTE:** Do not perform a **Rebuild All Project** because the Standby CPU may not switch to the *RUN STANDBY* state after the program download and **RUN**. The swap from Standby to Primary cannot be performed.

Step	Action
8	Connect Control Expert to the Standby CPU.
9	Open the modified application program.
10	Download the program in the Standby CPU.
11	Select **RUN**. **NOTE:** Check that the CPU is now in *WAIT* state. **NOTE:** If the CPU does not transition to *WAIT* state, proceed as indicated in the following **Workaround**, page 489.
12	For safety Hot Standby CPUs, check if the safe part of the new application has been modified (*SAFETY_LOGIC_MISMATCH* bit = 1). If so, set the operating mode of the standby PAC to maintenance mode.
13	On the Standby CPU set to 1 the *ALLOW_MISMATCH* bit of the *HSBY_BUILD_OFFLINE* function block. This setting authorizes the CPU to remain synchronized with its pair if a program is modified offline. **Result:** The Standby CPU switches from *WAIT* to *RUN STANDBY* state. **NOTE:** Verify that the section where the function block resides is executed by the Standby CPU (verify the CPU section execution settings in the **Hot Standby** tab).
14	Verify that: • The Primary CPU is in *RUN PRIMARY*. • The Standby CPU is in *RUN STANDBY*.
15	Perform a switchover using the `CMD_SWAP` command, page 542, or by clicking **Animation > Task > Swap controllers > Primary <-> Standby** in the CPU configuration window in Control Expert. **NOTE:** Verify that the Standby CPU switched to Primary CPU.
16	Perform an application transfer to the Standby CPU, page 485.
17	Perform an application **RUN** to the Standby CPU, page 485.
18	On the Standby and Primary CPU reset to 0 the *ALLOW_MISMATCH* bit of the *HSBY_BUILD_OFFLINE* function block.

NOTE: Application mismatch topic is discussed in the configuration compatibility, page 463 section.

Workaround When the Standby CPU Does Not Transition to *WAIT* State

If the Standby CPU does not transition to *WAIT* state after the **RUN** command in step 11 (a **Rebuild All Project** has been performed for example), the initial program and configuration need to be transferred in the Standby CPU.

Step	Action
1	Connect Control Expert to the Primary CPU
2	Upload the application program from the Primary CPU for future offline modifications. **NOTE:** The modifications done previously to the application program in Control Expert are lost.
3	Perform an application transfer to the Standby CPU, page 485.
4	Perform an application **RUN** to the Standby CPU, page 485.
5	Disconnect Control Expert from the CPU.
6	Modify the application program and repeat the**Procedure**, page 488.

Use Case

In an existing Hot Standby system, the process to modify an application offline and transfer it to the Primary and Standby CPU follows these macro steps (refer to the preceding detailed procedure for more information):

- Using the CCOTF online modification, page 475, insert the *HSBY_BUILD_OFFLINE* function block in the application program of the Primary and Standby CPU. The function block needs one input bit for control and provides a status output.
- Allow the application mismatch in the Primary CPU by setting to 1 the *ALLOW_MISMATCH* input bit of the *HSBY_BUILD_OFFLINE* function block in the Primary CPU.
- Modify the application program offline.
- **Build Changes** (do not perform a **Rebuild All Project**).
- Transfer the modified application program in the Standby CPU.
- Allow the application mismatch in the Standby CPU by setting to 1 the *ALLOW_MISMATCH* input bit of the *HSBY_BUILD_OFFLINE* function block in the Standby CPU.
- Perform a switchover.
- Transfer the application in the new Standby CPU.
- Reset to 0 the *ALLOW_MISMATCH* input bit of the *HSBY_BUILD_OFFLINE* function block in the Primary and Standby CPU.

Restoring and Backing Up Projects

Restoring and Backing Up Projects

The CPU application RAM (see Modicon M580, Hardware, Reference Manual) and the CPU flash memory automatically and manually perform the following:

- Restore a project in the CPU from the flash memory (and the memory card if inserted):
  - Automatically after a power cycle
  - Automatically on a warm restart
  - Automatically on a cold start
  - Manually with a Control Expert command: **PLC > Project Backup > Backup Restore**

    NOTE: If a memory card is inserted with a different application than the application in the CPU, the application is transferred from the memory card to the CPU application RAM when the restore function is carried out.
- Save the CPU project in the flash memory (and the memory card if inserted):
  - Automatically after an online modification is performed in the application RAM
  - Automatically after a download
  - Automatically on detection of `%S66` system bit rising edge
  - Manually with a Control Expert command: **PLC > Project Backup > Backup Save**

    NOTE: Backup begins after the completion of the current MAST cycle and before the start of the next MAST cycle.

    Because MAST is configured as periodic for all Hot Standby CPUs, set the MAST period to a value larger than the actual MAST execution time. This lets the processor complete an entire backup without interruption.

    If the MAST period is set to a value less than the actual MAST execution time, backup processing is fragmented and requires a longer time to finish.
- Compare the CPU project and the flash memory project:
  - Manually with a Control Expert command: **PLC > Project Backup > Backup Compare**

    NOTE: When a valid memory card is inserted, page 78 with a valid application, the application backup and restore operations are performed as follows:
    - The application backup is performed on the memory card first and then on the flash memory.
    - The application restore is performed from the memory card to the CPU application RAM first and then copied from the application RAM to the flash memory.

Managing M580 Hot Standby Data Exchanges

What's in This Chapter

Overview

This chapter describes M580 Hot Standby system data management and the `T_M_ECPU_HSBY` DDT.

Exchanging M580 Hot Standby Data

Periodic Data Exchanges

The Hot Standby CPUs perform two periodic data exchanges:

- Before each MAST cycle, the primary CPU transmits to the standby CPU application variables, system status and I/O data.
- Periodically, both CPUs exchange the contents of the `T_M_ECPU_HSBY` DDT.

Data Transmitted Each MAST Cycle

Before each MAST task, the primary controller transmits data to the standby controller in two ways. The primary controller uses:

- The Hot Standby link to send application variables, system status, and I/O data.
- The Ethernet RIO link to send application variables and system status.

When communication is lost on the Hot Standby link, the standby controller does not receive updated I/O data and application variables. If communication is lost for three (3) seconds or more, the standby controller enters wait state.

Your application needs to regularly check the data synchronization of the MAST, FAST, and SAFE (for safety controllers) tasks through the Hot Standby link. You can do this using the `MAST_SYNCHRONIZED`, `FAST_SYNCHRONIZED` and `SAFE_SYNCHRONIZED` bits in the `T_M_ECPU_HSBY` DDT.

NOTE: Due to I/O data size and transfer time constraints, I/O data is not exchanged by the primary controller with the standby controller over the Ethernet RIO link.

Transfer of the Hot Standby DDT

The exchange of the `T_M_ECPU_HSBY` DDT is a 2-way data exchange made while both controllers are running. This exchange is made over both the Hot Standby link and the Ethernet RIO link.

The exchange occurs every 5 ms over the Hot Standby link, and every 10 ms over the EIO link. The exchange occurs regardless of the Hot Standby state of the controllers (primary, standby, wait, or stop). This exchange includes up to 64 words of variable items where the **Exchange On STBY** attribute is editable and has been selected.

Identifying Exchanged Data

Only data items with the **Exchanged On STBY** attribute set to **YES** are included in the data exchange. This attribute is editable for some data variables, but is automatically set for other variables:

Variable type	Exchange On STBY default setting	Editable?
State RAM	Yes	No
Located variables	Yes	No
Unlocated variables	Yes	Yes
Device DDT (managed)	Yes	No
Device DDT (unmanaged)	Yes	Yes

You can specify which unmanaged DDDT variables are included in the data exchange by setting the **Exchange On STBY** flag to **NO**.

When you create a variable and set its **Exchange On STBY** flag to **YES**, that variable appears in the `LOCAL_HSBY_STS` area of the instantiated `T_M_ECPU_HSBY` DDDT, under the `REGISTER` element. The `REGISTER` element can contain up to 32 `DWORD`s (64 `WORD`s of data).

The maximum amount of data that can be exchanged depends on Refer to the *Modicon M580 High Availability System Planning Guide*. If the amount of data in your Hot Standby system exceeds the maximum amount the controller can transmit, you can:

- Use a controller with a higher data transfer capacity.
- De-select the **Exchange On STBY** attribute for some unmanaged DDDT variables.

- Re-design your Hot Standby network so that the amount of Hot Standby data to be exchanged does not exceed controller capacity.

Associating Variables with Tasks

Each data item is associated with a task. When you create a new data item in the **Data Editor**, you need to associate it with a task:

- A MAST task is required by the Hot Standby system, and can be assigned to data items related to the Hot Standby controller and RIO drops (both Quantum and M580).
- FAST tasks are optional for all Hot Standby controllers, and can be assigned only to M580 (e)X80 drops.

  NOTE: In an M580 Hot Standby system, variables related to Quantum RIO drops cannot be assigned to a FAST task.
- Safe data are automatically associated only with the SAFE task.

Preconditions for Data Exchange: Primary and Standby Controllers

The Hot Standby data exchange is made while one Hot Standby controller remains the primary and the other is the standby. Both the primary controller and a standby controller can continue in their roles as long as the Hot Standby link remains operational.

A single break, page 537 in the Ethernet RIO main ring will not cause an interruption of Ethernet RIO communication between the primary and standby controllers. The controllers continue to function as primary and standby respectively. The primary controller continues to exchange data with the standby over both the Hot Standby and the Ethernet RIO links.

Two breaks, page 538 in the Ethernet RIO main ring (depending on their location) can cause a loss of Ethernet RIO communication between the primary and standby controllers. However, even if the two controllers are isolated from each other on the Ethernet RIO ring, they can still communicate over the Hot Standby link. If both controllers continue to communicate with RIO drops, page 540, the controllers continue to function as primary and standby respectively. The primary controller continues to exchange data with the standby over the Hot Standby link.

Effects of Online Modifications to Hot Standby Data

When you modify the configuration of – or application in – the primary controller, those changes are not applied to the configuration of the standby controller. The exchange of Hot Standby application variables from the primary to the standby is affected, as follows:

- Data objects added to the primary controller configuration do not exist in the standby controller. In this case, the new data objects are not exchanged and:
  - The `DATA_LAYOUT_MISMATCH` DDT element is set.
  - The `DATA_DISCARDED` DDT element indicates the quantity, in kB (rounded upwards), of data sent by the primary controller but rejected by the standby controller.
- Data objects deleted from the primary controller configuration continue to exist in the standby controller. No updates can be exchanged for these data objects. In this case, the standby controller applies the previous value for this data and:
  - The `DATA_LAYOUT_MISMATCH` DDT element is set.
  - The `DATA_NOT_UPDATED` DDT element indicates the quantity, in kB (rounded upwards), of data that is retained by the standby controller but not updated.
- Unchanged data objects remain common to both the primary controller and the standby controller, and continue to be included in the data exchange.

The data structure of the primary controller and standby controller will be equalized on next application transfer.

Hot Standby DDT Data Structure

Introduction

The `T_M_ECPU_HSBY` DDT is the exclusive interface between the M580 Hot Standby system and the application running in a BMEH58•040 or BMEH58•040S controller. The DDT instance should appear as: `ECPU_HSBY_1`.

NOTICE
UNMONITORED LOSS OF REDUNDANCY IN HOTSTANDBY SYSTEM Review and manage the `T_M_ECPU_HSBY` DDT for proper operation of the system. **Failure to follow these instructions can result in equipment damage.**

The `T_M_ECPU_HSBY` DDT presents three distinct sections:

- `LOCAL_HSBY_STS`: Provides information about the local PAC. Data is both auto-generated by the Hot Standby system, and provided by the application. This data is exchanged with the remote PAC.
- `REMOTE_HSBY_STS`: Provides information about the remote PAC, and contains the image of the last received exchange from the counterpart PAC. The validity of this information is represented by the `REMOTE_STS_VALID` flag in the common part of this DDT.

  NOTE: The structure of both the `LOCAL_HSBY_STS` and `Remote_HSBY_STS` sections are determined by the `HSBY_STS_T` data type, and are therefore identical. Each is used to describe data relating to one of the two Hot Standby PACs.
- A common part of the DDT: Consists of several objects, including status data, system control objects, and command objects:
  - Status data is provided by the Hot Standby system as a result of diagnostic checking.
  - System control objects enable you to define and control system behavior.
  - Command data objects include executable commands you can use to modify the system state.

Local PAC versus Remote PAC

The `T_M_ECPU_HSBY` DDT employs the terms *local* and *remote*:

- *Local* refers to the Hot Standby PAC to which your PC is connected.

- *Remote* refers to the other Hot Standby PAC.

Data Boundary Alignment

M580 BMEH58•040 and BMEH58•040S controllers feature a 32-bit data design. For this reason, stored data objects are placed on a four-byte boundary.

T_M_ECPU_HSBY DDT

You must confirm that the standby controller is ready to assume the primary role before executing a swap command.

Verify that the value of the REMOTE_HSBY_STS.EIO_ERROR bit of the standby controller is 0 before you execute a swap command (either by application logic or in Control Expert).

The `T_M_ECPU_HSBY` DDT consists of these objects:

Element	Type	Description	Written by
REMOTE_STS_VALID	BOOL	• TRUE: Both HSBY_LINK_ERROR and HSBY_SUPPLEMENTARY_LINK_ERROR are set to 0. • FALSE (default): Both HSBY_LINK_ERROR and HSBY_SUPPLEMENTARY_LINK_ERROR are set to 1.	System
APP_MISMATCH	BOOL	The original application in the two PACs is different. (Default = FALSE)	System
LOGIC_MISMATCH_ALLOWED	BOOL	• TRUE: The standby remains standby in case of logic mismatch. • FALSE (default): The standby goes into wait state in case of logic mismatch.	Application
LOGIC_MISMATCH	BOOL	Different revisions of the same application exist in the two PACs. (Default = FALSE)	System
SFC_MISMATCH	BOOL	• TRUE: The applications in the primary PAC and the standby PAC are different in at least one SFC section. In the event of a switchover, the graphs that are different are reset to their initial state. • FALSE (default): All SFC sections are identical.	System
OFFLINE_BUILD_MISMATCH	BOOL	The two PACs are running different revisions of the same application. In this condition: • A data exchange between the two PACs may not be possible. • A swap or switchover may not be transparent. • Neither PAC can be standby	System

Element	Type	Description	Written by
		(Default = FALSE)	
APP_BUILDCHANGE_DIFF	UINT	The number of build change differences between the applications in the primary PAC versus the standby PAC. Evaluated by the primary.	System
MAX_APP_ BUILDCHANGE_DIFF	UINT	Maximum number of build change differences permitted by the Hot Standby system, from 0...50 (default = 20). Set in the **Hot Standby** tab as **Number of modifications**.	Application
FW_MISMATCH_ALLOWED	BOOL	Allows mismatched firmware between primary and standby controllers: • TRUE: the standby remains standby in case of FW mismatch. • FALSE (default): the standby goes into wait state in case of FW mismatch. (Default = FALSE)	Application
FW_MISMATCH	BOOL	The OS are different in the two PACs. (Default = FALSE)	System
DATA_LAYOUT_MISMATCH	BOOL	The Data layout are different on the two PACs. The data transfer is partially performed. (Default = FALSE)	System
DATA_DISCARDED	UINT	Number of KB sent by the primary and discarded by the standby (rounded up to the next KB). Represents data for variables added to primary, but not to standby. (Default = 0)	System
DATA_NOT_UPDATED	UINT	Number of KB not updated by the standby (rounded up to the next KB). Represents variables deleted from the primary that remain in the standby. (Default = 0)	System
BACKUP_APP_MISMATCH	BOOL	• FALSE (default): The backup application In the 2 Hot Standby PACs are equal. **NOTE:** The backup application resides in flash memory or on the SD memory card of the PAC. It is created either by the **PLC > Project Backup... > Backup Save** command, or by setting the %S66 system bit (Application Backup) to 1. • TRUE: All other cases.	System
PLCA_ONLINE	BOOL	PAC A is configured to enter the primary or standby state. (Default = TRUE) **NOTE:** Executable only on PAC A.	Configuration
PLCB_ONLINE	BOOL	PAC B is configured to enter the primary or standby state. (Default = TRUE) **NOTE:** Executable only on PAC B.	Configuration

Element	Type	Description	Written by
CMD_SWAP	BOOL	• Set to 1 by program logic or animation table to initiate a switchover. The primary goes into wait, then the standby goes primary, finally the wait goes standby. The command is ignored if there is no standby. **NOTE:** Executable on both primary and standby. • Reset to 0 (default) by the system on switchover completion or if there is no standby. **NOTE:** • This command is designed to be used by the application in response to detected errors. It is not intended to be used for periodic switchovers. • If the application has to switchover periodically, the period between switchovers must not be less than 120 seconds.	Application / System
CMD_APP_TRANSFER	BOOL	• Set to 1 by program logic or animation table to start an application transfer from the primary to the standby. Executable only on the primary. **NOTE:** The application transferred is the backup application, stored in flash memory or on the SD card. If the application running does not match the backup application, perform an application backup (**PLC > Project Backup... > Backup Save** or set the %S66 system bit to 1) before performing the transfer. • Reset to 0 (default) by the system on transfer completion.	Application / System
CMD_RUN_AFTER_ TRANSFER	BOOL[0...2]	• Set to 1 by program logic or animation table to automatically start in Run after a transfer. **NOTE:** Executable only on the primary. • Reset to 0 (default) by the system after transfer completion and: ◦ remote PAC is in Run ◦ PAC is not primary ◦ by animation table or logic command	Application / System
CMD_RUN_REMOTE	BOOL	• Set to 1 by program logic or animation table to run the remote PAC. This command is ignored if the CMD_STOP_REMOTE is TRUE. **NOTE:** Executable only on the primary. • Reset to 0 (default) by the system when the remote PAC enters standby or wait state.	Application / System

Element	Type	Description	Written by
CMD_STOP_REMOTE	BOOL	• Set to 1 by program logic or animation table to stop the remote PAC. **NOTE:** Executable on the primary, the standby, or a stopped PAC. • Reset to 0 (default) by the application to end the stop command.	Application
CMD_COMPARE_INITIAL_ VALUE	BOOL	• Set to 1 by program logic or animation table to begin a comparison of the initial values of variables exchanged by the two Hot Standby PACs. **NOTE:** Executable on both primary and standby only in Run mode. • Reset to 0 (default) by the system when the comparison is complete, or if the comparison is not possible.	Application / System
INITIAL_VALUE_MISMATCH	BOOL	• TRUE: if the initial values for exchanged variables are different or if the comparison is not possible. • FALSE: if the initial values for exchanged variables are identical.	System
MAST_SYNCHRONIZED (1)	BOOL	• TRUE: if the exchanged data from the previous MAST cycle was received by the standby. • FALSE (default): if the exchanged data from at least the previous MAST cycle was not received by the standby. **NOTE:** Closely monitor the MAST_ SYNCHRONIZED and FAST_SYNCHRONIZED variables related to the MAST and FAST tasks as indicated at the end of this table.	System
FAST_SYNCHRONIZED (1)	BOOL	• TRUE: if the exchanged data from the previous FAST cycle was received by the standby. • FALSE (default): if the exchanged data from at least the previous FAST cycle was not received by the standby. **NOTE:** Closely monitor the MAST_ SYNCHRONIZED and FAST_SYNCHRONIZED variables related to the MAST and FAST tasks as indicated at the end of this table.	System
SAFE_SYNCHRONIZED	BOOL	• TRUE: if the exchanged data from the last SAFE cycle was received by the standby. • FALSE (default): if, at least, the exchanged data from the last SAFE cycle was not received by the standby.	System
SAFETY_LOGIC_ MISMATCH	BOOL	• TRUE: the SAFE logic part of the application is different in the two PACs. • FALSE (default): the SAFE logic part of the application is identical in the two PACs.	–

Element	Type	Description	Written by
		NOTE: The content for this element is determined by comparing system word %SW169 for each PAC.	
LOCAL_HSBY_STS	T_M_ ECPU_ HSBY_STS	Hot Standby status for the local PAC	(see below)
REMOTE_HSBY_STS	T_M_ ECPU_ HSBY_STS	Hot Standby status for the remote PAC	(see below)
(1): • Closely monitor the MAST_SYNCHRONIZED, FAST_SYNCHRONIZED, and SAFE_SYNCHRONIZED variables related to the MAST, FAST and SAFE tasks. If its value is zero (FALSE), then the database exchanged between the primary and the standby PACs is not transmitted at each cycle. In this situation, change the configured period of this task with a higher value than its current execution time (for the MAST task: %SW0 > %SW30; for the FAST task %SW1 > %SW33; for the SAFE task %SW4 > %SW42. More details on %SW0 + %SW1 and %SW30 + %SW31 in EcoStruxure™ Control Expert, System Bits and Words, Reference Manual). • Example of consequence: upon an Application Program Transfer (APT) command, the primary PAC might not be able to transfer the program to the standby PAC.			

T_M_ECPU_HSBY_STS Data Type

The `T_M_ECPU_HSBY_STS` data type presents the following elements:

Element	Type	Description	Written by
HSBY_LINK_ERROR	BOOL	• TRUE: No connection on the Hot Standby link. • FALSE: The Hot Standby link is operational.	System
HSBY_SUPPLEMENTARY_ LINK_ERROR	BOOL	• TRUE: No connection on the Ethernet RIO link. • FALSE: The Ethernet RIO link is operational.	System
WAIT	BOOL	• TRUE: The PAC is in Run state but waiting to go primary or standby. • FALSE: The PAC is in standby, primary or stop state.	System
RUN_PRIMARY	BOOL	• TRUE: The PAC is in primary state. • FALSE: The PAC is in standby, wait or stop state.	System
RUN_STANDBY	BOOL	• TRUE: The PAC is in standby state. • FALSE: The PAC is in primary, wait or stop state.	System
STOP	BOOL	• TRUE: The PAC is in stop state. • FALSE: The PAC is in primary, standby or wait state.	System

Element	Type	Description	Written by
PLC_A	BOOL	• TRUE: the PAC A/B/Clear switch, page 58 is in "A" position. • FALSE: the PAC switch is not in "A" position.	System
PLC_B	BOOL	• TRUE: the PAC A/B/Clear switch, page 58 is in "B" position. • FALSE: the PAC switch is not in "B" position.	System
EIO_ERROR	BOOL	• TRUE: The PAC does not detect any of the configured Ethernet RIO drops. • FALSE: The PAC detects at least one configured Ethernet RIO drop. **NOTE:** This bit is always FALSE when no drop is configured.	System
SD_CARD_PRESENT	BOOL	• TRUE: A valid SD card is inserted. • FALSE: No SD card, or an invalid SD card is inserted.	System
LOCAL_RACK_STS	BOOL]	• TRUE: The local rack configuration is OK. • FALSE: The local rack configuration is not OK (for example, modules missing or in incorrect slots, etc.)	Application
MAST_TASK_STATE	BYTE	State of the MAST task: • 0: Not existent • 1: Stop • 2: Run • 3: Breakpoint • 4: Halt	System
FAST_TASK_STATE	BYTE	State of the FAST task: • 0: Not existent • 1: Stop • 2: Run • 3: Breakpoint • 4: Halt	System
SAFE_TASK_STATE	BYTE	State of the SAFE task: • 0: Not existent • 1: Stop • 2: Run • 3: Breakpoint • 4: Halt	System
REGISTER	WORD[0...63]	Unmanaged data added to the application via the **Exchange on STBY** attribute.	Application

Data Storage Elementary Functions

Data Storage Elementary Functions

The following DataStorage_EF elementary functions are supported in Control Expert for all tasks in the M580 BMEH58•040 non-safety-related Hot Standby controllers, and for process tasks in the M580 BMEH58•040S safety Hot Standby controllers.

EF	Hot standby controller state		
	Primary	Standby	Wait
CREATE_FILE	X	X	X
DELETE_FILE	X	X	X
GET_FILE_INFO*	X	X	X
GET_FREESIZE*	X	X	X
OPEN_FILE	X	X	X
RD_FILE_TO_DATA	X	X	X
SET_FILE_ATTRIBUTES	X	X	X
WR_DATA_TO_FILE	X	X	X
* Read-only function			

NOTE: Changes made to an SD card in either the primary or standby controller, using an elementary function, are not replicated in the SD card of the other controller in the event of a switchover.

CREATE_FILE

The CREATE_FILE (see EcoStruxure™ Control Expert, System, Block Library) function creates a file called *FILENAME*, if it does not already exist. If a file by that name already exists, the CREATE_FILE command behaves the same as the OPEN_FILE command.

DELETE_FILE

The `DELETE_FILE (see EcoStruxure™ Control Expert, System, Block Library)` function deletes a file identified by its *FILENAME*. Close a file, using the `CLOSE_FILE` function before deleting it.

GET_FILE_INFO

The `GET_FILE_INFO (see EcoStruxure™ Control Expert, System, Block Library)` function retrieves information about a specified target file. Execute the `OPEN_FILE` function for the target file before executing the `GET_FILE_INFO` function, because the identity of the target file comes from the output parameter of the `OPEN_FILE` block.

GET_FREESIZE

The `GET_FREESIZE (see EcoStruxure™ Control Expert, System, Block Library)` function displays the amount of available space on the SD memory card.

OPEN_FILE

The `OPEN_FILE (see EcoStruxure™ Control Expert, System, Block Library)` function opens a specified file, provided the file already exists.

RD_FILE_TO_DATA

The `RD_FILE_TO_DATA (see EcoStruxure™ Control Expert, System, Block Library)` function allows data to be read from a file, at the current position of the file, and enables it to be copied to a variable.

SET_FILE_ATTRIBUTES

The `SET_FILE_ATTRIBUTES (see EcoStruxure™ Control Expert, System, Block Library)` function enables the setting of file attributes that set or clear the read-only flag for that file.

WR_DATA_TO_FILE

The `WR_DATA_TO_FILE` (see EcoStruxure™ Control Expert, System, Block Library) function writes the value of a specified variable to the selected file. The data written is added after the current position in the file.

M580 CPU Programming and Operating Modes

What's in This Chapter

Overview

This chapter provides information on M580 CPU I/O exchanges, tasks, memory structure, and operating modes.

I/O and Task Management

Overview

This section presents information on M580 I/O addressing and management, tasks allowed, and I/O scanning capabilities.

I/O Exchanges

I/O Vision

Each module uses a structure that represents inputs, outputs, control, and diagnostic data. The structures can be represented using:

- topological addressing / IODDT
- Device DDT

I/O Module Location	I/O Family	Topological Addressing / IODDT	Device DDT
local rack	(e)X80	X	X
	Premium	X	–

I/O Module Location	I/O Family	Topological Addressing / IODDT	Device DDT
RIO	(e)X80	–	X
	Quantum	–	X
distributed equipment	Schneider Electric or third party	–	X
X Supported. When both visions are supported, select one of the exchange types when adding the equipment. **–** Not supported.			

Adding an I/O Module in Control Expert

When you insert an I/O module on a rack in Control Expert, the type of addressing appears in the bottom of the **New Device** dialog box. Choose between the following:

- **I/O data type: Topological** (default)
- **I/O data type: Device DDT**

  NOTE: If you want to change the type of addressing you selected when you added an I/O module to your application, delete the module from your application and then insert the module again selecting the appropriate addressing type.

Exchange Types

I/O modules in an M580 system can be controlled, read, or written with 2 types of exchanges:

- implicit exchanges

  Implicit exchanges are performed automatically on each cycle of the task (MAST, FAST, AUX0, AUX1) associated with the I/O modules. They are used to read inputs from and write outputs o the modules.
- explicit exchanges

  Explicit exchanges are performed on application request. They are typically for detailed diagnostics and to set/read command and adjust parameters. They use specific function blocks.

  An acknowledgment or reply is sent once the requested action is performed. This reply may be received a few cycles after the request was sent.

  NOTE: Explicit exchanges are performed in the MAST task.

Explicit Exchanges

Function block usage depends on the module location and I/O vision selected for the module:

I/O Module Location	I/O Vision	Function Block
Local rack	Topological addressing/ IODDT	READ_PARAM
		READ_STS
		READ_TOPO_ADDR
		RESTORE_PARAM
		SAVE_PARAM
		WRITE_CMD
		WRITE_PARAM
		READ_VAR
		WRITE_VAR
		DATA_EXCH
	Device DDT	READ_PARAM_MX
		READ_STS_MX **NOTE:** MOD_FAULT parameter is not automatically updated; perform a READ_STS_MX.
		RESTORE_PARAM_MX
		SAVE_PARAM_MX
		WRITE_CMD_MX
		WRITE_PARAM_MX
RIO and local rack	Device DDT	READ_STS_MX
		WRITE_CMD_MX

The function blocks mentioned in previous table are detailed in the *Explicit Exchange* part of *Control Expert, I/O Management, Block Library manual*, and in the *Extended* part of *Control Expert, Communication, Block Library manual*.

CPU Tasks

Introduction

An M580 CPU can execute single-task and multi-task applications. Unlike a single-task application which only executes the MAST task, a multi-task application defines the priorities of each task.

There are four tasks available (see *Application Program Structure* chapter in *Control Expert Program Languages and Structure Reference Manual*) and two types of event tasks:

- MAST
- FAST
- AUX0
- AUX1
- I/O event in a local rack only
- timer event in a local rack only

NOTE: The time to perform an *update init values with current values* operation is not taken into account in the watchdog calculation.

Task Characteristics

The time model, task period, and maximum number of tasks per CPU are defined according to the standalone or Hot Standby CPU reference.

Standalone CPUs:

Task	Time Model	Task Period (ms)		BMEP58 References					
		Range	Default Value	1020 (H)	20•0 (H)	30•0	40•0	5040 (C)	6040 (C)
MAST(1.)	cyclic(2.) or periodic	1...255	20	X	X	X	X	X	X
FAST	periodic	1...255	5	X	X	X	X	X	X
AUX0	periodic	10...255-0 by 10	100	X	X	X	X	X	X

Task	Time Model	Task Period (ms)		BMEP58 References					
		Range	Default Value	1020 (H)	20•0 (H)	30•0	40•0	5040 (C)	6040 (C)
AUX1	periodic	10...255-0 by 10	200	X	X	X	X	X	X

1. MAST task is mandatory.

2. When set to cyclic mode, the minimum cycle time is 8 ms if there is a RIO network and 1 ms if there is no RIO network in the system.

X This task is supported.

Hot Standby CPUs:

Task	Time Model	Task Period (ms)		CPU Reference (BMEH58 ...		
		Range	Default Value	2040(C)	4040(C)	6040(C)
MAST(1.)	periodic(2.)	1...255	20	X	X	X
FAST(3.)	periodic	1...255	5	X	X	X
AUX0(4.)	—	—	—	—	—	—
AUX1(4.)	—	—	—	—	—	—

1. MAST task is mandatory.

2. Only periodic is supported; cyclic is not supported.

3. Supported for (e)X80 ERIO drops.

4. Not supported.

X This task is supported.

BMEP58•••• CPU Memory Structure

Overview

This section explains the CPU memory structure.

Memory Structure

CPU Memory

3 types of memories are available in a BMEP58•••• CPU:

- non-persistent application RAM: run the application program and store temporary data
- flash memory: back up the application program and a copy of %MW values
- optional SD memory card: store application and data in parallel to the CPU flash memory, allowing a fast CPU hardware replacement

Application Download to the CPU Memory

CPU memory involved during an application download from a programming terminal:

- Application is transferred into the non-persistent application RAM.
- If a memory card is inserted, working and not write protected, then an internal backup is performed in the memory card.
- The application backup is performed in the the flash memory.

  NOTE: A write protected memory card inserted disables the application download.

Application Upload from the CPU Memory

The application upload reads and copies non-persistent application content from RAM to your selected location.

Application Online Modification Backup

An application program modification is performed in the CPU non-persistent memory with an automatic backup performed as follows:

- If a memory card is inserted, working and not write protected, then the backup is performed in the memory card.
- The application backup is performed in the flash memory.

NOTE: The online modification is disabled when a write protected memory card is inserted.

Application Memory Self Modification

The user code may modify the application content (for example to save I/O parameters or replace variables initial value by the current value).

In such a case, only the non-persistent application RAM content is modified.

To back up the application in the memory card and to the flash memory, use the system bit `%S66`.

BMEP58•••• CPU Operating Modes

Overview

This section provides information on the CPU operating modes.

Managing Run/Stop Input

Input Run/Stop

The `%Ir.m.c` input can be parameterized to switch the PAC to **Run/Stop** mode as follows:

- Set `%Ir.m.c` to 1: The PAC switches to **Run** mode (executing the program).
- Set `%Ir.m.c` to 0: The PAC switches to **Stop** mode (stopping program execution).

NOTE:

- A Stop command takes priority over a Run command. A Stop command sent from a terminal or via the network has priority over the `%Ir.m.c` input.

  An error detected on the Run/Stop input causes the PAC to switch to **Stop** mode.

  Do not enable this option if the associated discrete input is mapped in state RAM because this inhibits the start-up of the PAC.
- The input format is either `%Ir.m.c` or *Device DDT* from a non-safety-related input module.

Memory Protect

The input `%Ir.m.c` can be parameterized to protect the internal application RAM and the memory card as follows:

- `%Ir.m.c` to 0: The internal application and the memory card **are not** protected.
- `%Ir.m.c` to 1: The internal application and the memory card **are** protected.

NOTE:

- If the input is in error, `%Ir.m.c` is considered at 1 (memory is protected). To remove this protection in the configuration screen, the input should not be in error.
- The input format is either `%Ir.m.c` or *Device DDT* from a non-safety-related input module.

Managing Run/Stop Remote Access

When configuring the M580 PAC, you can help prevent remote commands/requests from accessing the controller **Run/Stop** modes. Select the respective **Run/Stop input** and **Run/Stop by input only** check boxes according to the following table parameters to determine the type of remote access for your system.

Run/Stop Input	Run/Stop by Input Only	Description
–	–	Allows remote access to run/stop the controller by request.
X	–	• Allows remote access to stop the controller by request • You can run the controller by input only.
X	X	Denies remote access to run/stop the controller by request.
X: check box selected –: check box deselected		

Power Cut and Restore

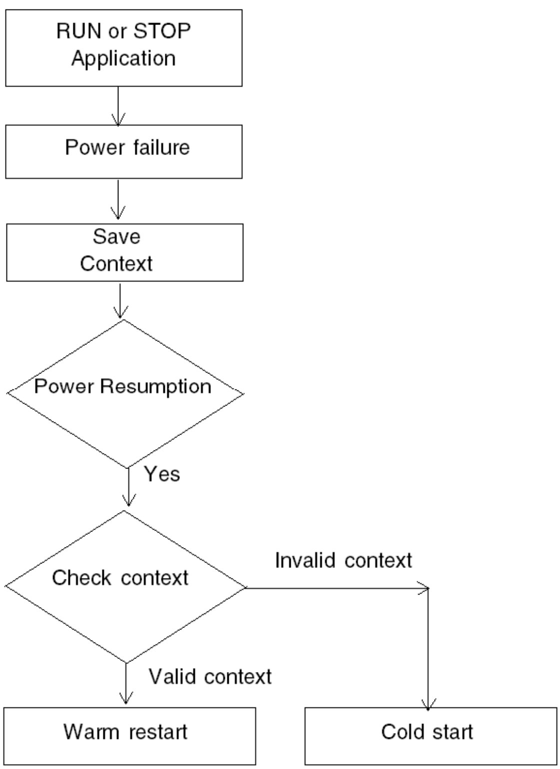
Introduction

If the duration of the outage is shorter than the power supply filtering time, it has no effect on the program which continues to run normally.

If the duration of the outage is longer than the power supply filtering time, the program is interrupted and power restoration processing is activated. The CPU then restarts in warm restart or cold start as described in the following diagram.

Illustration

Power cycle phases:



Power Supply Filtering Times

The BMX CPS 2000, BMX CPS 3500, and BMX CPS 3540T power supplies, which provide Vac power, have a filtering time of 10 ms.

The BMX CPS 2010 and BMX CPS 3020 power supplies, which provide Vdc power, have a filtering time of 1 ms.

Power Outage Processing Phases

When power to the system is lost, it recovers in 3 phases:

Phase	Description
1	On power outage, the system saves the application context, the values of application variables, and the state of the system on internal flash memory.
2	The system sets all the outputs into fallback state (state defined in configuration).
3	On power restoral, some actions and checks are done to verify if warm restart is available: • restore internal flash memory application context • verify application and context validity If all checks are correct a warm restart, page 523 is performed, otherwise a cold start, page 519 is carried out.

Cold Start

Overview

A cold start is an initialization initiated by the **Reset** button of the power supply or the **Cold start** command in Control Expert.

The consequence of a cold start is the re-initialization of all the variables. They get their default values.

NOTE: After an application download the variables are reinitialized like a cold start.

CPU Cold Start Causes and States

Cold start causes and resulting CPU states:

Cause	Resulting CPU State
End of the application download.	STOP
Application restored from flash memory is different than the one in the non-persistent application RAM. Use case: • application restored from a memory card if a compatible memory card is in the card slot • application restored from the CPU flash memory	STOP(1.)
Application restored from persistent memory with Control Expert command **PLC > Project backup > ....** is different than the one in the non-persistent application RAM: • application restored from a memory card if a compatible memory card is in the card slot • application restored from the CPU flash memory	STOP(1.)
Power supply **RESET** button pressed.	STOP(1.)
Power supply **RESET** button pressed less than 500 ms after a power down.	STOP(1.)
Power supply **RESET** button pressed after a CPU detected error, except in the case of a watchdog detected error (halt state).	STOP(2.)
Init requested with one of the 3 following means: • `%S0` system bit set to 0 • INIT request • **Cold Start** command in Control Expert	The CPU does not change its state. It only initializes the application. It is a simulation of cold start.

Cause	Resulting CPU State
Restoral after power down with a loss of context.	STOP(1.)
1. CPU state is set to RUN if **Automatic start in Run** option is selected. **2. Automatic start in Run** option does not set the CPU to RUN state.	

Loading or transferring an application to the CPU involves initialization of unlocated variables.

You need to assign a topological address to the data if the process requires keeping the current values of the data when transferring the application.

To save the located variables, avoid the initialization of the `%MWi` by unchecking **Initialize %MWi on cold start** parameter in the CPU configuration screen.

> **NOTE:** Pressing the **RESET** button on the power supply resets `%MWi` and initial values are loaded.

> **NOTE:** Do not press the **RESET** button on the power supply if you do not want `%MWi` to be reset and loaded with initial values.

Executing a Cold Start

Use these steps to perform a cold start:

Phase	Description
1	The startup is performed in RUN or in STOP state depending on one of the 2 following conditions: • The status of the **Automatic start in Run** parameter defined in the CPU configuration. If the parameter is selected, the start will be performed in RUN. • The state of the I/O defined in the **Run/Stop input** parameter in the CPU configuration. Program execution is resumed at the start of the cycle.
2	The system carries out the following: • Disable FAST, AUX, and event tasks. • MAST task is executed until the end of data initialization. • Initialize data (bits, I/O image, words, and so on) with the initial values defined in the data editor (value set to 0 if no other initial value has been defined). For `%MW` words, the values can be retrieved on a cold start when these conditions are met: ◦ The **Initialize %MWi on cold start** parameter is not checked in the CPU configuration screen, ◦ The internal flash memory has a valid backup (see %SW96). **NOTE:** If the number of `%MW` words exceeds the backup size during the save operation the remaining words are set to 0. • Initialize elementary function blocks (initial data). • Initialize data declared in the DFBs: either to 0 or to the initial value declared in the DFB type. • Initialize system bits and words. • Position charts to initial steps. • Cancel any forcing action. • Initialize message and event queues. • Send configuration parameters to all I/O and application-specific modules.
3	To start a cycle, the system performs these tasks: • Relaunch the MAST task with the `%S0` (cold start) and `%S13` (first cycle in RUN) system bits set to 1. `%SW10` (first cycle after cold start) system word is set to 0. • Reset the `%S0` and `%S13` system bits to 0 and set each bit of `%SW10` system word to 1 at the end of this first cycle of the MAST task. • Activate the FAST and AUX tasks and event processing at the end of the first cycle of the MAST task.

Processing a Cold Start by Program

Test `%SW10.0` system bit to detect a cold start and adapt the program consequently.

NOTE: It is possible to test the `%S0` system bit on the first execution cycle if the **Automatic start in RUN** parameter is selected. If it is not selected, the CPU starts in STOP state and the bit `%S0` switches to 1 on the first cycle after start (not visible for the program).

Output Changes

As soon as a power outage is detected the outputs are set in the fallback position configured (programmed fallback value or current value).

On power down, the outputs are not driven and remain at 0.

After power restoral, the outputs remain at 0 until they are updated by the task.

Warm Restart

Introduction

A warm start is initiated by a power cut.

After a warm restart, the variables get the values that they had before the power cut as a restore is done by the PLC.

Executing a Warm Restart

Phase	Description
1	Program execution does not resume from the element where the power outage occurred. The remaining program is discarded during the warm restart. Each task restarts from the beginning.
2	The system carries out the following: • Restore the application variable values, • Set `%S1` system bit to 1. • Initialize message and event queues, • Send configuration parameters to all I/O and application-specific modules, • If the application was reserved, the CPU removes the reservation. • Reset communication. • If needed, the CPU configures the I/O modules with the current adjustment parameters. • Disable FAST, AUX, and event tasks.
3	The system performs a restart cycle during which it: • Restarts the MAST task from beginning of cycle, • Sets `%S1` system bit to 0 when the MAST task is completed. • Enable FAST, AUX, and event tasks at the end of the first MAST task cycle. • CPU state set to the value before power down. If the CPU was in HALT state, it is set to STOP state.

Processing a Warm Restart by Program

On warm restart, if the application needs to be processed in a particular way, the program needs to test that `%S1` system bit is set to 1 at the start of the MAST task program.

SFC Warm Restart Specific Features

The warm start on Modicon M580 CPU is not considered as a real warm start by the CPU. SFC interpreter does not depend on tasks.

SFC publishes a `ws_data` memory area to the OS that contains SFC section-specific data to be saved on power down.

At the beginning of chart processing the active steps are saved to ws_data and processing is marked to be in a section that is essential to the application. At the end of chart processing the essential section is unmarked.

If a power down hits into the essential section, it could be detected if this state is active at the beginning (as the scan is aborted and MAST task is restarted from the beginning). In this case, the workspace may be inconsistent and is restored from the saved data.

Additional information from *SFCSTEP_STATE* variable in located data area is used to reconstruct the state machine.

When a power down occurs, the following is performed:

- During first scan, `%S1` = 1, MAST task is executed but FAST and event tasks are not executed.

On power restoral, the following is performed:

- clear chart, deregister diagnostics, keep set actions
- set steps from saved area
- set step times from *SFCSTEP_STATE*
- suppress execution of the P / P1 actions
- restores elapsed time for timed actions

  NOTE: SFC interpreter is independent, if the transition is valid, the SFC chart evolves while `%S1` = 1.

Output Changes

As soon as a power outage is detected the outputs are set in the fallback position configured: either programmed fallback value or current value.

After power restoral, the outputs remain at 0 until they are updated by the task.

M580 Hot Standby System Operation

What's in This Chapter

Overview

This chapter describes operation of the M580 Hot Standby system.

Starting an M580 Hot Standby System

Preconditions

During the start-up sequence, each PAC is assigned a Hot Standby state (Primary, Standby, or Wait) according to the:

- State of the Ethernet remote I/O network
- State of the Hot Standby link
- A/B/Clear rotary switch position, page 58
- Operating state (Run or Stop) of the CPU

On initial start-up, confirm that the:

- Hot Standby link is connected.
- PAC you start first has been fully programmed.
- A/B/Clear rotary switches on the back of the two Hot Standby CPUs are set to different positions: one to "A", the other to "B".

  NOTE: The first controller to power up becomes the primary controller, regardless of its designation as A or B.

Starting the Hot Standby System

The following chart provides the appropriate steps for starting your Hot Standby system.

<table>
<tr><th>Step</th><th colspan="3">Action</th></tr>
<tr><td>1</td><td colspan="3">Turn on power to the first backplane.
NOTE: In this example, this is the backplane with the controller A/B/Clear rotary switch position, page 58 set to “A”.</td></tr>
<tr><td>2</td><td colspan="3">Connect your PC with both Control Expert and the program you want to download.</td></tr>
<tr><td>3</td><td colspan="3">Download the program to the controller.</td></tr>
<tr><td>4</td><td colspan="3">Start the controller in that backplane.

If all necessary preconditions exist, the controller becomes the primary Hot Standby CPU.</td></tr>
<tr><td>5</td><td colspan="3">Turn on power to the second backplane.
NOTE: In this example, this is the backplane with the controller A/B/Clear switch set to “B”.</td></tr>
<tr><td>6</td><td colspan="3">If necessary, repeat steps 2 and 3 for the second controller, and download the program to it.
NOTE: If the second controller is not configured, the primary CPU automatically downloads the program to the second controller, which becomes the standby.</td></tr>
<tr><td>7</td><td colspan="3">Start the second controller.</td></tr>
<tr><td rowspan="12">8</td><td colspan="3">Check the LED display for each CPU. If both CPUs are operating as intended, the LEDs will appear as follows:</td></tr>
<tr><td>LED</td><td>First CPU (A)</td><td>Second CPU (B)</td></tr>
<tr><td>RUN</td><td>Solid Green</td><td>Solid Green</td></tr>
<tr><td>REMOTE RUN</td><td>Solid Green</td><td>Solid Green</td></tr>
<tr><td>ETH MS</td><td>Solid Green</td><td>Solid Green</td></tr>
<tr><td>ETH MS</td><td>Solid Green</td><td>Solid Green</td></tr>
<tr><td>A</td><td>Solid Green</td><td>OFF</td></tr>
<tr><td>B</td><td>OFF</td><td>Solid Green</td></tr>
<tr><td>PRIM</td><td>Solid Green</td><td>OFF</td></tr>
<tr><td>STBY</td><td>OFF</td><td>Solid Green</td></tr>
<tr><td>SRUN (safety PAC)</td><td>Solid Green</td><td>Solid Green</td></tr>
<tr><td>SMOD (safety PAC)</td><td>Solid Green</td><td>Solid Green</td></tr>
</table>

NOTE: For a description of:

- BMEH58•040 CPU LEDs, refer to *LED Diagnostics*, page 67.
- Startup states of the BMEH58•040 CPU, refer to *Hot Standby State Assignments*, page 529.

A/B/Clear Rotary Switch Role Assignment

The A/B/Clear rotary switch, page 58 assignment does not by itself determine the Hot Standby primary or standby role of a CPU. Typically, the first controller to power up becomes the primary controller, regardless of its designation as A or B; the secondary controller to power up becomes the standby.

The A/B rotary switch settings determine the role of a CPU only in the case of a simultaneous power up. In that case:

- The CPU set to “A” becomes primary.
- The CPU set to “B” becomes secondary.

Conflicting A/B/Clear Rotary Switch Role Assignment

If you mistakenly set the A/B/Clear rotary switch, page 58 to the same setting – “A” or “B” – for both Hot Standby CPUs, the first CPU to power up becomes the primary, and the second CPU to power up enters wait state.

If you mistakenly set the A/B rotary switch to “Clear” for both CPUs, both CPUs remain non-configured.

This condition can be determined by examining the following LEDs for each CPU:

If both A/B CPU Switches set to:	LED	First CPU to power-up	Second CPU to power-up
A	A	Blink Green	Blink Green
	B	OFF	OFF
	PRIM	Blink Green	OFF
	STBY	OFF	OFF
B	A	OFF	OFF
	B	Blink Green	Blink Green
	PRIM	Blink Green	OFF
	STBY	OFF	OFF
Clear	A	Blink Green	Blink Green
	B	Blink Green	Blink Green
	PRIM	OFF	OFF
	STBY	OFF	OFF

NOTE: If the A/B rotary switches for both CPUs are set to the same position (“A” or “B”), and if both CPUs start-up simultaneously, both CPUs enter wait state.

Hot Standby State Assignments and Transitions

Hot Standby State Assignments

The purpose of assigning start-up states to Hot Standby PACs is to avoid the situation where two PACs simultaneously assume the role of primary and simultaneously attempt to drive the state of remote outputs. Assignment of the primary and secondary roles for PACs is determined by the following factors:

- The health of the Hot Standby link between the PACs.
- The health of the Ethernet link between the PACs over the Ethernet RIO main ring.
- The existence of one or more Ethernet connections between each PAC and configured devices via the Ethernet RIO main ring.
- The online state, page 473 of PAC A and PAC B.
- The A/B/Clear rotary selector switch, page 58 selection on the rear of the CPU.
- The PAC state (RUN or STOP).

The following matrix describes Hot Standby state assignments for paired PACs during several start-up and run-time scenarios:

Network preconditions				**Initial state**		**Final state**	
EIO link[1]	**RIO device connections**[2]		**Hot Standby link**	**PAC_A**	**PAC_B**	**PAC_A**	**PAC_B**
	PAC_A	**PAC_B**					
OK	OK	OK	OK	Starting	Starting	Run Primary[3]	Run Standby
OK	OK	Not OK	OK	Starting	Run Primary	Run Primary[4]	Wait
OK	Not OK	OK	OK	Starting	Starting	Wait	Run Primary[4]
OK	OK	OK	OK	Run Primary	Starting	Run Primary	Run Standby
OK	OK	OK	OK	Starting	Run Primary	Run Standby	Run Primary
OK	OK	OK	Not OK	Run Primary	Starting	Run Primary	Wait
OK	OK	OK	Not OK	Starting	Starting	Run Primary	Wait
OK	OK	OK	Not OK	Starting	Run Primary	Wait	Run Primary
OK	Not OK	Not OK	OK	Starting	Starting	Run Primary	Run Standby
OK	Not OK	Not OK	OK	Run Primary	Starting	Run Primary	Run Standby
OK	Not OK	Not OK	OK	Starting	Run Primary	Run Standby	Run Primary
Not OK	Not OK	Not OK	OK	Starting	Starting	Run Primary	Run Standby

Network preconditions				Initial state		Final state	
EIO link[1]	RIO device connections[2]		Hot Standby link	PAC_A	PAC_B	PAC_A	PAC_B
	PAC_A	PAC_B					
Not OK	Not OK	Not OK	OK	Run Primary	Starting	Run Primary	Run Standby
Not OK	Not OK	Not OK	OK	Starting	Run Primary	Run Standby	Run Primary
Not OK	OK	OK	Not OK	Starting	Starting	Run Primary	Run Primary
Not OK	OK	OK	Not OK	Run Primary	Starting	Run Primary	Run Primary
Not OK	OK	OK	Not OK	Starting	Run Primary	Run Primary	Run Primary
Not OK	Not OK	Not OK	Not OK	Starting	Starting	Run Primary[3]	Run Primary[3]
Not OK	Not OK	Not OK	Not OK	Run Primary	Starting	Run Primary[3]	Run Primary[3]
Not OK	Not OK	Not OK	Not OK	Starting	Run Primary	Run Primary[3]	Run Primary[3]

1.The supplementary link between PAC A and PAC B over the RIO or DIO ring.

2. The connection between a PAC and RIO drop over the ERIO network. OK indicates the CPU recognizes at least one drop. Not OK indicates the PAC recognizes no drops for 3 seconds.

3. Priority is given to PAC designated “A” via A/B rotary selection switch on the rear of the CPU.

4. Priority is given to PAC that recognizes at least one RIO drop.

Hot Standby PAC State Transitions During Operations

A PAC in a Hot Standby system transitions between states in the following circumstances:

Transition	This transition occurs when...
Wait to Standby	All of the following exist: • PAC is in RUN state. • PAC is operating online, page 473. • Connected to a primary PAC via a Hot Standby link. • All other preconditions for standby state exists, for example: ◦ Firmware mismatch is allowed, if a firmware mismatch exists. ◦ Logic mismatch is allowed, if a logic mismatch exists. ◦ Online modifications are allowed, if modifications have been made.
Wait to Primary	All of the following exist: • PAC is operating online, page 473. • PAC is allowed to enter primary state (PAC transitions from STOP to RUN, or warm start in RUN).

Transition	This transition occurs when...
	• PAC is controlling the Ethernet RIO link, or connected via the Hot Standby link to a counterpart PAC that is not in RUN state.
Standby to Primary	One of the following exists: • The counterpart PAC enters wait or standby state. • Communication with the counterpart PAC is interrupted on both the Ethernet RIO link and the Hot Standby link. • The counterpart PAC is in primary state and receives a swap command.
Standby to Wait	The following exists: • Communication is interrupted with the counterpart PAC over the Hot Standby link for more than 3 seconds. • The ERIO link between the 2 PACs remains OK. • Online modification mismatch is not allowed, if modifications have been made. • Firmware update is not allowed, if a firmware update exists. • For safety PACs only: Online modification mismatch is allowed, if modifications have been made in the safe part of the application (SAFETY_LOGIC_ MISMATCH = 1) and maintenance mode has not been set on either the Primary PAC or Standby PAC (i.e. each PAC is operating in safety mode).
Primary to Wait	One of the following exists: • The PAC has lost communication with all (e)X80 EIO adapter modules, and the counterpart PAC is in standby state and continues to communicate with at least one (e)X80 EIO adapter module. • The PAC is designated “B” via the A/B/Clear rotary selector switch, page 58, and the counterpart PAC (also designated as “B”) is in primary state.

Primary to Standby[1]	One of the following exists:
	• During operations, all of the following occur:
	  ◦ The primary PAC is disconnected from all (e)X80 EIO adapter modules.
	  ◦ The standby PAC remains connected to at least one (e)X80 EIO adapter module.
	  ◦ The Hot Standby link between PAC A and PAC B remains healthy.
	• The primary is in Halt (because at least one task is in Halt) and the counterpart PAC is in Standby state with all tasks in RUN.
	• The primary PAC receives a swap command, and the counterpart PAC is in standby state.
	• All other preconditions for standby state exists, for example:
	  ◦ Firmware mismatch is allowed, if a firmware mismatch exists.
	  ◦ Logic mismatch is allowed, if a logic mismatch exists.
	  ◦ Online modifications are allowed, if modifications have been made.

Primary/Standby/Wait to Stop	• The PAC transitions from RUN to STOP state.
1. While the PAC is switching from Primary to Standby state, the PAC will pass to an intermediate Wait state for a duration of at least one cycle.	

Hot Standby System State Examples

Introduction

This topic presents visual examples of several Hot Standby system states. The focus of each example is the condition of the:

- Hot Standby link between controller A and controller B
- Ethernet RIO link between controller A and controller B
- Ethernet RIO connections between each controller and one or more (e)X80 EIO adapter modules over the RIO main ring

In each example, controller A is the module with its A/B/Clear rotary selector switch, page 58 set to A; controller B is the module with its A/B rotary switch set to B.

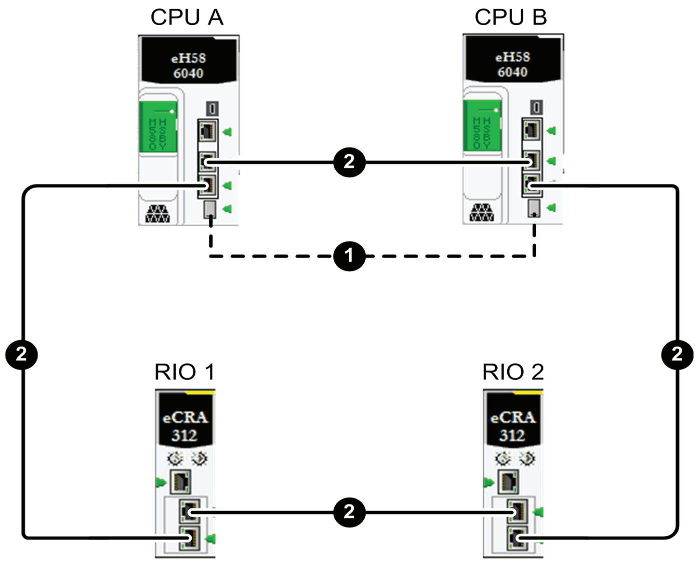
Each example presumes that every other necessary precondition exists for Hot Standby system operation. For example:

- If a firmware mismatch exists, the `FW_MISMATCH_ALLOWED` flag is set.
- If a logic mismatch exists, both the `LOGIC_MISMATCH_ALLOWED` flag and the **Online modification in RUN or STOP** parameter are set.
- For safety PACs only: If a logic mismatch and safe logic mismatch exist, the `LOGIC_MISMATCH_ALLOWED` flag, the **Online modification in RUN or STOP** parameter and the Maintenance mode are set.

All Communication Links are OK for both Controllers

In this example, all Hot Standby system connections are operational:

Communication link	Controller A	Controller B
Hot Standby link between controller A and controller B	OK	OK
Ethernet RIO link between controller A and controller B	OK	OK
Ethernet RIO connections between controller and one or more (e)X80 EIO adapter modules	OK	OK



1 Hot Standby fiber optic link between controller A (CPU A) and controller B (CPU B)

2 Ethernet RIO main ring

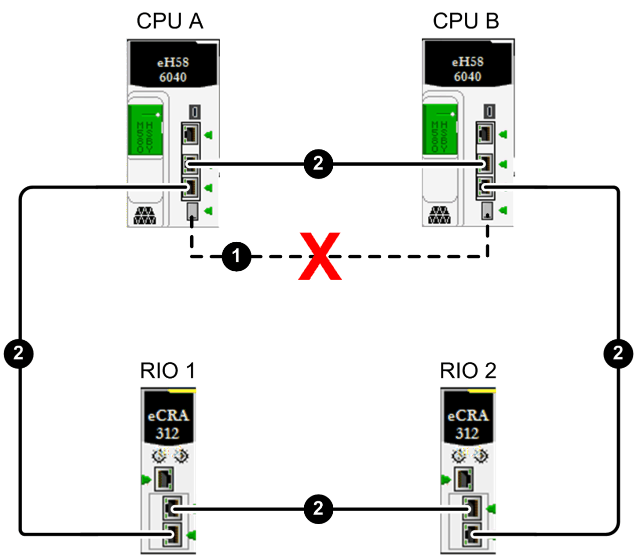
In this example, controller A and controller B enter the following Hot Standby states:

If this Hot Standby system state arises during:	Controller A and Controller B perform the following roles:
Sequential start-up of ontroller A and controller B	• The first controller to start up is primary. • The second controller to start up is standby.
Simultaneous start-up of controller A and controller B	• controller A is primary. • controller B is standby.
Run-time	• The primary controller remains primary. • The standby controller remains standby.

Hot Standby Link is Not OK for both Controllers

In this example, the Hot Standby link is not operational in both directions, from controller A to controller B and from controller B to controller A. All other Hot Standby system connections are functioning:

Communication link	Controller A	Controller B
Hot Standby link between controller A and controller B	Not OK	Not OK
Ethernet RIO link between controller A and controller B	OK	OK
Ethernet RIO connections between controller and one or more (e)X80 EIO adapter modules	OK	OK



1 Hot Standby fiber optic link between controller A (CPU A) and controller B (CPU B)

2 Ethernet RIO main ring

X Indicates a broken communication link

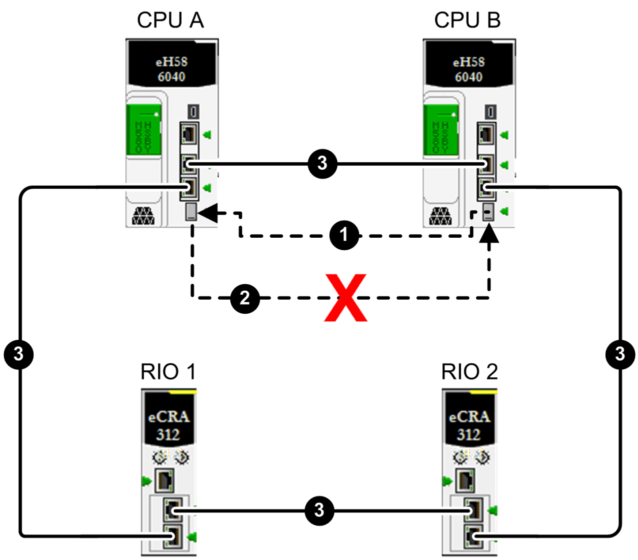
In this example, controller A and controller B enter the following Hot Standby states:

If this Hot Standby system state arises during:	Controller A and Controller B perform the following roles:
Sequential start-up of controller A and controller B	• The first controller to start up is primary. • The second controller to start up enters wait state, because there can be no standby controller if the Hot Standby link is not operational.
Simultaneous start-up of controller A and controller B	• Controller A is primary. • Controller B enters wait state.
Run-time	• The primary controller remains primary. • The standby controller enters wait state.

Hot Standby Link is Not OK for One Controller and is OK for the Other Controller

In this example, a one-directional break exists in the fiber optic cable used to implement the Hot Standby link. controller A receives transmissions from controller B over the Hot Standby link, but controller B does not receive transmissions from controller A over the link. All Ethernet RIO connections are OK for both controllers:

Communication link	Controller A	Controller B
Hot Standby link between controller A and controller B	OK	Not OK
Ethernet RIO link between controller A and controller B	OK	OK
Ethernet RIO connections between controller and one or more (e)X80 EIO adapter modules	OK	OK



1 Operational Hot Standby fiber optic link from controller B (CPU B) to controller A (CPU A)

2 Broken Hot Standby fiber optic link from controller A (CPU A) to controller B (CPU B)

3 Ethernet RIO main ring

X Indicates a broken communication link

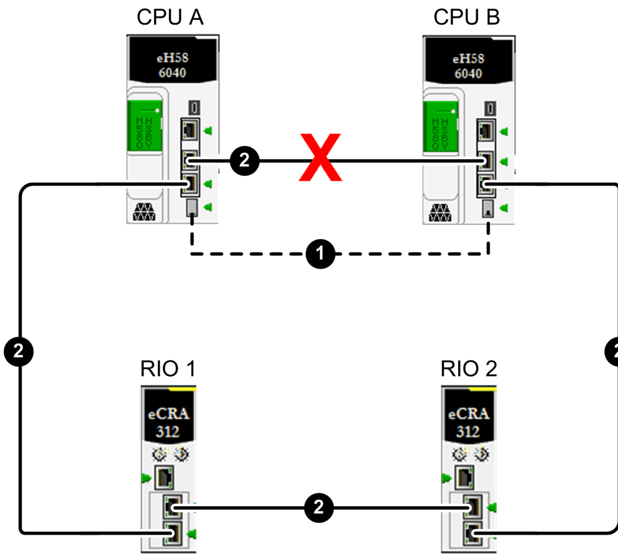
In this example, controller A and controller B enter the following Hot Standby states:

If this Hot Standby system state arises during:	Controller A and Controller B perform the following roles:
Sequential start-up of controller A and controller B	• The first controller to start up is primary. • When controller A starts up (after controller B), it is standby. • When controller B starts up (after controller A) it enters wait state.
Simultaneous start-up of controller A and controller B	• Controller A is primary. • Controller B enters wait state.
Run-time	• Controller A remains primary and controller B enters wait state. – or – • Controller B remains primary and controller A remains standby.

One Break Exists in the Ethernet RIO Main Ring

In this example, a single break exists in the Ethernet RIO main ring. Although the break occurs in the segment between the two controllers, in this example, the break could be located at any point along the Ethernet RIO main ring (2). All other Hot Standby system connections are functioning:

Communication link	Controller A	Controller B
Hot Standby link between controller A and controller B	OK	OK
Ethernet RIO link between controller A and controller B	OK[1]	OK[1]
Ethernet RIO connections between controller and one or more (e)X80 EIO adapter modules	OK	OK
1. RSTP calculates and implements a redundant path between controller A and controller B in case of a single break in the Ethernet RIO main ring.		



1 Hot Standby fiber optic link between controller A (CPU A) and controller B (CPU B)

2 Ethernet RIO main ring

X Indicates a broken communication link

In this example, controller A and controller B enter the following Hot Standby states:

If this Hot Standby system state arises during:	Controller A and Controller B perform the following roles:
Sequential start-up of controller A and controller B	• The first controller to start up is primary. • The second controller to start up is standby.
Simultaneous start-up of controller A and controller B	• Controller A is primary. • Controller B is standby.
Run-time	• The primary controller remains primary. • The counterpart controller remains standby.

Two Breaks in the Ethernet RIO Main Ring Isolate One Controller

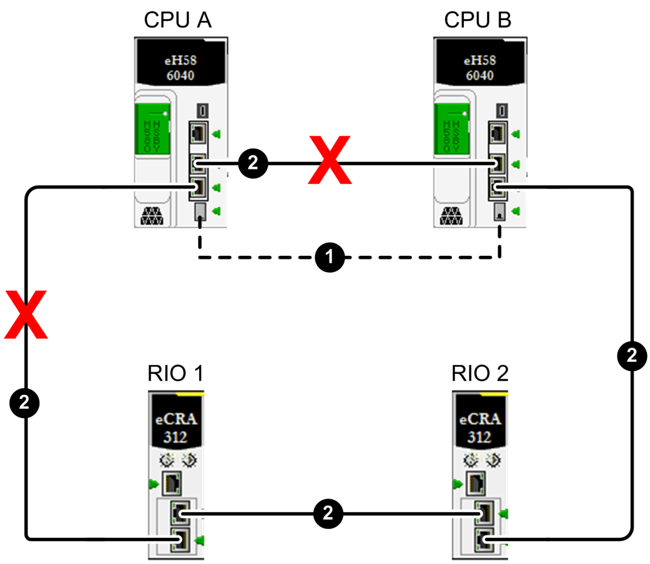
In this example, two breaks in the Ethernet RIO main ring have the following effects:

- Loss of the Ethernet RIO link between the controllers

- Isolation of controller A from the (e)X80 EIO adapter modules on the Ethernet RIO main ring

The Hot Standby link remains operational.

Communication link	Controller A	Controller B
Hot Standby link between controller A and controller B	OK	OK
Ethernet RIO link between controller A and controller B	Not OK	Not OK
Ethernet RIO connections between controller and one or more (e) X80 EIO adapter modules	Not OK	OK



1 Hot Standby fiber optic link between controller A (CPU A) and controller B (CPU B)

2 Ethernet RIO main ring

X Indicates a broken communication link

In this example, controller A and controller B enter the following Hot Standby states:

If this Hot Standby system state arises during:	Controller A and Controller B perform the following roles:
Sequential start-up of controller A and controller B	• Controller A starts up as primary. • Controller B starts up as standby.
Simultaneous start-up of controller A and controller B	• Controller A is primary. • Controller B is standby.
Run-time	• Controller B remains or becomes primary. • Controller A enters standby state.

This example occurs due to a double RIO cable break. (The first error was not detected or not treated.) The M580 Hot Standby system is not multi-RIO cable break-tolerant. Instead, the primary controller (A) isolates from the RIO drops, and the standby controller (B) can still view the primary controller and, therefore, cannot take control. controller A must check all drops before surrendering its primary role and during this phase, may read default input values (flagged by input or drop health diagnostics), which are transferred to the standby controller (B) and reused by controller B when it becomes primary.

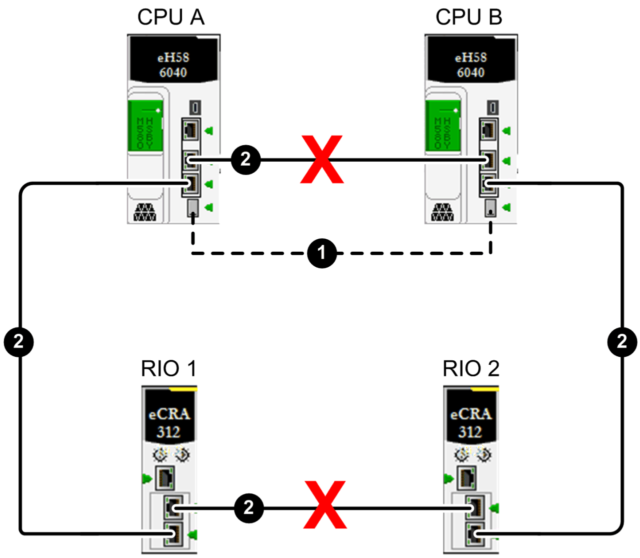
To summarize:

- Consider the health diagnostics when you design the logic.
- Perform maintenance as soon as possible when a first error is detected.
- Delay the last valid value of the inputs in the logic if this type of scenario is required.

Two Ethernet RIO Main Ring Breaks Cause Controllers to be Connected to Different Sets of Ethernet RIO Devices

In this example, two breaks exist in the Ethernet RIO main ring, causing the loss of the Ethernet RIO link between controller A and controller B. The location of the breaks cause each controller to be connected to a different collection of (e)X80 EIO adapter modules on the Ethernet RIO main ring. The Hot Standby link remains operational:

Communication link	Controller A	Controller B
Hot Standby link between controller A and controller B	OK	OK
Ethernet RIO link between controller A and controller B	Not OK	Not OK
Ethernet RIO connections between controller and one or more (e)X80 EIO adapter modules	OK	OK



1 Hot Standby fiber optic link between controller A (CPU A) and controller B (CPU B)

2 Ethernet RIO main ring

X Indicates a broken communication link

In this example, controller A and controller B enter the following Hot Standby states:

If this Hot Standby system state arises during:	Controller A and Controller B perform the following roles:
Sequential start-up of controller A and controller B	• The first controller to start up is primary. • The second to start up is standby.
Simultaneous start-up of controller A and controller B	• Controller A is primary. • Controller B is standby.
Run-time	• The primary controller remains primary. • The standby controller remains standby.

Executing Hot Standby Commands

Introduction

This topic shows you how to execute Hot Standby commands for an M580 BMEH58•040 or BMEH58•040S CPU. Hot Standby commands can be executed using:

- The Control Expert graphical user interface CPU configuration screens, which include:
  - The **Task** tab of the **Animation** window.
  - The **Hot Standby** window.
- The `T_M_ECPU_HSBY` and `T_M_ECPU_HSBY_STS` DDTs, which can be called using:
  - Program logic.
  - An **Animation Table**, where you can use the **Force** and **Modification** commands.

  NOTE: The M580 Hot Standby system does not support the use of the Quantum Hot Standby elementary function blocks (EFBs), including: `HSBY_RD`, `HSBY_ST`, `HSBY_WR` and `REV_XFER`. Instead, these functions are directly managed by DDDT commands.

For information on how to operate the non-Hot Standby functions for the CPU, refer to the *M580 Hardware Reference Manual* (see Modicon M580, Hardware, Reference Manual).

Hot Standby Commands

You must confirm that the standby controller is ready to assume the primary role before executing a swap command.

Verify that the value of the REMOTE_HSBY_STS.EIO_ERROR bit of the standby controller is 0 before you execute a swap command (either by application logic or in Control Expert).

Refer to the *EcoStruxure™ Control Expert Program Languages and Structure Reference Manual* (see EcoStruxure™ Control Expert, System Bits and Words, Reference Manual) for more details on the %SW182-%SW183 and %SW176-%SW177 system words.

The M580 BMEH58•040 and BMEH58•040S CPUs support the following Hot Standby commands:

Command	Description	Executable on Primary or Standby	Supported by:	
			DDDT	GUI
CMD_APP_TRANSFER[4]	Transfers the application in the primary PAC to the standby PAC. **NOTE:** The backup application resides in flash memory or on the SD memory card of the PAC. It is created either by the **PLC > Project Backup... > Backup Save** command, or by setting the %S66 system bit (Application Backup) to 1.	Both	X	X
CMD_COMPARE_INITIAL_VALUE	Compares the initial values of variables included in the Hot Standby data exchange.	Both (in RUN mode)	X	–
CMD_RUN_AFTER_TRANSFER	Places the primary PAC into RUN operating mode upon completion of transfer of application to standby PAC.	Primary only	X	–
CMD_RUN_REMOTE	Places the remote[1] PAC into RUN operating mode. Executable only on the primary CPU.	Primary only	X	X[3]
CMD_STOP_REMOTE	Places the remote[1] PAC into STOP operating mode.	Primary only	X	X[3]
CMD_SWAP	Manually performs a Hot Standby switchover. The primary goes into wait; the standby goes into primary; then the wait goes into standby. Executable on both the primary and the standby CPU. **NOTE:** • This command is designed to be used by the application in response to detected errors. It is not intended to be used for periodic switchovers. • If the application has to switchover periodically, the period between switchovers must not be less than 120 seconds.	Both	X	X[3]
FW_MISMATCH_ALLOWED	When changes have been made to the firmware in the primary CPU, this command lets the standby CPU continue to operate as standby. If this command is set to 0, the standby goes into wait state.	Primary only	X	–
LOGIC_MISMATCH_ALLOWED[4]	When changes have been made to the application in the primary CPU (for example, as a a result of CCOTF changes), this command lets the standby CPU continue to operate as standby. If this	Primary only	X	–

Command	Description	Executable on Primary or Standby	Supported by:	
			DDDT	GUI
	command is set to 0, the standby goes into wait state.			
PLCA_ONLINE	Lets the CPU with its A/B/Clear rotary selector switch, page 58 set to “A” serve as either primary or standby, depending on other operating conditions. If set to 0, PAC A goes into either wait or stop state.	PAC A only	X	X[2]
PLCB_ONLINE	Lets the CPU with its rotary switch set to “B” serve as either primary or standby, depending on other operating conditions. If set to 0, PAC B goes into either wait or stop state.	PAC B only	X	X[2]

X: Command is supported.

–: Command is not supported.

1. *Remote* refers to the PAC to which your PC and Control Expert is not connected.

2. In the CPU configuration window **Hot Standby** tab.

3. In the CPU configuration window **Animation > Task** tab.

4. These commands can be executed only if the remote CPU is also the standby CPU.

Memory Usage

Introduction

The memory usage function is used to view:

- The physical distribution of the PAC memory.
- The space taken up in the memory by a project (data, program, configuration, system and diagnostic).

It can also be used to reorganize the memory where possible.

NOTE: The memory usage screen is not available in simulation mode. This screen is only available in standard mode when you have built the application.

Procedure

To access the memory usage details of the PAC:

Step	Action
1	Select **PLC > Memory Consumption:**. The **Memory usage** window opens. The memory usage statistics of a project can only be accessed if you have generated its executable in advance.
2	To optimize memory organization, click **Pack**.

NOTE: If the application has been built and if it is in NOT BUILT state due to a program modification, the screen is accessible, but it corresponds to the application built previously. Modifications will be taken into account at the next build.

Description of the parameters

The following information fields are available:

Parameter	Description
User Data	This field indicates the memory space (in words) taken up by user data (objects relating to configuration): • **saved Data**: located data associated with the processor (%M, %MW, %S, %SW, etc.) or the input/output modules. This data is retained by the CPU in the event of a CPU warm start. • **saved Declared Data**: unlocated data (declared in the data editor) that is retained by the CPU in the event of a CPU warm start. • **unsaved Declared Data** unlocated data (declared in the data editor) that is not retained by the CPU in the event of a CPU warm start.
User program	This field indicates the memory space (in words) taken up by the project program: • **Constants**: static constants associated with the processor (%KW) and the input/output modules; initial data values, • **Executable code**: executable code of the project program, EFs, EFBs and DFB types, • **Upload information**: information for uploading a project (graphic code of languages, symbols, etc.).
Other	This field indicates the memory space (in words) taken up by other data relating to the configuration and the project structure: • **Configuration**: other data relating to configuration (Page0 for a Quantum PAC, hardware configuration, software configuration), • **System**: data used by the operating system (task stack, catalogs, etc.), • **Diagnostic**: information relating to process or system diagnostics, diagnostics buffer, • **Data Dictionary**: dictionary of symbolized variables with their characteristic (address, type....)
Internal memory	This field shows the organization of the PAC's internal memory, for both program and data storage. It indicates the memory space available (**Total**), the largest possible contiguous memory space (**Greatest**) and the level of **Fragmentation** (due to online modifications).
Pack	This command is used to reorganize the memory structure.

Memory re-organization

Memory re-organization is activated using the **Pack** command.

Memory re-organization can be performed in online or offline mode (Even if the PAC is in Run or in Stop).

NOTE: Certain blocks cannot be moved in online mode. You will attain a lower level of fragmentation by re-organizing the memory in offline mode.

M580 Hot Standby Diagnostics

What’s in This Chapter

Overview

This chapter describes M580 Hot Standby diagnostic tools provided by the:

- BMEH58•040 CPU Hot Standby LEDs
- Control Expert graphical user interface

Control Expert M580 Hot Standby Diagnostics

Overview

This sections described diagnostic tools for the M580 BMEH58•040(S) Hot Standby CPUs that are available in Control Expert.

M580 Hot Standby System Diagnostics in Control Expert

Introduction

EcoStruxure Control Expert provides M580 Hot Standby System diagnostic information in these GUI screens:

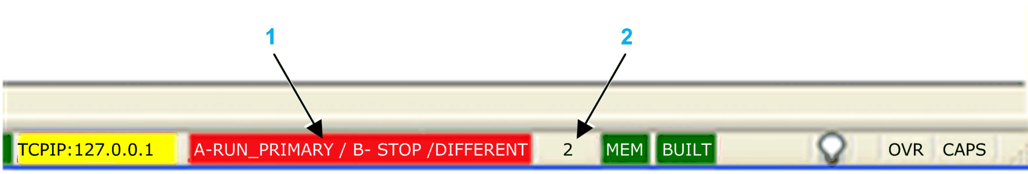
- the Hot Standby status viewer embedded in the EcoStruxure Control Expert Task Bar.
- the **Information** tab of the controller **Animation** window

Hot Standby Status Viewer

When EcoStruxure Control Expert is connected to the Hot Standby system, it displays the Hot Standby status of each controller, including:

- The status of controllers A and B.
- The comparative state of logic running in the standby controller.
- If a logic mismatch exists, the number of modifications, page 471 made to the application running in the primary controller.

The Hot Standby Status Viewer looks like this:



1 Hot Standby status

2 Number of changes

The status values for controllers A and B include:

- RUN_PRIMARY
- RUN_STANDBY
- STOP
- WAIT

Also presented is the logic state of the standby controller, which can be either:

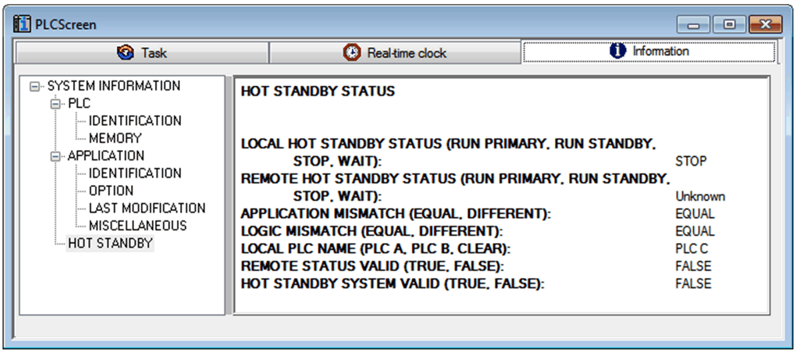
- EQUAL (green background): There is no logic mismatch.
- DIFFERENT (red background): Online changes have been made to the primary controller application that have not been transferred to the standby controller.

An additional status information is displayed when at least one task (MAST, FAST, or SAFE) is not synchronized between the Primary and the Standby controllers: TASK NOSYNC (red background):

In this case, analyze "MAST_SYNCHRONIZED", "FAST_SYNCHRONIZED" and "SAFE_SYNCHRONIZED" data provided in T_M_ECPU_HSBY DDT to detect the task which is not synchronized.

Hot Standby Information Tab

Use the controller configuration window **Animation > Information** tab to view the status of the Hot Standby system:



The **Information** tab contains one word of status data:

Hot Standby status of the local controller: • Primary • Standby • Stop • Wait	Local controller name (position of A/B/Clear rotary selector switch, page 58): • PLC A • PLC B • CLEAR
Hot Standby status of the remote controller: • Primary • Standby • Stop • Wait	Remote status valid: • True • False
Application mismatch status: • Equal • Different	Hot Standby system valid: • True • False
Logic mismatch status: • Equal • Different	–

Synchronizing Configuration of Distributed Equipment

Introduction

The M580 BMEH58•040(S) Hot Standby CPU DTMs include a **Hot Standby Synchronization** page where you can synchronize the storage of configuration (.prm) files for distributed equipment in the primary and standby CPUs. Distributed equipment configuration files stored in Hot Standby CPUs are used by the fast device replacement (FDR) service.

Use this page to:

- View the synchronization status of distributed equipment configuration files stored by the Hot Standby system CPUs.
- Stop synchronization.
- Force a manual synchronization.

The standby CPU synchronizes with the primary CPU by pulling data every 10 seconds to verify that the data in the standby has been updated in the primary. If the standby unsuccessfully synchronizes with the primary, it continues polling the primary every 10 seconds.

If the data in the standby and primary PACs are different, an application mismatch, page 463 condition exists. In this case, synchronization stops and a synchronization error is detected in the standby CPU.

NOTE:

- When the standby CPU is offline, it does not synchronize.
- If you disable the TFTP service, Hot Standby synchronization cannot be performed, because this function is based on TFTP.

Accessing the Hot Standby Synchronization Page

To access the CPU **Hot Standby Synchronization** page, follow these steps:

Step	Action
1	In Control Expert, open the **DTM Browser** (**Tools** > **DTM Browser**).
2	Right-click the CPU in the **DTM Browser**.
3	Select **Connect**.
4	Right-click the CPU in the **DTM Browser**.
5	Select **Device menu** > **Diagnosis**.
6	Click the **Hot Standby Synchronization** tab.

Using the Hot Standby Synchronization Page

The **Hot Standby Synchronization** page presents the following parameters and controls:

Parameter	Description
Refresh Every 500ms	Select this to display synchronization data in this page, and refresh displayed data every 500ms.
Status area:	
Synchronizing	• True: Synchronization is executing. • False: Synchronization is not executing.
Synchronized	• True: Data in both primary and standby are synchronized. • False: Data in both primary and standby are not synchronized.
Error Status	• Green: No synchronization error is detected. • Red: A synchronization error has been detected.
Manual Synchronization > Stop Synchronization area:	
Stop Synchronization Service	Select this then click **Send** to stop the synchronization service. **NOTE:** To re-start the synchronization service, select one of the **Force Manual Synchronization** options (below), then click **Send**.
Manual Synchronization > Force Manual Synchronization area:	
Copy Files from Standby to Primary	Select this then click **Send** to push DIO device configuration (.prm) files from the standby CPU to the primary.
Copy Files from Primary to Standby	Select this then click **Send** to pull DIO device configuration (.prm) files from the primary CPU to the standby.
Clear Files in Primary	Select this then click **Send** to delete the DIO device configuration (.prm) files from the primary. If synchronization is enabled, the standby CPU synchronizes with the primary and any DIO device configuration files in the standby are also deleted.

M580 Hot Standby System Diagnostics

Overview

This section describes the diagnostic messages that can be displayed by the M580 Hot Standby system.

M580 Hot Standby System Diagnostics

Introduction

The M580 Hot Standby system continuously monitors the system state, and adds to its diagnostic buffer an entry for each detected error or change of state event. You can view and handle this collection of events using the following tools:

- **Alarm Viewer** web page (see Modicon M580, Hardware, Reference Manual), for events relating to the selected CPU.
- **Diagnostic Viewer** in Control Expert (see EcoStruxure™ Control Expert, Operating Modes), for detected events relating to the Hot Standby system.

M580 Hot Standby System Messages

Each detected system event presents:

- A message describing the event type.
- An explanatory text symbol entry, more particularly describing the event.
- A numeric decimal identifier, representing the combination of message and symbol.

The M580 Hot Standby system can display the following messages

ID (dec)	Message (Event Text)	Symbol (Event Type)	Possible Cause
14101	Switch from Wait to Primary	No Error	–
14102	Switch From Wait to Standby	Linked to Primary. No Error	–
14103	Switch from Standby to Primary	No remote PLC connection	No Hot Standby link and EIO link between CPUs.
14104	Switch from Standby to Primary	Remote PLC not Primary	• Loss of power on former primary. • Former primary stopped. • Error detected on former primary.

ID (dec)	Message (Event Text)	Symbol (Event Type)	Possible Cause
14105	Switch from Standby to Wait	Hsby Link Error	• Break in Hot Standby link cable • Transceiver inoperable in either CPU.
14106	Switch from Standby to Stop	PLC not in RUN	Standby CPU stopped.
14107	Switch from Primary to Wait	Loc RIO err and no peer RIO err	Former primary CPU lost connection to all (e) X80 EIO adapter modules; former standby (now primary) CPU maintains connection to at least one (e)X80 EIO adapter module.
14108	Switch from Primary to Wait	Swap Command	Former primary CPU received swap command.
14109	Switch from Primary to Stop	PLC not in RUN	Former primary CPU stopped (PAC in STOP or one task in HALT)
14110	Switch from Primary to Wait	PLC_B linked to Primary	–
14111	Peer PLC disconnection on RIO Link	RIO Link Error	Two breaks in Ethernet RIO cable have isolated the remote CPU.
14112	Peer PLC disconnection on Hsby Link	Hsby Link Error	• Break in Hot Standby link cable • Transceiver inoperable in either CPU.
14113	Mismatch Error	FW mismatch	Different firmware versions in each CPU.
14114	Mismatch Error	Logic mismatch	Different application logic revisions running in each CPU.
14115	Mismatch Error	Application mismatch	Different applications running in each CPU.
14116	Degraded Hsby Data transfer	Data Layout mismatch	Online changes to data structure have been made to primary CPU, but not transferred to standby.
14117	Bad peer rotary switch config	Not in a PLC_A and PLC_B config	Rotary switch settings do not specify an A and a B PAC.
14118	Power supply error	Loss of redundancy	One of the BMXCPS4002 redundant power supply units is no longer functioning.

M580 System Words

Modicon M580-specific System Words %SW132 to %SW167

Diagnostic System Words

⚠WARNING

UNEXPECTED APPLICATION BEHAVIOR

Do not use system objects (%Si, %SWi) as variables when they are not documented.

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

Control Expert presents the following M580-specific system words you can use when diagnosing the state of your M580 Hot Standby system:

- **%SW132** to **%SW134**: CPU MAC Address.
- **%SW135** to **%SW137**: CPU serial number
- **%SW146** and **%SW147**: SD card serial number
- **%SW160** to **%SW167**: Detected errors for racks 0...7

For a more detailed description of these system words, refer to the M580 section (see EcoStruxure™ Control Expert, System Bits and Words, Reference Manual) of the *EcoStruxure™ Control Expert System Bits and Words Reference Manual*.

Replacing M580 Hot Standby CPUs

What's in This Chapter

Replacing Hot Standby Hardware Modules

Overview

Replace the modules in this order:

- Standby PAC (PAC B in this example)
- Primary PAC (PAC A in this example)

Replacing PAC B Procedure

⚠ WARNING

SYSTEM NO LONGER ACTIVE NOR REDUNDANT

In a HotStandBy system, before stopping one of the controllers, confirm that no critical operation is in progress because the system is inactive and non-redundant.

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

Replace the modules in the standby PAC:

Step	Action
1	Confirm that the application program running on the M580 Hot Standby PAC has been exported in the ZEF format and is available on the computer. If not, upload the application program from one of the two PACs to Control Expert.
2	Export the application in the ZEF format on the Control Expert workstation.
3	If not yet installed, install Unity Pro XL version 11.0 (or any subsequent supporting version(s)). **NOTE:** Unity Pro is the former name of Control Expert for version 13.1 or earlier.

4	Stop the standby PAC (PAC B) and power it off. **NOTE:** At this point, the system is no longer operating redundantly.
5	Disconnect the Hot Standby sync link cable from PAC B.
6	Replace hardware or update the PAC B firmware with version 2.10 or any subsequent supporting version(s).
7	Confirm that there is no program in PAC B: **a.** Set the A/B/Clear rotary selector switch, page 58 to **Clear**. **b.** Power up the PAC. **c.** Wait approximately one minute, until LEDs A and B are blinking. **d.** Power down the PAC. **e.** Set the rotary switch to **B**.
8	Power on PAC B.
9	If using an SD memory card, insert the card in PAC B. (Refer to the SD memory card instructions for information about existing programs on the card.) **NOTE:** Confirm that the PAC is in a NOCONF state (see Modicon M580, Hardware, Reference Manual).
10	Import the ZEF file of the application.
11	In the **PLC Bus** editor, replace the current version of the PAC with the new firmware PAC version.
12	Select the **Online modification in RUN or STOP** check box in the PAC **Configuration** tab to enable the configuration change.
13	Rebuild the application (**Build > Rebuild All Project**) and download into PAC B. The PAC is in STOP mode.
14	Connect the Hot Standby sync link cable to PAC B.
15	Connect Control Expert to PAC A.
16	Stop PAC A. **NOTE:** The system is no longer active nor redundant.
17	Connect Control Expert to PAC B.

18	Put PAC B in RUN mode. **⚠ WARNING** **UNEXPECTED APPLICATION BEHAVIOR - LOSS OF DATA** Before you change the mode of PAC B to RUN, confirm that the application can restart with the initial values. **Failure to follow these instructions can result in death, serious injury, or equipment damage.** **NOTE:** At the end of the application download, all application data in the PAC B have their initial value.
19	Confirm that PAC B is now the primary.

Replacing PAC A Procedure

After you PAC, page 556, follow these steps to replace PAC A:

Step	Action
1	Power off PAC A, which is in STOP mode. **NOTE:** At this point, the system is no longer operating redundantly.
2	If using an SD memory card, remove it.
3	Disconnect the Hot Standby sync link cable from PAC A.
4	Replace hardware or update the PAC B firmware with version 2.10 or any subsequent supporting version(s).
5	Power on PAC A.
6	If using an SD memory card, insert it in PAC A. **NOTE:** Confirm that the PAC is in a **No Conf** state.
7	Connect the Hot Standby sync link cable to PAC A.
8	An automatic transfer from primary to standby occurs.
9	Execute a RUN command on PAC A.
10	Confirm that PAC A is now the standby.

Verifying the Network Configuration

What's in This Chapter

Using the Ethernet Network Manager

Introduction

In Control Expert, click **Tools > Ethernet Network Manager** to visualize and verify a complex network configuration. The tool can:

- provide a global view of your network
- edit IP addresses and device identifiers for (e)X80 EIO adapter modules

Use either method to access the **Ethernet Network Manager**:

- Select **Tools > Ethernet Network Manager**.
- Select **Ethernet Network Manager** in the **Project Browser**.

NOTE: The **Ethernet Network Manager** tool is available on all M580 PACs. Only devices enabled in the address server (DHCP) are controlled.

Network Topology Configuration

The **Ethernet Network Manager** tool provides a snapshot of IP address settings for devices included in network topologies that are part of your application. If the tool detects an addressing error, it displays the detected error against a red background. If the tool detects an error, you can re-configure the affected setting in Control Expert.

Parameters in the **Ethernet Network Manager**:

Parameter	Description
Name	Ethernet communication device name
Type	The device type: • Scanner • Module
Subtype	The device sub-type: • RIO/DIO

Parameter	Description
	• CRA
Profiles	The kind of control network communications: • Remote (RIO) • Distributed (DIO)
Topo address	The topological address of the device, in the sequence: bus, drop, rack, slot.
DHCP Enable	Indicates if the device is a DHCP client and receives its IP address(es) from a DHCP server (yes/no).
IP Address	The IP address, or addresses, assigned to the device. **NOTE:** Editable for scanned modules.
Subnet Mask	The subnet mask related to each assigned IP address.
Gateway Address	The IP address of the default gateway, to which messages for other networks are transmitted.
Identified By	For scanned devices, the type of network identifier - the device Name,
Identifier	The string used to identify a scanned device. The default value is the device Name. **NOTE:** Editable for scanned modules.
SNMP	For scanning devices, the IP address of up to two SNMP network manager devices.
NTP State	The role or roles of the of the CPU’s NTP service: **NOTE:** CPU firmware versions earlier than V4.01 use SNTP; CPU firmware V4.01 and any subsequent supporting version(s) use NTPv4) • Disabled (SNTP and NTPv4): The service is not enabled in the CPU configuration. • Server (SNTP): The CPU is configured as an SNTP server. • Server only (NTPv4): The CPU is configured as an NTPv4 server, but not also as a client. • Client (SNTP): The CPU is configured as an SNTP client. • Client / Server (NTPv4): The CPU is configured as both an NTPv4 client and server.
NTP Configuration	Lists the IP addresses of the SNTP or NTPv4 servers that send updates to the NTP client resident in the device: • Primary and Secondary SNTP server configured IP addresses are displayed when the CPU is configured as Client or Server. • Up to 8 NTPv4 system peer IP Addresses can be displayed, with the Preferred server identified for NTPv4, when the CPU is configured as Client / Server.

NOTE:

- The red cells indicate detected errors (defined by network management rules).
- After editing a scanned module **IP Address** or **Identifier** setting, click the validate button to save your edits.

Verifying a Hot Standby Network

Follow these steps to use the **Ethernet Network Manager** tool while building your network in Control Expert:

Step	Action
1	In Control Expert, click **Tools > Ethernet Network Manager**. A preliminary, read-only global view of your network displays.
2	Check for settings with a red background, indicating the tool has detected a configuration error.
3	Click **OK** to close the **Network Inspector** tool.
4	If the tool displayed a detected error: • in a scanning device, go to the specific device editor and change the IP configuration settings. • in a scanned device, you can edit the **IP address** and **Identifier** settings in the **Ethernet Network Manager**, or go to the specific device editor and change the IP configuration settings. When you finish your edits, run the **Ethernet Network Manager** again.
5	Add distributed equipment and/or RIO modules to the **EIO Bus**. **NOTE:** Only devices enabled in the address server (DHCP) are controlled.
6	Configure all scanners.
7	Repeat steps 1, 2, 3, and 4 until the **Ethernet Network Manager** no longer detects any errors.

Network Manager Services

The network manager starts automatically when you open the **Network Inspector** tool. The global network management system (GNMS) is responsible for global network consistency. The following checks are performed:

- GNMS verifies that all IP addresses are unique for the modules in the application.
- Each gateway that exists on your network is displayed in the network manager. By default, Control Expert notifies you if one of the gateways is missing an IP address. You can change this notification by clicking **Tools > Project Settings > General > Management of build messages > Missing gateway IP @ generates**. The options are a `warning` (default value) or nothing.
- Only a single RSTP switch can be configured as a root for a given network.
- The range of IP addresses is 1.0.0.0 … 126.255.255.255 or 128.0.0.0 … 223.255.255.255. Otherwise, an error is detected. Addresses 224.0.0.0 and up are multicast or experimental addresses. Addresses starting at 127 are loopback addresses. Addresses 169.254/16 are reserved for automatic private IP addressing (APIPA).

- The tool verifies that the network address of the IP address is valid.
- The tool verifies that the host address of the IP address is valid, including that broadcast IP addresses are blocked.
- While an M580 CPU uses *classless inter-domain routing* (CIDR), some IP addresses are not allowed to maintain compatibility:
  - in a class A network, IP addresses that end in 255.255.255
  - in a class B network, IP addresses that end in 255.255
  - in a class C network, IP addresses that end in 255
- The IP address is configured to access the gateway address. Therefore, the gateway address is within the subnetwork defined by the mask. The gateway is not accessible when it is not on the same subnetwork as the IP address.

Network Bandwidth Considerations

Control Expert alerts you when there are possible bandwidth considerations.

Ethernet RIO bandwidth:

- Control Expert displays a detected error message in the log window if the RIO bandwidth (originator -> target) or (target->originator) is greater than 8%.
- Control Expert displays a **warning** in the log window if the RIO bandwidth (originator -> target) or (target->originator) is greater than 6%.

Device network bandwidth (DIO and RIO combined):

- Control Expert displays a detected **error** in the log window if total Modbus and EIP bandwidth (originator -> target) or (target->originator) is greater than 40%.
- Control Expert displays a **warning** in the log window if total Modbus and EIP bandwidth (originator -> target) or (target->originator) is greater than 30%.

Appendices

What's in This Part

Function Blocks

What's in This Chapter

ETH_PORT_CTRL: Executing a Security Command in an Application

Function Description

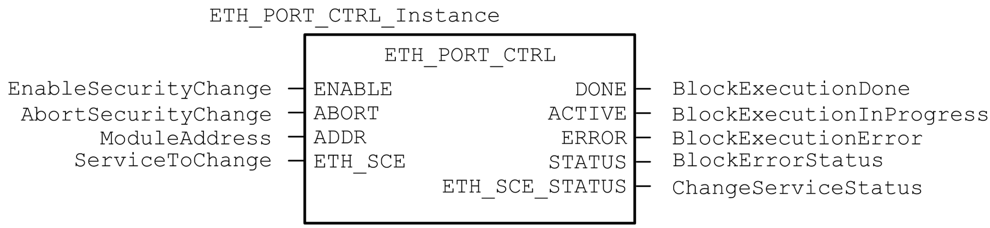
Use the ETH_PORT_CTRL function block to control the FTP TFTP, HTTPS, and DHCP / BOOTP protocols when they are enabled in the Control Expert **Security** screen (see *Modicon M580 BMENOC0301/11, Ethernet Communication Module, Installation and Configuration Guide*). (By default, these protocols are disabled.) For cyber security reasons (to help protect data against requests to modify in the monitoring mode), map the inputs on variables and on unlocated variables in which the HMI property is disabled (the variable is not in the data dictionary).

The additional parameters `EN` and `ENO` may also be configured.

NOTE: For M580 controller firmware versions 4.20 and later, if **Engineering Link Mode** is set to **Enforced** or **Filtered**, and if the ETH_PORT_CTRL function block is used to programmatically disable the HTTPS service, it will not be possible to connect Control Expert to the controller.
If you intend to programmatically disable HTTPS using the ETH_PORT_CTRL function block, first verify that your program logic allows the re-enabling of HTTPS. If HTTPS is disabled and cannot be re-enabled, you need to reset the controller.

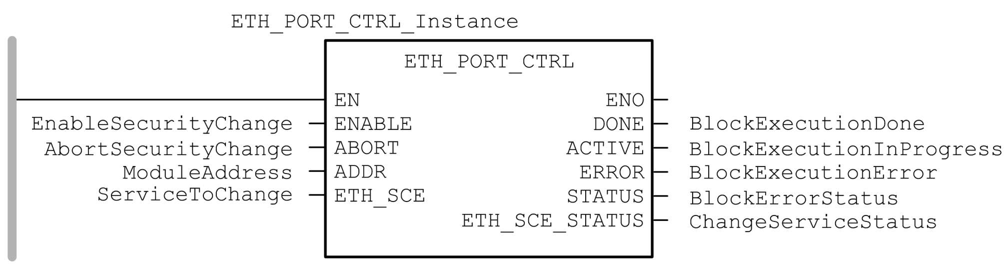
FBD Representation

Representation:



LD Representation

Representation:



IL Representation

```
CAL ETH_PORT_CTRL_Instance (ENABLE := EnableSecurityChange, ABORT :=
AbortSecurityChange, ADDR := ModuleAddress, ETH_SCE := ServiceToChange,
DONE => BlockExecutionDone, ACTIVE => BlockExecutionInProgress, ERROR
=> BlockExecutionError, STATUS => BlockErrorStatus, ETH_SCE_STATUS =>
ChangeServiceStatus)
```

ST Representation

```
ETH_PORT_CTRL_Instance (ENABLE := EnableSecurityChange, ABORT :=
AbortSecurityChange, ADDR := ModuleAddress, ETH_SCE := ServiceToChange,
DONE => BlockExecutionDone, ACTIVE => BlockExecutionInProgress, ERROR
```

```
=> BlockExecutionError, STATUS => BlockErrorStatus, ETH_SCE_STATUS =>
ChangeServiceStatus);
```

Description of Parameters

This table describes the input parameters:

Parameter	Type	Comment
ENABLE	BOOL	Set to 1 to enable the operation.
ABORT	BOOL	Set to 1 to abort the currently active operation.
ADDR	ANY_ARRAY_INT	This array contains the address of the entity for which you want to change the security state, which is the result of the ADDMX (see EcoStruxure™ Control Expert, Communication, Block Library) or ADDMX or ADDM function (see EcoStruxure™ Control Expert, Communication, Block Library). For example: • ADDM('0.0.10') for a M580 controller • ADDM('0.3.0') for a BMENOC0301 or BMENOC0311(C) module plugged in slot 3 of main rack
ETH_SCE	WORD	For each protocol, use these binary values to control the protocol: • 00: The protocol is unchanged. • 01: Enable the protocol. • 10: Disable the protocol. • 11: reserved **NOTE:** A value of 11 reports a detected error in ETH_SCE_STATUS. These bits are used for the different protocols: • 0, 1: FTP • 2, 3: TFTP (Only available for Modicon M580) • 4, 5: HTTPS Before disabling HTTPS protocol, refer to the NOTE in this topic's Function Description, page 564. • 6, 7: DHCP / BOOTP • 8...15: reserved (value = 0)
(1) To address a module in the local rack, enter 0.0.10 (controller main server address).		

This table describes the output parameters:

Parameter	Type	Comment
DONE	BOOL	Operation completed indication. Set to 1 when the execution of the operation is completed successfully.
ACTIVE	BOOL	Operation in progress indication. Set to 1 when the execution of the operation is in progress.
ERROR	BOOL	Set to 1 if an error is detected by the function block.
STATUS	WORD	Code providing the detected error identification (see EcoStruxure™ Control Expert, I/O Management, Block Library).
ETH_SCE_STATUS	WORD	For each protocol, these values contain the response to any attempt to enable or disable the FTP, TFTP, HTTPS, or DHCP / BOOTP protocols: • 0: command executed • 1: command not executed Reasons for not executing the command can be: • The communication service has been disabled by the configuration. • The communication service is already in the state requested by the command (**Enabled** or **Disabled**). • The communication service (x) is not supported by the module or is a non-existing service. These bits are used for the different protocols: • 0: FTP • 1: TFTP • 2: HTTPS • 3: DHCP / BOOTP • 4 ... 15: reserved (value = 0)

Execution Type

Synchronous:

When used on the following M580 controller modules, the ETH_PORT_CTRL function block is executed ***synchronously***. As a result, the DONE output turns **ON** as soon as the ENABLE input is set to **ON**. In this case, the ACTIVE output remains **OFF**.

- BMEP581020
- BMEP582020
- BMEP582040
- BMEP583020
- BMEP583040
- BMEP584020

- BMEP584040
- BMEP585040
- BMEP586040
- BMEH582040*
- BMEH584040*
- BMEH586040*
  * In BMEH58•040 Hot Standby controllers, verify that the ETH_PORT_CTRL function block is executed equally on both primary and standby controllers.

Asynchronous:

When used on the following modules, the ETH_PORT_CTRL function block is executed ***asynchronously*** and may take several cycles until the DONE output turns **ON**. Therefore, the ACTIVE output is set to **ON** until the completion of the ETH_PORT_CTRL function block.

- **M340 modules**:
  - BMXNOC0401
  - BMXNOE0100
  - BMXNOE0110
- **M580 modules**:
  - BMENOC0301
  - BMENOC0311(C)

How to Use the ETH_PORT_CTRL EFB

Use the `ETH_PORT_CTRL` EFB:

Step	Action
1	Set the bits of the services you want to activate in `ETH_SCE`.
2	Set `ENABLE` input to activate the EFB.
3	`ENABLE` input should be an OR between a pulse command and the `ACTIVE` output of the EFB.
4	Check `STATUS` output value: • `STATUS`<>0: There is a communication issue. • `STATUS` = 0: Check `ETH_SCE_STATUS`. The services for which the bits are set haven't been modified as they should be.

Glossary

A

adapter:

An adapter is the target of real-time I/O data connection requests from scanners. It cannot send or receive real-time I/O data unless it is configured to do so by a scanner, and it does not store or originate the data communications parameters necessary to establish the connection. An adapter accepts explicit message requests (connected and unconnected) from other devices.

B

BCD:

(*binary-coded decimal*) Binary encoding of decimal numbers.

BOOTP:

(*bootstrap protocol*) A UDP network protocol that can be used by a network client to automatically obtain an IP address from a server. The client identifies itself to the server using its MAC address. The server, which maintains a pre-configured table of client device MAC addresses and associated IP addresses, sends the client its defined IP address. The BOOTP service utilizes UDP ports 67 and 68.

C

CCOTF:

(*change configuration on the fly*) A feature of Control Expert that allows a module hardware change in the system configuration while the system is operating. This change does not impact active operations.

CIP™:

(*common industrial protocol*) A comprehensive suite of messages and services for the collection of manufacturing automation applications (control, safety, synchronization, motion, configuration and information). CIP allows users to integrate these manufacturing applications with enterprise-level Ethernet networks and the internet. CIP is the core protocol of EtherNet/IP.

CPU:

(*central processing unit*) The CPU, also known as the processor or controller, is the brain of an industrial manufacturing process. It automates a process as opposed to relay control systems. CPUs are computers suited to survive the harsh conditions of an industrial environment.

D

determinism:

For a defined application and architecture, you can predict that the delay between an event (change of value of an input) and the corresponding change of a controller output is a finite time *t*, smaller than the deadline required by your process.

Device DDT (DDDT):

A Device DDT is a DDT predefined by the manufacturer and not modifiable by user. It contains the I/O language elements of an I/O module.

device network:

An Ethernet-based network within an RIO network that contains both RIO and distributed equipment. Devices connected on this network follow specific rules to allow RIO determinism.

DFB:

(*derived function block*) DFB types are function blocks that can be defined by the user in ST, IL, LD or FBD language.

Using these DFB types in an application makes it possible to:

- simplify the design and entry of the program
- make the program easier to read
- make it easier to debug
- reduce the amount of code generated

DHCP:

(*dynamic host configuration protocol*) An extension of the BOOTP communications protocol that provides for the automatic assignment of IP addressing settings, including IP address, subnet mask, gateway IP address, and DNS server names. DHCP does not require the maintenance of a table identifying each network device. The client identifies itself to the DHCP server using either its MAC address, or a uniquely assigned device identifier. The DHCP service utilizes UDP ports 67 and 68.

DIO cloud:

A group of distributed equipment that is not required to support RSTP. DIO clouds require only a single (non-ring) copper wire connection. They can be connected to some of the copper ports on DRSs, or they can be connected directly to the CPU or Ethernet communications modules in the *local rack*. DIO clouds **cannot** be connected to *sub-rings*.

DIO:

(*distributed I/O*) Also known as distributed equipment. DRSs use DIO ports to connect distributed equipment.

DNS:

(*domain name server/service*) A service that translates an alpha-numeric domain name into an IP address, the unique identifier of a device on the network.

DRS:

(*dual-ring switch*) A ConneXium extended managed switch that has been configured to operate on an Ethernet network. Predefined configuration files are provided by Schneider Electric to downloaded to a DRS to support the special features of the main ring / sub-ring architecture.

DSCP:

(*differentiated service code points*) This 6-bit field is in the header of an IP packet to classify and prioritize traffic.

DTM:

(*device type manager*) A DTM is a device driver running on the host PC. It provides a unified structure for accessing device parameters, configuring and operating the devices, and troubleshooting devices. DTMs can range from a simple graphical user interface (GUI) for setting device parameters to a highly sophisticated application capable of performing complex real-time calculations for diagnosis and maintenance purposes. In the context of a DTM, a device can be a communications module or a remote device on the network.

See FDT.

E

EDS:

(*electronic data sheet*) EDS are simple text files that describe the configuration capabilities of a device. EDS files are generated and maintained by the manufacturer of the device.

EFB:

(*elementary function block*) This is a block used in a program which performs a predefined logical function.

EFBs have states and internal parameters. Even if the inputs are identical, the output values may differ. For example, a counter has an output indicating that the preselection value has been reached. This output is set to 1 when the current value is equal to the preselection value.

EF:

(*elementary function*) This is a block used in a program which performs a predefined logical function.

A function does not have any information on the internal state. Several calls to the same function using the same input parameters will return the same output values. You will find information on the graphic form of the function call in the [*functional block (instance)*]. Unlike a call to a function block, function calls include only an output which is not named and whose name is identical to that of the function. In FBD, each call is indicated by a unique [number] via the graphic block. This number is managed automatically and cannot be modified.

Position and configure these functions in your program to execute your application.

You can also develop other functions using the SDKC development kit.

EIO network:

(Ethernet I/O) An Ethernet-based network that contains three types of devices:

- local rack
- X80 remote drop (using a BM•CRA312•0 adapter module), or a BMENOS0300 network option switch module
- ConneXium extended dual-ring switch (DRS)

NOTE: Distributed equipment may also participate in an Ethernet I/O network via connection to DRSs or the service port of X80 remote modules.

EtherNet/IP™:

A network communication protocol for industrial automation applications that combines the standard internet transmission protocols of TCP/IP and UDP with the application layer common industrial protocol (CIP) to support both high speed data exchange and industrial control. EtherNet/IP employs electronic data sheets (EDS) to classify each network device and its functionality.

Ethernet:

A 10 Mb/s, 100 Mb/s, or 1 Gb/s, CSMA/CD, frame-based LAN that can run over copper twisted pair or fiber optic cable, or wireless. The IEEE standard 802.3 defines the rules for configuring a wired Ethernet network; the IEEE standard 802.11 defines the rules for configuring a wireless Ethernet network. Common forms include 10BASE-T, 100BASE-TX, and 1000BASE-T, which can utilize category 5e copper twisted pair cables and RJ45 modular connectors.

explicit messaging:

TCP/IP-based messaging for Modbus TCP and EtherNet/IP. It is used for point-to-point, client/server messages that include both data, typically unscheduled information between a client and a server, and routing information. In EtherNet/IP, explicit messaging is considered class 3 type messaging, and can be connection-based or connectionless.

F

FDR:

(*fast device replacement*) A service that uses configuration software to replace an inoperable product.

FDT:

(*field device tool*) The technology that harmonizes communication between field devices and the system host.

FTP:

(*file transfer protocol*) A protocol that copies a file from one host to another over a TCP/IP-based network, such as the internet. FTP uses a client-server architecture as well as separate control and data connections between the client and server.

G

gateway:

A gateway device interconnects two different networks, sometimes through different network protocols. When it connects networks based on different protocols, a gateway converts a datagram from one protocol stack into the other. When used to connect two IP-based networks, a gateway (also called a router) has two separate IP addresses, one on each network.

H

HMI:

(*human machine interface*) System that allows interaction between a human and a machine.

Hot Standby:

A Hot Standby system uses a primary PAC (PLC) and a standby PAC. The two PAC racks have identical hardware and software configurations. The standby PAC monitors the current system status of the primary PAC. If the primary PAC becomes inoperable, high-availability control is maintained when the standby PAC takes control of the system.

HTTP:

(*hypertext transfer protocol*) A networking protocol for distributed and collaborative information systems. HTTP is the basis of data communication for the web.

I

implicit messaging:

UDP/IP-based class 1 connected messaging for EtherNet/IP. Implicit messaging maintains an open connection for the scheduled transfer of control data between a producer and consumer. Because an open connection is maintained, each message contains primarily data, without the overhead of object information, plus a connection identifier.

IP address:

The 32-bit identifier, consisting of both a network address and a host address assigned to a device connected to a TCP/IP network.

L

local rack:

An M580 rack containing the CPU and a power supply. A local rack consists of one or two racks: the main rack and the extended rack, which belongs to the same family as the main rack. The extended rack is optional.

local slave:

The functionality offered by Schneider Electric EtherNet/IP communication modules that allows a scanner to take the role of an adapter. The local slave enables the module to publish data via implicit messaging connections. Local slave is typically used in peer-to-peer exchanges between PACs.

M

MAST:

A master (MAST) task is a deterministic processor task that is run through its programming software. The MAST task schedules the RIO module logic to be solved in every I/O scan. The MAST task has two 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.

MB/TCP:

(*Modbus over TCP protocol*) This is a Modbus variant used for communications over TCP/IP networks.

Modbus:

Modbus is an application layer messaging protocol. Modbus provides client and server communications between devices connected on different types of buses or networks. Modbus offers many services specified by function codes.

%MW:

According to the CEI standard, `%MW` indicates a language object of type memory word.

N

NIM:

(*network interface module*) A NIM resides in the first position on an STB island (leftmost on the physical setup). The NIM provides the interface between the I/O modules and the fieldbus master. It is the only module on the island that is fieldbus-dependent — a different NIM is available for each fieldbus.

NTP:

(*network time protocol*) Protocol for synchronizing computer system clocks. The protocol uses a jitter buffer to resist the effects of variable latency.

P

PAC:

programmable automation controller. The PAC is the brain of an industrial manufacturing process. It automates a process as opposed to relay control systems. PACs are computers suited to survive the harsh conditions of an industrial environment.

port 502:

Port 502 of the TCP/IP stack is the well-known port that is reserved for Modbus TCP communications.

R

RIO drop:

One of the three types of RIO modules in an Ethernet RIO network. An RIO drop is an M580 rack of I/O modules that are connected to an Ethernet RIO network and managed by an Ethernet RIO adapter module. A drop can be a single rack or a main rack with an extended rack.

RIO network:

An Ethernet-based network that contains 3 types of RIO devices: a local rack, an RIO drop, and a ConneXium extended dual-ring switch (DRS). Distributed equipment may also participate in an RIO network via connection to DRSs or BMENOS0300 network option switch modules.

RPI:

(requested packet interval) The time period between cyclic data transmissions requested by the scanner. EtherNet/IP devices publish data at the rate specified by the RPI assigned to them by the scanner, and they receive message requests from the scanner at each RPI.

RSTP:

(*rapid spanning tree protocol*) Allows a network design to include spare (redundant) links to provide automatic backup paths if an active link stops working, without the need for loops or manual enabling/disabling of backup links.

S

SFP:

(*small form-factor pluggable*). The SFP transceiver acts as an interface between a module and fiber optic cables.

SNMP:

(*simple network management protocol*) Protocol used in network management systems to monitor network-attached devices. The protocol is part of the internet protocol suite (IP) as defined by the internet engineering task force (IETF), which consists of network management guidelines, including an application layer protocol, a database schema, and a set of data objects.

SNTP:

(*simple network time protocol*) See NTP.

sub-ring:

An Ethernet-based network with a loop attached to the main ring, via a dual-ring switch (DRS) or BMENOS0300 network option switch module on the main ring. This network contains RIO or distributed equipment.

T

TCP:

(*transmission control protocol*) A key protocol of the internet protocol suite that supports connection-oriented communications, by establishing the connection necessary to transmit an ordered sequence of data over the same communication path.

TFTP:

(*trivial file transfer protocol*) A simplified version of *file transfer protocol* (FTP), TFTP uses a client-server architecture to make connections between two devices. From a TFTP client, individual files can be uploaded to or downloaded from the server, using the user datagram protocol (UDP) for transporting data.

trap:

A trap is an event directed by an SNMP agent that indicates one of these events:

- A change has occurred in the status of an agent.
- An unauthorized SNMP manager device has attempted to get data from (or change data on) an SNMP agent.

U

UDP:

(*user datagram protocol*) A transport layer protocol that supports connectionless communications. Applications running on networked nodes can use UDP to send datagrams to one another. Unlike TCP, UDP does not include preliminary communication to establish data paths or provide data ordering and checking. However, by avoiding the overhead required to provide these features, UDP is faster than TCP. UDP may be the preferred protocol for time-sensitive applications, where dropped datagrams are preferable to delayed datagrams. UDP is the primary transport for implicit messaging in EtherNet/IP.

UMAS:

(*Unified Messaging Application Services*) UMAS is a proprietary system protocol that manages communications between Control Expert and a controller.

UTC:

(*coordinated universal time*) Primary time standard used to regulate clocks and time worldwide (close to former GMT time standard).

Index

A

B

C

D

E

L

M

N

O

P

T

U

W

Schneider Electric
35 rue Joseph Monier
92500 Rueil Malmaison
France

+ 33 (0) 1 41 29 70 00

www.se.com

As standards, specifications, and design change from time to time, please ask for confirmation of the information given in this publication.

EIO0000001578.15