

AC 800M

Communication Protocols

System Version 5.1 Feature Pack

Power and productivity
for a better world™



AC 800M

Communication Protocols

System Version 5.1 Feature Pack

NOTICE

This document contains information about one or more ABB products and may include a description of or a reference to one or more standards that may be generally relevant to the ABB products. The presence of any such description of a standard or reference to a standard is not a representation that all of the ABB products referenced in this document support all of the features of the described or referenced standard. In order to determine the specific features supported by a particular ABB product, the reader should consult the product specifications for the particular ABB product.

ABB may have one or more patents or pending patent applications protecting the intellectual property in the ABB products described in this document.

The information in this document is subject to change without notice and should not be construed as a commitment by ABB. ABB assumes no responsibility for any errors that may appear in this document.

In no event shall ABB be liable for direct, indirect, special, incidental or consequential damages of any nature or kind arising from the use of this document, nor shall ABB be liable for incidental or consequential damages arising from use of any software or hardware described in this document.

This document and parts thereof must not be reproduced or copied without written permission from ABB, and the contents thereof must not be imparted to a third party nor used for any unauthorized purpose.

The software or hardware described in this document is furnished under a license and may be used, copied, or disclosed only in accordance with the terms of such license. This product meets the requirements specified in EMC Directive 2004/108/EC and in Low Voltage Directive 2006/95/EC.

TRADEMARKS

All rights to copyrights, registered trademarks, and trademarks reside with their respective owners.

Copyright © 2003-2013 by ABB.
All rights reserved.

Release: February 2013
Document number: 3BSE035982-511

TABLE OF CONTENTS

About This User Manual

General	15
User Manual Conventions	16
Warning, Caution, Information, and Tip Icons	16
Terminology.....	17
Released User Manuals and Release Notes	18

Section 1 - Introduction

Product Overview	19
Product Scope.....	19
Network Communication	19
ABB I/O Systems	21
Protocols and Controllers Supported by Control Builder	22
Properties of Different Protocols.....	23
Peer-to-Peer Communication Between AC 800M Controllers	25
Methods of Access to Other Controller Systems	28
Clock Synchronization.....	29

Section 2 - MMS

Introduction	33
Services Provided.....	34
MMS Server	34
Design.....	35
Configuration Parameters.....	35
Network Areas.....	36
Explicit and Implicit Addressing	38

MMS on RS-232C (PPP)	38
Separation of Plant Intranet, Client/Server and Control Network	42
Types in MMSCommLib	43
Hardware	43
Redundancy	43
CPU Redundancy	44
Performance	45
Limitations	47
Advanced	48
Default Gateway	48
Default Process Number	50
Troubleshooting	51

Section 3 - IAC

Introduction	53
Characteristics of Communication Variables	53
Services Provided	55
Design	55
Safety Measures	59
Redundancy	60
Online Upgrade	60
Performance	60
Limitations	62
Troubleshooting	64

Section 4 - MasterBus 300

Introduction	67
Services Provided	67
Design	68
Introduction	68
Design Example	68
Communication Function Blocks	69
Redundancy	71

Limitations.....	71
Performance.....	71
Hardware	71
Advanced	72
Troubleshooting.....	72

Section 5 - COMLI

Introduction	73
Services Provided.....	73
Design.....	74
Introduction.....	74
Design Examples.....	74
Redundancy	76
Limitations.....	76
Performance.....	77
Hardware	78
Advanced	78
Procedure for Linking Control Systems with COMLI.....	78

Section 6 - SattBus on TCP/IP

Introduction	79
Services Provided.....	79
Design.....	80
Introduction.....	80
Redundancy	80
Limitations.....	80
Performance.....	81
Advanced	81

Section 7 - INSUM

Introduction	83
Services Provided.....	84
Design.....	84

Introduction.....	84
Design Example.....	85
Redundancy.....	87
Limitations.....	87
Performance.....	87
Hardware.....	88
Troubleshooting.....	88

Section 8 - Siemens 3964R

Introduction.....	89
Services Provided.....	89
Design.....	90
Introduction.....	90
Limitations.....	91
Performance.....	91
Hardware.....	91
Advanced.....	92
Communicating Integers.....	92
Communicating Bits.....	93

Section 9 - MODBUS RTU

Introduction.....	95
Services Provided.....	95
Design.....	96
Introduction.....	96
Design Examples.....	96
Hardware.....	98
Performance.....	98
Limitations.....	98
Redundancy.....	98
Troubleshooting.....	98

Section 10 - MODBUS TCP

Introduction	99
Services Provided	99
Design	100
Introduction	100
Design Examples	101
Connection Methods	103
Connection of MODBUS RTU MODBUS Devices	103
Redundancy	104
Online Upgrade	105
Limitations	106
Performance	107
Hardware	109
Troubleshooting	109
 Section 11 - AF 100	
Introduction	111
Services Provided	112
Design	112
DataSet Peripheral Communication	115
Bus Master Function	118
Network Configuration	118
Online Upgrade	119
Hardware	119
Troubleshooting	120
 Section 12 - MOD5-to-MOD5	
Introduction	121
Services Provided	121
Design	122
Connection Examples	123
Configuration Considerations	124
Module Redundancy	126
Online Upgrade	128

Troubleshooting.....	129
Section 13 - EtherNet/IP and DeviceNet	
Introduction	131
Services Provided.....	131
Design	132
Design Example.....	133
Redundancy	134
Online Upgrade	135
Limitations	136
EtherNet/IP Limitations	137
LD 800DN Limitations	137
Performance	137
Section 14 - IEC 61850	
Introduction	139
Services Provided.....	139
Design	140
Introduction.....	140
Design Examples	141
IEC 61850 Hardware Objects	141
Connection Methods	143
Redundancy	143
Module Redundancy by IEC 61131-3 Application Logic	143
Online Upgrade	144
CI868 Performance	145
Capacity of IEC 61850 solution using CI868.....	145
Diagnostic Information for LN0 and MMS Diag Hardware Object on CI868..	147
Troubleshooting.....	148
Section 15 - FOUNDATION Fieldbus HSE	
Introduction	149
Advantages.....	149

Design	150
Introduction	150
Design Example	151
Connection Methods	152
Redundancy	153
Limitations and Performance	153
General	153
Dimensioning Limits, Linking Device	153
Dimensioning Limits, FOUNDATION Fieldbus HSE Communication Interface Module CI860	154
Hardware	155
Advanced	155
Troubleshooting	156

Section 16 - DriveBus

Introduction	157
Services Provided	157
Advantages	157
Design	158
Design Example	158
Dataset Communication	159
Configuration	160
Redundancy	160
Limitations	161
Performance	161
Hardware	162
Advanced	162

Section 17 - PROFIBUS DP

Introduction	163
Services Provided	164
Advantages	164
Design	164

Introduction.....	164
Design Example.....	165
Redundancy.....	166
Limitations.....	166
Performance.....	166
Hardware.....	167
Advanced.....	167
Troubleshooting.....	167
CI854 Web Interface.....	168

Section 18 - PROFINET IO

Introduction.....	169
Services Provided.....	170
Advantages.....	170
Design.....	171
Introduction.....	171
Design Examples.....	172
Redundancy.....	172
Online Upgrade.....	173
Technical Data.....	173
Hardware.....	174
Troubleshooting.....	174
CI871 Web Interface.....	174

Section 19 - Self-defined UDP Communication

Introduction.....	175
Design.....	176
Hardware.....	178
Performance.....	178
Limitations.....	178
Redundancy.....	179
Online Upgrade.....	179
Troubleshooting.....	179

Section 20 - Self-defined TCP Communication

Introduction 181
 Design..... 181
 Hardware 182
 Performance..... 182
 Limitations..... 182
 Redundancy 183
 Online Upgrade 183
 Troubleshooting..... 183

Section 21 - Self-defined Serial Communication

Introduction 185
 Design..... 185
 Hardware 186
 Performance..... 186
 Limitations..... 186
 Redundancy 187
 Advanced 187
 Troubleshooting..... 188

Section 22 - Modem Communication

Introduction 189
 Short Distance Modem..... 189
 Dial-Up Modem..... 190
 Limitations..... 192
 Performance..... 192
 Troubleshooting..... 192

Appendix A - OSI Profile for MMS

MMS Services 193
 Reduced OSI Implementation 195

Appendix B - Used Port Numbers

Used Ports 197

Appendix C - Configuration of HART Devices

Introduction 199
Configuration Example 200
Toolrouting 201

Appendix D - PROFIBUS PA

PROFIBUS PA 203

Appendix E - ABB Drives

Introduction 205
Types of ABB Drives 206
 ABB Standard Drives 206
 ABB Engineered Drives 206
Parameter Group Configuration 207

INDEX

About This User Manual

General



Any security measures described in this User Manual, for example, for user access, password security, network security, firewalls, virus protection, etc., represent possible steps that a user of an 800xA System may want to consider based on a risk assessment for a particular application and installation. This risk assessment, as well as the proper implementation, configuration, installation, operation, administration, and maintenance of all relevant security related equipment, software, and procedures, are the responsibility of the user of the 800xA System.

This user manual describes the criteria for selecting networks and communication protocols and is intended for engineers who are planning the design of a new network or the expansion of an existing one. Note, however, that pictures of devices are for illustrative purposes only. Refer to the relevant hardware user's guides for information on how to connect cables.

The described functions may contain some restrictions that are specific to the release. Please refer to the latest Product Guides and Release Notes regarding possible restrictions.

It is recommended to use *Control Network*. This uses the MMS protocol and Inter Application Communication (IAC) with Ethernet and/or RS-232C as physical media.

However, with regard to existing equipment or other circumstances other protocols and fieldbuses are also used. MB 300, COMLI, Siemens 3964R, MODBUS RTU, MODBUS TCP, MOD5-to-MOD5, and SattBus are standard protocols for general data communication between controllers. FOUNDATION Fieldbus, PROFIBUS DP, PROFINET IO, DriveBus, INSUM, AF 100, EtherNet/IP and DeviceNet, and IEC 61850 are dedicated to I/O communication.

Self-defined TCP communication and self-defined UDP communication are also used if the controller need to communicate with external equipment. These use TCP and UDP protocols running on Ethernet.

This user manual provides the following specific information about the used protocols:

- Design
- Redundancy
- Online Upgrade
- Limitations
- Performance
- Hardware
- Troubleshooting

For detailed information on the use of the programming tool *Control Builder Professional*, refer to the online help and the *System 800xA Control, AC 800M, Configuration (3BSE035980*)* manual.

[Section 1, Introduction](#), presents an overview of communication protocols supported by Control Builder, and the main criteria for selection.

The remaining sections describe the MMS protocol, IAC, and other communication protocols supported, and well as modem communication. The appendices deal with the OSI profile for MMS, HART devices, PROFIBUS PA, and ABB Drives.

User Manual Conventions

Microsoft Windows conventions are normally used for the standard presentation of material when entering text, key sequences, prompts, messages, menu items, screen elements, etc.

Warning, Caution, Information, and Tip Icons

This User Manual includes Warning, Caution, and Information where appropriate to point out safety related or other important information. It also includes Tip to point

out useful hints to the reader. The corresponding symbols should be interpreted as follows:



Electrical warning icon indicates the presence of a hazard that could result in *electrical shock*.



Warning icon indicates the presence of a hazard that could result in *personal injury*.



Caution icon indicates important information or warning related to the concept discussed in the text. It might indicate the presence of a hazard that could result in *corruption of software or damage to equipment/property*.



Information icon alerts the reader to pertinent facts and conditions.



Tip icon indicates advice on, for example, how to design your project or how to use a certain function

Although Warning hazards are related to personal injury, and Caution hazards are associated with equipment or property damage, it should be understood that operation of damaged equipment could, under certain operational conditions, result in degraded process performance leading to personal injury or death. Therefore, fully comply with all Warning and Caution notices.

Terminology

A complete and comprehensive list of terms is included in *System 800xA System Guide Functional Description (3BSE038018*)*. The listing includes terms and definitions that apply to the 800xA System where the usage is different from commonly accepted industry standard definitions and definitions given in standard dictionaries such as Webster's Dictionary of Computer Terms.

Released User Manuals and Release Notes

A complete list of all User Manuals and Release Notes applicable to System 800xA is provided in *System 800xA Released User Manuals and Release Notes (3BUA000263*)*.

System 800xA Released User Manuals and Release Notes (3BUA000263)* is updated each time a document is updated or a new document is released. It is in pdf format and is provided in the following ways:

- Included on the documentation media provided with the system and published to ABB SolutionsBank when released as part of a major or minor release, Service Pack, Feature Pack, or System Revision.
- Published to ABB SolutionsBank when a User Manual or Release Note is updated in between any of the release cycles listed in the first bullet.



A product bulletin is published each time *System 800xA Released User Manuals and Release Notes (3BUA000263*)* is updated and published to ABB SolutionsBank.

Section 1 Introduction

Product Overview

Product Scope

The products described are utilized for general network communication of real-time data between controllers and computers in an industrial environment.

Network Communication

It is recommended to use *Control Network* which is a private IP network domain especially designed for industrial applications. This means that all communication handling will be the same, regardless of network type or connected devices. Control Network is scalable from a very small network with a few nodes to a large network containing a number of *network areas* with hundreds of addressable nodes (there may be other restrictions such as controller performance).

Control Network uses the *MMS* communication protocol on Ethernet and/or RS-232C to link workstations to controllers. MMS (Manufacturing Message Specification) is an ISO 9506 standard. In order to support Control Network on RS-232C links, the *Point-to-Point Protocol* (PPP) is used. The *Redundant Network Routing Protocol* (RNRP) developed by ABB handles alternative paths between nodes and automatically adapts to topology changes. MMS is described in [Section 2, MMS](#).

In addition, other protocols such as MB 300, COMLI, Siemens 3964R, MODBUS RTU, MODBUS TCP, SattBus, and MOD5-to-MOD5 can be used. Fieldbuses such as FOUNDATION Fieldbus HSE, PROFIBUS DP (according to IEC 1158-2 and EN 50170), PROFINET IO, DriveBus, INSUM, IEC 61850, Advant Fieldbus 100 (AF 100), and EtherNet/IP and DeviceNet can be connected to the network via communication interface units. UDP/TCP Communication Libraries are used to communicate with external devices through Ethernet with user-defined protocols.

Table 1– Table 4 give concise information to be used when selecting protocols.

The Control Network, as well as other protocols and fieldbuses, is configured by means of the project explorer in Control Builder (see Figure 1). The Control Network is specified by settings in the parameter lists, accessed by right-clicking the symbols for the CPUs and the Ethernet and/or PPP symbols (see Section 2, MMS for further information). Hardware configuration is explained in the Control Builder online help. PC nodes are specified in the PC setup wizard.

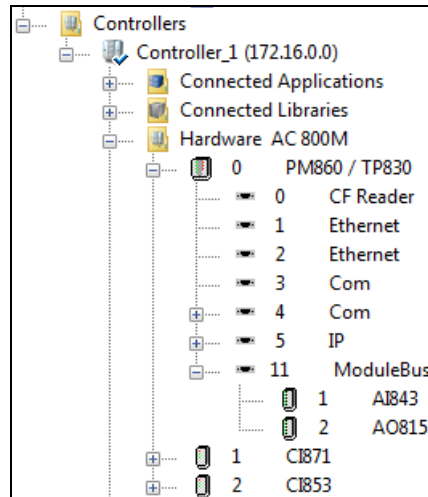


Figure 1. Project Explorer

ABB I/O Systems

- S100 I/O is connected to AC 800M via CI856.
- S200 I/O is connected to AC 800M via CI854 and CI865. Please refer to the manual *S200 I/O Hardware, Hardware and Installation, User's Guide (3BSE021356*)* for more information.
- S800 I/O is connected to AC 800M via modulebus and CI854. Please refer to the manual *S800 I/O - General Information and Installation - User's Guide (3BSE020923*)* for more information.
- S900 I/O via CI854.
- TRIO is connected to AC 800M via CI862. Please refer to the manual 800xA for *TRIO/Genius - Introduction and Installation (3BUR002459*)* for more information.
- Satt 19" Rack I/O is connected to AC 800M via CI865. Please refer to the manual *Satt I/O Interface for AC 800M (3BSE042821*)* for more information.

Protocols and Controllers Supported by Control Builder

Table 1 lists controllers and protocols supported by the current version of Control Builder.

Table 1. Protocols and Controllers supported by Control Builder.

Protocol	AC 800M	AC 800M HI
IAC	YES	YES
MMS on Ethernet	YES	YES
MMS on RS-232C (PPP)	YES	YES
MasterBus 300	YES	YES
SattBus on TCP/IP	YES	YES
COMLI ⁽¹⁾	YES	YES
Siemens 3964R ⁽²⁾	YES	YES
MODBUS RTU ⁽³⁾	YES	YES
MODBUS TCP ⁽⁴⁾	YES	YES
FOUNDATION Fieldbus HSE	YES	NO
PROFIBUS DP	YES	YES
PROFINET IO	YES	NO
DriveBus	YES	NO
INSUM	YES	YES
IEC 61850	YES	YES
MOD5-to-MOD5	YES	YES
AF 100	YES	YES
EtherNet/IP and DeviceNet	YES	YES
UDP	YES	NO
TCP	YES	NO

- (1) Both master and slave
- (2) Master only
- (3) Master only
- (4) Both master and slave

Properties of Different Protocols

Table 2 shows access modes used, variable types handled and maximum message size permitted for various protocols, as well as which protocols that require interface units with separate CPUs, and protocols that support dial-up modems.

Table 2. Properties of the different protocols.

Protocol	Access method	Separate CPU for communication	Dial-up modem	Variable types ⁽¹⁾						Max. number of bits/registers or bytes per message
				Boolean ⁽²⁾	Integer ⁽²⁾	Real	String	Word ⁽²⁾	Struct ⁽³⁾	
IAC MMS	Ethernet			x	x	x	x	x	x	(4)
MMS on Ethernet	Ethernet			x	x	x	x	x	x	(3)
MMS on RS-232C (PPP)	Point-to-point			x	x	x	x	x	x	(3)
MasterBus 300	Ethernet	x		x	x	x				
SattBus on TCP/IP	Ethernet			x	x	x	x		x	31 bytes
COMLI	Multidrop		x	x	x					512/32
Siemens 3964R	Point-to-point			x	x					512/32
MODBUS RTU	Multidrop			x	x					1968/123
MODBUS TCP	Ethernet	x		x	x	x				1968/123

Protocol	Access method	Separate CPU for communication	Dial-up modem	Variable types ⁽¹⁾						Max. number of bits/registers or bytes per message
				Boolean	Integer ⁽²⁾	Real	String	Word ⁽²⁾	Struct ⁽³⁾	
Self-defined in Serial Communication Library	Point-to-point			x	x	x	x	x	x	140 bytes
IEC 61850	Ethernet	x		x	x	x				
MOD5-to-MOD5	Fiber Optic	x		x	x					417 bytes
AF 100	Twisted Pair	x		x	x	x		x		32 bytes
PROFIBUS DP	RS-485	x		x	x	x		x		244 bytes
PROFINET IO	Ethernet	x		x	x	x		x		1440 bytes
EtherNet/IP	Ethernet	x		x	x	x		x		508 bytes
UDP	Ethernet				x			x		1472 bytes
TCP	Ethernet				x			x		1420 bytes

- (1) When transferring variables it is important to use data types having the same range on both client and server. However, a dInt variable on the server can be connected to an Int variable on the client if the values are within the Int variable's range
- (2) 16 bits and 32 bits
- (3) MMS and SattBus can transfer structured variables of the data types given in the table. No protocol can transfer variables of types ArrayObject or QueueObject.
- (4) See [Table 5](#) on page 47.

Peer-to-Peer Communication Between AC 800M Controllers

Table 3 provides details on the protocols supported by AC 800M which can be used for peer-to-peer communication between controllers.:

Table 3. Summary of characteristics

Protocol	Determinism	Redundancy	Throughput (4-byte values per second)
IAC MMS on the Control Network	Delay at Transmission and reception depends on CPU load	Network (RNRP)	Depends on Interval Time. See Interval Time and Timeout on page 57.
MMS on the Control Network	Delay at Transmission and reception depends on CPU load	Network (RNRP)	Server: 4000..13200 Client: 5700..19800
MB 300	Delay at Transmission and reception depends on CPU load. Supervision of transmission rate	Network	2400 (per CI855)
FF HSE	Transmission and reception independent of CPU load	CI Module	660 (per CI860)

Table 3. Summary of characteristics

Protocol	Determinism	Redundancy	Throughput (4-byte values per second)
MODBUS TCP	Transmission and reception independent of CPU load	CI Module	3750 (per CI867) Maximum transactions per second per CI867 is 60 with request sent at 50ms. CI867 connected to one AC 800M is considered as master and CI867 connected to the second controller is configured as slave and then performance throughput is calculated.
MOD5-to-MOD5	Transmission and reception independent of CPU load.	CI Module + Network	One remote MOD5 controller transmits 100 variables and receives 100 variables every second.
PROFIBUS DP	Transmission and reception independent of CPU load	CI Module + Network	75000 with 12Mbits/sec
PROFINET IO	Transmission and reception independent of CPU load	CI Module of Application Logic	Maximum of 360.000 signals in. Maximum of 360.000 signals out.

Table 3. Summary of characteristics

Protocol	Determinism	Redundancy	Throughput (4-byte values per second)
IEC 61850	Transmission and reception independent of CPU load	CI Module of Application Logic	Receive: 22500, with 150 analog data attributes within each of the 150 datasets (maximum possible). Send: 9000, with 150 analog data attributes within each of the 60 datasets (maximum possible).
AF 100	Transmission and reception independent of CPU load	CI Module + Network	24000, if 32 byte CDPs are used. 8000, if 4 byte CDPs are used. A mix of 4, 8, 16 and 32 byte CDPs gives a throughput between 8000 and 24000 4-byte data per second.
UDP	Delay at Transmission and reception depends on CPU load	Network (RNRP)	
TCP	Delay at Transmission and reception depends on CPU load	Network (RNRP)	

Methods of Access to Other Controller Systems

The [Table 4](#) indicates protocols that can be used for communication between AC 800M and other legacy controllers by ABB.

Table 4. Methods of access to legacy controllers.

Protocol	MB 300	SattBus on TCP/IP	COMLI ⁽¹⁾	Siemens 3964R	MOD -BUS	Self-def. in Serial Comm. Library	MMS	AF 100
SattLine 200		×	×			×	x ⁽²⁾	
AC 210			×			×		
AC 250 with ACB ver. 1		×	×			×		
AC 250 with CB 2 or later		×	×			×	×	
AC 800C		×	×			×	×	
MP 200/1	×			x ⁽³⁾	x ⁽⁴⁾			
AC 55						×		
AC 70					×	×		×
AC 110, 160				×	×	×		×
AC 410, 450	×			×	×	×		×
SattCon05			x ⁽⁵⁾			×		
SattCon15, 31, 35, 60, 115, 125			×			×		
SattCon200		×	×			×		

(1) Supported message types differ between the controllers; refer to the relevant programmer's manuals.

(2) Support in SattLine CPU50 v2.3 or later, and SattLine Workstation v2.3 or later.

(3) From version MP 200/1, version 4.0.

(4) From version MP 200/1, version 2.1.

(5) With control board CU05-25, CU05-45 or CU05-65.

Clock Synchronization

AC 800M supports clock synchronization by four different protocols: *CNCP*, *SNTP*, *MB 300 Clock Sync*, and *MMS Time Service*. In addition to these, AC 800M also supports the protocol type *SNTP on CI* for clock synchronization using communication interfaces (CI) that independently function as time master or time slave.

The protocol to be used for receiving time is chosen in the Hardware Editor of the Control Builder.

AC 800M can send clock synchronization with all protocols simultaneously, but it uses one configured protocol (by the parameter *CS protocol type*) to receive clock synchronization from another source. Advantage of AC 800M is that it can receive time with a protocol and distributes to other nodes with another protocol, and acts as a router.

CNCP is the normal protocol for clock synchronization on the Control Network. An AC 800M controller selected as Clock Master (i.e. with *Clock Master Order No*>0) multicasts synchronization messages on the network (see [Figure 2](#)). All nodes that are configured to receive time with CNCP (`CS protocol type = CNCP`) is synchronized from the Clock Master.

SNTP is a standardized protocol used by AC 800M controllers that need to be synchronized from an external time server which is connected to the Control Network. Set `CS protocol type = SNTP` to configure AC 800M to be an SNTP client. As SNTP is a simplified version of NTP, both NTP and SNTP servers can be used.

To get a good clock accuracy in the AC 800M, an (S)NTP server with high precision should be used. The SNTP server is typically synchronized via a GPS receiver. An SNTP server can typically handle many SNTP clients that receive clock synchronization through GPS. Therefore, all controllers on a control network can use SNTP to be synchronized from the same SNTP server. AC 800M contains an SNTP server that always is enabled. It can be used by other units that need to be synchronized if no external NTP server is used. CNCP and SNTP can both operate at the same time on the network.

SNTP on CI is a protocol that is used by AC 800M controllers, which have communication interfaces that can handle clock synchronization independently (for example, the CI869 that communicates with AF 100).

Set CS protocol type = SNTP on CI to configure the CI (which is connected to AC 800M) to be an SNTP client. These communication interfaces have separate clock synchronization setting (Master/Slave).

The OPC server for the AC 800M supports the MMS Time Service for small systems where no AC 800M is used for backward compatibility with older products.

MB 300 Clock Sync is a protocol for clock synchronization of Advant/Master products on a MasterBus 300 network. AC 800M can receive its synchronization via CI855 (CS protocol type = MB300).

CI855 can act as Clock Sync master on MB 300 (CI855 Parameter Time Sync = MB300 Master)

For more details on Clock Synchronization refer to 3BSE03446*, *Automation System Network: Design and Configuration*.

Clock Synchronization

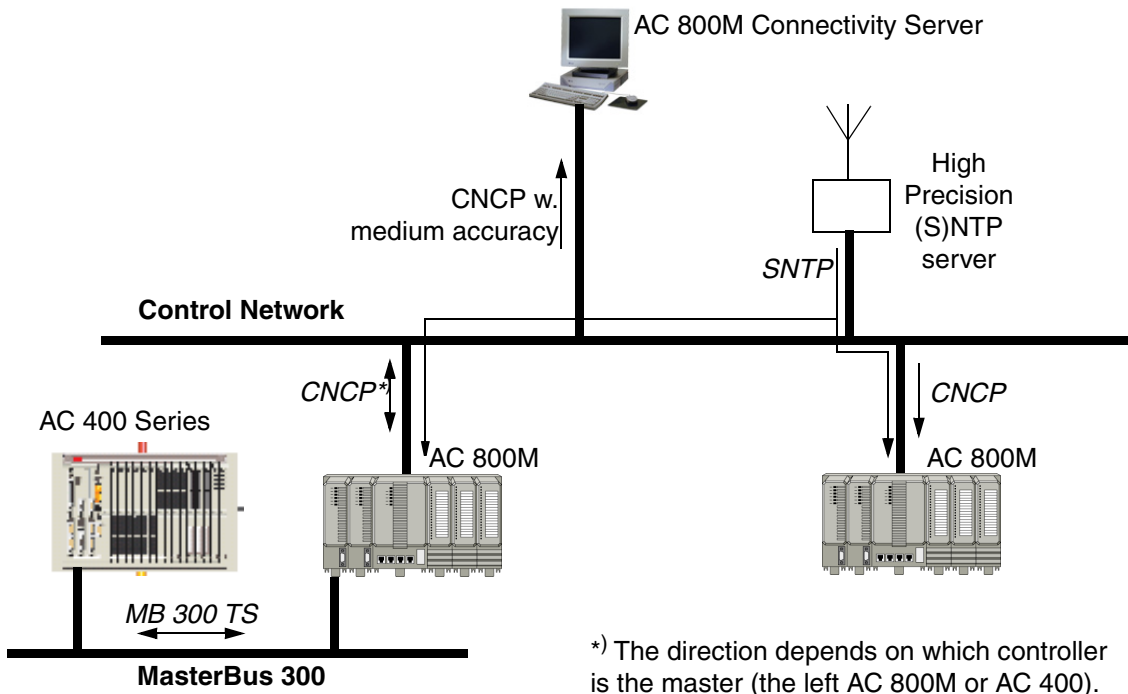


Figure 2. Clock Synchronization

Intermediate Clock Master

AC 800M can act as intermediate clock master. This means that it relays time synchronization between two Network Areas with CNCP. To do this, it shall have a Clock Master order number that is at least two numbers higher than any ordinary Clock Master on the network area with the time source.

The standard and recommended synchronization interval is 20 seconds.

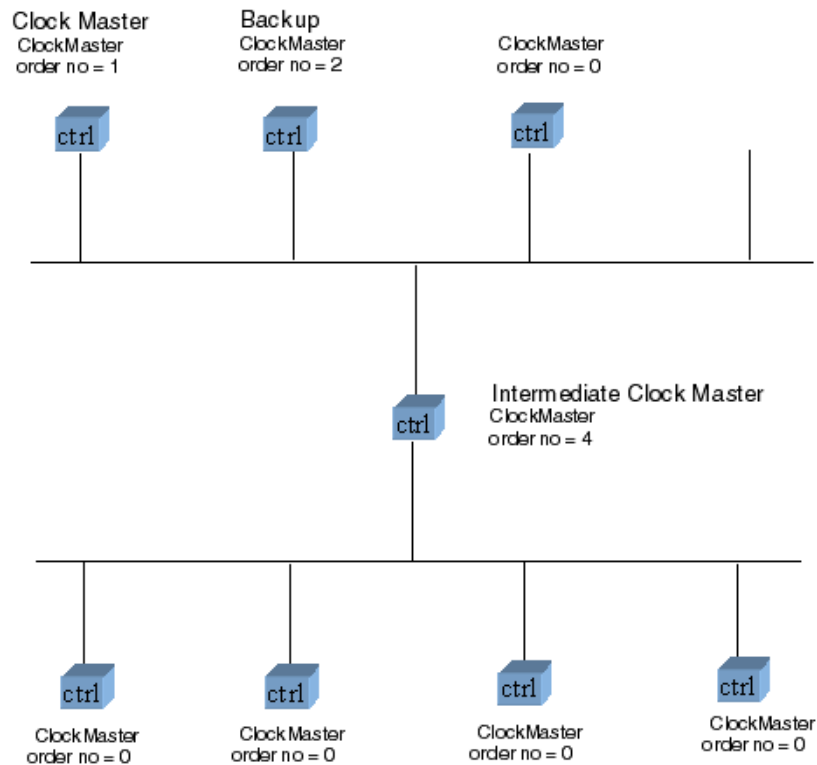


Figure 3. Intermediate Clock Master - Configuration

Section 2 MMS

Introduction

Control Network uses the MMS protocol and a reduced Open Systems Interconnection (OSI) stack with the TCP/IP protocol in the transport/network layer, and Ethernet and/or RS-232C as physical media. MMS (Manufacturing Message Specification) is an ISO 9506 standard developed for industrial applications. The protocol defines communication messages transferred between controllers as well as between the engineering station (such as Control Builder) and the controller (e.g. downloading an application or reading/writing variables). It has been developed especially for industrial applications. See also [Appendix A, OSI Profile for MMS](#).

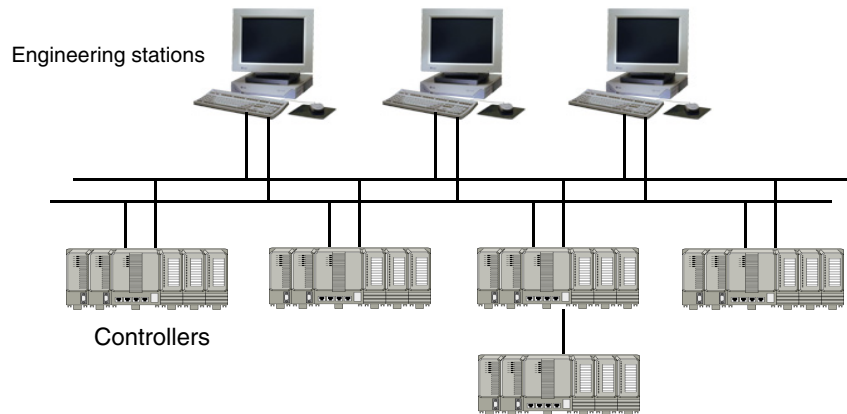


Figure 4. The MMS protocol defines communication messages transferred between controllers as well as between engineering stations and controllers.

Services Provided

The MMS protocol provides several services¹ within a network:

- Downloading an application, e.g. executable code and data from an engineering station (such as Control Builder) to a controller.
- Creating, deleting, starting, and stopping programs over the network.
- Reading and writing variables located in other systems on the network.
- Obtaining information about applications being executed and about error conditions in remote systems.
- Reading and writing files over the network.
- Handling alarm conditions.
- Obtaining information on remote system capability, model identification and revision of remote systems.
- Support of Safe peer-to-peer. For more information please refer to Communication Handling in Control Builder online help.

Main advantages:

- The MMS protocol is an ISO 9506 standard protocol, which means all communication handling will be the same, regardless of network type and connected devices.
- The protocol can be used on many different networks, but preferably on the TCP/IP network, which is the most commonly used network today. ABB only uses MMS on the TCP/IP network.

MMS Server

The function of the MMS Server resembles a multiplexer between Control Builder, OPC Server and controllers, see *AC 800M, OPC Server (3BSE035983*)* manual. The MMS Server is automatically installed with Control Builder or OPC Server.

1. See also [Appendix A, OSI Profile for MMS](#).

Design

Configuration Parameters

The Control Network is configured through the Project Explorer in Control Builder. The different alternatives are described in the hardware manual for the respective controller. Settings for the controller and the communication channel (Ethernet or PPP) are entered via the Control Builder. PC nodes are configured in the PC setup wizard (refer to the Setup Wizard online help and the manual *Automation System Network, Design and Configuration, 3BSE034463**).

To display the *parameter list* from the hardware tree:

1. Expand *Controllers*
2. Find your controller
3. Expand *Hardware AC 800M*
4. Expand the processor unit
5. Click the *Ethernet* channel
6. Select the *Settings* tab.

Parameter	Value	Type	Unit	Min	Max
IP address	172.16.0.0	string			
IP subnet mask	255.255.0.0	string			
Network Area	0	dint		0	35
Path Number	0	dint		0	1
Node Number	0	dint		0	500
Network Area Local	false	bool			
Send Period	1	dint	s	1	60
Max Lost Messages	3	dint		1	10
Proxy router	0.0.0.0	string			
Target address	0.0.0.0	string			

Figure 5. Ethernet parameter list in Control Builder.

The *IP address* and *IP subnet mask* are standard IP terms, whereas the remaining parameters are used by the *Redundant Network Routing Protocol (RNRP)*.



For more information on RNRP setup, see the manual *Automation System Network, Design and Configuration (3BSE034463*)*.

Network Areas

The Control Network normally covers one manufacturing plant. A large Control Network can be divided into *network areas* (subnetworks), for example to keep most of the time-critical communication within smaller areas, thereby improving performance.

Network areas are interconnected by RNRP routers. The node type most commonly used as a router is the AC 800M Connectivity Server, see [Figure 6](#). The AC 800M controller also has router capability, but since it only has two network ports it can only be a router between two non-redundant network areas. It is not possible to do routing between two redundant network areas by using two AC 800M with one on each path. In addition it is not possible to build a fully redundant solution by only using standard routers that do not use RNRP. For more information about this, please refer to the manual *Automation System Network, Design and Configuration (3BSE034463*)*.

AC 800M will act as a router if it detects that its two ports are connected to two different network areas. It automatically starts routing without any need for extra configuration data



A path connection between two nodes must not contain more than three routers (a *hop count* greater than 3 is not permitted).

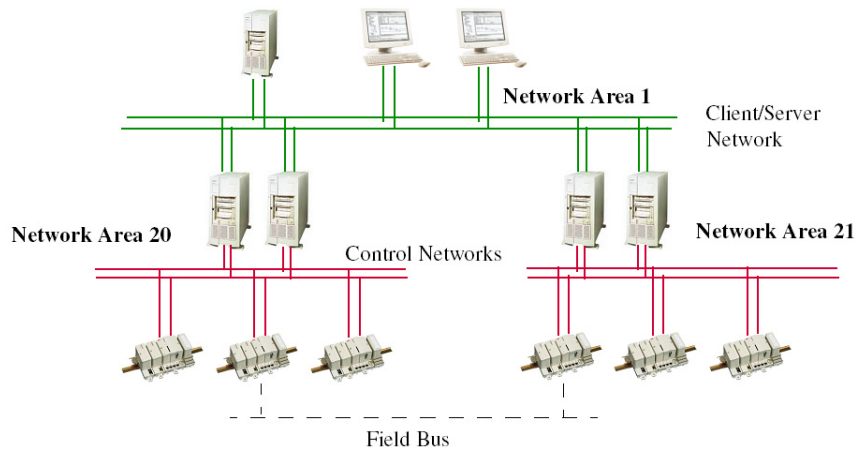


Figure 6. Non-redundant Control Network with three network areas which are connected via AC 800M controllers used as routers.

Example:

The two Ethernet channels of an AC 800M with node number 8 are used to connect network areas 1 and 3.

Port 1	Parameter	Value	Type	Unit	Min	Max
	IP address	172.16.4.8	string			
	IP subnet mask	255.255.252.0	string			
	Network Area	1	dint		0	31
	Path Number	0	dint		0	1
	Node Number	8	dint		0	500
	Network Area Local	false	bool			
	Send Period	10	dint	0.5s	1	12000
	Max Lost Messages	3	dint		1	10
Port 2	Parameter	Value	Type	Unit	Min	Max
	IP address	172.16.12.8	string			
	IP subnet mask	255.255.252.0	string			
	Network Area	3	dint		0	31
	Path Number	0	dint		0	1
	Node Number	8	dint		0	500
	Network Area Local	false	bool			
	Send Period	10	dint	0.5s	1	12000
	Max Lost Messages	3	dint		1	10

Figure 7. AC 800M used as a router between two network areas.

Explicit and Implicit Addressing

In the previous example the *explicit addressing* method is used. That is, in addition to entering the IP address the parameters *network area*, *path number* and *node number* are explicitly entered in the parameter list. While using the RNRP conventions, these parameters will automatically be extracted from the IP address and mapped onto the RNRP parameters if the corresponding entries in the parameter list are left zero. This is called *implicit addressing*. The parameter list for network area 3 above will have the appearance shown in [Figure 8](#).



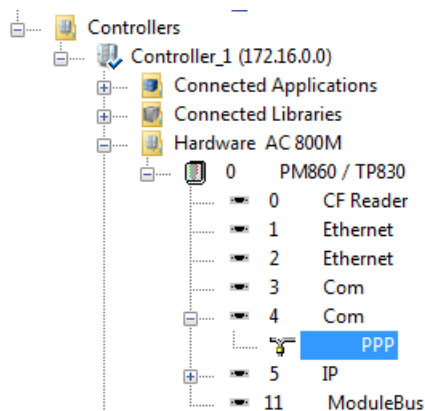
In order to obtain supervision of the Network connection, and the PPP connection done with explicit addressing, RNRP must be configured (enabled at all time).

Parameter	Value	Type	Unit	Min	Max
IP address	172.16.12.8	string			
IP subnet mask	255.255.252.0	string			
Network Area	0	dint		0	31
Path Number	0	dint		0	1
Node Number	0	dint		0	500
Network Area Local	false	bool			
Send Period	10	dint	0.5s	1	12000
Max Lost Messages	3	dint		1	10

Figure 8. Parameter list when the implicit addressing method is used.

MMS on RS-232C (PPP)

RS-232C is a point-to-point communication link for direct interconnection of two controllers. The point-to-point protocol (PPP) is used to support the IP network.



Parameter	Value	Type	Unit	Mi
IP address	192.168.255.25	string		
IP subnet mask	255.255.255.0	string		
Network Area	0	dint		0
Path Number	0	dint		0
Node Number	0	dint		0
Network Area Local	false	bool		
Send Period	5	dint	s	1
Max Lost Messages		dint		1
Proxy router	0.0.0.0	string		
Target address	0.0.0.0	string		
Remote IP address	192.168.255.32	string		

Figure 9. MMS on RS-232C.

Clicking *PPP* displays a parameter list similar to that for Ethernet, but the destination must be known and entered as the *remote IP address*.



Implicit addressing cannot be used in this case. Consequently *network area*, *path number*, and *node number* have to be entered in the parameter list.



PPP running on CI853 does not support Hot Swap and a controller using PPP can not be upgraded using the Online Upgrade function.



How to configure a PPP connection is described in detail in Control Builder online help.

To communicate with most products on the market, you must select an address from the class C private internet address space 192.168.0.0-192.168.255.0 with the subnet mask 255.255.255.0. Allowed node numbers are 1-254 and must be the same as the host ID part (the last part) of the IP address.



PPP communication must not use the same network id as Control Network communication. Using the same network id will result in IP addressing conflicts.



Changes to PPP network interface settings will not take effect until either a controller restart has taken place or the cable has been removed and then connected again.

To make it possible for a controller to use PPP connection and not use the Ethernet ports, the Ethernet ports must be either disabled or set to use IP addresses that is not used somewhere else in the network system.

Example

- Controller_A is connected to the Control Network with the network address 172.16.84.1
- Controller_A is connected to Controller_B via a PPP link.
- Controller_B is not connected to any Ethernet network.

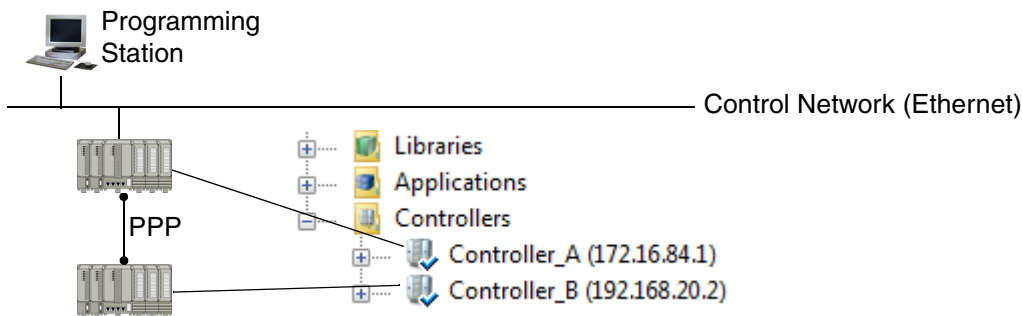


Figure 10. Controller_A connected to Controller_B via PPP link

The following settings must be done to the Ethernet ports of Controller_B to make the PPP communication to work:

1. Ethernet port 1:
The IP address must be set to an IP address that is not used anywhere else in the network system, for example 172.0.0.1

Parameter	Value
IP address	172.0.0.1
IP subnet mask	255.255.252.0
Network Area	1

Figure 11. IP address setting of Controller_B for Ethernet port 1

2. Ethernet port 2:
The parameter *Enable Ethernet channel* must be set to false (disabled) or be set to an IP address that is not used anywhere else in the network system (for example 172.0.0.2).

Parameter	Value
Proxy router	0.0.0.0
Target address	0.0.0.0
Enable Ethernet channel	false

Figure 12. Ethernet port 2 of Controller_B is disabled by setting the Enable Ethernet channel parameter to false

To make it possible to access Controller_B from the Control Network, the *Network Area Local* parameter of the PPP ports must be set to false for both controllers.



Com port 4 is pre-configured to be used as Tool Port. Use Com port 3 or a new added Com port (CI853 unit) with PPP for the connection. See Control Builder online help for additional information.

- *Remote IP address* of PPP for Controller_A must be set to the PPP IP address of Controller_B and vice versa, see [Figure 13](#).

Parameter	Value	Parameter	Value
IP address	192.168.20.1	IP address	192.168.20.2
IP subnet mask	255.255.255.0	IP subnet mask	255.255.255.0
Network Area	2	Network Area	2
Path Number	0	Path Number	0
Node Number	0	Node Number	0
Network Area Local	false	Network Area Local	false
Send Period	5	Send Period	5
Max Lost Messages	3	Max Lost Messages	3
Proxy router	0.0.0.0	Proxy router	0.0.0.0
Target address	0.0.0.0	Target address	0.0.0.0
Remote IP address	192.168.20.2	Remote IP address	192.168.20.1

Figure 13. PPP settings for Controller_A and Controller_B

All routable (non-local) PPP links in the system must be given a unique Network Area number. However, the same node number can be used in different Network Areas, within same network (see [Figure 14](#)).

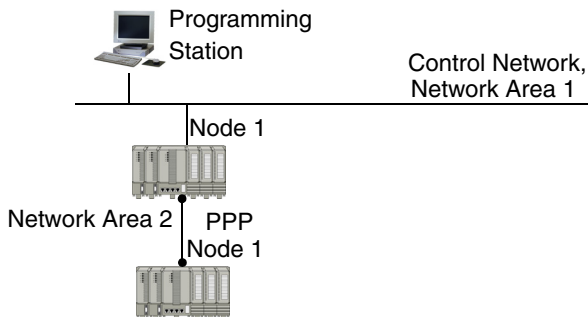


Figure 14. Node number 1 used in different Network Areas

Cables to use for PPP/COMLI connections

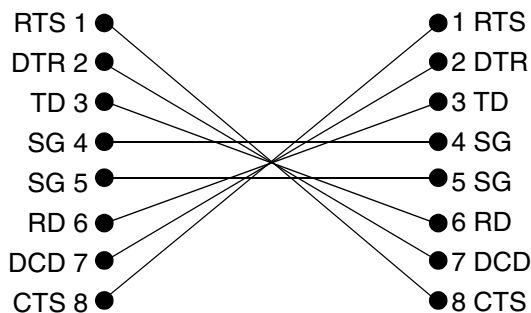


Figure 15. RJ45 to RJ45 serial cable.

Separation of Plant Intranet, Client/Server and Control Network

The Control Network must be protected from public traffic that can be a security risk and also cause undesired load on both the nodes and network. To avoid these risks the Control Network should be physically separated from the Plant Intranet and protected by servers and/or firewalls. In large configurations such separation may also be desirable between the Control Network and client/server networks. Refer to the manual *Automation System Network, Design and Configuration*, (3BSE034463*) for more information.

Types in MMSCommLib

The MMS Communication Library (MMSCommLib) contains MMS function block types and control module types for establishing communication with systems using the MMS protocol.

You can select an object type and drop this object into an editor. Double-click the object type in the Project Explorer to display the contents of the object type in an editor.

For details about the available types and their usage, see the Online Help of Control Builder.

Hardware

All hardware complying with the Ethernet IEEE 803.2 standard can be used for MMS communication. Typical hardware units that can use MMS are Ethernet transceivers, hubs, switches, and routers.

Although the Ethernet standard used in automation technology is the same as in an office environment, the requirements for network products differ considerably. In industrial applications, networks are expected to work reliably under extreme conditions, such as electromagnetic interference, high operating temperatures and mechanical loads.

Redundancy

The RNRP protocol is based on alternative redundant paths between systems in order to quickly respond to network failures. If one path becomes faulty, another path will be used instantly. All paths use different physical networks to maximize the redundancy.



Devices connected with redundancy must have one interface to the primary network and one to the secondary network. The node number must be the same on both networks.

A Control Network may contain both redundant and non-redundant network areas. Moreover, nodes with redundant interfaces and those with a single interface can be

mixed in the same network area. A node with only one interface must be connected to the primary network



Network applications must always have address nodes on the primary network. In the case of an error the RNRP redirects traffic to the secondary network without involving an application program.

CPU Redundancy

AC 800M can have two redundant processor units working in dual CPU mode with the same functionality as a system running with only one CPU. The backup CPU is running in standby mode, ready to smoothly take over execution from the primary CPU in case of hardware failure.



CPU redundancy in a non-redundant network requires that Ethernet port 1 (CN1) on both CPU units are connected to **the same** network.

CPU redundancy in a redundant network requires that Ethernet port 1 (CN1) on both units are connected to the primary network, and Ethernet port 2 on both units are connected to the secondary network.

The assignment of IP addresses for the Ethernet ports (CN1 and CN2) of the primary CPU unit is performed from Control Builder, in the same manner as for a non-redundant processor unit. The Ethernet ports of the secondary CPU unit will have to be assigned another IP address, using the IPConfig tool. The IP address of the secondary CPU unit is only used for internal communication and is never used by other nodes in the control network. When the backup processor becomes the primary processor, it automatically takes over the primary IP address. In this way, the IP address used for communication throughout the network stays the same.



Which IP addresses to use for the secondary CPU unit depends on the strategy used for redundant units throughout the network. For more information about how to set IP addresses, see online help for the IPConfig tool.

The user application need not be aware of the redundant CPU option. It is possible to reconfigure a system running in single mode to include a backup CPU without any changes to the application.

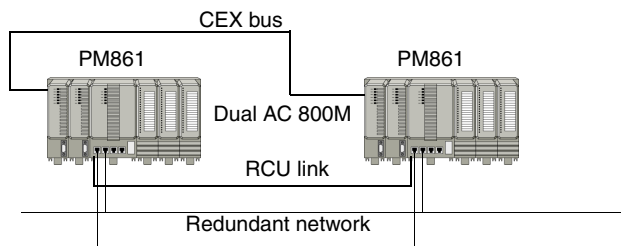


Figure 16. Example of redundant CPU configuration.

Performance

Performance is affected by *transmission speed*, *message length* and *application load*.

For Ethernet channels AC 800M currently supports 10 Mbit/s transmission speed (half duplex). For RS-232C channels the *baud rate* can be selected between 2400 and 19200 bit/s. To send one byte requires 11 bits (start bit, 8 data bits, parity bit and stop bit). Consequently $9600/11 = 872$ B/s can be sent if the baud rate is 9600.

In AC 800M, servicing the S800 I/O via ModuleBus has highest priority. Execution of the application program (IEC 61131-3 code written by the user) has next highest priority. Communication handling in the controller has the lowest priority.

If a IEC 61131-3 task executes continuously for a long time, the controller communication will go through latency. If the continuous task execution is very long (> 300 ms), it results in resends on the TCP/IP level.

The MMS performance decreases during download of applications to the controller and controller configuration.

If the MMS requests exceed the available capacity, the communication rate slows down and it adjusts to the available capacity. In that case, the requested communication cycle time will not be fulfilled.

Master functionality is implemented by function blocks provided by the communication libraries, such as MMSWrite and MMSRead-used to write/read data between controllers. In a system acting as master, the communication

performance is of course affected by the execution interval of the communication function blocks in the application program. The response is handled in the background and is not triggered by the application program in the slave system, but slowed if the application load is high.



If the network is disconnected from the controller, the parameter *Valid* is true for 15 seconds. During this time, the MMSRead block will show Valid and status 2 (=success with warning). This means that if there has not been any traffic within a period the TCP will strobe the other end to check if the connection is up. The following settings have been selected for a connection:

The period time is 7 seconds, the strobe is 8 messages with one second separating them. This will give a time-out of: 7 seconds + 8 messages * 1 second = 15 seconds.

A long message takes longer to transmit than a short one, but it is always more efficient to use long messages if a large data area is to be transmitted.

With the MMS protocol, the maximum message size available for user data when communicating between controllers is 997 bytes when reading information and 957 bytes when writing. The table below is calculated for read operations (997 bytes). This limitation does not apply to MMS communication between applications in the same controller.

Variables require different amounts of message space depending on the variable type (see [Table 5](#)). In addition, the message header requires 60-70 bytes.



The Ethernet standard allows bandwidth transmission at 10 Mbit/s, 100 Mbit/s (fast Ethernet), and 1000 Mbit/s (gigabit Ethernet), but the Control Software currently supports only 10 Mbit/s (half duplex).

Table 5. Space requirements of different variable types

Variable type	Size in telegram	Maximum number of variables in one telegram
Bool	3 bytes	332
Dint, Int, word, dword	6 bytes	159
Real	7 bytes	142
String	4 bytes header + 1 byte/character	190 string [1] 6 string [140]
Struct	4 bytes header + components as above	



The maximum value for number of variables are valid for read request. A write request also includes the access variable name strings in the message.

Limitations

- Redundancy on PPP-link is not supported.
 - PPP does not support online upgrade.
- A maximum of four PPP links are allowed: One tool port link and one PPP link (integrated in the CPU unit), plus additional PPP links via a CI853 unit, can be used.
- CPU redundancy is not supported for the PM851, PM856 and PM860 AC 800M processor units.
- Routing is only allowed in the following situations:

- Routing of MMS via an OPC Server PC between Control Network and the Operator Workplace network is allowed.
- Routing via PPP from one controller to another is allowed, but only to the far ends in the network (only one hop). Using PPP to connect different Ethernet Control Networks is not allowed.
- The maximum number of RNRP nodes in a network area is limited to 50.

Advanced

Default Gateway

Systems that do not need network redundancy can, instead of RNRP, use the alternative routing method with a *default gateway* configured in each controller.

In the example ([Figure 17](#)) the controllers on the 3 control networks can communicate with each other, and the Control Builder can reach all controllers even though RNRP is not used on the controller backbone. However, the routers do need to be configured in some way to know about each others networks and the Control Builder PC also needs this information. This configuration can be done manually or by using a standard routing protocol such as for example OSPF or RIP. To use this in a PC typically a Windows Server Operating System is needed.

For each controller the parameter *Default Gateway* should be set to the control network address of the router, see [Figure 18](#). RNRP and routing with default gateway can be used simultaneously. In the example this could be used if the control networks need to be redundant, but the backbone and the connection to it does not.

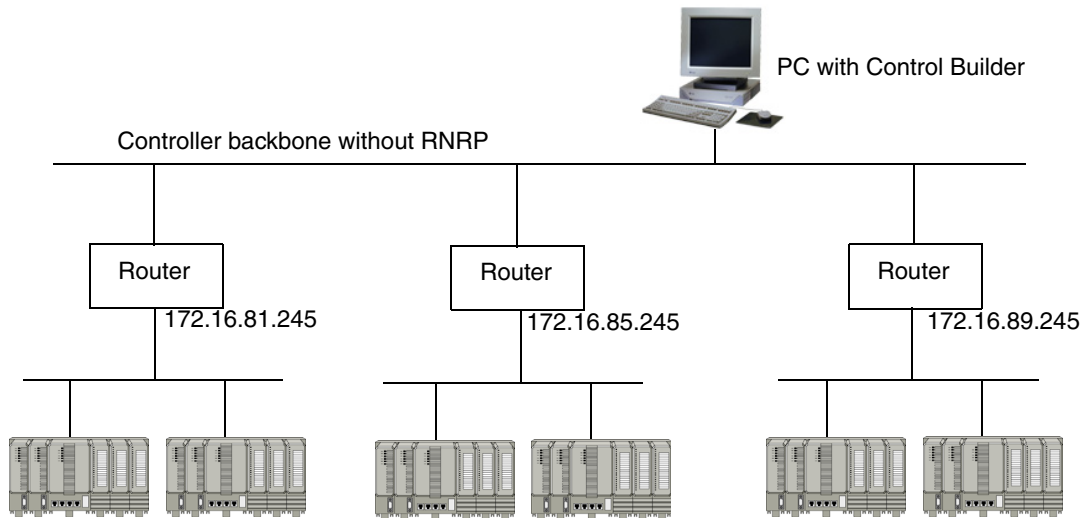


Figure 17. Routing to external servers.

Parameter	Value	Type	Unit	Min	Max
Enable SattBus on TCP/IP	false	bool			
Routing method	default gateway	enum			
Default gateway	172.16.85.245	string			
Event queue size	300	dint		10	3000
Event subscription queue size	1000	dint		200	3000
Max number of condition names	50	dint		10	200
Max number of subscriptions	2	dint		0	5
Event subscription time out	360	dint	min	0	1440
RNRP Default network ID	172.16.0.0	string			
RNRP Number of own areas	2	dint		0	31
RNRP Number of remote areas	4	dint		0	31
RNRP Max Lost Messages	3	dint		0	10
RNRP Send Period	2	dint	s	1	60
RNRP Max no of hops	3	dint		1	3
RNRP System type	controller	enum			

Figure 18. Parameter list specifying default gateway.



It is possible to define two static routing paths. These are defined using the IPConfig help tool, which is installed with the system and can be accessed via the Windows Start menu. For more information, see online help for IPConfig.

Default Process Number

When a PC has several applications running and another PC wants to communicate with either of these applications, a process number must be taken into consideration. This number is added to the system ID, separated by a colon, for example 172.16.84.1:2.

For default process numbers, see [Table 6](#).

Table 6. Default process number.

Product	Default process number
MMS Server	0
Control Builder	1 ¹
OPC Server	22
Tool Routing	30
AC 800M	1
Soft Controller	2

¹ The MMS process number of the Control Builder process is usually 1. It will still be 1 for a Control Builder session that is executing locally on the terminal server console. For a remote Control Builder session the MMS process number will be in the interval of 31-40. The MMS process number will be 31 for the first remote Control Builder session, 32 for the second and so on.

Troubleshooting

The following sources indicate the communication status of Control Network nodes.

1. The Control Builder hardware tree shows the status of interfaces.
2. In Control Builder, right-click the controller to show Remote System. This is used to list the systems connected to a particular network.

3. Controller log file. Printouts of node failures and complete network failures.
4. In Command Prompt on PC, use the command "ipconfig/all" to list installed interfaces and show routes to accessible networks.
5. RNRP monitor that can be installed from the Industrial IT 800xA DVD.
6. The function block SystemDiagnostics (Basic library), shows Ethernet statistics (number of received/lost and transmitted/lost packages).

Section 3 IAC

Introduction

Inter Application Communication (IAC) is the variable communication between applications (both SIL and Non-SIL). In Control Builder, IAC is implemented using communication variables.

The communication variables are used for cyclic communication, and they are declared and used in three Program Organization Units (POUs):

- Top level diagrams
- Programs
- Top level single control modules

For IAC, these POUs can exist in the same application, the same controller or in another controller (peer to peer).

Characteristics of Communication Variables

The communication variables use cyclic reading of data for communication, based on the client-server concept. In the server, the data is copied-out through the communication variable, after the execution of the code. In the client, the data is copied-in through the communication variable, before the execution of the code.

The IAC is configured by declaring an *out* variable in one POU that supports IAC, and one or more *in* variables (with the same name) in another POU that supports IAC. It is also possible to have bidirectional communication variables, if the relationship between the *in* and *out* variable is one-to-one.



The name of the communication variable with direction *out* must be unique within the 800xA System. This helps to resolve the IP-address between the *in* and *out* variable during compilation.

The communication variables can be used in non-SIL, SIL1-2, and SIL3 applications.

For non-SIL communication, a communication variable need to have a name, a direction, and a data type.

For SIL1-2 or SIL3 communication, a communication variable need to have a name, a Unique Id, a direction and a data type. The *in* variables must additionally have an Expected SIL value of the *out* variable, an Acknowledge Group value, and an ISP value.

Communication variables behave differently depending on where the *in* and *out* variables are placed:

- *In* and *out* variables are in the same application and connected to the same IEC 61131-3 task
 - In this case, the *in* and *out* variable represents the same physical memory location; hence no communication is setup.
- *In* and *out* variables are in the same application, but connected to different IEC 61131-3 tasks; or the *in* and *out* variables are in different applications in the same controller
 - In this case, fast data copying is performed at each IEC 61131-3 task scan for the *in* variable. This type of IAC is called internal IAC, where the data is copied between different memory locations, and this does not involve any real communication. This is controlled by the task time, hence no external communication is setup.
 - If the *in* variable is defined in a SIL3 application and the *out* variable is defined in a lower SIL application, communication is driven by considering the *in* variable as external variable. This results in external IAC, and the client receives new values based on the configured interval time. This case is different from the normal internal IAC.

- *In* and *out* variables are in different applications in different controllers (external communication)
 - In this case, the actual external IAC occurs, and the client receives new values based on the configured interval time. The protocol used is MMS based on User Datagram Protocol (UDP). Five different interval time categories are used and these are configured on the *IAC_MMS* hardware unit in Control Builder (see [Interval Time and Timeout](#) on page 57).

Services Provided

For external IAC, the used protocol is MMS based on UDP (User Datagram Protocol). UDP is a Transport Layer protocol under the OSI Reference Model.

The port number used by IAC is 2757.

Re-transmission Schemes

Since UDP does not handle re-transmission, the re-transmission in IAC is based on the configured interval time of the communication variable:

- For IAC interval time above 500 ms, a re-transmission is performed every 500 ms, if no response is received from the server.
- For IAC interval time below 500 ms, re-transmission is performed according to the interval time, if no response is received from the server.

See also [Interval Time and Timeout](#) on page 57.

Design

In Control Builder, the communication variables for IAC are declared in the diagram editor, program editor or top level single control module editor

If the direction of a communication variable is *out*, the POU that holds this variable is the server. If the direction of a communication variable is *in*, the POU that holds this variable is the client.

The data type of the communication variable can be a simple type or a structured type.

For communication variables with structured data types, bidirectional communication is also possible if the *reverse* attribute is set on the required components of the data type. The components with *reverse* attribute communicate in the backward direction, while others communicate in forward direction. For this type of communication, only one server and one client can be connected (1:1).



If the applications using bidirectional communication variables are of different SIL, the Expected SIL value need to be configured for both *in* variable and *out* variable.

ISP and Acknowledgment

The ISP (Input Set as Predefined) value is also entered while defining the *in* communication variable. This value is assigned for the communication variable if there is an error in communication. For non-SIL applications, if the ISP value is not entered, the last good value is assigned if there is an error in communication. For SIL1-2 and SIL3 applications, entering the ISP value for communication variables is mandatory.

For structured data types, the ISP values can only be set in the data type for each individual component. Hence, it is not possible to configure instance specific ISP values for structured data types.

After fault detection, the communication goes to ISP, and the ISP values are latched. The latched ISP functionality can be configured in the IAC POU editor by a column named Acknowledge Group, for each communication variable.

For Non-SIL communication, the default Acknowledge Group is *auto*, which means that the communication resumes automatically after the fault is removed.

For SIL1-2 or SIL3 communication, the default value is zero, which is not allowed in a SIL application. Therefore, it is mandatory to configure the Acknowledge Group to a value (either *auto* or a specific group ID) . If a group ID is specified, the acknowledgment is performed through the CVAckISP control module, for a particular group or cascaded groups. The CVAckISP control module is available in the BasicLib.

A maximum of 32 communication variables of simple data type can be grouped together with the same group ID. For structured variables, a maximum of 32 components can be grouped together with the same group ID.

The same group ID can be used in several applications, but the scope of acknowledge operation is limited to the particular application.

Interval Time and Timeout

The hardware object *IP*, which is reserved at position 0.5 under the AC 800M controller in the hardware tree in Control Builder, holds the hardware object *IAC MMS*.

The *IAC MMS* object controls the cyclic communication using communication variables.

The interval time for communication variables is classified into five categories — VerySlow, Slow, Normal, Fast, and VeryFast. The category is selected while declaring the communication variables.

It is possible to change the value of interval time (in milliseconds) for these categories in the *IAC MMS* hardware editor in Control Builder.



The shortest IAC interval time is 60ms.

The timeout before the communication variable is assigned the ISP value is also defined corresponding to the interval time category in the *IAC MMS* hardware editor.



The configured interval time and timeout of communication variables are applicable for external communication (communication between controllers), and not for internal communication (communication between applications in the same controller). These values are also applicable for special case of internal communication, when reading from lower SIL (non-SIL/SIL2) to a SIL3 application.

Figure 19 shows the relationship between the configured interval time of communication variable, the task cycle time, and the IEC 61131-3 code execution, in an IAC client.

The copy-in of communication variable value happens just before the IEC 61131-3 code execution, based on the latest response value from the IAC server.

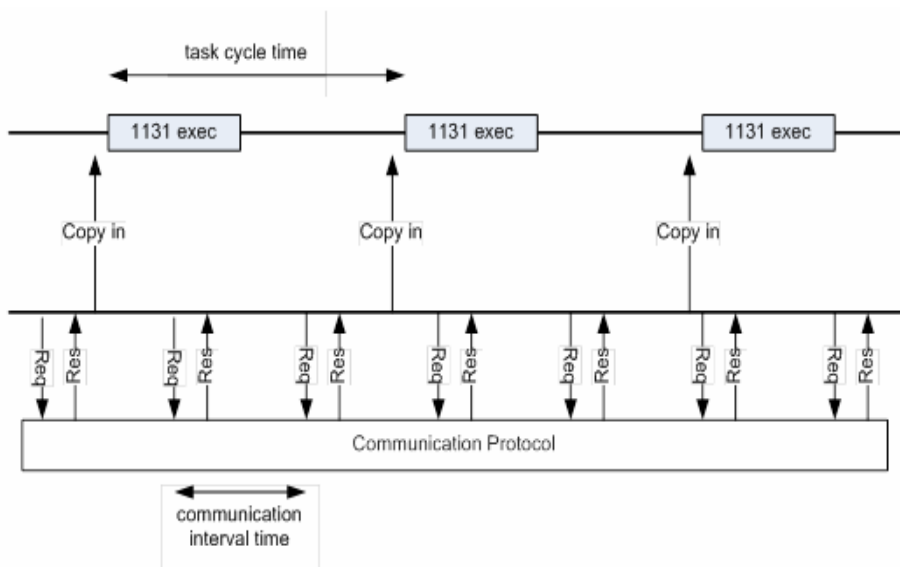


Figure 19. Relationship between interval time and code execution time in IAC client

Resolving the IP address between In and Out Variables

An *In* variable is resolved if a corresponding *out* variable is found within the same 800xA System.

If the IP address is configured as *auto* while declaring an *in* communication variable, and the corresponding *Out* variable is in the same controller or in another controller in the same 800xA system, the IP address of the variable is resolved automatically. However, if an IP address is specified, this address is always used, and the automatic resolve mechanism is not used. This means that the IP address is either specified or it is resolved automatically.

If the *Out* variable is in another controller in another 800xA system in the control network, the IP address is resolved only if this IP address is specified while declaring the *In* variables.

Safety Measures

Inter Application Communication supports up to SIL3 peer-to-peer communication. Safe IAC (IAC involving SIL applications in HI controllers) is implemented according to the IEC 61508 and ISO-13849-1 standards.

Possible Communication Combinations for IAC

Figure 20 shows the different combinations that are possible between client and server for IAC, and the resulting communication status. Safe IAC and its resulting status is highlighted as yellow.

		Client					
		Simulated PA	Simulated HI	Non-Simulated PA (PM86x)	Non-Simulated HI (PM865)	Non-Simulated PA (Soft)	Non-Simulated HI (Soft)
Server	Simulated PA	OK with warning	OK with warning	OK with warning	ISP	OK with warning	OK with warning
	Simulated HI	OK with warning	OK with warning	OK with warning	ISP	OK with warning	OK with warning
	Non-Simulated PA (PM86x)	OK	OK	OK	OK	OK	OK
	Non-Simulated HI (PM865)	OK	OK	OK	OK	OK	OK
	Non-Simulated PA (Soft)	OK with warning	OK with warning	OK with warning	ISP	OK with warning	OK with warning
	Non-Simulated HI (Soft)	OK with warning	OK with warning	OK with warning	ISP	OK with warning	OK with warning

Figure 20. Different combinations of client and server types and the resulting communication status

Unique ID

The Unique ID is a 32-bit unique identifier that is specified when declaring the communication variables in the POU editor. The same ID must be used for both the *in* and *out* variable with the same name, as the *in* variable accepts only a safe communication that contains a matching Unique ID along with the matching name.

The default value for Unique ID is 0. This value is not accepted if the *out* variable is declared in a POU in a SIL application, which means that the UniqueID must be set to a valid value (other than 0) in that case.

Even if the *in* variable is located in a SIL3 or SIL2 application and the *out* variable is located in a non-SIL application, the Unique ID must be specified for the SIL3 or SIL1-2 communication variable.

Redundancy

RNRP is used for redundant communication using IAC.

Online Upgrade

If SIL IAC using communication variables is used for communication between applications, the timeout during Online Upgrade is automatically extended to 30000ms or to the configured timeout value if that is longer. This avoids any interruption in connection during the whole Online Upgrade process.

Performance



With IAC, it is possible to configure the interval time for communication. This makes the communication using IAC faster compared to the peer to peer communication using MMS function blocks.

Cyclic Load and Total Load in Controller

By using IAC, the cyclic load and total load in the controller is reduced compared to usage of MMS function blocks for communication.

The following factors affect the load:

- External IAC
 - In general, external IAC adds to both cyclic load and the total load in the controller. The copy-out/copy-in of variables runs in the context of the IEC 61131-3 execution and adds to the cyclic load, while the communication itself, adds to the total load. The actual copying of data into the application occurs only when new data has arrived.
- Internal IAC
 - For internal IAC, there is no actual communication involved, hence it adds only to the cyclic load through the copy routine (which is called in each scan). An exception to this behavior is when reading to SIL3 from lower SIL, in which case, the loopback communication is running, which adds to the total load as well.
- How the variables are structured
 - Simple variables are faster during copy-in/copy-out (less cyclic load), while the total load in terms of the header information for SIL is more significant. For structured variables it is the opposite; copying the variables out or into the application takes longer, but the impact of header information is less (less total load).
- SIL
 - Server: For a SIL2/SIL3 server, the copy-out operation adds more to cyclic load since the safety measures are added. A SIL3 server also communicates with the SM (Safety Module), which adds to the cyclic load.
 - Client: A SIL2/SIL3 client verifies the safety measures, and this creates extra load. This affects both the total load and the cyclic load (unique ID and expected SIL). A SIL3 client also writes the received data to the SM which adds to the total load.



Demand response time for external IAC, is the sum of the asynchronous operations executed by the 1131 server application, IAC task in client, and 1131 client application. Internal IAC between different tasks does not include any real communication and hence the response time of the communication is only dependent on the 1131 server and client application. For internal IAC within one task, the update of the *in* variable is done immediately as the *out* variable is changed.

Factors that Affect Major Re-Resolve

A major re-resolve of communication variables happens when:

- Downloading the project for the first time
- Downloading the project to a controller that has been reset

All resolved variables are resolved again for controllers that have been reset.

A major re-resolve of communication variables requires more time during compilation of the application.

Limitations



Communication variables are not allowed in distributed application (an application that is executed in several controllers).



Communication variables cannot be directly connected to IO. This means that variables in the IEC 61131-3 application code must be used to transfer values between communication variables and IO.

Message Size

IAC uses UDP for external communication. One communication variable corresponds to one IAC frame. One UDP message can contain one or more IAC frames.

The request message contains the names of the requested communication variables. The response message from the server contains the values and the types of the requested variables. If all communication variables do not fit in one message, several requests are made.

The IAC frame length is different for SIL and Non-SIL. In a SIL IAC frame, the maximum number useful bytes are 78.

The maximum message size of UDP frame is 1400 bytes (for non-SIL).

Structured data types have a header of 6 bytes. Since the type information is in the header and not for each component, the structured data types are more efficient in communicating over the network compared to the simple data types.



For IAC, only simple data types and structured data types of the simple data types are supported. Other types like arrays, queues, and so on. are not supported.

Table 7 shows the message size for the supported data types for communication variables,

Table 7. Message size for supported data types

Data type	Message size (Bytes)
Bool	1
Real	4
Dint, Dword	4
Int, Uint, Word	2
String ⁽¹⁾	1 + String length

(1) Strings are supported for non-SIL and SIL2. It is not possible to declare and use communication variables of simple type string for SIL3, and for structured variables, the SIL3 server skips sending the string contents completely.



If the message size of communication variable exceeds the maximum size, it will be indicated in the status component of the communication variable.

Handling of ISP with String Components When ExpectedSIL is SIL3

In SIL3 communication, if an error occurs, ISP is not set for the string components of a structured data type communication variable. The IAC client skips copying the ISP value of string components when the error is introduced.

The IAC client handles the string components of a structured data type based on the Expected SIL, rather than the actual SIL of the IAC server.

For example, if the Expected SIL is incorrectly set to SIL2, while the server is actually SIL3, there will be an error and ISP is copied also for the string components, since the Expected SIL is known. Later, if the Expected SIL is changed on the client to the correct value (SIL3) and downloaded, the IAC client will start copying values except for strings. But, the string components will continue to show ISP values.

Troubleshooting

Each communication variable has a built-in status word. For external communication, the status goes to Uncertain if no response is received within one interval time cycle, and to Bad if no response is received within the timeout period.

IAC uses the built-in alarm and event handling for the unit status on the hardware unit. Whenever a connection is down or data is not communicated in time (timeout), the unit status indicates this. A system alarm is also generated when a connection goes down and events are triggered if timeout occurs.

There are two methods to supervise the status of communication variables:

- Using the *:status* notation. This is used for communication variables in non-SIL applications.

For example,

```
dword1:=CVMain:status;
```

In this example, the *:status* notation is used to obtain the status of the communication variable, *CVMain*. The status appears as *dword*.

- Using the *GetCVStatus* function (available in System library). This is used for communication variables in SIL1-2, SIL3, and non-SIL applications.

The *GetCVStatus* function accepts the communication variable as input, and provides the complete status and extracted statuses through different output parameters. These output parameters for extracted statuses can be connected to variables to control the logic (critical loop) in the SIL code.

The available parameters for extracting the status are:

- **Quality** – (dword) – The OPC quality of communication.
- **ServerSIL** – (dint) – The SIL of the server application (the application that holds the *out* communication variable).
- **ManualAckRequired** – (bool) – Whether manual acknowledgment is required for the communication variable to restart the communication after failure.
- **ServerInOLU** – (bool) – Whether an Online Upgrade switch is in progress in server or client.
- **ServerIsSimulated** – (bool) – Whether the controller that runs the server application is either hardware simulated or Soft Controller.

The complete status code also indicates the errors (if any). For details about interpreting the complete status code, refer to *System 800xA Control, AC 800M Configuration (3BSE035980*)* manual.

Before downloading, a check can be made in the editor of the POU to verify the proper declaration of the communication variable. While downloading the application, the application compiler detects errors related to communication variables within the application and aborts the download if bad data has been entered.

The usage of communication variables is also checked while compilation. It is not allowed to use a communication variable with lower *Expected SIL* in ST and SFC code, or to connect it to ports in FD without a graphical representation of the variable as a block.

A diagnostic tool for communication variables is also available in Control Builder. This tool can be accessed from the Remote System dialog of the controller (right click the controller, select **Remote System**, and click **Show Diagnostics for Communication Variables**). For details, refer to *System 800xA Control, AC 800M Configuration (3BSE035980*)* manual.

Section 4 MasterBus 300

Introduction

MasterBus 300 (MB 300) can be used for communication between AC 800M and AC 400 Master, MasterPiece 200 or other AC 800M controllers. A communication unit CI855 for AC 800M provides connectivity to AC 400 via MB 300. Refer to the relevant user's guides and reference manuals regarding the process interface that can be used with AC 400.

The CI855 unit is configured by means of Control Builder in the hardware configuration tree.

CI855 has two Ethernet channels to provide network redundancy.

Services Provided

- DataSet (DS) communication with other controllers on MasterBus 300.
- Function blocks in the AC 800M are used to cyclically send and receive DataSets on MB 300.
- Time synchronization on MB 300 is supported in the AC 800M with the accuracy provided on MB 300.
- The CI855 unit status, watchdog supervision and logged system messages are reported to the AC 800M for display in the Control Builder and the Plant Explorer Workplace status system.
- Support of MB 300 network redundancy.

Design

Introduction

The three function blocks *MB300Connect*, *MB300DSSend* and *MB300DSReceive*, handle communication between DataSets belonging to different controllers connected to MasterBus 300. A DataSet consists of an address part and up to 24 elements (32-bit values). A value can be a 32-bit integer, a 16-bit integer, a real or 32 Boolean. The address part is the destination network node, the source network and a DataSet identity.

Each CI855 unit behaves as a unique node on the MB 300 network to which it is connected and must be configured accordingly in the Control Builder hardware configuration tree. Parameters downloaded to the CI855 are:

- A personal node number
- Network numbers for the two network links
- The MB 300 Protocol type, i.e. MB 300, MB 300E or MB 300F
- Clock-synchronization function

Design Example

An AC 800M controller is connected to a redundant MasterBus 300 network via a CI855 unit and can exchange DataSets with other controllers connected to the MasterBus 300 network, such as AC 410 and AC 450. The same AC 800M controller can also communicate with other controllers on Control Network. For small applications, MB 300 and Control Network may use the same physical Ethernet cable.

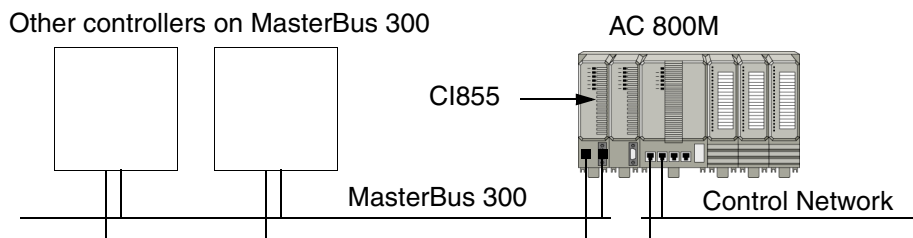


Figure 21. AC 800M connected to MasterBus 300 and Control Network.



MasterBus 300 network may be either redundant or singular.

Communication Function Blocks

An AC 800M on Control Network connects to a controller on MB 300 by means of an *MB300Connect* function block. The *MB300DSReceive* and *MB300DSSend* function blocks with the same *Id* parameter value as the *MB300Connect* function block can then be used repeatedly for communication with that controller. See the example in [Figure 22](#). Refer to the online help for an explanation of the function block parameters.

The *CIPos* parameter specifies the position number of the *CI855* unit in the hardware tree (identical to its position on the CEX bus). *CAPos* specifies the MB 300 network number, and *NodePos* the node position of the controller on the network. *DataSetId* is an integer specifying the DataSet identity. *SupTime* specifies the time interval between receive or send operations. The extensible parameters *Rd* and *Sd* of *AnyType* data type indicate the total number of application variable names. They allow the user to specify personal parameters, the only restriction being that the total number of parameters must equal the total number of allocated elements in the DataSet.

[Table 8](#) indicates the mapping between data types used in AC 800M and data types used in other controllers on MB 300.

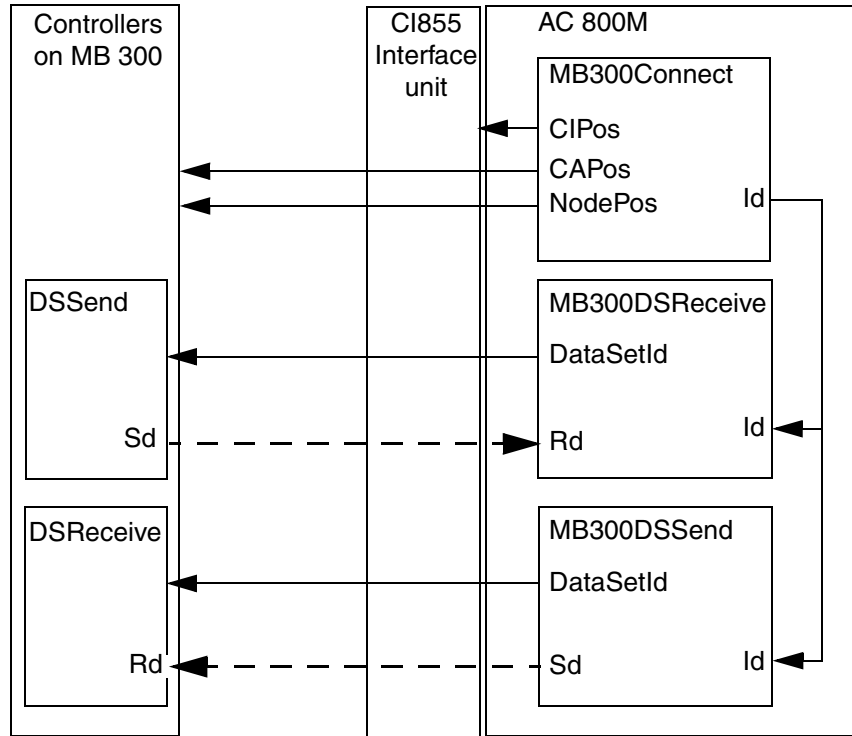


Figure 22. Exchange of DataSets (DS) via MB 300 by means of function blocks.

Table 8. Mapping of data types

Data types in other controllers on MB 300	Data types in AC 800M
Boolean32	dint
16-bit integer	int
32-bit integer	dint
real	real

Redundancy

The CI855 unit houses two Ethernet channels to provide network redundancy. The routing tables in CI855 that indicate the network, node address and port to use when sending to an MB 300 node, are continuously recalculated according to the latest topology information in the routing messages. In the case of link/node failures, switch-over to redundant links is automatic.

Limitations

MasterBus 300 in AC 800M is used for communication with other nodes such as AC 400 Master, MP 200 and AC 800M.

A DataSet consists of up to 24 elements (32-bit values).

Performance

Transmission speed: 200 packets/s.

Clock synchronization: 3 ms.

Hardware

- MasterBus 300 interface unit CI855 connects to the CEX bus of the AC 800M.
- Twisted pair 10BASE-T Ethernet cable with RJ45 connector is used. The installation should comply with Category 5 specification according to IEEE 802.3.

Advanced

Time synchronization on MasterBus 300 is supported in the AC 800M by the accuracy provided on MB 300.

The CI855 editor in the Control Builder is used to specify the clock synchronization mode:

- No synchronization
- CI855 is synchronized by AC 800M:
 - CI855 does not synchronize MB 300 network.
- CI855 is synchronized by MB 300:
 - AC 800M may be synchronized by CI855.
- CI855 is synchronized by AC 800M:
 - CI855 is clock-master in the MB 300 network.

If AC 800M is to be synchronized from the CI855 unit, it is also necessary to select MB 300 as clock synchronization type in the CPU hardware editor.

Troubleshooting

The CI855 device status, watchdog supervision and logged system messages are reported to the AC 800M for displaying in the Control Builder and Plant Explorer Workplace status system.

Watchdog mechanisms are used by the AC 800M to supervise the CI855, which supervises the AC 800M. The watchdog function is cyclically called and interrupts the CI855 unit, which, if it does not receive an interrupt within a certain time, stops communication at its ports. The CI855 responds with a watchdog signal to the AC 800M, which expects the CI855 unit to cyclically generate watchdog interrupts. The unit is considered out of function if an interrupt is missing. The CI855 operating system has its own watchdog/stall handling, which will halt the CI855 processor in the event of hardware or software errors.

Section 5 COMLI

Introduction

COMLI (COMMunication LInk) is a standard protocol for data transmission/communication between controllers. It is a conventional communication link using serial, asynchronous data transmission according to the master/slave principle, in one direction at a time (half duplex mode). It is used for reading and writing variables between control network devices, using point-to-point communication or multidrop communication. COMLI can be used in serial communication and in SattBus-TCP/IP communication.

COMLI is suitable for communication with controllers such as SattCon 05, 31, 35 or 200. COMLI ensures that:

- Maximum use is made of the communication line, resulting in compact storage of data transmitted or received,
- Transmission is secured by checking every character as well as the entire message.

Services Provided

Master

- COMLI ReadPhys (Read Physical Value) (message G)
- COMLI WriteDT (Write Date and Time) (message J)
- Read and Write in registers and bits (messages 0, 2, 3, 4)
- Read and Write in high registers (message <, =)

Slave

- Read and Write in registers and bits (messages 0, 2, 3, 4)
- Read and Write in high registers (message <, =)
- COMLI WriteDT (Write Date and Time) (message J)

Design

Introduction

Master and slave can be linked together in two different ways to achieve the desired function. They are:

- Multidrop (multipoint) communication.
- Point-to-point communication.

Master and slave are linked through the serial channels on the different systems that are communicating with each other. The master and slave need not use the same physical channel numbers in both systems. They must, however, have the same character format, transmission speed, and so on.

When the slave receives a message, it responds either by sending the information requested or by acknowledging the information received. The slave does not respond if it receives a faulty message.

To change the status of a system/device from master to slave, a new configuration must be downloaded from Control Builder.

Design Examples

Multidrop Communication

In multidrop communication several slave systems are connected to a master, see [Figure 23](#). Communication takes place between the master and one slave at a time. Direct communication between slave systems is not possible. A particular message from the master is sent to all slaves, but only the slave, whose unique identity corresponds to the identity contained in the message accepts the data.

The number of slaves that can be connected to each master is limited by the communication interface. A RS-232C converter (for example to RS-485) must be used in multidrop communication. When using RS-485, both 2-line and 4-line connection can be used. When using 4-line connection, the master transmit line is connected to all slave receive lines and all slave transmit lines are made common and connected to the master receive line.

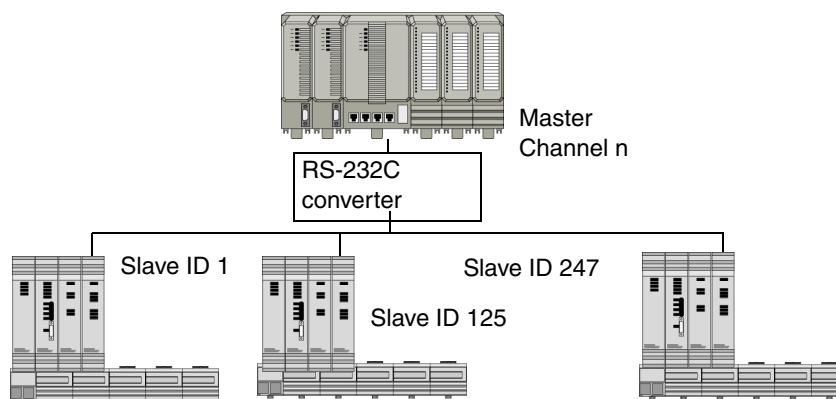


Figure 23. Example of multidrop communication.

Point-to-Point Communication

In point-to-point communication, only one slave system is connected to the master through one communication interface, see [Figure 24](#). Several slaves can be connected to the master, but this must take place through different communication interfaces. This form of point-to-point configuration can reach a capacity higher than that achieved with multidrop communication.

The electrical interface is RS-232C or any other interface using an appropriate converter.

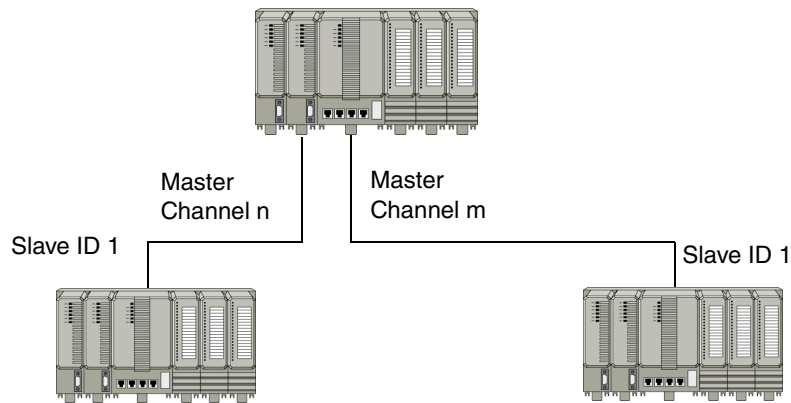


Figure 24. Example of point-to-point communication.

Redundancy

Redundancy is not built in, but can be implemented on application level or physical (cable) level by the user, by adding another CI853 module and configure the necessary redundant functions in the application program.

Limitations

- Using a RS-485 converter, a maximum of 31 slave systems can be connected to each serial channel.
- The COMLI communication protocol can administer a maximum of 247 slave identities, see [Figure 23](#)).
- The maximum message size is 512 bits or 32 16-bit registers (integer reading).
- The range for COMLI address is 0-247.
The master address is 0 and the slave address is in the range 1-247.

Performance

Performance is affected by *transmission speed, message length and application load*.

For RS-232C channels a *baud rate* can be selected between 2400 and 19200 bit/s. To send one byte requires 11 bits (start bit, 8 data bits, parity bit and stop bit). Consequently $9600/11 = 872$ bytes per second can be sent if the baud rate is 9600.

The maximum transmission distance depends on the interface used and the transmission speed. Use RS-232C for short distances, maximum 15 meters. Use an appropriate converter for longer distances. Note that a noisy electrical environment may require shorter distances.

In AC 800M, servicing the S800 I/O via ModuleBus has highest priority. Execution of the application program (IEC 61131-3 code written by the user) has next highest priority. Communication handling has lowest priority, and communication performance is therefore affected by the load from higher prioritized tasks. Communication takes place serially and asynchronously according to the master/slave (or client/server) principle. The *master channel* of a system initiates the message transmission sequence, while a system acting as a slave simply responds to the calls from the master via a *slave channel*.

Master functionality is implemented by function blocks provided by the communication libraries, such as COMLIWrite and COMLIRead, used to write/read data between controllers. In a system acting as master, the communication performance is of course affected by the execution interval of the communication function blocks in the application program. Response is handled in the background; it is not triggered by the application program in the slave system, but is slowed down if the application load is high.

Register and I/O bits are two terms referring to the COMLI protocol. Registers (16 bit integers 0-65535) are mapped to the data types *int* (integer, 16 bits), *dint* (double integer, 32 bits) or *uint* (unsigned integer, 16 bits) in the Control Builder environment, and I/O bits are mapped to the data type *bool* (Boolean). If writing a *dint* variable it has to fit into 16 bits, otherwise COMLI will return an error message. All parameters (both registers and I/O bits) must be of the same type if sending several values in one function block. That is mixing of variables of different types (*int*, *dint*, *uint* or *bool*) within the same function block is not allowed. The maximum number of registers that can be sent in one message is 32, independent of the

variable type. Boolean variables must be transferred either as a single variable or in multiples of eight, maximum 512. A long message takes longer to transmit than a short one, but it is always more efficient to use long messages if a large data area is to be transmitted.

Hardware

COMLI can be used on the built in COM3 port and optionally on the CI853 ports. The cable length can be extended considerably (to several km) using a fiber optic converter. RS-232C is the standard communication interface used for serial communication with COMLI. The CI853 supports Hot Swap.

Advanced

Procedure for Linking Control Systems with COMLI

The procedure for connecting different control systems to a common COMLI communication network can be summarized as follows:

- Select the message types by establishing the type of information to be transmitted between systems.
- Select the network configuration. Select multidrop or point-to-point and the communication interface to be used.
- Select the channel (channels) to be used. The choice depends on which channels are available and whether the channel can transfer the required information at the required speed.
- Decide which system is to be master and which is to be slave. This is specified in the channel parameter list. The master controls data transmission operations since only the master can initiate a message transmission sequence.
- Connect the systems to the network with suitable cables.

When the above has been completed, the various communication parameters can be defined in a number of data fields.

For further information about COMLI, refer to the COMLI System Description (493-0192-11).

Section 6 SattBus on TCP/IP

Introduction

SattBus is a communication network for linking controllers, intelligent I/O devices, sensors, etc.

SattBus is an open protocol that can be implemented by any manufacturer. Due to its low memory requirements, SattBus can be integrated with an application in a single-chip microcontroller. Different types of interfaces, for example for PCs, are also available.

The multimaster operation allows event communication, which increases the efficiency and lowers utilization of the network. SattBus is optimized for the transfer of small amounts of data. All this contributes to making SattBus a high-performance network for systems with highly distributed data reaching a maximum effective data transmission rate of 3000 bytes per second.

Services Provided

SattBus provides the following services:

- Variable names.
- Managing and accessing variables in remote nodes.
- COMLI transparent messages over SattBus:
 - Master: COMLI ReadPhys (Read Physical Value) (message G)
 - Master: COMLI WriteDT (Write Date and Time) (message J)
 - Master: Read and Write in registers and bits (messages 0, 2, 3, 4)
 - Slave: Only Read and Write in registers and bits (messages 0, 2, 3, 4)

In total, 16 services are defined in the SattBus protocol. The most important ones relate to variable transfer.

All nodes on SattBus are equipped with the basic ability to identify the node by a short, five character name. This service also provides the program version and defines the other SattBus services the node can handle. Nodes with different sets of variables have different identities.

Design

Introduction

Communication is performed via SattBus function blocks found in the SattBus communication library: SBConnect, SBRead, SBWrite and ComliSB function blocks (ComliSBConnect, ComliSBRead, ComliSBReadCyc, ComliSBReadPhys, ComliSBWrite, ComliSBWriteDT). ComliSB function blocks are used for directly addressed communication (e.g. X100, R1000). SBRead/SBWrite are used for named SattBus communication.

In named SattBus, a structured variable can have 254 components of simple data types.

Redundancy

SattBus does not support redundancy.

Limitations

Physical SattBus is not supported.

For SattBus on a TCP/IP connection, the valid node number range is 2-127, for example '88'.

Performance

If each node is allowed a maximum of one message frame per token rotation, then the SattBus data link layer can transfer up to 3000bps within message frames. Transfer efficiency is over 30%. SattBus is stable in an overload situation, that is the throughput does not decrease as the load increases and the system does not become blocked.

Advanced

SattBus communication uses the Ethernet network, SattBus messages are transferred using TCP/IP. Communication is also possible using COMLI function blocks via SattBus on TCP/IP.

SattBus application messages are sent 'as is' using the User Datagram Protocol (UDP) of the TCP/IP suite. A small heading is added for a transport protocol implemented on top of UDP. This protocol is responsible for sequence and transport control, assuring that SattBus messages are received in order, and that they are transmitted up to 4 times until they are acknowledged (on the transport level) by the receiver.

The node status supervision of physical SattBus is simulated on the transport level above UDP by sending background supervision traffic a number of times per minute (in the absence of other traffic). This enables node status reports to perform in a similar way to physical SattBus, although the time resolution is lower. However, this applies only to nodes where logical connections exists.

Section 7 INSUM

Introduction

INSUM (INtegrated System for User-optimized Motor control) utilizes microprocessor-based technology for protection and control of motors and switchgear, and for the transmission of messages and measured values.

There are different device types in the INSUM System. The AC 800M can communicate with the most commonly used device types:

- Motor Control Unit (MCU),
- Circuit Breaker.

The INSUM devices are connected to a 78 kbit/s LonWorks Network. Up to four of these networks can be connected together via routers in a switchgear unit. The routers and the other devices in the switchgear unit communicates with 1.25 Mbit/s. In the switchgear unit the following device types can be connected:

- Routers for one or two 78 kbit/s LonWorks subnets,
- Man Machine Interface (MMI) for commissioning and maintenance,
- TCP/IP Gateway,
- System Clock.

The INSUM TCP/IP Gateway is used when the INSUM System is connected to the 800xA system via the CI857 module in the AC 800M controller.

Services Provided

- Multiple controllers can access the same MCU in an INSUM system.
- There are three IEC 61131-3 function blocks for communication with the INSUM system:
 - INSUMConnect: Establishes connection,
 - INSUMReceive: Reads a process data value from an INSUM device,
 - INSUMWrite: Writes a value to an INSUM device.
- A number of different motor types are supported, such as reversing motors, two-speed drives, actuators, and solenoid valves.
- Protection against thermal overload, underload, phase loss, ground fault, high motor temperature, locked rotor, etc.
- Protection functions can be parameterized to specify pre-warnings before a motor is tripped. The reset can be selected as *automatic*, *remote*, *local* or *remote and local*.

Design

Introduction

INSUM hardware is configured with the Project Explorer (see [Figure 25](#)). The AC 800M is equipped with two INSUM TCP/IP CI857 interface units, located as numbers 3 and 4 to the left of the PM860 CPU unit. Three INSUM gateways are connected to unit number 3. Each gateway is supervising an INSUM motor control system.

Gateway number 2 has three MCUs and one tier switch connected to its subnet number 1, with node numbers 1, 2, 3 and 32. Two MCUs are also connected to subnet number 2 and one circuit breaker to subnet number 4.

The CI857 units and the INSUM gateways have IP addresses that must be specified in the parameter lists.

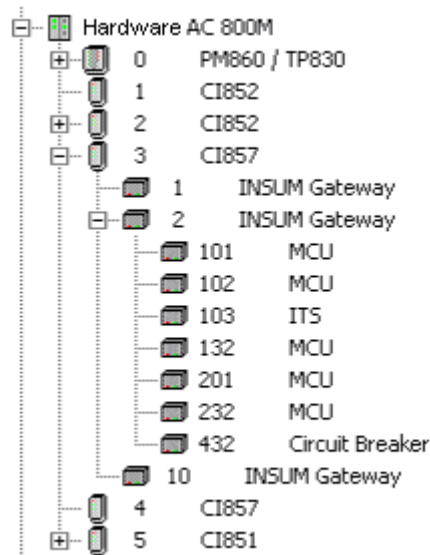


Figure 25. Project Explorer.

Design Example

The AC 800M controllers use the CI857 to connect to an INSUM Ethernet network running TCP/IP at 10 Mbit/s half duplex. An INSUM TCP/IP gateway connects the Ethernet to the LonWorks network that communicates via routers with the INSUM devices arranged in four subnets. A complete INSUM system is shown with 128 INSUM devices (motor control units, circuit breakers and intelligent tier switches [ITS]). An MMI (man-machine interface) is connected to LonWorks.



Due to performance considerations, the Control Network and the TCP/IP Ethernet between CI857 and the INSUM TCP/IP gateways must be kept separate. Also, see INSUM documentation.

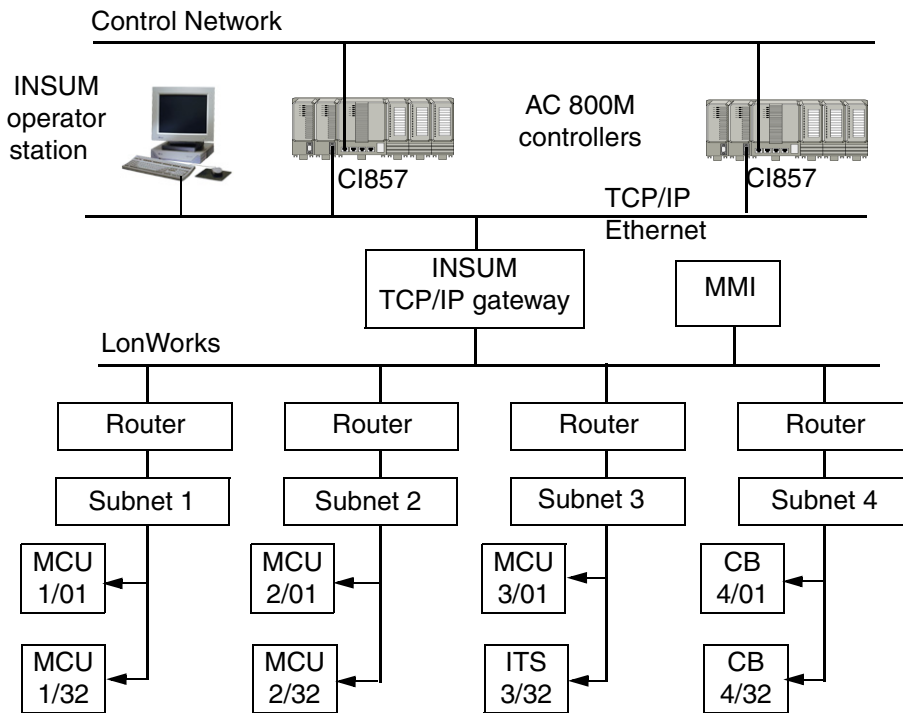


Figure 26. INSUM integration with Control Network.

INSUM Alarms

All INSUM devices (MCU, Circuit Breaker) have supervision functions that can report alarms. The different device types supervise and report specific alarm types. The alarms are reported in specific Network Variables.

MCUs report the alarms in the Network Variable NVAlarmReport.

The user can decide if there should be a summary entry that tells that there are some alarms (one or more) in the device. From System Version 4.1 it is possible to have a separate summary alarm for warnings and a separate alarm for trips. For more information about INSUM alarms, refer to the manual *System 800xA Control AC 800M Configuration (3BSE035980*)*.

Redundancy

Redundancy is not built in, but can be implemented on the application level by the user. The ethernet connection between CI857 and the INSUM TCP/IP Gateway can be made redundant by using network switches with a redundancy protocol, for example ring redundancy.

Limitations

INSUM system limits:

- Maximum 128 INSUM devices per INSUM TCP/IP gateway.
- Four LonWorks subnets per INSUM TCP/IP gateway.
- Maximum 32 devices per LonWorks subnet.
- Circuit breakers (CB) require a special router and cannot be mixed with other INSUM devices on the same subnet.
- Time Synchronization of the INSUM system via TCP/IP is not supported. There is however an INSUM system clock for time synchronization from GPS.

Limitations due to CPU load and memory consumption in the AC 800M (verified with PM861):

- Maximum 128 MCUs (or other INSUM devices) per AC 800M.
- Maximum 6 CI857 per AC 800M.

Limitations due to CPU load and memory consumption in the CI857:

- Maximum 128 MCUs (or other INSUM devices) per CI857.
- Maximum 2 INSUM TCP/IP Gateways per CI857.

Performance

It is possible to start or stop one MCU based on input from another MCU within 500 ms. It is also possible to start or stop 128 MCUs in one INSUM system and to get feedback within 20 s. Note that this includes the time for the messages with the feedback after the actual reaction of the MCUs.

Hardware

- INSUM TCP/IP interface units CI857 connect to the CEX bus of the AC 800M.
- Twisted pair 10BASE-T Ethernet cable with RJ45 connector. The installation should comply with the Category 5 specification according to IEEE 802.3.
- The LonWorks bus is integrated in the INSUM system backplane.
- INSUM routers and gateways, power supply, motor control units, circuit breakers, tier switches and man-machine interfaces are devices belonging to the INSUM system.
- INSUM routers, gateways and power supplies are connected directly to the INSUM backplane. Motor control units, circuit breakers, tier switches and man-machine interfaces are connected by means of prefabricated cables.
- If the speed and duplex of the network equipment used can be manually configured it is recommended to set it to fixed 10 Mbit/s and half duplex. This gives the best performance. Both the CI857 and the INSUM TCP/IP Gateway use these settings.

Troubleshooting

The INSUM TCP/IP CI857 interface units, the INSUM routers and gateways have LEDs indicating communication activity and unit error states. The Control Builder hardware tree shows the status of the different hardware units. The function blocks have outputs indicating error condition and status code.

The system software handling the INSUM communication in the controller writes diagnostic information to the controller log. The INSUM TCP/IP Gateway writes diagnostic information to a log file that can be fetched with the INSUM OS. The INSUM functions in the controller and CI857 have extra log facilities that can be enabled by system experts.

Section 8 Siemens 3964R

Introduction

Siemens 3964R¹ is a rather widespread communication protocol designed by Siemens. It is a standard serial point-to-point master/slave protocol. No special hardware is required. Siemens 3964R is convenient for communication with instruments (e.g. scales) or controllers also using this protocol.

Services Provided

- Multiple registers can be read/written.
- Multiple I/O bits can be read.
- One message can handle a maximum of 512 I/O bits or a maximum of 32 registers.
- Writing of single I/O bits is also supported, with the following limitation: Writing of a single I/O bit is done to data block 222, using a special bit message which is not implemented in Siemens 3964R. Special application software is needed in the Siemens system to handle this. It is possible to change the data block via the SiemensBitTransferDB system variable in the controller.
- Messages can have 32 registers, but they must not exceed data block boundaries. This means that the data blocks in Siemens systems communicating with this system are limited to data blocks 3-14.

1. Protocol 3964R can be distinguished from 3964 simply by the presence of the control character (BCC), providing more reliable transmission.

Table 9. Services provided.

Service	Direction	Comment
“E” message, data type D	Controller to Siemens	Request for data, register
“E” message, data type E, A, M	Controller to Siemens	Request for data, byte
“E” message, data type E, A, M	Controller to Siemens	Request for data, bit
“E” message, data type D, E, A, M	Siemens to controller	Answer to request for data
“A” message, data type D	Controller to Siemens	Transfer of data, register
“A” message, data type D	Controller to Siemens	Transfer of data, bit
“A” message, data type D	Siemens to controller	Answer to transfer of data

In [Table 9](#), controller means AC 800M acting as the client, and Siemens means Siemens PLC as a server.

Design

Introduction

Communication is performed via function blocks. The controller sends one read or write message to the subsystem and then awaits the answer. This means only one message will be outstanding per channel, i.e. master/slave principle

[Figure 27](#) shows a Siemens 3964R network in principle.

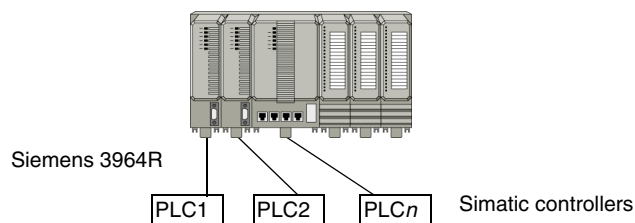


Figure 27. Siemens 3964R network principle.

Before Siemens 3964R communication can be started, the normal RS-232C parameters (baud rate, parity, data bits, stop bits, flow control and delay of the RTS-signal) must be set for the Control Builder Com port. Refer to the Control Builder online help for details (section “Serial (Com) Port Settings”).

Limitations

- The controller can act only as master, i.e. only client functionality is supported for Siemens systems.
- Only point-to-point communication is possible, i.e. only one slave may be connected to each communication channel.
- “Interpreter RK512” must be installed on the Siemens system (the slave).
- Writing of multiple I/O bits is not supported.

Performance

Similar to COMLI, see [Performance](#) on page 77.

Hardware

Siemens 3964R can be used on the build in COM3 port and optionally on the CI853 ports. A standard RS-232C/485 communication channel is required.

Cable lengths:

- RS-232C: 15m,
- RS-485: 1200m.

The length can be extended considerably (to several km), using a fiber optic converter.

Both 2-thread and 4-thread communication can be used for the RS-485 port.

The CI853 supports Hot Swap.

Advanced

Communicating Integers

Integers can be read and written. The value range for integers fetched from controller subsystems is 0-65535, while the range for data words in SIMATIC is -32768 to +32767. This means that a value between 0 and 32766 in an integer that is loaded down to a data word in SIMATIC will be interpreted correctly, while values greater than 32767 will be interpreted as negative.

Messages can have 32 integers but must not exceed data block boundaries. This means that the data blocks in Siemens systems communicating with a controller must keep to data blocks 3-14 (see [Table 10](#)).

Table 10. I/O-bits, integers and data blocks.

I/O bits in the controller (max 512/message)	I/O bits in Siemens	Integers in the controller	Data blocks in Siemens	Word for S5 system	Word for S7 system	
0	Inputs	0.0	0	3	0	0
1		0.1	1	3	1	2
...		3
7		0.7	255	3	255	510
10		1.0	256	4	0	0
...		4
1777		127.7	511	4	255	510
3000	Outputs	0.0
...		...	3071	14	255	510
4777		127.7				
6000	Flag	0.0				
...		...				
11777		255.7				

Communicating Bits

Groups of bits can only be read, **not** written. During reading of bit values and reading and writing of integer values, communication takes place directly with the bit and data block areas in the SIMATIC system. Up to 512 bits per message are handled during reading of bits, and up to 32 integers per message are handled during reading or writing of integers.

When a large number of bits need to be sent concurrently it is better to use registers for the actual transmission and then convert them to and from bits in the controller and the SIMATIC system.

When using function block 3964RWrite to write a single bit to the subsystem, the controller will pack the type of bit value, the bit number and the new value in a word and send it to data word zero in a specific data block in the SIMATIC system.



The message is unpacked by a controller program in the SIMATIC system. This program must always be included in the SIMATIC control system. The program must be executed often enough to be able to handle a message before the next message can arrive from the controller.

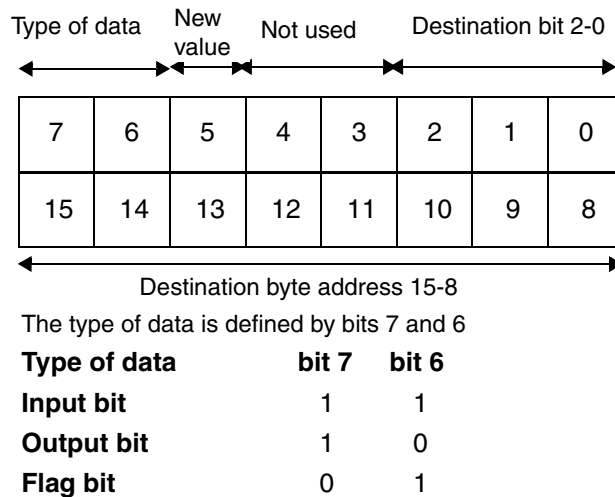


Figure 28. Code word for the writing of a bit.

The receiving data block number in the Siemens system is set in the controller with the system variable *SiemensBitTransferDB*. The default value is 222 and the range 1-255.

Section 9 MODBUS RTU

Introduction

MODBUS RTU is a standard protocol widely spread because of its ease of use and the fact that it supports communication over a wide variety of media, such as wire, fiber optics, radio and the telephone.

MODBUS is executed serially and asynchronously according to the master/slave principle, and in one direction at a time. However, only master functionality is implemented in the AC 800M controller. MODBUS is used mainly for reading and writing variables between control network devices, using point-to-point or multidrop communication. Message framing is implemented in RTU mode, which is a binary format. The MODBUS protocol is designed to transfer data securely by checking each byte as well as the entire message for transmission errors.

Services Provided

A number of MODBUS commands are supported, see [Table 11](#). The application programmer can access the protocol functions through function blocks, according to the IEC 61131-5 standard. The protocol software translates the request from connect, exception, read, and write blocks into MODBUS protocol commands.

Table 11. MODBUS protocol commands

Protocol	Description	Protocol	Description
FC1	Read coil status	FC6	Preset single register
FC2	Read input status	FC7	Read exception status
FC3	Read holding registers	FC8 ⁽¹⁾	Diagnostic request

Table 11. MODBUS protocol commands (Continued)

Protocol	Description	Protocol	Description
FC4	Read input registers	FC15	Force multiple coils
FC5	Force single coil	FC16	Preset multiple registers

(1) Some slaves do not understand FC8. To avoid problems, set Poll Time to zero (0).

Design

Introduction

Communication with MODBUS takes place serially and asynchronously according to the master/slave principle.

- The *master channel* of a system controls the communication of devices with *slave function*.
- A device with *slave function* is connected via a *slave channel* and its communication is controlled from a master channel.



Note that a specific device may have some channels specified for a master and some for a slave. Consequently the device may act as master in relation to some devices and as slave in relation to others. See also [Figure 29](#) & [Figure 30](#) for a graphical view of the concept.

Design Examples

Point-to-Point Communication

Point-to-point communication means that only one slave system is connected to the master channel, as shown in [Figure 29](#).



Figure 29. Master/slave in a point-to-point communication configuration.

Each channel has a channel definition defining the method of communication. Master and slave must not use the same channel number or address for communication. However, the channels must be set up in the same way with regard to character format and baud rate.

Several slaves may be connected to a controller, but via different serial channels (see [Figure 30](#)). This network configuration offers higher speed than multidrop communication. Communication is controlled in the master by defining a communication area which indicates the information the master must transmit to or receive from the slave(s).

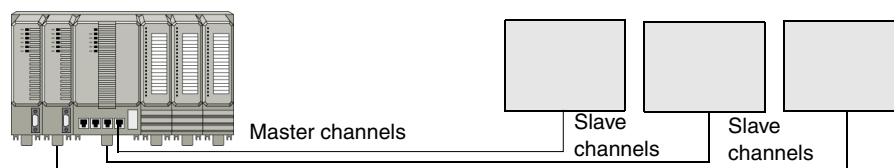


Figure 30. Point-to-point communication between a controller and several slaves.

Multidrop Communication

Multidrop communication means that several slaves are connected in parallel to the serial master channel (see [Figure 31](#)). Note that the master can only address one of the slaves at a time. Also note that, unlike the point-to-point design, only one RS-232C channel with an appropriate converter is needed.

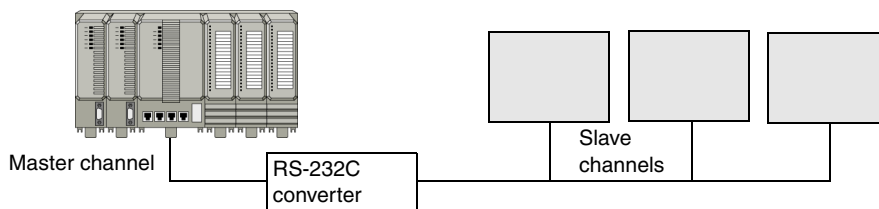


Figure 31. Master/slaves in a multidrop communication configuration.

Hardware

MODBUS RTU can be used on the built-in COM3 port and optionally on the CI853 ports. An RS-232C communication channel is required (and possibly an RS-232C/485 converter). Maximum cable lengths are 15 m for RS-232C and 1200 m for RS-485. Cable length can be extended considerably (to several km) using a fiber optic converter.

The CI853 supports Hot Swap.

Performance

Similar to COMLI, see [Performance](#) on page 77.

Limitations

- Only master functionality is implemented.
- Only RTU mode is implemented (only binary values are used).
- Communication using a dial-up modem is not possible.
- No support for broadcast.

Redundancy

Redundancy is not built in, but can be implemented on application level or physical (cable) level by the user, by adding another CI853 module and configuring the necessary redundant functions in the application program.

Troubleshooting

The operator can monitor the status of all hardware units using the Project Explorer in the Control Builder.

Section 10 MODBUS TCP

Introduction

MODBUS is an open industry standard protocol that is widely used. It is a request response protocol and offers services specified by function codes.

MODBUS TCP combines the MODBUS RTU with standard Ethernet and universal networking standard, TCP. It is an application-layer messaging protocol, positioned at level 7 of the OSI model.

MODBUS TCP is an open Industrial Ethernet network which has been specified by the MODBUS-IDA User Organization in co-operation with the Internet Engineering Task Force (IETF) as an RFC Internet standard.

Services Provided

AC 800M implements Master and the Slave functionality of MODBUS TCP.

The application programmer accesses the protocol functions through standard IEC 61131-5 function Blocks.

The Master implementation uses Functions Blocks like Connect, Exception, Read and Write to translate the request into MODBUS commands.

The Slave implementation uses Access Variable table for response handling.

The AC 800M implementation supports the following MODBUS Function Codes:

Table 12. Supported MODBUS Function Codes

Function code	Name	Supported in
FC 1	Read coils	Master and Slave
FC 2	Read input discreet	Master and Slave
FC 3	Read multiple registers	Master and Slave
FC 4	Read input register	Master and Slave
FC 5	Write coil	Master and Slave
FC 6	Write single register	Master and Slave
FC 7	Read exception status	Master and Slave
FC 8	Diagnostic	Master and Slave
FC 15	Force multiple coils	Master and Slave
FC 16	Write multiple registers	Master and Slave
FC 20	Read file record	Master
FC 21	Write file record	Master
FC 23	Read Write file record	Master
FC 30	Support for real data types	Master

Design

Introduction

The Communication Interface Module CI867 integrates MODBUS TCP into the AC 800M system.

The CI867 is a dual channel Ethernet Module with ports:

- Ch1: compliant to IEEE 802.3u standard for the 10/100 Base-T by the Media Independent Interface.

- Ch2: compliant to IEEE 802.3 standard for 10 Base-T by the 7-wires interface.

Table 13. Key features of MODBUS TCP implementation for AC 800M

Network Topology	The MODBUS does not specify any particular Network topology. All topologies that can be implemented in standard Ethernet (e.g. star, line or tree) including switched Ethernet can be used.
Speed	Ch1: 10/100 Mbps, Full duplex (Auto negotiation) Ch2: 10 Mbps, Half duplex
Maximum Stations per CI867 (sum of Ch1 and Ch2)	Slaves: 70 Master: 8

Design Examples

Figure 32 shows MODBUS TCP network implementation using Ethernet ring topology. For details, refer to *Automation System Network: Design and Configuration (3BSE03446*)*.

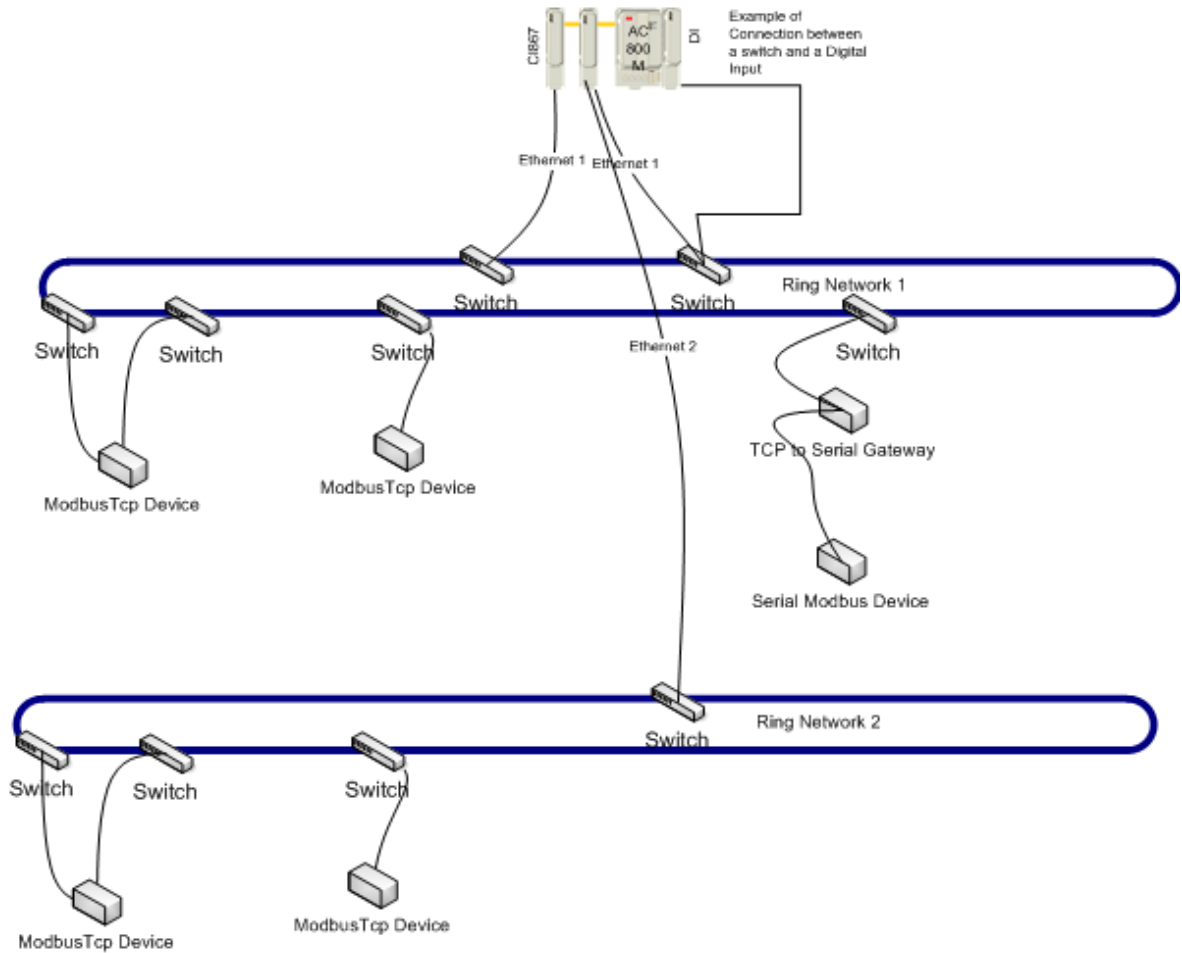


Figure 32. Example of a MODBUS network implementation

Connection Methods

- Use twisted pair 100BASE-T Ethernet cable with RJ45 connector.
- In case of Redundant CI Modules, connect the two CI modules to two different switches for better availability.
- The installation need to be compliant with Category 5 specification according to IEEE802.3.
- Use network hardware as described in ABB document *Third Party HW products verified for Industrial IT System 800xA(3BSE046579*)*.
- The MODBUS TCP network should be an independent network or connected to subnets within 800xA network for security reasons.
- Configure the third party slaves to have the address FF using the suppliers own tool and then connect them one by one to the Controller. Another solution is to add a dummy Gateway object below the CI867, and assign slave's IP address to the Gateway. Place a serial slave object below the Gateway at the same position as the slave's ID.
- Do not connect more than four MODBUS RTU slaves to a single Gateway. This number depends on the application and slave response times.
- The MODBUS TCP network should not be treated in isolation. The network design and IP address allocation need to be done within the overall framework of system 800xA. Refer *Automation System Network: Design and Configuration(3BSE03446*)*.



In Control Builder, the IP address configured in the Settings tab of the Ethernet ports, Gateway, and the Modbus TCP slave, should not have any of its four bytes with leading zeros.

For example, the IP address should be configured as 172.16.4.1, and not as 172.016.004.001

Connection of MODBUS RTU MODBUS Devices

MODBUS RTU slave devices can be connected to MODBUS TCP network by the use of suitable third party Gateways.

Redundancy

The MODBUS TCP supports the module redundancy. A failover never happens if there is a problem with an active port on the backup.

Only one module will be responding at a time. One module will be working as primary module while other remains as backup. If the Ethernet link on the primary goes down, the module communicates with the Controller and the Controller performs a failover to use other module.



The redundancy scheme is based on switchover of IP address. This causes a break in communication during failover.

It is not possible to get media redundancy by enabling the second Ethernet port (Ch2).

The redundancy implementation for Master ensures that the requests from applications do not get lost in case of a failure.

The redundancy mechanism for the slave side allows switching the IP address from the old primary to the backup in case of a failure on the primary. This makes it possible for the slaves to resume communication. Masters that were connected to the old primary need to reconnect to the new primary and resend their pending requests.

The backup unit always contains the current configuration of the primary, but it is configured as a backup. When controller gives out signal to the backup module for transforming as primary module, configuration settings of the backup module will be changed as primary configuration.

See [Figure 32](#) for an example of a redundant implementation. Use a switch with a relay connection that is activated when the backup link becomes active. This relay connection can be connected to a Digital Input that is read by the AC 800M. In that way an alarm/event can be raised when a failing link is detected in the network.

It is also recommended to not connect any hub / repeater on the link between the CI867 Ethernet port and the switch. Doing so increases the failover time if a break occurs after the hub / repeater, since break will not be detected by the cable break detection.



Do not connect any hub/repeater on the link between the CI867 Ethernet port and the switch as it increases the failover time if a break occurs after the hub / repeater, since break will not be detected by the cable break detection.



Use only Ethernet port 1 (Ch1) for redundancy as the cable break detection is only implemented on Ethernet port 1 (Ch1).

Examples of faults that will be covered:

- Hot remove of CI867
- Internal fault on CI867
- Cable break between CI867 and first closest switch
- The switch closest to CI Module fails

Online Upgrade

By using Online Upgrade functionality, you can upgrade a redundant CI867 without stopping the process.

To upgrade the CI867 module in a redundant network:

1. Disable the redundancy.
2. Upgrade the backup CI867.
3. Without stopping the process, make the upgraded CI867 as active module.
4. Upgrade the CI867 having old software.
5. When both CI867 modules are upgraded with new software, enable redundancy.



During the online upgrade the process should be put in a steady state. No critical operation should be performed during the upgrade.

The CI867 supports Online Upgrade. The CI867 will be restarted during Online Upgrade. This will be done even if only the Controller firmware is upgraded. During restart, no internal status will be transferred and communications will be disconnected.

Limitations

- Ch1 and Ch2 must be connected on two different sub nets.
- Ch2 does not support cable break detection feature. Do not use it in redundant networks.
- The maximum message size for write is 1968 bits or 123 registers.
- The maximum message size for read is 2000 bits or 125 registers.
- The Modbus TCP CI867 can read/write only 16 bits registers.
- The valid range for DINT datatype used for ModBus TCP communication is between -32768 and +32767.
- The CI867 configured as slave unit, treats DINT as unsigned integer and the range of values is between 0 and 65535.
- If CI867 is configured as master, the master CI867 treats DINT as a signed integer and the range of values is only between -32768 and +32767. Writing DINT values (through ModbusTCP Read/Write) below -32768 and above 32767 shows error -7005.
- When both master and slave CI867 are using DINT datatype, then the negative values written by master CI867 on the slave side, are converted to the equivalent unsigned integer values and shown up in the slave CI867.
 - For example, writing -32767 from the MBTCPWrite block shows as 32769 in slave CI867 and writing -1 from the MBTCPWrite block shows as 65535 in slave CI867.
- For reading, if slave CI867 has values below -32768 and above 32767 then these are shown as equivalent integer values on the master side.
 - For example, value -32769 is read as 32767 and value 32768 is read as -32768 in ModbusTCP Read function block.
- The CI867 slave module supports only INT data type for communicating negative values which ranges from -32768 to +32767. That is, when master CI867 is communicating to the slave CI867, the data type used must be INT if any negative value is used for communication (-32768 to +32767). Else, UINT is used when only positive values are communicated.

Performance

The Master functionality is implemented using Function Blocks like Connect, Read and Write. The execution interval of these function blocks, the number of function blocks, and type of data being communicated (that is, coils or registers), determines the communication throughput.

The peak throughput of MODBUS TCP is dependent on if the CI867 is used as a Master or Slave.

The peak throughput is about 150 requests per second if a single CI Module as master is connected to AC 800M and the transactions are divided over the slaves connected to the CI867. That is, if one slave is connected under CI867, then the peak throughput is 150 requests per second for that slave. If two slaves are connected, then the requests per second for each slave is 75, but the total requests per second for the master remains 150 requests per second.

The performance of master depends on the slave response. The throughput values reduces if the slave used takes more time to respond. The throughput is calculated considering the master requesting data every 50 ms. The throughput figures are calculated with Modbus Client tester-Simulator from Modbus.org as slave. The throughput may change based on the response of the slave connected.

In case of multiple CI Modules, the requests per second per CI Module may be reduced depending on the cycle time of the task and the number of function blocks. The request per second for each CI867 reduces by 15-20 requests for each additional CI867 added on the CEX bus. That is for one CI867, the throughput is approximately 150 requests per second. For two CI867 connected, the throughput for each CI867 is approximately 135 requests per second. For three CI867 connections, the throughput for each CI867 is approximately 120 requests per second.

The peak throughput for CI867 as slave is 20 transactions per second, when one master is connected to CI867. The peak throughput for CI867 as slave is 80 transactions per second if a maximum of 8 masters connected to CI867, with maximum of 20 transactions per second for each master connected.

The slave performance is driven by remote master. The throughput is calculated considering the master requesting data every 50 ms. These throughput figures are calculated with Modbus Server tester-Simulator from Modbus.org as master. The throughput may change based on the master that is connected.

The maximum number of slaves distributed over 12 CI867 units connected per AC 800M is 420 slaves, with maximum of 70 slaves per CI867.

The task time used should be more than 250 ms to achieve high throughput, especially when a large number of function blocks (or slaves) are connected.

The transmission of registers loads the AC 800M less than coils. This is because the coil data needs to be packed and unpacked in the controller.

The typical load on the controller when a single CI867 is used as a Master (cycle time of 250 ms):

- Single slave with 50 register data length is about 6%
- 30 slaves with 50 register data length is about 8%.
- Single slave with data length of 525 coils is 6%
- 30 slaves with data length of 525 coils is 9%.

The typical load on controller when a single CI867 is used as a Slave:

- Single Master communicating 50 byte message is about 4%
- 8 Masters communicating 50 byte message is about 12%
- Single Master communicating a single coil is about 4%
- 8 Masters communicating message length of 525 coils is about 22%



The CI867 Master sends a single request to each connected slave, and each CI867 slave can store up to 60 requests.



The total number of requests that can be send by CI867 Master to all the slaves at the same time is limited to 1120.

Limitations for function blocks

The Master/Slave functionality has the following limitations for function blocks:

- The total number of function blocks for all slaves put together must not exceed 1120.
- The total number of function blocks per slave must not exceed 60.

Hardware

The maximum number of CI867 connected per controller is 12 non-redundant units or 6 redundant units.

Troubleshooting

The CI867 device status, watchdog supervision, controller log, and CI log are reported to the AC 800M controller for displaying in the Control Builder.

Section 11 AF 100

Introduction

Advant Fieldbus 100 (AF 100) is a high performance fieldbus, which is used for communication between various Advant products.

AC 800M can use AF 100 for communication with Advant Controllers.

It is possible to reach up to 80 stations within a total physical distance of up to 13300 meters (43300 feet).

Advant Fieldbus supports three transmission media:

- Twisted pair (Twp)
- Coaxial (RG59 and RG11)
- Optical.

A bus can be built up with all the three media, where a part of one kind of media is denominated segment.

The following rules apply to the segments:

- To each twisted pair segment, 32 stations can be connected and the maximum segment length is 750 meters (2500 feet)
- The coaxial segment can be:
 - 300 meters (1000 feet) with cable RG59 or
 - 700 meters (2300 feet) with cable RG11.

- The optical media is only used in point-to-point communication, and allows the total length of a bus segment to be up to 1700 meters (5500 feet).
- By using back-to-back coupled optical segments, it is possible to reach a physical length of 13300 meters (43300 feet).

An Advant Fieldbus 100 may be installed with one or two physical bus lines (single or redundant media). Two bus lines are chosen when increased availability is required. The redundant bus line does not enhance the bus bandwidth when both bus cables are operating. It is a pure backup function.

For more details about AF 100, refer to *Advant Fieldbus 100 User Manual (3BSE000506*)*.

Services Provided

The communication interface - CI869 - is a CEX Module attached to the AC 800M controller that provides connectivity to other AC 800M, AC 160 or connectivity server over AF 100. An AC 800M controller with the communication interface CI869 behaves as an AF 100 station, receiving data from other AF 100 stations/devices.'

Two CI869s can operate as a pair, where one is Primary and the other is Backup in AC 800M controller configuration.

Design

The controllers (Advant Controller 70, Advant Controller 100 Series, Advant Controller 400 Series, and AC 800M) and other AF 100 Stations can be connected as stations on Advant Fieldbus 100. The AF 100 Stations comprise, for example, AC 100 OPC Server and S800 I/O Station. The AC 100 OPC Server together with 800xA for AC 100 provides a wide range of functionalities when connected to Advant Fieldbus 100. S800 I/O Station is a remote I/O station.



CI869 does not support communication with S800 I/O Station.

The Advant Fieldbus 100 provides communication between Advant Controllers and AF 100 Stations.

The Advant Fieldbus 100 supports three different kinds of communication:

- DSP (Data Set Peripheral) for peer-to-peer communication between controllers.
- IO Communication for communication between a controller and S800 I/O. This includes the communication for process data, configuration, status and time tagged events.
- SOE communication for time tagged events from Advant Controller 100 to Advant Controller 400.

An Advant Fieldbus 100 network is installed with single or redundant cables. In configurations with redundant cables, the data is always transmitted on both cables, but the receiver selects the cable to receive the data from. This is done automatically by the hardware.

Advant Fieldbus 100 is a high performance fieldbus which can be used to connect up to 80 Advant Controllers and/or AF 100 Stations (see [Figure 33](#)).

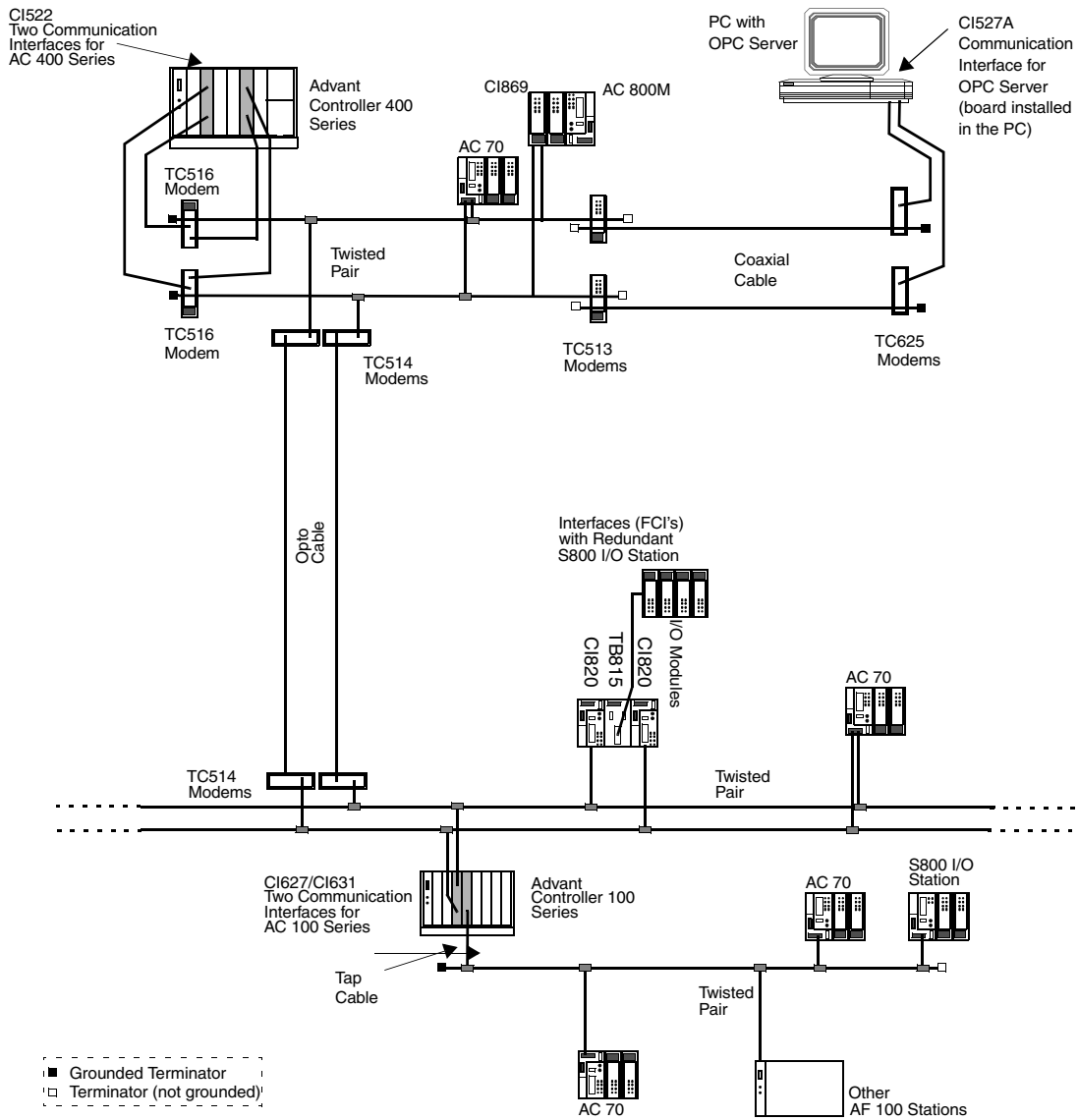


Figure 33. Network Overview of AF 100 with one Redundant media bus and one Single media bus

The allowed station numbers for Advant Controller 400 Series and AC 100 OPC Server are 1 to 80. For other controllers and AF 100 Stations, station numbers 1 to 79 are allowed.

DataSet Peripheral Communication

DataSet Peripheral (DSP) is a data set function for communication on Advant Fieldbus 100. In the controller, the DataSet Peripheral is represented with a data base element, For transmission of data between different nodes, DSPs are used.

The maximum number of configured DSPs is different for each type of controller:

- Advant Controller 400 Series can handle about 4000 DSPs
- AC 800M controller can handle 4000 DSPs
- Advant Controller 110 can handle 200 DSPs
- Advant Controller 160 can handle 400 DSPs
- Advant Controller 70 can handle 50 DSPs.

The Advant Fieldbus 100 can handle about 4000 CDP's. Each DSP uses one CDP and 50 CDP's are reserved for each S800 I/O station configured on the bus. This means that the actual maximum number of DSPs is reduced with 50 per configured S800 I/O station.

Defining DataSet Peripherals in AC 800M

The configuration of the CI869 hardware unit in the hardware tree in Control Builder defines the AF 100 bus with AC 800M controller.

The following hardware units under CI869 are used to configure AF 100:

- AF 100 Station with DSPs
- DSP Group
- My DSP Group
- My AF 100 Station Status
- AF 100 Station

The hardware units *AF 100 Station with DSPs*, *DSP Group*, and *My DSP Group*, holds the DSPs.

AF 100 Station with DSPs

The *AF 100 Station with DSPs* is an organization unit for up to 50 *DSP Receive* subunits. For each *AF 100 Station with DSPs*, a Station Status CDP is configured on the CI869 for supervision of a remote station on the bus. Its content is presented as *Extended Status* in the hardware editor of *AF 100 Station with DSPs*.

DSP Group

DSP Group is a hardware unit for organizing up to 50 DSP subunits (both receiving and sending), with the same Major DSP ID.

DSP Group is similar to the *AF 100 Station with DSPs*, but *DSP Group* does not cause CI869 to configure any Station Status CDP.

My DSP Group

My DSP Group is a hardware unit for organizing up to 50 Sending DSP subunits, with the same Major DSP ID, sent from this station (also referred to as *My Station*).

The receiving DSP subunits should not be added under *My DSP Group*.

DSP Send and DSP Receive

The two hardware units - DSP Send and DSP Receive - represent the DSPs that the CI869 sends and receives respectively.

The Minor DSP ID together with the Major DSP ID gives the DSP ID.

The Major DSP ID, which is determined by the position number of the hardware unit above the DSP, can be from 1 to 80.

The position number of the DSP [1...50] gives the Minor DSP ID. The ID can be modified by changing the position of the DSP (moving it up or down in the HW tree).

The DSPs are categorized as either DSP Send (the source communication unit) or as DSP Receive (the sink communication unit), and have a defined cycle time.

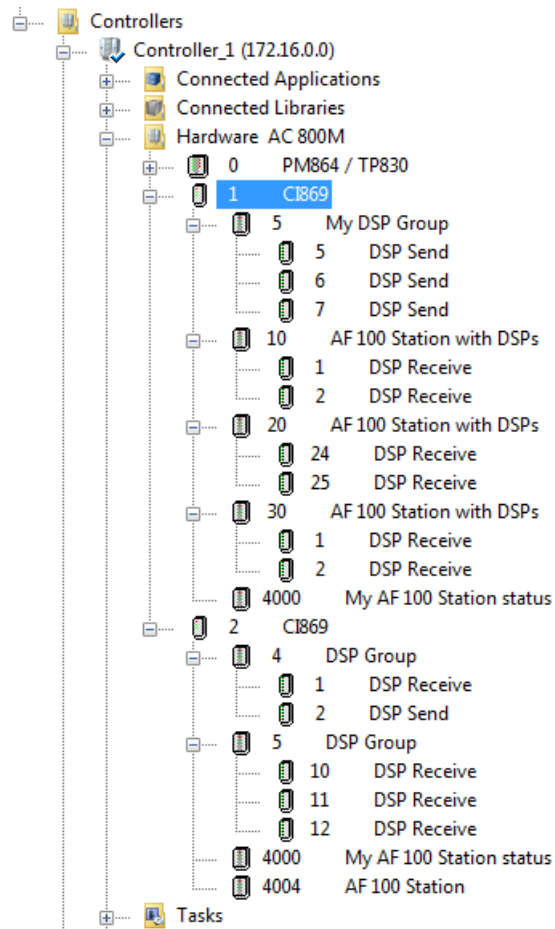


Figure 34. Example of hardware units inserted under CI869



The CI869 supervises AF 100 stations using the objects *My AF100 Station status* and *AF100 Station*.

Bus Master Function

All communication on AF 100 is driven by an active bus master.

The bus master function controls all the transmission on the Advant Fieldbus 100, while reception of data is controlled locally on the individual communication interface.

In an Advant Fieldbus 100 network with one or more communication interfaces, one of these functions as bus master while the other communication interfaces are active with supervising that the bus master operates correctly.



CI869 is not capable of acting as bus master. Another station acting as bus master is needed.

Products with bus master functionality are:

- Communication interfaces for the Advant Controllers AC 400 Series (CI520, CI522)
- Communication interfaces for the AC 100 Series (CI626, CI627, CI630, CI631)
- Communication interfaces for AC 100 Connect (CI526, CI527)
- Stand alone bus administrators CI626A, CI627A



When redundant CI522s and CI820s are used on the bus, then one of the following rules must be followed:

- The redundant CI522 pair must be assigned a station number lower than five (5).
- Two master defined communication interfaces must be assigned station numbers lower than five (5).

Network Configuration

When a new Advant Controller and/or AF 100 Station is connected to the Advant Fieldbus 100, it is automatically recognized and the new station participates in the communication as it is configured.

The starting, stopping, and restarting of stations can also be done without disturbing the traffic on the Advant Fieldbus 100, if some precautions are taken.

The Bus Administrator must know the configuration of all CDPs in the network in order to perform the bus master function.

The station which is configured as the owner of a CDP informs the bus master about how to configure it.

The Bus Administrator maintain a scan table containing information about all CDPs and when they must be transmitted on the Advant Fieldbus 100. When the configuration changes, the scan table is regenerated.

The distribution of information is always performed by the communication interface configured as the owner of the DSP. This is typically the sender of the DSP. This is done automatically whenever a sending CDP is defined, deleted, or updated.

Online Upgrade

CI869 supports Online Upgrade. This allows the upgrade of the CI869 firmware when the application is running in the controller (online mode).



During the online upgrade, the process should be put in a steady state. No critical operation should be performed during the upgrade.



When the Online Upgrade is performed with redundant CI869s, the communication through CI869 is interrupted only during the switchover of the redundant processor modules connected to the two CI869s.

When the Online Upgrade is performed with single CI869, the communication through CI869 is interrupted from the time the processor module switchover starts till the CI869 is upgraded with the new firmware.

For more information on Online Upgrade, refer to *System 800xA Control, AC 800M, Configuration (3BSE035980*)* manual.

Hardware

The maximum number of CI869 connected per AC 800M controller is 12 non-redundant units or 6 redundant units.

Troubleshooting

The CI869 device status, watchdog supervision, controller log, and CI log are reported to the AC 800M controller for displaying in the Control Builder.



Double-click any hardware unit to see the corresponding status message in the Control Builder. These messages are useful for troubleshooting.



The crash logs are useful for troubleshooting and contains crucial information for analyzing malfunctions.

Section 12 MOD5-to-MOD5

This section provides the information specific to the MOD5-to-MOD5 communication protocol and the MOD5 communication interface, CI872.

Introduction

MOD5-to-MOD5 protocol is a proprietary communication protocol that is used to enable peer-to-peer communications between MOD5 process control systems and AC 800M controllers. It provides a redundant serial communication over fiber-optic links, using a request and response mechanism.

Services Provided

The protocol consists of request and response messages that are exchanged each second.

The requests sent to other connected systems are determined by the control application. The response sent at each second is determined by the requests received at the previous second from other connected systems. The application programmer accesses the protocol functions through standard function blocks.

The implementation uses the functions blocks MTMConnect, MTMReadCyc, MTMDefCyc, and MTMDefERCyc to translate the request and to answer the MOD5 commands.

Design

The MOD5 communication interface module, CI872, helps the communication between MOD5 controllers and the AC 800M controller.

The CI872 modules are supported by the AC 800M controllers.

The CI872 has three pairs of full duplex fiber optic ports for connection to MOD5 controllers. One CI872 is intended to communicate with up to three left MOD5 controllers or three right MOD5 controllers. For redundancy, it is necessary to use two CI872 modules.

In order to handle 12 redundant MOD5 controllers, a total of four pairs of CI872 should be deployed.

Table 14. Design Specifications for MOD5 Implementation with AC 800M

Speed	500 kbps
Maximum remote MOD5 controllers per CI872	Three (or three pairs in case redundant modules).
Connection Topology	Refer Figure 35 .
Recommended Connection	Refer Table 15 , Figure 35 , Figure 36 , Figure 37 .
Limitations for function blocks per port of CI872	<p>The combined maximum number of MTMReadCyc, MTMDefERCyc, and MTMDefCyc function blocks that can communicate per port is five. These can be spread over multiple applications.</p> <p>In the above limit of five function blocks, the maximum number of MTMDefCyc and MTMDefERCyc function blocks that can communicate per port is three, with a maximum of 100 variables per MTMDefCyc or MTMDefERCyc function block.</p>

Connection Examples

Figure 35 shows an example of MOD5 to AC 800M connection.

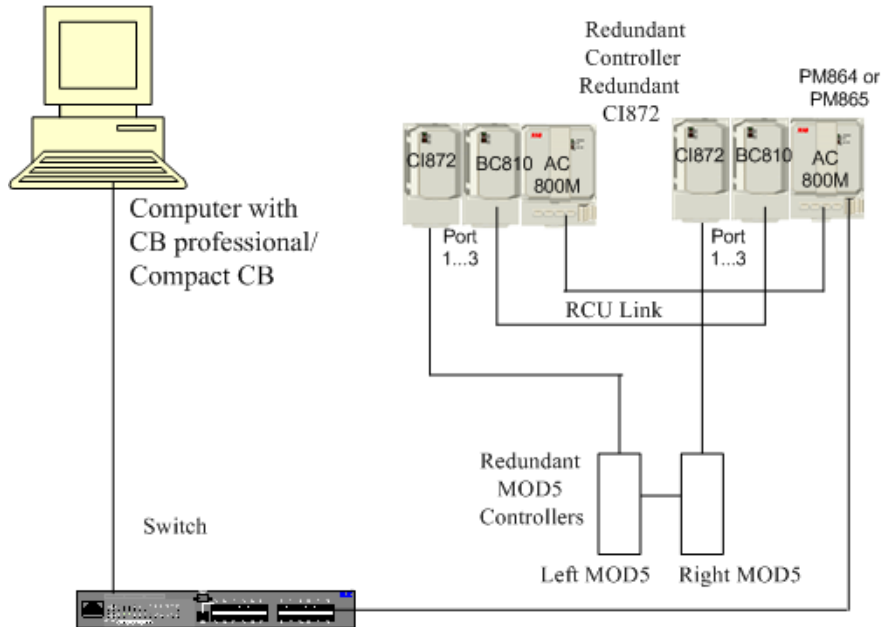


Figure 35. Example of a MOD5 to AC 800M Connection

Configuration Considerations

Consider the following while configuring CI872:

- The supported configuration is maximum four redundant pairs of CI872 enabling communication to 12 redundant MOD5 controllers.
- CI872 can be used in a non-redundant mode but the number of CEX-bus modules connected to the controller is limited to four.
- CEX-bus segment A (upper) communicates to all left MOD5, and segment B (lower) communicates to all right MOD5 controllers. A CI872 configured as a single module is located last in CEX segment A. The left/right communication to MOD5 shall be configured as per [Table 15](#).
- BC810 is required to ensure full controller redundancy and Online upgrade. See [Figure 36](#) and [Figure 37](#).
- If BC810 is not used, connect the CEX-bus cable to ensure controller redundancy. See [Figure 38](#).

Table 15. Recommended Positions of CI872 in Different Hardware Configurations

Possible Hardware Configuration	Recommended CI872 Positions ⁽¹⁾
AC 800M PA non-redundant, CI872 non-redundant	Positions 1, 2, 3, 4
AC 800M PA non-redundant, CI872 redundant	Positions 1(2), 3(4), 5 (6), 7(8)
AC 800M PA redundant, CI872 non-redundant	Positions 1, 2, 3, 4
AC 800M PA redundant, CI872 redundant	Positions 1(7), 2(8), 3(9), 4(10)
AC 800M HI, CI872 non-redundant	Positions 2, 3, 4, 5
AC 800M HI, CI872 redundant	Positions 2(8), 3(9), 4(10), 5(11)

Positions within parenthesis indicate the positions in lower segment when CIs are connected to Redundant Controller. If the both the Primary and Backup CI module are connected consecutively to the same side of Controller, the module connected at lower CEX bus position is LEFT system and the one at higher CEX bus position is RIGHT system.

(1) These recommended positions assume that only CI872 is used with the AC 800M controller.



Figure 36. Maximum Configuration Using PA Controller

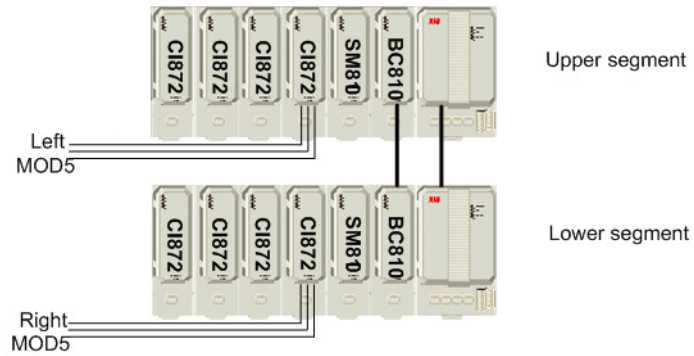


Figure 37. Maximum Configuration Using HI Controller

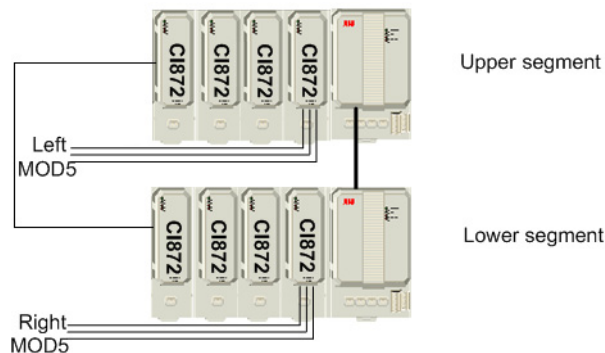


Figure 38. Redundant Connection Using CEX-bus Cable

Module Redundancy

CI872 supports module redundancy. The backup CI872 takes over as the primary module in case of a hardware failure in the primary CI872.

The redundant system ensures that the requests from IEC 61131-3 standard applications are not lost in case the primary CI872 fails.



During failover, there is a loss of data for maximum 3 to 4 cycles (in case of removal of a module) or 6 to 8 cycles (in case of removal of a fibre optic cable in the module). When cyclicity of the function blocks is one second, this results in a loss of data for 3 to 4 seconds or 8 to 10 seconds in the IEC 61131-3 standard application.

See [Figure 35](#) on page 123 for an example of a redundant implementation.

Any of the following situations can occur while receiving data from both the primary and the backup:

1. A hardware failure can cause defects in the operations of primary CI872 module due to one of the following reasons:
 - a. The primary CI872 has been hot removed.
 - b. A fault detected in the primary CI872.

- c. A break in the cable of the primary CI872 for any of the ports.
2. A communication failure can occur due to any of the following errors in the data from the primary:
 - a. There is a framing error.
 - b. There is checksum or parity error.
 - c. One request and response cycle exceeds the threshold of one second.

In case of a hardware failure, a switchover happens from the primary CI872 module to the backup CI872 module. In case of a communication failure, a switchover does not happen.

The time taken by the firmware to complete the failover is dependent on the Detection time, HWStatus update cycle time, and the Reaction time.

The maximum failover time is:

Detection time + Reaction time + HWStatus Update cycle time

where:

Detection time: Time taken by the controller to detect the failure of the primary module.

Reaction time: Time taken by the controller to complete a switch over from the time a failure was detected.

HWStatus update cycle time: HWStatus parameter is found on the hardware tree unit for the PM. This value is provided by the user and the minimum value is 500 ms.

Double-click the AC 800M controller in the hardware tree to open the configuration parameters window. To modify HWStatus parameter, select the parameter in the configuration parameters window, and enter the required values. See [Figure 39](#).

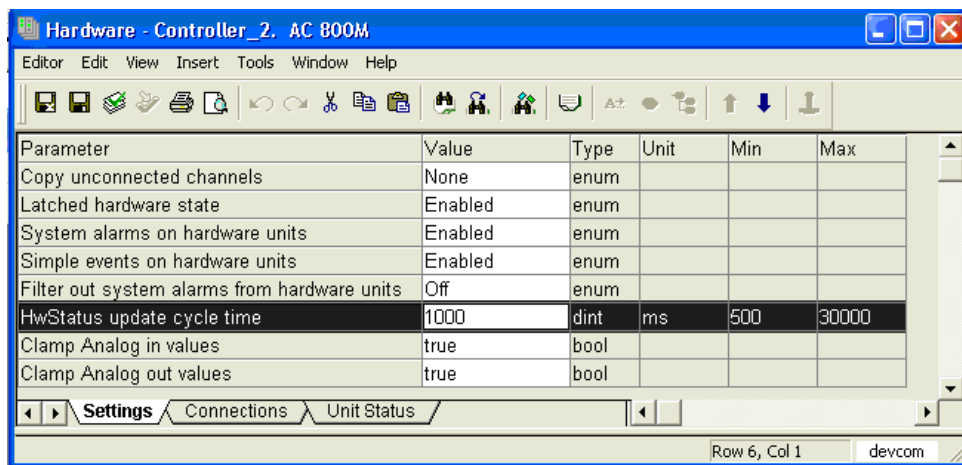


Figure 39. Configuration Parameters Window

Online Upgrade

CI872 supports Online Upgrade. This upgrade functionality allows the user to upgrade the CI872 firmware when the application is running in the controller (online mode).



During an Online Upgrade, the communication between the CI872 module and the connected MOD5 controller is interrupted.

For more information on Online Upgrade, refer to *System 800xA Control, AC 800M, Configuration (3BSE035980*)* manual.

Troubleshooting



Double-click any hardware unit to see the corresponding status message in the Control Builder. These messages are useful for troubleshooting.



The crash logs are useful for troubleshooting and contains crucial information for analyzing malfunctions.

Section 13 EtherNet/IP and DeviceNet

Introduction

EtherNet/IP is an application layer protocol built on the standard TCP/IP protocol suite used to communicate high level industrial devices. DeviceNet is also application layer protocol built on the standard CAN used to communicate low-level industrial devices.

Both are based on CIP (Common Industrial Protocol) and hence share all the common aspects of CIP.

DeviceNet devices connect to EtherNet/IP devices through the linking device, LD800 DN.

For more information, refer to the *AC 800M, Ethernet/IP DeviceNet, Configuration (9ARD000014*)* manual.

Services Provided

The CI873 EtherNet/IP-DeviceNet hardware Library, CI873EthernetIPHWLib, integrated with AC 800M consists of the communication interface (CI873, with one Ethernet port) and other hardware types to be used when configuring EtherNet/IP and DeviceNet.

The services provided by EtherNet/IP-DeviceNet implementation are:

- Configuring CI873 as EtherNet/IP scanner.
- I/O communication with DeviceNet devices using Class 1 connection to LD800 DN.
- I/O communication with EtherNet/IP slave devices through Class 1 connection.
- Status supervision of devices. CI873 originate a Class 1 connection to Linking device (LD 800DN) for DeviceNet device status,
- Hot swap of CI873, LD800 DN, DeviceNet, and EtherNet/IP devices.

- System command to change the Run/Idle state of LD800 DN.
- Logging of CI873 messages.
- CI873 Scanner diagnostics.
- LD800 DN Scanner diagnostics.
- CI873 Firmware Upgrade.
- Online upgrade for CI873 Firmware.
- CI873 module redundancy.
- CI873 supports logical segment Class 1 connection for reading and writing data to EtherNet/IP devices (except Allen Bradley PLC).
- CI873 originates Class 1 for tag reading and Class 3 for tag writing to Allen Bradley Logix 5000 series PLC.
- Device parameter configuration.

Design

The CI873EthernetIPHWLib Hardware Library is a part of the EtherNet/IP-DeviceNet integration in the 800xA System.

EtherNet/IP-DeviceNet is configured using the Control Builder. The configuration includes the planning of the hardware units in the hardware tree, specific configuration for the EtherNet/IP-DeviceNet communication interface CI873 and the DeviceNet devices. The device specific configuration data is described within the DeviceNet and EtherNet/IP EDS file provided by the device manufacturer. To configure the DeviceNet and EtherNet/IP device within the Control Builder, the EDS file must be imported into a hardware library and inserted to the project using the Device Import Wizard.

Design Example

Figure 40 shows an example of EtherNet/IP-DeviceNet installation with AC 800M controller.

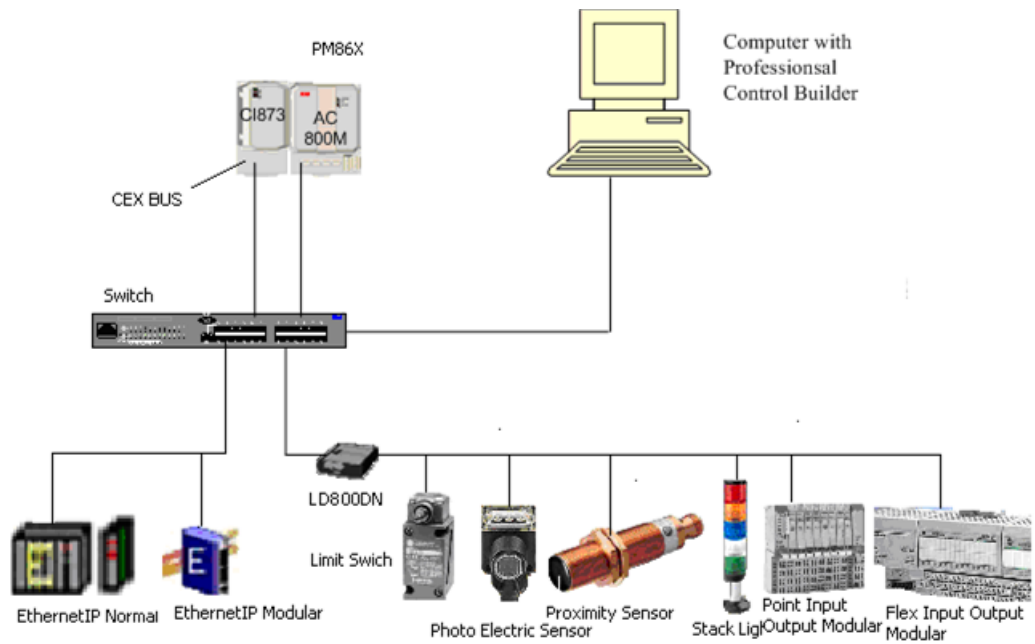


Figure 40. Connection of EtherNet/IP and DeviceNet devices with AC 800M

Redundancy

CI873 supports module redundancy. The backup CI873 takes over as the primary module in case of a hardware failure in the primary CI873.

The redundant system ensures that the requests from IEC 61131-3 standard applications are not lost in case the primary CI873 fails.

A hardware failure can cause defects in the operations of primary CI873 module due to one of the following reasons:

- a. The primary CI873 has been hot removed.
- b. A fault is detected in the primary CI873.
- c. A break occurs in the cable of the primary CI873 for any of the ports.

In case of such a hardware failure, a switchover happens from the primary CI872 module to the backup CI872 module.

The time taken by the firmware to complete the failover is dependent on the Detection time, HWStatus update cycle time, and the Reaction time.

The maximum failover time is:

Detection time + Reaction time + HWStatus Update cycle time

where:

Detection time: Time taken by the controller to detect the failure of the primary module.

Reaction time: Time taken by the controller to complete a switch over from the time a failure was detected.

HWStatus update cycle time: HWStatus parameter is found on the hardware tree unit for the PM. This value is provided by the user and the minimum value is 500 ms.

Double-click the AC 800M controller in the hardware tree to open the configuration parameters window. To modify HWStatus parameter, select the parameter in the configuration parameters window, and enter the required values. See [Figure 41](#).

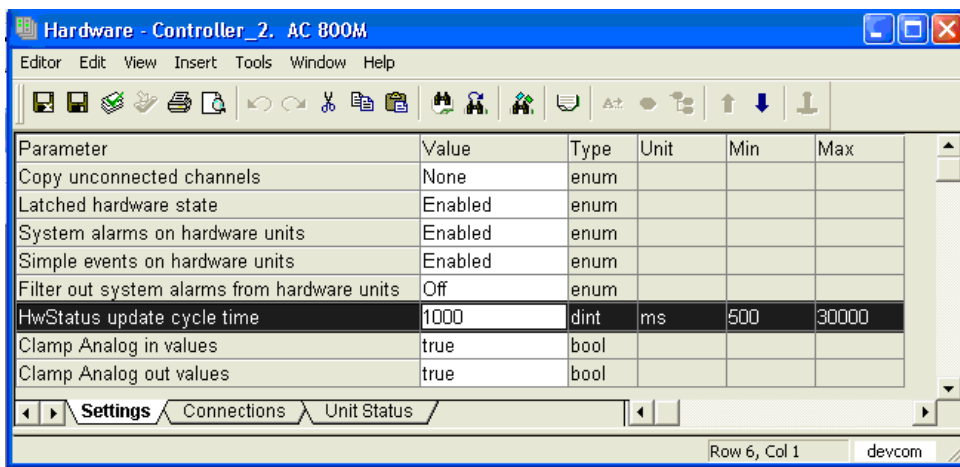


Figure 41. Configuration Parameters Window

Online Upgrade

CI873 supports Online Upgrade. This upgrade functionality allows the user to upgrade the CI873 firmware when the application is running in the controller (online mode).



During an Online Upgrade, the communication between the CI873 module and the connected AC 800M controller is interrupted.

For more information on Online Upgrade, refer to *System 800xA Control, AC 800M, Configuration (3BSE035980*)* manual.

Limitations

The limitations, with respect to the various devices in general are:

- The number of I/O modules that can be connected under EtherNet/IP or DeviceNet device adapter type device is 63.
- The number of configuration parameters supported per EtherNet/IP or DeviceNet device is 1000.
- The CI873 supports Listen only connection with EtherNet/IP device, provided there is already Exclusive owner connection in the device. The CI873 does not support Redundant owner connections for EtherNet/IP devices.
- The Read only parameter and monitoring parameters in EDS file are not supported in this release.
- The tag based Class 1 information should be there in EDS file for communication with Allen Bradley PLC where Class 3 tag can be added along with Class 1 connection.
- The total number of Input and Output bytes along with channel status bytes should not exceed more than 80Kb per CI873.
- The Configuration assembly size of 200 is supported per EtherNet/IP or DeviceNet device.
- The CI873 supports 20 CIP connections (including Class 1 and Class 3) per EtherNet/IP device. CI873 supports total of 128 connections.
- The CI873 only supports devices which uses EtherNet/IP encapsulation of CIP.
- It does not support PCCC, Modbus encapsulation.
- CI873 supports CH1 Ethernet interface with a speed of 100 Mbps. CH2 is not supported.
- A maximum of 6 non redundant CI873 can be connected to each AC 800M controller.

EtherNet/IP Limitations

The limitations, with respect to the EtherNet/IP device involved are:

- EtherNet/IP supports three Class 1 connection and three Class 3 tag per Allen Bradley Control Logix PLC. The CI873 supports three Class 3 tags with 100ms cycle time.
- The data transfer, using the Class 3 connection, will be slower than the Class 1 connection.
- The Class 3 connection is not supported for any EtherNet/IP devices except Allen Bradley Control Logix PLC. The CI873 uses tag based Class 3 to write data to it.
- The maximum number of bytes support for Class 1 read tag is 496 and for Class 3 write tag is 432.
- Only 1000 bytes per Class 1 connection is supported, for example O->T: 500 and T->O : 500.

LD 800DN Limitations

The limitations, with respect to the LD 800DN linking device (for DeviceNet) are:

- The maximum number of input bytes supported by LD 800DN is 496 bytes. If the total number of input bytes of all DeviceNet slaves configured under the linking device exceeds 496 bytes, download is stopped.
- The maximum number of output bytes supported by LD 800DN is 492 bytes. If the total number of output bytes of all DeviceNet slaves configured under the linking device exceeds 500 bytes, download is stopped.
- Multiple CI873 cannot listen to same LD 800DN data.
- The maximum number of DeviceNet connections per device is restricted to 5..

Performance

The time taken from changing an input channel to the time of setting an output channel is 240 ms.

Section 14 IEC 61850

Introduction

The CI868 is a communication interface between IEC 61850 devices in the substation network and the IEC 61131-3 Application in an AC 800M Control System. The communication between the CI868 and the IEC61850 is done through Generic Object Oriented Substation Event (GOOSE Send and Receive) or MMS Protocol (Receive and Control Commands).

Services Provided

The services provided by CI868 IEC 61850 Communication Interface.

GOOSE Protocol:

GOOSE protocol is used for sending signals from CI868 IEDs to other IEDs, and also receive status from other IEDs. For GOOSE protocol, the GOOSE sending LNs are listed under MyIED and the Receive Blocks are listed under other IEDs.

MMS Client Protocol:

CI868 module supports MMS client functionality on IEC 61850 network. The Logical Nodes CSWI and XCBR is used for sending MMS Control Commands from CI868 IEDs to other IEDs. Also receive MMS Signals from all Logical Nodes through RCB datasets from other IEDs.

Generate CI868 CID / ICD File:

Generate CI868 CID / ICD file from configured hardware tree under CI868 from Control Builder. The CID / ICD file can then be used further for scd-file engineering in CCT600 or 3rd party tools to generate fully configured scd-file.

Configure CI868 as an IED and download the CCF file to the CI868 module. It is also used to connect IEC 61131-3 variables to the IEC 61850 data.

Design

Introduction

The CI868 Communication Interface Module integrates IEC 61850 into the AC 800M system.

For more information on Engineering and Configuration on IEC 61850, refer to *AC 800M IEC 61850 Engineering and Configuration (9ARD171385*)* Manual.

The CI868 is a single channel Ethernet Module with ports:

- Ch1: compliant to IEEE 802.3u standard for the 10/100 Base-T by the Media Independent Interface.

Table 16. Key features of IEC 61850 implementation for AC 800M

Network Topology	The IEC 61850 specifies the usage of ring network.
Speed	Ch1: 10/100 Mbps, Full duplex (Auto negotiation)

Design Examples

Figure 42 shows Example of IEC 61850 network implementation.

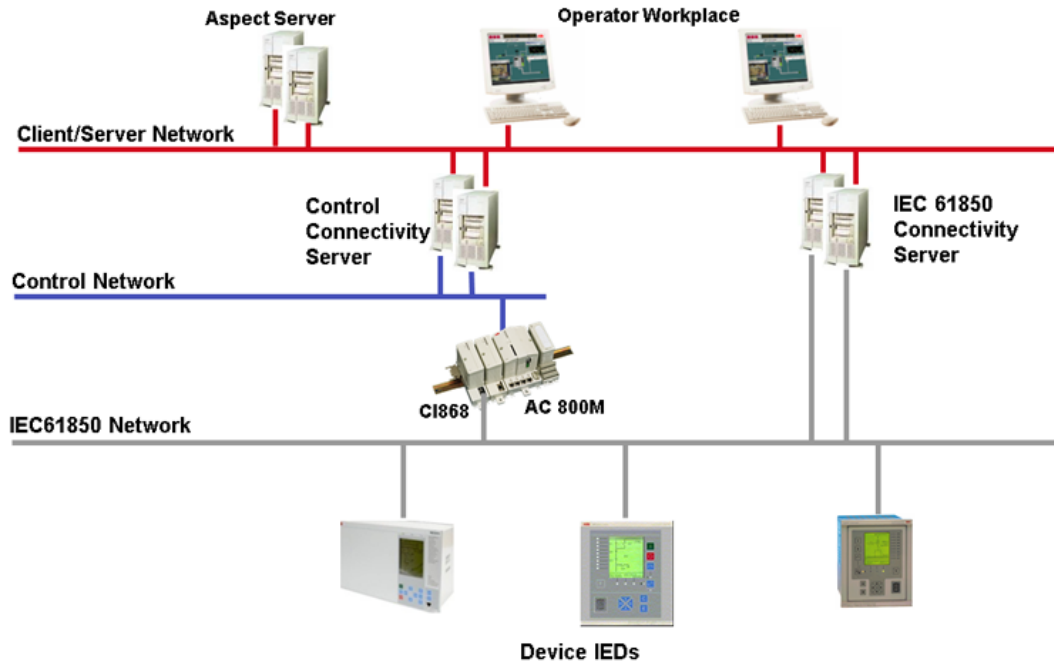


Figure 42. Example of IEC 61850 Network implementation

IEC 61850 Hardware Objects

Each CI868 is represented as IED with one access point. The access point is the position of CI868 in the Control Builder tree structure and is represented in the hardware object MyIED as shown in Figure 43

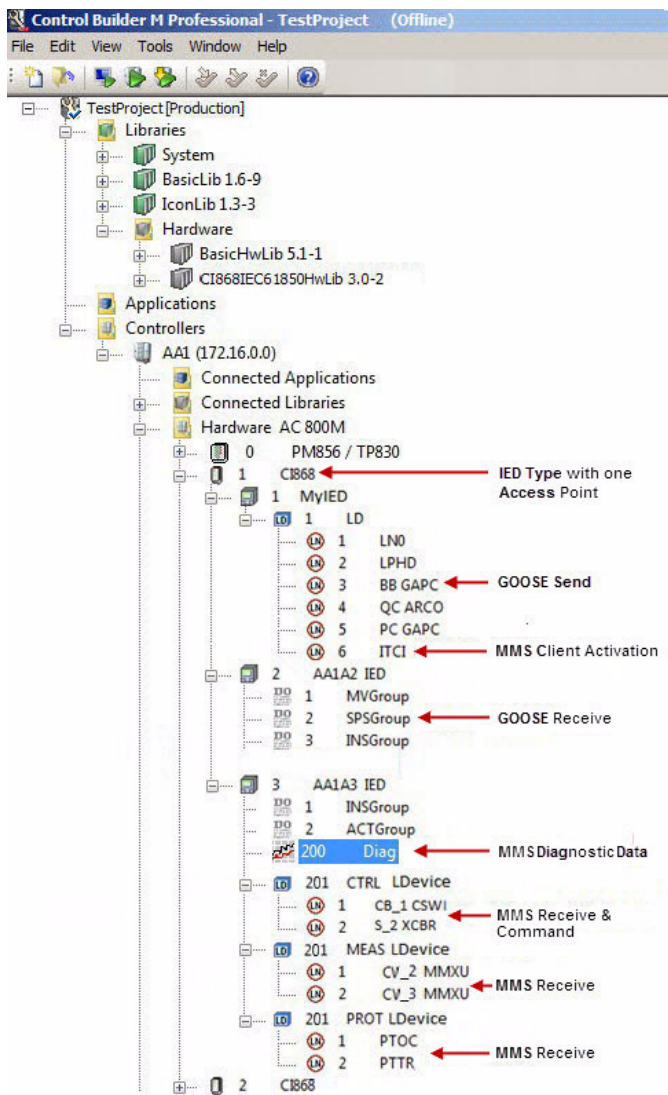


Figure 43. Communication Interface CI868 IED Tree view

Connection Methods

- Use twisted pair 100BASE-T Ethernet cable with RJ45 connector.
- In case of Redundant CI868 implemented through IEC 61131-3 application, connect the two CI modules to different switches for better availability.
- The installation need to be compliant with Category 5 specification according to IEEE802.3.
- Use network hardware as described in ABB document *Third Party HW products verified for Industrial IT system 800xA (3BSE046579*)*.
- The IEC 61850 network should be an independent network for security reasons.
- The IEC 61850 network should not be treated in isolation. The network design and IP address allocation need to be done within the overall framework of system 800xA. Refer *Automation System Network: Design and Configuration (3BSE03446*)*.

Redundancy

Module Redundancy by IEC 61131-3 Application Logic

Application redundancy is achieved by using two CI868 modules (each having one Access point) with AC 800M, connected in the same sub network and cable redundancy using external switches, which supports rapid spanning tree protocol.

- User creates a Feeder Block Diagram with the sending logical nodes in IET tool and connects the FBD to one Access point, copies the same FBD and connects to the second access point which is in the same sub network as the first one.

User configures two CI868 modules in Control Builder each containing same set of sending Logical Nodes under MyIED.
- User exports the configuration to communication configuration tool and creates same datasets under LNO for the two access points.

- User exports the configuration of both CI868 modules from Control Builder to Communication Configuration Tool and assign them to the same Bus and further creates same datasets under LN0 for the both CI86 modules.
- User creates the same input under LN0 of the access points by dragging the subscribed IED to the two access points respectively.
- Wizard creates the tree structure with two CI868 in the Control builder.
- User creates an IEC 61131-3 application to receive data from one access point and switch to another access point data in case of channel error or CI868 failure.
- Application writes the data to both the access points but read from only one of the two access points.

Online Upgrade

For CI868 Firmware Upgrade scenarios applicable during Control Builder project migration from earlier versions to Feature Pack version, refer to *AC 800M IEC 61850 Engineering and Configuration (9ARD171385*)* Manual.

CI868 Performance

Capacity of IEC 61850 solution using CI868

This section describes the configuration and performance limits for CI868 module.

For detailed information on CI868 Performance Data, refer to *System 800xA System Guide Technical Data and Configuration (3BSE041434*)* Manual.

Table 17. CI868 Configuration Limit for AC 800M

Description	Limit	Remarks
Maximum Number of CI868 modules connected per Non-Redundant AC800M controller	12	CI868 Non-redundant units
Maximum Number of CI868 modules connected per Redundant AC800M controller	6	CI868 Application-redundant units

Following are the recommendations to be followed while engineering the scd file:

Table 18. CI868 Performance Data

Description	Limit	Remarks
CI868 Performance for GOOSE Data Per CI868 Module:		
Maximum Number of IEDs connected	80	
Max number of Static Data Objects (signals) configured	800	Scd-file should not be configured for more than: <ul style="list-style-type: none"> • Max. 800 Data Objects (signals) from 80 IEDs • 50 IEDs for 800xA SV5.0 • 80 IEDs for 800xA SV5.1 Analog, Integer or Boolean type of Data objects.

Table 18. CI868 Performance Data

Description	Limit	Remarks
Maximum Number of changing Data Objects Received	160 / sec	Analog, Integer or Boolean type of Data Objects. This is subject to scd-file configuration.
Maximum Number of changing Data Objects Sent	10 / sec	Analog, Integer or Boolean type of Data objects. This is subject to scd-file configuration.
CI868 Performance for MMS Client Data Per CI868 Module:		
Maximum Number of IEDs connected	20	
Max number of Static Data Objects (signals) configured	1000	Scd-file should not be configured for more than <ul style="list-style-type: none"> Max. 1000 data objects (signals) from 20 IEDs 20 IEDs for 800xA SV5.1 FP4 Analog, Integer or Boolean type of Data objects. When creating scd file, high CI868 load for MMS signals can be avoided by grouping the frequently changing signals in the same dataset. For example: Measurement signals must be grouped in one dataset, Status signals must be grouped in another dataset.
Maximum Number of changing Data Objects Received	80 / sec	Analog, Integer or Boolean type of Data Objects
Maximum Number of MMS Control Command send.	1 / sec	MMS Control Commands sent from CI868 via CSWI & XCBR LNs.
CI868 Performance for MMS Client and GOOSE protocol combined usage:		
For GOOSE and MMS Configuration Limits, refer CI868 Performance for GOOSE and MMS Client Data per CI868 Module.		

Table 18. CI868 Performance Data

Description	Limit	Remarks
Maximum Number of IEDs connected	20	
Maximum Number of changing Data Objects Received	60 / sec	Analog, Integer or Boolean type of Data Objects
Maximum Number of MMS Control Command send.	1 / sec	MMS Control Commands via CSWI & XCBR LNs.

Diagnostic Information for LN0 and MMS Diag Hardware Object on CI868

Diagnostic information is available in LN0 and MMS Diag hardware object. To view diagnostics, traverse to the LN0 and MMS Diag hardware object to open the Hardware editor and click the connection tab to view the diagnostic information.

The Diagnostic information contains details about RCBs and GOOSE Signals provided by IEC 61850 stack and is explained in [Table 19](#).

Table 19. LN0 and Diag Hardware editor diagnostics information

Diagnostics Information	Description
LN0 Diag info	
GOOSE values received	Number of GOOSE data values received from other IEDs.
GOOSE Values Sent	Number of GOOSE data values sent to other IEDs.
GOOSE Values Received Per second	Number of GOOSE values received from other IEDs per second.
GOOSE Values Sent Per Second	Number of GOOSE values sent to other IEDs per second.
CPU Load	CPU load of the CI module in percentage updated every minute.

Table 19. LN0 and Diag Hardware editor diagnostics information

Diagnostics Information	Description
Sender Cycles per Second	Number of times the sender task has traversed the send list containing the Data object, to be sent out on change.
MMS Values Received	Number of MMS data values received from other IEDs.
MMS Commands Sent	Number of MMS data values sent to other IEDs.
MMS Values Received Per Second	Number of MMS values received from other IEDs per second.
Report Control Blocks Received Per Second	Number of RCBs received per second.
Number of IEDs Configured for Reporting	Number of IEDs configured for MMS Reporting.
Number of IEDs Connected for Reporting	Number of IEDs connected for MMS Reporting.
Number of Buffered Report Control Blocks Configured	Number of Buffered RCBs configured.
Number of Buffered Report Control Blocks Enabled	Number of Buffered RCBs enabled.
Number of Unbuffered Report Control Blocks Configured	Number of Unbuffered RCBs configured.
Number of Unbuffered Report Control Blocks Enabled	Number of Unbuffered RCBs enabled.
MMS Diag info	
Number of Report Control Blocks Configured	Total number of RCBs configured.
Number of Report Control Blocks Enabled	Total number of RCBs enabled.

Troubleshooting

The CI868 device status, watchdog supervision, controller log, and CI log are reported to the AC 800M controller for displaying in the Control Builder.

Section 15 FOUNDATION Fieldbus HSE

Introduction

FOUNDATION Fieldbus (FF) is a bi-directional protocol used for control system communication and meets ISA SP50 requirements. It is a fieldbus used for communication with distributed I/O units and fulfills the regulations and safety demands in high risk (explosive) environments, and supports process control without involving a controller. It is an open protocol, which means that devices from different certified manufacturers are compatible (interoperability).

FF defines two communication profiles, H1 and HSE. The H1-Profile with a data transmission rate of 31.25 kbit/s is preferably used for direct communication between field devices in one link (H1 link). The HSE profile with a transmission rate of 100 Mbit/s serves first and foremost as a powerful backbone for the link between H1 segments. FF linking devices serve as a gateway between the field devices on the H1 segments and the HSE backbone.

Advantages

- **Control in the field**
The FOUNDATION Fieldbus specification is uniquely different from other networking technologies in that it is not only a communication protocol but also a programming language for building control strategies which are distributed into the field devices. The control can be kept within the devices. In this case the number of controller can be drastically reduced.
- **Scheduled operation**
The execution of function blocks and transmission on H1 is deterministic and synchronized.
- **Redundancy**
See [Redundancy](#) on page 153

- **Online configuration**
Smaller configuration changes can be downloaded without disturbing the process. The Fieldbus Builder FF indicates if the download will interrupt the communication or not.
- **Online upgrade**
The Firmware/Software versions of all components can be upgraded online without disturbing the process.

Design

Introduction

FOUNDATION Fieldbus is flexible, supporting function block scheduling, which means that basic control and measurement features can be implemented similarly regardless of the device manufacturer.

Self-test and communication capabilities of microprocessor-based fieldbus devices reduce downtime and improve plant safety.

FOUNDATION Fieldbus (FF) is integrated into the controllers by the communication interface module for FOUNDATION Fieldbus HSE (CI860). In Control Builder, the CI860 is a hardware object created and configured in the project explorer.

The configuration of FF HSE Subnets is carried out with the Fieldbus Builder FOUNDATION Fieldbus (FBB FF). Thus configuration of CI860 requires both Fieldbus Builder FF and Control Builder M.

Design Example

Figure 44 shows the architecture of a system including engineering and operator station workplaces, controllers with FOUNDATION Fieldbus HSE CI860 communication interface units, linking devices, FF HSE devices, and H1 devices.

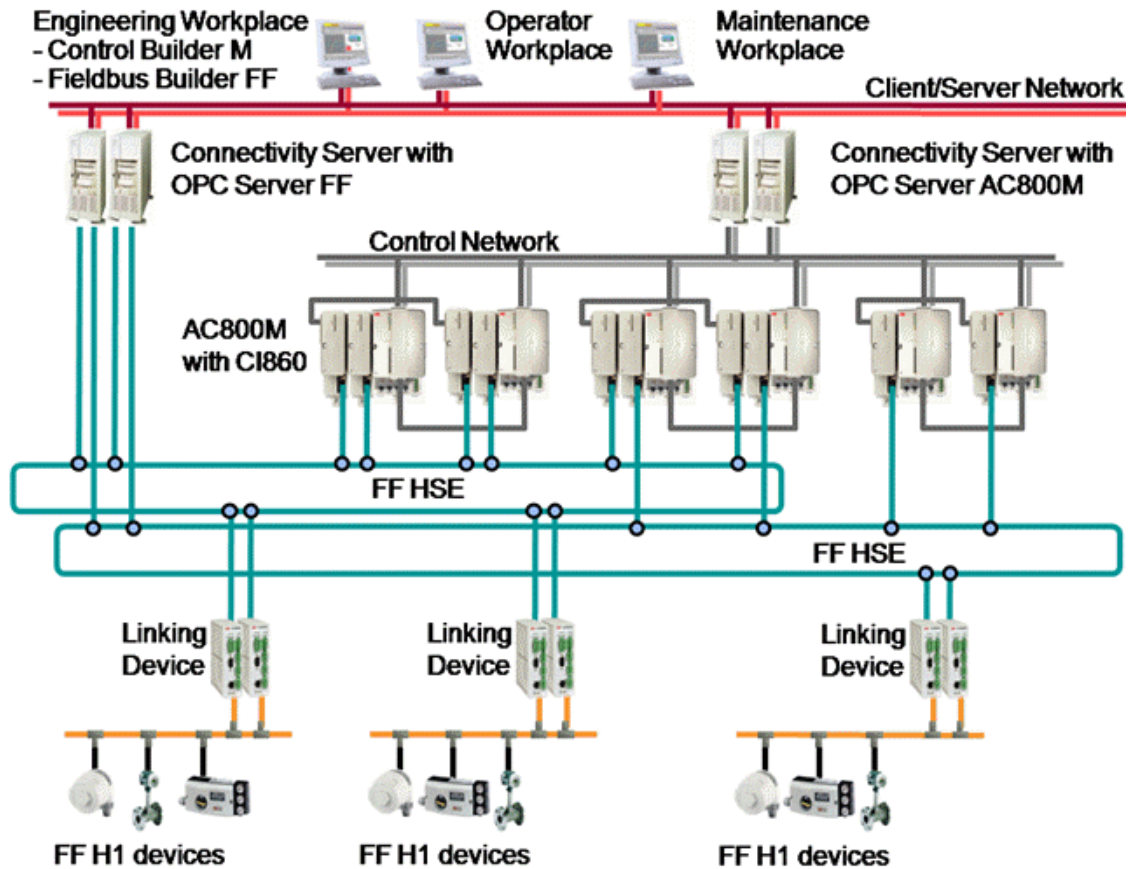


Figure 44. System 800xA topology with FF HSE

- Multiple HSE Subnets may be connected to a system.

- The FOUNDATION Fieldbus HSE CI860 communication interface units in the AC 800M Controller must be connected to a HSE Subnet.
- The Linking Device LD 800HSE connects H1 links to an HSE Subnet.
- FOUNDATION Fieldbus HSE Subnets should be physically separated from other networks as FOUNDATION Fieldbus HSE multicasts cause high load on the network.
- OPC Server FF provides tool routing functionality.

Connection Methods

The *publisher/subscriber* method signifies scheduled traffic on the FF H1 bus using publisher/subscriber connections between FF devices and the CI860. This connection must be setup in the Fieldbus Builder FF. Fieldbus Builder FF is used to map publisher/subscriber communicated FF signals to CI860 I/O channels. Thereby access to FF function block inputs and outputs being connected to an FF signal in Fieldbus Builder FF and being published or subscribed is possible.

A local connection between the controller and the CI860 must be setup using the Control Builder M. Therefore Control Builder M allows variables to be mapped to CI860 I/O channels. Dedicated control modules and function blocks allow for comfortable FF signal handling.

The *client/server* method describes unscheduled data traffic on the FOUNDATION Fieldbus. The OPC server FOUNDATION Fieldbus uses client/server communication. This allows access to all FF function block parameters for operation and maintenance purposes. The HSE Host CI860 module also allows client/server communication enabling access (read and write) to FF block contained parameters from the IEC 61131-3 controller application.

The *report distribution* method is typically used by fieldbus devices to send alarm notifications to the operator workstations.

For detailed information, refer to the *Device Management FOUNDATION Fieldbus - Configuration (3BDD012902*)* and *Control and I/O - FOUNDATION Fieldbus HSE - Engineering and Configuration (3BDD012903*)* manuals.

Redundancy

Redundancy is fully supported. This includes the communication interface module CI860, the Linking Device LD 800HSE, the OPC-Server FF and the HSE Subnet via redundant network components COTS.

Limitations and Performance

General

The CI860 communication interface unit cannot be used in an AC 800M High Integrity controller.

The following FF data types can be communicated:

- Publisher:
 - DS65
 - DS66
- Client/Server communication via CI860:
 - FFBitStrLen16DS14
 - FFBitStrLen8DS14
 - FFDiscreteSTatusDS66
 - FFFloatDS8
 - FFFloatStatusDS65

Dimensioning Limits, Linking Device

The linking device LD 800HSE supports up to 4 FF H1 links. For more information on linking device limitations, please refer to the manual *FOUNDATION Fieldbus Linking Device LD 800HSE, User instruction (3BDD013086*)*.

Dimensioning Limits, FOUNDATION Fieldbus HSE Communication Interface Module CI860

Cyclic communication via publisher/subscribe

The CI860 can handle a maximum of 1000 VCRs (Virtual Communication Relationships). Each VCR defines one I/O channel. Analog channels are mapped to the RealIO data type whereas discrete channels can be mapped to either the BoolIO or the DwordIO data type. The number of CI860 channels to which variables can be mapped is limited to the following numbers:

- 1000 channels of type Real for analog inputs
- 500 channels of type Real for analog outputs
- 500 channels for discrete input in total of type Bool and Dword
- 250 channels for discrete output in total of type Bool and Dword

The CI860 Hardware Editor contains 3000 channels, but only 1000 channels can be used at the same time.

To ensure a proper functionality under all conditions the CPU load of the CI860 shall not exceed 80% at a maximum. This gives the limit of the Average FF load of 100% that can be operated by the CI860 during runtime. The Average FF load is calculated and monitored by Fieldbus Builder FF depending on the actual configuration. The Average FF load is given by the following formula:

$$\text{Average FF load} = 1.25 * (9\% + T * 0.105\% + N * 0.015\%)$$

T: Number of transfers/sec (publish and subscribe in total)

N: Number of configured channels on CI860

9%: Idle load

Examples for Average FF load = 100%:

N = 100 channels configured => T = 662 transfers/sec

N = 500 channels configured => T = 605 transfers/sec

N = 1000 channels configured=> T = 533 transfers/sec

Acyclic communication via client/server

In Control Builder, it is possible to access the contained FF function block parameters, acyclically. This is done via client/server communication through CI860.

The following limitations apply for this communication per CI860 module:

- Maximum 30 Linking Devices
- Maximum 150 H1 devices
- Maximum 300 client/server signals

Hardware

The AC 800M controller needs to be connected to a CI860 CEX module to communicate with the Linking Device LD 800HSE on HSE Subnet.

HSE Subnets are based on the Ethernet standard. Therefore standard Ethernet components can be used to build an HSE Subnet. These components used in an HSE Subnet must be capable of handling multicasts as FOUNDATION Fieldbus uses multicast.

The Linking Device LD 800HSE connects the H1 links to an HSE Subnet. Since the Linking Device does not provide power to the H1 links, a power supply, a power conditioner and a bus termination is required for each H1 link.

Advanced

More information can be found in the manuals *Device Management FOUNDATION Fieldbus - Configuration (3BDD012902*)*, *FOUNDATION Fieldbus Linking Device LD 800HSE - User instruction (3BDD013086*)* and *Control and I/O FOUNDATION Fieldbus HSE - Engineering and Configuration (3BDD012903*)*. For information regarding wiring and installation of H1 please refer to the related documentation of the H1 devices and network components. Additional information can also be accessed from the Fieldbus Foundation web site <http://www.fieldbus.org>.

Troubleshooting

Errors for CI860 are indicated in the hardware tree and system alarms/events are generated. The Fieldbus Builder FF can be used to monitor FF HSE and H1 devices as well as the H1 links. Alarms/events are also generated for H1 devices and the Linking Device if so configured.

In case of a communication error, check the connections and terminations first.

Section 16 DriveBus

Introduction

The DriveBus protocol is used for communication between ABB Drives and the AC 800M controller, via the CI858 communication interface unit. The data exchange between the units is cyclic.

DriveBus communication is especially designed for sectional drive applications, for example ABB rolling mill drive systems and ABB paper machine control systems.

Supported media:

- DDCS (Distributed Drives Communication System) protocol,
- Optical fibers for improved interference immunity and large network distances,
- The CI858 communication interface unit is CE-marked, and meets the requirements specified in EMC Directive 89/336/EEC according to the standards EN 50081-2 and EN 61000-6-2.

Services Provided

- Dataset communication,
- Cyclic output/input to/from drives,
- Cyclic data to/from I/O units,

Advantages

- Supports many different types of drives and I/O units.
- Time synchronization of drives to common calendar time.
- Easy configurability of drives to be used with AC 800M.

- Identification method, self-checking and preventive systems to avoid incorrect configurations.
- Communication diagnostics for the application.
- No additional adapters required.

Design

DriveBus has specific definition parameters, required for device configuration. Examples of such parameters are *Configured application ID* and *Dataset x priority*.

The user connects all inputs and outputs to variables. DriveBus communication is automatically created when the application is downloaded to the controller.

Design Example

A DriveBus network typically consists of one or more drives, see [Figure 45](#).

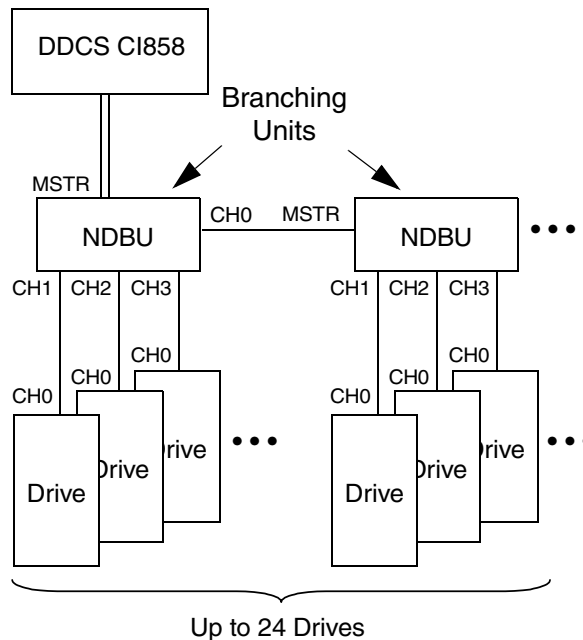


Figure 45. DriveBus system topology.

Dataset Communication

The data exchange between AC 800M, ABB Drives and I/O units, via CI858, consists of dataset pairs, which include input and output datasets, see [Figure 46](#). One dataset (DS) consists of three 16-bit words, called data words (DW).

Datasets are read from ABB Drives. Therefore, datasets need to be defined by setting ABB Drive dataset parameters during the system configuration. See [Configuration](#) on page 160.

It is possible to specify that 1-4 (depending on the drive type) datasets have higher communication priority than the others. This is done by specifying the parameter *Dataset x priority* in the ABB Standard Drive or in the ABB Engineered Drive configuration editor in the Settings tab.

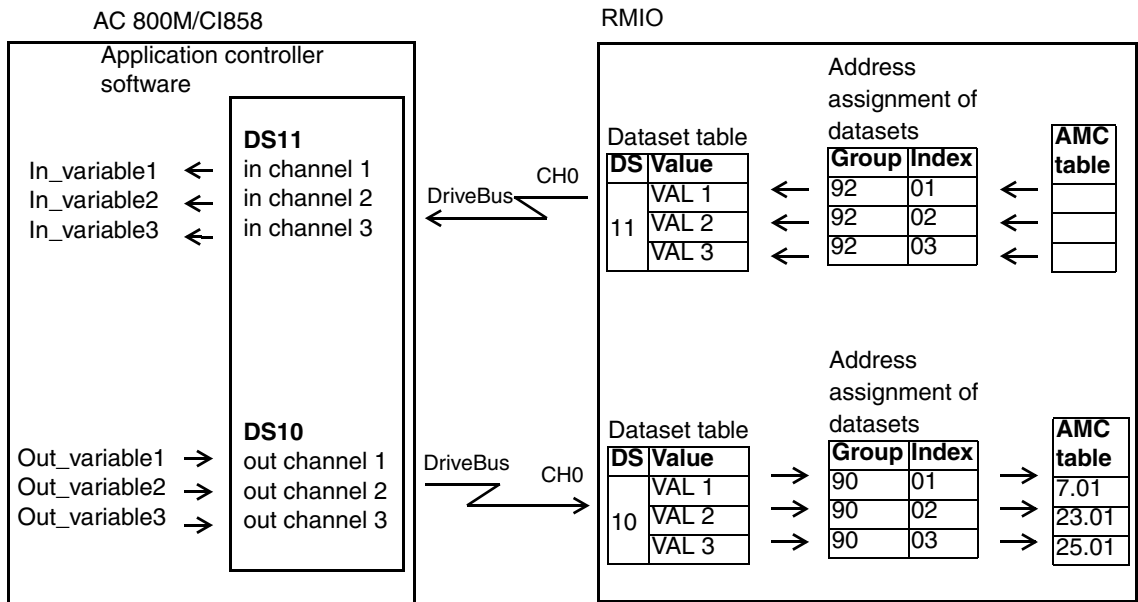


Figure 46. Dataset communication.

Configuration

To activate communication between the AC 800M, CI858, ABB Drives and I/O units, the system must be configured with valid parameters:

- Configure DriveBus communication using the Control Builder Professional engineering tool.
- Define datasets by setting ABB Drive dataset parameters, for example parameter groups 90...93 for Engineered Drives. See ABB Drives Firmware documentation for dataset and other required parameter settings.

The Control Builder configuration includes the following steps:

1. Add units to the hardware tree,
2. Define parameters,
3. Connect variables,
4. Download the project to the controller when all the required steps have been completed.



DriveBus communication may be halted during download. Refer to the Control Builder Professional online help for further details.

Redundancy

Redundancy is not supported.

Limitations

The CI858 communication unit cannot be used in an AC 800M High Integrity controller.

When a modified hardware configuration is downloaded to the controller, communication with hardware units may be interrupted:

- If modified CI858 parameters are downloaded to the controller, DriveBus communication is interrupted, and the affected CI858 will reboot.
- If modified drive parameters are downloaded to the controller, communication with the drive is interrupted, and a drive fault message, indicating communication loss, might be activated. If BusManager is not selected to monitor the connection, the fault can be avoided by adjusting the time delay of the drive communication loss supervision.
- If modified I/O parameters are downloaded to the controller, communication with the I/O unit is interrupted.
- If a drive or an I/O unit is added to or deleted from the hardware tree, and the changes are downloaded to the controller, the affected CI858 will reboot.
- If the hardware tree positions of different types of drives or I/O's are changed, and the changes are downloaded to the controller, the affected CI858 will reboot. Switching the position of two similar units will not result in a reboot of the affected CI858.
- Changing the connected channels of a drive or an I/O causes recalculation of the connections.

Performance

For each drive connected to the CI858 communication interface unit, 8 dataset pairs can be defined. The number of datasets per drive can be extended using special applications.

DriveBus is able to transfer a maximum of 8 dataset pairs/ms.

Hardware

The maximum number of CI858 units connected to the AC 800M is two. The CI858 unit has three channels. Every drive channel can be used for controlling up to 24 drives. The following drives are supported:

- ACS 800 / ACS 600 SingleDrive,
- ACS 800 / ACS 600 MultiDrive,
- ACS 800 / ACS 600 IGBT supply units,
- ACS 600 thyristor supply units,
- ACS 140 ... ACS 400,
- DCS 600 and DCS 400,
- ACS 6000 product family / large drives,
- ACS 1000 product family,
- Future drive types which are provided with DDCS interface,
- Special drive applications which require more than eight dataset pairs (the number of datasets is user-defined).

The PC Tool channel can be used for downloading firmware to CI858 units. CI858 firmware is downloaded with a special loading package, which does not involve Control Builder M.

CI858 connects to its units via three optical receiver/transmitter pairs. HP/Agilent Technologies Versatile Link Series (HFBR family) optical transmitter/receivers are used. The transmission speed of the fibre optic cables is 4 Mbit/s.

Advanced

For more information regarding DriveBus communication, see *CI858 DriveBus Communication Interface, User's Manual (3AFE68237432*)*.

Section 17 PROFIBUS DP

Introduction

PROFIBUS (PROcess Field BUS) is a fieldbus standard, especially designed for communication between systems and process objects. This protocol is open and vendor independent. With PROFIBUS, devices from different manufacturers can communicate without special interface adjustments. PROFIBUS can be used for both high speed, time critical transmission and extensive, complex communication tasks. PROFIBUS DP-V1 is implemented in AC 800M by using the CI854 module.



This manual only describes the CI854. Regarding legacy CI851 see older version of this manual.

Supported media:

- RS-485 transmission for universal applications in manufacturing automation
- IEC 61158-2 transmission for use in process automation
- Optical fibers for improved interference immunity and large network distances

The PROFIBUS DP (Decentralized Peripheral) fieldbus is based on European standard EN 50 170, and has been designed especially for communication between automation control systems and distributed peripherals at the device level.

PROFIBUS DP

The PROFIBUS DP communication profile is designed for efficient data exchange at the field bus level. The central automation devices, such as controllers, communicate through a fast serial connection with distributed field devices such as I/Os, drives, valves and measuring transducers. Data exchange with distributed devices is mainly cyclic.

PROFIBUS DP is suitable as a replacement for conventional, parallel signal transmission with 24V in manufacturing automation, as well as for analog signal transmission with 4-20mA or HART in process automation.

Services Provided

- PROFIBUS DP-V0 and PROFIBUS DP-V1 are supported.

Advantages

- High information transfer rate up to 12 Mbit/s,
- Supports many different types of I/O units,
- Master redundancy,
- Line redundancy,
- Slave redundancy,
- PROFIBUS Diagnostics,
- Multi-master,
- Online changes,
- Acyclic DP-V1 services (Tool Routing),
- Online upgrade.

Design

Introduction

A PROFIBUS (DP or PA) device has specific definition parameters, required for device configuration, stored in a GSD file (the GSD file format is given in European standard EN 50170). Examples of such parameters are the device version, the timing parameter of a device, the supported baudrate, the data format and the length of the I/O-data. For ABB devices - such as the S800 I/O, S900 I/O, and S200 I/O unit families - the device is already integrated in the Control Builder's hardware library.

For a device from another manufacturer the configuration parameters are stored in a .GSD file delivered with the device. This file has to be imported with the

Device Import Wizard. During the download an automatic PROFIBUS master configuration is done. This calculation is performed for all controllers in the project and for all PROFIBUS masters connected to the respective controllers. Typically the user only has to configure the node addresses, not the timing parameters (which are necessary to configure only if the passive components are used as optical links). The PROFIBUS communication can be started when the PROFIBUS masters are configured, the node addresses for the master and the slave are set, and if necessary the default baudrate (1,5 Mbit/s) is changed. Please refer to the manual *PROFIBUS DP, Engineering and Configuration (3BDS009030*)* for more information.

The user connects all inputs and outputs to variables. The PROFIBUS communication is automatically created when the application is downloaded to the controller. PROFIBUS is primarily used for cyclic I/O communication. When communication is defined, the master will begin to cyclically ask the slaves for data and send data. The two outermost nodes must be terminated.

Design Example

A PROFIBUS DP network typically consists of one or more masters and many slave devices, see [Figure 47](#).

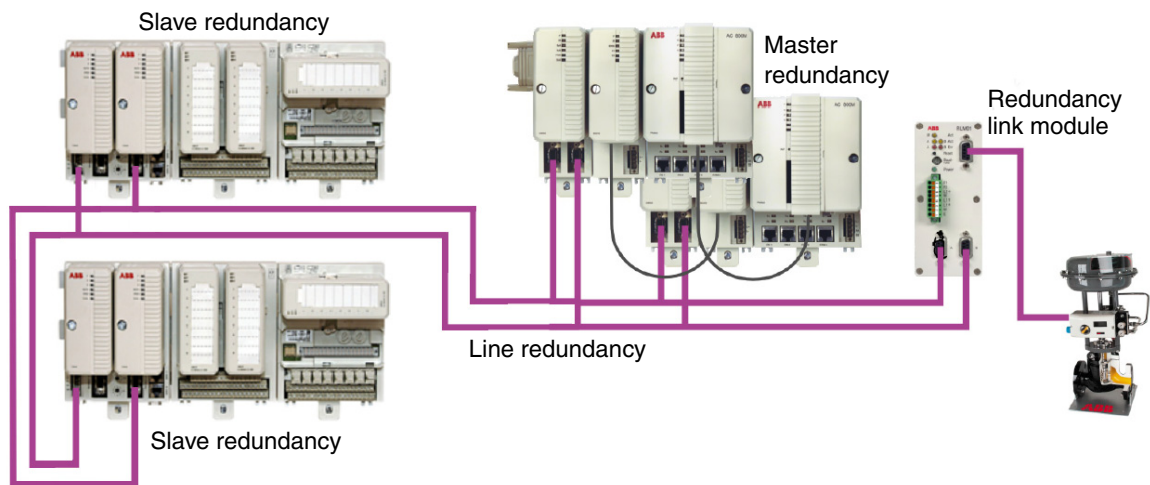


Figure 47. PROFIBUS DP network with I/O units.

Redundancy

Both line redundancy and slave redundancy are built in. Using two CI854A communication interface units adds master redundancy. For additional information and redundancy structures with optical links, please refer to the manual *PROFIBUS DP, Wiring and Installation (3BDS009029*)*.

Limitations

- CI854 can only act as master.
- The network can have a maximum of 126 nodes. A maximum of 124 slaves can be connected to a CI854 since the node addresses 0 and 1 are reserved for CI854.
- S800 I/O connected to CI840 and/or S900 I/O connected to CI920 supports cable redundancy together with slave redundancy.
- If the PROFIBUS master unit, CI854, loses contact with a slave unit, for example due to a disconnected cable, input values are set according to ISP configuration. If the I/O unit does not support ISP, all input values will freeze.
- Reset of PROFIBUS DP master, CI854, and the complete PROFIBUS is done if one of the following bus parameter settings are changed: Node address of CI854, baud rate or highest station address (HSA). A change of the other bus parameters does not affect the running communication.
- If the CI854 is running with 12 Mbit/s, then in total 4000 bytes input and output data for the cyclic communication are allowed to be configured. For lower Baudrate than 12 Mbit/s there is no limitation



Online changes are supported by S900 (CI920) and S800 (CI840 and CI801), that is, modules can be added/changed without data being sent to ISP or OSP.

Performance

The cycle time on PROFIBUS depends on the baud rate, the summary of I/O data and the slave timing parameter. The fastest cycle time is about 1 ms with a baud rate of 12 Mbit/s and only one slave device. The typical cycle time is about 10-20 ms with 1,5 Mbit/s and some slave devices.

CI854 slave devices can have node addresses in the range 2-125 (the node addresses 0 and 1 are reserved for the CI854). The baud rate can be configured to be in the range of 9,6 kbit/s - 12 Mbit/s. There is a maximum length of I/O data at 4000 bytes of input and output data in total when using 12 Mbit/s. For slower baud rate, up to 1,5 Mbit/s, there is no limitation of the length of the I/O data.

Hardware

The AC 800M controller needs to be connected to a CI854 CEX module to communicate with PROFIBUS DP devices. A shielded twisted pair cable with terminating resistors, or a fiber optic cable with optical link units is required.

The physical medium for PROFIBUS DP is RS-485, which allows 32 nodes in a segment and 126 nodes in a network. The maximum cable length may vary from 100 to 1200 m depending on transmission speed. The Cable length can be extended using fiber optic modems (yielding a more robust network).

Segment couplers can be used to attach PROFIBUS PA devices.

For a product guide presenting all available hardware, visit the PROFIBUS web site: <http://www.profibus.com>

Advanced

More information concerning PROFIBUS DP can be found in the manuals *PROFIBUS DP, Engineering and Configuration (3BDS009030*)* and *PROFIBUS DP, Wiring and Installation (3BDS009029*)*. Additional information can also be accessed from the PROFIBUS web site <http://www.profibus.com>.

Troubleshooting

The system software handling the PROFIBUS-DP communication in the controller writes diagnostic information to the controller log.

Errors are indicated in the hardware tree and system alarms/events are generated. There is also a log file (stored in the same place as the Control Builder log files) named "Profibus_DPV1_Calculation.txt" which is created by the automatic calculation of the PROFIBUS master parameters and can be useful for

troubleshooting. In case of a communication error, check the connections and terminations first. Use the web server, described in the manual *PROFIBUS DP, Engineering and Configuration (3BDS009030*)*, to get more information.

CI854 Web Interface

The CI854 Web Interface is needed in case of service activities. It is described in the manual *PROFIBUS DP, Engineering and Configuration (3BDS009030*)*. It also describes how to change the node addresses for devices that do not have any switches like the PROFIBUS PA devices.

Section 18 PROFINET IO

Introduction

PROFINET is an open Fieldbus standard for applications in manufacturing and process automation. PROFINET technology is an international standard that is part of IEC 61158 and IEC 61784.

The two perspectives of PROFINET are:

- PROFINET IO, which is used to integrate simple distributed I/O and time-critical applications into Ethernet communication.
- PROFINET CBA, which is used to integrate distributed automation system into Ethernet communication.

The PROFINET integration into ABB System 800xA focuses on the I/O connectivity. Therefore, only the PROFINET IO technology is used for the integration.

PROFINET IO is based on IEEE 802.3. It supports a transmission speed of 100 Mbps with auto negotiation and auto crossover in a switched Ethernet network. PROFINET IO uses Ethernet as well as TCP, UDP, and IP as the basis for communications. It is designed to work with other IP-based protocols on the same network.

Communication in PROFINET IO has different levels of performance:

- The transmission of non time-critical parameters and configuration data occurs in the standard channel of PROFINET IO based on TCP/IP or UDP.
- The transmission of time-critical process data within the production facility, occurs in the Real Time (RT) channel, also described as soft real-time.

For challenging tasks, the hardware based communication channel Isochronous Real-Time (IRT) is defined. For example, IRT can be used in motion control applications and high performance applications in factory automation.



The PROFINET IO implementation in System 800xA supports only RT channel. There is no support for IRT.

When distributed I/O applications are connected for communication through PROFINET IO, the familiar I/O view of PROFIBUS is retained. The peripheral data from the field devices are periodically transmitted into the process model of the control system.

PROFINET IO describes a device model oriented to the PROFIBUS framework, which consists of places of insertion (slots) and groups of I/O channels (subslots). The technical characteristics of the field devices are described by the General Station Description (GSD) file, which is based on XML. The PROFINET IO engineering is performed in a way familiar to PROFIBUS. The distributed field devices are assigned to the controllers during configuration.

The PROFINET IO is interfaced to the AC 800M controller (under System 800xA) using the PROFINET IO module CI871.

Services Provided

The services provided by PROFINET IO integration into ABB System 800xA are:

- PROFINET RT for I/O connectivity
- Sequence of events with ABB SOE profile

Advantages

The advantages of PROFINET IO are:

- Multi Controller Access.
- Transmission speed of 100 Mbps with auto negotiation and auto crossover in a switched Ethernet network.
- Supports third party PROFINET IO devices.
- Supports Application and Ethernet Network redundancy.
- PROFINET Diagnostics.

- Sequence of Events.
- Online changes. For more information on online changes, refer to *AC 800M PROFINET IO Configuration (3BDS021515*)* manual.

Design

Introduction

PROFINET IO describes a device model oriented to the PROFIBUS framework, which consists of places of insertion (slots) and groups of I/O channels (subslots).

PROFINET IO is configured using the Control Builder. The configuration includes the planning of the hardware units in the hardware tree, specific configuration for the PROFINET IO communication interface CI871 and the PROFINET IO devices. The device specific configuration data is described within the GSD file provided by the device manufacturer. To configure the PROFINET IO device within the Control Builder, the GSD file must be imported into a hardware library and inserted to the project using the Device Import Wizard.

Design Examples

Figure 48 shows an example of PROFINET IO installation with AC 800M controller.

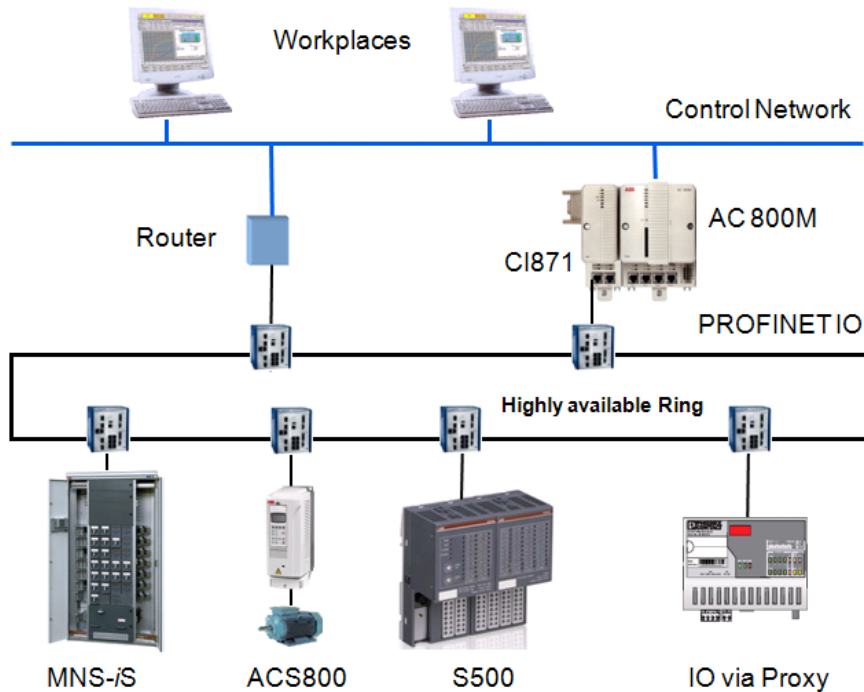


Figure 48. PROFINET IO with AC 800M

Redundancy

PROFINET IO supports application redundancy and Ethernet network redundancy. System integrated redundancy is not supported in PNIO. For more information on Ethernet network redundancy, refer to *AC 800M, PROFINET IO Configuration (3BDS021515*)*.

Online Upgrade

The CI871 supports Online Upgrade. The CI871 will be restarted during Online Upgrade. This will be done even if only the Controller firmware is upgraded. During restart, no internal status will be transferred and communications will be disconnected.

Technical Data

For PROFINET IO configurations in 800xA with CI871 the following dimensioning guidelines need to be taken into account:

- Up to 12 CI871 per AC 800M controller.
- Up to 126 PNIO devices per CI871.
- Up to 512 modules per PNIO device.
- One IOCR for each direction (Input and Output) per PNIO device, each IOCR up to 1440 bytes of I/O data.
- Update times down to 1 ms (only if CI871 has one device configured).
- For CPU-load calculation of CI871 the Ethernet frames for inputs and outputs need to be calculated. CI871 can handle as a maximum one frame per ms in each direction.

Example 1: Update times for all devices configured to 32 ms (default), then up to 32 devices can be connected to CI871.

Example 2: Update times for all devices configured to 8 ms, then up to 8 devices can be connected to CI871.



The limitation for the CPU load of CI871 is checked by the system during download. If the system detects that there is an CPU overload, then it is indicated the Compilation Summary window and the download is blocked. CI871 may not function properly when there is an overload. User can check the CPU load before and after download by use of the Web Interface. The limit for the CPU load is 100%. Up to that value the CI871 works stable without any problems or restrictions.

- The CI871 communication interface unit cannot be used in an AC 800M High Integrity controller.

Hardware

The CI871 can be used to connect a AC 800M controller to PROFINET IO devices. The maximum number of CI871 that can be connected to a AC 800M controller is 12 nonredundant units using the CEX-Bus.

Troubleshooting

The system software, which handles the PROFINET IO communication in the controller writes the diagnostic information to the controller log. Errors are indicated in the hardware tree and corresponding alarms or events are generated.

During download, the Control Builder creates a log file **PROFINET_Configuration.txt**. This log file will have the result of the download compilation for the current and previous configurations. The log file can store data upto 10 MB and is stored in the LogFiles directory in Control Builder.

In case of a communication error, check the connections and terminations. For more information, refer to Web Interface described in the *AC 800M, PROFINET IO, Configuration (3BDS021515*)* manual.



If communication errors occur, then check for the consistency of the symbolic name of PNIO device. The name stored on the device should be identical with the station name configured in the Control Builder.

The network configuration should also be checked in case of communication error.

CI871 Web Interface

The CI871 Web Interface is used for commissioning and maintenance. It can also be used to configure symbolic name for PNIO devices. For more information on Web Interface, refer to the Web Interface section in *AC 800M, PROFINET IO, Configuration (3BDS021515*)* manual.

Section 19 Self-defined UDP Communication

Introduction

The UDP Communication library (UDPCommLib) contains function block types for AC 800M controller communication with external devices through Ethernet, using UDP.

Some of the examples of usage are:

- Communication with different road-infrastructure network nodes as variable speed signs, traffic direction and information signs.
- Vision cameras. Many implement the Telnet protocol (ASCII TCP communication over standard port number 23).
- Information server - The controller may act as both client and server on the network. Example of server use is a SCADA application where a supervisory system connects to different servers and collects information periodically.

The function block types in UDPCommLib are non-SIL and cannot be used in an AC 800M HI controller. UDP communication does not establish a connection prior to sending/receiving. The UDPCommLib also supports broadcasting where one node can send a message received by many. But it doesn't ensure that a message is delivered.

Protocols implemented on UDP need to consider lost telegrams, message ordering and re-send timing. Some network equipment prioritizes down the UDP traffic when the network load is high.

Design

The UDPCommLib library supplies IEC 61131-3 function blocks that make it possible to read and write a struct of dints or dwords from/to controller's on-board Ethernet channels, CN1 and CN2.

The following function block types are available:

- UDPCConnect – The UDPCConnect function block is used to open and close a defined UDP communication channel.
- UDPWrite - Writes a struct of dints or dwords.
- UDPRead - Receives a struct of dints or dwords.

Every function block has an Id parameter and all function blocks are connected to each other through the Id parameter.

The read block will perform one read operation on positive edge of the Req parameter (Read request).

If no message is available the read operation will be pending as long as the Req parameter is set to true. When set to false any pending read operation will be aborted.

The read function can be used by setting the Req parameter to true and wait for a Ndr parameter (New data received) to be set to true. When Ndr is set to true then toggle the Req (False -True) to read next message.

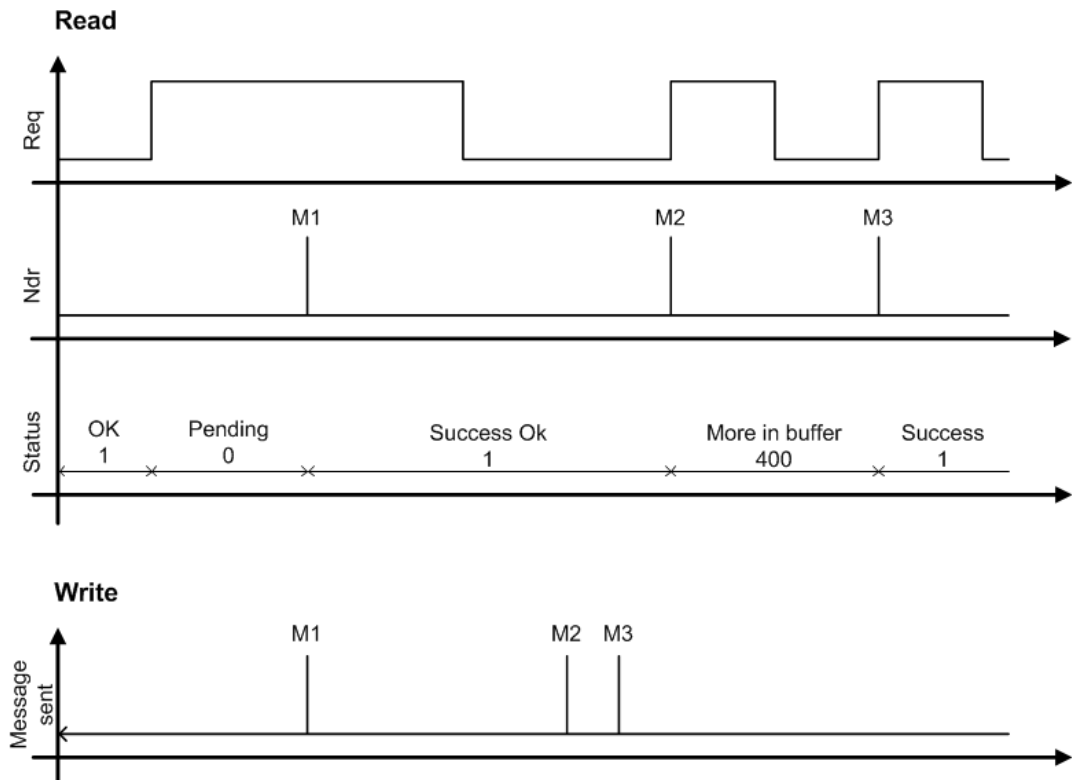


Figure 49. Execution of `UDPRead` and `UDPWrite` blocks

Example of the code:

```

IF ReadMessage THEN
  Req := NOT Ndr;
ELSE
  Req := False;
END_IF;

```

Hardware

The UDP communication uses the controller's Ethernet ports, CN1 and CN2.

The hardware configuration needed in Control Builder is to insert the *UDPProtocol* hardware type (from UDPHwLib), under the IP hardware type under the controller. The position of *UDPProtocol* is always 0.5.2.

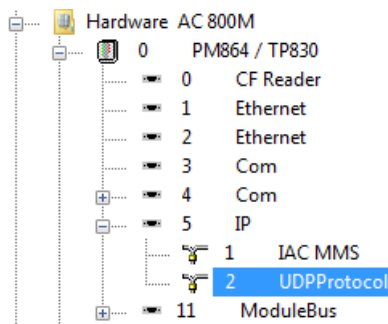


Figure 50. UDPProtocol hardware type inserted under IP

Performance

The memory consumed by downloading the UDP protocol does not exceed 100 kB. Dynamic memory is allocated for each UDP connection. The maximum communication throughput is dependent on the other tasks in the controller.

Limitations

- UDP is not supported in AC 800M High Integrity controllers.
- UDP is not supported in soft controllers.
- Fragmented Ethernet packets are not supported by the controller. This limits the IEC 61131-3 variable structure sent through the UDP to 1472 bytes. For example, if using 4 bytes per item in Sd/Rd, then the UDP variable structure can be maximum 368 dints/dwords.
- The IEC 61131-3 read/write blocks returns error message, if the size exceeds.

- Maximum of ten different UDP connections can exist in the controller.
- The UDP protocol defines available ports in the range 0-65535 (16 bits unsigned). Some of the ports cannot be used because they are used by other functions. For a list of used ports, see [Used Ports](#) on page 197. Port 0 is also not used because it is filtered away by the controller firewall.
- Time Critical tasks cannot be used with any of the function blocks in UDPCommLib.

Redundancy

RNRP can be utilized for redundant communication.

Online Upgrade

During Online Upgrade, the connections which are open will be disconnected before the switch, and opened again in the new primary. This is managed automatically.

Troubleshooting

The Status parameter can be a good guide when tracking errors. The status codes are listed and described in Control Builder online help.

Section 20 Self-defined TCP Communication

Introduction

The TCP Communication library (TCPCommLib) contains function block types for AC 800M controller communication with external devices through Ethernet, using TCP.

The typical application areas are the following:

- Communication with different road-infrastructure network nodes such as variable speed signs, traffic direction and information signs.
- Vision cameras. Many implement the Telnet protocol (ASCII TCP communication over standard port number 23).
- TCP is used in Information server. The controller may act as both client and server on the network. Example of server use is a SCADA application where a supervisory system connects to different servers and collects information periodically.

Design

The TCPCommLib library supplies IEC 61131-3 function blocks that make it possible to read and write a struct of dints or dwords from/to controller's on-board Ethernet channels, CN1 and CN2.

The following function block types are available:

- TCPServerConnect – The TCPServerConnect function block is used to let the controller become a TCP server waiting for connection requests initiated by other TCP clients on the network.
- TCPClientConnect – The TCPClientConnect function block is used to open and close a TCP connection to a remote TCP server on the network.

- TCPWrite - Writes a struct of dints or dwords.
- TCPRead - Receives a struct of dints or dwords.

Hardware

The TCP communication uses the controller's Ethernet ports, CN1 and CN2.

The hardware configuration needed in Control Builder is to insert the *TCPProtocol* hardware type (from TCPHwLib), under the IP hardware type under the controller. The position of *TCPProtocol* is always 0.5.3.

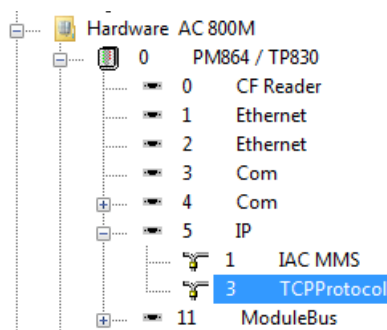


Figure 51. *TCPProtocol* hardware type inserted under IP

Performance

The memory consumed by downloading the TCP protocol does not exceed 100 kB. Dynamic memory is allocated for each TCP connection. The maximum communication throughput is dependent on the other tasks in the controller.

Limitations

- TCP is not supported in AC 800M High Integrity controllers.
- TCP is not supported in soft controllers.

- Fragmented Ethernet packets are not supported by the controller. This limits the IEC 61131-3 variable structure sent via TCP to 1420 bytes. For example, if 4 bytes per item is used in Sd/Rd, then the UDP variable structure can be max 355 dints/dwords. If the size exceeds the IEC 61131-3 read/write blocks will return error message.
- Maximum of ten different TCP connections can exist in the controller.
- The TCP protocol defines available ports in the range 0-65535 (16 bits unsigned). Some of the ports cannot be used because they are used by other functions. For a list of used ports, see [Used Ports](#) on page 197. Port 0 is also not used because it is filtered away by the controller firewall.
- Time Critical tasks cannot be used with any of the function blocks in TCPCCommLib.
- After a power-fail, the TCP communication server and client needs to be restarted to establish the communication again.

Redundancy

RNRP can be utilized for redundant communication.

Online Upgrade

During Online Upgrade, the connections which are open will be disconnected before the switch, and opened again in the new primary. If the controller is client, it needs to reconnect after the upgrade. If the controller is server, the connected client device/node need to reconnect since the server is taken down and then up.

Troubleshooting

The Status parameter can be a good guide when tracking errors. The status codes are listed and described in Control Builder online help.

Section 21 Self-defined Serial Communication

Introduction

Function blocks in the Serial Communication Library (SerialCommLib) allow implementation of a personal character-oriented protocol on a serial port and writing an application that both controls the characters sent and checks that the correct answer is received by using various checksum algorithms.

The typical application areas are the following:

- Read characters from an input device, such as a bar code reader.
- Read and write from/to a terminal.
- Implement simple serial communication protocols.
- Print alarm texts on a printer.

Design

The SerialCommLib library supplies IEC 61131-3 function blocks that make it possible to read and write a string value from/to a COM port on a hardware interface. The following function block types are available:

- *SerialConnect* - Creates and performs a setting, and also opens and closes a serial channel.
- *SerialSetup* - Changes settings that belong to a serial channel (for example checksum calculation and settings for echo handling at sending and receiving).
- *SerialWriteWait* - Writes a string and expects a reply (a string) from the unit it communicates with.
- *SerialListenReply* - Receives a string and then writes a string to the unit it communicates with.

- *SerialWrite* - Writes a string to the unit it communicates with.
- *SerialListen* - Receives a string that has been written from the unit it communicates with.

The parameter *Channel* of the *SerialConnect* function block is used for addressing the COM port which is used for the communication. Use one and only one *SerialConnect* for each COM port. Every function block has an *Id* parameter and all function blocks are connected to each other via the *Id* parameter.

Hardware

Self-defined Serial Communication can be used on the built in COM3 port (on an AC 800M Controller) and optionally on the CI853 ports. The CI853 supports Hot Swap.

Performance

The performance of the Self-defined Serial Communication depends of the way the information is handled, and it is not possible to make general statements.

Limitations

- The maximum size of a string to be received/sent is limited to 140 characters.
 - Please observe that a message often includes a *Checksum* part or *end character* and sometimes *character after end character*, which also have to fit in the 140 character message.
- The serial protocol can only be executed in half duplex. Accordingly it cannot send and receive simultaneously.
- A maximum of one function block enabled on each channel.
- Function blocks connected to a *SerialConnect* function block can only be active one at a time. This means that the communication channel is busy until the activities of the current function block have been completed.
 - Exception: Two *SerialWrite* function blocks can be combined in order to print a longer string than 140 characters, see [Advanced](#) on page 187.

- An enabled SerialListen will block a SerialWrite.
- Preferably use ASCII telegrams, since binary telegrams are difficult to implement, but possible.

Redundancy

Redundancy is not supported.

Advanced

To be able to print a string longer than the maximum length of 140 characters, call to subsequent SerialWrite function blocks in order like following:

```
Write1( Req := TRUE,  
        Id := Id,  
        EndChar := EndChar_Write1,  
        Done => Done_Write1,  
        Error => Error_Write1,  
        Status => Status_Write1,  
        Sd := Sd_Write1 );
```

```
Write2( Req := TRUE,  
        Id := Id,  
        EndChar := EndChar_Write2,  
        Done => Done_Write2,  
        Error => Error_Write2,  
        Status => Status_Write2,  
        Sd := Sd_Write2 ).
```

Troubleshooting

There is a *Status* parameter which can be a good guide when tracking errors. The status codes are listed and described in Control Builder online help.

Note that the duration of the *Error* and *Done* parameters is a pulse during one scan only. Therefore latching in the application is required to keep these signals.

If there is a power fail on the AC 800M unit while the *SerialListenReply* is treating a message, the error status -5331 appears. To restart the serial communication, disable the function block and enable it again.

Section 22 Modem Communication

Introduction

There are two types of modems:

1. Short distance modems for point-to-point private links (copper or fiber optic cable) and which can be used with twisted pair Ethernet, PPP, COMLI, Siemens 3964R, MODBUS RTU or PROFIBUS-DP.
2. Dial-up modems that use the public telephone system. COMLI is the only protocol that supports dial-up modems.

Short Distance Modem

There are two main reasons for using short distance modems:

1. Permitted increase of the allowed maximum length of RS-232C and twisted pair Ethernet connections.
2. Elimination of the risk of electromagnetic interference and unauthorized intrusion by use of fiber optic modems.

There are a large number of modems on the market with different types of connectors that can convert within a range of networks. Recommended types are Westermo, and for fiber optic modems Hirschmann.

Fiber optic modems are available that support cable redundancy.



Communication via twisted pair Ethernet must be half duplex. In Hirschmann modems this must be selected by setting a hardware switch.

Dial-Up Modem



In this section, the term “modem” refers to modems that are configured and controlled by a controller. It does not refer to modems that are transparent to the controller.

The COMLI master function can use a dial-up modem (Hayes modem). Recommended types are Westermo, for industrial applications, and US Robotics for office environments.

Two types of function block are associated with the COMLI modem function:

- *ModemDialUp*
Initiates a Hayes dial-up operation
- *ModemHangUp*
Initiates a Hayes hang-up operation
- *ModemConnStat*
Initiates a Hayes connection operation

Connection diagrams for modem cables are provided in the hardware and operation guides of the different controllers.

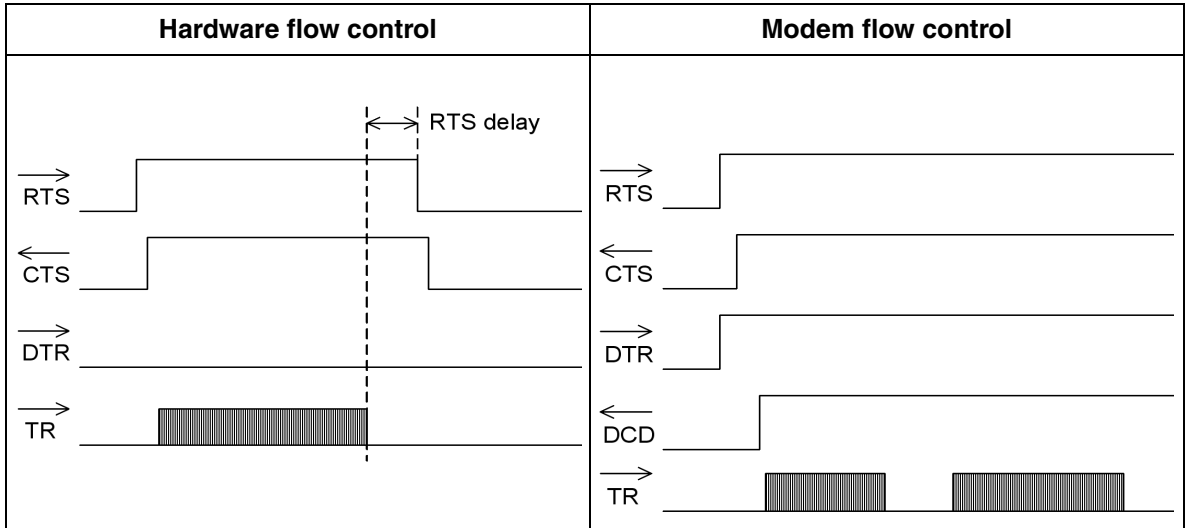
The procedures initiate a Hayes modem operation. These procedures are asynchronous; i.e. only one *dial* or *hang-up* operation is permitted at a time.

Hardware and modem flow control is illustrated in [Table 20](#) for their RTS (Request To Send), CTS (Clear To Send), DTR (Data Terminal Ready) and TR (Transmission) signals respectively. For the modem flow control, there is also an illustration for the DCD (Data Carrier Detect) signal.

The DTR signal from the controller must be high, or the modem disconnects the communication. The DCD signal from the modem is high when the carrier wave is present. The RTS and CTS signals are used for hardware flow control. The RTS signal is high when the controller has data to send, and the CTS signal is high when the modem is ready to receive data.

The modem must be set for echo off, verbal result codes, and auto answer.

Table 20. Illustration of hardware and modem flow control.



The Hayes init command in the parameter list has the default value `ATE0V1S0=1`, which means no echoing, verbal result codes, and auto answer. If the modem's default factory settings do not imply normal use of DTR and DCD, add the commands `&D2` and `&C1` to the init string. To use 9600 baud, add the command `F8` for the Westermo modem and `&N6` for US Robotics. For other modem manufacturers, refer to the relevant manual.



The init command is sent only to the modem connected to the dialing controller. To apply the same settings to the modem at the other end, at least one dial-up must also be performed from that controller. Also note that US Robotics modems use only one type of parity and that you must adapt the communication settings accordingly.

As an alternative to using function blocks, the Automatic Connect parameter can be set to imply automatic connection to the default phone number when data is sent through the serial port. Another parameter is the Idle Time which sets the (idle) time connection. This means the time between last send data and until the modem is disconnected.

Limitations

- Communication with short distance modems via twisted pair Ethernet must be half duplex.
- COMLI is the only protocol that supports dial-up modems.
- PPP via modem must use RS-232.

Performance

Refer to the relevant documentation from the modem manufacturer.

Troubleshooting

Refer to the relevant documentation from the modem manufacturer.



If you change the modem parameters, you may need to restart the modem before your changes will take place.

Appendix A OSI Profile for MMS

This appendix lists the available MMS services and describes the reduced OSI-profile used.

MMS Services

Table 21. MMS services supported in version 5.1 of Control Software for AC 800M

Environment and general management	
Initiate	Yes
Conclude	Yes
Abort	No
VMD management	
GetNameList	Yes
GetCapabilityList	Yes
Domain management	
InitiateDownloadSequence	Yes
DeleteDomain	Yes

Table 21. MMS services supported in version 5.1 of Control Software for AC 800M (Continued)

Variable access	
Read	Yes
Write	Yes
InformationReport	No
GetVariableAccessAttributes	No
DefineNamedVariable	No
DeleteNamedVariable	No
GetNamedVariableListAttributes	No
DefineNamedVariableList	No
DeleteNamedVariableList	No
ServiceError	Yes
Journals	
InitializeJournal	No
ReadJournal	No
File management	
FileOpen	Yes
FileRead	Yes
FileDelete	Yes
FileClose	Yes
FileDirectory	No

Reduced OSI Implementation

Table 22. Reduced OSI implementation

OSI model layer	Specification	Comments
Application	ISO/IEC 9506: Manufacturing Message Specification (MMS) ISO 8650: Protocol Specification for the Association Control Service Element	At application level only ISO/IEC 9506 is supported; i.e. ISO 8650 is not implemented
Presentation	ISO/IEC 8823: Connection Oriented Presentation Protocol Specification.	Not implemented, i.e. NULL layer
Session	ISO/IEC 8327: Basic Connection Oriented Session Protocol Specification.	Not implemented, i.e. NULL layer
Transport	IETF RFC1006 (OSI over TCP) IETF RFC 793 (TCP)	Fully implemented Fully implemented
Network	RFC 791 (IPv4) RNRP ⁽¹⁾	Fully implemented
Data link	ISO/IEC 8802: Logical Link Control ISO/IEC 8802-3: Carrier Sense Multiple Access with Collision Detection. CNCP ⁽²⁾	Ethernet ⁽³⁾
Physical	ISO/IEC 8802-3: Carrier Sense Multiple Access with Collision Detection.	

(1) In addition to MMS, the ABB Redundant Network Routing Protocol operates as network layer.

(2) ABB time synchronization.

(3) PPP can also be used as data link with RS-232 as physical layer.



The services are connection-oriented. Connection-less or multicast services for MMS are not supported

Appendix B Used Port Numbers

Used Ports

Table 23 is a summary of used ports on CN1 and CN2 on the PM8xx controller. Dynamically allocated ports is used by MMS Client (tcp).

Table 23. Used server port for each function

Function	Server Port
IAC	2757/udp
MMS	102/tcp
RNRP	2423/udp
SNTP	123/udp
SattBus on TCP/IP	2999/udp
Show Remote System	147/udp
CNCP	3341/udp
NIS	24230/udp
Web server for PROFIBUS	80/tcp



All unused ports are closed due to security reasons.



The built-in Windows firewall, by default, blocks port 147.

It means that this port must be opened in the firewall in order to use Control Builder's Show Remote System command.

Appendix C Configuration of HART Devices

Introduction

HART (Highway Addressable Remote Transducer) is an open system communication protocol that makes possible remote configuration and supervision of I/O devices with HART support. Scaling and calibration of I/O values is an implementation of *Tool Routing* and can be performed from workstations running Operate^{IT} via the AC 800M controller and ModuleBus or PROFIBUS DP/V1. ABB provides units with HART support in the S800 I/O and S900 I/O families, though units from other vendors may also be used.

To get access to the HART devices via Tool Routing the Field IT products for PROFIBUS and HART have to be installed. The installation procedures are described in the *800xA System Installation manual*.

Table 24. Units with HART support.

AC 800M communication interface	Fieldbus communication interface	I/O units
CI854 for PROFIBUS DP-V1	CI840, CI801	AI845, AO845, AI895, AO895
	CI920	AI930, AO930
AC 800M CPU	Modulebus	AI845, AO845, AI895, AO895, AI880A

Configuration Example

An AC 800M has S800 I/O units locally connected directly via ModuleBus, S900 I/O remotely connected via a CI854 PROFIBUS DP-V1 communication interface and a CI920 fieldbus communication interface. The workstation must be able to communicate directly with the controller.

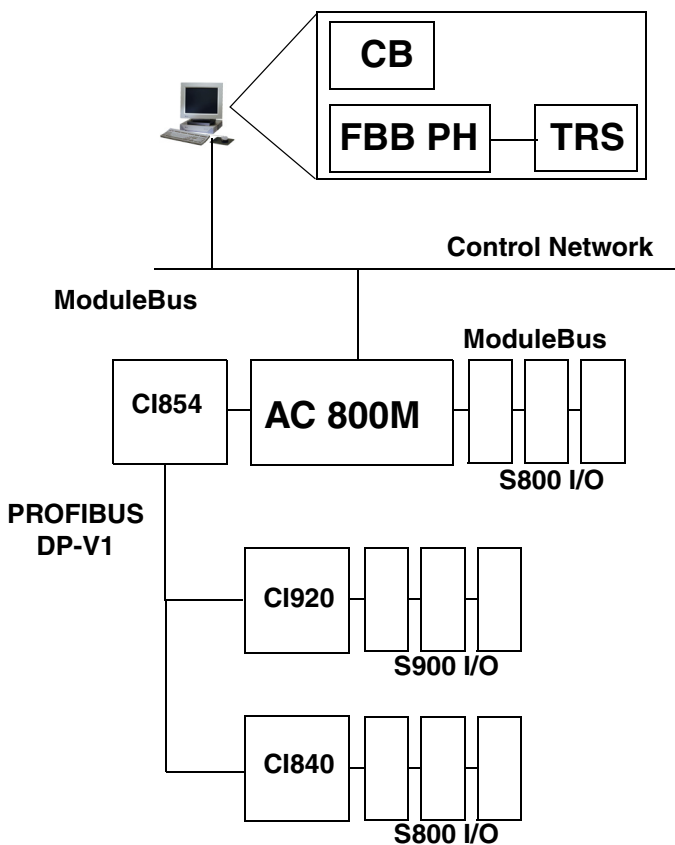


Figure 52. Configuration example.

The *Fieldbus Builder PH* (FBB PH), the *Control Builder* (CB) and the *Tool Routing Service* (TRS) must run on the same workstation. TRS communicates with the AC 800M controller via MMS. The only measure necessary is to enable the Tool Routing parameter belonging to the controller CPU in the project explorer.

DTMs (Device Type Managers) are device-specific software components, supplied by the field device manufacturer with the device. User interface device parameters, configuration data, etc., for a device, can be accessed through the DTM. It is the manufacturer's decision as to which services the DTM is to offer the user. Because DTMs typically are made by different vendors, data is supplied in XML format.

A device such as AC 800M, which supports gateway functionality, must have one DTM for each type of protocol: In our example one AC 800M ModuleBus DTM and one CI854 DTM for PROFIBUS.

Toolrouting

In order to establish toolrouting when devices with gateway functionality are involved, these must have DTMs in the FAS, with a communication interface for each channel.

[Figure 53](#) demonstrates data transfer from the device DTM to the AC 800M controller. The slave DTM can be a DTM for CI840/CI801 (S800 I/O) or for CI920 (S900 I/O). I/O units AI895, AO895, AI 930 and AO 930 also require DTMs.

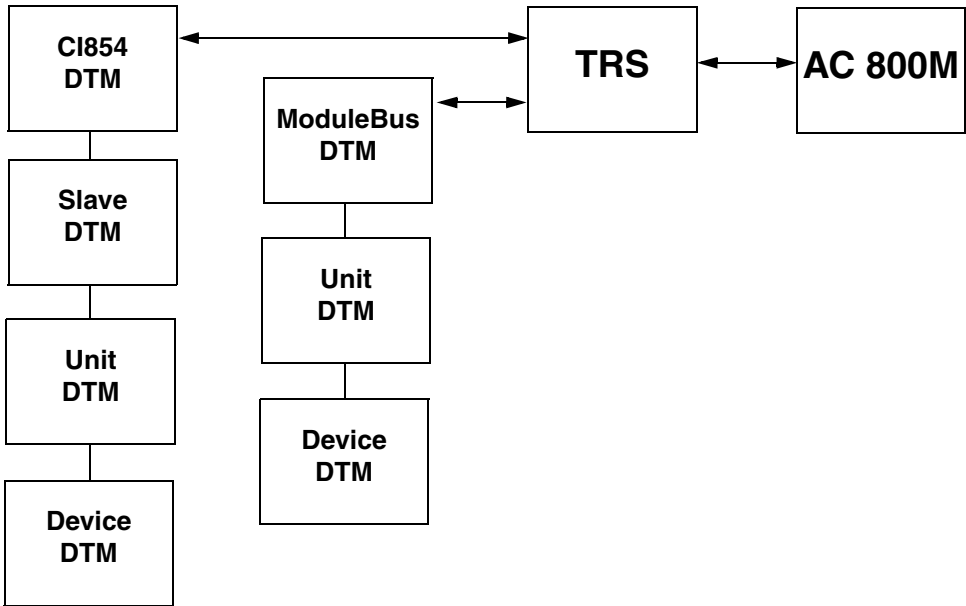


Figure 53. Nested communication.

Appendix D PROFIBUS PA

PROFIBUS PA

PROFIBUS PA (Process Automation) uses physical media according to IEC 61158-2. This standard has a much slower transmission speed (31.25 kbit/s) in order to obtain a more quiet communication. It is used for process automation by means of function blocks, often in environments where there is a considerable risk of explosion.

A subset of PROFIBUS PA (cyclic services) can be controlled from Control Network if a segment coupler (such as a repeater or DP/PA coupler) is used as a gateway between PROFIBUS DP and PROFIBUS PA.

Another way to link PROFIBUS DP and PROFIBUS PA is to use the Linking Device LD 800P, see [Figure 54](#). The LD 800P Linking Device converts the physical bus characteristics of the RS 485 interface for PROFIBUS DP into PROFIBUS PA physical bus characteristics MBP according to IEC 61158-2. Please refer to the manuals *PROFIBUS DP, Wiring and Installation (3BDS009029*)* and *PROFIBUS DP/PA Linking Device LD 800P, (3BDD011704*)* for more information about the LD 800P.

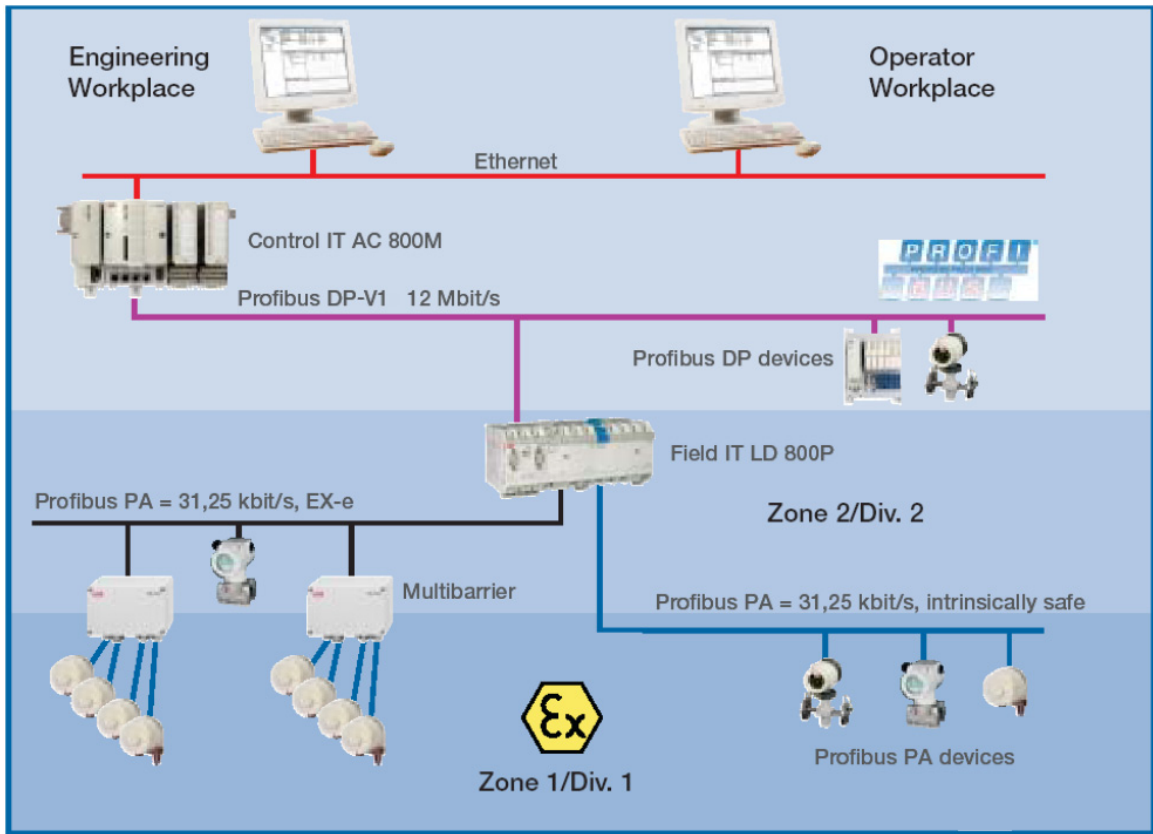


Figure 54. Linking PROFIBUS DP and PROFIBUS PA.

Appendix E ABB Drives

Introduction

ABB Standard Drives and ABB Engineered Drives can be connected to AC 800M in the following ways:

- ModuleBus optical link (not electrical)
- PROFIBUS DP-V1 via CI854-CI830, CI801 (configuration similar to ModuleBus, standard drive only)
- PROFIBUS DP-V1 via NPBA-12, RPBA-01, or FPBA-01
- PROFINET IO via RETA-02 or FENA-11
- DriveBus via CI858 (configuration almost similar to ModuleBus)

For more information regarding ABB Standard Drives and ABB Engineered Drives, see vendor documentation. See also [Section 16, DriveBus](#).

Types of ABB Drives

The ABB Standard Drives and ABB Engineered Drives families comprise the following types of drives and the applications they are directed to.

ABB Standard Drives

Table 25.

Name	Application
ACS400	Standard drive
ACS600	Crane application
ACS600	Pump and fan application
ACS600	Standard application
ACS800	Crane application
ACS800	Pump and fan application
ACS800	Standard application
DCS400	Standard drive
DCS500	Standard drive

ABB Engineered Drives

Table 26.

Name	Application
ACS600	IGBT supply (ISU) application
ACS600	System application
ACS600AD	Asynchronous drive
ACS600C	Cyclo converter drive
ACS600SD	Synchronous drive

Table 26. (Continued)

Name	Application
ACS800	IGBT supply (ISU) application
ACS800	System application
ACS880	All-compatible drive
ACS1000	Standard drive
DCS600	System application

Parameter Group Configuration

The ABB drives units are identified in the controller by their respective cluster and position address on the ModuleBus.

To establish the communication between the ABB drives and AC 800M, at least the following parameter groups shall be considered to be reconfigured in the ABB drive systems.

Table 27.

Parameter Group	Parameter Name	Setting
10.1	Ext1 Strt/Stp Dir	COMM. MODULE (CW)
10.2	Ext2 Strt/Stp Dir	COMM. MODULE (CW)
10.3	Direction	Request
11.2	Ext1/Ext2 Select	COMM. MODULE (CW)
11.3	EXT REF1 Select	COMM. REF or FAST COMM
11.6	EXT REF2 Select	COMM. REF or FAST COMM
16.1	RUN Enable	Yes or COMM. MODULE (CW)

Table 27. (Continued)

Parameter Group	Parameter Name	Setting
16.4	Fault Reset Select	COMM. MODULE (CW)
30.18	COMM FLT Function	Fault
70.1	Channel 0 Addr	See Drives Addressing in the online help for Control Builder M
98.2	COMM. MODULE Link	Advant

For newer drives (like the ACS880 series), when used with PROFIBUS DP or PROFINET IO (via FENA-11), the start trigger must be set to "Level" instead of "Trig". To do this, from the Main Menu in the drive, browse to Parameter -> Complete list -> 20 Start/stop/direction. Then, set:

- 20.01 -> Ext1 commands -> **Fieldbus A**
- 20.02 -> Ext1 Start Trigger -> **Level**

Example

The following parameter groups define the type of data you receive from the drive. This is only an example and you may find other configuration that suits your purpose.

Table 28.

Parameter Group	Parameter Name	Setting
92.2	Main DS ACT1	102 (Speed) Max value 20000
92.3	Main DS ACT2	105 (Torque) Max value 10000
92.4	AUX DS ACT 3	305 (Fault word 1)
92.5	AUX DS ACT 4	308 (Alarm word 1)
92.6	AUX DS ACT 5	306 (Fault word 2)

INDEX

A

- ABB Drives 205
 - Configure communication 207
 - DriveBus 205
 - PROFIBUS DP 205
 - ABB Engineered Drives 206
 - ABB Standard Drives 206
 - AC 800M
 - Maximum bandwidth 46
 - Access modes 23
 - Adapters
 - CI840 166
 - CI920 166
 - Address space 39
 - Addressing
 - Explicit 38
 - Implicit 38
 - Advant Fieldbus 100 112
 - AF 100 111
 - Bus Master 118
 - Design 112
 - Online upgrade 119
 - Process Data Transfer 118
 - Services 112
 - Alarms
 - INSUM 86
 - Application load
 - COMLI 77
 - MMS 45
- B**
- Bandwidth
 - AC 800M 46

C

- Cable length
 - Siemens 3964R 91
- Cables
 - SattBus 81
- Channel
 - Drives 162
 - PC Tool 162
- CI854
 - Web interface 168
- CI871
 - Web Interface 174
- Class C
 - IP address 39
- Client/server 42
- Client/server method
 - FF HSE 152
- Clock synchronization 29
 - MB 300 71
- CN1 44
- CN2 44
- CNCP 29
- COMLI
 - Application load 77
 - Function blocks 77
 - Hardware 78
 - Limitations 76
 - Link to control system 78
 - Master 73
 - Message format 78
 - Message length 77
 - Multidrop 74
 - Over SattBus 79

- Performance 77
- Point-to-point 75
- Protocol 73
- Redundancy 76
- RS-232C 75, 78
- SattBus messages 79
- Services 73
- Transmission distance 77
- Transmission speed 77
- Communication
 - Multidrop 73
 - Point-to-point 73
- Communication Interface Module CI868
 - Design
 - IEC 61850 140
- Communication interfaces
 - CI840 166
 - CI854 166
 - CI855 67
 - CI857 83
 - CI858 157
 - CI920 166
- Communication Interruption
 - Online Upgrade 128
- Communication profile
 - FF H1 149
 - FF HSE 149
- Communication Variables 53
- Configuration examples
 - HART 200
- Configure
 - ABB Drive communication 207
 - CI854 168
 - CI871 174
 - DriveBus 160
 - HART devices 199
- Connection Examples 123
- Control Builder 35
- Control Network 19, 33, 42
 - Troubleshooting 51

- Controllers
 - Supported 22
- CPU
 - Redundancy 44

D

- Default gateway 48
- Design 122
 - AF 100 112
 - DriveBus 158
 - Ethernet/IP and DeviceNet 132
 - FF HSE 150
 - IAC 55
 - IEC 61850 140
 - INSUM 84
 - MB 300 68
 - MMS 35
 - MODBUS RTU 96
 - PROFIBUS DP 164
 - PROFINET IO 171
 - SattBus 80
 - Siemens 3964R 90
- Design Examples
 - IEC 61850 141
- Design examples
 - DriveBus 158
 - FF HSE 151
 - INSUM 85
 - MB 300 68
 - MODBUS RTU 96
 - MODBUS TCP 101
 - PROFIBUS DP 165
 - PROFINET IO 172
- DeviceNet 131
- diagnostics 147
- Direct addressing
 - SattBus 80
- DriveBus
 - ABB Drives 205
 - Configuration 160

- Design 158
- Design example 158
- Hardware 162
- Limitations 161
- Performance 161
- Protocol 157
- Redundancy 160
- Services 157

E

- Editor
 - Parameters 35
- EDS File 132
- Ethernet 33
 - IEEE 803.2 standard 43
 - ports 44
- Ethernet/IP and DeviceNet 131
- Explicit addressing 38

F

- FF H1
 - Communication profile 149
- FF HSE
 - Client/server method 152
 - Communication profile 149
 - Connection methods 152
 - Design 150
 - Design example 151
 - Hardware 155
 - Limitations 153
 - Performance 153
 - Protocol 149
 - Publisher/subscriber method 152
 - Redundancy 153
 - Troubleshooting 156
- Files
 - GSD 164
- FOUNDATION Fieldbus HSE
 - Protocol 149
- Function blocks

- COMLI 77
- INSUM 84
- MB 300 68 to 69
- MODBUS TCP 99
- Modem 190
- SattBus 80

G

- GSD File 171
- GSD file 164

H

- Hardware 162
 - DriveBus 162
 - FF HSE 155
 - INSUM 88
 - MB 300 71
 - MODBUS RTU 98
 - PROFIBUS DP 167
 - PROFINET IO 174
 - Siemens 3964R 91
- HART 164
 - Configuration example 200
 - Protocol 199
 - Toolrouting 201
- HART devices
 - Configure 199
- Hayes modem 190
- Hirschmann modem 189
- Hubs 43

I

- IAC 53
 - Communication Variables 53
 - Design 55
 - Limitations 62
 - Performance 60
 - Services Provided 55
 - Troubleshooting 64
- IEC 1158-2

- PROFIBUS DP 163
- IEC 61131-5 95
- IEC 61158-2 203
- IEC 61850 139
 - Design 140
 - Communication Interface Module
 - CI868 140
 - Design Examples 141
 - Online Upgrade 144
 - Redundancy 143
 - Services Provided 139
 - Troubleshooting 148
- IEEE 803.2 standard 43
- Implicit addressing 38 to 39
- INSUM
 - Alarms 86
 - CI857 83
 - Design 84
 - Design example 85
 - Function blocks 84
 - Hardware 88
 - Limitations 87
 - Performance 87
 - Protocol 83
 - Services 84
 - Troubleshooting 88
- Integers
 - Siemens 3964R 92
- IP 169
- IP address 35
 - Class C 39
 - Remote 39
- IP subnet mask 35
- IRT 170
- ISO 9506 33
- Isochronous Real-Time 170

L

- LD800 DN 131
- Libraries

- Serial Communication Library 185
- Limitations
 - COMLI 76
 - DriveBus 161
 - FF HSE 153
 - IAC 62
 - INSUM 87
 - MB 300 71
 - MODBUS RTU 98
 - MODBUS TCP 106, 173
 - Modem 192
 - PROFIBUS DP 166
 - Redundancy 47
 - SattBus 80
 - Siemens 3964R 91
- Link
 - Control system to COMLI 78

M

- Master
 - COMLI 73
 - MODBUS RTU 96
 - PROFIBUS DP 166
- MasterBus 300 67
- Maximum message size 23
- Maximum number of variables
 - Per MMS telegram 47
- MB 300
 - Clock synchronization 71
 - Design 68
 - Design example 68
 - Function blocks 68 to 69
 - Hardware 71
 - Limitations 71
 - Performance 71
 - Protocol 67
 - Redundancy 71
 - Services 67
- MB 300 Clock Sync 29
- Message format

- COMLI 78
 - Message length
 - COMLI 77
 - MMS 45
 - MMS 33
 - Application load 45
 - Design 35
 - Maximum no of variables per telegram 47
 - Message length 45
 - OSI profile 193
 - Performance 45
 - Telegram size 47
 - Transmission speed 45
 - Troubleshooting 51
 - MMS Server 34
 - MMS Time Service 29
 - MOD5-to-MOD5 Communication Protocol 121
 - MODBUS RTU
 - Design 96
 - Design example 96
 - Hardware 98
 - Limitations 98
 - Master 96
 - Multi-drop 97
 - Performance 98
 - Point-to-point 96
 - Protocol 95
 - Redundancy 98
 - RS-232C 97
 - Services 95
 - Slave 96
 - Troubleshooting 98
 - MODBUS TCP
 - Hardware 109
 - Limitations 106, 173
 - On line upgrade 105
 - Performance 107
 - Redundancy 109
 - Modem
 - Dial-up 190
 - Function blocks 190
 - Hayes 190
 - Limitations 192
 - Performance 192
 - Protocol 189
 - Short distance 189
 - Troubleshooting 192
 - US Robotics modem 190
 - Westermo 190
 - Module Redundancy 126
 - Multi-drop
 - MODBUS RTU 97
 - Multidrop 73
 - COMLI 74
- N**
- Named SattBus 80
 - Network area 19, 36
- O**
- Online Upgrade
 - IEC 61850 144
 - Online upgrade 128
 - OPC Server 34
 - OSI implementation 195
 - OSI profile 193
- P**
- Parameters
 - Editor 35
 - Parity 191
 - PC Tool 162
 - Performance
 - COMLI 77
 - DriveBus 161
 - FF HSE 153
 - IAC 60
 - INSUM 87
 - MB 300 71
 - MMS 45

- MODBUS RTU 98
 - Modem 192
 - PROFIBUS DP 166
 - SattBus 81
 - Siemens 3964R 91
 - Plant intranet 42
 - Point-to-point 73
 - COMLI 75
 - MODBUS RTU 96
 - PPP 38
 - PROFIBUS 170
 - PROFIBUS DP 163
 - ABB Drives 205
 - Design 164
 - Design example 165
 - Hardware 167
 - IEC 1158-2 163
 - Limitations 166
 - Master 166
 - Performance 166
 - Protocol 163
 - Redundancy 166
 - RS-485 163, 167
 - Services 164
 - Slave 166
 - PROFIBUS PA
 - Protocol 203
 - PROFIBUS-DP
 - Troubleshooting 167
 - PROFINET IO 169
 - Design 171
 - Design examples 172
 - Hardware 174
 - On line upgrade 173
 - Redundancy 172
 - Services 172
 - Troubleshooting 174
 - Project Explorer 35
 - Protocols
 - COMLI 73
 - DriveBus 157
 - FF HSE 149
 - FOUNDATION Fieldbus HSE 149
 - HART 199
 - INSUM 83
 - MB 300 67
 - MOD5-to-MOD5 121
 - MODBUS RTU 95
 - Modem 189
 - PPP 38
 - PROFIBUS DP 163
 - PROFIBUS PA 203
 - SattBus 79
 - Serial 175, 181, 185
 - Siemens 3964R 89
 - Supported 22
 - Publisher/subscriber method
 - FF HSE 152
- R**
- Real-Time 170
 - Redundancy
 - COMLI 76
 - CPU 44
 - DriveBus 160
 - FF HSE 153
 - IEC 61850 143
 - Limitations 47
 - MB 300 71
 - MODBUS RTU 98
 - PROFIBUS DP 166
 - SattBus 80
 - Remote IP address 39
 - RNRP 19, 35, 44
 - RNRP monitor 51
 - Routers 36, 43
 - RS-232C 33, 38
 - COMLI 75, 78
 - MODBUS RTU 97
 - RS-485

PROFIBUS DP 163, 167
RT 169

S

SattBus
 Cables 81
 Design 80
 Direct addressing 80
 Function blocks 80
 Limitations 80
 Named 80
 Performance 81
 Protocol 79
 Redundancy 80
 Services 79, 99, 112
SattCon 73
Serial protocol 175, 181, 185
Server 42
Services
 COMLI 73
 DriveBus 157
 EtherNet/IP and DeviceNet 131
 IAC 55
 INSUM 84
 MOD5-to-MOD5 121
 MODBUS RTU 95
 PROFIBUS DP 164
 PROFINET IO 172
 SattBus 79, 99, 112
 Siemens 3964R 89
Services Provided
 IEC 61850 139
Setup Wizard 35
Siemens 3964R
 Cable lengths 91
 Design 90
 Hardware 91
 Integers 92
 Limitations 91
 Performance 91

 Protocol 89
 Services 89
SIMATIC 92
Slave
 MODBUS RTU 96
 PROFIBUS DP 166
SNTP 29
Subnetwork 36
Supported
 Controllers 22
 Protocols 22
Switches 43

T

TCP 181
 Design 181
 Hardware 182
 Limitations 182
 Performance 182
TCP/IP 34, 169
Telegram size
 MMS 47
Time synchronization 29
Toolrouting
 HART 201
Transceiver 43
Transmission distance
 COMLI 77
Transmission speed
 COMLI 77
 MMS 45
Troubleshooting 129
 Control Network 51
 FF HSE 156
 IEC 61850 148
 INSUM 88
 MMS 51
 MODBUS RTU 98
 Modem 192
 PROFIBUS-DP 167

PROFINET IO 174
Twisted-pair cable
 Shielded 167

U

UDP 81, 175
 Design 176
 Hardware 178
 Limitations 178
 Performance 178
UDP/IP 169
US Robotics 190
User Datagram Protocol 81

V

Variable types 23

W

Web interfaces
 CI854 168
 CI871 174
Westermo modem 189 to 190

Contact us

ABB AB

Control Technologies

Västerås, Sweden

Phone: +46 (0) 21 32 50 00

e-mail: processautomation@se.abb.com

www.abb.com/controlsystems

ABB Automation GmbH

Control Technologies

Mannheim, Germany

Phone: +49 1805 26 67 76

e-mail: marketing.control-products@de.abb.com

www.abb.de/controlsystems

ABB S.P.A.

Control Technologies

Sesto San Giovanni (MI), Italy

Phone: +39 02 24147 555

e-mail: controlsystems@it.abb.com

www.abb.it/controlsystems

ABB Inc.

Control Technologies

Wickliffe, Ohio, USA

Phone: +1 440 585 8500

e-mail: industrialitsolutions@us.abb.com

www.abb.com/controlsystems

ABB Pte Ltd

Control Technologies

Singapore

Phone: +65 6776 5711

e-mail: processautomation@sg.abb.com

www.abb.com/controlsystems

ABB Automation LLC

Control Technologies

Abu Dhabi, United Arab Emirates

Phone: +971 (0) 2 4938 000

e-mail: processautomation@ae.abb.com

www.abb.com/controlsystems

ABB China Ltd

Control Technologies

Beijing, China

Phone: +86 (0) 10 84566688-2193

www.abb.com/controlsystems

Copyright © 2003-2013 by ABB.

All rights reserved.

3ESE035982-511

Power and productivity
for a better world™

