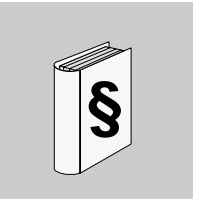
Modicon M340 Using Unity Pro S Architectures and Communication Services Reference Manual

November 2006 eng

35013341 00

Table of Contents

Safety Information



Important Information

NOTICE

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

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

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

⚠ DANGER

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

⚠ WARNING

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

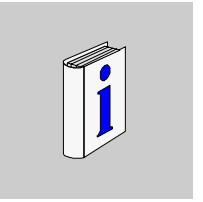
⚠ CAUTION

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

PLEASE NOTE

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

About the Book



At a Glance

Document Scope

This manual presents the architectures and communication services associated with Schneider PLCs programmed using Unity Pro S.

Validity Note

The data and illustrations found in this documentation are not binding. We reserve the right to modify our products in line with our policy of continuous product development.

The information in this document is subject to change without notice and should not be construed as a commitment by Schneider Electric.

Product Related Warnings

Schneider Electric assumes no responsibility for any errors that may appear in this document. If you have any suggestions for improvements or amendments or have found errors in this publication, please notify us.

No part of this document may be reproduced in any form or by any means, electronic or mechanical, including photocopying, without express written permission of Schneider Electric.

All pertinent state, regional, and local safety regulations must be observed when installing and using this product.

For reasons of safety and to ensure compliance with documented system data, only the manufacturer should perform repairs to components.

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

Failure to observe this product related warning can result in injury or equipment damage.

User Comments

We welcome your comments about this document. You can reach us by e-mail at techpub@schneider-electric.com

Introduction to the Communication Application

I

At a Glance

Subject of this Part

This part gives an overview of the communication application: the types of networks and buses, services and architectures available.

What's in this Part?

This part contains the following chapters:

General

1

At a Glance

Subject of this Chapter

This chapter gives an overview of the different characteristics of the communication application.

What's in this Chapter?

This chapter contains the following topics:

Introduction to the Application-Specific Communication Function

At a Glance

The application-specific communication function makes it possible to exchange data between different devices connected to a bus or a network.

This function applies to:

- Processors with an Ethernet, a built-in Fipio or a CANopen link,
- Specific rack-mounted communication modules,
- The terminal port of a processor,
- PCMCIA cards of a rack-mounted processor or module.

Communication Types

The different types of communication are:

- Ethernet TCP/IP or Ethway Network
- Fipway Network
- Modbus Plus Network
- Fipio bus (manager and agent)
- Uni-Telway bus
- Modbus/JBus bus
- Character mode serial link
- CANopen field bus
- Interbus field bus
- Profibus field bus
- USB-standard fast terminal port

Available Services

The available services can be classified into three categories:

- Explicit messaging (see *Messaging Service, p. 21*) services:
  - Modbus messaging
  - UNI-TE messaging
  - Telegrams
- Implicit database access services:
  - Global data (See Modicon M340, Premium, Atrium and Quantum Using Unity Pro, Communication Services and Architectures, Global Data Service),
  - Common words (See Modicon M340, Premium, Atrium and Quantum Using Unity Pro, Communication Services and Architectures, Fipway Common Words and Shared Tables),
  - Shared tables (See Modicon M340, Premium, Atrium and Quantum Using Unity Pro, Communication Services and Architectures, Fipway Common Words and Shared Tables).
- Implicit Input/Output management services:
  - I/O scanning (See Modicon M340, Premium, Atrium and Quantum Using Unity Pro, Communication Services and Architectures, IO Scanning Service),
  - Peer cop (See Modicon M340, Premium, Atrium and Quantum Using Unity Pro, Communication Services and Architectures, Peer Cop Service).

Characteristics of the Different Service Types

The following table gives an overview of the main characteristics of the types of services mentioned above:

Type of service	These services make it possible...	They are used...
Messaging services	for a device (Client) to send a message to another device (Server) and obtain a response without having to program anything into the server device.	to access data from time to time.
Implicit database access services	to share data which is refreshed automatically and on a regular basis.	to synchronize applications or to transparently obtain real time images of a system on several remote PLCs.
Implicit I/O management services	to transparently and automatically manage remote I/Os on a network.	to monitor a set of distributed systems across a network.

Summary of Communication Solutions

At a Glance

The services presented earlier in this chapter are available for certain types of communication.

For example, for messaging services, certain communication functions apply to networks, others to buses and others to serial links in character mode (See Modicon M340, Premium, Atrium and Quantum Using Unity Pro, Communication Services and Architectures, Characteristics of the Messaging Service Communication Functions).

Summary

The following table gives an overview of the different services available according to the types of communication:

Function	Fipway	Fipio	Uni-Telway	Character mode	Modbus /JBus	Modbus Plus	Ethway	TCP/IP	CANopen	USB
Messaging services										
Communication functions	The communication functions that can be used depend closely on the type of communication for which they are applied (See Modicon M340, Premium, Atrium and Quantum Using Unity Pro, Communication Services and Architectures, Characteristics of the Messaging Service Communication Functions).									
Implicit database access services										
Global Data	-	-	-	-	-	-	-	X	-	-
Common words	X	-	-	-	-	-	X	-	-	-
Shared tables	X	-	-	-	-	-	X	-	-	-
Implicit I/O management services										
I/O Scanning	-	-	-	-	-	-	-	X	-	-
Peer cop	-	-	-	-	-	X	-	-	-	-
Other	-	X	-	-	-	X	-	-	X	-
Legend:										
X	Yes									
-	No									

Services Available on Networks and Buses

2

At a Glance

Subject of this Section

This section describes the main services available on the communication buses and networks.

What's in this Chapter?

This chapter contains the following sections:

2.1 Global Data Service

Description of the Global Data Service

At a Glance

The aim of the **Global Data** service, which is supported by Ethernet modules, is to provide an automatic data exchange for the coordination of PLC applications. Data is shared according to an inter-device publication/subscription method.

How it Works

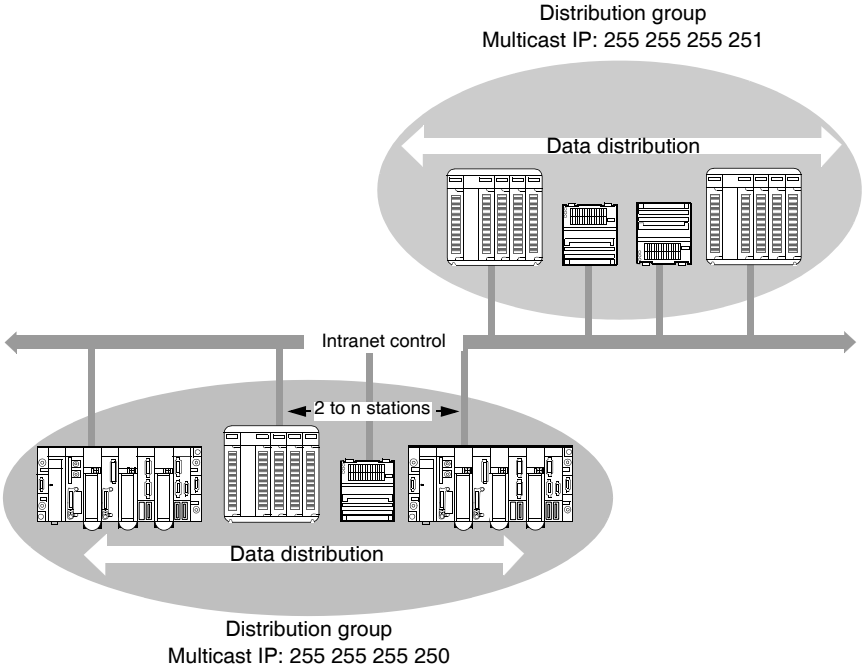
The communication modules are grouped into a **Distribution group**.

Each communication module publishes a local application variable for the other communication modules in the distribution group.

Each communication module can also subscribe to the application variables published by all other modules belonging to the distribution group.

The **Global Data** service should be configured to determine the location and the number of application variables of each communication module. Once the modules have been configured, exchanges between communication modules belonging to the same group are automatically carried out when the PLC is in RUN mode.

Illustration:



A **Distribution group** is a group of communication modules identified by the same **multicast IP** address. "Multicasting" exchanges are used to distribute **Global Data**. Several independent distribution groups can co-exist on the same sub-network with their own multicast address.

A Publication/Subscription protocol on UDP/IP is used for data distribution.

Limitations

There is no theoretical limit to the number of stations that may belong to a distribution group. The main limitation is the number of variables exchanged in a group (64 variables).

2.2 IO Scanning Service

Description of the IO Scanning Service

At a Glance

The IO scanner makes it possible to periodically read or write to/from remote inputs/outputs on the Ethernet network without requiring any specific programming.

This service comprises the following essential elements:

- A read field containing all the values of the remote inputs,
- A write field containing all the values of the remote outputs,
- Scanning periods independent of the PLC cycle and dedicated to checking each remote device.

How it Works

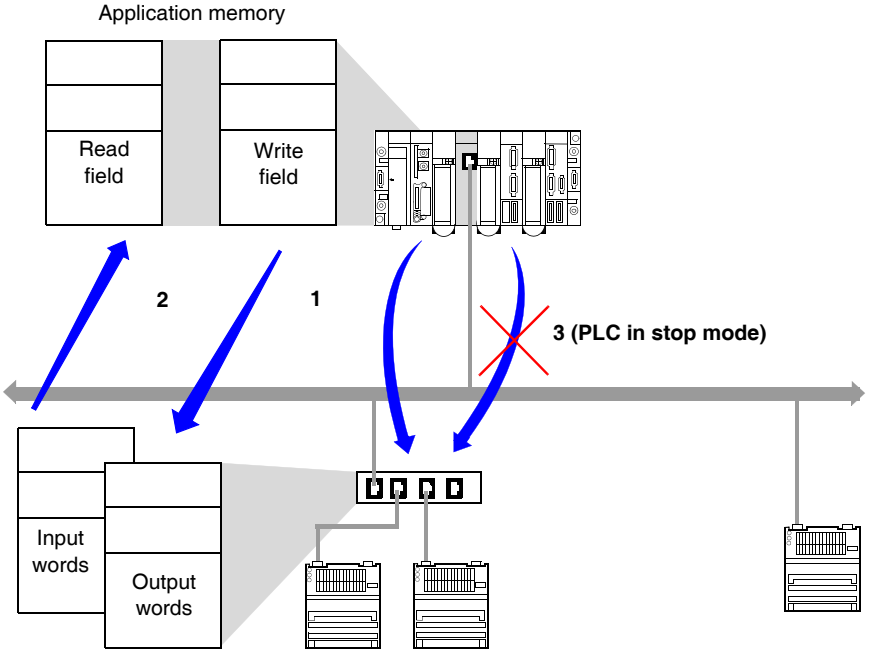
The scan will only be performed if the PLC is in Run mode.

This service works with all devices supporting Modbus communication on the TCP/IP profile in server mode.

The exchange mechanism, which is transparent for users, involves:

- Read requests
- Write requests
- Read and write requests

The following diagram shows how scanning of remote inputs/outputs works.



1. As soon as the PLC goes into Run mode, the module opens one connection per scanned device.
2. The module then periodically reads the input words and periodically writes the output words of each device.
3. If the PLC goes into Stop mode, the connections with each device are closed.

Summary of the Functions

The functions of the IO scanning service are:

- Management of the connection with each remote device (one connection per scanned device),
- Scanning of the inputs/outputs of the device by using the Modbus read/write requests on the TCP/IP profile,
- Update of the read/write fields in the application memory,
- Refreshing the status bits of each remote device.

Note: The status bits indicate whether the input/output words of the module have been refreshed.

2.3 Messaging Service

At a Glance

Subject of this Section

This section gives an overview of the messaging service available on Schneider PLCs.

What's in this Section?

This section contains the following topics:

Messaging Service

At a Glance

The messaging service makes it possible to perform inter-PLC data exchanges using communication functions.

Two types of messaging are used:

- Private: UNI-TE on Premium and Télémécanique installed base.
- Standard: Modbus on Quantum / Premium / Modicon M340 and Modicon installed base.

The destination entities of an exchange can either be located in a local station or in a remote station on a communication channel or directly in the CPU.

The communication functions provide an interface that is independent of the location of the destination entity. Furthermore, they mask the coding of the communication requests from the user. They thus guarantee compatibility of communication between Premium, Micro, Quantum, TSX 40, TSX 17, 1000 series and Modicon M340 PLCs.

Synchronous/ Asynchronous Communication

A communication function is said to be synchronous when it is wholly executed during the PLC task which activated it.

A communication function is said to be asynchronous when it is executed during one or more PLC tasks after the one which activated it.

Characteristics of the Messaging Service Communication Functions

At a Glance

These functions (See Unity Pro, Communication Block Library, Extended) enable communication between one device and another. Certain functions are common to several types of communication channel. Others may be specific to one communication function.

Note: Processing of communication functions is asynchronous in relation to the processing of the application task which allowed them to be activated. The send/receive telegram and stop operation functions are the only exceptions, as their execution is totally synchronous with the execution of the activation task.

Note: It is recommended that asynchronous functions be triggered on edge and not on state so as to avoid sending several identical requests in quick succession, thus saturating the communication buffers.

Communication Functions on Premium

The following table gives an overview of Premium communication functions:

Function (asynchronous)	Its role is...
READ_VAR	To read standard language objects: words and bits using UNI-TE or Modbus.
WRITE_VAR	To write standard language objects: words and bits using UNI-TE or Modbus.
SEND_REQ	To send UNI-TE requests.
DATA_EXCH	To send and/or request receipt of data.
PRINT_CHAR	To write a character string.
INPUT_CHAR	To read a character string.
OUT_IN_MBUS	To transmit a Modbus request and await its report.
OUT_IN_CHAR	To send a character string and await a response.
UNITE_SERVER	To process READ_VAR and WRITE_VAR requests immediately on Modbus (Immediate server).
READ_GDATA	To read common Modbus Plus data.
WRITE_GDATA	To write common Modbus Plus data.
READ_Asyn	To read 1K of messaging.
WRITE_Asyn	To write 1K of messaging.
Function (synchronous)	
SEND_TLG	To send a telegram.
RCV_TLG	To receive a telegram.
CANCEL	To stop an exchange in progress.

Communication Functions on Quantum

The following table gives an overview of Quantum communication functions:

Function	Its role is...
CREAD_REG	To read contiguous registers.
CWRITE_REG	To write contiguous registers.
ModbusP_ADDR	To define a MSTR Modbus Plus address.
SYMAX_IP_ADDR	To define a MSTR Symax address.
TCP_IP_ADDR	To define a MSTR TCP/IP address.
MBP_MSTR	To perform operations on Modbus Plus.
XMIT	To process Modbus master messages and character strings.
ICNT	To connect to and disconnect from an IB-S communication.
ICOM	To transfer data with an IB-S slave.

Communication Functions on Modicon M340

The following table gives an overview of Modicon M340 communication functions:

Function	Its role is...
ADDM	To convert a character string into an address that can be used directly by the communication functions READ_VAR, WRITE_VAR, INPUT_CHAR and PRINT_CHAR.
INPUT_BYTE	To send an array of read request bytes.
READ_VAR	To read standard language objects: words and bits using UNI-TE or Modbus.
WRITE_VAR	To write standard language objects: words and bits using UNI-TE or Modbus.
PRINT_CHAR	To write a character string.
INPUT_CHAR	To read a character string.

Availability of Functions According to Protocols

The following table lists the protocols that support the communication functions:

Function	Fipway	Fipio	Uni-Telway	Character mode	Modbus	Modbus Plus	TCP/IP	ETHWAY	CANopen
Premium									
READ_VAR	X	X	X	-	X	X	X	X	-
WRITE_VAR	X	X	X	-	X	X	X	X	-
SEND_REQ	X	X	X	-	X	X	X	X	X
DATA_EXCH	X	-	X	-	-	-	X	X	-
PRINT_CHAR	X	-	-	X	-	-	X	X	-
INPUT_CHAR	X	-	-	X	-	-	X	X	-
OUT_IN_MBUS	-	-	-	-	X	X	-	-	-
OUT_IN_CHAR	X	-	-	X	-	-	X	X	-
SEND_TLG	X	-	-	-	-	-	-	-	-
RCV_TLG	X	-	-	-	-	-	-	-	-
READ_GDATA	-	-	-	-	-	X	-	-	-
WRITE_GDATA	-	-	-	-	-	X	-	-	-
UNITE_SERVER	-	-	-	-	X	-	-	-	-
WRITE_Asyn	-	-	-	-	-	-	X	-	-
READ_Asyn	-	-	-	-	-	-	X	-	-

Function	Fipway	Fipio	Uni-Telway	Character mode	Modbus	Modbus Plus	TCP/IP	ETHWAY	CANopen
Quantum									
CREAD_REG	-	-	-	-	-	-	-	-	-
CWRITE_REG	-	-	-	-	-	-	-	-	-
ModbusP_ADDR	-	-	-	-	-	-	-	-	-
SYMAX_IP_ADDR	-	-	-	-	-	-	-	-	-
TCP_IP_ADDR	-	-	-	-	-	-	-	-	-
MBP_MSTR	-	-	-	-	X	X	X	-	-
XMIT	-	-	-	-	X	X	X	-	-
ICNT	-	-	-	-	X	X	X	-	-
Modicon M340									
READ_VAR	-	-	X	X	X	-	X	-	X
WRITE_VAR	-	-	X	X	X	-	X	-	X
ADDM	-	-	X	X	X	-	X	-	X
INPUT_BYTE	-	-	-	X	-	-	-	-	-
PRINT_CHAR	-	-	-	X	-	-	-	-	-
INPUT_CHAR	-	-	-	X	-	-	-	-	-
Legend:									
X	Yes								
-	No								

Communication Architectures

3

At a Glance

Subject of this Chapter

This chapter gives an overview of the different communication architectures.

What's in this Chapter?

This chapter contains the following topics:

Global Architecture

At a Glance

Schneider has a communications strategy based on open standards (core of the range) such as:

- Ethernet Modbus TCP/IP
- CANOpen
- AS-Interface
- Modbus Link series

This has not always been the case and there are a significant number of installed bases on networks or proprietary buses such as: Modbus +, Fipway, Ethway, X-way on TCP/IP, Fipio, Symax and Unitelway.

Schneider offers a connectivity range for the main standards available on the market through its Profibus, Interbus, and TCPopen ranges.

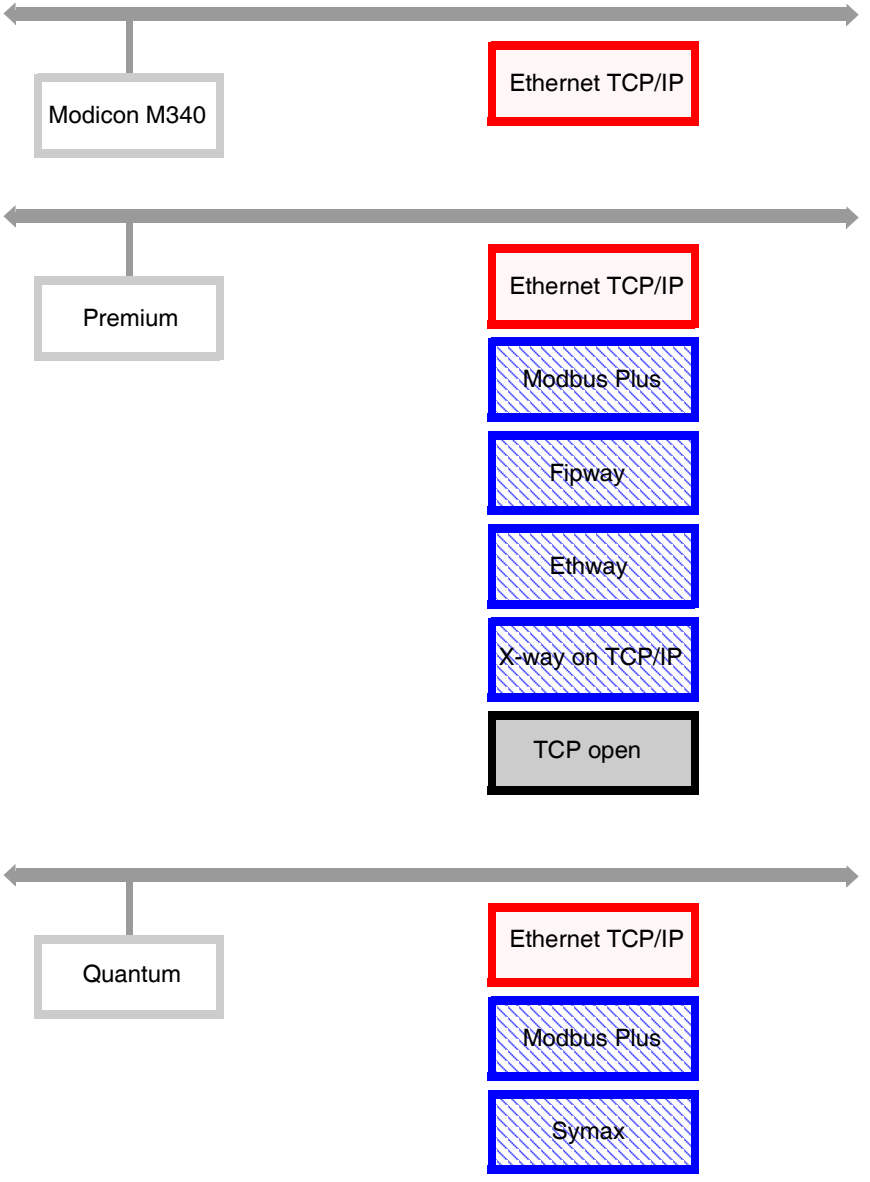
The possible and recommended communication architectures are presented in the following pages, according to the type of PLC used:

- At level 2: Inter-PLC network,
- At level 1: Field Bus.

The communication solutions for existing installations, from the Télémécanique or Modicon ranges, are then presented.

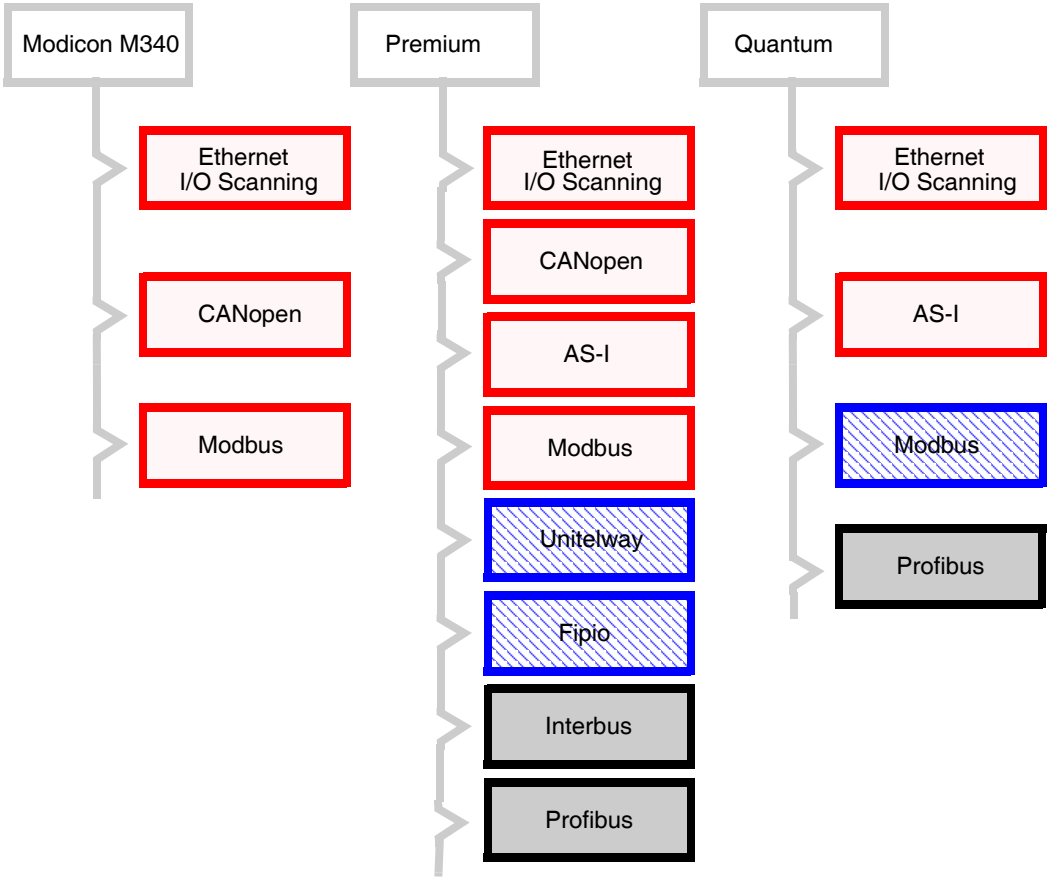
Network Architecture

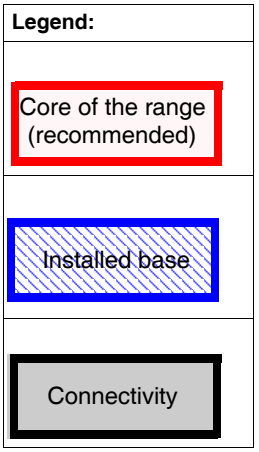
The network architectures available for Modicon M340, Premium and Quantum PLCs are summarized below:



Field Bus Architecture

The field bus architectures available for Modicon M340, Premium and Quantum PLCs are summarized below:

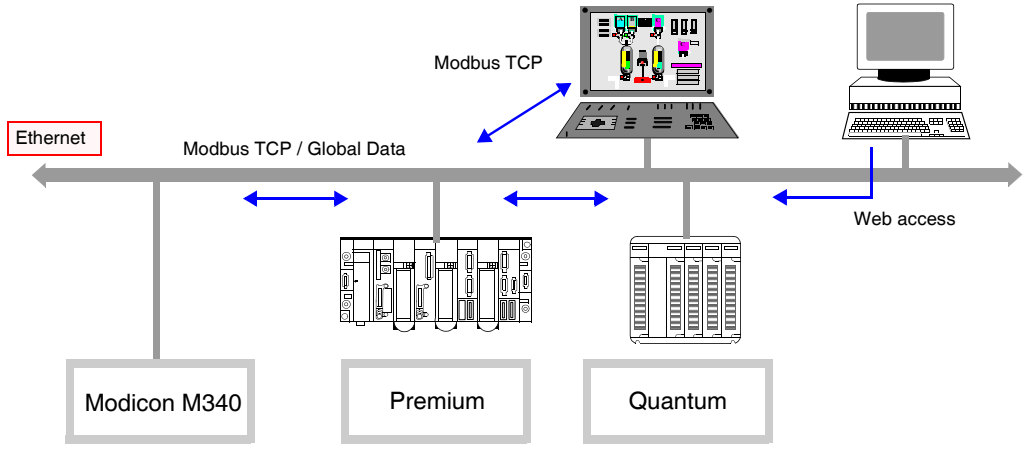



Legend:
Core of the range (recommended)
Installed base
Connectivity

Preferred Network Architecture

At a Glance

The Ethernet network architecture presented below is Schneider's so-called "preferred" solution.



Main services:

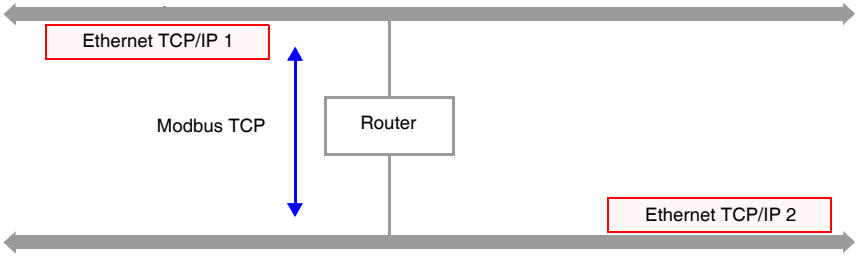
- Global Data: inter-PLC synchronization.
- Modbus TCP/IP: client/server device for accessing automation variables.
- Web Access: access to the variables and diagnostics from a standard workstation.

Depending on the type of device, other services may also operate simultaneously:

- SMTP: e-mail
- MTP: time distribution
- SNMP: network management
- FDR: faulty device replacement

IP Routers

As PLCs do not have IP routers, standard routers must be used to connect two Ethernet TCP/IP networks. The following diagram shows the communication between two Ethernet networks:



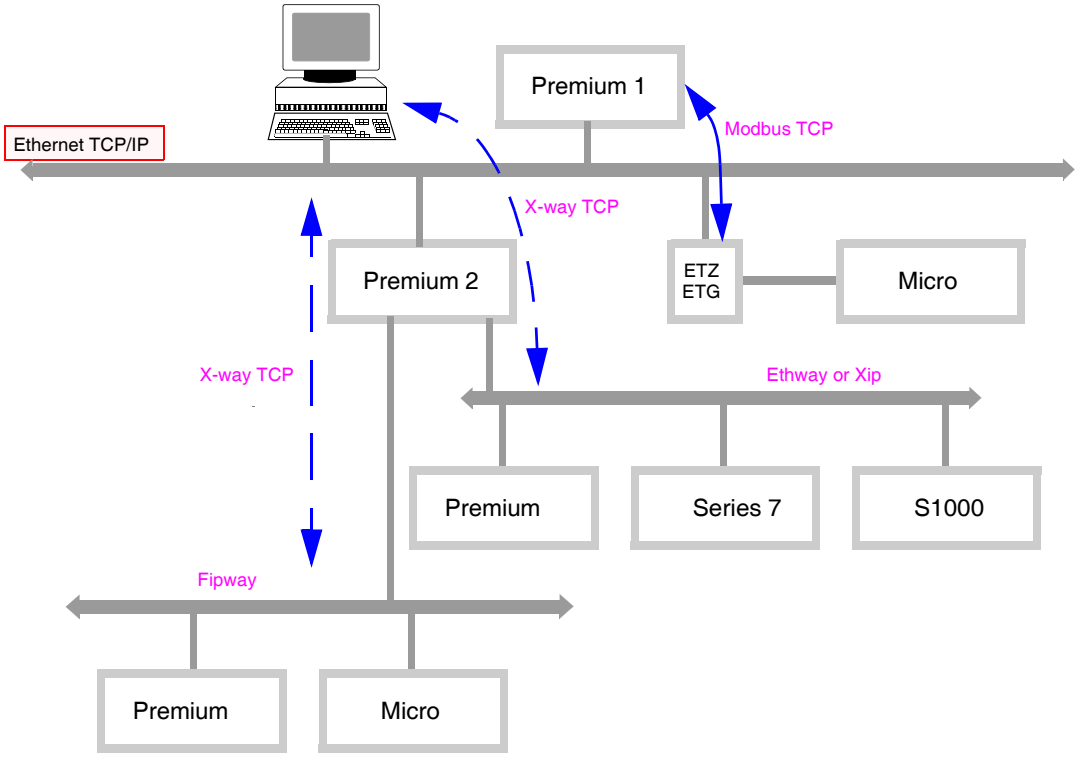
Communication with the Installed Base

"Télémécanique" Architecture

The Ethway, Xip, Fipway, Uni-telway and Fipio installed bases use the Télémécanique Uni-te private messaging protocol on a network layer called X-way.

This layer guarantees transparent routing of Uni-te messages between each of these networks. Only Premium and Micro PLCs support this protocol.

In the architecture below, transparency may be achieved by configuring the Premium 2 PLC as a Bridge, and provided the Premium 1 PLC or the Unity terminal uses the Xip protocol (X-way on TCP/IP).

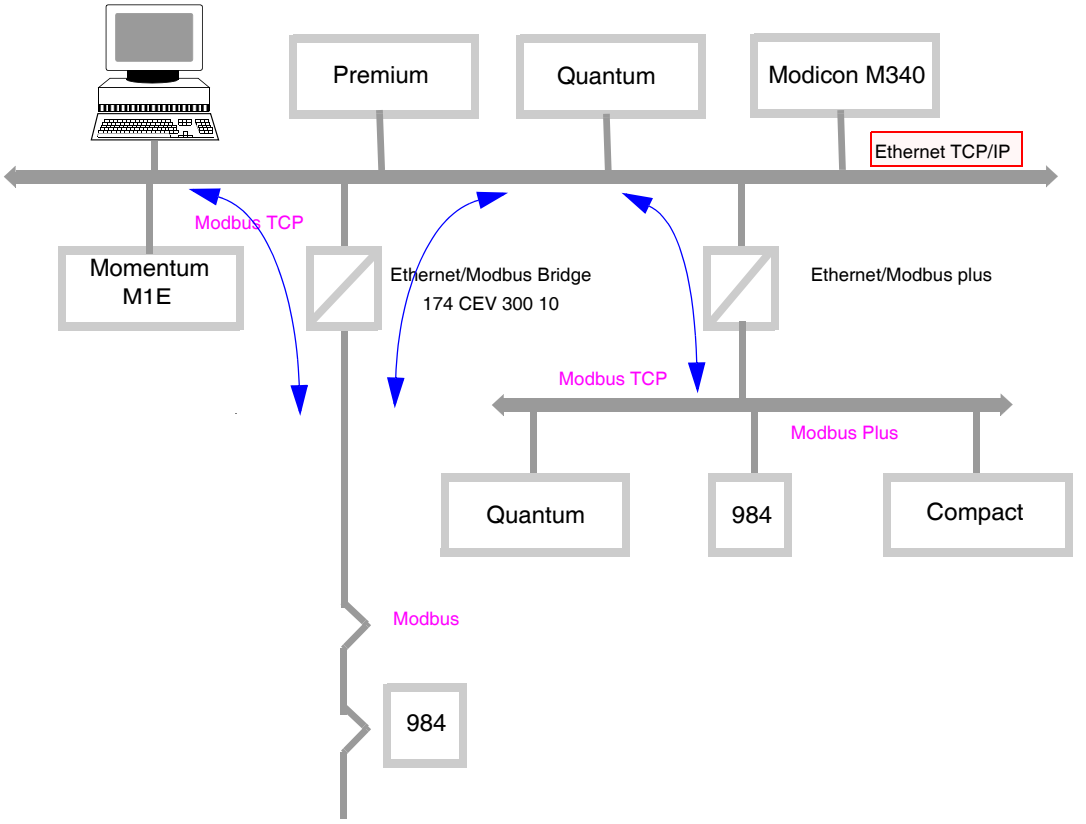


"Modicon" Architecture

The Modicon installed base uses the standard Modbus protocol on serial link or token bus.

It is not possible to perform routing using this protocol.

This is, however, possible using gateways or bridges.



Interoperability

4

List of Modbus Function Codes

At a Glance

Quantum, Premium and M340 PLCs have communication server kernels that accept the common Modbus function codes. These are listed in the table on this page.

As servers, Quantum, Premium and M340 PLCs recognize all **Class 0** and **Class 1** Modbus function codes, as stipulated in the Modbus specifications available at *http://www.Modbus.org*. Their server kernel also includes the function code 23 for reading/writing of consecutive variables.

For the list of Modbus function codes recognized by Quantum PLCs, please refer to the specific Quantum documentation.

For the list of function codes recognized by Premium PLCs, please refer to the specific Premium (See Premium and Atrium using Unity Pro, Asynchronous serial link, Modbus communication programming) documentation. In addition to this, Premium PLCs recognize certain UNI-TE (See Unity Pro, Communication Block Library, List of UNI-TE requests) requests.

List of Modbus Requests Recognized When Connected as a Server

The following table lists the function codes and the address of the Modbus function codes, recognized by Premium, Quantum and M340 platforms:

Function code	Quantum memory address	Premium memory address	M340 memory address	Meaning
1	16#0XXX	%MWi (0...)	%M	Read output bits
2	16#1XXX	%Mi (0...)	%M	Read input bits
3	16#4XXX	%MWi (0...)	%MW	Read consecutive integer values
4	16#4XXX	%MWi (0...)	%MW	Read consecutive integer variables in the input field
5	16#3XXX	%Mi (0...)	%M	Read a bit in the output memory field
6	16#0XXX	%MWi (0...)	%MW	Read an integer in the internal memory field
10	-	-	%MW	Write n output words
15	16#4XXX	%Mi (0...)	%M	Write n output bits
16	16#4XXX	%MWi (0...)	%MW	Write consecutive integer values
23	16#4XXX	%MWi (0...)	%MW	Read/write consecutive integer values

Note: The function codes 3 and 16 belong to **Class 0**. The others belong to **Class 1**, except for function code 23, which belongs to **Class 2**.

Use of Modbus Function Codes as a Client on Premium and M340

The table below lists the Modbus function codes and their use as a client on Premium, Quantum and M340 PLCs:

Function code	Quantum memory address	Premium memory address	M340 memory address	Modbus request	Communication function
1	-	%MWi (0...)	%M	Read output bits	READ_VAR
2	-	%Mi (0...)	%M	Read input bits	READ_VAR
3	16#4XXX	%MWi (0...)	%MW	Read consecutive integer values	READ_VAR
4	-	%MWi (0...)	%MW	Read consecutive integer variables in the input field	READ_VAR
15	-	%Mi (0...)	%M	Write n output bits	WRITE_VAR
16	16#4XXX	%MWi (0...)	%MW	Write consecutive integer values	WRITE_VAR

The way in which to use function codes with communication functions is described in the Modbus manual (See Premium and Atrium using Unity Pro, Asynchronous serial link, Modbus communication programming).

Note: Interoperability with Windows applications is provided in two ways:
- Access to the PLC variables can be easily achieved with OFS software.
- The application download function, import/export source format function and access to operating modes (RUN/STOP/INIT) functions can be performed using the UDE (Unity Development Edition) range.

Addressing

II

At a Glance

Subject of this Part

This part describes the different addressing solutions for devices on a communication bus or network.

What's in this Part?

This part contains the following chapters:

General Points Concerning Addressing

5

At a Glance

Subject of this Chapter

This chapter gives an overview of the different addressing solutions for devices in a communication architecture.

What's in this Chapter?

This chapter contains the following sections:

5.1 General

Description

At a Glance

In a communication architecture, each device must be identified by an address. This address is specific to each device, and enables the device initiating communication to determine the destination precisely. Similarly, for the configuration of services such as Global Data on Ethernet, the Peer Cop service on Modbus Plus or common words and shared tables on Fipway, these addresses make it possible to identify the stations that own different shared information.

Modicon M340 PLCs support 2 types of addressing depending on the type of device, network or bus used:

- IP addressing (see *IP Addressing, p. 61*),
- Modicon M340 PLCs addressing (see *Modicon M340 PLCs Addressing, p. 47*).

Modicon M340 PLC Addressing

6

At a Glance

Purpose of this Chapter

This chapter describes Modicon M340 PLC addressing and indicates its fields of application.

What's in this Chapter?

This chapter contains the following topics:

Modicon M340 Types of Communication Entities

At a Glance

There are different types of communication entities.

These exchanges are performed by the communication functions described in the Communication EF library.

It is possible to class addresses into 3 types:

- local addresses, identified by `r.m.c.SYS`, or more simply, `r.m.c`,
- remote addresses, to address a device (Modbus, CANopen or Ethernet) directly connected to the channel,
- broadcast addresses, depend on the network. For Modbus communication, broadcast address is obtained with the slave number set to 0. Note that a broadcast address can be used for all networks but requires that the communication channel supports broadcasting. This is not always the case.

SYS Keyword

`SYS` gives access to a local module or a channel server. `SYS` is used for character mode and can be ommitted.

Broadcast Addresses

Broadcast addresses depend on the destination devices:

Destination	Broadcast address
Broadcast to all Modbus slaves (the slave number equals 0)	rack.module.channel.0

Modicon M340 Addressing for a Communication Entity

At a Glance

With Modicon M340 PLCs, it is possible to address any Modicon M340 PLC communication channel and any device directly connected to a Modicon M340 PLC communication channel.

Each device is identified by a unique address, which consists of a device number or an IP address. The addresses then differ according to the protocol:

- Ethernet TCP/IP
- Modbus or CANopen
- Character Mode

Within a station, each communication entity is characterized by a topological address (access path) and a target entity.

Note: An address is expressed in the form of a character string. However, it can only be used in conjunction with the function `ADDM`, which is why the following notation will be used to describe an address: `ADDM('address string')`.

Modicon M340 addressing uses 3 concepts:

- The target entity depends on the communication EF and is chosen implicitly:
  - `MBS` for adressing a Modbus server,
  - `TCP.MBS` for adressing a TCP Modbus server,
  - `SYS` for adressing a channel server on Character mode. `SYS` can be ommitted.
- The communication channel is explicit (processor's or module's position and communication channel number) or symbolized with the Netlink name for Ethernet communication.
- The node address depends on the communication protocol:
  - IP address with Ethernet,
  - node address with CANopen,
  - slave address with Modbus.

Addressing a Station on Ethernet

The address of a station on Ethernet takes the form:

- `ADDM('Netlink{hostAddr}')`
- `ADDM('Netlink{hostAddr}TCP.MBS')`
- `ADDM('Netlink{hostAddr}MBS')`
- `ADDM('r.m.c{hostAddr}')`
- `ADDM('r.m.c{hostAddr}TCP.MBS')`
- `ADDM('r.m.c{hostAddr}MBS')`
- `ADDM('Netlink{hostAddr}node')`

Where:

- Netlink: network name set in the Net Link field of Ethernet channel
- hostAddr: IP address of device
- r: rack number (rack)
- c: channel number (channel)
- node: Modbus or CANopen node behind a gateway (gateway identified with hostAddr)

Note: If the netlink name is ommitted, the system takes the default netlink connection that is the closest link to the processor (usually the processor Ethernet channel).

Addressing a Device on a CANopen Bus

The address of a device on a CANopen bus takes the form `ADDM('r.m.c.e')`, where:

- r: rack number (rack)
- m: rack module position
- c: channel number (channel) of CANopen port (2)
- e: CANopen slave node (equipment) (range 1 to 127)

Addressing a Device on a Modbus

The address of a device on a Modbus bus takes the form `ADDM('r.m.c.e')`, where:

- r: rack number (rack)
- m: rack module position
- c: channel number (channel) of Modbus port (0)
- e: Modbus slave number (equipment) (range 1 to 247)

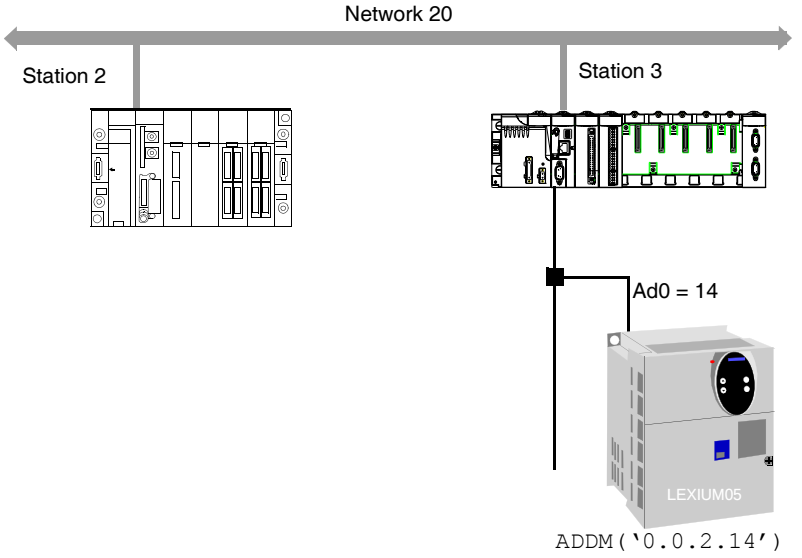
Addressing a Device in Character mode

To send or receive a character string, you can use `ADDM('r.m.c')` or `ADDM('r.m.c.SYS')`, where:

- r: rack number (rack)
- m: rack module position
- c: channel number (channel) of Character mode port (0)
- SYS: keyword used to stipulate the station server system (see *Modicon M340 Types of Communication Entities, p. 48*). `SYS` can be ommitted.

Example

The figure below describes the address of the servodrive. The example here shows slave 14 on channel 2 (CANopen) of the module in rack 0, slot 0:



Processor Communication Channels Addressing

At a Glance

Following are examples of the different types of addressing for a processor's communication channels.

The examples are based on a Modicon M340 type processor.

The modules have a topological address that is a function of the module's position in the rack.

The first two slots of the rack (marked PS and 00) are reserved for the rack's power supply module (BMX CPS ••••) and the processor (BMX P34 ••••) respectively.

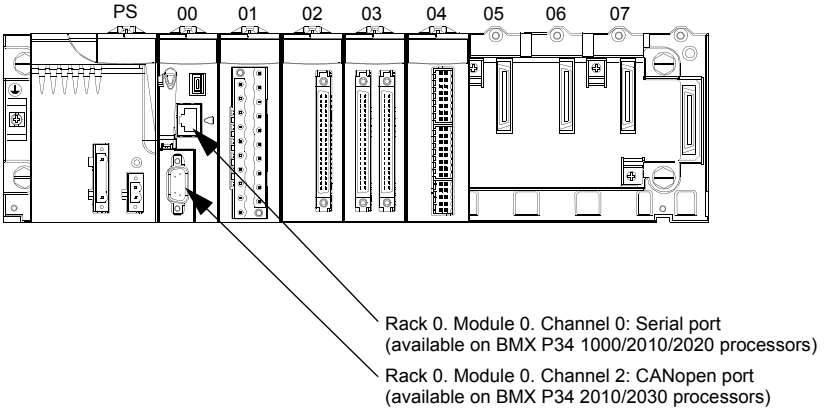
Available Communication Channels

The available communication channels vary depending on the processor:

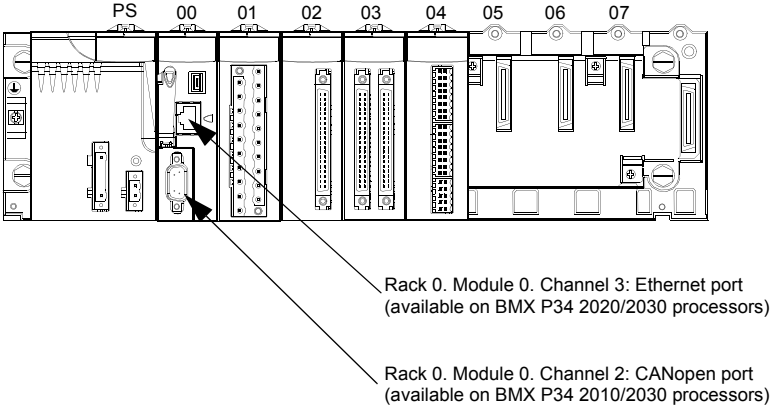
Processor	Integrated Modbus Connection	Integrated CANopen Master Connection	Integrated Ethernet Connection
BMX P34 1000	X	-	-
BMX P34 2010	X	X	-
BMX P34 2020	X	-	X
BMX P34 2030	-	X	X
Key			
X Available			
- Not available			

Processor Communication Channels Adressing

The diagram below shows an example of Modicon M340 configuration including a BMX P34 2010 processor and the addresses of the processor communication channels:



The diagram below shows an example of Modicon M340 configuration including a BMX P34 2030 processor and the addresses of the processor communication channels:



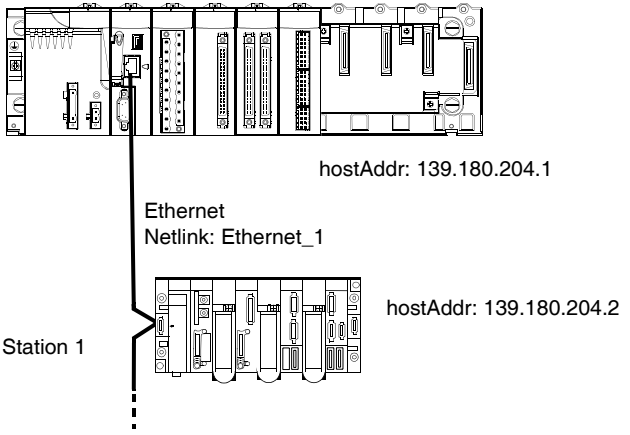
Example of Modicon M340 Ethernet Addressing

At a Glance

With this type of addressing, a station can access different station connected to logical network.

Connection via CPU Ethernet Port

A device with the IP address 139.180.204.2 is connected to the Ethernet network. It is the processor Ethernet port configured with Netlink name `Ethernet_1`.



Address settings station 1: `ADDM('0.0.3{139.180.204.2}')`

or Address settings station 1: `ADDM('Ethernet_1{139.180.204.2}')`

Example of Modicon M340 CANopen Addressing

At a Glance

With this type of addressing, a master station can access different slaves connected to CANopen bus.

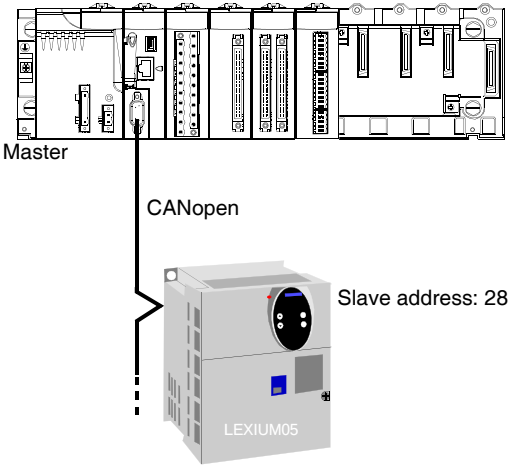
Addressing Rules

The syntax of CANopen addressing is `ADDM ('r.m.c.node')` . The meaning of the string parameter is as follows:

- r: rack address. The processor's rack address is always 0.
- m: module address. The Modicon M340 processor's slot number in the rack is always 0.
- c: channel address. The Modicon M340 CANopen port is always channel 2.
- node: slave number to which the request is being sent. The range for configured slave numbers is from 1 to 127.

Example

In the following example, the Modicon M340 processor's bus manager addresses the Lexium 05 device at connection point 28:



Address settings of slave 28: `ADDM('0.0.2.28')`.

Note: In addition to the address defined by `ADDM`, the `READ_VAR` and `WRITE_VAR` functions use another parameter `NUM`, which must be defined to address the SDO to be read or written.

Examples of Modicon M340 Modbus and Character Mode Addressing

At a Glance

The following examples deal with:

- Modbus adressing
- Character mode adressing

Modbus Addressing Rules

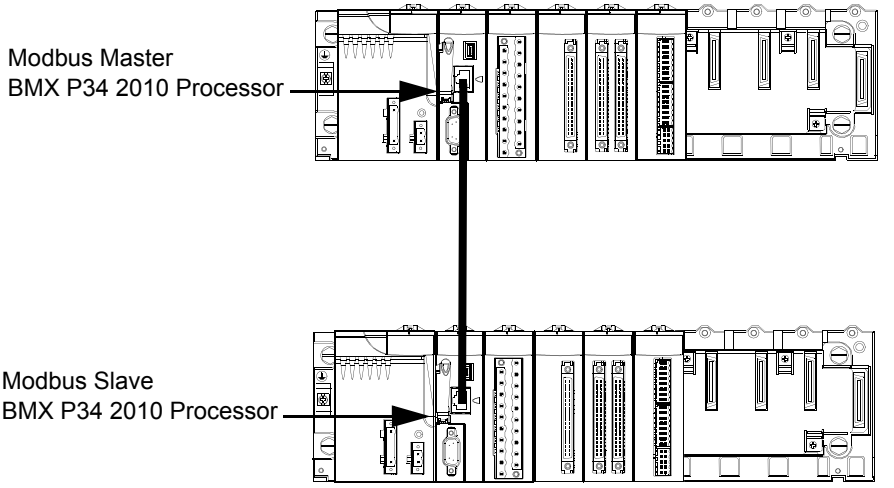
The syntax of Modbus addressing is `ADDM ('r.m.c.node')`. The meaning of the string parameter is as follows:

- r: rack address. The processor's rack address is always 0.
- m: module address. The Modicon M340 processor's slot number in the rack is always 0.
- c: channel address. The Modicon M340 processor's serial port is always channel 0.
- node: slave number to which the request is being sent. The range for configured slave numbers is from 1 to 247.

Note: In a Modbus Slave configuration, an additional address, number 248, is used for a point-to-point serial communication.

Serial Link Using Modbus Protocol

The diagram below shows two Modicon M340 processors connected via a serial link and using Modbus protocol:



The address settings of the slave processor number 8 are `ADDM('0.0.0.8')`.

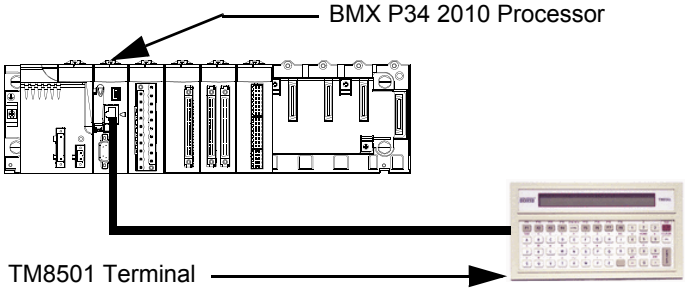
Character Mode Addressing Rules

The syntax of Character mode addressing is `ADDM ('r.m.c')` or `ADDM ('r.m.c.SYS')` (`SYS` can be omitted). The meaning of the string parameter is as follows:

- r: rack address of the connected device,
- m: module address of the connected device,
- c: channel address of the connected device,
- SYS: keyword used to stipulate the station server system. SYS can be omitted.

Serial Link Using Character Mode Protocol

The diagram below shows a Modicon M340 processor linked to a data entry/display terminal TM8501:



The address settings of the TM8501 terminal are `ADDM('0.0.0')` or `ADDM('0.0.0.SYS')`.

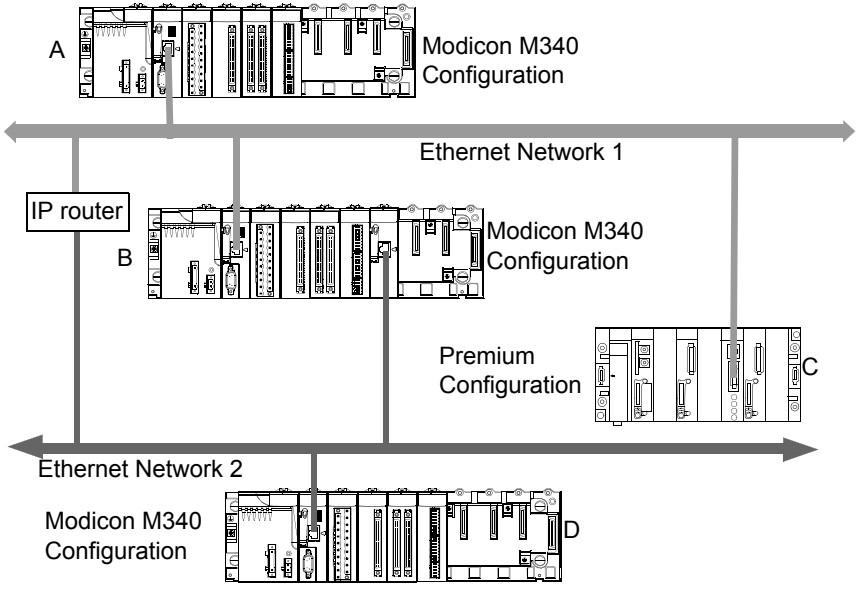
Examples of Modicon M340 Communication EFs Addressing

At a Glance

The multi-network addressing available on Modicon M340 PLCs is described below.

Example 1

The first example is a multi-network configuration as follows:



In the diagram above there are the following configurations:

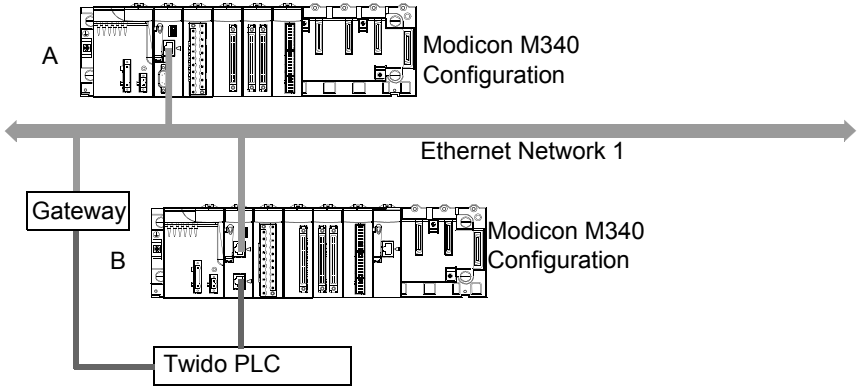
- Three Modicon M340 configurations called A, B and D,
- One Premium configuration called C.

All the configurations can communicate because of the following statements:

- A and B: communication between two Modicon M340 PLCs on an Ethernet network is possible.
- A and C: communication between a Modicon M340 PLC and a Premium PLC is possible on an Ethernet network.
- A or C, and D: communication between two Modicon M340 PLCs or between a Modicon M340 PLC and a Premium PLC on Ethernet multi-network is possible. An IP router is required.

Example 2

The second example is a multi-network configuration as follows:



In the diagram above there are two Modicon M340 configurations which are called A and B. The configuration B is directly connected to a Twido PLC via Modbus communication channel.

Communication between the two Modicon M340 PLCs is possible because the configurations are linked to the same Ethernet network.

Communication between the configuration A and the Twido PLC is possible only if you use an Ethernet/Modbus gateway. In case of it is a CANopen device which is connected to the configuration B, an Ethernet/CANopen gateway is required.

Note: To address the CANopen or Modbus device on the configuration A you must use the following syntax : `ADDM('Netlink{hostAddr}node')`, the gateway being identified with `hostAddr` field. For example, if the Netlink is set to `Ethernet_1`, the gateway address is `139.160.234.64` and the slave number of the Twido PLC is set to 247, the syntax of the ADDM function is as follows: `ADDM('Ethernet_1{139.160.230.64}247')`

IP Addressing

7

Reminder about IP Addressing

IP Address

On a TCP/IP Ethernet network, each device must have a unique IP address. This address is made up of two identifiers, one of which identifies the network, while the other identifies the connected machine.

The uniqueness of the addresses is managed as follows:

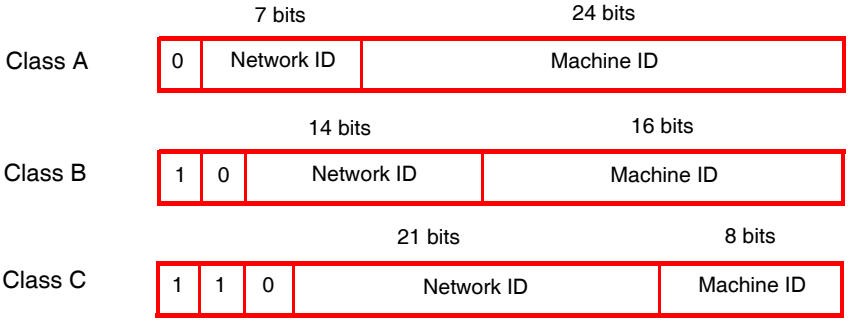
- If the type of network environment is open, the uniqueness of the address is guaranteed by the attribution of a network identifier by the relevant authority in the country where the network is located.
- If the type of environment is closed, the uniqueness of the address is managed by the company's network manager.

An IP address is defined as 32 bits. It consists of 4 numbers, one for each byte of the address.

Note: Standardized and made common largely thanks to the Internet, IP addressing is described in detail in RFCs (Request For Comment) 1340 and 791 which stipulate the Internet standards as well as in computing manuals describing networks. You can refer to these sources for further information.

Example

Depending on the size of the network, three classes of address can be used:



Spaces reserved for the different classes of IP address:

Class	Range
A	**0**.0.0.0 to **127**.255.255.255
B	**128**.0.0.0 to **191**.255.255.255
C	**192**.0.0.0 to **223**.255.255.255

- Class A addresses are intended for large-scale networks which have a large number of connected sites.
- Class B addresses are intended for medium-scale networks which have fewer connected sites.
- Class C addresses are intended for small-scale networks which have a small number of connected sites.

Sub-Addressing and Subnet Mask

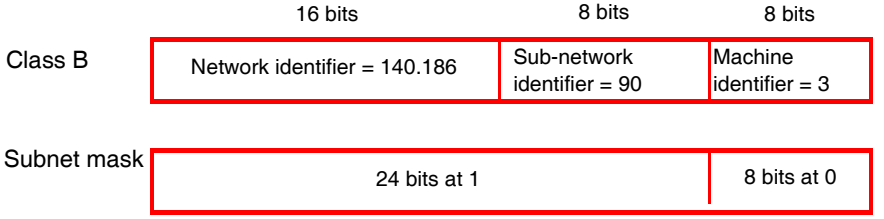
An IP address is composed of two identifiers, one identifies the network while the other identifies the connected machine. In reality, the machine identifier can also hold a subnet identifier.

In an open environment, having received a network identifier from the relevant authority, the local system administrator has the possibility of managing many networks. This means that local networks can be installed without having any effect on the outside world which still sees only one network, the one designated by the network identifier.

The subnet mask makes it possible to see the number of bits attributed respectively to the network identifier and to the subnet identifier (bits at 1), and then to the machine identifier (bits at 0).

Example

Example: 140.186.90.3



The segmentation allows for 254 possible sub-networks, each with 254 sub-network machines.

The value of the subnet mask should be chosen so that it is consistent with the IP address class.

The subnet mask will have the following value:

- For a class A address: 255.xxx.xxx.xxx
- For a class B address: 255.255.xxx.xxx
- For a class C address: 255.255.255.xxx

Where xxx is an arbitrary value which can be chosen by the user.

Gateway

The term Gateway is used in this manual in the sense of "router". If the target machine is not connected to the local network, the message will be sent to the "default gateway" connected to the local network which will manage the routing to another gateway or towards its final destination.

Operating Modes

III

At a Glance

Subject of this Part

This part describes the operating modes associated with expert communication.

What's in this Part?

This part contains the following chapters:

Network Configuration

8

At a Glance

Subject of this Chapter

This chapter presents the tools for configuring a network at the global level and at the station level.

What's in this Chapter?

This chapter contains the following topics:

Network Configuration Principle Using Unity Pro

At a Glance

With Unity Pro, the installation of a network is done using the project browser and the hardware configuration editor.

The method involves the following four steps:

- Creation of a logic network,
- Configuration of the logic network,
- Declaration of the module or of the PCMCIA card,
- Association of the card or of the module with the logic network.

These four steps are presented further on in this documentation.

Note: The advantage of this method is that from the second step onwards, you can design your communication application (you do not need to have the hardware to start work) and use the simulator to test its operation.

Note: The first two steps are performed using the project browser and the next two using the hardware configuration editor.

This manual introduces the method. For detailed information on how to configure the various networks, please refer to the specific documentation:

- Ethernet configuration (See Premium and Atrium Using Unity Pro, Ethernet Network Modules, Configuration Parameters)
- Modbus Plus configuration (See Premium and Atrium using Unity Pro, Modbus Plus Network, Modbus Plus configuration parameters)
- Fipway configuration (See Premium and Atrium using Unity Pro, Fipway Network, Fipway Configuration Screen)

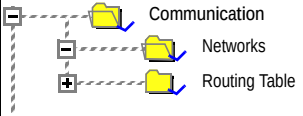
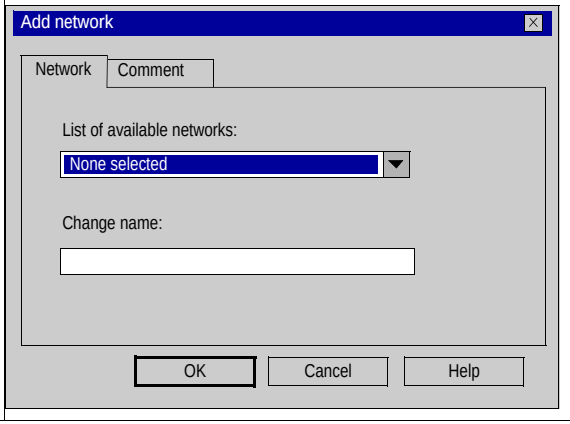
Creating a Logic Network

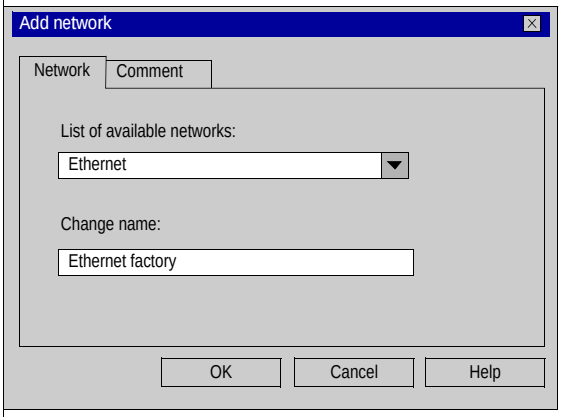
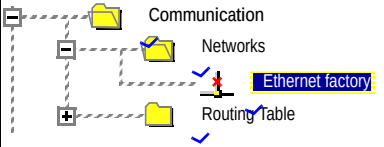
At a Glance

The first step in implementing a communication network is to create a logic network.

Creating a Logic Network

The following table describes how to create a network using the project browser.

Step	Action
1	Expand the *Communication* directory in the **project browser**. **Result**: 
2	Right-click in the *Networks* sub-directory and select the **New network** option. **Result**: 

Step	Action
3	Select the network that you wish to create from the list of available networks and give it a meaningful name. **Result**: Example of an Ethernet network:  **Note**: You can also add a comment, if you so wish, by clicking on the **Comment** tab.
4	Click OK and a new logic network is created. **Result**: We have just created the Ethernet network that appears in the project browser.  **Note**: As you can see, a small icon indicates that the logic network is not associated with any PLC hardware. Furthermore, the small blue "v" sign indicates that the project needs to be rebuilt before it can be used in the PLC.

Configuring a Logic Network

At a Glance

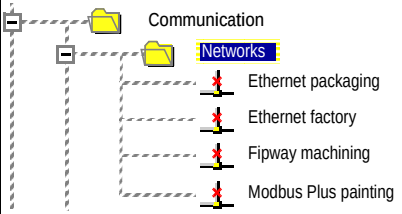
The second step in implementing a communication network consists of configuring a logic network.

This manual introduces the access to network configuration. For further information on configuring the various networks please refer to the specific documentation:

- Ethernet configuration (See Premium and Atrium Using Unity Pro, Ethernet Network Modules, Configuration Parameters)
- Modbus Plus configuration (See Premium and Atrium using Unity Pro, Modbus Plus Network, Modbus Plus configuration parameters)
- Fipway configuration (See Premium and Atrium using Unity Pro, Fipway Network, Fipway Configuration Screen)

Configuring a Logic Network

The table below describes how to access the configuration of a network using the project browser.

Step	Action
1	In the project browser, expand the **Networks** sub-tab located in the **Communication** tab of the tree directory to display all the networks of the project. **Example**: 
2	Double-click the network you wish to configure to obtain the network's configuration window. **Note**: The windows differ according to the network family selected. However, for all networks it is from this window that you can configure the Global Data, IO scanning and Peer Cop services, the common words, etc. **Note**: For Ethernet networks an intermediate step is necessary which involves selecting the family of the module that will be used in the hardware configuration.

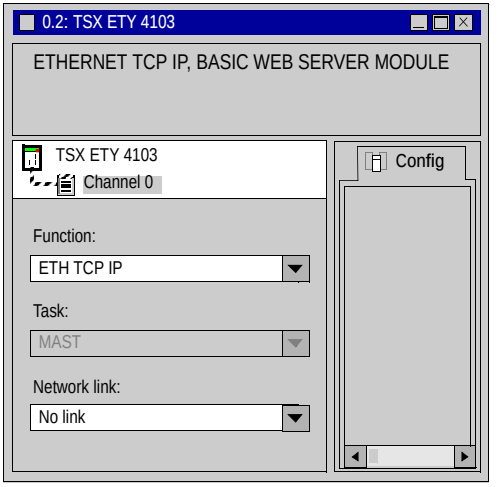
Associating a Logic Network with Network Hardware

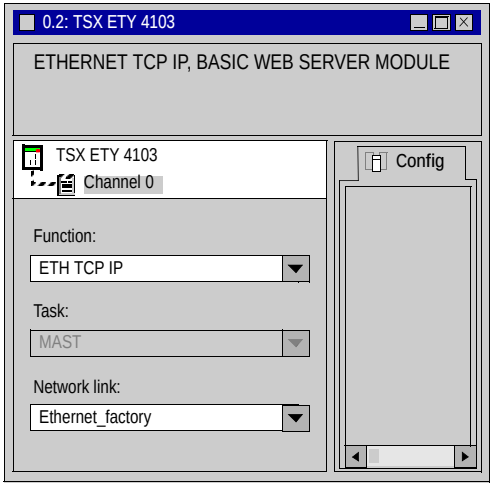
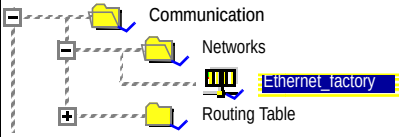
At a Glance

The final step in implementing a communication network is to associate a logic network with a network module, Modbus Plus card or Fipway card. Although the screens differ, the procedure is the same for each network device.

How to Associate a Logic Network

The following table describes how to associate a logic network to a network device declared in the hardware configuration editor.

Step	Action
1	Open the hardware configuration editor.
2	Right-click the device (Ethernet module, Fipway PCMCIA card or Modbus Plus PCMCIA card) that you wish to associate with a logical network.
3	Select the channel and function. **Result:** For a TSX ETY 4103 module: 

Step	Action
4	In the **Network link** field, select the network to be associated with the card. **Result:** 
5	Confirm your choice and close the window. **Result:** The logic network is associated with the device. The icon associated with this logic network changes and indicates the existence of a link with a PLC. Furthermore, the rack, module and channel numbers are updated in the logic network configuration screen. In our example we obtain the following project browser: 

Debugging

9

Description of the Communication Debug Screens

At a Glance

The debug screen dedicated to the application-specific communication function may be accessed via the **Debug** tab. It is broken down into two distinct sections:

- The top and left-hand part of the screen, common to all types of debug screen, is dedicated to module and communication channel information.
- The right-hand part of the screen is dedicated to debugging data and parameters. This area, specific to the type of communication selected, is detailed in the documentation relating to the various types of communication.

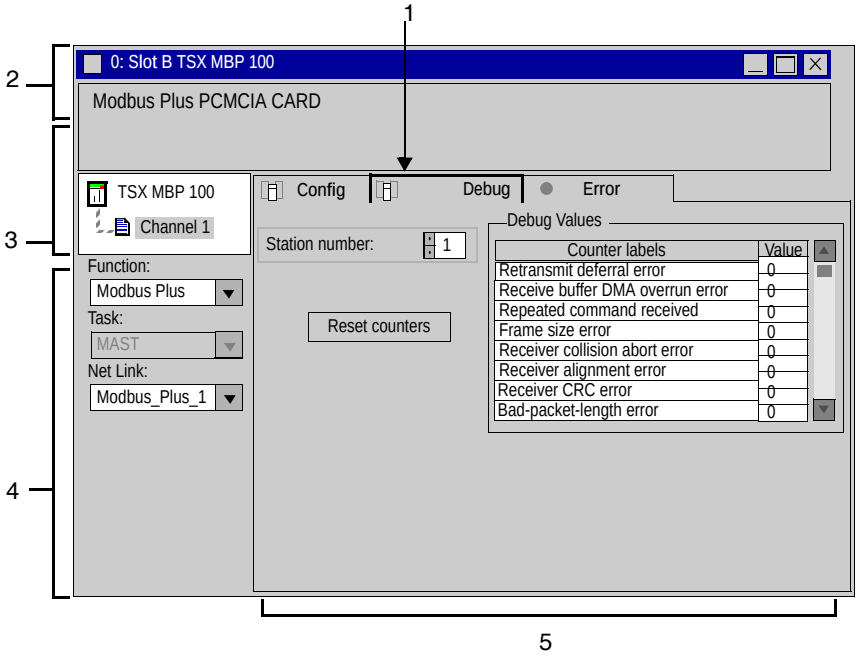
How to Access the Screen

The debug mode may only be accessed in online mode.

Step	Action
1	Access the configuration screen.
2	Select the **Debug** mode by clicking on the corresponding tab.

Illustration

This area is used to access diagnostics for a communication channel.



Description

The table below shows the various elements of the debug screen and their functions.

Address	Element	Function
1	Tabs	The tab in the foreground indicates the mode in progress (**Debug** in this example). Each mode can be selected by the corresponding tab. The modes available are: • **Debug** which can be accessed only in online mode. • **Diagnostics** which may be accessed only in online mode. • **Configuration**. • **Adjust**.
2	**Module** area	This area displays the abbreviated module indicator. In the same area there are 3 LEDs which indicate the status of the module in online mode: • **RUN** indicates the operating status of the module. • **ERR** indicates an internal fault in the module. • **I/O** indicates a fault from outside the module or an application fault.

Address	Element	Function
3	**Channel** area	This area is used to select the channel to be debugged: ● **Channel:** module channel number. To the left of the symbol there is a copy of the **CHx** channel LED.
4	**General parameters** area	This area shows the communication channel parameters: ● **Function:** shows the configured communication function. This information cannot be modified. ● **Task:** shows the task (configured **MAST**). This information cannot be modified.
5	**Mode parameters** area	This area contains the parameters of the mode selected by the tab.

Note: Unavailable LEDs and commands appear grayed out.

Communication Function Programming and Entry Help

10

At a Glance

Subject of this Chapter

This chapter presents the various entry help tools.

What's in this Chapter?

This chapter contains the following topics:

Communication Functions Entry Help

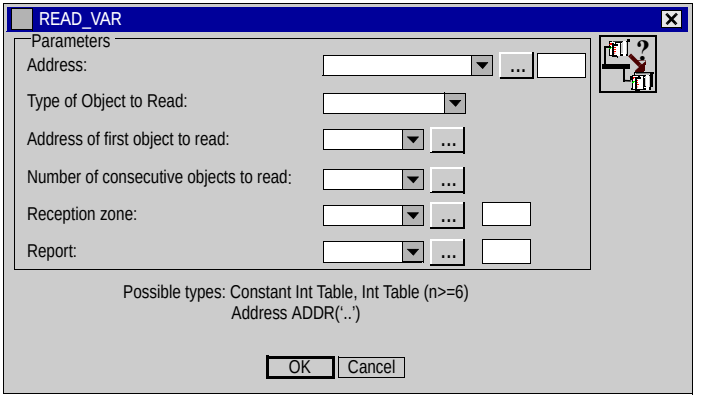
At a Glance

During programming you can access an entry help screen, allowing you to complete all the parameters of a communication function.

This help can be obtained from the library functions of Unity Pro.

Illustration

The following illustration shows the entry help screen for the communication function `Read_Var`.



Note: The number and type of fields vary according to the communication function selected.

Availability

This screen is available for the following communication functions:

- `DATA_EXCH`
- `INPUT_CHAR`
- `OUT_IN_CHAR`
- `PRINT_CHAR`
- `READ_VAR`
- `SEND_REQ`
- `SEND_TLG`
- `WRITE_VAR`

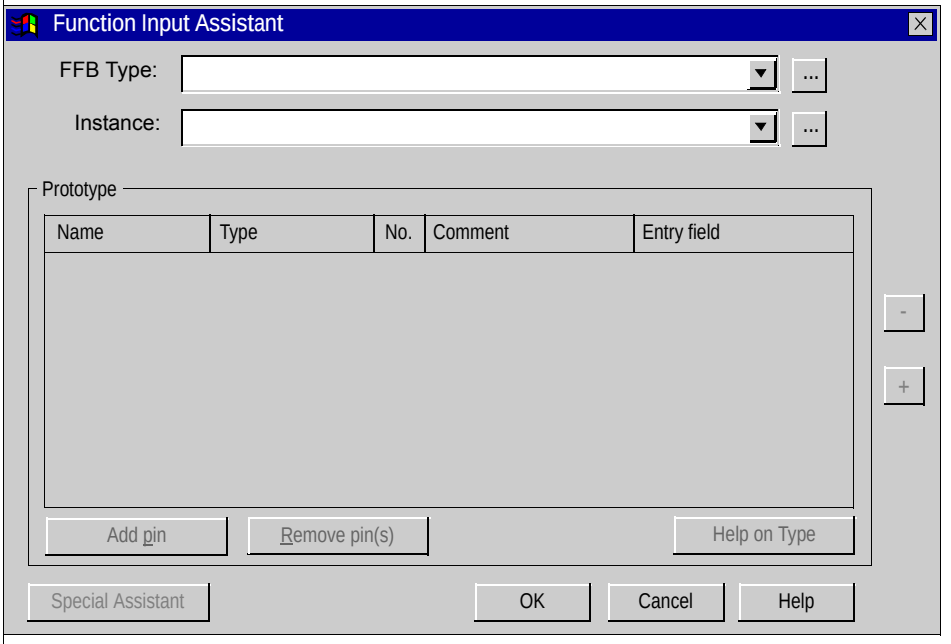
How to Access a Function, Function Block or DFB-Type Specific Instruction

At a Glance

The application-specific function may be accessed:

- by direct entry of the instruction and its parameters in an operate block
- via the entry help function that may be accessed in the program editors (FBD, LD, IL, ST).

How to Call a Function

Step	Action
1	Access the required editor.
2	Depending on the editor, select one of the following methods to open the function library: ● Select the function to enter with the data editor. Once in the editor, right-click on the function (LD, FBD editors). ● Right-click in the program editor and select the option **FFB Input Assistant**. **Note:** The function entry help window appears. 
3	Select the type of FFB required (if it is not already entered).
4	Then select the name of the instance (where necessary and if available).

Step	Action
5	Many instructions have a customized entry help screen. Access this screen by clicking the **Special Assistant** button.
6	Enter each parameter of the instruction (each instruction is expanded upon in the relevant application-specific documentation): ● In the customized detailed data entry screen or ● In the **Entry field** field, in the **Prototype** area.
7	Validate with **Ok**.

Address Entry Help

At a Glance

To assist you in entering the address, a help screen is available to you.

This screen provides a description of the architecture in which the communication function is integrated and generated.

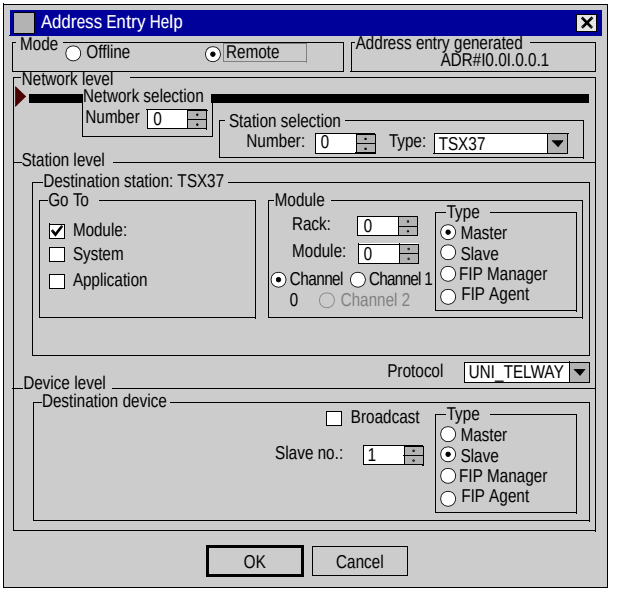
By completing the fields of this description, the address is automatically generated.

How to Access the Help

When entering the parameters of the communication function, you can use the  button to access the address entry help.

Illustration

The following illustration shows the address entry help screen for a communication function.



Mode

The first parameter to select is the **Mode**. It allows you to select one the following communication modes:

- offline (communication by bus)
- remote (communication by network)

Network Level

For remote communications only, the **network level** is used to:

- enter the network number,
- enter the station number,
- select the station type.

Station Level

Depending on the communication function, this parameter allows you to select the type of exchange:

- The **Application** box selects an exchange with a PL7 application (corresponds to APP addressing).
- The **System** box selects the PLC system of the station designated by the network level (corresponds to SYS addressing).
- The **Module** box means that the destination device is connected to the station via a link (Uni-Telway, Modbus, Modbus Plus or Fipio). This case requires you to specify:
  - The position of the module supporting this link.
  - The type of this module.

Protocol

The **Protocol** field defines the exchange protocol used between the network's station and the exchange's destination device.

Device Level

This parameter is used to specify:

- The type of destination device.
- The address of this device.

Limitations

In the address entry help screen, some communications (from a Uni-Telway slave) require coding of the destination address in the transmission buffer (See Unity Pro, Communication Block Library, SEND_REQ: Sending requests).

The help window can be used to fully enter the section corresponding to `ADDR()`, advising the user that he must code the additional buffer.

Coding of the addresses of the remote stations is only supported by the following devices: TSX 17, TSX 37, TSX 47-107 and TSX 57.

For third-party devices, only entry of the port number is proposed. In other cases, the address must be entered manually.

Index

N

S