Device and Communication Manual

For Cement Library V1.1
a SoCollaborative Library based on DPL

07/2011

EIO0000000982.00

www.schneider-electric.com

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 help ensure compliance with documented system data, only the manufacturer should perform repairs to components.

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

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

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

Table of Contents

Safety Information

Important Information

NOTICE

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

The addition of this symbol to a Danger 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 or serious injury.

⚠ CAUTION

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

CAUTION

CAUTION, used without the safety alert symbol, indicates a potentially hazardous situation which, if not avoided, **can result in** 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.

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

BEFORE YOU BEGIN

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

⚠ WARNING

UNGUARDED MACHINERY CAN CAUSE SERIOUS INJURY

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

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

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

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

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

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

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

START-UP AND TEST

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

⚠ CAUTION

EQUIPMENT OPERATION HAZARD

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

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

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

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

Verify that the completed system is free from all short circuits and grounds, except those grounds installed according to local regulations (according to the National Electrical Code in the U.S.A, for instance). If high-potential voltage testing is necessary, follow recommendations in equipment documentation to prevent accidental equipment damage.

Before energizing equipment:
- Remove tools, meters, and debris from equipment.
- Close the equipment enclosure door.
- Remove ground from incoming power lines.
- Perform all start-up tests recommended by the manufacturer.

OPERATION AND ADJUSTMENTS

The following precautions are from the NEMA Standards Publication ICS 7.1-1995 (English version prevails):
- Regardless of the care exercised in the design and manufacture of equipment or in the selection and ratings of components, there are hazards that can be encountered if such equipment is improperly operated.
- It is sometimes possible to misadjust the equipment and thus produce unsatisfactory or unsafe operation. Always use the manufacturer's instructions as a guide for functional adjustments. Personnel who have access to these adjustments should be familiar with the equipment manufacturer's instructions and the machinery used with the electrical equipment.
- Only those operational adjustments actually required by the operator should be accessible to the operator. Access to other controls should be restricted to prevent unauthorized changes in operating characteristics.

About the Book

At a Glance

Document Scope

This document provides information on operating a project developed with Unity Application Generator (UAG) and the Device and Communication components which is based on the Device and Process Library (DPL).

This document is not intended to cover any development procedures and internal functionality details.

The Device and Communication components are intended for end users and system integrators.

Users of the Device and Communication components must have very good knowledge in the employment of:
- Unity Application Generator V3.2 or V3.3
- Unity Pro V4.1 S, M, L, XL, and 5.0
- Vijeo Citect V7.1

Validity Note

This manual is valid for the listed versions of the following products:
- Unity Application Generator V3.2 or V3.3
- Unity Pro V4.1 S, M, L, XL, and 5.0
- Vijeo Citect V7.1

Related Documents

Title of Documentation	Reference Number
Unity Application Generator 3.3 Basic User Manual	33002830
Unity Application Generator 3.3 Extended User Manual	33003669
Unity Application Generator Quick Start Guide	33003222
SCoD Editor 3.3 User Manual	33002608

UAG Process Application Library for Vijeo Citect V2.0, User Manual Level 1	33004234
UAG Process Application Library for Vijeo Citect V2.0, User Manual Level 2	EIO0000000011

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

Product Related Information

⚠ WARNING

UNINTENDED EQUIPMENT OPERATION

The application of this product requires expertise in the design and programming of control systems. Only persons with such expertise should be allowed to program, install, alter, and apply this product.

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

⚠ WARNING

UNINTENDED EQUIPMENT OPERATION

All examples in this manual are given for information only. Before being used in an industrial application, they must be suitably adapted to the specific functions and safety requirements of the application concerned.

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

User Comments

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

Device Components

I

Overview

This part describes the Device function blocks.

What's in this Part?

This part contains the following chapters:

Introduction

1

Introduction

Overview

The DPL control system provides resources or components that are specifically designed to automate the control of Schneider Electric devices like variable speed drives, starters, power meters, etc.

The control resources (included in the DPL platform for Unity (DPU)) and monitoring resources (included in the DPL platform for Vijeo Citect (DPC)) provide the most commonly required functions, facilitating the development of control systems for devices and reducing the risk of detected errors.

Unity functional blocks (DFB) are provided at the control level, and dynamic representations (Genies) and Faceplates (implemented through windows with SuperGenies syntax) are provided at the monitoring level.

To automate and simplify the implementation of device control systems, these resources can be used together with the tools provided for generating code massively and for synchronizing the control and monitoring subsystems.

This document includes a detailed list of the functional blocks belonging to the Cement Library that refers to the Device Library.

The Communications library can be used to supplement control systems with extra data monitoring. These libraries can be used for Ethernet, CANopen and Modbus communications.

General Overview of Device Resources

2

Overview

This chapter describes the general overview of device resources and profile descriptions of the control blocks.

What's in this Chapter?

This chapter contains the following sections:

2.1 Resources for Devices

Overview

This section describes the general overview of resources.

What's in this Section?

This section contains the following topics:

DPU Functional Blocks for Devices

General

The functional blocks described in this document, and which have been designed specifically for device control automation, are listed below.

Components have been classified into profiles, according to their respective device range, in order to standardize information and make this information easier to use.

ATV Profile Variable Speed Drives *(see page 29)*
TesysT Profile Motor Control *(see page 30)*
PM Profile Power Meters *(see page 31)*

DPU Component Library for Devices

Overview

The functional blocks listed in the previous section are DPU components that implement device control with the CANopen, serial Modbus, and Ethernet communication buses for the most common Schneider Electric devices.

The DPU components are grouped in the DPU Device Library with SCoD Editor.

DPU Components

The DPU components in the DPU Device Library include the following resources for the automatic generation of Unity projects:

- Variables
- Control logic
- Operator Screens

The component categories included in the DPU Device Library are as follows:

- Devices on Modbus networks
- Devices on CANopen networks
- Devices on Ethernet networks

DPC for Devices

Genie

The genie names are as follows:

- ATV
- PM
- TESYST
- EM6400
- EM7300

Faceplates

Pop-up windows with SuperGenies syntax: Device

2.2 Profile Descriptions

Overview

This chapter describes the profile descriptions of the different control blocks.

What's in this Section?

This section contains the following topics:

ATV Profile Description

Overview

The objective of these functions is to manage ALTIVAR variable speed drives (ATV31, ATV61 and ATV71) in a communication based manner.

ATV Profile DFB List

Depending on the communication technologies, the different ATV Profile blocks manage different ALTIVAR variable speed drives as described:

ATV7161Dev: This is the control block for ATV61 and ATV71 variable speed drives:

- on an Ethernet network
  Refer to ATV7161Dev - Ethernet Communication *(see page 273)*.
- on a CANopen network
  Refer to ATV7161Dev - CANopen Communication *(see page 85)*.

ATV31Dev: This is the control block for ATV31 variable speed drives on CANopen networks.
Refer to ATV31Dev - CANopen Communication *(see page 41)*.

MBATVDev: This is the control block for ATV31, ATV61 and ATV71 variable speed drives on Modbus network.
Refer to MBATVDev - Modbus Communication *(see page 195)*.

These functional blocks are DPU components that implement device control with CANopen, serial Modbus, and Modbus TCP/IP (Ethernet) standard communication buses for the most common Schneider Electric devices.

ATV Profile Functional Description

This table describes the main functions of the ATV Profile control blocks:

Function	Description
Control	Selects the direction of rotation (forward, reverse).
Speed	Enables the speed set-point of the device.
Device Error	Monitors the errors occured in the device.
Remote Resetting	This enables device resets by monitoring unit [HMI].
Control or Monitoring	Enables either controlling or monitoring the device. Selecting **Control** enables the use of either the controller or the wired speed driver inputs or outputs to control either the direction of rotation or the speed set-point. The **Control** or **Monitoring** functions allows the control and set point to be controlled separately.
Owner	Use the Owner to select the control system (operator or program) that is responsible for setting the control.

TesysT Profile Description

Overview

The TesysT Profile block is a part of the device control block family.

The objective of these functions is to manage the TesysT family of motor control devices in a communications-based manner.

TesysT Profile DFB List

Depending on the communication technologies, the TesysT Profile manages the TesysT devices with the following DFB:

TesysTDev: This is the control block for the TesysT controller on an CANopen communication *(see page 107)*.

MBTESYSTDev: This is the control block for the TesysT controller on a Modbus Communication network *(see page 247)*.

These functional blocks are DPLU components that implement device control with serial Modbus and Modbus TCP/IP (Ethernet) standard communication buses for the most common Schneider Electric devices.

TesysT Profile Functional Description

This table shows the functional description of the TesysT Profile control block:

Function	Description
Control	Device Operation.
Direction	Forward/reverse direction of rotation.
Speed	This function enables the change of speed (fast or slow),if you are working with a 2-speed motor start (double winding).
Device Failure	This function monitors the detection of errors that are associated with a device.
Remote Resetting	This function enables the resetting of the device by control and monitor unit.
Control	This function enables to control or monitor the device.
Owner	The module manages which control system level (Operator or programmer) Program is the owner; therefore, it is responsible for setting the control.

PM Profile Description

Overview

The PM profile block is a part of the device control block family.

The objective of these functions is to manage the PowerLogic Series Power Meters in a communications-based manner.

PM Profile Functional Description

This table sdescribes the functions of the PM Profile:

Function	Description
Remote Resetting	Enables to reset the device
Monitoring	Allows the most important parameter devices to be monitored

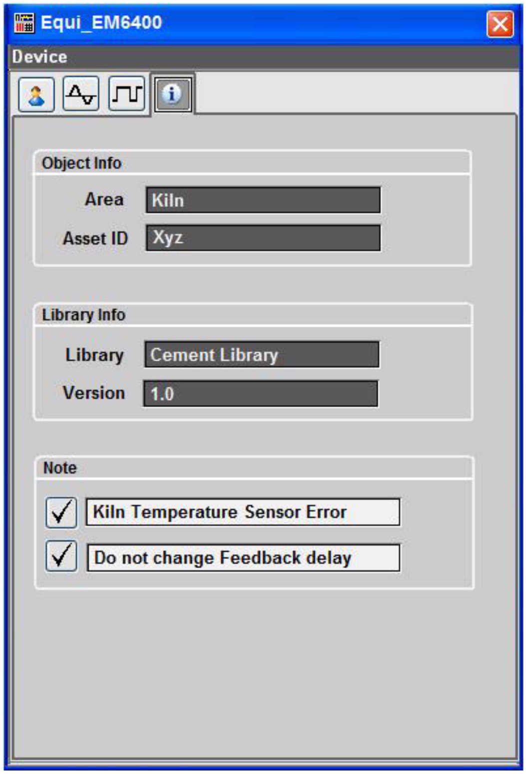
2.3 Commonly Used Tabs

Commonly Used Tabs

Overview

This provides information about the commonly used tabs and sub-tabs. These tabs are commonly available as popups in most of the function blocks.

Information Tab



Each sub-tab has 3 sub-sections:

- **Object Info**
- **Library Info**
- **Note**

The following table describes the sub-section **Object Info**:

Item	Description
Area	Displays the area to which the object belongs. This is defined during the instantiation of the block and cannot be edited in the popup.
Asset ID	Displays the asset ID of the object. This is defined during the instantiation of the block and cannot be edited in the popup.

The following table describes the sub-section **Library Info**:

Item	Description
Library	Displays the library name. This is a static text.
Version	Displays the library version. This is a static text.

The following table describes the sub-section **Note**:

Symbol	Description
	The operator can write 2 short notes. It can be acknowledged by clicking the button available before the text. This acknowledgement erases the note.

General Concepts

3

Overview

This chapter describes the general concepts of the device control block family.

What's in this Chapter?

This chapter contains the following topics:

Functional Blocks (DFB) Interface

General

The Unity function blocks for Devices provide an interface that allows the blocks to be configured, monitored, and controlled both from the monitoring subsystem and from the control subsystem itself.

The following interfaces are provided:
- Basic configuration
- Control
- States and Monitoring
- Variables for communications with devices

Basic Configuration

Public variables belonging to the DFBs for static, known data in engineering time (e.g., range for an input channel, communication variable refreshing).

Control

DFB input and output pins. Enables to issue commands from other program blocks or parts, and provides the block status in order to allow the implementation of switching operations (e.g., device controlled from HMI/SCADA, ExtControlled), detected communications interruptions, alarms (e.g., thermal trip alarm), etc.

States and Monitoring

Depending on their type, the blocks have several input/output pins that must be connected to variables used to maintain either the pins states or data, and which also enables to manage commands and parameters received from the monitoring subsystem.

Status and control (identified by the _ST suffix): These variables are used to maintain the status and control used from the monitoring subsystems first level (dynamic symbols on the synoptics). This is an input/output variable.

Configuration and information (identified with the _CFG suffix): These variables are used to hold second-level information for configuring device control or information parameters (alarms, detected error codes, etc.).
This information is normally accessible from the monitoring subsystems Faceplates.
This is an input/output variable.

Status and control (identified by means of the _ST suffix): This information is normally accessible from the monitoring subsystems Faceplates for digital or analog values.
This is an output variable and is available depending on the device information and on the communications system used (Modbus, CANopen, or Ethernet).

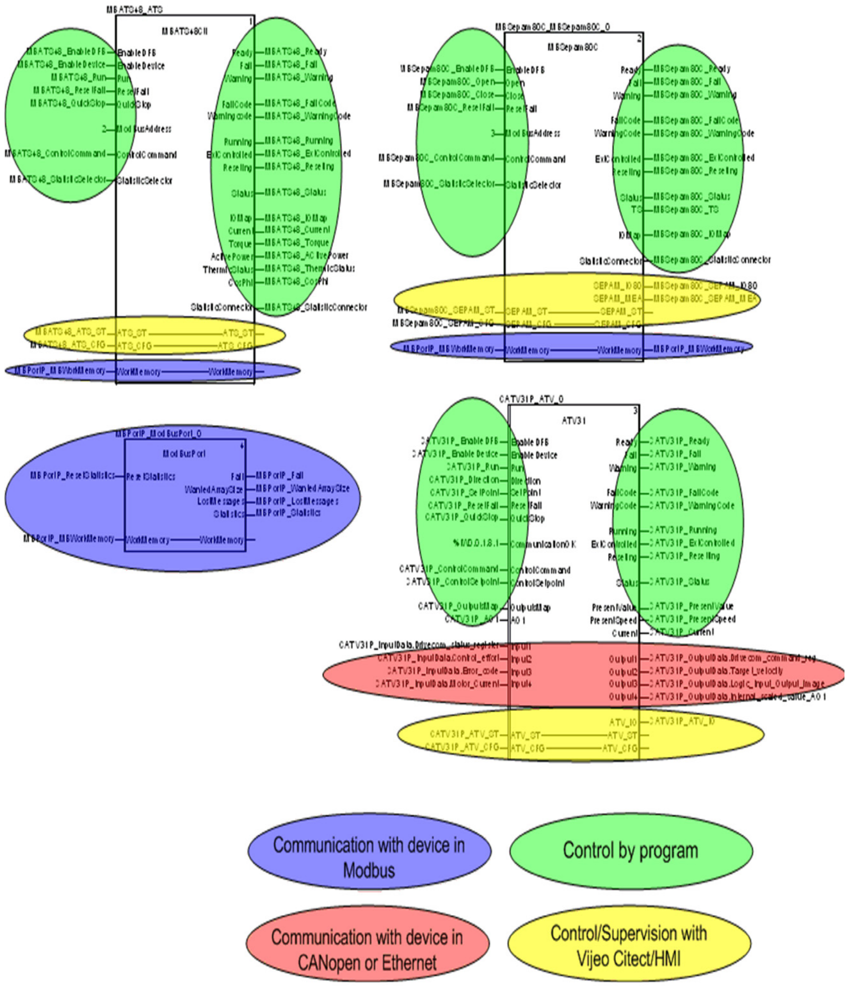
Variables for Communications with Devices

In the event that components are used with CANopen or Ethernet communications, the blocks feature an input DDT (input data) and an output DDT (output data) for carrying out communications with the physical device.

These variables have to be allocated (%MWX) on the memory zone reserved for communications to allow the device controls to function properly.

For Modbus communications, a "Modbus Port" that serializes requests issued to bus devices (messaging) must be used.

Graphical View



Set-Point Management

Description

The device control blocks manage the set-points coming from the following sources, according to the Owner:

Operator (Operator/Local set-point): Commands received from the monitoring subsystem (Vijeo Citect/HMI).
These commands are received at the functional blocks through the •••_ST and •••_CFG structures.
The owner variable is implemented in the devices with •••_ST.CFGW.1, and must be set to 1 for Operator-based control.

Program: Commands generated from the control program (Unity).
These commands are received at the functional blocks through the DFB input and output pins.
The owner variable is implemented in the devices with •••_ST.CFGW.1, and must be set to 0 for Operator-based control.

To avoid unwanted transitions in the set-point, the local set-points (Operator or Program) are continuously set to match the selected set-point.

The functional block Owner (Operator or Program) must not be modified from the control subsystem with the program in the PLC.

This will ensure that the only commands issued towards the block are generated from the monitoring subsystem.

NOTE: In the DPC device dynamic symbols (Genies) currently available as part of the DPL resource for Vijeo Citect, only the ResetFail block reset is implemented if control is carried out in "Operator" mode. To carry out additional control actions (e.g., Run, QuickStop, Open, etc.), it is possible to implement a control configuration by using the control blocks available from the Process or Device libraries.

Monitoring

Description

Functional blocks in the Device library have a monitoring mode (ControlCommand) that enables a device to be monitored in a communications-based manner, independently of the control inputs state.

This way, the control block only performs read operations on the device, updating the blocks output variables without sending any set-points.

This operating mode can be used if switching operations are carried out locally on the device with a physical screw terminal.

NOTE: If the block is in monitoring mode (ControlCommand set to FALSE), it will only be possible to reset a detected communications interruption through the program or HMI/SCADA. Set the ControlCommand variable to TRUE, if the device is to be configured to be completely controlled and monitored by the PLC in a communications-based manner.

ATV31 - CANopen ATV31 Drive (CATV31M SCoD)

4

Overview

This chapter describes the DFB for the control block for varaible speed drives on a CANopen network, represented by CATV31M SCoD.

What's in this Chapter?

This chapter contains the following sections:

4.1 Description

Overview

This section describes the ATV31Dev DFB.

What's in this Section?

This section contains the following topics:

Functional Description

General

ATV31Dev is a control block for the ATV 31 on a CANopen network.

The ATV31Dev block is implemented as listed below:

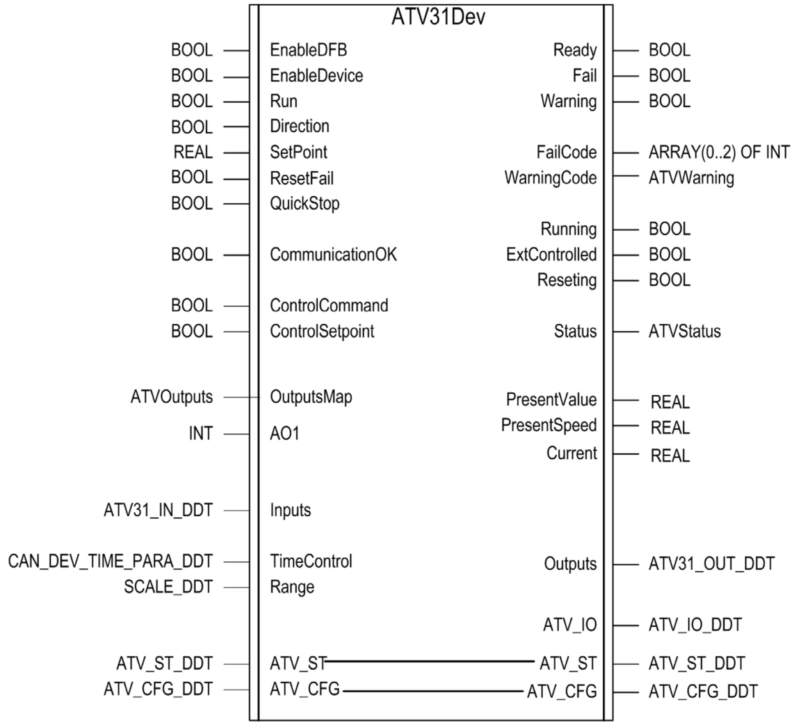
Software	Implemented as
Unity Pro	DFB ATV31Dev
Vijeo Citect	DPC_Devices
UAG	SCoD CATV31M

The main function blocks are described in the ATV Profile Functional Description *(see page 29)*.

FBD Representation

Representation in FBD

The functional module that will be used in the program has the following aspect (at the FBD block level) after PLC generation in UNITY Pro:



4.2 Inputs

Overview

This section describes the inputs of the ATV31Dev FBD.

What's in this Section?

This section contains the following topics:

Generic Inputs Description

Overview

This table is the same for all ATV Profiles.

Inputs Description Table

This table describes the generic inputs of the ATV Profile FBD:

Input	Type	Description
EnableDFB	BOOL	This input enables the normal execution of the control block. EnableDFB = 0. The entire DFB is restarted (statuses, output values, counters, etc. are lost) and all output values are set to 0 or FALSE. EnableDFB = 1. This input enables communications with devices for their operation. Public variable values are loaded during the first enabling cycle.
EnableDevice	BOOL	This input variable is true only if EnableDFB is true. To control the speed of the drive, it must be enabled.
Run	BOOL	This input makes the motor run in the direction selected with the Direction variable.
Direction	BOOL	This input activates the direction of motor rotation: **1 =** forward **0 =** reverse **NOTE:** Setpoint assumes positive values. Changes in the direction of rotation cannot be made with Setpoint sign.
Setpoint	REAL	This is the speed set-point *(see page 54)* requested from the speed of the drive. It can only have positive, real values.
ResetFail	BOOL	Existing detected errors are reset on the rising edge of this signal. Both communications and detected device errors can be reset. If the detected error is a device error, it sends a reset command to the device if ControlCommand = TRUE. This figure shows the input ResetFail: Fail AND (NOT Reseting) —| | ResetFail
QuickStop	BOOL	This input enables a quick stops the drive speed. This variable is state-based. If there is a QuickStop, the Run bit has to be reset to resume operation.

Input	Type	Description
ControlCommand	BOOL	This input indicates if the starter is controlled from an external source. If ControlCommand=TRUE the device is controlled from an external source. The generation of follow-up alarms is disabled. If ControlCommand=FALSE, the block only performs read operations on the device status, the transmission of device control or reset commands is not possible. **NOTE:** This input does not configure the starter; it only defines the DFB operating mode.
ControlSetPoint	BOOL	This input indicates if the speed of the drive is controlled from an external source. (e.g., from the console or a potentiometer)

⚠ WARNING

UNEXPECTED MOTOR START

Reset the Run variable before resuming operation.

The Run input must be inactive while the device is being enabled or while a detected error is being reset.

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

NOTICE

UNEXPECTED BEHAVIOUR

The ControlCommand input value must coincide with the speed driver configuration.

Failure to follow these instructions can result in equipment damage.

ATV31Dev Specific Inputs

Inputs Description Table

This table describes the specific inputs of the ATV31Dev control block:

Input	Type	Description
CommunicationOK	BOOL	A bit is used to indicate whether or not the node is present on the bus.
OutputsMap	ATVOutputs	The OutputsMap *(see page 283)* holds a data structure used to control the speed driver outputs.
AO1	INT	Controls the value of the AO1 analog output. This value can be found on all devices on CANopen and Ethernet networks. The speed of the drive analog output can only be controlled with this input variable. It cannot be controlled from HMI or SCADA.
Inputs	ATV31_IN_DDT	Inputs *(see page 49)* holds a data array structure that contains data obtained from the device. The drive can be controlled with this input variable.
TimeControl	CAN_DEV_TIME_ PARA_DDT	TimeControl *(see page 51)* is a parameter for device control.
Range	SCALE_DDT	Range *(see page 53)* is a definition for inverter (Raw and engineering range).

Input Detail - Inputs

Description

This function is a data array structure with data obtained from the device. The speed of the drive can be controlled with this input variable.

It is reserved for the DFB and cannot be used directly.

Allocate the structure (%MWx) to help ensure correct operation of the control block. Refer to Communication Technologies *(see page 435)*.

Depending on the drive speed and the communication technology, the structure varies follows:
ATV31_IN_DDT: CANopen for the ATV31.
ATV7161_IN_DDT: CANopen and Ethernet for the ATV71 and ATV61.
ATV_STB_IN_DDT: All speed of the drives on Advantys STB.

ATV31_IN_DDT Structure Description

This table shows the ATV31_IN_DDT structure:

Name	Type	Description
InputData [0]	INT	Status Word (ETA)
InputData [1]	INT	Control Effort (rpm)
InputData [2]	INT	Last detected error code
InputData [3]	INT	Current

ATV7161_IN_DDT Structure Description

This table shows the ATV7161_IN_DDT structure:

Name	Type	Description
InputData [0]	INT	Status Word (ETA)
InputData [1]	INT	Control Effort (rpm)
InputData [2]	INT	Last detected error code
InputData [3]	INT	Current
InputData [4]	INT	Current Torque
InputData [5]	INT	IL1I (digital inputs)
InputData [6]	INT	AI2 (analog input 1)
InputData [7]	INT	AI2 (analog input 2)

ATV_STB_IN_DDT Structure Description

This table shows the ATV_STB_IN_DDT structure:

Name	Type	Description
InputData [0]	INT	Status Word (ETA)
InputData [1]	INT	Control Effort (rpm)

Input Detail - TimeControl

Description

The TimeControl variable is the time parameter for device control.

The type of this input is different according to the communication technology:
- CAN_DEV_TIME_PARA_DDT with CANopen/Advantys/Ethernet.
- MB_DEV_TIME_PARA_DDT with Modbus.

CAN_DEV_TIME_PARA_DDT Structure Description

This table describes the CAN_DEV_TIME_PARA_DDT structure:

Name	Type	Description
CommandCtrlWindow	TIME	Control time for operations. This is the time that the block waits for operations to be carried out by the device. If a command is sent and executed within the time indicated by this variable, a “follow-up alarm“ is issued. The commands controlled are: ● EnableDevice ● Run A ResetFail is not interpreted as an alarm. Instead, the detected error continues and has to be reset using the Resetting output. (This is the time for the device waits to execute orders).
ScanTime	TIME	ScanTime enables ● the configuration of the time during which signals are kept active ● an ability to ensure that the monitoring subsystem will acquire the data for alarms

MB_DEV_TIME_PARA_DDT Structure Description

This table describes the MB_DEV_TIME_PARA_DDT structure:

Name	Type	Description
CommandCtrlWindow	TIME	Control time for operations. This is the time that the block waits for operations to be carried out by the device. If a command is sent and executed within the time indicated by this variable, a “follow-up alarm“ is issued. The commands controlled are: • EnableDevice • Run A ResetFail is not interpreted as an alarm. Instead, the detected error continues and has to be reset using the Resetting output (this is the time for the device waits to execute orders).
ScanTime	TIME	Enables configuring the time that the alarm signals are kept active. The monitoring subsystem acquires data for alarms.
Refresh	TIME	Time to refresh the cyclic data.

Input Detail - Range

Description

Range definition for speed driven motors (raw and engineering range).

SCALE_DDT Structure Description

This table describes the SCALE_DDT structure:

Name	Type	Description
HighRangeRpm	INT	High range of input signal in rpm
LowRangeRpm	INT	Low range of input signal in rpm
HighRangeEngUnit	REAL	High range for the measurement in engineering units
LowRangeEngUnit	REAL	Low range for the measurement in engineering units

NOTE:

The following rules must be true:
- 0 <= LowRangeRpm < HighRangeRpm0
- 0 <= LowRangeEngUnit < HighRangeEngUnit

Input Detail - SetPoint

Description

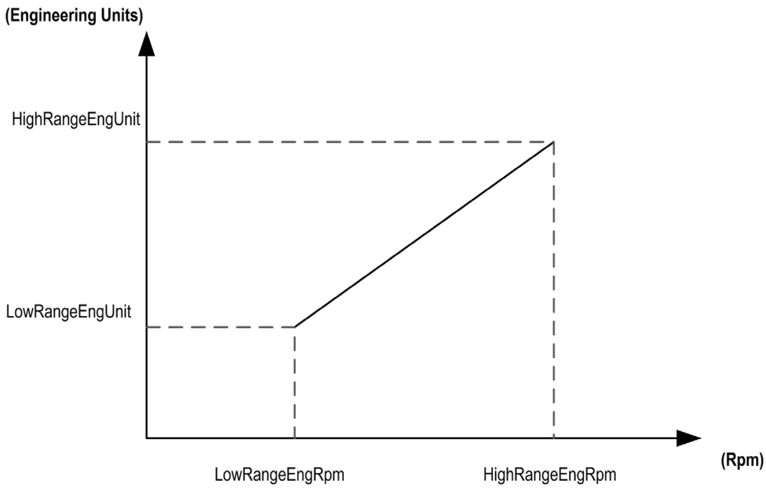
The speed set-point requested from the speed driver accepts only positive values.

It is measured in engineering units, which can be configured with the members of the input pin ‘Range’.

- HighRangeRpm
- LowRangeRpm
- HighRangeEngUnit
- LowRangeEngUnit

This variable must be within the correct range. The DFB makes the conversion between engineering units and drive speed RPMs.

This figure shows the input SetPoint:



4.3 Outputs

Overview

This section describes the outputs of ATV31Dev FBD.

What's in this Section?

This section contains the following topics:

Generic Outputs Description

Overview

This table is the same for all ATV Profile.

Outputs Description Table

This table describes the outputs of the ATVProfile FBD:

Output	Type	Description
Ready	BOOL	This output indicates if the device is enabled and free of detected errors. The device is ready to carry out or is carrying out a run or stop command. If QuickStop is enabled, this output is reset to 0. This variable is TRUE as long as there is no detected communications error and the device is in status 5 of the DSP402.
Fail	BOOL	This output indicates a detected error in the control block (parameter configuration, cannot be reset), detected error in the device, or a detected communications error. To reset the detected error, the ResetFail input has to be activated. The last detected error code is in FailCode.
Warning	BOOL	This output indicates if an alarm has been activated for the device. It does not affect the blocks operation and does not need to be reset. The signal remains active until the cause for the alarm disappears. The output is set to 0 only when the alarm is acknowledge.
FailCode	ARRAY	This output indicates the detection of an error. When this output is activated, this variable contains the code for the current detected error. If this output is not activated, this variable contains the last detected error. The detected error source is specified by a 3-level structure.
WarningCode	ATVWarning	The WarningCode *(see page 60)* output is a data structure that contains information about the alarm currently on the speed of the drive.
Running	BOOL	This output indicates that the speed of the drive is running and has an output frequency. **NOTE:** If SetPoint is 0 and Run is active, the Running signal is activated.

Output	Type	Description
ExtControlled	BOOL	Indicates if the drive is controlled from an external source (a console, a push-button panel or a monitoring system) to the system (1) or not (0). It provides information for programming.
Resetting	BOOL	The Resetting *(see page 61)* output indicates whether or not a reset is being carried out.
Status	ATVStatus	The Status *(see page 62)* output holds a structure with the information that the module extracts from the drives status variable.
PresentValue	REAL	Indicates the current rotation speed in EU (Engineering Units).
PresentSpeed	REAL	Indicates the current motor rotation speed in rpm.
NOTE: The variables can not be read from the device when there is a detected communications error. The variables keep their last value.		

ATV31Dev Specific Outputs

Outputs Description Table

This table describes the outputs of the ATV31Dev FBD:

Output	Type	Description
Current	REAL	Current motor current in %. This value can be found on all drives on CANopen networks and ATV71 and ATV61 drives on Ethernet networks.
OutputData	ATV31_OUT_DDT	OutputData *(see page 59)* holds a data array structure that contains data sent to the device. The drive can be controlled with this output variable.
ATV_IO	ATV_IO_DDT	ATV_IO *(see page 291)* is a device data structure that holds information for performing monitoring. The information is used by the operator screen and can be read from HMI or SCADA.

Output Detail - OutputData

Description

This function is a data array structure (ATV_STB_OUT_DDT) with data sent to the device. The speed of the drive can be controlled with this output variable.

It is reserved for the DFB and cannot be used directly.

ATV_STB_OUT_DDT Structure Description

This table describes the ATV_STB_OUT_DDT structure:

Name	Type	Description
OutputData [0]	INT	CWD (Control Word)
OutputData [1]	INT	Speed set-point
OutputData [2]	INT	OL1R (ATV digital outputs)
OutputData [3]	INT	AO1 (Analog output 1)

Output Detail - WarningCode

Description

This output is a data structure (ATVWarning) that contains information on the alarm currently on the speed of the drive.

ATVWarning Structure Description

This table shows the ATVWarning structure with TRUE values:

Name	Type	Description
Device	BOOL	Alarm present on device.
Order	BOOL	Follow-up alarm. The device is not responding to the control command (Run, QuickStop, etc.) within the time specified in CommandCtrlWindow.
ForcedLocalMode	BOOL	Device status is ForcedLocalMode because it has been forced locally. Control is though the wires attached to the screw terminals.

Output Detail - Resetting

Description

This output indicates whether or not a reset has been carried out.

The CommandCtrlWindow variable indicates the maximum time for resetting the detected error.

When a device or communications reset is carried out with ResetFail, the DFB tries to reset the detected error within the time period defined in CommandCtrlWindow.

If the detected error is reset:

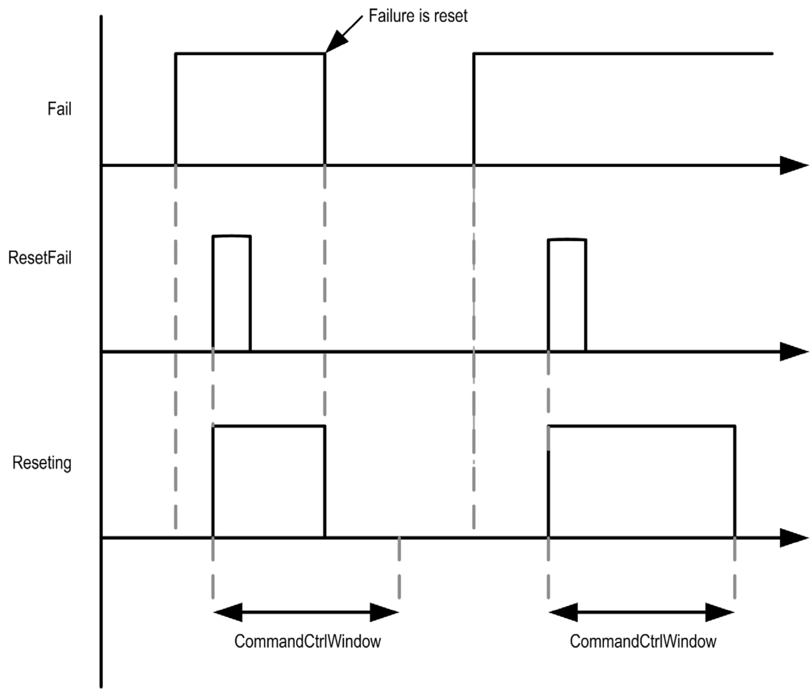
- The Fail and Resetting output variables are reset (they are set to FALSE).

If the detected error is not reset:

- The Resetting variable is set to FALSE.
- The Fail variable is held as TRUE.

ResetFail is edge-based.

This figure shows the Resetting output:



Output Detail - Status

Description

This output is a data structure (ATVStatus) that contains information that the module extracts from the speed drivers status variable.

ATVStatus Structure Description

This table shows the ATVStatus structure with TRUE values:

Name	Type	Description
ReadyToSwitchOn	BOOL	Ready to Switch ON (DRIVECOM [ETA.0])
SwitchedOn	BOOL	Switched ON ([ETA.1])
OperationEnabled	BOOL	Operation Enabled ([ETA.2])
Malfunction	BOOL	Detected device error ([ETA.3])
VoltageEnabled	BOOL	Voltage on device terminals ([ETA.4])
QuickStop	BOOL	Quick stop activated ([ETA.5])
SwitchOnDisabled	BOOL	Switch on Disabled ([ETA.6])
Alarm	BOOL	Alarm present on device ([ETA.7])
ForcedLocalMode	BOOL	Forced local mode ([ETA.9])
ReferenceReached	BOOL	Speed set-point has been reached ([ETA.10])
ReferenceExceeded	BOOL	Speed set-point outside limits ([ETA.11])
StompImposed	BOOL	Stop forced by remote control STOP key ([ETA.14])
ForwardReverseRotation	BOOL	Running direction (1=forward, 0=reverse) ([ETA.15])
State	INT	Numerical code of the speed drivers state
Info	INT	Code with information on statuses and required actions

State Integer Values Description

This table shows the State integer values:

Variable Value	Status
-2	Detected DFB error
-1	Not initialized, waiting for data
0	Disabled
2	Switch on Disabled (NST state)

Variable Value	Status
3	Ready to Switch On (NST state)
4	Switched On (NST state)
5	Operation Enabled (RDY status)
6	QuickStop (FST status)
8	Non-operational Device (FLT status)

Info Integer Values Description

This table shows the Info integer values:

Variable Value	Status
1	Detected error in parameter configuration
11	MISSING EnableDFB
12	Detected communications error
10	Waiting for device information
13	ETA value is 0
14	REMOVE Local forcing should be 0
21	MISSING EnableDevice
23	REMOVE Run should be 0
24	REMOVE ResetFail. Reset again
3	WAITING ReadyToSwitchOn
2	WAITING SwitchOnDisable
4	WAITING SwitchOn
5	WAITING OperationEnabled
6	WAITING QuickStopActive
55	WAITING speed-driver operation (Run)
51	Speed of the drive stopped
58	Speed set-point is 0
59	Speed outside limits
56	WAITING speed driver speed
52	Speed of the drive running with speed 0
53	Speed driver running
57	WAITING speed driver stop
54	Speed of the drive at rated running

Variable Value	Status
61	REMOVE QuickStop should be 0
62	QuickStop is activated
81	MISSING ResetFail, Detected speed driver error
82	DO speed driver reset
99	Unexpected state

4.4 Inputs/Outputs

Overview

This section describes the Inputs/Outputs of the ATV31Dev FBD.

What's in this Section?

This section contains the following topics:

Generic Inputs/Outputs Description

Overview

This table is the same for all ATV Profiles.

Inputs/Outputs Description Table

This table describes the inputs/outputs of the ATV Profile FBD:

Name	Type	Description
ATV_ST	ATV_ST_DDT	The ATV_ST *(see page 67)* device data structure contains the minimum information required for performing control and monitoring. The information is used by the operator screen and can be read from HMI or SCADA.
ATV_CFG	ATV_CFG_DDT	The ATV_CFG *(see page 69)* device data structure contains the information used by the operator screen and can be read/write from HMI or SCADA.

Inputs/Outputs Detail - ATV_ST

Description

The device data structure (ATV_ST_DDT) contains the minimum information required for performing control and monitoring.

This information is used by the operator screen and is read/write for HMI or SCADA.

ATV_ST_DDT Structure Description

This table shows the ATV_ST_DDT structure:

Name	Type	Description
STW	WORD	Device status
CFGW	WORD	Device control
PresentValue	REAL	Current speed value
SetPoint	REAL	Set-point value

STW Word Description

Access to the data in this bit word is read-only.

This table shows the STW Word description:

Bit	Description
0	Unknown technological module status. No variable refreshing.
1	Not Ready
2	Technological module running
3	Detected device error
4	Alarm on device or DFB (follow-up or screw terminal control)
5	Detected communications interruption
6	ResetFail required
7	ExtControlled
8	Running
9	EnableDFB
15	Direction of Rotation

CFGW Word Description

CFGW provides the means necessary to control the device.

Owner Value	Control is from
0	Supervision subsystem or from the operator screen
1	Supervision subsystem, the DFB input variables values are used for reading from HMI or SCADA

This table shows the CFGW Word Description:

Bit	Description
0	ResetFail
1	Owner
3	Direction
4	QuickStop
5	EnableDevice
6	Run
7	ControlCommand
8	ControlSetpoint

NOTE: The Owner bit enables control of the block from the •••_ST_DDT Inputs / Outputs structure, while ignoring the block input signals. It enables control from a monitoring system (HMI, SCADA, operator screen) in the manual mode without using the programmed switching operation.

Inputs/Outputs Detail - ATV_CFG

Description

This function is a data structure (ATV_CFG_DDT) that contains device information. The information is used by the operator screen and can be read from HMI or SCADA.

ATV_CFG_DDT Structure Description

This table shows the ATV_CFG_DDT structure:

Name	Type	Description
DataStatus	WORD	Information on the Device status
Info	WORD	Device information
State	INT	Speed of the drive status code
FailCode 0	INT	Code of last Level 0 detected error
FailCode 1	INT	Code of last Level 1 detected error
FailCode 2	INT	Code of last Level 2 detected error

DataStatus Word Description

DataStatus Word provides the device status and information on the status output structure.

This table shows the DataStatus Word description:

Bit	Status
0	ReadyToSwitchOn
1	SwitchedOn
2	OperationEnabled
3	Malfunction
4	VoltageEnabled
5	QuickStop
6	SwitchedOnDisabled
7	Alarm
8	ForcedLocalMode
9	ReferenceReached
10	ReferenceExceeded
11	StompImposed
12	ForwardReverseRotation

Info Word Description

This table shows the Info Word Description:

Name	Type	Description
State	INT	Status code information of the drive. Its value is Status.state.
FailCode 0	INT	Indicates which detected error has occurred (FailCode[0]).
FailCode 1	INT	Indicates which detected error has occurred (FailCode[1]).
FailCode 2	INT	Indicates which detected error has occurred (FailCode[2]).

4.5 Operator Screen

General

Description

The component includes an operator screen that enables interaction with the communication interface blocks.

The operator screen data controls a monitoring system:

- ATV_ST_DDT and ATV_CFG_DDT - Inputs/Outputs structures.
- ATV_IO_DDT and ATV_IOEXT_DDT depending on the component - Output structures.

4.6 Error Management

Overview

This section describes the Error Management of ATV31Dev FBD.

What's in this Section?

This section contains the following topics:

Detected Error Code

Description

The detected error codes the device can return are read from the FailCode output variable.

The detected error codes are the same as for the ATVDevError Management *(see page 74)*.

Detected Error Code Description

Overview

The detected error codes the device can return are read from the FailCode output variable.

Parameter Configuration Detected Error Codes

This detected error indicates that a public variable for parameter configuration has a value that is not allowed.

To reload new values, an edge is required on the EnableDFB variable that cannot be reset:

- FailCode[0]: 16#0003
- FailCode[1]: 16#0000
- FailCode[2]: 16#0004

The Failcode[0] variable can have the following codes:

Error Code	Symbol	Meaning
16#0000	nOF	No detected error
16#1000	CrF, OLF, SOF	Detected Error in capacitor pre-charge Motor overload Overspeed
16#2310	OCF	Over current
16#2320	SCF	Short circuit impedance Power stage detected error
16#2330	SCF	Motor ground detected error
16#2340	SCF	Short-circuit between motor phases
16#3110	OCF	Supply over voltage detected error
16#3120	USF	Low bus voltage
16#3130	PHF	Phase dropout detected error
16#3310	ObF, OPF	Over voltage on DC bus Motor phase dropout
16#4210	OHF	Drive overheating
16#5520	EEF	EEPROM memory detected error
16#6100	InF	Internal detected error
16#6300	CFF, CFI	Incorrect (parameter) configuration or invalid parameters
16#7300	LFF	4...20 mA AI3 input detected error
16#7510	SLF	Modbus detected error
16#8100	COF	CANopen detected error

Error Code	Symbol	Meaning
16#9000	EPF	External detected error
16#FF00	tnF	Auto-tuning detected error
16#FF01	bIF	Detected error in brake control unit

The device interruption can be reset as long as ControlCommand=TRUE. Otherwise, it can be reset the speed of the drive or through the screw terminal with the appropriate parameter configuration.

Modbus Communications Detected Error

For Modbus communications, this code is used to indicate that communications have not been established. This code can be reset.

- FailCode[0]: 16#0002
- FailCode[1]: 16#0000
- FailCode[2]: 16#0004

Once Modbus communications have been established, check "Modbus Client" error codes for FailCode[0] and [1]. All components make a distinction between read and write requests:

- FailCode[2]: 16#0001 Read
- FailCode[2]: 16#0002 Write

Error Code Example

For a detected error, the code is:

- FailCode[1]: 16#0000
- FailCode[2]: 16#0005

NOTE: The error codes of the devices are available for all speed drivers except for those that communicate through the Advantys STB island.

4.7 SCoD Representation and Parameter Description

SCoD Representation and Parameter Description

Overview

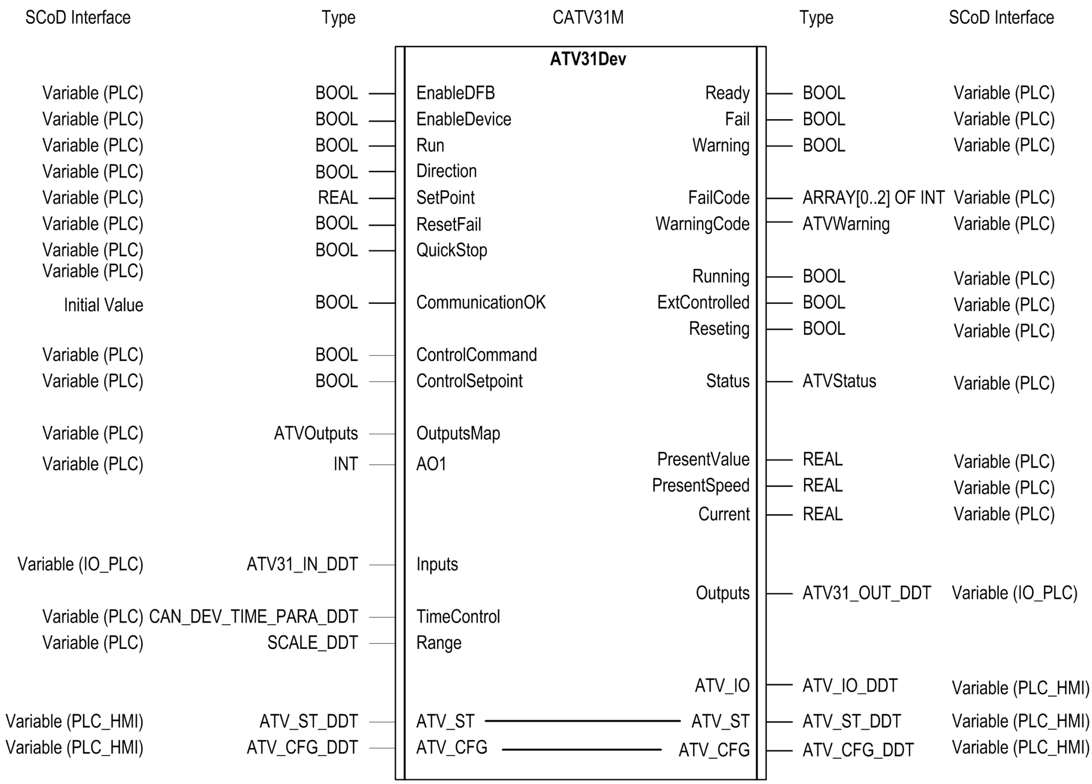
The usage of the DFB pins in the SCoD editor is explained in the input/output diagram below.

The variables defined in the SCoD editor have in principle the same name as the pin (except that an underscore in the pin name is skipped for the variable name).

If the variables defined in the SCoD have the property ’Initial Value’ then the values of the instantiated SCoD variables can be modified manually in Unity Pro before PLC restart.

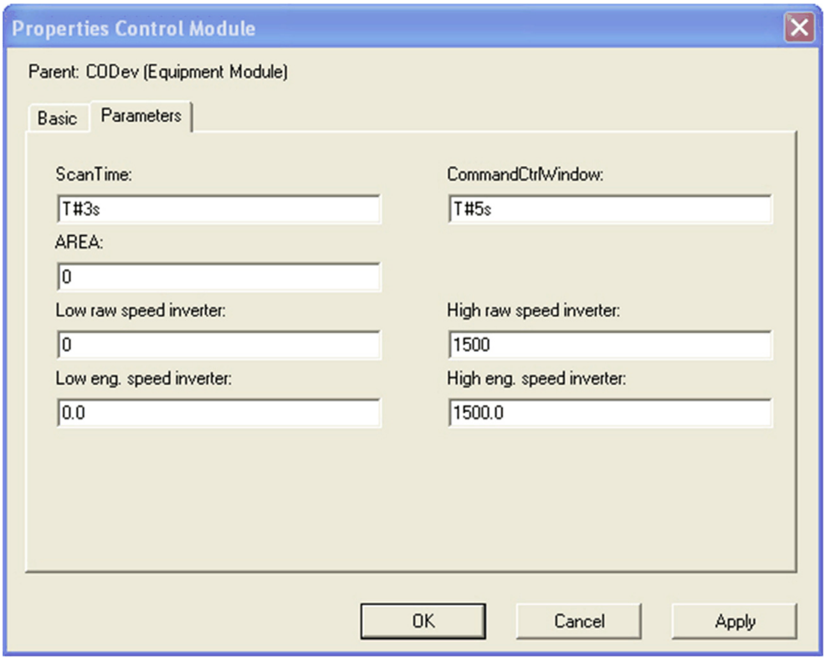
SCoD Representation

This figure represents the CATV31M SCoD:



Parameter Tab

This figure shows the CATV31M SCoD parameter tab:



The following table describes the properties:

Property	**Type**	**Description**
ScanTime	TIME	Minimum time to maintain detecetd error signals
CommandControlWindow	TIME	The time waited before executing orders
AREA	BYTE	Area to which the object that you wish to instantiate belongs. Enables to control access to the functions provided by this object.
Low raw Speed Inverter	INT	Low range value measured in raw units
High raw Speed Inverter	INT	High range value measured in raw units
Low eng. Speed Inverter	REAL	Low range value measured in engineering units
High eng. Speed Inverter	REAL	High range value measured in engineering units

4.8 HMI Representation

ATV31 Drive - Representation in the HMI

CANOpen ATV31 (Genie)

The CANOpen ATV31 block is represented in the HMI by the following genie.

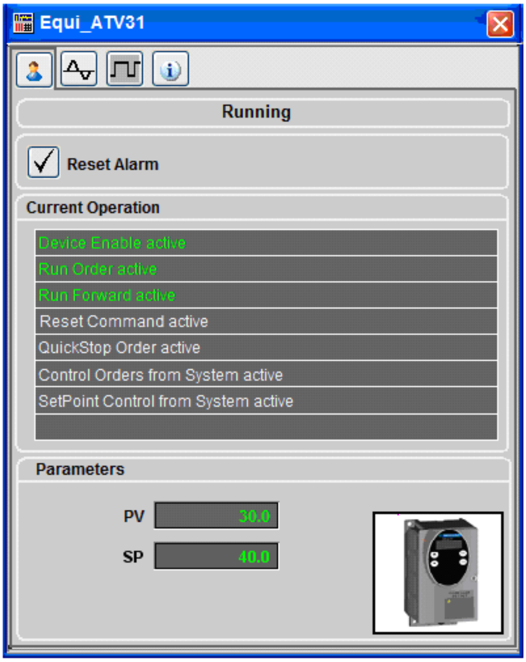
Genie Elements

Symbol	Meaning
	Opens the CANOpen ATV31 popup. Displays the device name, that is, ATV and PV value.
	The instance name of the CANOpen ATV31 block is only visible by selecting the **Hide**/**Unhide** button in the template.
	When there is an unacknowledged note, the ! symbol is displayed in the genie.
	When there is an active and unacknowledged alarm or communication failure, then outer rectangle flashes in red.
	When there is an active and unacknowledged alarm or communication failure, then outer rectangle is steady red.

Popup

The CANOpen ATV31 block popup has the following tabs:
- **Operator** tab
- **Analog** tab
- **Digital** tab
- **Information** tab

Operator Tab



This tab has 3 sub-sections:
- **Status**
- **Reset Alarm**
- **Current Operation**
- **Parameters**

The sub-section **Status** displays the current status.

The possible displays are listed as follows:
- **Unknown Status**
- **Not Ready**
- **Running**
- **Communication Failure**
- **Failure**
- **Warning**

The following table describes the sub-section **Reset Alarm**:

Item	Description
Reset Alarm	Resets alarm.

The following table describes the sub-section **Current Operation**:

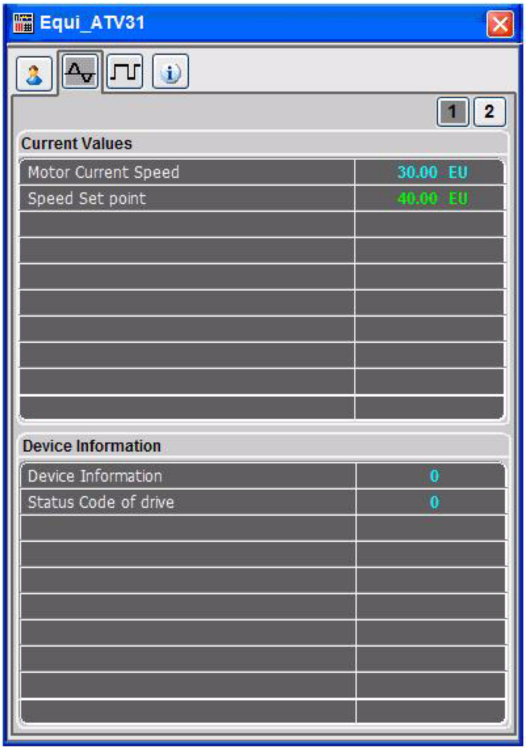
Item	Description
Device Enable active	Displays the text in green color when the operation is active. Displays the text in white color when the operation is not active.
Run Order active	
Run Forward active	
Reset Command active	
QuickStop Order active	
Control Orders from System active	
SetPoint Control from System active	

The following table describes the sub-section **Parameters**:

Item	Description
PV	Displays the present value.
SP	Displays the set point value.

The sub-section **Parameters** also shows the picture of ATV.

Analog Tab



This tab has 2 sub-tabs:

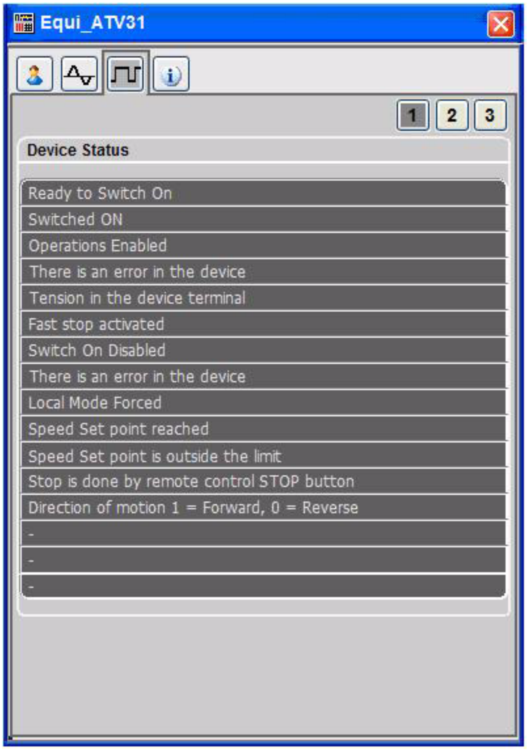
- **1**
- **2**

Each sub-tab has 2 sub-sections:

- **Current Values**
- **Device Information**

The title of these sections are configurable in the *cem_devices.dbf* file. The analog parameters and their values are displayed in these sub-sections.

Digital Tab



This tab has 3 sub-tabs:

- **1**
- **2**
- **3**

Each sub-tab has a sub-section, which shows the device status. The digital parameter descriptions are displayed in this sub-section.

When the status is not active, the text is displayed in white color. When the status is active, the text is displayed in green color. The text is configurable in the *cem_devices.dbf* file.

Information Tab

For description about **Information** tab, refer to Commonly Used Tabs *(see page 32)*.

ATV7161 - CANopen ATV Drive (CATV61M and CATV71M SCoDs)

5

Overview

This chapter describes the DFB for the control blocks for variable speed drives on a CANopen network, represented by CATV61M and CATV71M SCoDs.

What's in this Chapter?

This chapter contains the following sections:

5.1 Description

Overview

This section describes the ATV7161Dev DFB.

What's in this Section?

This section contains the following topics:

Functional Description

General

The ATV7161Dev profile is the control block for the CATV71 and CATV61 on an Advantys I/O island.

The ATV7161Dev block is implemented as listed below:

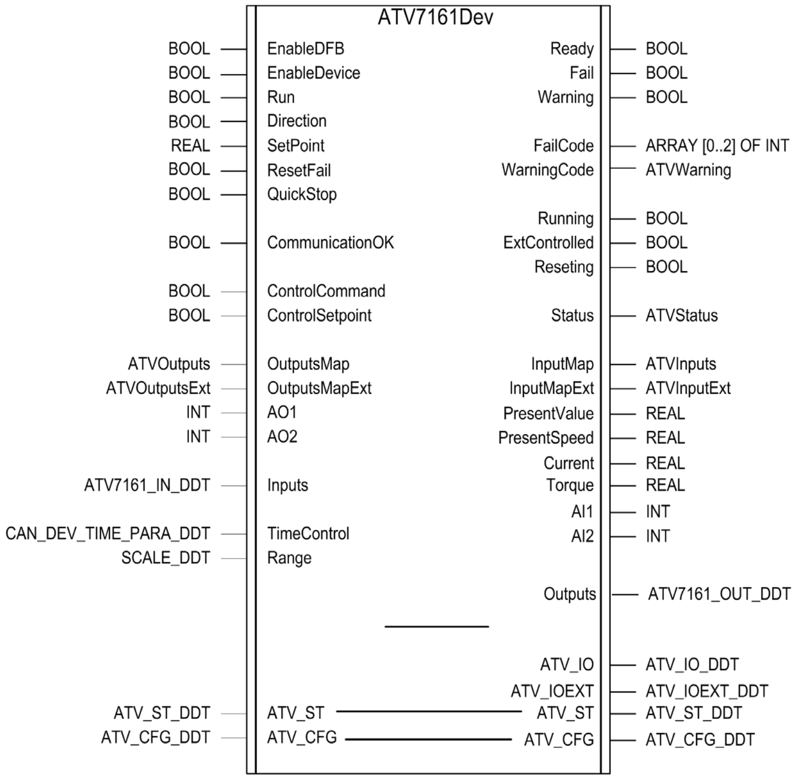
Software	Implemented as
Unity Pro	FBD ATV7161Dev
Vijeo Citect	DPC_Devices
UAG	Depending on the speed drive: • SCoD CATV61 • SCoD CATV71

The main function blocks are described in the ATV Profile Functional Description *(see page 29)*.

FBD Representation

Representation in FBD

The functional module that will be used in the program has the following aspect (at the FBD block level) after PLC generation in UNITY Pro:



5.2 Inputs

Overview

This section describes the inputs of the ATV7161Dev FBD.

What's in this Section?

This section contains the following topics:

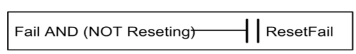
Generic Inputs Description

Overview

This table is the same for all ATV Profiles.

Inputs Description Table

This table describes the generic inputs of the ATV Profile FBD:

Input	Type	Description
EnableDFB	BOOL	This input enables the normal execution of the control block. EnableDFB = 0. The entire DFB is restarted (statuses, output values, counters, etc. are lost) and all output values are set to 0 or FALSE. EnableDFB = 1. This input enables communications with devices for their operation. Public variable values are loaded during the first enabling cycle.
EnableDevice	BOOL	This input variable is true only if EnableDFB is true. To control the speed of the drive, it must be enabled.
Run	BOOL	This input makes the motor run in the direction selected with the Direction variable.
Direction	BOOL	This input activates the direction of motor rotation: **1 =** forward **0 =** reverse **NOTE:** Setpoint assumes positive values. Changes in the direction of rotation cannot be made with Setpoint sign.
Setpoint	REAL	This is the speed set-point *(see page 54)* requested from the speed of the drive. It can only have positive, real values.
ResetFail	BOOL	Existing detected errors are reset on the rising edge of this signal. Both communications and detected device errors can be reset. If the detected error is a device error, it sends a reset command to the device if ControlCommand = TRUE. This figure shows the input ResetFail: 
QuickStop	BOOL	This input enables a quick stops the drive speed. This variable is state-based. If there is a QuickStop, the Run bit has to be reset to resume operation.

Input	Type	Description
ControlCommand	BOOL	This input indicates if the starter is controlled from an external source. If ControlCommand=TRUE the device is controlled from an external source. The generation of follow-up alarms is disabled. If ControlCommand=FALSE, the block only performs read operations on the device status, the transmission of device control or reset commands is not possible. **NOTE:** This input does not configure the starter; it only defines the DFB operating mode.
ControlSetPoint	BOOL	This input indicates if the speed of the drive is controlled from an external source. (e.g., from the console or a potentiometer)

⚠ WARNING

UNEXPECTED MOTOR START

Reset the Run variable before resuming operation.

The Run input must be inactive while the device is being enabled or while a detected error is being reset.

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

NOTICE

UNEXPECTED BEHAVIOUR

The ControlCommand input value must coincide with the speed driver configuration.

Failure to follow these instructions can result in equipment damage.

ATV7161Dev Specific Inputs

Inputs Description Table

The below table describes the inputs of the ATV7161Dev FBD:

Input	Type	Description
CommunicationOK	BOOL	A bit is used to indicate whether or not the node is present on the bus.
AO1	INT	Controls the value of the AO1 analog output. This value can be found on all devices on CANopen and Ethernet networks. The speed driver analog output can only be controlled with this input variable. It cannot be controlled from HMI/SCADA.
AO2	INT	This input Controls the value of the AO2 analog output. This value can be found on ATV71 and ATV61 on CANopen and Ethernet networks. The speed driver analog output can only be controlled with this input variable. It cannot be controlled from HMI/SCADA.
OuputsMap	ATVOutputs	OutputsMap *(see page 283)* holds a data structure used to control the speed driver outputs. Information available on CANopen and Ethernet. The speed driver outputs can only be controlled with this input variable. They cannot be controlled from HMI/SCADA.
OuputsMapExt	ATVOutputsExt	OutputsMapExt *(see page 284)* holds a data structure used to control the drive outputs for ATV71 and ATV61 extended cards on CANopen and Ethernet networks. The drive outputs can only be controlled with this input variable.(They cannot be controlled from HMI/SCADA).
Inputs	ATV7161_IN_DDT	Inputs *(see page 49)* holds a data array structure that contains data obtained from the device. The drive can be controlled with this input variable. The input is reserved for the DFB and cannot be used directly. Allocate the structure (%MWx) to allow correct operation of the control block.

Input	Type	Description
TimeControl	CAN_DEV_ TIME_PARA_DDT	TimeControl *(see page 51)* is a parameter for device control.
Range	SCALE_DDT	Range *(see page 53)* definition for inverter (Raw and engineering range).

5.3 Outputs

Overview

This section describes the outputs of the ATV7161Dev FBD.

What's in this Section?

This section contains the following topics:

Generic Outputs Description

Overview

This table is the same for all ATV Profiles.

Outputs Description Table

This table describes the outputs of the ATVProfile FBD:

Output	Type	Description
Ready	BOOL	This output indicates if the device is enabled and free of detected errors. The device is ready to carry out or is carrying out a run or stop command. If QuickStop is enabled, this output is reset to 0. This variable is TRUE as long as there is no detected communications error and the device is in status 5 of the DSP402.
Fail	BOOL	This output indicates a detected error in the control block (parameter configuration, cannot be reset), detected error in the device, or a detected communications error. To reset the detected error, the ResetFail input has to be activated. The last detected error code is in FailCode.
Warning	BOOL	This output indicates if an alarm has been activated for the device. It does not affect the blocks operation and does not need to be reset. The signal remains active until the cause for the alarm disappears. The output is set to 0 only when the alarm is acknowledge.
FailCode	ARRAY	This output indicates the detection of an error. When this output is activated, this variable contains the code for the current detected error. If this output is not activated, this variable contains the last detected error. The detected error source is specified by a 3-level structure.
WarningCode	ATVWarning	The WarningCode *(see page 60)* output is a data structure that contains information about the alarm currently on the speed of the drive.
Running	BOOL	This output indicates that the speed of the drive is running and has an output frequency. **NOTE:** If SetPoint is 0 and Run is active, the Running signal is activated.

Output	Type	Description
ExtControlled	BOOL	Indicates if the drive is controlled from an external source (a console, a push-button panel or a monitoring system) to the system (1) or not (0). It provides information for programming.
Resetting	BOOL	The Resetting *(see page 61)* output indicates whether or not a reset is being carried out.
Status	ATVStatus	The Status *(see page 62)* output holds a structure with the information that the module extracts from the drives status variable.
PresentValue	REAL	Indicates the current rotation speed in EU (Engineering Units).
PresentSpeed	REAL	Indicates the current motor rotation speed in rpm.
NOTE: The variables can not be read from the device when there is a detected communications error. The variables keep their last value.		

ATV7161Dev Specific Outputs

Outputs Description Table

This table describes the general outputs of the ATV7161Dev FBD.

Output	Type	Description
InputsMap	ATVInputs	The OutputsMap *(see page 289)* holds a data structure that contains information on the state of the drive inputs CANopen and Ethernet communications.
InputsMapExt	ATVInputsExt	TheOutputMapExt *(see page 290)* holds a data structure that contains information on the state of the drive inputs for ATV71 and ATV61 extended cards on CANopen and Ethernet networks.
ATV_IO	ATV_IO_DDT	The ATV_IO *(see page 291)* device data structure holds information for performing monitoring. The information is used by operator screen and can be read from HMI or SCADA
ATV_IOEXT	ATV_IOEXT_DDT	The ATV_IOEXT *(see page 292)* output holds a data structure that contains information for performing monitoring. The information is used by operator screen and can be read from HMI or SCADA.
Current	REAL	Current motor current in %. This value can be found on all drives on CANopen networks, and ATV71 and ATV61 on Ethernet networks.
Torque	INT	Current motor torque in 0.1A increments.
AI1	INT	Value of the AI1 analog device input. This value can be found on all devices on CANopen networks, and ATV71 and ATV61 on Ethernet networks.
AI2	INT	Value of the AI2 analog device input. This value can be found on ATV71, ATV61 on CANopen and Ethernet networks.
OutputData	ATV6171_OUT_DDT	OutputData *(see page 59)* holds a data array structure with contains data sent to the device. The drive can be controlled with this output variable. This output is reserved for the DFB and cannot be used directly. Allocate the structure (%MWx) in order to ensure the control blocks correct operation.

5.4 Inputs/Outputs

Generic Inputs/Outputs Description

Overview

This table is the same for all ATV Profile.

Inputs/Outputs Description Table

This table describes the inputs/outputs of the ATV Profile FBD:

Name	Type	Description
ATV_ST	ATV_ST_DDT	The ATV_ST *(see page 67)* device data structure contains the minimum information required for performing control and monitoring. The information is used by the operator screen and can be read from HMI or SCADA.
ATV_CFG	ATV_CFG_DDT	The ATV_CFG *(see page 69)* device data structure contains the information used by the operator screen and can be read/write from HMI or SCADA.

5.5 Operator Screen

General

Description

The component includes an operator screen that allows interaction with the communication interface blocks.

- ATV_ST_DDT and ATV_CFG_DDT - Inputs/Outputs structures.
- ATV_IO_DDT and ATV_IOEXT_DDT - Output structures.

5.6 Error Management

Detected Error Code

Description

The detected error codes the device can return are read from the FailCode output variable. The detected error codes are the same as for the ATVDev Error Management *(see page 74)*.

5.7 SCoD Representation and Parameter Description

SCoD Representation and Parameter Description

Overview

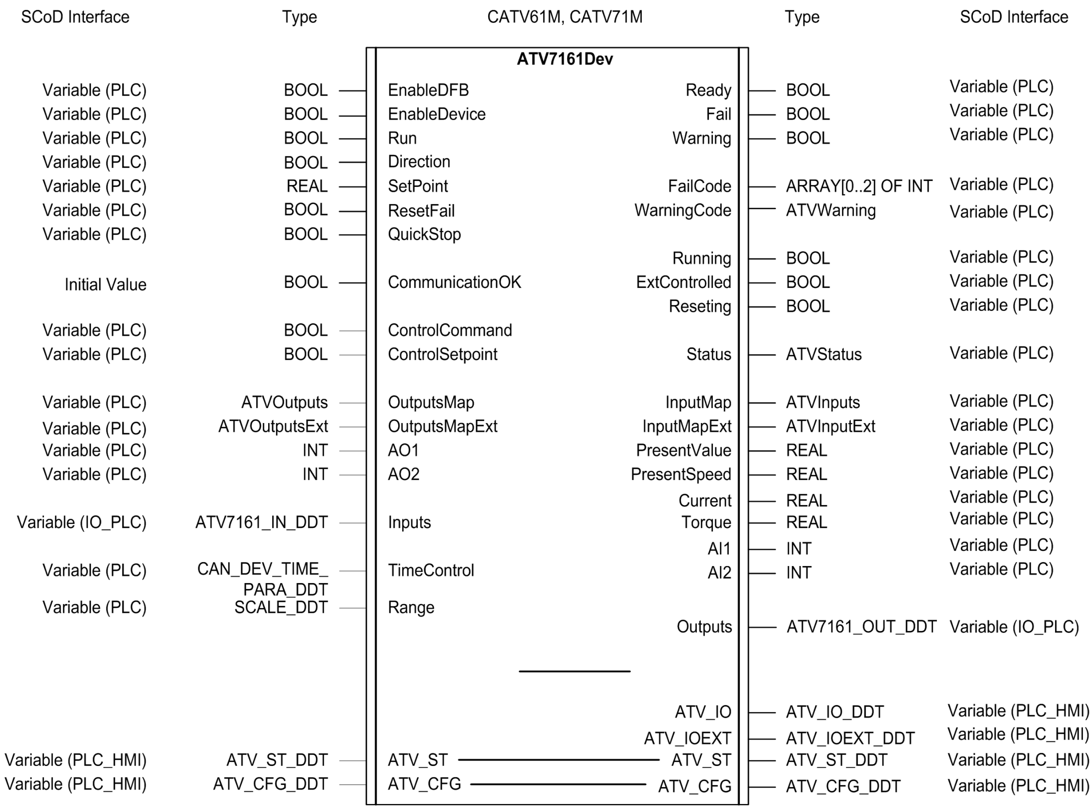
The usage of the DFB pins in the SCoD editor is explained in the input/output diagram below.

The variables defined in the SCoD editor have in principle the same name as the pin (except that an underscore in the pin name is skipped for the variable name).

If the variables defined in the SCoD have the property 'Initial Value' then the values of the instantiated SCoD variables can be modified manually in Unity Pro before PLC restart.

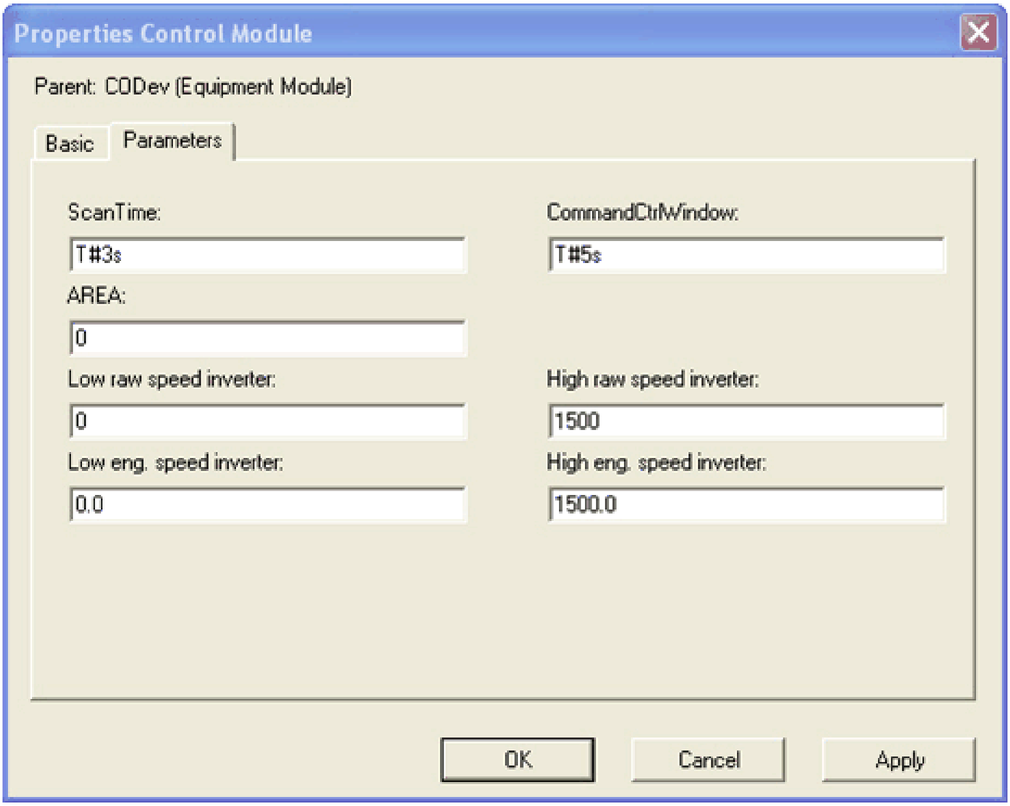
SCoD Representation

This figure represents the CATV61 and CATV71 SCoDs:



Parameter Tab

This figure shows the CATV61 and CATV71 SCoDs parameter tab:



The following table describes the properties:

Property	Type	Description
ScanTime	TIME	Minimum time to maintain detected error signals
CommandControlWindow	TIME	The time waited before executing orders
AREA	BYTE	Area to which the object that you wish to instantiate belongs. Enables to control access to the functions provided by this object.
Low raw Speed Inverter	INT	Low range value measured in raw units
High raw Speed Inverter	INT	High range value measured in raw units
Low eng. Speed Inverter	REAL	Low range value measured in engineering units
High eng. Speed Inverter	REAL	High range value measured in engineering units

5.8 HMI Representation

ATV7161 Drive - Representation in the HMI

CANOpen ATV (Genie)

The CANOpen ATV block is represented in the HMI by the following genie.

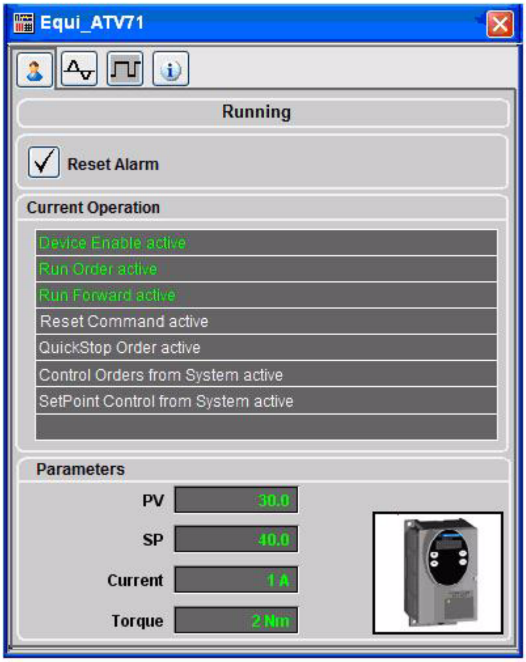
Genie Elements

Symbol	Meaning
	Opens the CANOpen ATV popup. Displays the device name, that is, ATV and PV value.
	The instance name of the CANOpen ATV block is only visible by selecting the **Hide**/**Unhide** button in the template.
	When there is an unacknowledged note, the ! symbol is displayed in the genie.
	When there is an active and unacknowledged alarm or communication failure, then outer rectangle flashes in red.
	When there is an active and acknowledged alarm or communication failure, then outer rectangle is steady red.

Popup

The CANOpen ATV block popup has the following tabs:
- **Operator** tab
- **Analog** tab
- **Digital** tab
- **Information** tab

Operator Tab



This tab has 3 sub-sections:

- **Status**
- **Reset Alarm**
- **Current Operation**
- **Parameters**

The sub-section **Status** displays the current status.

The possible displays are listed as follows:

- **Unknown Status**
- **Not Ready**
- **Running**
- **Communication Failure**
- **Failure**
- **Warning**

The following table describes the sub-section **Reset Alarm**:

Item	Description
Reset Alarm	Resets alarm.

The following table describes the sub-section **Current Operation**:

Item	Description
Device Enable active	Displays the text in green color when the operation is active. Displays the text in white color when the operation is not active.
Run Order active	
Run Forward active	
Reset Command active	
QuickStop Order active	
Control Orders from System active	
SetPoint Control from System active	

The following table describes the sub-section **Parameters**:

Item	Description
PV	Displays the present value (speed/EU).
SP	Displays the set point value.
Current	Displays the value of current.
Torque	Displays the value of torque.

The sub-section **Parameters** also shows the picture of `ATV`.

Analog Tab

For detailed description, refer to Analog Tab *(see page 82)*.

Digital Tab

For detailed description, refer to Digital Tab *(see page 83)*.

Information Tab

For description about **Information** tab, refer to Commonly Used Tabs *(see page 32)*.

TESYST - CANopen Tesys T Motor Management (CTESYST SCoD)

6

Overview

This chapter describes the DFB for the control block for the Tesys T motor controller on CANopen network, represented by CTESYST SCoD.

⚠ WARNING

MISAPPLICATION OF MODULES

The modules in this section do not reflect any specific installation. Before adopting these modules for use in a specific application, the engineer must:

- conduct a safety analysis for the application and equipment installed
- verify that all modules are appropriate for the equipment or function in the installation
- supply appropriate parameters, particularly for limits
- check that all sensors and actuators are compatible with the modules selected
- thoroughly test all functions during verification and commissioning
- provide independent paths for critical control functions (emergency stop, over-limit conditions etc.) according to the safety analysis and applicable codes and regulations.

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

What's in this Chapter?

This chapter contains the following sections:

6.1 Description

Overview

This section describes the TesysTDev DFB.

What's in this Section?

This section contains the following topics:

Functional Description

General

TesysTDev block is the control block for TesysT controller on an Advantys STB I/O island and CANopen communications.

The TesysTDev block is implemented as listed below:

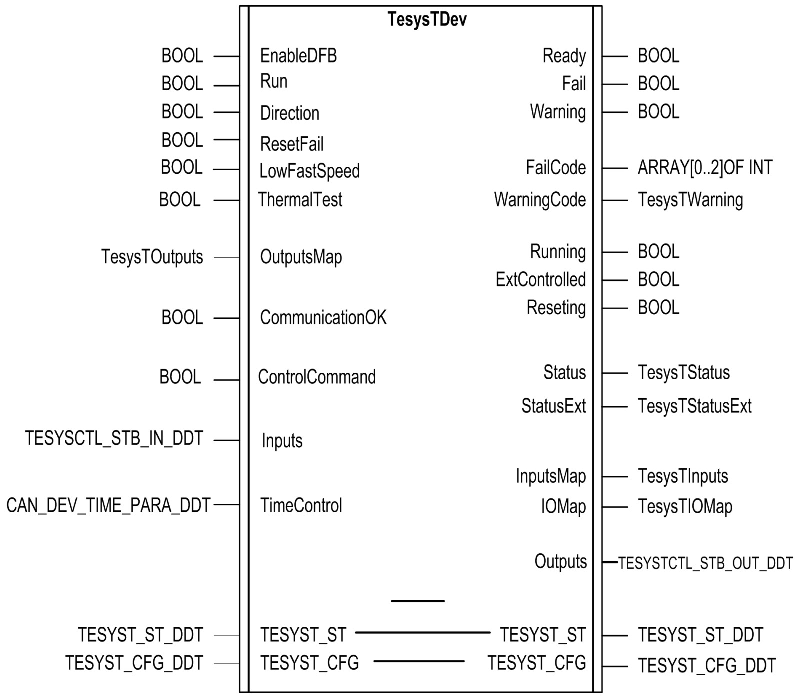
Software	**Implemented as**
Unity Pro	DFB TesysTDev
Vijeo Citect	DPC_Devices
UAG	Depending on the communication technology: • SCoD ASTESYST for Advantys. • SCoD CTESYST for CANopen.

The main function blocks are described in the TesysT Profile Functional Description *(see page 30)*.

FBD Representation

Representation in FBD

The functional module that will be used in the program has the following aspect (at the FBD block level) after PLC generation in UNITY Pro:



6.2 Inputs

Overview

This section describes the inputs of the TesysTDev FBD.

What's in this Section?

This section contains the following topics:

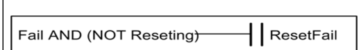
Generic Inputs Description

Overview

This table is the same for all TesysT Profiles.

Inputs Description Table

This table describes the inputs of the TesysT Profile FBD:

Input	Type	Description
EnableDFB	BOOL	This input enables the normal execution of the control block. EnableDFB = 0. The entire DFB is restarted (statuses, output values, counters, etc. are lost) and all output values are set to 0 or FALSE. EnableDFB = 1. This input enables communications with devices for their operation. Public variable values are loaded during the first enabling cycle.
Run	BOOL	This input makes the motor run in the direction selected with the Direction variable.
Direction	BOOL	This input activates the direction of motor rotation: **1 =** forward **0 =** reverse **NOTE:** Setpoint assumes positive values. Changes in the direction of rotation cannot be made with Setpoint sign.
ResetFail	BOOL	Existing detected errors are reset on a rising edge of this signal. Both Communications and detected device errors can be reset. If the detected error is a device error, it sends a reset command to the device if ControlCommand = TRUE. This figure shows the input ResetFail: 
LowFastSpeed	BOOL	Slow/fast speed of rotation. For 2-speed motor start configurations. Check the TesysT user manual.
ThermalTest	BOOL	This bit is used to check the forced thermal detected error input on the Tesys T.

Input	Type	Description
OutputsMap	TesysUOutputsMap	OutputsMap *(see page 116)* holds a structure used to control the Tesys outputs. If one of the variables is not available, the comment will indicate that the information cannot be accessed.
ControlCommand	BOOL	This input indicates if the starter is controlled from an external source. If ControlCommand=TRUE the device is controlled from an external source. The generation of follow-up alarms is disabled. If ControlCommand=FALSE, the block only performs read operations on the device status, the transmission of device control or reset commands is not possible. **NOTE:** This input does not configure the starter; it only defines the DFB operating mode.
TimeControl	CAN_DEV_TIME_ PARA_DDT	TimeControl *(see page 119)* is a parameter for device control.

⚠ WARNING

UNEXPECTED MOTOR START

Reset the Run variable before resuming operation.

The Run input must be inactive while the device is being enabled or while a detected error is being reset.

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

NOTICE

UNEXPECTED BEHAVIOUR

The ControlCommand input value must coincide with the speed driver configuration.

Failure to follow these instructions can result in equipment damage.

Specific TesysTDev Inputs

Inputs Description Table

This table describes the inputs of the TesysTDev FBD:

Input	Type	Description
CommunicationOK	BOOL	A bit is used to indicate whether or not the node is present on the bus.
InputData	TesysTCTL_STB_ IN_DDT TesysT_IN_DDT	InputData *(see page 117)* holds a data array structure that contains data obtained from the device. The TesysT can be controlled with this input variable. The input is reserved for the DFB and cannot be used directly.

Input Detail - OutputsMap

Description

This function holds a data structure (TesysTOutputsMap) used to control actuator outputs.

These outputs hold an image of this structure values.

TesysTOutputsMap Structure Description

This table shows the TesysTOutputsMap structure:

Name	Type	Description
Output1	BOOL	Controls the state of the output1
Output2	BOOL	Controls the state of the output2
Output3	BOOL	Controls the state of the output3
Output4	BOOL	Controls the state of the output4

Input Detail - InputData

Description

This function holds a data array structure that contains data obtained from the device.

The TesysT controller can be controlled with this input variable.

The function is reserved for the DFB and cannot be used directly.

Allocate the structure (%MWx) to allow the correct operation of the control block. Refer to Communication Components *(see page 303)*.

Depending on the device and the communications used, one of the following structures is used.
TESYSTCTL_IN_DDTT Information available on CANopen for Controllers.
TESYST_IN_DDT Information available on Advanced STB for Controllers.

TESYST_IN_DDT Structure Description

Information available on Advanced STB for Controllers:

Name	Type	Description
InputData [0]	INT	Status register 1
InputData [1]	INT	Status register 2
InputData [2]	INT	Logic input status
InputData [3]	INT	Logic Output status
InputData [4]	INT	Warning register
InputData [5]	INT	Detected Fault code
InputData [6]	INT	Average Current
InputData [7]	INT	Thermal Capacity
InputData [8	INT	FLC Ground current
InputData [9]	INT	Current Phase
InputData [10]	INT	Frequency
InputData [11	INT	Starts Count

TESYSTCTL_STB_IN_DDT Structure Description

Information available on CANopen for Controllers:

Name	Type	Description
InputData [0]	INT	Status register 1
InputData [1]	INT	Status register 2
InputData [2]	INT	Logic input status
InputData [3]	INT	Logic output status

Input Detail - TimeControl

Description

The TimeControl variable is the time parameter for device control.

The input type varies in accordance with the communication technology:
- CAN_DEV_TIME_PARA_DDT with CANopen/Advantys/Ethernet.
- MB_DEV_TIME_PARA_DDT with Modbus.

CAN_DEV_TIME_PARA_DDT Structure Description

This table shows the CAN_DEV_TIME_PARA_DDT structure:

Name	Type	Description
CommandCtrlWindow	TIME	Control time for operations. This is the time that the block waits for operations to be carried out by the device. A “follow-up alarm” is issued, if a command is sent and executed within the time indicated by this variable. The commands controlled are: ● EnableDevice ● Run In the event of a ResetFail, this is not interpreted as an alarm. Instead, the detected error continues and has to be reset the Resetting output (the time the device waits before executing orders).
ScanTime	TIME	Enables to configure the time that the alarm signals are kept active. The monitoring subsystem will acquire the data for alarms automatically.

MB_DEV_TIME_PARA_DDT Structure Description

This table shows the MB_DEV_TIME_PARA_DDT structure:

Name	Type	Description
CommandCtrlWindow	TIME	Control time for operations. This is the time that the block waits for operations to be carried out by the device. A “follow-up alarm” is issued, if a command is sent and executed within the time indicated by this variable. The commands controlled are: ● EnableDevice ● Run In the event of a ResetFail, this is not interpreted as an alarm. Instead, the error continues and has to be reset the Resetting output (the time the device waits before executing orders).
ScanTime	TIME	Enables to configure the time that the alarm signals are kept active. The monitoring subsystem will acquire the data for alarms automatically.
Refresh	TIME	Time to refresh the cyclic data.

6.3 Outputs

Overview

This section describes the outputs of the TesysTDev FBD.

What's in this Section?

This section contains the following topics:

Generic Outputs Description

Overview

This table is the same for all TesysT Profiles.

Outputs Description Table

This table describes the Outputs of the TesysT Profile FBD:

Output	Type	Description
Ready	BOOL	This output indicates if the device is enabled and free of errors. The device is ready to carry out or is carrying out any run or stop command. If QuickStop is enabled, this output is reset to 0.
Fail	BOOL	This output indicates an detected error in the control block (Parameter configuration, cannot be reset) or detected error in the device, or a detected communications error. In order to reset the detected error, the ResetFail input has to be activated. The last detected error code is shown on FailCode.
Warning	BOOL	This output indicates if an alarm has been activated for the device. It does not affect the blocks operation and does not need to be reset. The signal remains active until the cause for the alarm disappears.
FailCode	ARRAY	This output indicates the detection of an error. When this output is activated, this variable contains the code for the current detected error. If this output is not activated, this variable holds the last detected error. The detected error source is specified by a 3-level structure.
WarningCode	TesysTWarning	The WarningCode *(see page 125)* output holds a data structure that contains information about the alarm currently on the controller.
Running	BOOL	This output indicates that the Tesys is running. Current higher than 10% of FLA for Advanced and Multifunction controllers.
ExtControlled	BOOL	This output indicates if the device is controlled from an external source (for example, from the console, from a push-button panel, or from the monitoring system) to the system (1) or not (0). Provides information for programming.

Output	Type	Description
Reseting	BOOL	The Reseting *(see page 126)* output indicates whether or not a reset is carried out.
Status	TesysUStatus	The Status *(see page 127)* output holds a structure with the information that the module extracts from the status variable of controllers with Advanced and Multifunction control modules.
IOMap	TesysTIOMap	The IOMap *(see page 130)* indicates current speed of rotation in EU (Engineering Units).
NOTE: The variables can not be read from the device when there is a detected communications error. The variables keep their last value.		

Specific TesysTDev Outputs

Outputs Description Table

This table describes the Outputs of the TesysTDev FBD:

Output	Type	Description
OutputData	TESYSTCTL_ STB_OUT_DDT TESYST_OUT_ DDT	OutputData *(see page 131)* holds a data structure that contains data sent to the device. The controller can be controlled with this output variable. The output is reserved for the DFB and cannot be used directly.

Output Detail - Warningcode

Description

This output holds a data structure (TesysTWarning) that contains information on the alarm currently on the device.

TesysTWarning Structure Description

This table shows the TesysTWarning structure:

Name	Type	Description
Ground	BOOL	Ground alarm.
Thermal	BOOL	Thermal alarm on device.
Jam	BOOL	Mechanical jam alarm.
Phase	BOOL	Phase alarm.
UnderCurrent	BOOL	Undercurrent alarm.
HMIPort	BOOL	HMI port communications failure alarm.
InternalTmp	BOOL	Internal temperature alarm.
Internal	BOOL	Detected internal error alarm.
Network	BOOL	Network port communications failure alarm.
Order	BOOL	Follow-up alarm. The device is not responding to the control command within the time specified in CommandCtrlWindow.
ForcedLocalMode	BOOL	Device forced locally; indicated by the device having the Status. ForcedLocalMode status. Control through screw terminal.

Output Detail - Resetting

Description

This output indicates whether or not a reset is carried out.

The CommandCtrlWindow variable indicates the maximum time for resetting the detected error.

When a device or communications reset is carried out with ResetFail, the DFB tries to reset the error within the time period defined in CommandCtrlWindow.

If the error is reset:

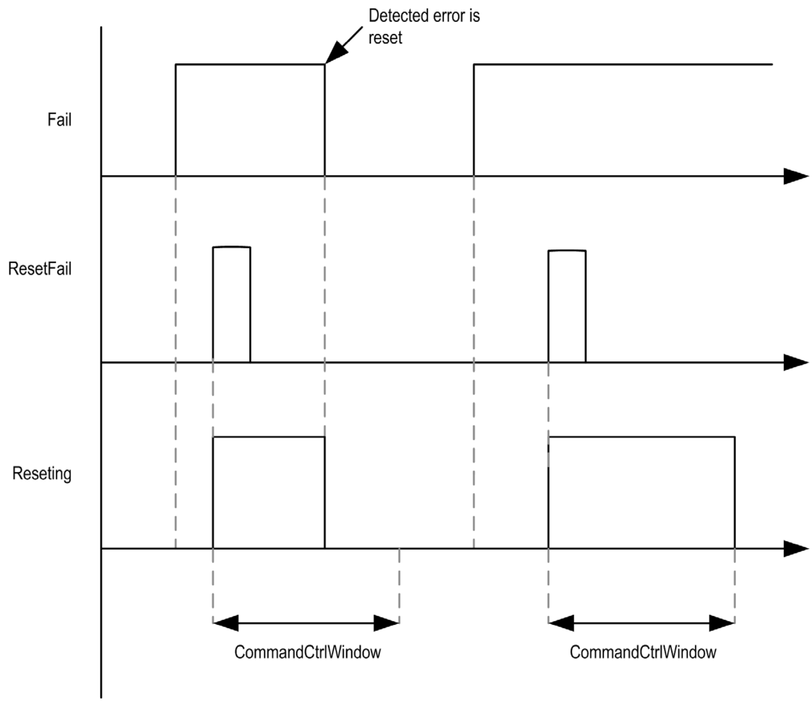
- The Fail and Resetting output variables are reset (they are set to FALSE).

If the error is not reset:

- The Resetting variable is set to FALSE.
- The Fail variable is held as TRUE.

ResetFail is edge-based.

This figure shows the Resetting output:



Output Detail - Status

Description

This output holds a data structure (TesysTStatus) that contains information that the module extracts from the status variable of device.

TesysTStatus Structure Description

This table shows the TesysUstatus structure:

Name	Type	Description
Activated	BOOL	Activated contact
Ready	BOOL	Switch in ON position; No detected device error
Fail	BOOL	There is a failure on the TesysT
Trip	BOOL	The Tesys protection was tripped
Warning	BOOL	Alarm present on Tesys
ResetAuth	BOOL	Authorized detected error reset
Controller Power	BOOL	Powered system
MotorRunning	BOOL	MotorRunning
HMIControlled	BOOL	Control through HMI port
MotorStarting	BOOL	The motor is starting
State	INT	Current machine status
Info	INT	Device status

State Integer Values Description

This table shows the State integer values:

Variable Value	Status
-2	Detected DFB error
-1	Not initialized. Waiting for data.
0	Disabled
5	Ready
8	Detected device error
9	Device notification

Info Integer Values Description

This table shows the Code with the information shown on the Unity operator screen:

Variable Value	Status
1	DFB parameter configuration interruption
2	WAITING Ready
8	Tesys working OK
10	Waiting for device information
11	MISSING EnableDFB
12	MISSING CommunicationOK Detected communications error
13	Status Word value is 0
14	Local forcing should be 0
23	REMOVE Run should be 0
24	REMOVE ResetFail. Reset again
81	MISSING ResetFail. Device detected error
82	A RESET is needed
99	Unknown status

Output Detail - StatusExt

Description

This output holds a data structure (TesysTStatusExt) that contains information that the module extracts from the device Status Word variable.

TesysTStatusExt Structure Description

This table shows the TesysTStatusExt structure:

Name	Type	Description
AutomaticResetActive	BOOL	Active automatic reset
FaultRequested	BOOL	Detected Error, restart (off/on) request
RestartTimeUnd	BOOL	Motor, undetermined restart time
RapidCicleLockout	BOOL	Rapid lockout cycle
LoadSheding	BOOL	Load Shedding
MotorSpeed	BOOL	Motor speed
HMILostComms	BOOL	HMI, loss of communications with the port
LostComms	BOOL	Network port, loss of communications
MotorTransitionLockout	BOOL	Motor, transition lockout

Output Detail - IOMap

Description

This function holds a data structure (TesysTIOMap) that information on the state of the inputs and outputs of the Tesys.

TesysTIOMap Structure Description

This table shows the TesysTIOMap structure:

Name	Type	Description
Output 1	BOOL	Shows the state of digital output 1
Output 2	BOOL	Shows the state of digital output 2
Output 3	BOOL	Shows the state of digital output 3
Output 4	BOOL	Shows the state of digital output 4

Output Detail - OutputData

Description

This function holds a data array structure (TESYST_OUT_DDT) that contains data sent to the device. The controller can be controlled with this output variable.

The function is reserved for the DFB and cannot be used directly.

Allocate the structure (%MWx) to allow the correct operation of the control block.

Depending on the device and the communications used, one of the following structures is used.

TESYSCTL_OUT_DDT Structure Description

This table shows the information available on an Advantys STB based TESYSCTL_OUT_DDT structure:

Name	Type	Description
OutputData [0]	INT	System control
OutputData [1]	INT	Analog signal control
OutputData [2]	INT	Output control

TESYST_OUT_DDT Structure Description

This table shows the information available on a CANopen based TESYST_OUT_DDT structure:

Name	Type	Description
OutputData [0]	INT	System control
OutputData [1]	INT	Analog signal control
OutputData [2]	INT	Output control
OutputData [3]	INT	System 2 control

6.4 Inputs/Outputs

Overview

This section describes the Inputs/Outputs of the TesysTDev FBD.

What's in this Section?

This section contains the following topics:

Generic Inputs/Outputs Description

Overview

This table is the same for all TesysT Profiles.

Inputs/Outputs Description Table

This table describes the inputs/outputs of the TesysT Profile FBD:

Name	Type	Description
TESYST_ST	TESYST_ST_DDT	TESYST_ST *(see page 134)* is a device data structure that holds the minimum information required for performing control and monitoring. The information is used by the operator screen and usable from HMI or SCADA.
TESYST_CFG	TESYST_CFG_DDT	TESYST_CFG *(see page 136)* is a device data structure that holds the information used by the operator screen. It can be read from HMI or SCADA.

Inputs/Outputs Detail - TESYST_ST

Description

The device data structure (TESYST_ST_DDT) holds the minimum information required for performing control and monitoring.

The information is used by operator screen and usable from HMI or SCADA.

TESYST_ST_DDT Structure Description

This table shows the TESYST_ST_DDT structure:

Name	Type	Description
STW	WORD	Device status
CFGW	WORD	Device control
AverageCurrent	REAL	Current current value in FLC%

STW Word Description

Access to the data held in this bit word is read-only.

This table shows the STW Word description:

Bit	Description
0	Unknown device status or communications interruption. No variable refreshing.
1	Not ready
2	Module running
3	Detected device error
4	Alarm on device or detected follow-up error (detected follow-up errors require resetting).
5	Communications interruption
6	Requires resetting. A ResetFail is required.
7	ExtControlled
8	Resetting
9	EnableDFB

CFGW Word Description

CFGW provides the means necessary to control the device.

Owner Value	Control is from
1	supervision subsystem or from the opeartor screen
0	supervision subsystem, the DFB input variables values are used for reading from HMI or SCADA

This table shows the CFGW Word Description:

Bit	Description
0	ResetFail
1	Owner
3	Direction
6	Run
7	ControlCommand
11	LowFastSpeed
12	ThermalTest

NOTE: The Owner bit enables to control the block from the •••_ST_DDT structure, ignoring the block inputs signals. It enables to effect control from a monitoring system (HMI, SCADA, Operator screen) in manual mode without using the programmed switching operation.

Inputs/Outputs Detail - TESYST_CFG

Description

This function holds a data structure (TESYST_CFG_DDT) that contains device information. The information is used by the operator screen and can be read from HMI or SCADA.

TESYST_CFG_DDT Structure Description

This table shows the TESYST_CFG_DDT structure:

Name	Type	Description
DataStatus	WORD	Information on the Device status
Info	INT	Device information
WarningCode	WORD	Alarm Code
FailCode 0	INT	Code of last Level 0 error
FailCode 1	INT	Code of last Level 1 error
FailCode 2	INT	Code of last Level 2 error

DataStatus Word Description

DataStatus Word provides the device status and information on the status output structure.

This table shows the DataStatus Word description:

Bit	Description
0	TesysT.Ready
1	TeysTStatus.Active
2	TeysTStatus.Detected error
3	TeysTStatus.Warning
4	TeysTStatus.Trip
5	TeysTStatus.ResetAuth
6	TeysTStatus.ControlledPower
7	TeysTStatus.MotorRunning
8	TeysTStatusExt.HMIControlled
9	TeysTStatusExt.MotorStarting
10	TeysTStatusExt.AutoResetActive
11	TeysTStatusExt.FaultRequested
12	TeysTStatusExt.RapidCicleLockout

Bit	Description
13	TeysTStatusExt.LoadSheding
14	TeysTStatusExt.HMILostComms
15	TeysTStatusExt.MotorTransitionLockout

WarningCode Word Description

This table shows the WarningCode Word description:

Bit	Description
2	WarningCode.Ground
3	WarningCode.Thermal
5	WarningCode.Jam
6	WarningCode.Phase
7	WarningCode.UnderCurrent
8	WarningCode.Order
9	WarningCode.ForcedLocalMode
10	WarningCode.HMIPort
11	WarningCode.InternalTmp
12	WarningCode.Internal
14	WarningCode.Module

Info Word Description

This table shows the Info Word Description:

Name	Type	Description
FailCode 0	INT	Indicates which error has occurred, FailCode[0]
FailCode 1	INT	Indicates which error has occurred, FailCode[1]
FailCode 2	INT	Indicates which error has occurred, FailCode[2]

6.5 Operator Screen

General

Description

The component includes an operator screen that enables interaction with the communication interface blocks.

The operator screen data is used to deal with a monitoring system by means of the following:

- TESYST_ST_DDT and TESYST_CFG_DDT - Inputs/Outputs structures.
- TESYSU_MEA depending on the component - Output variables.

6.6 Error Management

Detected Error Code Description

Overview

The detected error codes the device can return are read from the FailCode output variable.

Parameter Configuration Detected Error Codes

This detected error indicates that a parameter configuration public variable has a value that is not allowed.

To reload new values, an edge is required on the EnableDFB variable that cannot be reset:

- FailCode[0]: 16#0003
- FailCode[1]: 16#0000
- FailCode[2]: 16#0004

The Failcode[0] variable can have the following codes:

Error Code	Meaning
0	No detected Error
3	Ground current
4	Thermal overload
5	Extended start
6	Mechanical jam
7	Phase current imbalance.
8	Undercurrent
10	Test
11	HMI port error
12	Loss of HMI port communications
13	Internal network port error
18	Diagnosis
19	Wiring
20	Overcurrent
21	Phase current dropout
22	Phase currents inverted

Error Code	Meaning
23	Motor temperature sensor
24	Phase voltage imbalance
25	Phase voltage dropout
26	Control over voltage
26	Phase voltages inverted
27	Undervoltage
28	Overvoltage
29	Insufficient power
30	Excessive power
31	Insufficient power factor
32	Excessive power factor
33	Load shedding
51	Internal controller temperature error
55	Internal controller error (stack overflow)
56	Internal controller error (RAM error)
57	Internal controller error (RAM checksum error)

NOTE: The device failure reset is sent to the device

Modbus Communications Detected Error

For Modbus communications, this code is used to indicate that communications have not been established It can be reset.

- FailCode[0]: 16#0002
- FailCode[1]: 16#0000
- FailCode[2]: 16#0004

Once Modbus communications have been established, check “Modbus client” error codes for FailCode [0] and [1]. All components make a distinction between read request failures and write request failures:

- FailCode[2]: 16#0001 - Read
- FailCode[2]: 16#0002 - Write

Error Code Example

For a detected error, the code will be:

- FailCode[1]: 16#0000
- FailCode[2]: 16#0005

6.7 SCoD Representation and Parameter Description

SCoD Representation and Parameter Description

Overview

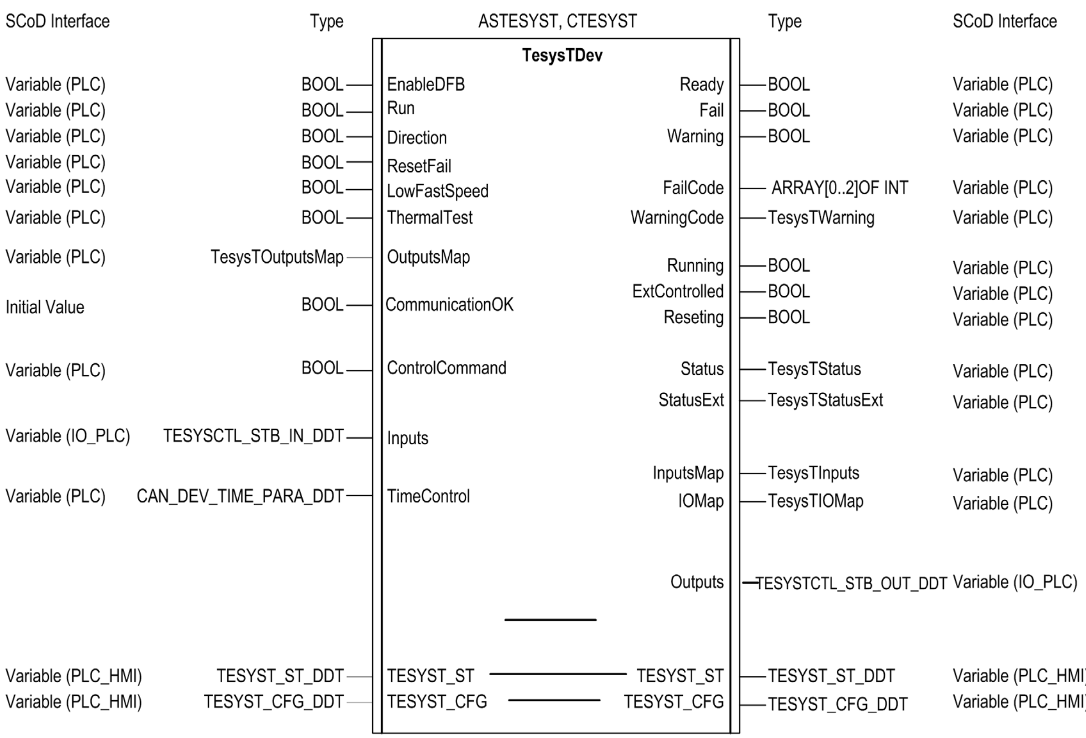
The usage of the DFB pins in the SCoD editor is explained in the input/output diagram below.

The variables defined in the SCoD editor have in principle the same name as the pin (except that an underscore in the pin name is skipped for the variable name).

If the variables defined in the SCoD have the property 'Initial Value' then the values of the instantiated SCoD variables can be modified manually in Unity Pro before PLC restart.

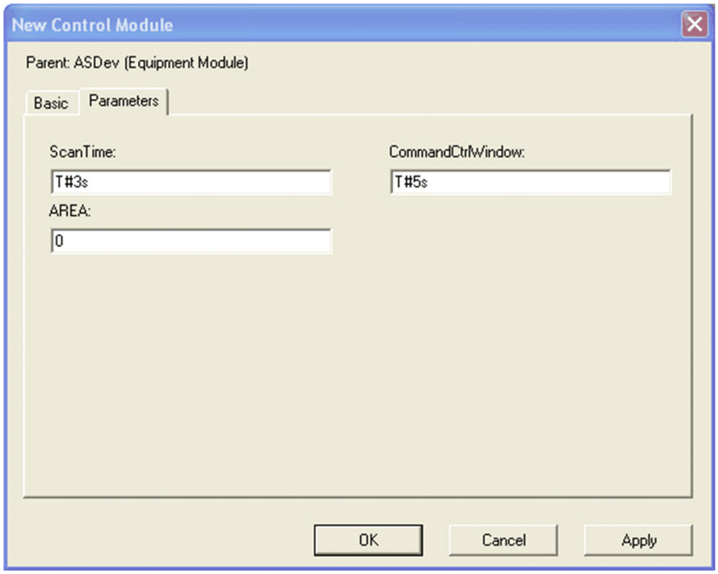
SCoD Representation

This figure represents the ASTESYST and CTESYST SCoDs:



Parameter Tab

This figure shows the ASTESYST and CTESYST parameter tab:



The following table describes the properties:

Property	Type	Description
ScanTime	TIME	Minimum time to maintain detected error signals
CommandControlWindow	TIME	The time waited before executing orders
AREA	BYTE	Area to which the object that you wish to instantiate belongs. Enables to control access to the functions provided by this object.

6.8 HMI Representation

TESYST - Representation in the HMI

CANOpen TeSysT (Genie)

The CANOpen TeSys T block is represented in the HMI by the following genie.

Genie Elements

Symbol	Meaning
	Opens the CANOpen TeSys T popup. Displays the device name, that is, TeSys T and average current value.
	The instance name of the CANOpen TeSys T block is only visible by selecting the **Hide**/**Unhide** button in the template.
	When there is an unacknowledged note, the ! symbol is displayed in the genie.
	When there is an active and unacknowledged alarm or communication failure, then outer rectangle flashes in red.
	When there is an active and unacknowledged alarm or communication failure, then outer rectangle is steady red.

Popup

The CANOpen TeSys T block popup has the following tabs:
- **Operator** tab
- **Analog** tab
- **Digital** tab
- **Information** tab

Operator Tab

This tab has 3 sub-sections:
- **Status**
- **Reset Alarm**
- **Current Operation**
- **Parameters**

The sub-section **Status** displays the current status.

The possible displays are listed as follows:
- **Unknown Status**
- **Not Ready**
- **Running**
- **Communication Failure**
- **Failure**
- **Warning**

The following table describes the sub-section **Reset Alarms**:

Item	Description
Reset Alarm	Resets alarm.

The following table describes the sub-section **Current Operation**:

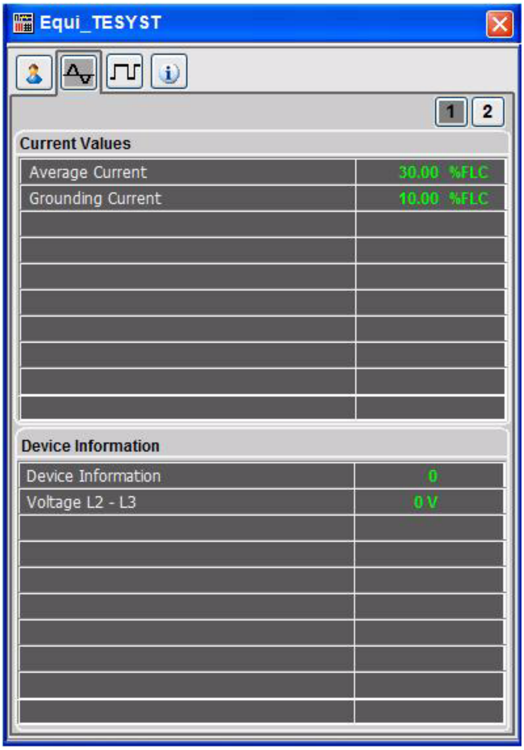
Item	Description
Run Order active	Displays the text in green color when the operation is active. Displays the text in white color when the operation is not active.
Run Forward active	
Reset Command active	
Control Orders from System active	
Fast Speed active	
Self Test Order active	

The following table describes the sub-section **Parameters**:

Item	Description
Module Ready (ON)	Indicates that the module is ready.
Tripped	Indicates that the module is tripped.
Fault Reset Authorized	Indicates the authorization of fault reset.
Current Average	Indicates the average current value.

The sub-section **Parameters** also shows the picture of TeSys T.

Analog Tab



This tab has 2 sub-tabs:

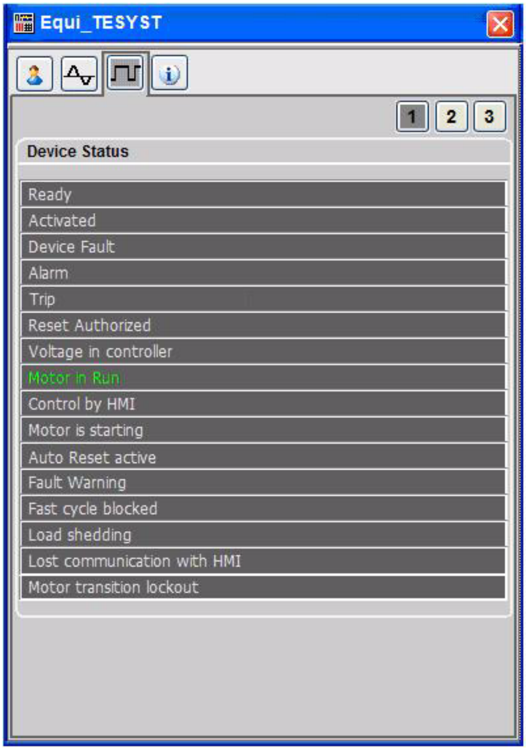
- **1**
- **2**

Each sub-tab has 2 sub-sections:

- **Current Values**
- **Device Information**

The title of these sections are configurable in the *cem_devices.dbf* file. The analog parameters and their values are displayed in these sub-sections.

Digital Tab



This tab has 3 sub-tabs:

- **1**
- **2**
- **3**

Each sub-tab has a sub-section, which shows the device status. The digital parameter descriptions are displayed in this sub-section.

When the status is not active, the text is displayed in white color. When the status is active, the text is displayed in green color. The text is configurable in the *cem_devices.dbf* file.

Information Tab

For description about **Information** tab, refer to Commonly Used Tabs *(see page 32)*.

MBEM6400 - Modbus Energy Meter 6400

7

Overview

This chapter describes the MBEM6400 block.

What's in this Chapter?

This chapter contains the following sections:

7.1 Description

Overview

This section describes the MBEM6400Dev DFB.

What's in this Section?

This section contains the following topics:

MBEM6400 - Description

General

The MBEM6400 block is implemented as listed in the following table:

Software	Implemented as
Unity Pro	DFB MBEM6400
Vijeo Citect	Genie EM6400
UAG	SCoD MBEM6400

Function Description

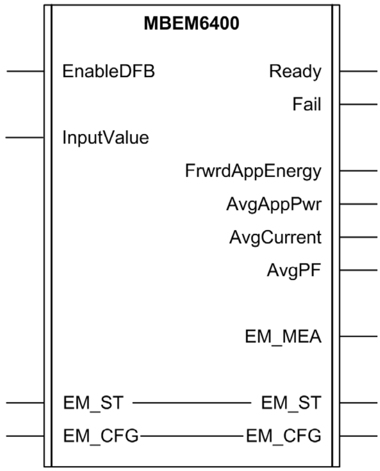
The MBEM6400 block is used to manage the energy meters in a communications based manner.

The monitoring function of the energy meter profile control block allows the most important parameter devices to be monitored.

FBD Representation

Representation in FBD

The functional module that will be used in the program has the following aspect (at the FBD block level) after PLC generation in UNITY Pro:



7.2 Inputs

Overview

This section describes the inputs of the `MBEM6400Dev` FBD.

What's in this Section?

This section contains the following topics:

Inputs Description

Inputs Description Table

The following table describes the input parameters of the `MBEM6400` block:

Parameter	Type	Description
`EnableDFB`	`BOOL`	This input enables the normal execution of the control block. Activating this input enables communications with devices for their operation. The public variable values are loaded during the first enabling cycle.
`InputValue`	`EM64_INPUTS`	IO_PLC variable for reading information from complex IO device, containing measurement data from EM6400. Start address: 3900. End address: 3965. Refer to the EM64_INPUTS *(see page 155)* structure description.

Input Detail - InputValue

EM64_INPUTS Structure Description

Name	Type	Description
AvgAppPwr	REAL	Average apparent power (kVARH)
AvgActPwr	REAL	Average active power (Kw)
AvgReactPwr	REAL	Average reactive power (Kw)
AvgPF	REAL	Average power factor
AvgLLVoltage	REAL	Average line to line voltage (V)
AvgLNVoltage	REAL	Average line to neutral voltage (V)
AvgCurrent	REAL	Average Current (Amps)
Freq	REAL	Frequency (Hz)
RPhAppPwr	REAL	R - Phase apparent power
RPhActPwr	REAL	R - Phase active power
RPhReactPwr	REAL	R - Phase reactive power
RYPhVoltage	REAL	R - Y Phase voltage (V)
RNPhVoltage	REAL	R - N Phase voltage (V)
RPhCurrent	REAL	R - Phase current (A)
YPhAppPwr	REAL	Y - Phase apparent power
YPhActPwr	REAL	Y - Phase active power
YPhReactPwr	REAL	Y - Phase reactive power
YBPhVoltage	REAL	Y - B Phase voltage (V)
YNPhVoltage	REAL	Y - N Phase voltage (V)
YPhCurrent	REAL	Y - Phase current (A)
BPhAppPwr	REAL	B - Phase apparent power
BPhActPwr	REAL	B - Phase active power
BPhReactPwr	REAL	B - Phase reactive power
BRPhVoltage	REAL	B - R Phase voltage (V)
BNPhVoltage	REAL	B - N Phase voltage (V)
BPhCurrent	REAL	B - Phase current (A)
FrwrdAppEnergy	REAL	Forward apparent energy
FrwrdActEnergy	REAL	Forward active energy
FrwrdReactIndEnergy	REAL	Forward reactive inductive energy
FrwrdReactCapEnergy	REAL	Forward reactive capacitive energy

7.3 Outputs

Overview

This section describes the outputs of the MBEM6400Dev FBD.

What's in this Section?

This section contains the following topics:

MBEM6400 - Description

Parameter Description

The following table describes the output parameters of the MBEM6400 block:

Parameter	Type	Description
Ready	BOOL	This output indicates if the device is enabled and free of detected errors. The device is ready to carry out any read operation.
Fail	BOOL	This output indicates a detected error in the control block (parameter configuration cannot be reset) or a detected communications interruption.
FrwrdAppEnergy	REAL	Apparent power consumption (kVAh).
AvgAppPwr	REAL	Average apparent power (kVA).
AvgCurrent	REAL	Current - average between 3 phases (V).
AvgPF	REAL	Average power factor.
EM_MEA	EM_MEA_DDT	The EM_MEA is a data structure that contains device measurement. Refer to the EM_MEA_DDT *(see page 158)* structure description.

Output Detail - EM_MEA

EM_MEA_DDT Structure Description

Name	Type	Description
AvgAppPwr	REAL	Average apparent power (kVARH)
AvgActPwr	REAL	Average active power (Kw)
AvgReactPwr	REAL	Average reactive power (KVAR)
AvgPF	REAL	Average power factor
AvgLLVoltage	REAL	Average line to line voltage (V)
AvgLNVoltage	REAL	Average line to neutral voltage (V)
AvgCurrent	REAL	Average Current (Amps)
Freq	REAL	Frequency (Hz)
RPhAppPwr	REAL	R - Phase apparent power
RPhActPwr	REAL	R - Phase active power
RPhReactPwr	REAL	R - Phase reactive power
RYPhVoltage	REAL	R - Y Phase voltage (V)
RNPhVoltage	REAL	R - N Phase voltage (V)
RPhCurrent	REAL	R - Phase current (A)
YPhAppPwr	REAL	Y - Phase apparent power
YPhActPwr	REAL	Y - Phase active power
YPhReactPwr	REAL	Y - Phase reactive power
YBPhVoltage	REAL	Y - B Phase voltage (V)
YNPhVoltage	REAL	Y - N Phase voltage (V)
YPhCurrent	REAL	Y - Phase current (A)
BPhAppPwr	REAL	B - Phase apparent power
BPhActPwr	REAL	B - Phase active power
BPhReactPwr	REAL	B - Phase reactive power
BRPhVoltage	REAL	B - R Phase voltage (V)
BNPhVoltage	REAL	B - N Phase voltage (V)
BPhCurrent	REAL	B - Phase current (A)
FrwrdAppEnergy	REAL	Forward apparent energy
FrwrdActEnergy	REAL	Forward active energy
FrwrdReactIndEnergy	REAL	Forward reactive inductive energy
FrwrdReactCapEnergy	REAL	Forward reactive capacitive energy

7.4 Inputs/Outputs

Overview

This section describes the inputs/outputs of the MBEM6400Dev FBD.

What's in this Section?

This section contains the following topics:

MBEM6400 - Description

Parameter Description

The following table describes the input/output parameter of the MBEM6400 block:

Parameter	Type	Description
EM_ST	EM_ST_DDT	The EM_ST device data structure that holds the minimum information required for performing control and monitoring. The information is used by the operator screen and can be read from HMI or SCADA. Refer to the EM_ST *(see page 161)* structure description.
EM_CFG	EM_CFG_DDT	The EM_CFG device data structure holds information used by the operator screen. It can be read from HMI or SCADA. Refer to the EM_CFG *(see page 162)* structure description.

Inputs/Outputs Detail - EM64_ST_DDT

EM_ST Structure Description

Name	Type	Description
STW	WORD	Device main status: • Bit 0 - Unknown state • Bit 1 - Not ready • Bit 2 - Running • Bit 3 - Device fail • Bit 4 - Warning • Bit 5 - Communication interruption • Bit 6 - Necessary resetting • Bit 9 - Enable DFB
CFGW	WORD	Device main control orders Owner = 1 HMI control Owner = 0 information from inputs • Bit 0 - Reset fail • Bit 1 - Owner
AvgApparentPower	REAL	Current aparent energy (kVAh)
AvgActivePower	REAL	Total aparent power detected (kVA)
AvgPF	REAL	Total power factor in use
Frequency	REAL	Average current (A)

Inputs/Outputs Detail - EM64_CFG_DDT

EM64_CFG_DDT Structure Description

Name	Type	Description
WarningCode	WORD	Warning of device. Data from warning data from device
FailCode0	INT	Code of last level 0 detected error
FailCode1	INT	Code of last level 1 detected error
FailCode2	INT	Code of last level 2 detected error

Bit description for the warning word (W4112):

Bit	Description
0	Phase 1 voltage out of range
1	Phase 2 voltage out of range
2	Phase 3 voltage out of range
3	Phase 1 current out of range
4	Phase 2 current out of range
5	Phase 3 current out of range
6	Frequency out of range, or phase 1 voltage insufficient to determine frequency
7...15	Not used

Bit description for the warning word (W3254):

Bit	Description
0	Summary bit (activated if any other bit is activated)
1	Configuration detected error
2	Scaling detected error
3	Phase dropout
4	Cabling detected error
5	Incremental energy could be incorrect due to the meters resetting
6	External demand synchronization waiting time
7...15	Not used

7.5 SCoD Representation and Parameter Description

MBEM6400 - Representation as SCoD in UAG

Overview

The usage of the DFB pins in the SCoD editor is explained in the input/output diagram below.

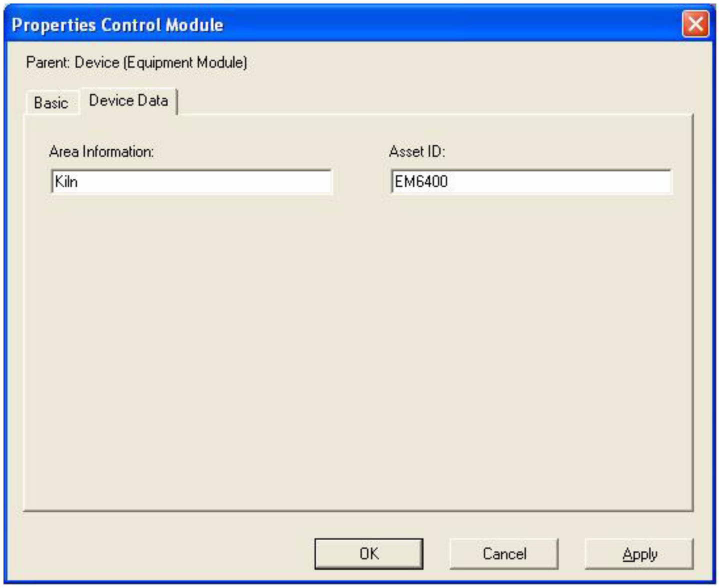
The variables defined in the SCoD editor have in principle the same name as the pin (except that an underscore in the pin name is skipped for the variable name).

If the variables defined in the SCoD have the property 'Initial Value' then the values of the instantiated SCoD variables can be modified manually in Unity Pro before PLC restart.

Input Output Diagram

SCoD-Interface	TYPE	MBEM6400		TYPE	SCoD-Interface
Variable (PLC)	BOOL	EnableDFB	Ready	BOOL	Variable (PLC)
			Fail	BOOL	Variable (PLC)
Variable (IO_PLC)	EM64_INPUTS	InputValue			
			FrwrdAppEnergy	REAL	Variable (PLC)
			AvgAppPwr	REAL	Variable (PLC)
			AvgCurrent	REAL	Variable (PLC)
			AvgPF	REAL	Variable (PLC)
			EM_MEA	EM64_MEA_DDT	Variable (PLC_HMI)
Variable (PLC_HMI)	EM64_ST_DDT	EM_ST	EM_ST	EM64_ST_DDT	Variable (PLC_HMI)
Variable (PLC_HMI)	EM64_CFG_DDT	EM_CFG	EM_CFG	EM64_CFG_DDT	Variable (PLC_HMI)

Device Data Tab



The following table describes the properties:

Name	Initial Value	Property Tab	Position	Description
Area Information	Not available	**Device Data**	1	Area to which the instantiated object belongs. Enables to control access to the functions provided by this object.
Asset ID			2	Asset ID

7.6 HMI Representation

MBEM6400 - Representation in the HMI

Modbus Energy Meter 6400 (Genie)

The MBEM6400 block is represented in the HMI by the following genie.

Genie Elements

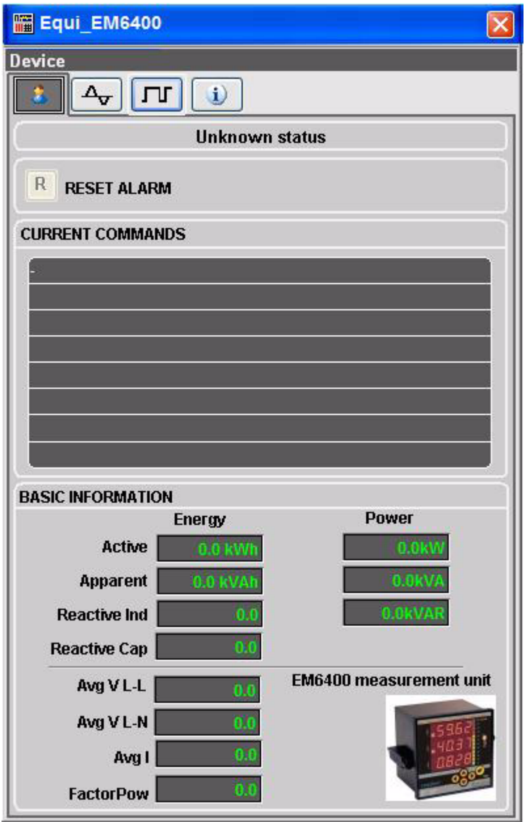
Symbol	Meaning
	Opens the MBEM6400 popup. Displays the device name, that is, energy meter and energy value.
	The instance name of the MBEM6400 block is only visible by selecting the **Hide**/**Unhide** button in the template.
	When there is an unacknowledged note, the ! symbol is displayed in the genie.
	When there is an active and unacknowledged alarm or communication failure, then outer rectangle flashes in red.
	When there is an active and unacknowledged alarm or communication failure, then outer rectangle is steady red.

Popup

The `MBEM6400` block popup has the following tabs:

- **Operator** tab
- **Analog** tab
- **Digital** tab
- **Information** tab

Operator Tab



This tab has a sub-section called as **Basic Information**, which is described in the following table:

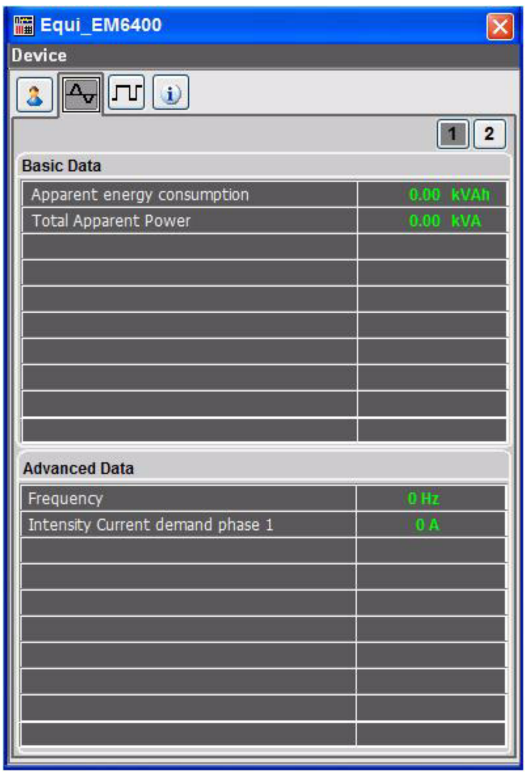
Item	Description
Active	Displays the values of active energy and active power.
Apparent	Displays the values of apparent energy and apparent power.

Item	Description
Reactive Ind	Displays the values of reactive inductive energy and reactive inductive power.
Reactive Cap	Displays the values of reactive capacitive energy and reactive capacitive power.
Avg V L-L	Displays the value of average line to line voltage.
Avg V L-N	Displays the value of average line to neutral voltage.
Avg I	Displays the value of average current.
FactorPow	Displays the value of power factor.

The sub-section **Parameters** also shows the picture of energy meter.

NOTE: The sub-sections **Status**, **Reset Alarm**, and **Current Commands** are not applicable in the **Operator** tab of the `MBEM6400` block.

Analog Tab



This tab has 2 sub-tabs:

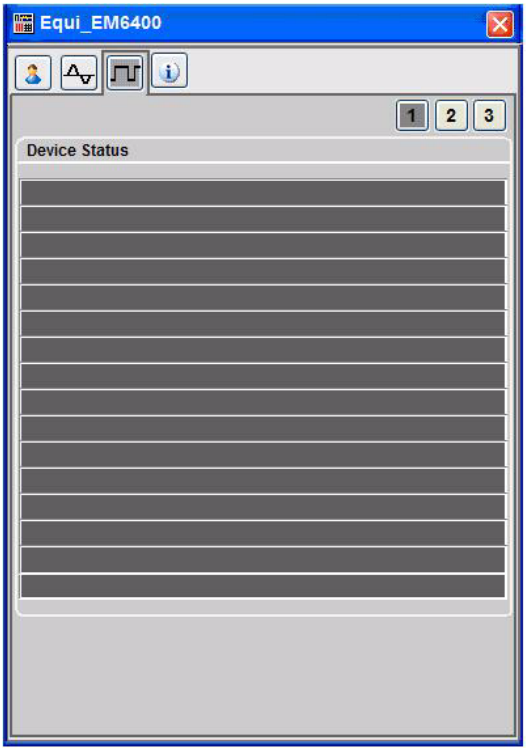
- **1**
- **2**

Each sub-tab has 2 sub-sections:

- **Basic Data**
- **Advanced Data**

The title of these sections are configurable in the *cem_devices.dbf* file. The analog parameters and their values are displayed in these sections.

Digital Tab



This tab has 3 sub-tabs:

- **1**
- **2**
- **3**

Each sub-tab has a sub-section, which shows the device status. The digital parameter descriptions are displayed in this sub-section.

When the status is not active, the text is displayed in white color. When the status is active, the text is displayed in green color. The text is is configurable in the *cem_devices.dbf* file.

Information Tab

For description about **Information** tab, refer to Commonly Used Tabs *(see page 32)*.

MBEM7300 - Modbus Energy Meter 7300

8

Overview

This chapter describes the MBEM7300 block.

What's in this Chapter?

This chapter contains the following sections:

8.1 Description

Overview

This section describes the `MBEM7300Dev` DFB.

What's in this Section?

This section contains the following topics:

Functional Description

General

The MBEM7300 block is implemented as listed in the following table:

Software	Implemented as
Unity Pro	DFB MBEM7300
Vijeo Citect	Genie EM7300
UAG	SCoD MBEM7300

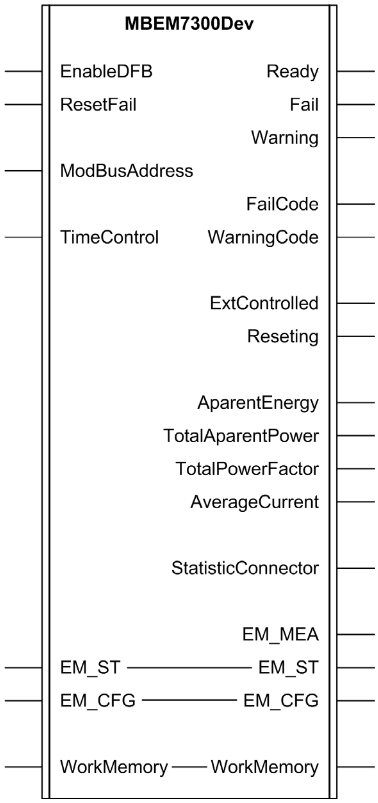
The main functions of the energy meter profile control block is explained as follows:

Function	Description
Remote Resetting	Enables to reset the device.
Monitoring	Allows the most important parameter devices to be monitored.

FBD Representation

Representation in FBD

The functional module that will be used in the program has the following aspect (at the FBD block level) after PLC generation in UNITY Pro:



8.2 Inputs

Inputs Description

Inputs Description Table

The following table describes the input parameters of the `MBEM7300` block:

Parameter	Type	Description
`EnableDFB`	`BOOL`	This input enables the normal execution of the control block. When this input is disabled, the entire DFB is restarted (Statuses, output values, counters, and so on are lost) and all the output values are set to 0 or FALSE as long as signal remains on FALSE. Activating this input enables communications with devices for their operation. The public variable values are loaded during the first enabling cycle.
`ResetFail`	`BOOL`	Current detected errors are reset on the rising edge of this signal. Both communications and detected device errors can be reset. If the detected error is a device error and if control command = TRUE, it sends a reset command to the device. Fail AND (NOT Reseting) ResetFail
`ModBusAddress`	`INT`	Device address within the Modbus network.
`TimeControl`	`MB_DEV_Time_PARA_DDT`	Time control is a parameter for device control.

8.3 Outputs

Overview

This section describes the outputs of the MBEM7300Dev FBD.

What's in this Section?

This section contains the following topics:

Outputs Description

Outputs Description Table

The following table describes the output parameters of the MBEM7300 block:

Parameter	Type	Description
Ready	BOOL	This output indicates if the device is enabled and free of detected errors. The device is ready to carry out any read operation.
Fail	BOOL	This output indicates a detected error in the control block or a detected communications interruption. To reset the detected error, the Rst input has to be activated. The last error is shown on the FailCode.
Warning	BOOL	This output indicates that an alarm is activated for the drive. It does not affect the block operation and need not be reset. This signal remains active until the cause for alarms disappears.
FailCode	ARRAY [0...2] of INT	This output indicates the detection of an error. When the output is activated, this variable contains the code for the current detected error. If the output is not activated, this variable contains the last detected error. The detected error is specified by a 3-level structure.
WarningCode	WORD	The alert code is a variable that holds the alarm.
ExtControlled	BOOL	This output indicates if the device is controlled from an external source (from the monitoring system) to the system (1) or not (0).
Reseting	BOOL	This output indicates whether or not a reset is carried out.
AparentEnergy	DINT	Apparent power consumption (kVAh).
TotalAparentPower	REAL	Total apparent power (kVA).
TotalPowerFactor	REAL	Total power factor.
AverageCurrent	REAL	Average current between 3 phases.

Parameter	Type	Description
StatisticConnector	StatisticConnector	The statistic connector represents information that is used with Modbus communications to obtain statistics on the Modbus network. Refer to the StatisticConnector *(see page 179)* structure description.
EM_MEA	EM_MEA_DDT	The EM_MEA is a data structure that contains device measurement. Refer to the EM_MEA_DDT *(see page 180)* structure description.

Output Detail - `StatisticConnector`

`StatisticConnector` Structure Description

Name	Type	Description
`PartialTime`	`DINT`	Time partial
`Start`	`BOOL`	Operation started
`EndOK`	`BOOL`	Operation ended correctly
`EndNOK`	`BOOL`	Operation finished with detected error
`Reserve`	`BOOL`	Reserved

Output Detail - EM_MEA

EM_MEA_DDT Structure Description

Name	Type	Description
ActiveEnergy	REAL	Current active energy (kWh)
ReactiveEnergy	REAL	Current reactive energy (kVARh)
TotalActivePower	REAL	Total active power detected (kW)
TotalReactivePower	REAL	Total reactive power detected (kVAR)
AverageLineToLineVoltage	REAL	Average voltage between lines (V)
AverageLineToNeutralVoltage	REAL	Average voltage between line and ground (V)
Frequency	REAL	Detected frequency from line 1(Hz)
ActualCurrentL1	REAL	Instant current from line 1 (A)
ActualCurrentL2	REAL	Instant current from line 2 (A)
ActualCurrentL3	REAL	Instant current from line 3 (A)
VoltageL1ToL2	REAL	Voltage between line 1 and line 2 (V)
VoltageL2ToL3	REAL	Voltage between line 2 and line 3 (V)
VoltageL1ToL3	REAL	Voltage between line 1 and line 3 (V)
VoltageL1ToNeutral	REAL	Voltage between line 1 and ground (V)
VoltageL2ToNeutral	REAL	Voltage between line 2 and ground (V)
VoltageL3ToNeutral	REAL	Voltage between line 3 and ground (V)
ActivePowerL1	REAL	Active power at line 1 (kW)
ActivePowerL2	REAL	Active power at line 2 (kW)
ActivePowerL3	REAL	Active power at line 3 (kW)
AparentPowerL1	REAL	Aparent power at line 1 (kVA)
AparentPowerL2	REAL	Aparent power at line 2 (kVA)
AparentPowerL3	REAL	Aparent power at line 3 (kVA)
ReactivePowerL1	REAL	Reactive power at line 1 (kVAR)
ReactivePowerL2	REAL	Reactive power at line 2 (kVAR)
ReactivePowerL3	REAL	Reactive power at line 3 (kVAR)
THDCurrentL1	REAL	Current THD from line 1 (%)
THDCurrentL2	REAL	Current THD from line 2 (%)

Name	Type	Description
THDCurrentL3	REAL	Current THD from line 3 (%)
THDVoltageL1	REAL	Voltage THD from line 1 to line 2 (%)
THDVoltageL2	REAL	Voltage THD from line 2 to line 3 (%)
THDVoltageL3	REAL	Voltage THD from line 1 to line 3 (%)

8.4 Inputs/Outputs

Overview

This section describes the inputs/outputs of the MBEM7300Dev FBD.

What's in this Section?

This section contains the following topics:

Inputs/Outputs Description

Inputs/Outputs Description Table

The following table describes the input/output parameter of the MBEM7300 block:

Parameter	Type	Description
EM_ST	EM_ST_DDT	The EM_ST device data structure that holds the minimum information that is required for performing control and monitoring. The information is used by the operator screen and can be read from HMI or SCADA. Refer to the EM_ST *(see page 184)* structure description.
EM_CFG	EM_CFG_DDT	The EM_CFG device data structure holds information used by the operator screen. It can be read from HMI or SCADA. Refer to the EM_CFG *(see page 185)* structure description.
WorkMemory	ARRAY [0...544] of INT	Array used for Modbus communications. This variable is meant for use with a Modbus port that serializes all Modbus requests in an optimum manner.

Inputs/Outputs Detail - `EM_ST`

`EM_ST_DDT` Structure Description

Name	Type	Description
`STW`	`WORD`	Device main status: • Bit 0 - Unknown state • Bit 1 - Not ready • Bit 2 - Running • Bit 3 - Device fail • Bit 4 - Alert • Bit 5 - Communication interruption • Bit 6 - Necessary resetting • Bit 9 - Enable DFB
`CFGW`	`WORD`	Device main control orders Owner = 1 HMI control Owner = 0 information from inputs • Bit 0 - Reaset fail • Bit 1 - Owner
`AparentEnergy`	`REAL`	Current aparent energy (kVAh)
`TotalAparentPower`	`REAL`	Total aparent power detected (kVA)
`TotalPowerFactor`	`REAL`	Total power factor in use
`AverageCurrent`	`REAL`	Average current (A)

Inputs/Outputs Detail - EM_CFG

EM_CFG_DDT Structure Description

Name	Type	Description
WarningCode	WORD	Alarm code
FailCode0	INT	Code of last level 0 detected error
FailCode1	INT	Code of last level 1 detected error
FailCode2	INT	Code of last level 2 detected error

Bit description for the warning word (W4112):

Bit	Description
0	Phase 1 voltage out of range
1	Phase 2 voltage out of range
2	Phase 3 voltage out of range
3	Phase 1 current out of range
4	Phase 2 current out of range
5	Phase 3 current out of range
6	Frequency out of range, or phase 1 voltage insufficient to determine frequency
7...15	Not used

Bit description for the warning word (W3254):

Bit	Description
0	Summary bit (activated if any other bit is activated)
1	Configuration detected error
2	Scaling detected error
3	Phase dropout
4	Cabling detected error
5	Incremental energy could be incorrect due to the meters resetting
6	External demand synchronization waiting time
7...15	Not used

8.5 Error Management

Detected Error Code

Description

The detected error codes the device can return are read from the FailCode output variable. The detected error codes are the same as for the MBPM710, MBPM800 Error Management *(see page 239)*.

8.6 SCoD Representation and Parameter Description

SCoD Representation and Parameter Description

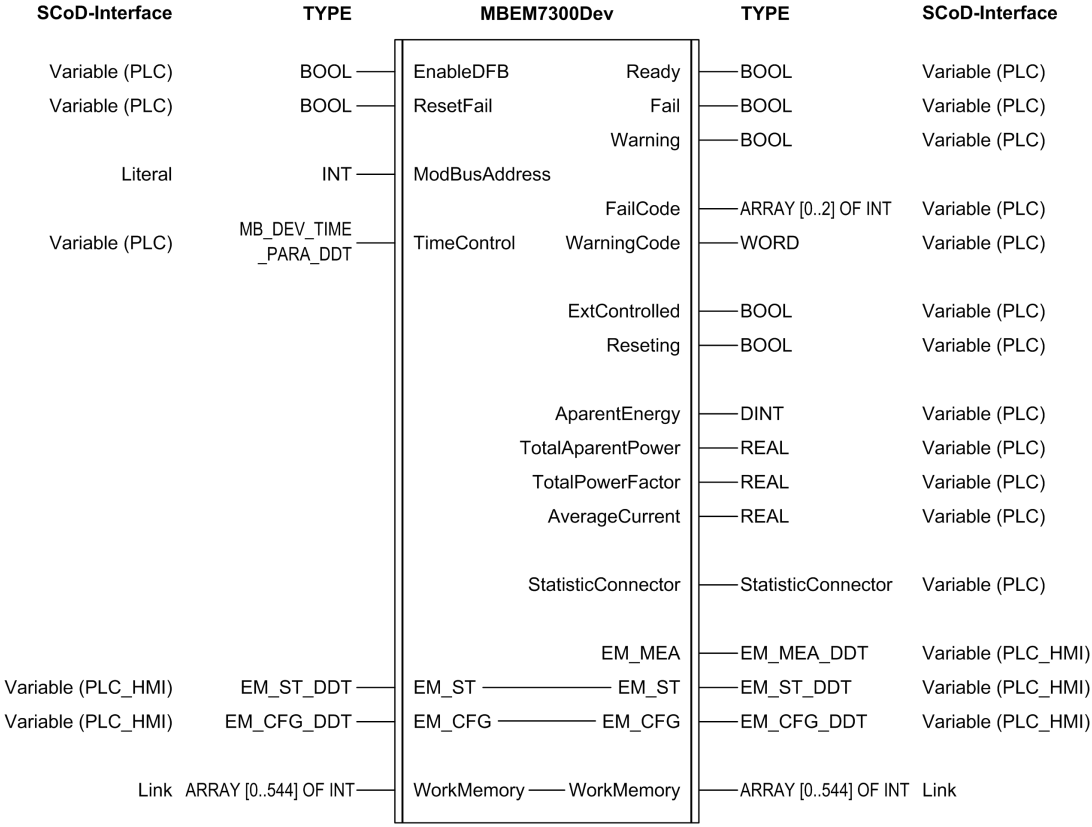
Overview

The usage of the DFB pins in the SCoD editor is explained in the following SCoD diagram.

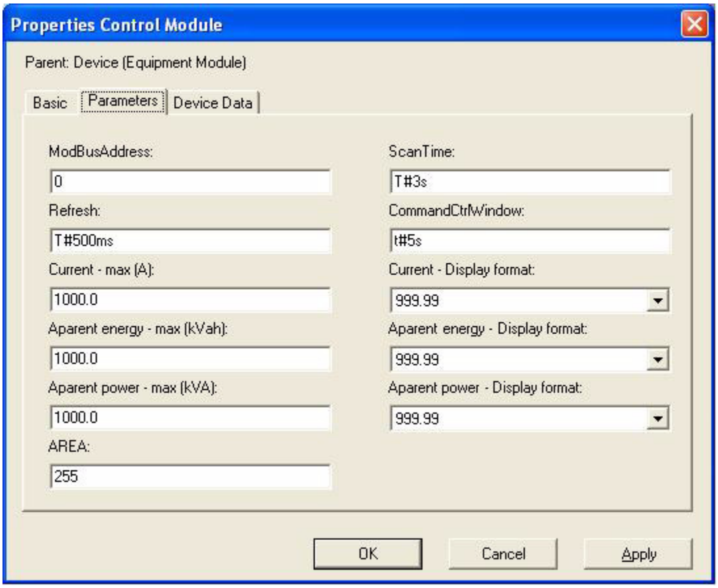
The variables defined in the SCoD editor have in principle the same name as the pin (except that an underscore in the pin name is skipped for the variable name).

If the variables defined in the SCoD have the property ‘Initial Value’ then the values of the instantiated SCoD variables can be modified manually in Unity Pro before PLC is restarted.

SCoD Representation



Parameters Tab

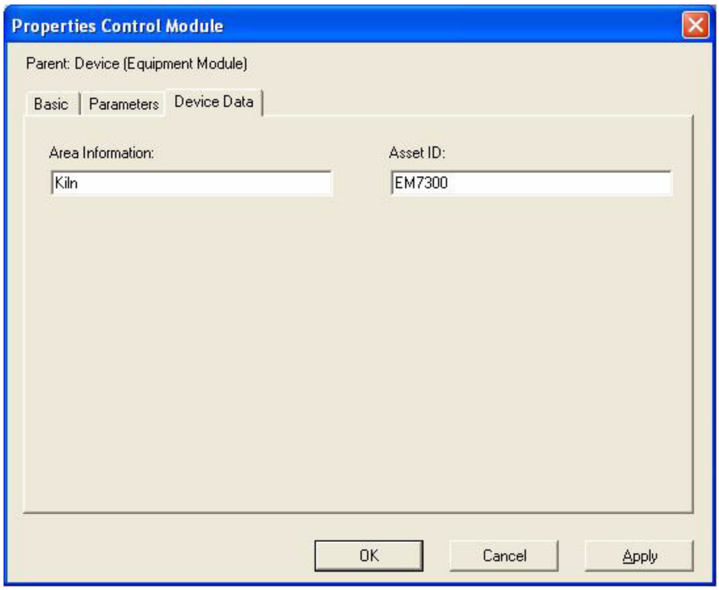


The following table describes the properties:

Name	Initial Value	Description
ModBus Address	0	This property represents the device address on the Modbus network. This variable can be found in Modbus communications.
Scan Time	T#3s	Minimum time to maintain detected error signals.
Refresh	T#500ms	Refresh time for device read data on Modbus communications.
CommandCtrlWindow	T#5s	The time waited before executing orders.
Current - max (A)	0	Maximum scaling range for average current.
Current - Display format	T#3s	Display format for average current.
Apparent energy - max (kVah)	T#500ms	Maximum scaling range for apparent energy.
Apparent energy - Display format	T#5s	Display format for apparent energy.
Apparent power - max (kVA)	T#500ms	Maximum scaling range for total apparent power.

Name	Initial Value	Description
Apparent power - Display format	T#5s	Display format for total apparent power.
AREA	T#5s	Area to which the instantiated object belongs. Enables to control the acess to the functions provided by this object.

Device Data Tab



The following table describes the properties:

Name	Initial Value	Description
Area Information	Not available	Area to which the instantiated object belongs. Enables to control the acess to the functions provided by this object.
Asset ID		Asset ID

8.7 HMI Representation

MBEM7300 - Representation in the HMI

Modbus Energy Meter 7300 (Genie)

The MBEM7300 block is represented in the HMI by the following genie.

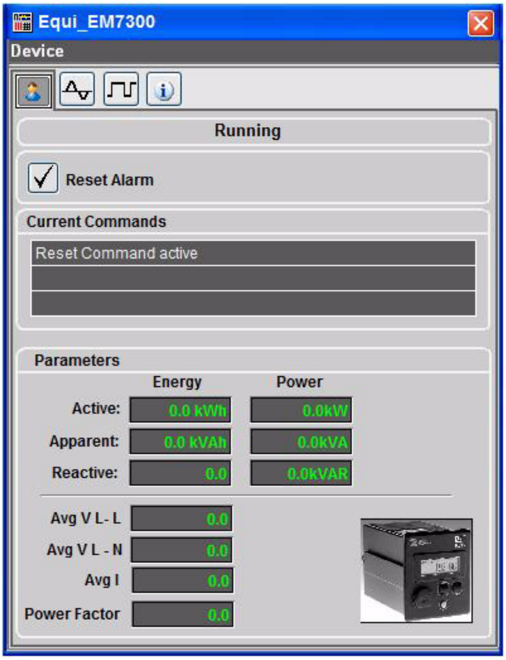
Genie Elements

Symbol	Meaning
	Opens the MBEM7300 popup. Displays the device name, that is, energy meter and active energy value.
	The instance name of the MBEM7300 block is only visible by selecting the **Hide**/**Unhide** button in the template.
	When there is an unacknowledged note, ! symbol is displayed in the genie.
	When there is an active and unacknowledged alarm or communication failure, then outer rectangle blinks in red.
	When there is an active and unacknowledged alarm or communication failure, then outer rectangle is steady red.

Popup

The `MBEM7300` block popup has the following tabs:
- **Operator** tab
- **Analog** tab
- **Digital** tab
- **Information** tab

Operator Tab



This tab has 3 sub-sections:
- **Status**
- **Reset Alarm**
- **Current Commands**
- **Parameters**

The sub-section **Status** displays the current status.

The possible displays are listed as follows:
- **Unknown Status**
- **Not Ready**
- **Running**
- **Communication Failure**
- **Failure**
- **Warning**

The following table describes the sub-section **Reset Alarm**:

Item	Description
Reset Alarm	Resets alarm.

The following table describes the sub-section **Current Commands**:

Item	Description
Reset Command active	Displays the text in green color when the operation is active. Displays the text in white color when the operation is not active.

The following table describes the sub-section **Basic Information**:

Item	Description
Active	Displays the values of active energy and active power.
Apparent	Displays the values of apparent energy and apparent power.
Reactive Ind	Displays the values of reactive inductive energy and reactive inductive power.
Reactive Cap	Displays the values of reactive capacitive energy and reactive capacitive power.
Avg V L-L	Displays the value of average line to line voltage.
Avg V L-N	Displays the value of average line to neutral voltage.
Avg I	Displays the value of average current.
FactorPow	Displays the value of power factor.

The sub-section **Parameters** also shows the picture of energy meter.

Analog Tab

For detailed description, refer to Analog Tab *(see page 167)*.

Digital Tab

For detailed description, refer to Digital Tab *(see page 168)*.

Information Tab

For description about **Information** tab, refer to Commonly Used Tabs *(see page 32)*.

MBATV - Modbus ATV Drive (MBATV31, MBATV61, and MBATV71 SCoDs)

9

Overview

This chapter describes the DFB for the control blocks for variable speed drives on a Modbus network, represented by MBATV31, MBATV61, and MBATV71 SCoDs.

What's in this Chapter?

This chapter contains the following sections:

9.1 Description

Overview

This section describes the MBATVDev FBD.

What's in this Section?

This section contains the following topics:

Functional Description

General

MBATVDev is the control block for the ATV on Modbus Communication.

The MBATVDev block is implemented as listed below:

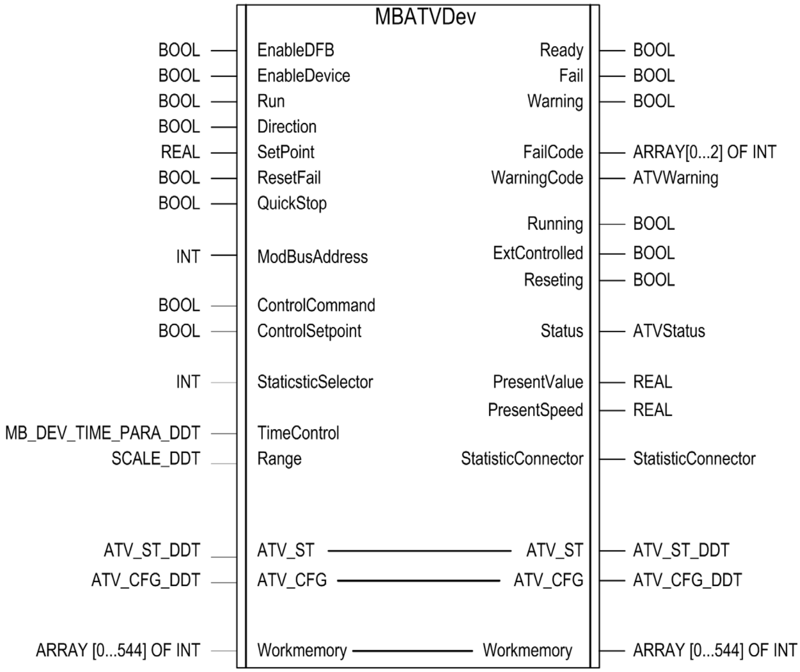
Software	Implemented as
Unity Pro	FBD MBATVDev
Vijeo Citect	DPC_Devices
UAG	Depending on the speed drive: ● SCoD MBATV31 ● SCoD MBATV61 ● SCoD MBATV71

The main function blocks are described in the ATV Profile Functional Description *(see page 29)*.

FBD Representation

Representation in FBD

The functional module that will be used in the program has the following aspect (at the FBD block level) after PLC generation in UNITY Pro:



9.2 Inputs

Overview

This section describes the inputs of the MBATVDev FBD.

What's in this Section?

This section contains the following topics:

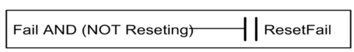
Generic Inputs Description

Overview

This table is the same for all ATV Profiles.

Inputs Description Table

This table describes the generic inputs of the ATV Profile FBD:

Input	Type	Description
EnableDFB	BOOL	This input enables the normal execution of the control block. EnableDFB = 0. The entire DFB is restarted (statuses, output values, counters, etc. are lost) and all output values are set to 0 or FALSE. EnableDFB = 1. This input enables communications with devices for their operation. Public variable values are loaded during the first enabling cycle.
EnableDevice	BOOL	This input variable is true only if EnableDFB is true. To control the speed of the drive, it must be enabled.
Run	BOOL	This input makes the motor run in the direction selected with the Direction variable.
Direction	BOOL	This input activates the direction of motor rotation: **1 =** forward **0 =** reverse **NOTE:** Setpoint assumes positive values. Changes in the direction of rotation cannot be made with Setpoint sign.
Setpoint	REAL	This is the speed set-point *(see page 54)* requested from the speed of the drive. It can only have positive, real values.
ResetFail	BOOL	Existing detected errors are reset on the rising edge of this signal. Both communications and detected device errors can be reset. If the detected error is a device error, it sends a reset command to the device if ControlCommand = TRUE. This figure shows the input ResetFail: 
QuickStop	BOOL	This input enables a quick stops the drive speed. This variable is state-based. If there is a QuickStop, the Run bit has to be reset to resume operation.

Input	Type	Description
ControlCommand	BOOL	This input indicates if the starter is controlled from an external source. If ControlCommand=TRUE the device is controlled from an external source. The generation of follow-up alarms is disabled. If ControlCommand=FALSE, the block only performs read operations on the device status, the transmission of device control or reset commands is not possible. **NOTE:** This input does not configure the starter; it only defines the DFB operating mode.
ControlSetPoint	BOOL	This input indicates if the speed of the drive is controlled from an external source. (e.g., from the console or a potentiometer)

WARNING

UNEXPECTED MOTOR START

Reset the Run variable before resuming operation.

The Run input must be inactive while the device is being enabled or while a detected error is being reset.

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

NOTICE

UNEXPECTED BEHAVIOUR

The ControlCommand input value must coincide with the speed driver configuration.

Failure to follow these instructions can result in equipment damage.

MBATVDev Specific Inputs

Inputs Description Table

This table describes the inputs of the MBATVDev FBD:

Input	Type	Description
CommunicationOK	BOOL	A bit is used to indicate whether or not the node is present on the bus.
ModbusAddress	INT	Device address within the Modbus network. This variable can be found in Modbus communications.
StatisticSelector	INT	StatisticSelector *(see page 203)* is a variable used to obtain statistics for the Modbus network.
TimeControl	MB_DEV_TIME_ PARA_DDT	TimeControl *(see page 51)* is a parameter for device control.
Range	SCALE_DDT	Range *(see page 53)* definition for inverter (Raw and engineering range).

Input Detail - StatisticsSelector

Description

This variable is used to obtain statistics for the Modbus network (requests carried out, time between requests, etc.).

This data provides information for using the StatisticsConnector with the Communications library "Statistics Module."

StatisticsSelector Structure Description

This table describes the structure of the StatisticsSelector:

Variable Value	Type	Description
1	INT	Read Statistics, client
2	INT	Write Statistics, client

9.3 Outputs

Overview

This section describes the outputs of MBATVDev FBD.

What's in this Section?

This section contains the following topics:

Generic Outputs Description

Overview

This table is the same for all ATV Profile.

Outputs Description Table

This table describes the outputs of the ATVProfile FBD:

Output	Type	Description
Ready	BOOL	This output indicates if the device is enabled and free of detected errors. The device is ready to carry out or is carrying out a run or stop command. If QuickStop is enabled, this output is reset to 0. This variable is TRUE as long as there is no detected communications error and the device is in status 5 of the DSP402.
Fail	BOOL	This output indicates a detected error in the control block (parameter configuration, cannot be reset), detected error in the device, or a detected communications error. To reset the detected error, the ResetFail input has to be activated. The last detected error code is in FailCode.
Warning	BOOL	This output indicates if an alarm has been activated for the device. It does not affect the blocks operation and does not need to be reset. The signal remains active until the cause for the alarm disappears. The output is set to 0 only when the alarm is acknowledge.
FailCode	ARRAY	This output indicates the detection of an error. When this output is activated, this variable contains the code for the current detected error. If this output is not activated, this variable contains the last detected error. The detected error source is specified by a 3-level structure.
WarningCode	ATVWarning	The WarningCode *(see page 60)* output is a data structure that contains information about the alarm currently on the speed of the drive.
Running	BOOL	This output indicates that the speed of the drive is running and has an output frequency. **NOTE:** If SetPoint is 0 and Run is active, the Running signal is activated.

Output	Type	Description
ExtControlled	BOOL	Indicates if the drive is controlled from an external source (a console, a push-button panel or a monitoring system) to the system (1) or not (0). It provides information for programming.
Resetting	BOOL	The Resetting *(see page 61)* output indicates whether or not a reset is being carried out.
Status	ATVStatus	The Status *(see page 62)* output holds a structure with the information that the module extracts from the drives status variable.
PresentValue	REAL	Indicates the current rotation speed in EU (Engineering Units).
PresentSpeed	REAL	Indicates the current motor rotation speed in rpm.
NOTE: The variables can not be read from the device when there is a detected communications error. The variables keep their last value.		

MBATVDev Specific Output

Output Description Table

This table describes the general output of the MBATVDev FBD.

Output	Type	Description
StatisticConnector	Statistic Connector	The StatisticConnector *(see page 208)* represents information that is used with Modbus communications to obtain statistics on the Modbus network.

Output Detail - StatisticConnector

Description

The StatisticConnector represents information that is used with Modbus communications to obtain statistics on the Modbus network (requests carried out, time between requests, etc.).

This structure has been created for its use together with the Communications library Statistics module.

StatisticConnector Structure Description

This table describes the structure of StatisticConnector:

Command	Type	Description
Start	BOOL	Operation has started
EndOk	BOOL	Operation ended correctly
EndNOk	BOOL	Operation ended with an error
PartialTime	DINT	Partial Time

9.4 Inputs/Outputs

Generic Inputs/Outputs Description

Overview

This table is the same for all ATV Profiles.

Inputs/Outputs Description Table

This table describes the inputs/outputs of the ATV Profile FBD:

Name	Type	Description
ATV_ST	ATV_ST_DDT	The ATV_ST *(see page 67)* device data structure contains the minimum information required for performing control and monitoring. The information is used by the operator screen and can be read from HMI or SCADA.
ATV_CFG	ATV_CFG_DDT	The ATV_CFG *(see page 69)* device data structure contains the information used by the operator screen and can be read/write from HMI or SCADA.

9.5 Operator Screen

General

Description

The component includes an operator screen that allows interaction with the communication interface blocks

- ATV_ST_DDT and ATV_CFG_DDT - Inputs/Outputs structures.
- ATV_IO_DDT and ATV_IOEXT_DDT - Output structures.

9.6 Error Management

Detected Error Code

Description

The detected error codes the device can return are read from the FailCode output variable.

The detected error codes are the same as for the ATVDev Error Management *(see page 74)*.

9.7 SCoD Representation and Parameter Description

SCoD Representation and Parameter Description

Overview

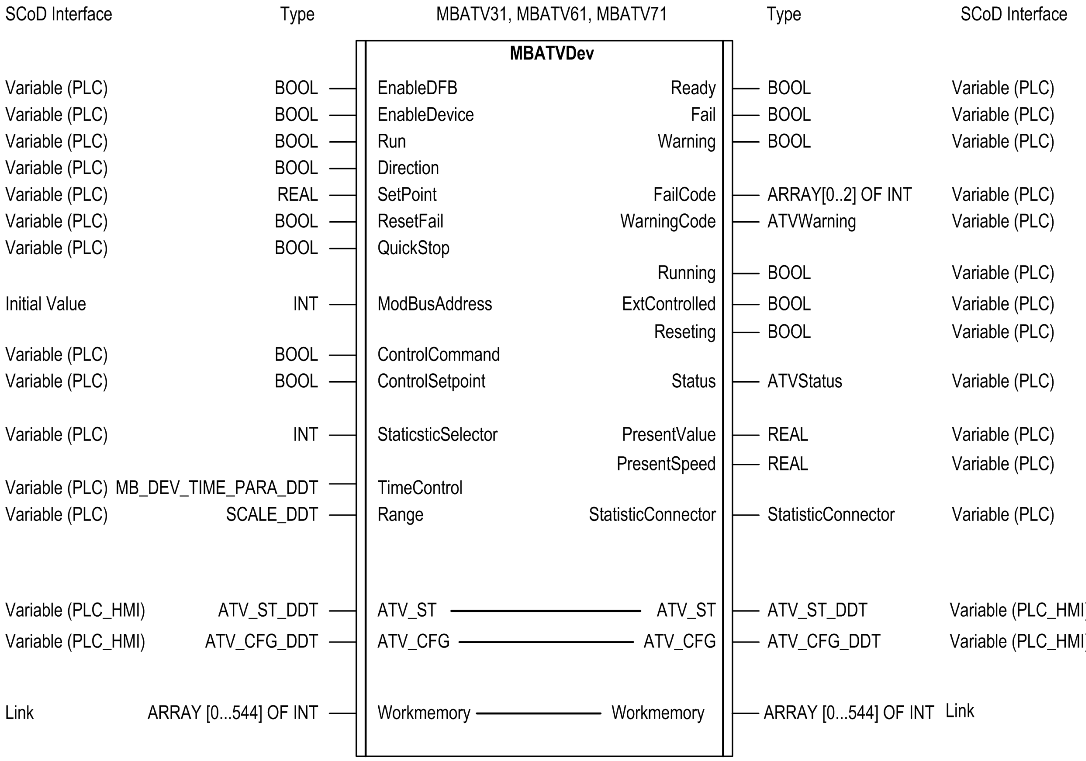
The usage of the DFB pins in the SCoD editor is explained in the input/output diagram below.

The variables defined in the SCoD editor have in principle the same name as the pin (except that an underscore in the pin name is skipped for the variable name).

If the variables defined in the SCoD have the property 'Initial Value' then the values of the instantiated SCoD variables can be modified manually in Unity Pro before PLC restart.

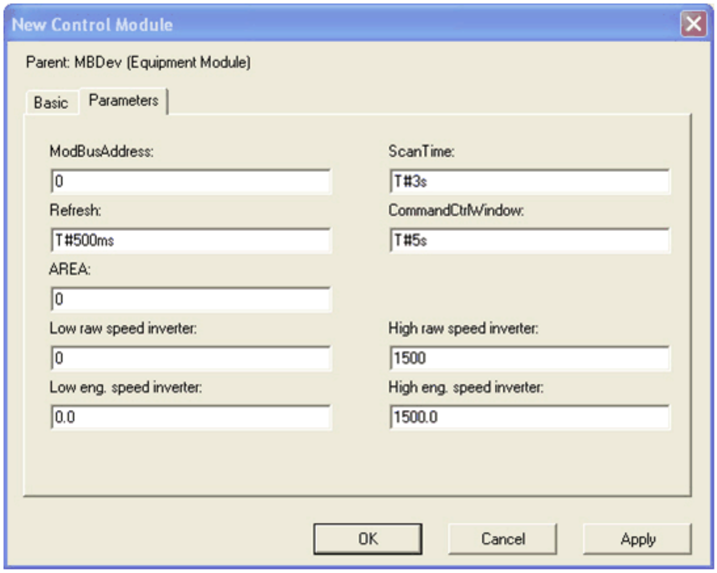
SCoD Representation

This figure represents the MBATV31, MBATV61 and MBATV71 SCoDs:



Parameter Tab

This figure shows the MBATV31,MBATV61 and MBATV71 parameter tab:



This table describes the properties:

Property	Type	Description
ModbusAddress	INT	This property represents the device address on the Modbus network. This variable can be found in Modbus communications.
ScanTime	TIME	Minimum time to maintain detected error signals
Refresh	TIME	Refresh time for device read data on Modbus communications
CommandControlWindow	TIME	The time waited before executing orders
AREA	BYTE	Area to which the object that you wish to instantiate belongs. Enables to control access to the functions provided by this object.
Low raw Speed Inverter	INT	Low range value measured in raw units
High raw Speed Inverter	INT	High range value measured in raw units
Low eng. Speed Inverter	REAL	Low range value measured in engineering units
High eng. Speed Inverter	REAL	High range value measured in engineering units

9.8 HMI Representation

MBATV- Representation in the HMI

Modbus ATV (Genie)

The Modbus ATV block is represented in the HMI by the following genie.

Genie Elements

Symbol	Meaning
	Opens the Modbus ATV popup. Displays the device name, that is, ATV and PV value.
	The instance name of the Modbus ATV block is only visible by selecting the **Hide**/**Unhide** button in the template.
	When there is an unacknowledged note, the ! symbol is displayed in the genie.
	When there is an active and unacknowledged alarm or communication failure, then outer rectangle flashes in red.
	When there is an active and unacknowledged alarm or communication failure, then outer rectangle is steady red.

Popup

The Modbus ATV block popup has the following tabs:
- **Operator** tab
- **Analog** tab
- **Digital** tab
- **Information** tab

Operator Tab

For detailed description, refer to **Operator Tab** of ATV31 *(see page 80)* and ATV7161 *(see page 105)* .

Analog Tab

For detailed description, refer to Analog Tab *(see page 82)*.

Digital Tab

For detailed description, refer to Digital Tab *(see page 83)*.

Information Tab

For description about **Information** tab, refer to Commonly Used Tabs *(see page 32)*.

MBPM710, MBPM800 - Modbus PM710, PM800 (MBPM710, MBPM800 SCoDs)

10

Overview

This chapter describes the DFB for the control blocks for the PM 710 and PM800 family power meters on a Modbus network, represented by MBPM710 and MBPM800 SCoDs.

What's in this Chapter?

This chapter contains the following sections:

10.1 Description

Overview

This section describes the MBPM710Dev DFB and MBPM800Dev DFB.

What's in this Section?

This section contains the following topics:

Functional Description

General

The PM profile is part of the device control block family.

The MBPM700Dev and MBPM800Dev blocks are implemented as listed below:

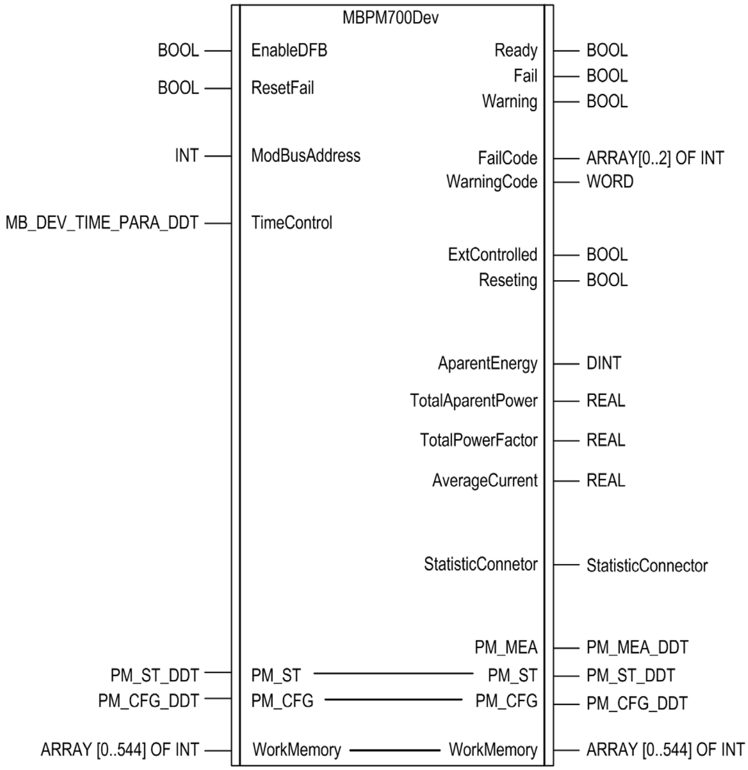
Software	Implemented as
Unity Pro	DFB MBPM710Dev, MBPM800Dev
Vijeo Citect	DPC_Devices
UAG	Depending on the power meters: ● SCoD MBPM710 ● SCoD MBPM800

The main function blocks are described in the PM Profile Functional Description *(see page 31)*.

FBD Representation

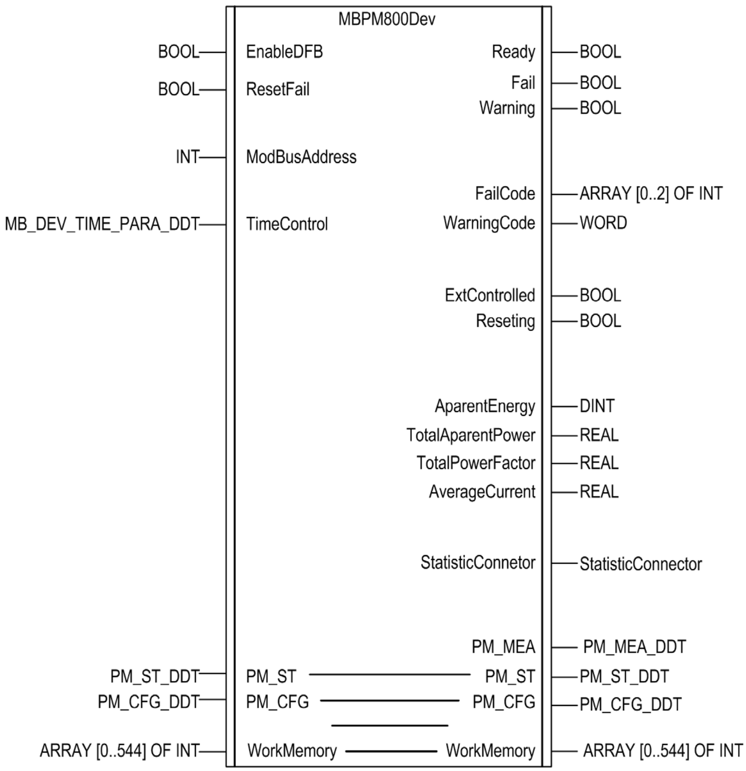
MBPM710Dev Representation in FBD

The functional module that will be used in the program has the following aspect (at the FBD block level) after PLC generation in UNITY Pro:



MBPM800Dev Representation in FBD

The functional module that will be used in the program has the following aspect (at the FBD block level) after PLC generation in UNITY Pro:



10.2 Inputs

Overview

This section describes the inputs of the MBPM710Dev and MBPM800Dev FBD.

What's in this Section?

This section contains the following topics:

Inputs

Inputs Description Table

This table describes the inputs of the MBPM710Dev and MBPM800Dev control blocks:

Input	Type	Description
EnableDFB	BOOL	The input enables the normal execution of the control block. When this input is disabled, the entire DFB is restarted (Statuses, Output values, counters, etc. are lost) and all the output values are set to 0 or FALSE as long as the signal remains on FALSE. Activating this input enables communications with devices for their operation. The Public variable values are loaded during the first enabling cycle.
ResetFail	BOOL	Current detected errors are reset on the rising edge of this signal. Both Communications and detected device errors can be reset. If the detected error is a device error, it sends a reset command to the device if ControlCommand = TRUE. This figure shows the input ResetFail: Fail AND (NOT Reseting)——| | ResetFail
ModbusAddress	INT	Device address within the Modbus network.
StatisticSelector	INT	The input StatisticsSelector *(see page 224)* is a variable used to obtain statistics for the Modbus network (requests carried out, time between requests, etc.).

Input Detail - StatisticsSelector

Description

StatisticSelector is a variable used to obtain statistics for the Modbus network (requests carried out, time between requests, etc.).

This data provides information for using the StatisticsConnector with the Communications library "Statistics Module".

This table describes the variables:

Variable Value	Description
1	Read Statistics, client
2	Write Statistics, client

10.3 Outputs

Overview

This section describes the outputs of the MBPM710Dev and MBPM800Dev FBD.

What's in this Section?

This section contains the following topics:

Outputs Table Description

General

This table describes the outputs of the MBPM710Dev and MBPM800Dev FBD:

Output	Type	Description
Ready	BOOL	This output indicates if the device is enabled and free of detected errors. The device is ready to carry out any read operation.
Fail	BOOL	This output indicates a detected error in the control block (Parameter configuration, cannot be reset) or a detected communications interruption. To reset the detected error, the ResetFail input has to be activated. The last error code is shown on FailCode.
Warning	BOOL	This output indicates an alarm has been activated for the drive. It does not affect the blocks operation and does not need to be reset. This signal remains active until the cause for the alarm disappears.
WarningCode	WORD	The WarningCode *(see page 227)* is a variable that holds the alarm.
ExtControlled	BOOL	Indicates if the device is controlled from an external source (from the monitoring system) to the system (1) or not (0).
Resetting	BOOL	The Resetting *(see page 228)* output indicates whether or not a reset is carried out.
ApparentEnergy	DINT	Apparent power consumption (kVAh)
TotalApparentPower	REAL	Total apparent power (kVA)
TotalPowerFactor	REAL	Total power factor
AverageCurrent	REAL	Current, average between the 3 phases (Volt)
StatisticConnector	StatisticConnector	The StatisticConnector *(see page 229)* represents information that is used with Modbus communications to obtain statistics on the modbus network.
PM_MEA	PM_MEA_DDT	The PM_MEA *(see page 230)* is a data structure that contains device measurement.
NOTE: The variables can not be read from the device when there is a detected communications error. The variables keep their last value.		

Output Detail - WarningCode

Description

The output WarningCode is a variable that holds the alarm code.

WarningCode Word Description

This table shows the PM700 (W 4112) Description:

Bit	Description for PM700 (W 4112)
0	Phase 1 voltage out of range
1	Phase 2 voltage out of range
2	Phase 3 voltage out of range
3	Phase 1 current out of range
4	Phase 2 current out of range
5	Phase 3 current out of range
6	Frequency out of range, or phase 1 voltage insufficient to determine frequency.

This table shows the PM700 (W 3254) Description:

Bit	Description for PM700 (W 3254)
0	Summary bit (activated if any other bit is activated)
1	Configuration error
2	Scaling error
3	Phase dropout
4	Cabling error
5	Incremental energy could be incorrect due to the meters resetting
6	External demand synchronization waiting time

Output Detail - Resetting

Description

This output indicates whether or not a reset is carried out.

The CommandCtrlWindow variable indicates the maximum time for resetting the detected error.

When a device or communications reset is carried out with ResetFail, the DFB tries to reset the error within the time period defined in CommandCtrlWindow.

If the detected error is reset:

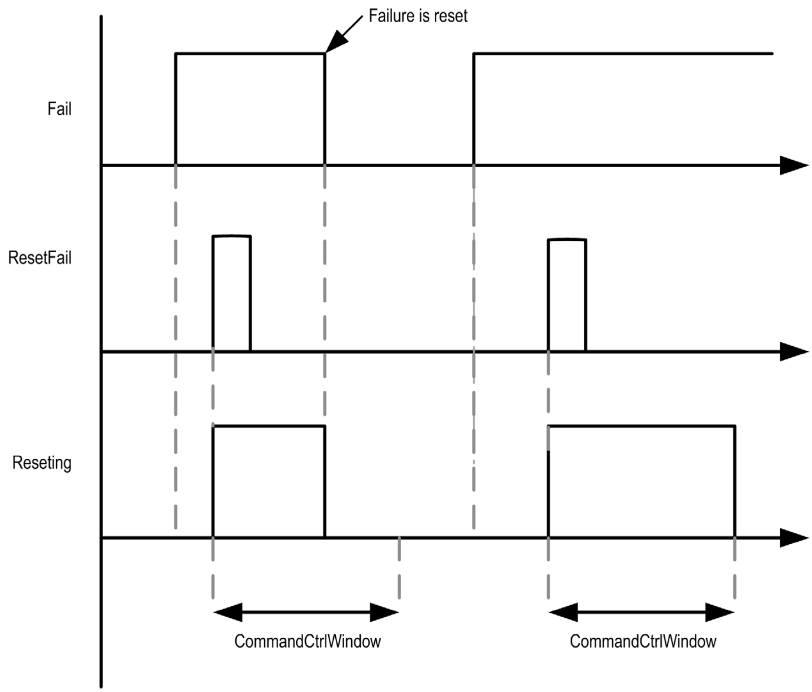
- The Fail and Resetting output variables are reset (they are set to FALSE).

If the detected error is not reset:

- The Resetting variable is set to FALSE.
- The Fail variable is held as TRUE.

ResetFail is edge-based.

This figure shows the Resetting output:



Output Detail - StatisticConnector

Description

The StatisticConnector represents information that is used with Modbus communications to obtain statistics on the Modbus network (requests carried out, time between requests, etc.).

This data provides information for using the StatisticConnector with the Communications library “Statistics Module”.

StatisticConnector Structure Description

This table shows the StatisticConnector structure:

Name	Type	Description
Start	BOOL	Operation has started
EndOk	BOOL	Operation ended correctly
EndNOK	BOOL	Operation ended with an error
PartialTime	DINT	Partial time

Output Detail - PM_MEA

Description

The output PM_MEA is a data structure that contains device measurement information.

Name	Type	Description
ActiveEnergy	REAL	Real power consumption
ReactiveEnergy	REAL	Reactive power consumption
TotalActivePower	REAL	Total real power
TotalReactivePower	REAL	Total reactive power
AverageLineToLineVoltage	REAL	Voltage, L-L, average between the 3 phases
AverageLineToNeutralVoltage	REAL	Voltage, L-N, average between the 3 phases
Frequency	REAL	Frequency (derived from phase1)
ActualCurrentL1	REAL	Current, instantaneous, phase 1
ActualCurrentL2	REAL	Current, instantaneous, phase 2
ActualCurrentL3	REAL	Current, instantaneous, phase 3
VoltageL1TOL2	REAL	Voltage, phases 1-2
VoltageL2TOL3	REAL	Voltage, phases 2-3
VoltageL1TOL3	REAL	Voltage, phases 1-3
VoltageL1TONeutral	REAL	Voltage, phase 1-N
VoltageL2TONeutral	REAL	Voltage, phase 2-N
VoltageL3TONeutral	REAL	Voltage, phase 3-N
ActivePowerL1	REAL	Real power, phase 1
ActivePowerL2	REAL	Real power, phase 2
ActivePowerL3	REAL	Real power, phase 3
ApparentPowerL1	REAL	Apparent power, phase 1
ApparentPowerL2	REAL	Apparent power, phase 2
ApparentPowerL3	REAL	Apparent power, phase 3
ReactivePowerL1	REAL	Reactive power, phase 1
ReactivePowerL2	REAL	Reactive power, phase 2
ReactivePowerL3	REAL	Reactive power, phase 3
THDCurrentL1	REAL	THD, current, phase 1
THDCurrentL2	REAL	THD, current, phase 2
THDCurrentL3	REAL	THD, current, phase 3
THDVoltageL1ToNeutral	REAL	THD, voltage, 1-N

Name	Type	Description
THDVoltageL2ToNeutral	REAL	THD, voltage, 2-N
THDVoltageL3ToNeutral	REAL	THD, voltage, 3-N
THDVoltageL1ToL2	REAL	THD, voltage, 1-2
THDVoltageL2ToL3	REAL	THD, voltage, 2-3
THDVoltageL1ToL3	REAL	THD, voltage, 1-3

10.4 Inputs/Outputs

Overview

This section describes the Inputs/Outputs of the MBPM710Dev and MBPM800Dev FBD.

What's in this Section?

This section contains the following topics:

Inputs/Outputs

Inputs/Outputs Description Table

This table describes the inputs/outputs of the MBPM710Dev and MBPM800Dev DFB:

Name	Type	Description
PM_ST	PM_ST_DDT	The PM_ST *(see page 234)* device data structure that holds the minimum information required for performing control and monitoring. The information is used by the operator screen and can be read from HMI or SCADA.
PM_CFG	PM_CFG_DDT	The PM_CFG *(see page 236)* device data structure holds information used by the operator screen. It can be read from HMI or SCADA.

Inputs/Outputs Detail - PM_ST

Description

The device data structure (PM_ST_DDT) holds the minimum information required for performing control and monitoring.

The information is used by the operator screen and is read/write from HMI or SCADA.

PM_ST_DDT Structure Description

This table shows the PM_ST_DDT structure:

Name	Type	Description
STW	WORD	Device status
CFGW	WORD	Device control
ApparentEnergy	REAL	Apparent power consumption (kVAh)
TotalApparentPower	REAL	Total apparent power (kVA)
TotalPowerFactor	REAL	Total power factor
AverageCurrent	REAL	Current, average between the 3 phases (Volt)

STW Word Description

The STW Word provides the device status to display in the foreground the monitoring subsystem.

Access to the data held in this bit word is read-only.

This table shows the STW Word description:

Bit	Description
0	Unknown device status or communications interruption. No variable refreshing.
1	Not Ready
2	Module running
3	Detected device error
4	Alarm on device or follow-up detected fault (these require resetting)
5	Communications interruption
6	ResetFail required
8	Resetting
9	EnableDFB
5	Communications interruption

CFGW Word Description

CFGW provides the means necessary to control the device.

Owner Value	Control is from
1	supervision subsystem or from the opeartor screen
0	supervision subsystem, the DFB input variables values are used for reading from HMI or SCADA

This table shows the CFGW Word Description:

Bit	Description
0	ResetFail
1	Owner

Inputs/Outputs Detail - PM_CFG

Description

This function is a data structure (PM_CFG_DDT) that contains device information.

The information is used by the operator screen and can be read from HMI or SCADA.

PM_CFG_DDT Structure Description

The PM_CFG_DDT data structure contains device information that is used by the operator screen and can be read from HMI or SCADA.

This table shows the PM_CFG_DDT structure:

Name	Type	Description
WarningCode	WORD	Alarm code
FailCode 0	INT	Code of last level 0 detected error
FailCode 1	INT	Code of last level 1 detected error
FailCode 2	INT	Code of last level 2 detected error

WarningCode Word Description

PM alarm code information.

Takes values from the WarningCode output.

This table shows the PM700 description:

Bit	Description for PM700
0	Phase 1 voltage out of range
1	Phase 2 voltage out of range
2	Phase 3 voltage out of range
3	Phase 1 current out of range
4	Phase 2 current out of range
5	Phase 3 current out of range
6	Frequency out of range, or phase 1 voltage insufficient to determine frequency.

This table shows the PM820 description:

Bit	Description for PM820
0	Summary bit (activated if any other bit is activated)
1	Configuration interruption

Bit	Description for PM820
2	Scaling interruption
3	Phase dropout
4	Cabling interruption
5	Incremental energy could be incorrect due to the meters resetting
6	External demand sinc waiting time

10.5 Operator Screen

General

Description

The component includes an operator screen that enables interaction with the communication interface blocks.

The operator screen data is used to deal with a monitoring system by means of the following:

- PM_ST and PM_CFG - Inputs/Outputs structures
- PM_MEA - Inputs/Outputs structures

10.6 Error Management

Detected Error Code Description

Overview

The detected error codes the device can return are read from the ErrorCode output variable.

Parameter Configuration Detected Error Codes

This detected error indicates that a parameter configuration public variable has a value that is not allowed.

To reload new values, an edge is required on the EnableDFB variable that cannot be reset:

- FailCode[0]: 16#0003
- FailCode[1]: 16#0000
- FailCode[2]: 16#0004

Modbus Configuration Detected Error Codes

For Modbus, this code is used to indicate that communications have not been established.

It can be reset.

- FailCode[0]: 16#0002
- FailCode[1]: 16#0000
- FailCode[2]: 16#0004

Once Modbus communications have been established, check “Modbus client” detected error codes for FailCode[0] and [1].

All components make a distinction between read and write requests.

- FailCode[2]: 16#0001 - Read
- FailCode[2]: 16#0002 - Write

10.7 SCoD Representation and Parameter Description

SCoD Representation and Parameter Description

Overview

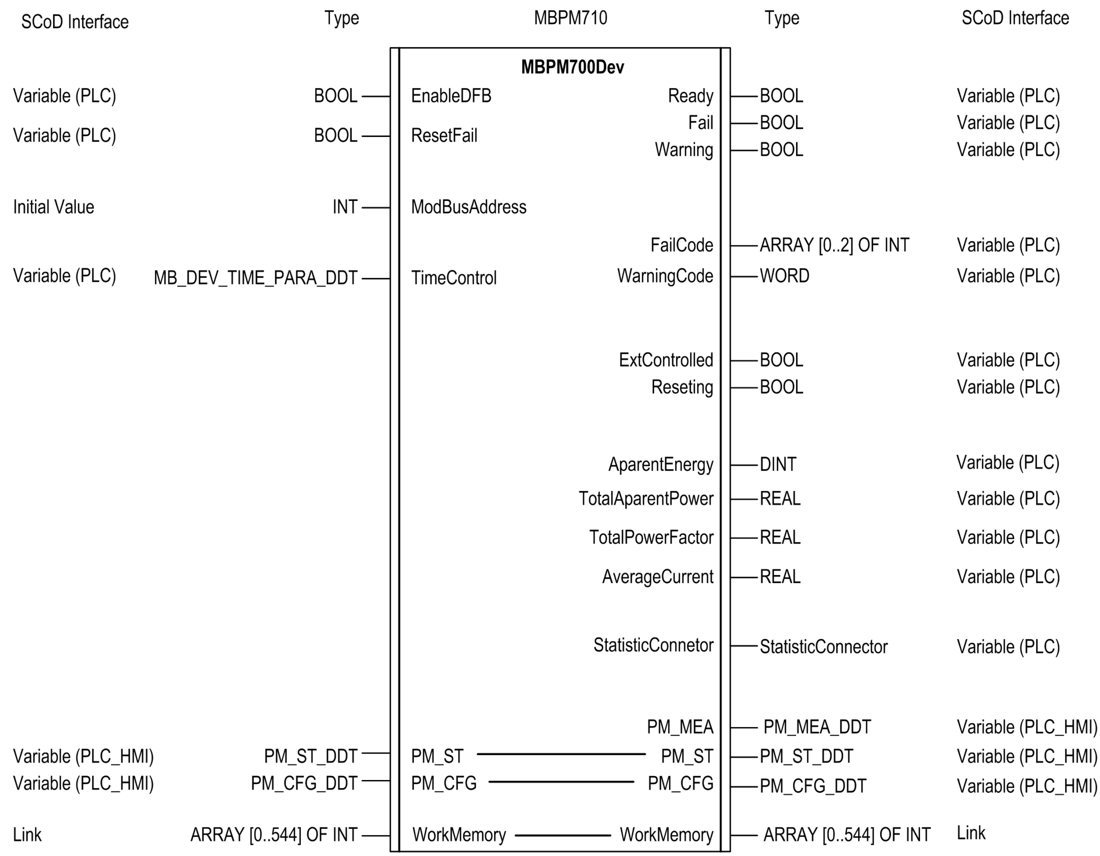
The usage of the DFB pins in the SCoD editor is explained in the input/output diagram below.

The variables defined in the SCoD editor have in principle the same name as the pin (except that an underscore in the pin name is skipped for the variable name).

If the variables defined in the SCoD have the property 'Initial Value' then the values of the instantiated SCoD variables can be modified manually in Unity Pro before PLC restart.

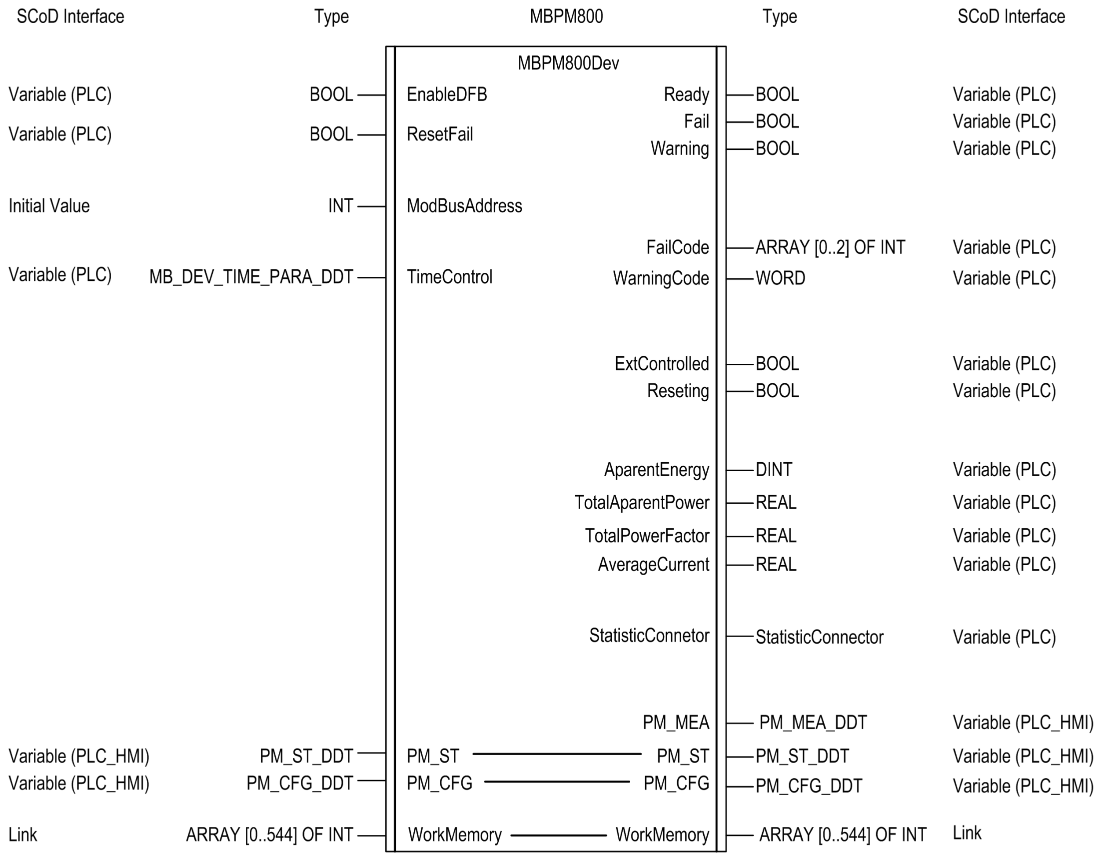
MBPM710 SCoD Representation

This figure represents the MBPM710 SCoD:



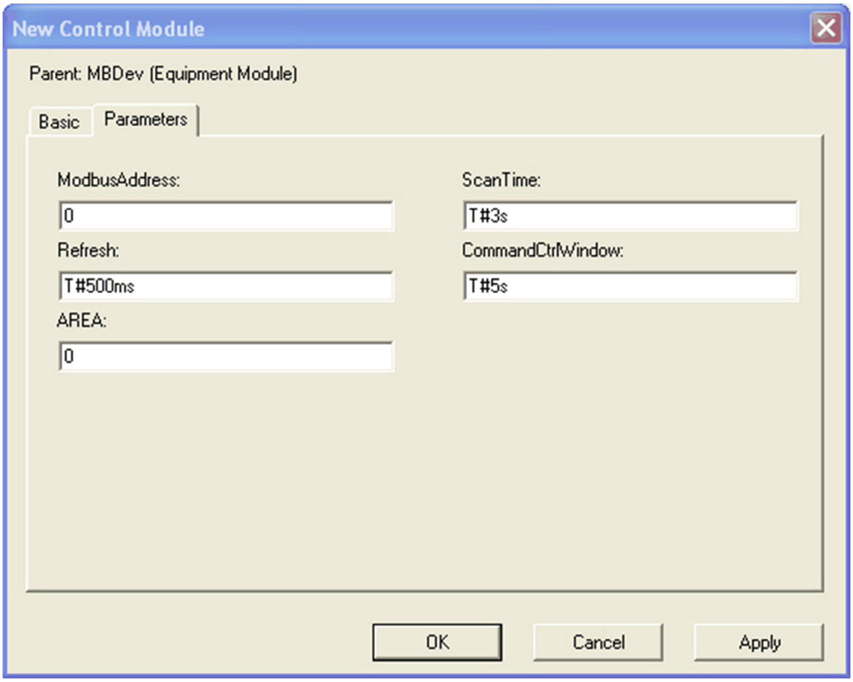
MBPM800 SCoD Representation

This figure represents the MBPM800 SCoD:



Parameter Tab

This figure shows the MBPM710, MBPM800 parameter tab:



The following table describes the properties:

Property	Type	Description
ModbusAddress	INT	Device address within the Modbus network. This variable can be found in Modbus communications.
ScanTime	TIME	Minimum time to maintain detected error signals
Refresh	TIME	Refresh time for device read data on Modbus communications
CommandControlWindow	TIME	The time waited before executing orders
AREA	BYTE	Area to which the object that you wish to instantiate belongs. Enables to control access to the functions provided by this object.

10.8 HMI Representation

MBPM710, MBPM800 - Representation in the HMI

Modbus Power Meter (Genie)

The Modbus Power Meter block is represented in the HMI by the following genie.

Genie Elements

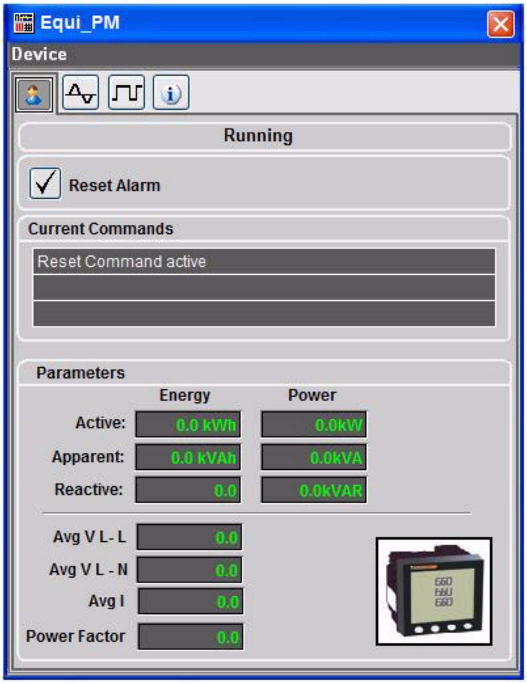
Symbol	Meaning
	Opens the Modbus Power Meter popup. Displays the device name, that is, power meter and active power value.
	The instance name of the Modbus Power Meter block is only visible by selecting the **Hide**/**Unhide** button in the template.
	When there is an unacknowledged note, the ! symbol is displayed in the genie.
	When there is an active and unacknowledged alarm or communication failure, then outer rectangle flashes in red.
	When there is an active and unacknowledged alarm or communication failure, then outer rectangle is steady red.

Popup

The Modbus Power Meter block popup has the following tabs:

- **Operator** tab
- **Analog** tab
- **Digital** tab
- **Information** tab

Operator Tab



This tab has 3 sub-sections:

- **Status**
- **Reset Alarm**
- **Current Commands**
- **Parameters**

The sub-section **Status** displays the current status.

The possible displays are listed as follows:

- **Unknown Status**
- **Not Ready**
- **Running**
- **Communication Failure**
- **Failure**
- **Warning**

The following table describes the sub-section **Reset Alarm**:

Item	Description
Reset Alarm	Resets alarm.

The following table describes the sub-section **Current Commands**:

Item	Description
Reset Command active	Displays the text in green color when the operation is active. Displays the text in white color when the operation is not active.

The following table describes the sub-section **Basic Information**:

Item	Description
Active	Displays the values of active energy and active power.
Apparent	Displays the values of apparent energy and apparent power.
Reactive Ind	Displays the values of reactive inductive energy and reactive inductive power.
Reactive Cap	Displays the values of reactive capacitive energy and reactive capacitive power.
Avg V L-L	Displays the value of average line to line voltage.
Avg V L-N	Displays the value of average line to neutral voltage.
Avg I	Displays the value of average current.
FactorPow	Displays the value of power factor.

The sub-section **Parameters** also shows the picture of power meter.

Analog Tab

For detailed description, refer to Analog Tab *(see page 167)*.

Digital Tab

For detailed description, refer to Digital Tab *(see page 168)*.

Information Tab

For description about **Information** tab, refer to Commonly Used Tabs *(see page 32)*.

MBTESYST - Modbus Tesys T Motor Management (MBTESYST SCoD)

11

Overview

This chapter describes the DFB for the control block for the Tesys T motor controller on a Modbus network, represented by MBTESYST SCoD.

What's in this Chapter?

This chapter contains the following sections:

11.1 Description

Overview

This section describes the MBTESYSTDev DFB.

What's in this Section?

This section contains the following topics:

MBTESYSTDev - Functional Description

General

The MBTESYSTDev profile block is the control block for TesysT controller on Modbus communication.

The objective of these functions is to manage the TesysT family of devices in a communications-based manner.

The TesysTDev block is implemented as listed below:

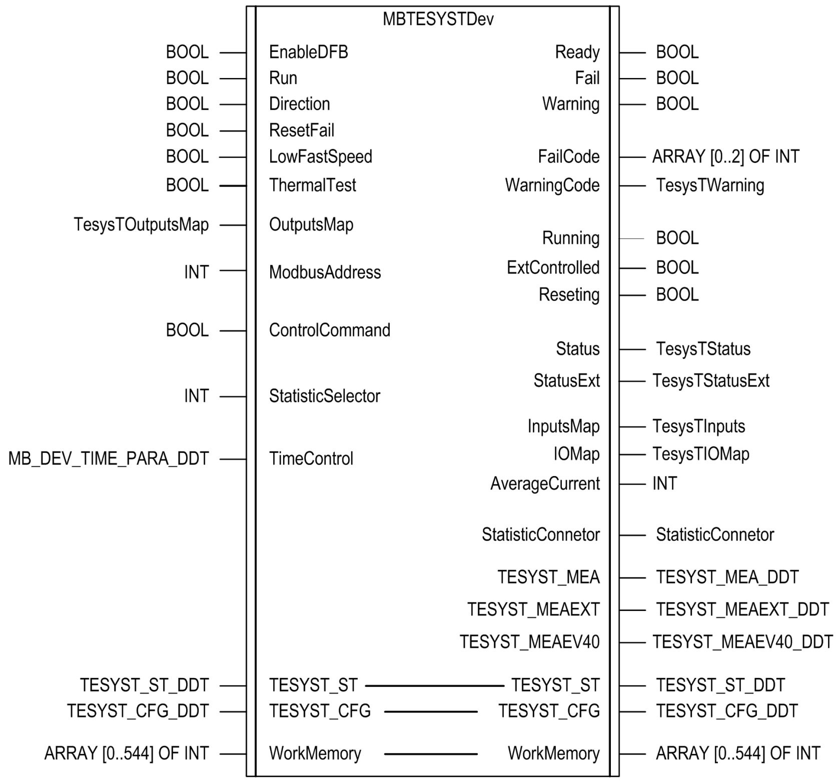
Software	Implemented as
Unity Pro	DFB TesysTDev
Vijeo Citect	DPC_Devices
UAG	Depending on the controller: • SCoD MBTESYST for Advance controller.

The main function blocks are described in the TesysT Profile Functional Description *(see page 30)*.

FBD Representation

Representation in FBD

The functional module that will be used in the program has the following aspect (at the FBD block level) after PLC generation in UNITY Pro:



11.2 Inputs

Overview

This section describes the inputs of the MBTESYSTDev FBD.

What's in this Section?

This section contains the following topics:

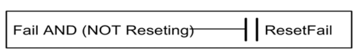
Generic Inputs Description

Overview

This table is the same for all TesysT Profiles.

Inputs Description Table

This table describes the inputs of the TesysT Profile FBD:

Input	Type	Description
EnableDFB	BOOL	This input enables the normal execution of the control block. EnableDFB = 0. The entire DFB is restarted (statuses, output values, counters, etc. are lost) and all output values are set to 0 or FALSE. EnableDFB = 1. This input enables communications with devices for their operation. Public variable values are loaded during the first enabling cycle.
Run	BOOL	This input makes the motor run in the direction selected with the Direction variable.
Direction	BOOL	This input activates the direction of motor rotation: **1** = forward **0** = reverse **NOTE:** Setpoint assumes positive values. Changes in the direction of rotation cannot be made with Setpoint sign.
ResetFail	BOOL	Existing detected errors are reset on a rising edge of this signal. Both Communications and detected device errors can be reset. If the detected error is a device error, it sends a reset command to the device if ControlCommand = TRUE. This figure shows the input ResetFail: 
LowFastSpeed	BOOL	Slow/fast speed of rotation. For 2-speed motor start configurations. Check the TesysT user manual.
ThermalTest	BOOL	This bit is used to check the forced thermal detected error input on the Tesys T.

Input	Type	Description
OutputsMap	TesysUOutputsMap	OutputsMap *(see page 116)* holds a structure used to control the Tesys outputs. If one of the variables is not available, the comment will indicate that the information cannot be accessed.
ControlCommand	BOOL	This input indicates if the starter is controlled from an external source. If ControlCommand=TRUE the device is controlled from an external source. The generation of follow-up alarms is disabled. If ControlCommand=FALSE, the block only performs read operations on the device status, the transmission of device control or reset commands is not possible. **NOTE:** This input does not configure the starter; it only defines the DFB operating mode.
TimeControl	CAN_DEV_TIME_ PARA_DDT	TimeControl *(see page 119)* is a parameter for device control.

⚠ WARNING

UNEXPECTED MOTOR START

Reset the Run variable before resuming operation.

The Run input must be inactive while the device is being enabled or while a detected error is being reset.

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

NOTICE

UNEXPECTED BEHAVIOUR

The ControlCommand input value must coincide with the speed driver configuration.

Failure to follow these instructions can result in equipment damage.

Specific MBTESYSTDev Inputs

Inputs Description Table

This table describes the inputs of the MBTESYSTDev FBD:

Input	Type	Description
ModbusAddress	INT	Device address within the Modbus network. This variable can be found in Modbus communications.
StatisticsSelector	INT	StatisticsSelector *(see page 255)* is a variable used to obtain statistics for the Modbus network.

Input Detail - StatisticsSelector

Description

Variable used to obtain statistics for the Modbus network (requests carried out, time between requests, etc.).

This data provides information for using the StatisticsConnector with the Communications library "Statistics Module."

StatisticsSelector description depending on the value:

Variable Value	Description
1	Read Statistics, client
2	Write Statistics, client

11.3 Outputs

Overview

This section describes the outputs of the MBTESYSTDev FBD.

What's in this Section?

This section contains the following topics:

Generic Outputs Description

Overview

This table is the same for all TesysT Profiles.

Outputs Description Table

This table describes the Outputs of the TesysT Profile FBD:

Output	Type	Description
Ready	BOOL	This output indicates if the device is enabled and free of errors. The device is ready to carry out or is carrying out any run or stop command. If QuickStop is enabled, this output is reset to 0.
Fail	BOOL	This output indicates an detected error in the control block (Parameter configuration, cannot be reset) or detected error in the device, or a detected communications error. In order to reset the detected error, the ResetFail input has to be activated. The last detected error code is shown on FailCode.
Warning	BOOL	This output indicates if an alarm has been activated for the device. It does not affect the blocks operation and does not need to be reset. The signal remains active until the cause for the alarm disappears.
FailCode	ARRAY	This output indicates the detection of an error. When this output is activated, this variable contains the code for the current detected error. If this output is not activated, this variable holds the last detected error. The detected error source is specified by a 3-level structure.
WarningCode	TesysTWarning	The WarningCode *(see page 125)* output holds a data structure that contains information about the alarm currently on the controller.
Running	BOOL	This output indicates that the Tesys is running. Current higher than 10% of FLA for Advanced and Multifunction controllers.
ExtControlled	BOOL	This output indicates if the device is controlled from an external source (for example, from the console, from a push-button panel, or from the monitoring system) to the system (1) or not (0). Provides information for programming.

Output	Type	Description
Reseting	BOOL	The Reseting *(see page 126)* output indicates whether or not a reset is carried out.
Status	TesysUStatus	The Status *(see page 127)* output holds a structure with the information that the module extracts from the status variable of controllers with Advanced and Multifunction control modules.
IOMap	TesysTIOMap	The IOMap *(see page 130)* indicates current speed of rotation in EU (Engineering Units).
NOTE: The variables can not be read from the device when there is a detected communications error. The variables keep their last value.		

Specific MBTESYSTDevOutputs

Outputs Description Table

This table describes the Outputs of the MBTESYSTDev FBDk:

Output	Type	Description
StatisticConnector	StatisticConnetor	The StatisticConnector *(see page 260)* represents information that is used with Modbus communications to obtain statistics on the Modbus network.
TESYST_MEAEXT	TESYST_MEA EXT_ DDT	TESYST_MEAEXT *(see page 261)* is a data structure that contains extensive device information related to Modbus communications.
TESYST_MEAEV40	TESYST_MEA EV40_ DDT	TESYST_MEAEV40 *(see page 262)* is a data structure that contains device information related to an EV40 expansion module with Modbus communications.

Output Detail - StatisticConnector

Description

The StatisticConnector represents information that is used with Modbus communications to obtain statistics on the Modbus network (requests carried out, time between requests, etc.).

This data provides information for using the StatisticsConnector with the Communications library “Statistics Module.”

StatisticConnector Structure Description

This table shows the StatisticConnector structure:

Name	Type	Description
Start	BOOL	Operation has started
EndOk	BOOL	Operation ended correctly
EndNOK	BOOL	Operation ended with an detected error
PartialTime	DINT	Partial time

Output Detail - TESYST_MEAEXT

Description

This function holds a data structure (TESYST_MEAEXT_DDT) that contains extended information on the device on Modbus communications.

TESYST_MEAEXT_DDT Structure Description

This table shows the TESYST_MEAEXT_DDT structure:

Name	Type	Description
FLCCurrent L1	INT	L1 current (%FLC)
FLCCurrent L2	INT	L2 current (%FLC)
FLCCurrent L3	INT	L3 current (%FLC)
FLCCurrent L4	INT	Internal controller temperature (° C)
Temperature	INT	Frequency (Hz)
Current	REAL	Average current (A)
Current L1	REAL	L1 Current (A)
Current L2	REAL	L2 Current (A)
Current L3	REAL	L3 Current (A)
GroundCurrent	INT	GroundCurrent (A)
MotorTemperature	INT	Motor sensor temperature (%)
TimeTrip	INT	Trip time (x 1s)
LastStartCurrent	INT	Last starting current (%FLC)
LastStartDuration	INT	Last amount of time that it took the motor to start (s)

Output Detail - TESYST_MEAEV40

Description

This function holds a data structure (TESYST_MEAEV40_DDT) that contains information on the device with an EV40 expansion module on Modbus communications.

TESYST_MEAEV40_DDT Structure Description

This table shows the TESYST_MEAEV40_DDT structure:

Name	Type	Description
PowerFactor	REAL	Power factor
AverageVoltage	INT	Average voltage (V)
L3L1Voltage	INT	Voltage between L3-L1 (V)
L1L2Voltage	INT	Voltage between L1-L2 (V)
L2L3Voltage	INT	Voltage between L2-L3 (V)
VoltageImbalance	INT	Phase voltage imbalance (%)
ActivePower	INT	Real power
ReactivePower	INT	Reactive Power (kVAR)
Reserved	INT	Reserve

11.4 Inputs/Outputs

Overview

This section describes the Inputs/Outputs of the MBTESYSTDev FBD.

What's in this Section?

This section contains the following topics:

Generic Inputs/Outputs Description

Overview

This table is the same for all TesysT Profiles.

Inputs/Outputs Description Table

This table describes the inputs/outputs of the TesysT Profile FBD:

Name	Type	Description
TESYST_ST	TESYST_ST_DDT	TESYST_ST *(see page 134)* is a device data structure that holds the minimum information required for performing control and monitoring. The information is used by the operator screen and usable from HMI or SCADA.
TESYST_CFG	TESYST_CFG_DDT	TESYST_CFG *(see page 136)* is a device data structure that holds the information used by the operator screen. It can be read from HMI or SCADA.

Specific MBTESYSTDev Inputs/Outputs

Inputs/Outputs Description Table

This table describes the WorkMemory array of the MBTESYSTDev FBD:

Name	Type	Description
WorkMemory	ARRAY	Array used for Modbus communications. This variable is meant for use with a Modbus Port that serializes all Modbus requests in an optimum manner.

11.5 Operator Screen

General

Description

The component includes an operator screen that enables interaction with the communication interface blocks.

The operator screen data is used to deal with a monitoring system by means of the following:

- TESYST_ST_DDT and TESYST_CFG_DDT - Inputs/Outputs structures.
- TESYST_MEA and TESYSTMEAEV40 depending on the component - Output variables.

11.6 Error Management

Detected Error Code

Description

The detected error codes the device can return are read from the FailCode output variable.

The detected error codes are the same as for the TesysUCDev Error Management *(see page 139)*.

11.7 SCoD Representation and Parameter Description

SCoD Representation and Parameter Description

Overview

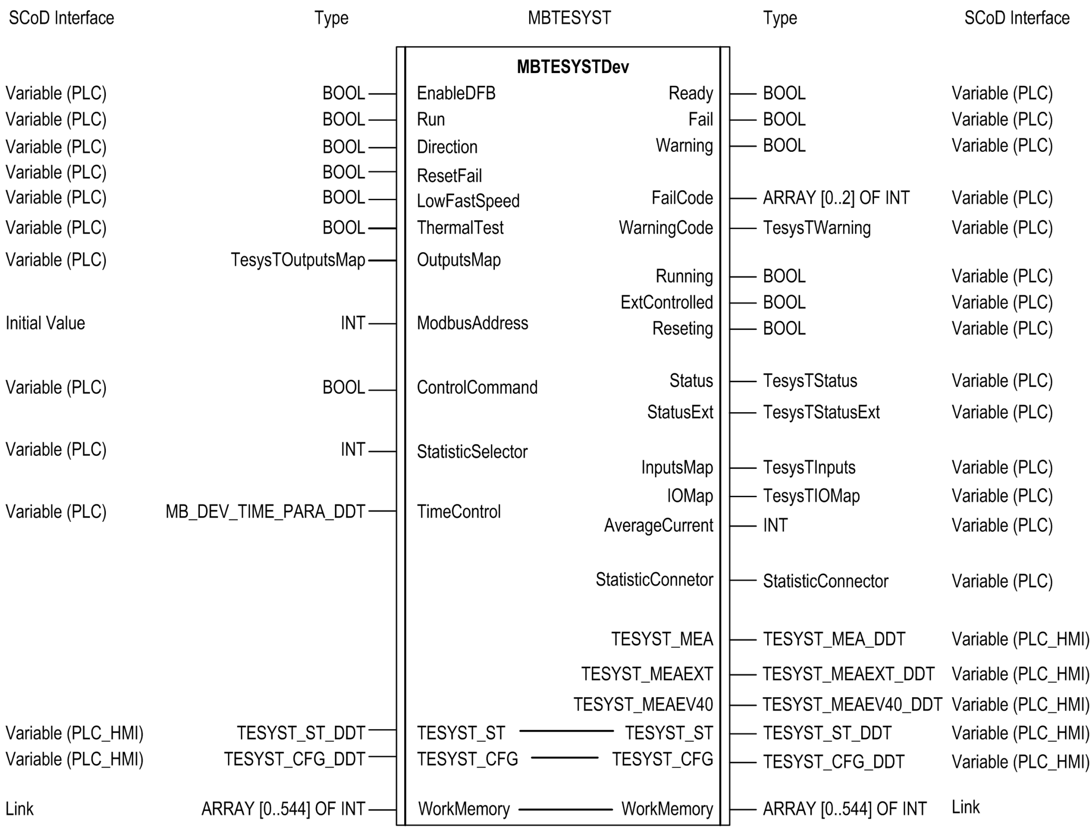
The usage of the DFB pins in the SCoD editor is explained in the input/output diagram below.

The variables defined in the SCoD editor have in principle the same name as the pin (except that an underscore in the pin name is skipped for the variable name).

If the variables defined in the SCoD have the property 'Initial Value' then the values of the instantiated SCoD variables can be modified manually in Unity Pro before PLC restart.

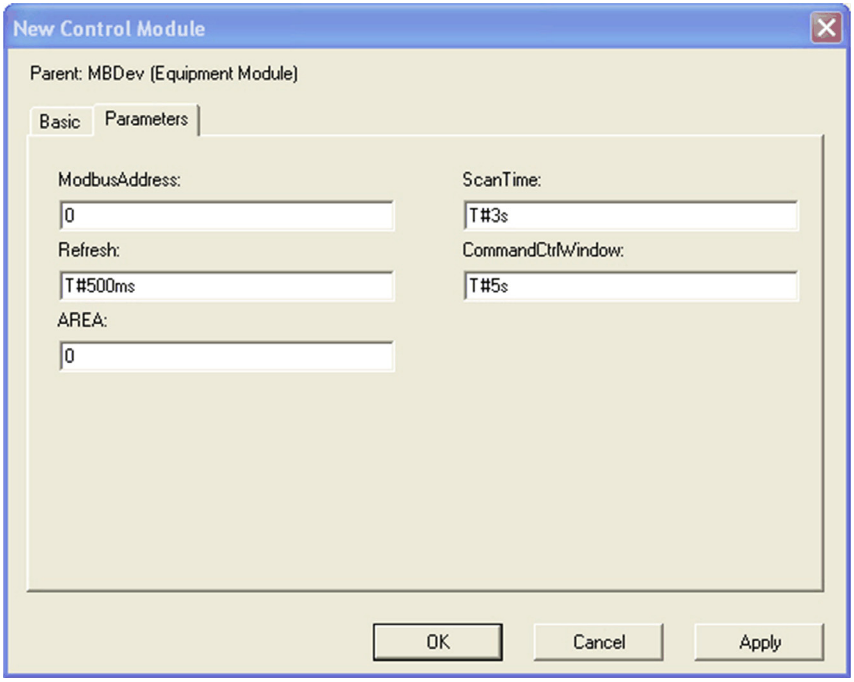
SCoD Representation

This figure represents the MBTESYST SCoD:



Parameter Tab

This figure shows the MBTESYST parameter tab:



The following table describes the properties:

Property	Type	Description
ScanTime	TIME	Minimum time to maintain detected error signals
CommandControlWindow	TIME	The time waited before executing orders
AREA	BYTE	Area to which the object that you wish to instantiate belongs. Enables to control access to the functions provided by this object.

11.8 HMI Representation

MBTESYST- Representation in the HMI

Modbus TeSys T (Genie)

The Modbus TeSys T block is represented in the HMI by the following genie.

Genie Elements

Symbol	Meaning
	Opens the Modbus TeSys T popup. Displays the device name, that is, TeSys T and average current value.
	The instance name of the Modbus TeSys T block is only visible by selecting the **Hide**/**Unhide** button in the template.
	When there is an unacknowledged note, the ! symbol is displayed in the genie.
	When there is an active and unacknowledged alarm or communication failure, then outer rectangle flashes in red.
	When there is an active and unacknowledged alarm or communication failure, then outer rectangle is steady red.

Popup

The Modbus TeSys T block popup has the following tabs:
- **Operator** tab
- **Analog** tab
- **Digital** tab
- **Information** tab

Operator Tab

For detailed description, refer to Operator Tab *(see page 144)*.

Analog Tab

For detailed description, refer to Analog Tab *(see page 146)*.

Digital Tab

For detailed description, refer to Digital Tab *(see page 147)*.

Information Tab

For description about **Information** tab, refer to Commonly Used Tabs *(see page 32)*.

ATV7161 - Ethernet ATV61, ATV71 (EATV61 and EATV71 SCoDs)

12

Overview

This chapter describes the DFB for the control blocks for variable speed drives on an Ethernet network, represented by EATV61 and EATV71 SCoDs.

⚠ WARNING

MISAPPLICATION OF MODULES

The modules in this section do not reflect any specific installation. Before adopting these modules for use in a specific application, the engineer must:

- conduct a safety analysis for the application and equipment installed
- verify that all modules are appropriate for the equipment or function in the installation
- supply appropriate parameters, particularly for limits
- check that all sensors and actuators are compatible with the modules selected
- thoroughly test all functions during verification and commissioning
- provide independent paths for critical control functions (emergency stop, over-limit conditions etc.) according to the safety analysis and applicable codes and regulations.

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

What's in this Chapter?

This chapter contains the following sections:

12.1 Description

Overview

This section describes the ATV7161Dev DFB.

What's in this Section?

This section contains the following topics:

Functional Description

General

The ATV7161Dev profile is the control block for the ATV61 and ATV71 on Ethernet communication.

The ATV7161Dev block is implemented as listed below:

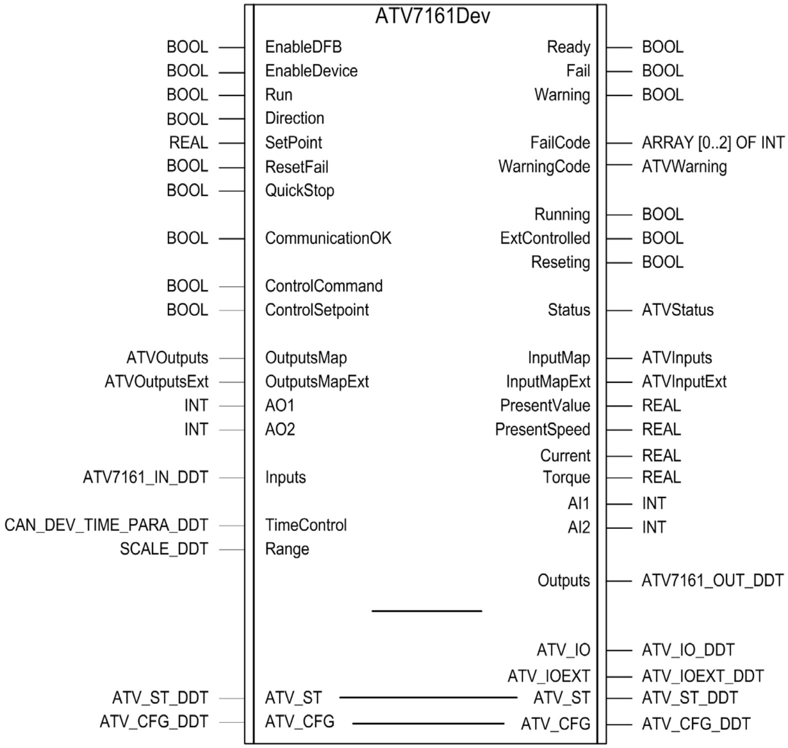
Software	Implemented as
Unity Pro	DFB ATV7161Dev
Vijeo Citect	DPC_Devices
UAG	Depending on the speed drive: • SCoD EATV61 • SCoD EATV71

The main function blocks are described in the ATV Profile Functional Description *(see page 29)*.

FBD Representation

Representation in FBD

The functional module that will be used in the program has the following aspect (at the FBD block level) after PLC generation in UNITY Pro:



12.2 Inputs

Overview

This section describes the inputs of the ATV7161Dev FBD.

What's in this Section?

This section contains the following topics:

Generic Inputs Description

Overview

This table is the same for all ATV Profiles.

Inputs Description Table

This table describes the generic inputs of the ATV Profile FBD:

Input	Type	Description
EnableDFB	BOOL	This input enables the normal execution of the control block. EnableDFB = 0. The entire DFB is restarted (statuses, output values, counters, etc. are lost) and all output values are set to 0 or FALSE. EnableDFB = 1. This input enables communications with devices for their operation. Public variable values are loaded during the first enabling cycle.
EnableDevice	BOOL	This input variable is true only if EnableDFB is true. To control the speed of the drive, it must be enabled.
Run	BOOL	This input makes the motor run in the direction selected with the Direction variable.
Direction	BOOL	This input activates the direction of motor rotation: **1 =** forward **0 =** reverse **NOTE:** Setpoint assumes positive values. Changes in the direction of rotation cannot be made with Setpoint sign.
Setpoint	REAL	This is the speed set-point *(see page 54)* requested from the speed of the drive. It can only have positive, real values.
ResetFail	BOOL	Existing detected errors are reset on the rising edge of this signal. Both communications and detected device errors can be reset. If the detected error is a device error, it sends a reset command to the device if ControlCommand = TRUE. This figure shows the input ResetFail: Fail AND (NOT Reseting)──| |ResetFail
QuickStop	BOOL	This input enables a quick stops the drive speed. This variable is state-based. If there is a QuickStop, the Run bit has to be reset to resume operation.

Input	Type	Description
ControlCommand	BOOL	This input indicates if the starter is controlled from an external source. If ControlCommand=TRUE the device is controlled from an external source. The generation of follow-up alarms is disabled. If ControlCommand=FALSE, the block only performs read operations on the device status, the transmission of device control or reset commands is not possible. **NOTE:** This input does not configure the starter; it only defines the DFB operating mode.
ControlSetPoint	BOOL	This input indicates if the speed of the drive is controlled from an external source. (e.g., from the console or a potentiometer)

⚠ WARNING

UNEXPECTED MOTOR START

Reset the Run variable before resuming operation.

The Run input must be inactive while the device is being enabled or while a detected error is being reset.

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

NOTICE

UNEXPECTED BEHAVIOUR

The ControlCommand input value must coincide with the speed driver configuration.

Failure to follow these instructions can result in equipment damage.

ATV7161Dev Specific Inputs

Inputs Description Table

This table describes the inputs of the ATV7161Dev FBD:

Input	Type	Description
CommunicationOK	BOOL	A bit is used to indicate whether or not the node is present on the bus.
AO1	INT	Controls the value of the AO1 analog output. This value can be found on all devices on CANopen and Ethernet networks. The speed driver analog output can only be controlled with this input variable. It cannot be controlled from HMI/SCADA.
AO2	INT	This input controls the value of the AO2 analog output. This value can be found on ATV71 and ATV61 on CANopen and Ethernet networks. The speed driver analog output can only be controlled with this input variable. It cannot be controlled from HMI/SCADA.
OuputsMap	ATVOutputs	TheOutputsMap *(see page 283)* holds a data structure used to control the speed of the drive outputs. The information is available on CANopen and Ethernet. The speed of the drive outputs can only be controlled with this input variable.(They cannot be controlled from HMI/SCADA).
OuputsMapExt	ATVOutputsExt	The OutputsMapExt *(see page 284)* holds a data structure used to control the speed driver outputs for ATV71 and ATV61 extended cards on CANopen and Ethernet networks. The speed of the drive outputs can only be controlled with this input variable.(They cannot be controlled from HMI/SCADA).
Inputs	ATV7161_ IN_ DDT	The Inputs *(see page 49)* holds a data array structure that contains data obtained from the device. The speed of the drive can be controlled with this input variable. The input is reserved for the DFB and cannot be used directly. Allocate the structure (%MWx) for the correct operation of the control block.

Input	Type	Description
TimeControl	CAN_DEV_TIM_ PARA_DDT	TimeControl *(see page 51)* is a parameter for device control
Range	SCALE_DDT	Range *(see page 53)* definition for inverter (raw and engineering range)

Input Detail - OutputsMap

Description

This function is a data structure (ATVOutputs) used to control the speed of the drive outputs. The information is available on CANopen and Ethernet.

The speed of the drive outputs can only be controlled with this input variable.

They cannot be controlled from the HMI or SCADA.

ATVOutputs Structure Description

This table describes the ATVOutputs structure:

Name	Type	Description
R1	BOOL	Controls the state of the R1 relay output
R2	BOOL	Controls the state of the R2 relay output

Input Detail - OutputsMapExt

Description

This function is a data structure (ATVOutputsExt) used to control the speed driver outputs for ATV71 and ATV61 extended cards on CANopen and Ethernet networks.

The speed drivers outputs can only be controlled with this input variable.

They cannot be controlled from HMI or SCADA.

ATVOutputsExt Structure Description

This table describes the ATVOutputsExt structure:

Name	Type	Description
R3	BOOL	Controls the state of the R3 relay output
R4	BOOL	Controls the state of the R4 relay output
LO1	BOOL	Controls the state of digital output LO1
LO2	BOOL	Controls the state of digital output LO2
LO3	BOOL	Controls the state of digital output LO3
LO4	BOOL	Controls the state of digital output LO4

12.3 Outputs

Overview

This section describes the outputs of ATV7161Dev FBD.

What's in this Section?

This section contains the following topics:

Generic Outputs Description

Overview

This table is the same for all ATV Profiles.

Outputs Description Table

This table describes the outputs of the ATVProfile FBD:

Output	Type	Description
Ready	BOOL	This output indicates if the device is enabled and free of detected errors. The device is ready to carry out or is carrying out a run or stop command. If QuickStop is enabled, this output is reset to 0. This variable is TRUE as long as there is no detected communications error and the device is in status 5 of the DSP402.
Fail	BOOL	This output indicates a detected error in the control block (parameter configuration, cannot be reset), detected error in the device, or a detected communications error. To reset the detected error, the ResetFail input has to be activated. The last detected error code is in FailCode.
Warning	BOOL	This output indicates if an alarm has been activated for the device. It does not affect the blocks operation and does not need to be reset. The signal remains active until the cause for the alarm disappears. The output is set to 0 only when the alarm is acknowledge.
FailCode	ARRAY	This output indicates the detection of an error. When this output is activated, this variable contains the code for the current detected error. If this output is not activated, this variable contains the last detected error. The detected error source is specified by a 3-level structure.
WarningCode	ATVWarning	The WarningCode *(see page 60)* output is a data structure that contains information about the alarm currently on the speed of the drive.
Running	BOOL	This output indicates that the speed of the drive is running and has an output frequency. **NOTE:** If SetPoint is 0 and Run is active, the Running signal is activated.

Output	Type	Description
ExtControlled	BOOL	Indicates if the drive is controlled from an external source (a console, a push-button panel or a monitoring system) to the system (1) or not (0). It provides information for programming.
Resetting	BOOL	The Resetting *(see page 61)* output indicates whether or not a reset is being carried out.
Status	ATVStatus	The Status *(see page 62)* output holds a structure with the information that the module extracts from the drives status variable.
PresentValue	REAL	Indicates the current rotation speed in EU (Engineering Units).
PresentSpeed	REAL	Indicates the current motor rotation speed in rpm.
NOTE: The variables can not be read from the device when there is a detected communications error. The variables keep their last value.		

ATV7161Dev Specific Outputs

Outputs Description Table

This table describes the outputs of the ATV7161Dev control block.

Output	Type	Description
InputsMap	ATVInputs	The OutputsMap *(see page 289)* holds a data structure that contains information on the state of the drive inputs CANopen and Ethernet communications.
InputsMapExt	ATVInputsExt	The OutputMapExt *(see page 290)* holds a data structure that contains information on the state of the drive inputs for ATV71 and ATV61 extended cards on CANopen and Ethernet networks.
ATV_IO	ATV_IO_DDT	The ATV_IO *(see page 291)* device data structure holds information for performing monitoring. The information is used by operator screen, it can be read from HMI or SCADA.
ATV_IOEXT	ATV_IOEXT_DDT	The ATV_IOEXT *(see page 292)* output holds a data structure that contains information for performing monitoring. The information is used by operator screen, it can be read from HMI or SCADA.
Current	REAL	Current motor current in %.
Torque	INT	Current motor torque in 0.1A increments.
AI1	INT	Value of the AI1 analog device input.
AI2	INT	Value of the AI2 analog device input.
OutputData	ATV6171_OUT_DDT	The OutputData *(see page 59)* holds a data array structure that contains data sent to the device. The speed of the drive can be controlled with this output variable. The output is reserved for the DFB; cannot be used directly. Allocate the structure (%MWx) for a correct control blocks operation.

Output Detail - InputsMap

Description

This function is a data structure (ATVInputs) that contains information on the state of the drive inputs CANopen and Ethernet communications.

ATVInputs Structure Description

The below table shows the ATVInputs structure:

Name	Type	Description
LI1	Bool	Shows the state of digital input L1
LI2	Bool	Shows the state of digital input L2
LI3	Bool	Shows the state of digital input L3
LI4	Bool	Shows the state of digital input L4
LI5	Bool	Shows the state of digital input L5
LI6	Bool	Shows the state of digital input L6

Output Detail - InputsMapExt

Description

This function is a data structure (ATVInputsMapExt) that contains information on the state of the speed drivers inputs for ATV71, ATV61 extended cards on CANopen and Ethernet networks.

ATVInputsMapExt Structure Description

This table shows the ATVInputsMapExt structure:

Name	Type	Description
LI7	BOOL	Shows the state of digital input L7
LI8	BOOL	Shows the state of digital input L8
LI9	BOOL	Shows the state of digital input L9
LI10	BOOL	Shows the state of digital input L10
LI11	BOOL	Shows the state of digital input L11
LI12	BOOL	Shows the state of digital input L12
LI13	BOOL	Shows the state of digital input L13
LI14	BOOL	Shows the state of digital input L14

Output Detail - ATV_IO

Description

This device data structure (ATV_IO_DDT) holds information for performing monitoring. The information is used by operator screen and it can be read from HMI or SCADA.

ATV_IO_DDT Structure Description

This table shows the ATV_IO_DDT structure:

Name	Type	Description
OutputsMap	WORD	Data with the drive speed outputs The OutputsMap provides control over device outputs.
AO1	INT	Device control This output indicates the value of the AO1 analog output (value of input variableAO1).
Current	REAL	Current value This output indicates the current in the motor in 0.1 A increments (Value of the current output variable).

OutputsMap Word Description

This table describes the OutputsMap:

Bit	Description
0	R1 output state (OutputsMap.R1)
1	R2 output state (OutputsMap.R2)
2	R3 output state (OutputsMap.R3). For CATV31M
3	R4 output state (OutputsMap.R4). For CATV31M
4	LO1 output state (OutputsMapExt.LO1). For CATV31M
5	LO2 output state (OutputsMapExt.LO2). For CATV31M
6	LO3 output state (OutputsMapExt.LO3). For CATV31M
7	LO4 output state (OutputsMapExt.LO4). For CATV31M

Output Detail - ATV_IOEXT

Description

This device data structure (ATV_IOEXT_DDT) holds information for performing monitoring.

The information is used by operator screen and it can be read from HMI or SCADA.

ATV_IOEXT_DDT Structure Description

This table shows the ATV_IOEXT_DDT structure:

Name	Type	Description
InputsMap	WORD	State of the device digital inputs
AI1	INT	AI1 analog input
AI2	INT	AI2 analog input
AO2	INT	AO2 analog output
Torque	DINT	Current Torque value

InputsMap Word Description

The InputsMap provides the value of the device inputs.

This table describes the InputsMap:

Bit	Description
0	Value of digital input LI1 (InputsMap.LI1)
1	Value of digital input LI2 (InputsMap.LI2)
2	Value of digital input LI3 (InputsMap.LI3)
3	Value of digital input LI4 (InputsMap.LI4)
4	Value of digital input LI5 (InputsMap.LI5)
5	Value of digital input LI6 (InputsMap.LI6)
6	Value of digital input LI7 (InputsMap.LI7)
7	Value of digital input LI8 (InputsMap.LI8)
8	Value of digital input LI9 (InputsMap.LI9)
9	Value of digital input LI10 (InputsMap.LI10)
10	Value of digital input LI11 (InputsMap.LI11)
11	Value of digital input LI12 (InputsMap.LI12)
12	Value of digital input LI13 (InputsMap.LI13)
13	Value of digital input LI14 (InputsMap.LI14)
14	Value of digital input LI15 (InputsMap.LI15)

12.4 Inputs/Outputs

Generic Inputs/Outputs Description

Overview

This table is the same for all ATV Profiles.

Inputs/Outputs Description Table

This table describes the inputs/outputs of the ATV Profile FBD:

Name	Type	Description
ATV_ST	ATV_ST_DDT	The ATV_ST *(see page 67)* device data structure contains the minimum information required for performing control and monitoring. The information is used by the operator screen and can be read from HMI or SCADA.
ATV_CFG	ATV_CFG_DDT	The ATV_CFG *(see page 69)* device data structure contains the information used by the operator screen and can be read/write from HMI or SCADA.

12.5 Operator Screen

General

Description

The component includes an operator screen that allows interaction with the communication interface blocks.

- ATV_ST_DDT and ATV_CFG_DDT - Inputs/Outputs structures.
- ATV_IO_DDT and ATV_IOEXT_DDT - Output structures.

12.6 Error Management

Detected Error Code

Description

The detected error codes the device can return are read from the FailCode output variable.

The detected error codes are the same as for the ATVDev Error Management *(see page 74)*.

12.7 SCoD Representation and Parameter Description

SCoD Representation and Parameter Description

Overview

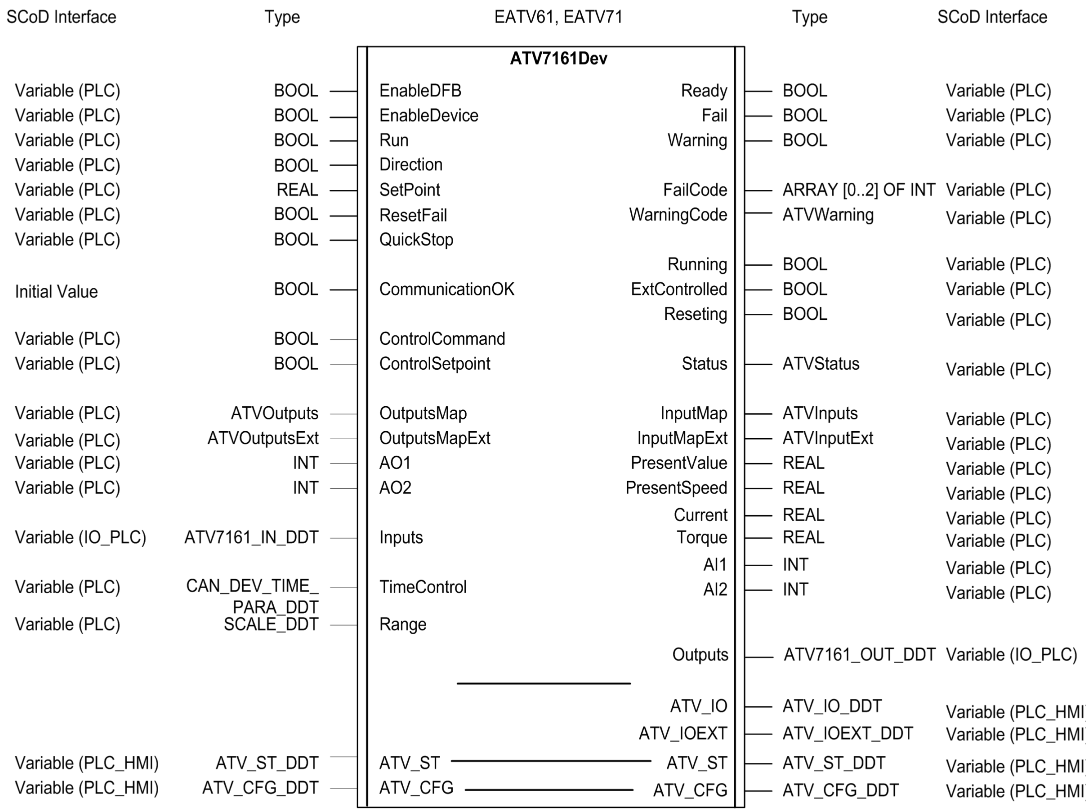
The usage of the DFB pins in the SCoD editor is explained in the input/output diagram below.

The variables defined in the SCoD editor have in principle the same name as the pin (except that an underscore in the pin name is skipped for the variable name).

If the variables defined in the SCoD have the property ’Initial Value’ then the values of the instantiated SCoD variables can be modified manually in Unity Pro before PLC restart.

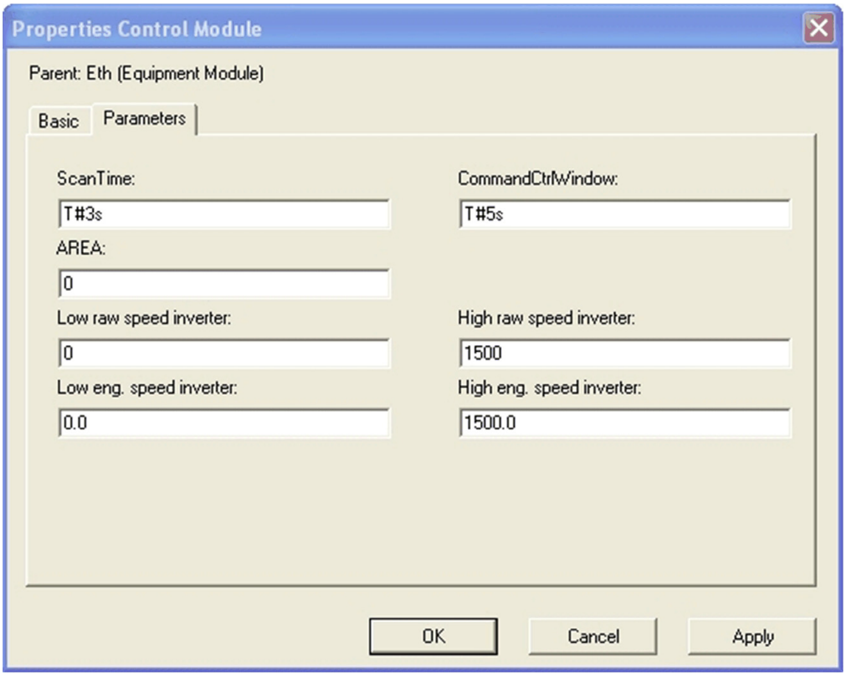
SCoD Representation

This figure represents the EATV61 and EATV71 SCoDs:



Parameter Tab

This figure shows the EATV61 and EATV71 SCoDs parameter tab:



The following table describes the properties:

Property	Type	Description
ScanTime	TIME	Minimum time to maintain detected error signals
CommandControlWindow	TIME	The time waited before executing orders
AREA	BYTE	Area to which the object that you wish to instantiate belongs. Enables to control access to the functions provided by this object.
Low raw Speed Inverter	INT	Low range value measured in raw units
High raw Speed Inverter	INT	High range value measured in raw units
Low eng. Speed Inverter	REAL	Low range value measured in engineering units
High eng. Speed Inverter	REAL	High range value measured in engineering units

12.8 HMI Representation

Ethernet ATV61, ATV71 - Representation in the HMI

Ethernet ATV (Genie)

The Ethernet ATV block is represented in the HMI by the following genie.

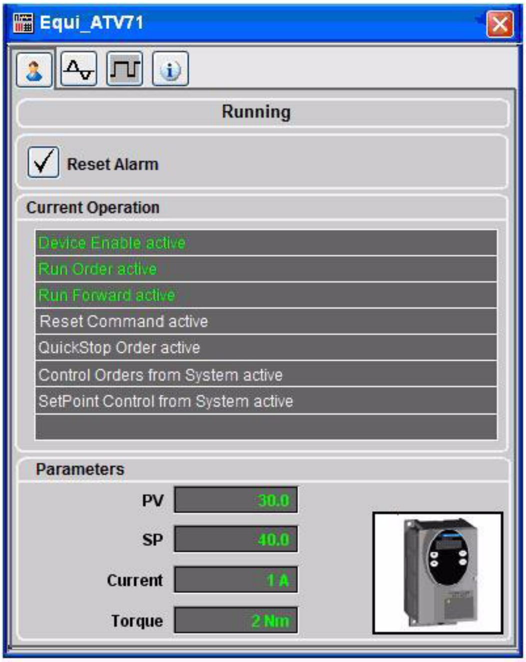
Genie Elements

Symbol	Meaning
	Opens the Ethernet ATV popup. Displays the device name, that is, ATV and PV value.
	The instance name of the Ethernet ATV block is only visible by selecting the **Hide**/**Unhide** button in the template.
	When there is an unacknowledged note, the ! symbol is displayed in the genie.
	When there is an active and unacknowledged alarm or communication failure, then outer rectangle flashes in red.
	When there is an active and unacknowledged alarm or communication failure, then outer rectangle is steady red.

Popup

The Ethernet ATV block has the following tabs:
- **Operator** tab
- **Analog** tab
- **Digital** tab
- **Information** tab

Operator Tab



This tab has 3 sub-sections:
- **Status**
- **Reset Alarm**
- **Current Operation**
- **Parameters**

The sub-section **Status** displays the current status.

The possible displays are listed as follows:
- **Unknown Status**
- **Not Ready**
- **Running**
- **Communication Failure**
- **Failure**
- **Warning**

The following table describes the sub-section **Reset Alarm**:

Item	Description
Reset Alarm	Resets alarm.

The following table describes the sub-section **Current Operation**:

Item	Description
Device Enable active	Displays the text in green color when the operation is active. Displays the text in white color when the operation is not active.
Run Order active	
Run Forward active	
Reset Command active	
QuickStop Order active	
Control Orders from System active	
SetPoint Control from System active	

The following table describes the sub-section **Parameters**:

Item	Description
PV	Displays the present value.
SP	Displays the set point value.
Current	Displays the value of current.
Torque	Displays the value of torque.

The sub-section **Parameters** also shows the picture of `ATV`.

Analog Tab

For detailed description, refer to Analog Tab *(see page 82)*.

Digital Tab

For detailed description, refer to Digital Tab *(see page 83)*.

Information Tab

For description about **Information** tab, refer to Commonly Used Tabs *(see page 32)*.

Communication Components

II

Overview

This part describes the Communication function blocks.

What's in this Part?

This part contains the following chapters:

General Overview of Communication Resources

13

Overview

This Chapter lists the resources that were designed specifically for Network communication or bus based communication.

What's in this Chapter?

This chapter contains the following topics:

Technological Blocks for Communications

General

Following is a list of the components that are described in this document and that have been specifically designed for the automation of Modbus, CANopen, and Ethernet Modbus TCP/IP communication systems.

Single Read/Write Requests

MBClientBs: (Basic Modbus Client). Issues a read or write request for n registers on a Modbus communication bus.
CClient: (CANopen Client). Issues a read or write request for n registers on a CANopen communication bus.
EClient: (Ethernet Client). Issues a read or write request for n registers on an Ethernet communication network.

Multiple Read/Write Requests

MBScanner: (Modbus Scanner). Periodically refreshes read or write requests for n registers issued to a single node on a Modbus communication bus.
CScanner: (CANopen Scanner). Periodically refreshes read or write requests for n registers issued to a single node on a CANopen communication bus.
EScanner: (Ethernet Scanner). Periodically refreshes read or write requests for n registers issued to a single node on an Ethernet communications network.

Gateways Between Different Communication Buses

ASRTPBdg: (Gateway between CANopen and Advantys devices). Issues requests to a device that is connected to an Advantys STB input/output island.
EGtwMB (Modbus – Ethernet Gateway). Serial Modbus-Modbus TCP/IP Gateway (Ethernet). Issues requests via Ethernet to a device connected to a serial gateway (TSXETG100)

Serialization of Requests to the Port

MBPortM: (M340 Modbus Port). Serializes Modbus requests to the controller port.
MBPortP: (Premium Modbus Port). Serializes Modbus requests to the controller physical port.
CPortM: (M340 CANopen Port). Serializes CANopen requests to the controller physical port.
CPortP: (Premium CANopen Port). Serializes CANopen requests to the controller physical port.
EPortM: (M340 Ethernet Port). Serializes Ethernet requests to the controller physical port or communication modules.

EPortP: (Premium Ethernet Port). Serializes Ethernet requests to the controller's physical port or communication modules.

EPortQ: (Quantum Ethernet Port). Serializes Ethernet requests to the controller physical port or communication modules.

Statistical Tools

Statistics: Performs calculations on the transmitted requests. Time between requests, average response time, etc.

Component Library for Communications

General

The technological blocks listed in the previous section, together with other standard al blocks from the Unity libraries, are combined with each other in order to generate DPU Components that automate communications systems on standard SCHNEIDER ELECTRIC buses.

The DPU Components include the following resources for automatically generating Unity projects:

- Variables
- Control logic
- Operator Screens

General Concepts

14

Overview

This Chapter describes the basic concepts implemented by the communications control resources.

What's in this Chapter?

This chapter contains the following topics:

Logical Architecture – Communications

Basics

The communication functions basic operation consists of the various Clients or Scanners storing issued requests in a memory that is managed by the ports.

The Ports are linked to a physical port on the controller and, based on a defined algorithm, manage priorities and waiting times, send the requests to the correct destination, and finally return the corresponding response to the Client or Scanner that generated the request.

Memory Management

The data exchange zone consists of a dynamic structure that adapts to the needs of each program in order to provide appropriate memory management.

The necessary work memory (WorkMemory) is calculated during the first scanning cycle.

NOTE: Clients and Scanners must already be configured and program instances must be executed when the controller first scanning cycle takes place. This will enable the port to carry out a calculation regarding work memory and enable the correct order of execution to be maintained: First, the clients/scanners; then the Port that serializes the requests.

CANopen

Systems with communications to devices on an Advantys STB island, the CANopen-Advantys device gateway functions as a port in relation to the Client or Scanner requests on CANopen.

Gateway

A gateway is a hardware device which is used to connect Modbus Ethernet networks with Serial Modbus networks. In order to use this device, a DFB that converts requests from serial clients to Ethernet client requests must be used.

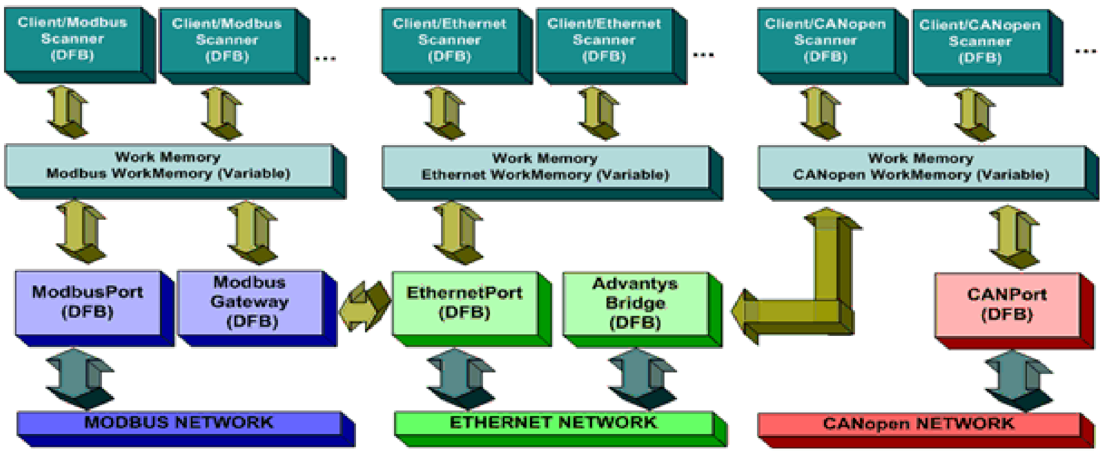
Once these requests are received by the gateway, they are converted into serial requests again and are sent to the corresponding devices. The main difference between serial requests and Ethernet requests consists of device addressing.

Internally, a gateway is made up of a modified serial Modbus port and an Ethernet client. Instead of calling the serial communication functions (which is what a normal port does), the serial Modbus port enters the request data into the Ethernet client, waits for its response, and returns the data which has been returned like a normal port would.

Communication Process Diagram

Communication Diagram

The following diagram represents the communication process:



What is a Communication Scanner?

General

A Communication Scanner is a function block that uses an internal Client to issue several sequential requests to different memory positions in the remote device. It makes it possible to program up to 10 read/write requests that will be issued sequentially; activated either cyclically or with one single cycle.

By using a Communication Scanner, it is possible to access a set of memory addresses (consecutive or non-consecutive) in remote devices that cannot be accessed normally with components generated with the DPU solution, just like with a client, but by programming several accesses to different memory positions in the device sequentially and individually, without the use of multiple clients.

Access to this data is explicit, i.e., the user must program access to this data. This is different from the implicit access used in other communications, as with CANopen PDOs or Ethernet IO Scanning, in which access must be configured but not programmed.

The following names will be used in this manual to refer to the scanners:
ModBusScanner: Modbus Scanner (Modbus Messaging)
CANScanner: CANopen Client (CANopen messaging. SDO)
ModBusEthernetScanner: Ethernet Scanner (Ethernet Messaging)

Communication Scanner Functions

The module main functions are described below:

Function	Description
Cyclic reading/writing	Enables reading or writing, with cycling timing.
Multiple register reading/writing	Enables several consecutive registers to be read (only available with ModBusEthernetClient and ModBusClientBasic).
Individual activation of scanner requests	Individually select which requests should be activated (out of the ten available ones).
Error management	Monitors transaction errors, identifying them on three levels in order to determine the source of the error.
Priorities	Define priorities for systems with multiple client accesses.
Statistics	Obtains the transactions OK/NOK status, as well as their access times.

Communication Scanner Blocks

The communications scanner blocks are implemented as listed below:

Software	Implemented as
Unity Pro	Depending on which communication technology is used. There are 3 different blocks: **CANOpen:** CanScannerCTL *(see page 318)* **ModBus** ModBusScannerCTL *(see page 354)* **Ethernet** ModBusEthernetScannerCTL *(see page 390)*
UAG	Depending on which technology is used, there are different available SCoDs: **CANOpen:** SCoD CScanner *(see page 318)* **ModBus:** SCoD MBScanner *(see page 354)* **Ethernet:** SCoD EScanner *(see page 390)*

What is a Communication Client?

General

A communications client enables to access the remote device data.

For example, this enables to read/write from/to a variable speed drive parameter if this parameter is not available on the respective speed driver control block.

Access to this data is explicit and the access to this data must be programmed. This is different from the implicit access used in other communications, as with CANopen PDOs or Ethernet IO Scanning, in which access must be configured but not programmed.

The following names will be used in this manual to refer to the clients:
ModBusClientBasic: Modbus Client (Modbus Messaging)
CANClient: CANopen Client (CANopen messaging. SDO).
ModBusEthernetClient: Ethernet Client (Ethernet Messaging)

Communication Clients

The communications clients block is implemented as listed below:

Software	Implemented as
Unity Pro	Depending on which communication technology is used. There are 2 different blocks: **ModBus:** ModBusClientBsCTL *(see page 361)* **CANClient:** CANClientCTL *(see page 328)* **Ethernet:** ModBusEthernetClientCTL *(see page 398)*
UAG	Depending on which communication technology is used, different SCoD should be used: **ModBusClientBasic:** SCoD MBClientBs *(see page 361)* **CANClient:** SCoD CClient *(see page 328)* **ModBusEthernetClient:** SCoD EClient *(see page 398)*

What is a Port Function?

Function Description

A Port DFB is a function which is capable of serializing and managing requests sent to a physical medium working on CANopen, serial Modbus, or Modbus TCP/IP.

A client is a Modbus communications (ATV, PM, ATS, etc.) device DFB or a generic read/write DFB to a device on the physical medium (Clients, Scanners).

The basic operation consists of:
1. Storing the various Clients requests in a memory that is managed by the Port (Modbus, CANopen, or Ethernet).
2. Taking out the requests from the queue, following on a defined algorithm which manages priorities and waiting times
3. Sending them to the appropriate destination.
4. Returning the destination response to the client that generated the request.

Requirements

Using a port DFB for messaging requests is required.

A minimum of one port per controller hardware is necessary in order to be able to effect communications of this type.

NOTE: In a serial Modbus (RTU) project, it is necessary to use at least one port DFB, since the communications on a Modbus take place through messaging by definition.

If the Device library is used with CANopen or Modbus TCP/IP networks, port DFBs are not required for their control, since these devices exchange cyclic data by means of IO Scanning technology on Modbus TCP/IP and PDOs on CANopen.

Information that is not included in the Device library blocks must be requested through a cyclic exchange, i.e. through requests sent to a port DFB.

Chosing the Port

Select the port SCoD fitting to the PLC and the communication:

Communication	**M340**	**Premium**	**Quantum**
Modbus RTU	ModbusPortM *(see page 368)*	ModbusPortP *(see page 375)*	-.
CANOpen	CanPortM *(see page 346)*	CanPortP *(see page 337)*	-
Modbus TCP/IP	ModBusEthernetPortM *(see page 414)*	ModBusEthernetPortP *(see page 421)*	ModBusEthernetPortQ *(see page 428)*

CANopen Communication

15

Overview

This chapter describes the CANopen communication SCoDs and their representation in Unity Pro.

⚠ WARNING
MISAPPLICATION OF MODULES The modules in this section do not reflect any specific installation. Before adopting these modules for use in a specific application, the engineer must: • conduct a safety analysis for the application and equipment installed • verify that all modules are appropriate for the equipment or function in the installation • supply appropriate parameters, particularly for limits • check that all sensors and actuators are compatible with the modules selected • thoroughly test all functions during verification and commissioning • provide independent paths for critical control functions (emergency stop, over-limit conditions etc.) according to the safety analysis and applicable codes and regulations. **Failure to follow these instructions can result in death, serious injury, or equipment damage.**

What's in this Chapter?

This chapter contains the following sections:

15.1 CScanner: CANopen Scanner

Overview

This section describes the CScanner SCoD, Communication Scanner SCoD using CANopen technology (CANopen messaging, SDO).

What's in this Section?

This section contains the following topics:

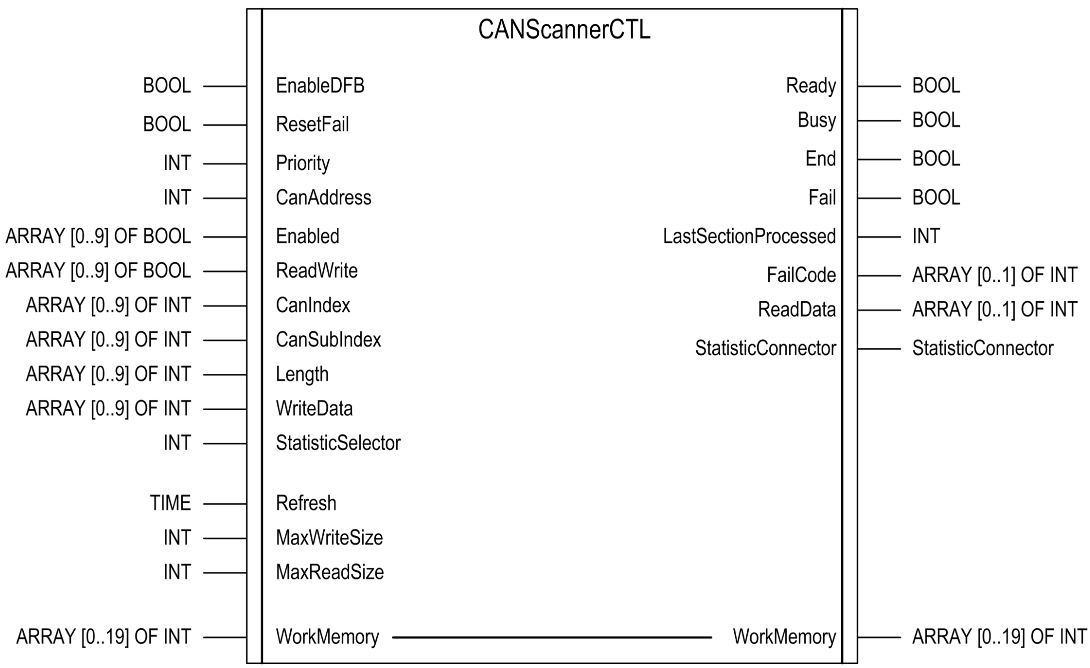
Description

Function Description

The CANScannerCTL is a Communications Scanner *(see page 312)* that uses an internal Client to issue several sequential requests to different memory positions in the remote device.

Representation in FBD

The CScanner SCoD is represented with the CanScanner DFB.



Inputs

General

Indexes (Memory Address, CanIndex, CanSubIndex) are usually expressed in decimal values in device manuals. In order to express the index as a hexadecimal value, use the string “16#” before the index, i.e., 16#index.

Input Parameter Description

Parameter	Type	Description
EnableDFB	BOOL	Activating this input enables communications. If the function is not enabled, initialize the scanner.
ResetFail	BOOL	Resets a communications or parameter error.
Priority	INT	Command priority. The lower the value, the higher the priority, i.e., 0 is the maximum priority. Any value is valid with a conditions: If integrating a Modbus Client with any element from the DPL Devices Components is requested, the priorities that are used by the latter must be taken into account in order not to interfere with these elements: **1** Control commands. They have maximum priority. **2** Command confirmation. The read operations carried out to confirm that the commands have been executed. Medium priority. **3** Data reading. If these priority levels are not observed, it is possible that the Client requests will never be executed or that they will not allow other requests to be transmitted. If integrating a Modbus Client with other DPL Devices Components elements is not requested, any priority levels can be used In other cases, it is recommended to use priority levels greater than or equal to 4.
CanAddress	INT	Address of the CANopen node that the client should access. The range of possible values goes from 1 to 127.
Enabled	ARRAY of INT	Bit array. These bits enable the scanner lines individually. When set to true, the scanner element will be read or written to cyclically, based on the associated parameters.

Parameter	Type	Description
ReadWrite	ARRAY of INT	Bit array. These bits select, for each scanner line, whether the operation that should be carried out is a: **when =1** read operation **when =0** write operation **NOTE:** Modifying the read/write operations for each request misaligns the WriteData and/or ReadData arrays.
CanIndex	ARRAY of INT	Each position in the array holds the number of the index for the object with which the operation should be carried out.
CanSubIndex	ARRAY of INT	Each position in the array holds the number of the sub-index for the object with which the operation should be carried out.
Length	ARRAY of INT	Integer array. The integers hold the length of the data to be read/written in each scanner line. For Modbus (ModbusScanner, ModbusEthernetScanner and CANScanner.) or Ethernet (ModBusEthernetClient) communications, the length can be in bits or Words, depending on the BitWord variable. The maximum length for words is 120. If communications are carried out via CANopen (CANClient), this size is specified in bytes, and the size of any CAN request can be 1, 2, or 4 bytes. **NOTE:** Modifying the lengths of the write operations for each request will misalign the WriteData array.
WriteData	ARRAY of INT	Integer array. The integers hold the data to be written in each scanner line. This table holds the data to be written for the read requests. The size of the variables used for each request is marked by the Length variable. As a result, each write request reserves its zone in this array.
StatisticSelector	INT	Variable used to obtain network statistics (requests carried out, time beten requests, etc.). This data provides information for using the StatisticsConnector with the Communications library “Statistics Module. Depending on the value: **1** Read Statistics, Scanner **2** Read Statistics, Client **3** Write Statistics, Scanner **4** Write Statistics, Client
Refresh	TIME	Time that the scanner will wait before it repeats an operation cycle.

Parameter	Type	Description
MaxReadSize	INT	Configuration parameter that indicates how many words will be read in the request issued by this client. If no value is specified, the maximum possible value is used by default (125 words for Ethernet/Modbus; 4 for CANopen). This parameter is used for calculating and managing the work memory area (WorkMemory).
MaxWriteSize	INT	Configuration parameter that indicates how many words will be written in the largest request issued by this client. If no value is specified, the maximum possible value is used by default (125 words for Ethernet/Modbus; 4 for CANopen). This parameter is used for calculating and managing the work memory area (WorkMemory).

Outputs

Output Parameter Description

Parameter	Type	Description
Ready	BOOL	Indicates that the scanner is ready for operation.
Busy	BOOL	Indicates that the scanner is busy waiting for the response to a request.
End	INT	Indicates that the operation in progress has ended.
Fail	BOOL	Shows an error in the function during the course of the current request. If communications with the slave is in error state, it stops requests until a ResetFail is carried out.
FailCode	INT[3]	Indicates the last error that took place, according to three error levels.
LastSectionProcessed	INT	If End is “true,” this output shows the index of the scanner operation that has ended. **NOTE:** ReadData holds the read data for LastSectionProcessed. Because of this, this data is updated every time a new read operation is processed. This data must be unloaded from this array by the program in case it is needed after the next request.
StatisticConnector	StatisticConnector	Information data used to obtain network statistics (requests carried out, time between requests, etc.). This structure has been created for its use together with the Communications library Statistics module.

This tables describes the StatisticConnector type:

Name	Type	Description
Start	BOOL	Operation has started
EndOk	BOOL	Operation ended correctly
EndNOK	BOOL	Operation ended with an error
PartialTime	DINT	Partial time

Inputs/Outputs

Inputs/Outputs Parameter Description

Parameter	Type	Description
WorkMemory	Array	Work memory. Array used for communications. This variable is used in a CanPort / ModBusPort / ModBusEthernetPort, which has the objective of serializing client requests in an optimum manner.

Representation as SCoD in UAG

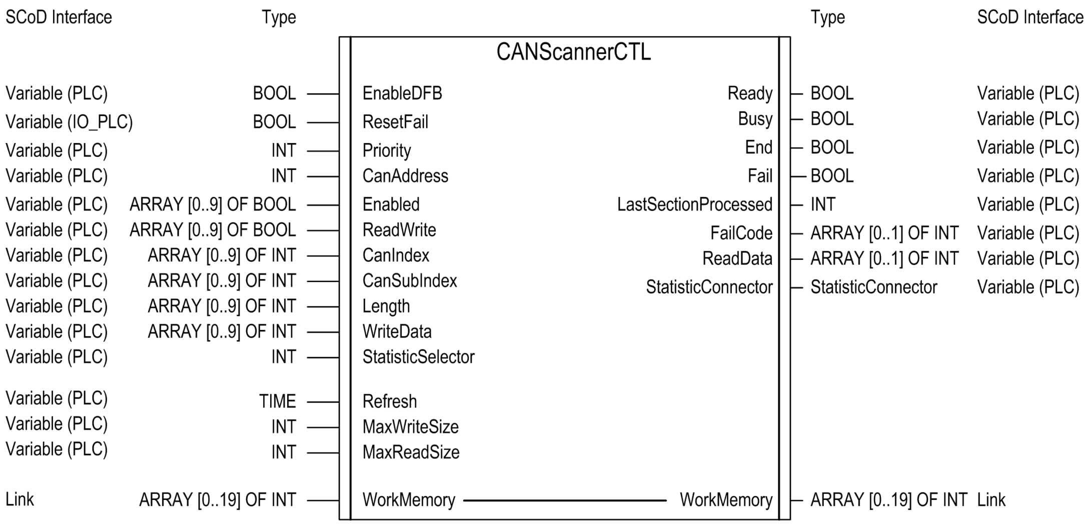
Overview

Use of the DFB pins in the SCoD editor is explained in the input/output diagram below.

The variables defined in the SCoD editor have in principle the same name as the pin (exception is that an underscore in the pin name will be skipped for the variable name).

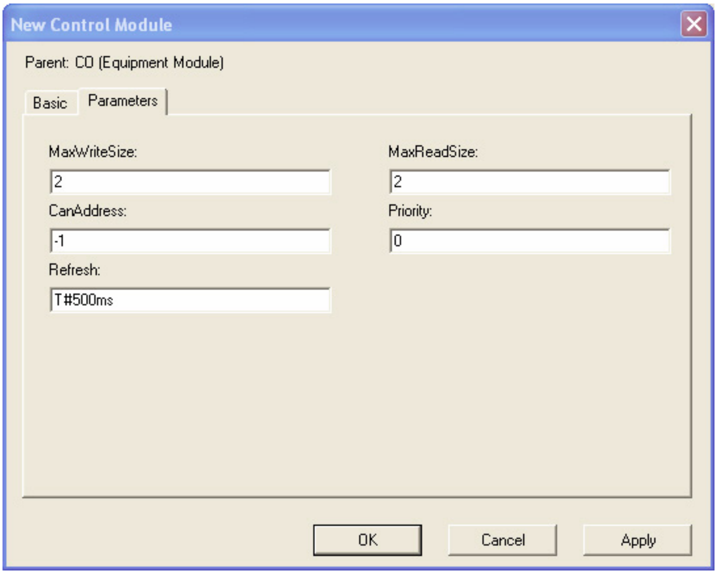
If the variables defined in the SCoD have the property 'Initial Value' then the values of the instantiated SCoD variables can be modified manually in Unity Pro before the PLC restart.

CANScannerCTL Input Output Diagram



Parameter Tab

The figure below shows the **Parameter tab** of the **Properties Control Module** window.



The following table describes the properties:

Property	Type	Description
CANAddress	INT	Address of the CANopen node that the client should access. The range of possible values goes from 1 to 127.
MaxReadSize	INT	Configuration parameter that indicates how many words will be read in the request issued by this client. This parameter is used for calculating and managing the work memory area (WorkMemory). **NOTE:** If several requests are made with the same client, use the maximum length of the read requests to be issued as the value of the parameter.
MaxWriteSize	INT	Configuration parameter that indicates how many words will be read in the request issued by this client. This parameter is used for calculating and managing the work memory area (WorkMemory). **NOTE:** If several requests are made with the same client, use the maximum length of the read requests to be issued as the value of the parameter.

Property	Type	Description
Priority	INT	Command priority. The lower the value, the higher the priority. (0 is the maximum priority)
Refresh	TIME	Time that the scanner will wait before it repeats an operation cycle.

15.2 CClient: CANOpen Client

Overview

This section describes the CClient SCoD, the Client SCoD with CANopen communication technology.

The module main functions are described below:

Function	Description
Read/Write	Enables select whether the parameter has read or write access.
Error management	Monitors transaction errors, identifying them on three levels in order to determine the source of the error.
Priorities	Enables to define priorities for systems with multiple client accesses.
Statistics	Obtains the transactions OK/NOK status, as well as their access times.

What's in this Section?

This section contains the following topics:

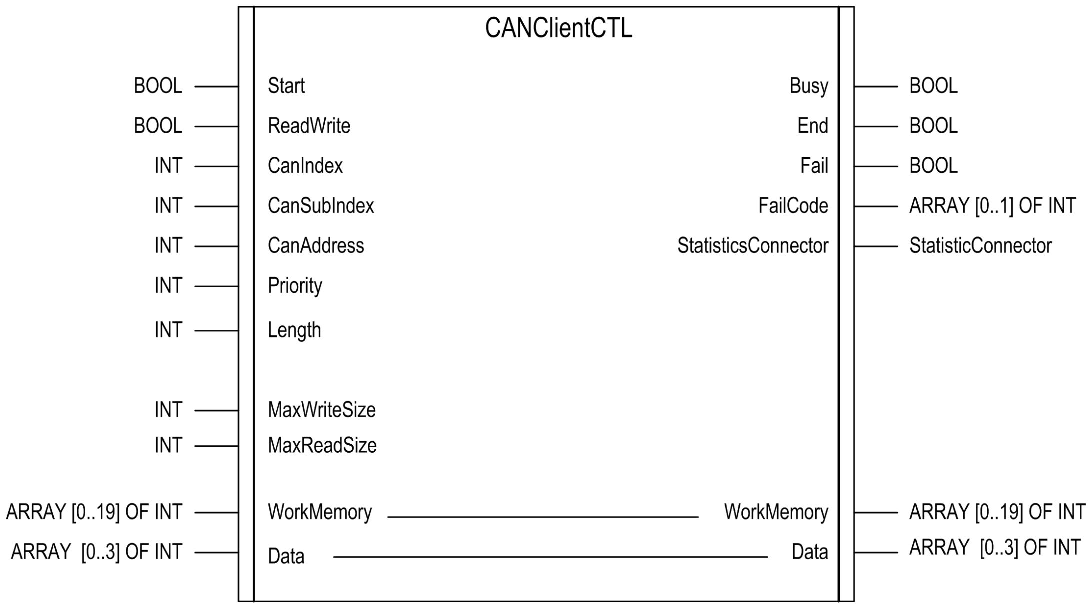
Description

Function Description

This Communication Client *(see page 314)* enables device data to be written or read to/from CANopen technologies.

Representation in FBD

The CClient SCoD is represented with the CanClientCTL DFB in Unity Pro:



Inputs

General

Indexes (Memory Address, CanIndex, CanSubIndex) are usually expressed in decimal values in device manuals. In order to express the index as a hexadecimal value, use the string “16#” before the index, i.e., 16#index.

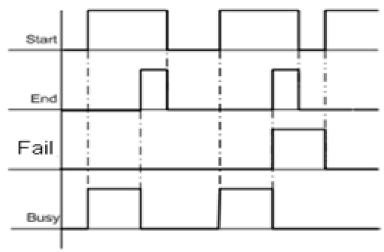
Input Parameter Description

Parameter	Type	Description
Start	BOOL	A rising edge on this signal indicates, to the client, that the data on the inputs is valid and that a request to a server should be issued. The rising edge also resets errors. A falling edge on this signal triggers an ACK of the end-of-operation notification, and the client is ready for the next cycle. When the signal is activated, the parameters are copied to the function, so modifying them has no effect.
ReadWrite	BOOL	Selects whether the operation to be carried out is a read operation (when set to false) or a write operation (when set to true). Selects whether the operation to be carried out is a: **when =1** read operation **when =0** write operation
CanIndex	INT	Number of the index for the object with which the operation should be carried out. It can be expressed as a decimal or hexadecimal value.
CanSubIndex	BOOL	Indicates the number of the sub-index for the object with which the operation should be carried out. It can be expressed as a decimal or hexadecimal value.
CanAddress	INT	Address of the CANopen node that the client should access. The range of possible values goes from 1 to 127.

Parameter	Type	Description
Priority	INT	Command priority. The lower the value, the higher the priority, i.e., 0 is the maximum priority. Any value is valid with a conditions: If integrating a Modbus Client with any element from the DPL Device Component is requested, the priorities that are used by the latter must be taken into account in order not to interfere with these elements: **1** Control commands. They have maximum priority. **2** Command confirmation. The read operations carried out to confirm that the commands have been executed. Medium priority. **3** Data reading. If these priority levels are not observed, it is possible that the Client requests will never be executed or that they will not allow other requests to be transmitted. If integrating a Modbus Client with other DPL Device Component elements is not requested, any priority levels can be used In other cases, it is recommended to use priority levels greater than or equal to 4.
Length	INT	Length of the data involved in the request that has been carried out. The maximum length for words is 120. If communications are carried out via CANopen (CANClient), this size is specified in bytes, and the size of any CAN request can be 1, 2, or 4 bytes.
MaxReadSize	INT	Configuration parameter that indicates how many words will be read in the request issued by this client. If no value is specified, the maximum possible value is used by default (=4). This parameter is used for calculating and managing the work memory area (WorkMemory).
MaxWriteSize	INT	Configuration parameter that indicates how many words will be written in the largest request issued by this client. If no value is specified, the maximum possible value is used by default (=4). This parameter is used for calculating and managing the work memory area (WorkMemory).

Outputs

Output Parameter Description

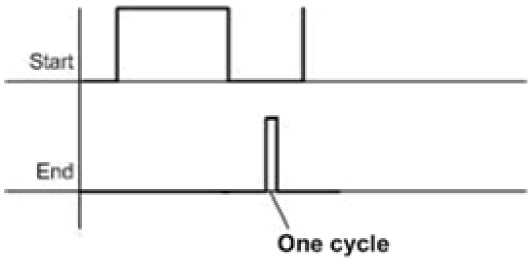
Parameter	Type	Description
Busy	BOOL	Activated while a request is taking place. Indicates that the client is busy and that no new requests can be made because of this.
End	INT	Activated when a request cycle ends. Indicates that the operation has ended (with or without error). This signal is acknowledged (ACK) by setting the Start signal to Low.
Fail	BOOL	Activated when there is an error in the request transmission. In order to reset the error, activate the client again by setting the Start signal to High. Time diagram: 
FailCode	INT[3]	Indicates the last error that took place, according to three error levels.
StatisticsConnector	StatisticsConnector	Information data used to obtain network statistics (requests carried out, time between requests, etc.). This structure has been created for its use together with the Communications library Statistics module.

This table describes the StatisticConnector type:

Name	Type	Description
Start	BOOL	Operation has started
EndOk	BOOL	Operation ended correctly
EndNOK	BOOL	Operation ended with an error
PartialTime	DINT	Partial time

NOTE:

If, for any reason, the Start signal is FALSE when the End signal is activated, only an edge for one cycle of the End signal will be produced. This can result in the program (depending on how it is designed) not detecting the End signal, which is why this must not be allowed to happen for any reason what so ever. The corresponding behavior would be as follows:



Inputs/Outputs

Inputs/Outputs Parameter Description

Parameter	Type	Description
WorkMemory	Array	Work memory. Array used for communications. This variable is used in a CanPort / ModBusPort / ModBusEthernetPort, which has the objective of serializing client requests in an optimum manner.
Data	Array	Holds the write data or the read data, depending on the ReadWrite input parameter.

Representation as SCoD in UAG

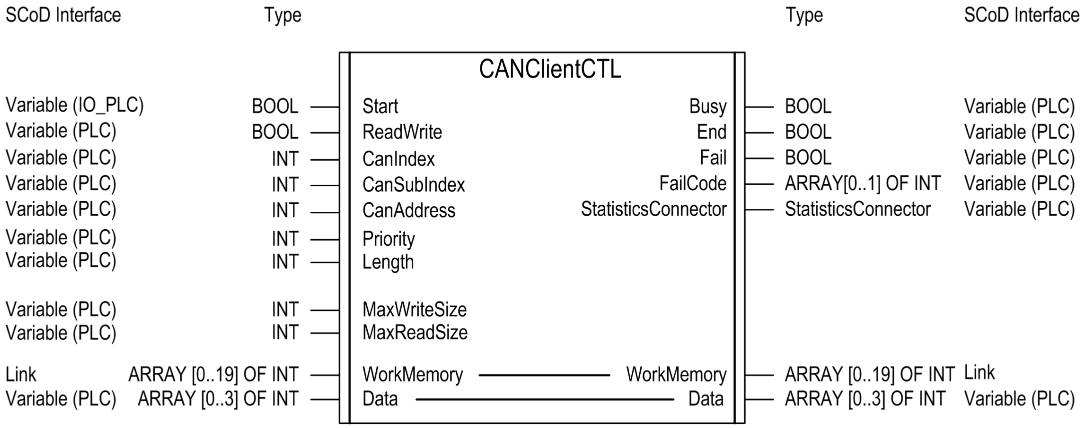
Overview

Use of the DFB pins in the SCoD editor is explained in the input/output diagram below.

The variables defined in the SCoD editor have in principle the same name as the pin (exception is that an underscore in the pin name will be skipped for the variable name).

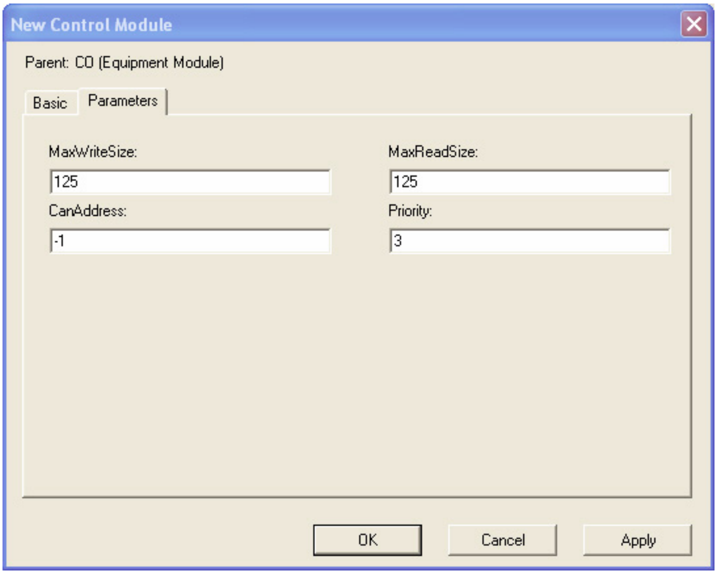
If the variables defined in the SCoD have the property ’Initial Value’ then the values of the instantiated SCoD variables can be modified manually in Unity Pro before the PLC restart.

Input Output Diagram



Parameter Tab

The figure below shows the **Parameter tab** of the **Properties Control Module** window.



The following table describes the properties:

Property	Type	Description
MaxReadSize	INT	Configuration parameter that indicates how many words will be read in the request issued by this client. This parameter is used for calculating and managing the work memory area (WorkMemory).
MaxWriteSize	INT	Configuration parameter that indicates how many words will be read in the request issued by this client. This parameter is used for calculating and managing the work memory area (WorkMemory).
CANAddress	INT	Address of the CANopen node that the client should access. The range of possible values goes from 1 to 127.
Priority	INT	Command priority. The lower the value, the higher the priority. (0 is the maximum priority)

15.3 CPortP: CANopen Port Premium

Overview

This section describes CPortP, the Port SCoD when using CANOpen technology with Premium PLC.

What's in this Section?

This section contains the following topics:

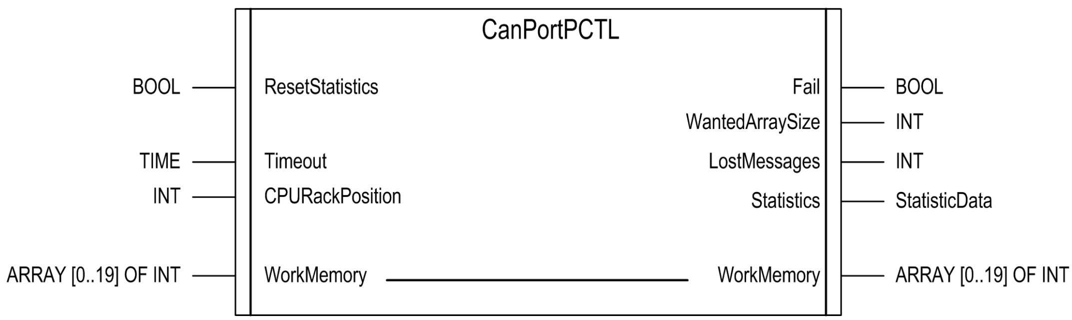
Description

General

CPortP is a Port *(see page 315)* function which is capable of serializing and managing requests sent to a physical medium working on CANopen.

Representation in FBD

The functional module that will be used in the program has the following aspect (at the FBD block level) after PLC generation in UNITY Pro:



Inputs

Input Parameter Description

Parameter	Type	Description
Reset Statistics	BOOL	Enabling this input resets output variables. When =1, the data on the Statistics output variable is reset to its default values (0). The input is level-based, i.e., statistics will remain set to 0 as long as the input is held as true..
Timeout	TIME	Defines the timeout value to be used in the DFB to activate the Fail output variable.
CPURackPosition	INT	Defines the slot in which the CPU is located on the rack.

Outputs

Output Parameter Description

Parameter	Type	Description
Fail	BOOL	Indicates that an error has occurred while transmitting a request. The bit cannot be reset as soon as a request is issued without an error, it is set to 0. The error codes must be checked in each DFB belonging to devices or serial Modbus clients.
WantedArraySize	INT	Indicates the size necessary for the WorkMemoryRS485 array to function correctly. Since arrays start on 0, the array must be declared with the minimum size of [0..WantedArraySize-1]. If additional serial Modbus (RTU) clients or devices are added to the program, the size of this variable will increase, making it necessary to monitor the WorkMemoryRS485 Array. This variable is calculated during the first Scanning cycle so adding serial Modbus (RTU) clients or devices during “online” mode has 2 consquences: • The size of the Array will not increase. • The ModbusGateway DFB will not take into account requests from these new clients.
LostMessages	INT	Indicates the number of messages that have been lost (that the client has not received). A value ≠ 0 often comes from a CANClient instance is probably not being executed on every single cycle or a programming error. If messages are being lost on the CANopen bus timing between calls of different clients, hardware (wiring) problem should be checked.
Statistics	StatisticData	Holds a structure with statistical data on the operation of the EthernetGatewayModbus DFB. The information obtained in this data structure is the statistic for the requests managed by this EthernetGatewayModbus DFB. Requests sent by clients that are associated with this DFB by means of the WorkMemory485 Array.

This tables describes the StatisticData type:

Name	Type	Description
RequestsSended	DINT	Total number of sent requests.
RequestsOk	DINT	Total number of requests that ended correctly.
RequestsError	DINT	Total number of requests that ended in an error
MinTime	DINT	Minimum time required to end a request (msec)
AvgTime	REAL	Average time required to end a request (msec)

Name	Type	Description
MaxTime	DINT	Maximum time required to end a request (msec)
LastTime	DINT	Time required to end the last request (msec)
CurrentTime	DINT	The time spent on processing the current request (0 if there is no request in progress)
RequestsSecond	DINT	Number of requests that can be sent per second (calculated with the AvgTime value).
LastCycleNumber	DINT	Number of controller cycles that the last request took to be executed.
CurrentCycleNumber	DINT	Number of controller cycles that the current request has taken so far (0 if there is no request in progress).
TimeOnQueue	DINT	Time that the last request has been waiting in the queue before being sent (only useful for client statistics).

Inputs/Outputs

WorkMemory Description

This Array holds the read/write requests of the clients/scanners.

During the first Scanning cycle, the devices reserve read/write %Mwords from this Array, based on the exchange words required by each DFB. The ModbusGateway DFB then counts them, and as a result specifies and knows how many pieces of data will be serialized.

⚠ CAUTION
UNEXPECTED EQUIPMENT OPERATION The WorkMemory variable must be adjusted previously to set: ● The size of the data exchange area between the device and client DFB ● The CanPortPCTL DFB to the size needed for the elements that are being used **Failure to follow these instructions can result in injury or equipment damage.**

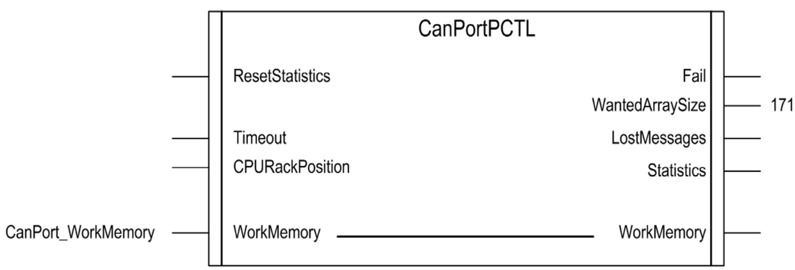
The system will be limited by fixed sizes or use memory that is not being used if this is not done.

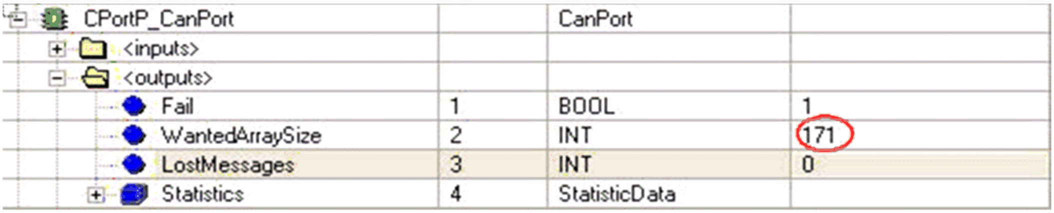
Calculating the WorkMemory Array Size

The necessary size for the array of the variable associated to WorkMemoryRS485 is automatically calculated by the DFBs of the port shown by the WantedArraySize output.

But it is necessary to check that the size of the Array for the WorkMemoryRS485 variable of the CanPortCTL DFB has a size greater than or equal to the one that the WantedArraySize variable requires.

Example with a CanPort DFB:





Editing this in Variables:

In order to check the value:

1. Execute the program with the calculated array
2. Check the value returned by the CanPortCTL DFB
3. Resize the array to the WantedArraySize value (the array must have a size of [0 .. WantedArraySize-1] as a minimum).

Representation as SCoD in UAG

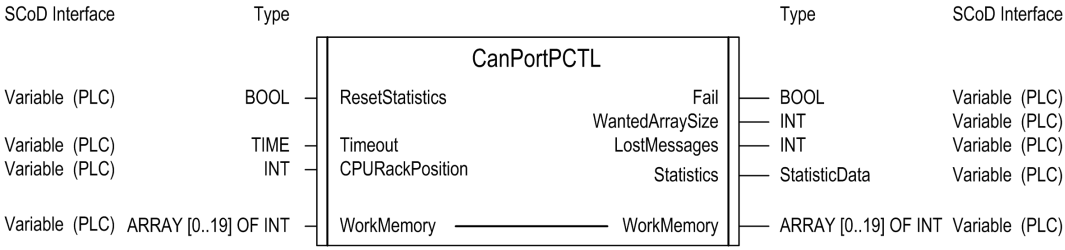
Overview

Use of the DFB pins in the SCoD editor is explained in the input/output diagram below.

The variables defined in the SCoD editor have in principle the same name as the pin (exception is that an underscore in the pin name will be skipped for the variable name).

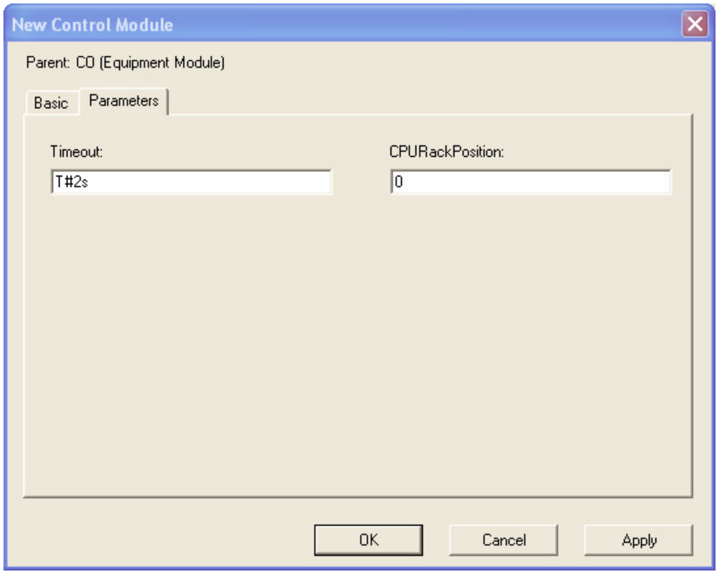
If the variables defined in the SCoD have the property ’Initial Value’ then the values of the instantiated SCoD variables can be modified manually in Unity Pro before the PLC restart.

Input Output Diagram



Parameter Tab

The figure below shows the **Parameter tab** of the **Properties Control Module** window.



The following table describes the properties:

Property	Type	Description
Timeout	TIME	Defines the timeout value to be used in the DFB to activate the Fail output variable.
CPURackPosition	INT	Defines the slot in which the CPU is located on the rack.

15.4 CPortM: CANopen Port Modicon

Overview

This section CPortM, the Port function when using CANOpen technology with Modicon PLC.

What's in this Section?

This section contains the following topics:

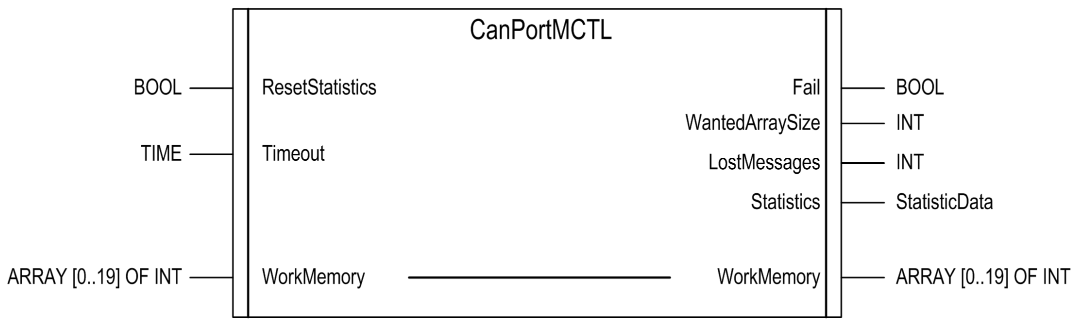
Description

General

CPortM is a Port *(see page 315)* function which is capable of serializing and managing requests sent to a physical medium working on serial Modbus.

Representation in FBD

The functional module that will be used in the program has the following aspect (at the FBD block level) after PLC generation in UNITY Pro:



Inputs

Input Parameter Description

Parameter	Type	Description
Reset Statistics	BOOL	Described in CPortP Inputs. *(see page 339)*
TimeOut	TIME	Described in CPortP Inputs. *(see page 339)*

Outputs

Output Parameter Description

Parameter	Type	Description
Fail	BOOL	Described in CPortP Outputs. *(see page 340)*
WantedArraySize	INT	Described in CPortP Outputs. *(see page 340)*
LostMessages	INT	Described in CPortP Outputs. *(see page 340)*
Statistic	StatisticData	Described in CPortP Outputs. *(see page 340)*

Inputs/Outputs

WorkMemoryRS485 (Array)

This Array is described in the CPortP Inputs/Outputs. *(see page 342)*

Representation as SCoD in UAG

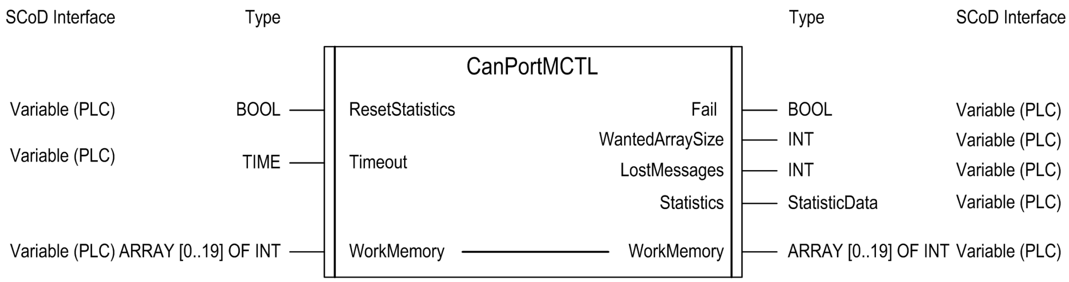
Overview

Use of the DFB pins in the SCoD editor is explained in the input/output diagram below.

The variables defined in the SCoD editor have in principle the same name as the pin (exception is that an underscore in the pin name will be skipped for the variable name).

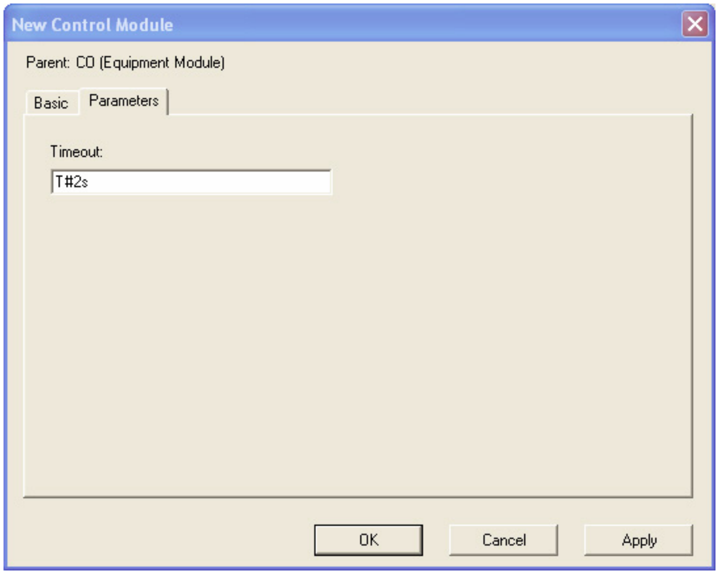
If the variables defined in the SCoD have the property 'Initial Value' then the values of the instantiated SCoD variables can be modified manually in Unity Pro before the PLC restart.

Input Output Diagram



Parameter Tab

The figure below shows the **Parameter tab** of the **Properties Control Module** window.



The following table describes the properties:

Property	Type	Description
Timeout	TIME	Defines the timeout value to be used in the DFB to activate the Fail output variable.

ModBus Communication

16

Overview

This chapter describes the ModBus communication SCoDs and their representation in Unity Pro.

WARNING

MISAPPLICATION OF MODULES

The modules in this section do not reflect any specific installation. Before adopting these modules for use in a specific application, the engineer must:

- conduct a safety analysis for the application and equipment installed
- verify that all modules are appropriate for the equipment or function in the installation
- supply appropriate parameters, particularly for limits
- check that all sensors and actuators are compatible with the modules selected
- thoroughly test all functions during verification and commissioning
- provide independent paths for critical control functions (emergency stop, over-limit conditions etc.) according to the safety analysis and applicable codes and regulations.

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

What's in this Chapter?

This chapter contains the following sections:

16.1 MBScanner: ModBus Scanner

Overview

This section describes the MBScanner, Communication Scanner SCoD using Modbus technology (Modbus Messaging).

What's in this Section?

This section contains the following topics:

Description

General

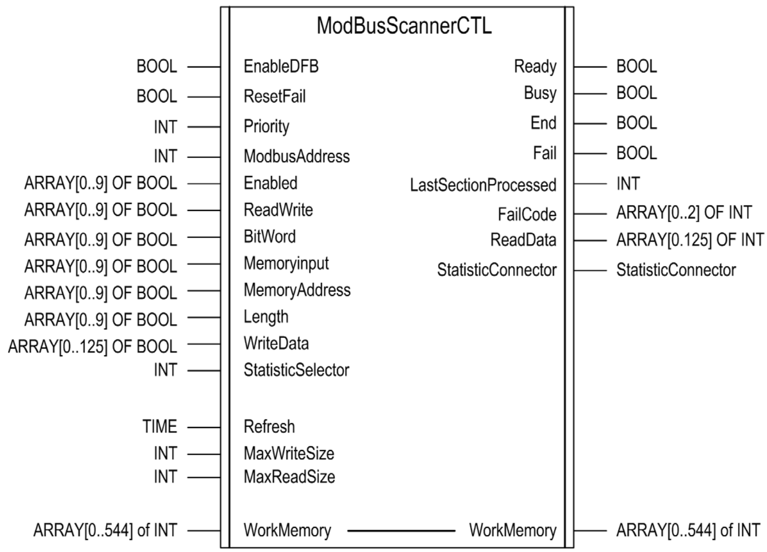
The ModBusScannerCTL Communications Scanner *(see page 312)* is a function block that uses an internal Client to issue several sequential requests to different memory positions in the remote device.

The communications scanner blocks are implemented as listed below:

Software	Implemented as
Unity Pro	ModBusScannerCTL
UAG	SCoD MBScanner

Representation in FBD

The MBScanner SCoD is represented with the ModBusScannerCTL DFB.



Inputs

Input Parameter Description

Parameter	Type	Description
EnableDFB	BOOL	Described in CScanner - Inputs *(see page 320)*
ResetFail	BOOL	Described in CScanner - Inputs *(see page 320)*
Priority	INT	Described in CScanner - Inputs *(see page 320)*
ModBusAddress	INT	Address of the Modbus slave that the client should access. The range of possible values goes from 1 to 255.
Enabled	ARRAY of INT	Described in CScanner - Inputs *(see page 320)*
ReadWrite	ARRAY of INT	Described in CScanner - Inputs *(see page 320)*
BitWord	ARRAY of INT	Described in CScanner - Inputs *(see page 320)*
MemoryInput	ARRAY of INT	Each position in the array selects memory zones with false (40000 or %MW registers) or input zones with true (30000 or %IW registers).
MemoryAddress	ARRAY of INT	Each position in the array holds the memory address where the requested scanner operation will start. Any value is valid for the master, since the slave must validate the address based on its memory map.
Length	ARRAY of INT	Described in CScanner - Inputs *(see page 320)*
WriteData	ARRAY of INT	Described in CScanner - Inputs *(see page 320)*
StatisticsSelector	INT	Described in CScanner - Inputs *(see page 320)*
Refresh	TIME	Described in CScanner - Inputs *(see page 320)*
MaxReadSize	INT	Described in CScanner - Inputs *(see page 320)*
MaxWriteSize	INT	Described in CScanner - Inputs *(see page 320)*

Outputs

Output Parameter Description

Parameter	Type	Description
Ready	BOOL	Described in CScanner - Outputs *(see page 323)*
Busy	BOOL	Described in CScanner - Outputs *(see page 323)*
End	INT	Described in CScanner - Outputs *(see page 323)*
Fail	BOOL	Described in CScanner - Outputs *(see page 323)*
FailCode	INT[3]	Described in CScanner - Outputs *(see page 323)*
LastSectionProcessed	INT	Described in CScanner - Outputs *(see page 323)*
StatisticConnector	StatisticConnector	Described in CScanner - Outputs *(see page 323)*

Inputs/Outputs

Inputs/Outputs Parameter Description

Parameter	Type	Description
WorkMemory	Array	Described in CScanner - Inputs/Outputs *(see page 324)*

Representation as SCoD in UAG

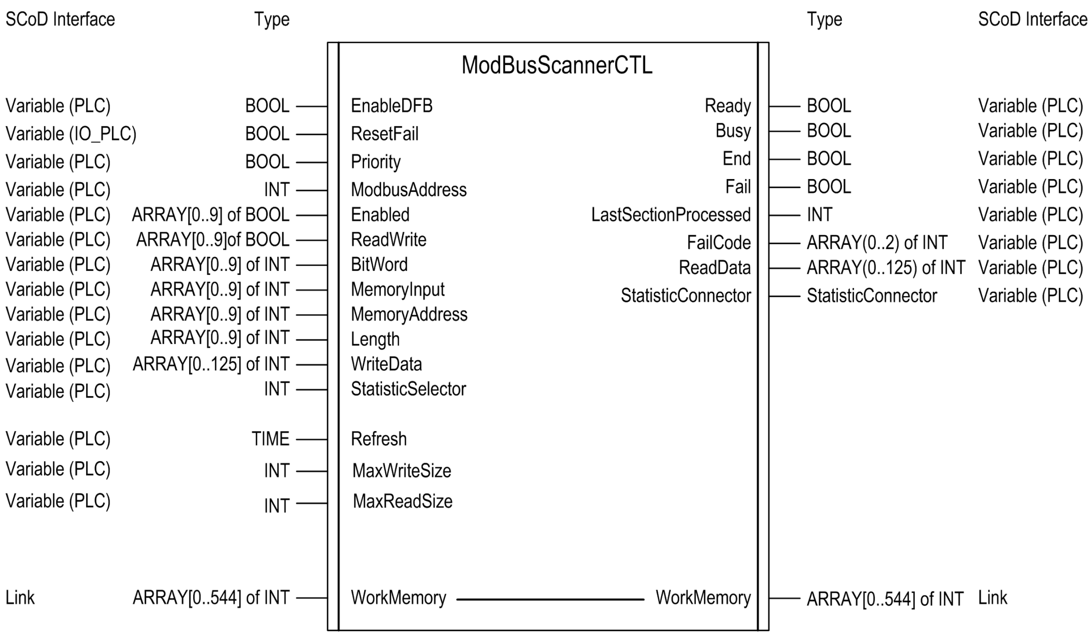
Overview

Use the DFB pins in the SCoD editor is explained in the input/output diagram below.

The variables defined in the SCoD editor have in principle the same name as the pin (exception is that an underscore in the pin name will be skipped for the variable name).

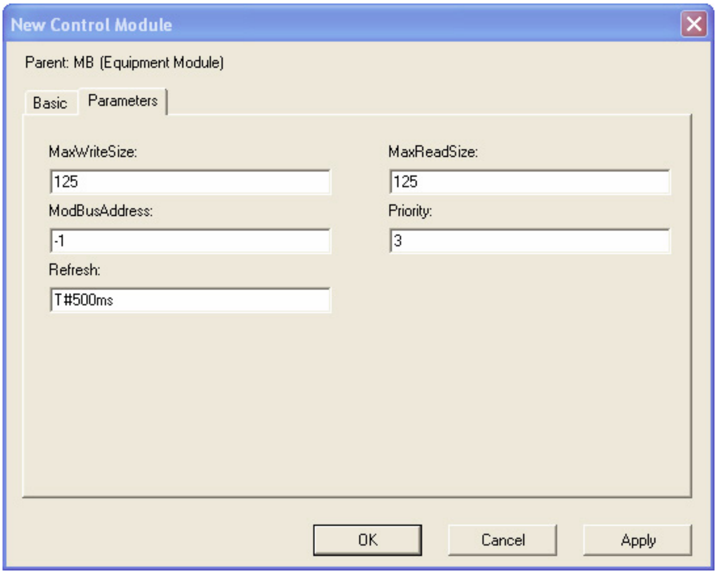
If the variables defined in the SCoD have the property ’Initial Value’ then the values of the instantiated SCoD variables can be modified manually in Unity Pro before the PLC restart.

Input Output Diagram



Parameter Tab

The figure below shows the **Parameter tab** of the **Properties Control Module** window.



The following table describes the properties:

Property	Type	Description
MaxReadSize	INT	Configuration parameter that indicates how many words will be read in the request issued by this client. This parameter is used for calculating and managing the work memory area (WorkMemory).
MaxWriteSize	INT	Configuration parameter that indicates how many words will be written in the request issued by this client. This parameter is used for calculating and managing the work memory area (WorkMemory).
ModBusAddress	INT	Address of the Modbus slave that the client should access. The range of possible values goes from 1 to 255.
Priority	INT	Command priority. The lower the value, the higher the priority. (0 is the maximum priority)
Refresh	TIME	Time that the scanner will wait before it repeats an operation cycle.

16.2 MBClientBs: ModBus Client Basic

Overview

This section describes the MBClientBs SCoD, the Client SCoD with ModBus technology.

What's in this Section?

This section contains the following topics:

Description

Function Description

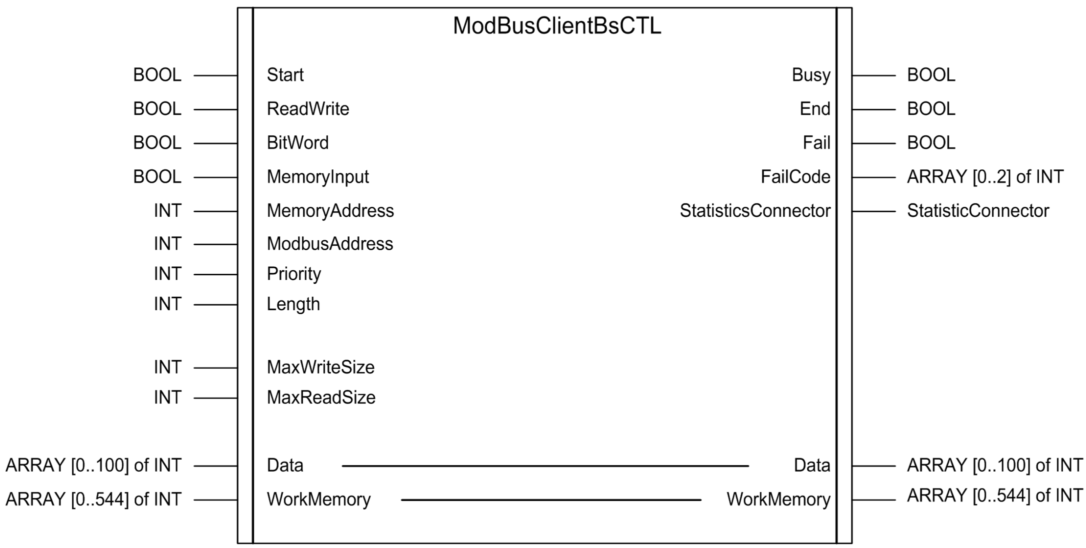
This Communication Client *(see page 314)*enables device data to be written or read to/from ModBus technologies.

The communications clients block is implemented as listed below:

Software	Implemented as
Unity Pro	ModBusClientBsCTL
UAG	SCoD MBClientBs

Representation in FBD

The MBClientBs SCoD is represented with the ModBusClientBsCTL DFB.



Inputs

Input Parameter Description

Parameter	Type	Description
Start	BOOL	Described in the CClient - Inputs section. *(see page 330)*
ReadWrite	BOOL	Described in the CClient - Inputs section. *(see page 330)*
BitWord	BOOL	Described in the CClient - Inputs section. *(see page 330)*
MemoryInput	BOOL	Described in the EClient - Inputs section *(see page 400)*
MemoryAddress	INT	Described in the EClient - Inputs section *(see page 400)*
ModBusAddress	INT	Address of the Modbus slave that the client should access. The range of possible values goes from 1 to 255.
Priority	INT	Described in the CClient - Inputs section. *(see page 330)*
Length	INT	Described in the CClient - Inputs section. *(see page 330)*
MaxReadSize	INT	Described in the CClient Client - Inputs/Outputs section. *(see page 334)*
MaxWriteSize	INT	Described in the CClient Client - Inputs/Outputs section. *(see page 334)*

Outputs

Output Parameter Description

Parameter	Type	Description
Busy	BOOL	Described in the CClient - Outputs section. *(see page 332)*
End	INT	Described in the CClient - Outputs section. *(see page 332)*
Fail	BOOL	Described in the CClient - Outputs section. *(see page 332)*
FailCode	INT[3]	Described in the CClient - Outputs section. *(see page 332)*
StatisticsConnector	StatisticsConnector	Described in the CClient - Outputs section. *(see page 332)*

Inputs/Outputs

Inputs/Outputs Parameter Description

Parameter	Type	Description
Data	Array	Described in the CClient - Inputs/Outputs section. *(see page 334)*
WorkMemory	Array	Described in the CClient - Inputs/Outputs section. *(see page 334)*

Representation as SCoD in UAG

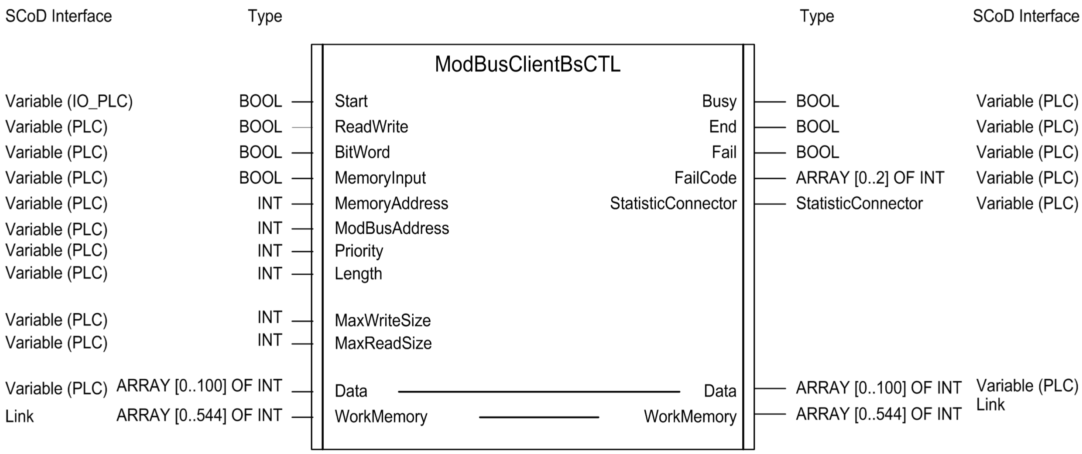
Overview

Use of the DFB pins in the SCoD editor is explained in the input/output diagram below.

The variables defined in the SCoD editor have in principle the same name as the pin (exception is that an underscore in the pin name will be skipped for the variable name).

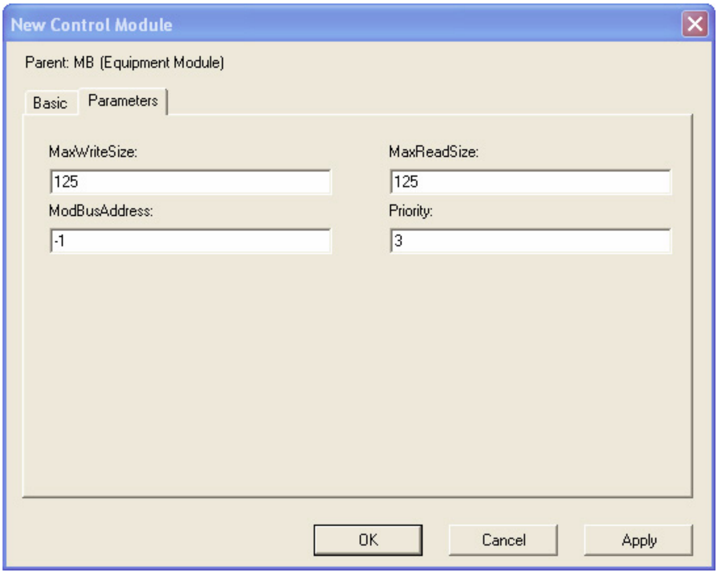
If the variables defined in the SCoD have the property 'Initial Value' then the values of the instantiated SCoD variables can be modified manually in Unity Pro before the PLC restart.

Input Output Diagram



Parameter Tab

The figure below shows the **Parameter tab** of the **Properties Control Module** window.



The following table describes the properties:

Property	Type	Description
MaxReadSize	INT	Configuration parameter that indicates how many words will be read in the request issued by this client. This parameter is used for calculating and managing the work memory area (WorkMemory).
MaxWriteSize	INT	Configuration parameter that indicates how many words will be read in the request issued by this client. This parameter is used for calculating and managing the work memory area (WorkMemory).
ModBusAddress	STRING [26]	Address of the Modbus slave that the client should access. The range of possible values goes from 1 to 255.
Priority	INT	Command priority. The lower the value, the higher the priority. (0 is the maximum priority)

16.3 MBPortM: ModBus Port Modicon 340

Overview

This section describes MBPortM, the Port SCoD when using ModBus technology with Modicon PLC.

What's in this Section?

This section contains the following topics:

Description

Overview

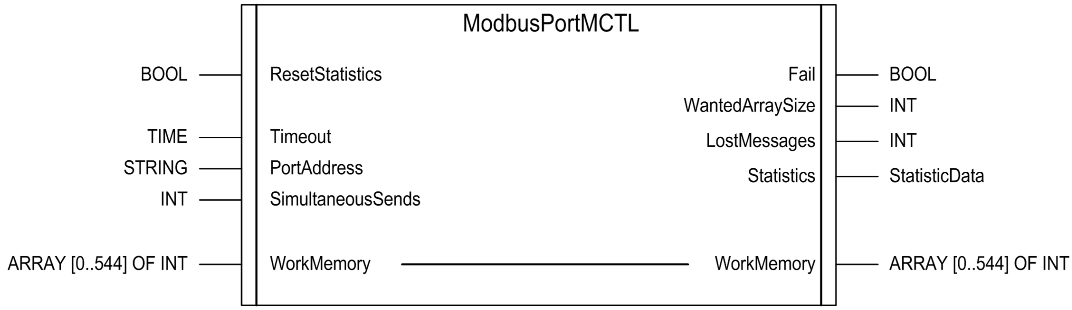
The MBPortM is a Port *(see page 315)* function which is capable of serializing and managing requests sent to a physical medium working on serial Modbus.

The communications port block is implemented as listed below:

Software	Implemented as
Unity Pro	ModBusPortMCTL
UAG	SCoD ModbusPortM

Representation in FBD

The functional module that will be used in the program has the following aspect (at the FBD block level) after PLC generation in UNITY Pro:.



Inputs

Input Parameter Description

Parameter	Type	Description
Reset Statistics	BOOL	Described in CPortP Inputs. *(see page 339)*
TimeOut	TIME	Described in CPortP Inputs. *(see page 339)*
PortAddress	STRING	Described in EPortM Inputs. *(see page 416)*
SimultaneousSends	INT	Described in EPortQ Inputs. *(see page 430)*

Outputs

Output Parameter Description

Parameter	Type	Description
Fail	BOOL	Described in CPortP Outputs. *(see page 340)*
WantedArraySize	INT	Described in CPortP Outputs. *(see page 340)*
LostMessages	INT	Described in CPortP Outputs. *(see page 340)*
Statistics	StatisticsData	Described in CPortP Outputs. *(see page 340)*

Inputs/Outputs

WorkMemory (Array)

This Array is described in the CPortP Inputs/Outputs. *(see page 342)*

Representation as SCoD in UAG

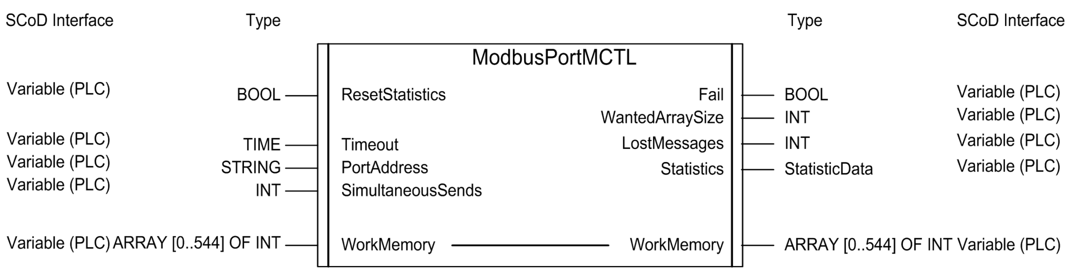
Overview

Use of the DFB pins in the SCoD editor is explained in the input/output diagram below.

The variables defined in the SCoD editor have in principle the same name as the pin (exception is that an underscore in the pin name will be skipped for the variable name).

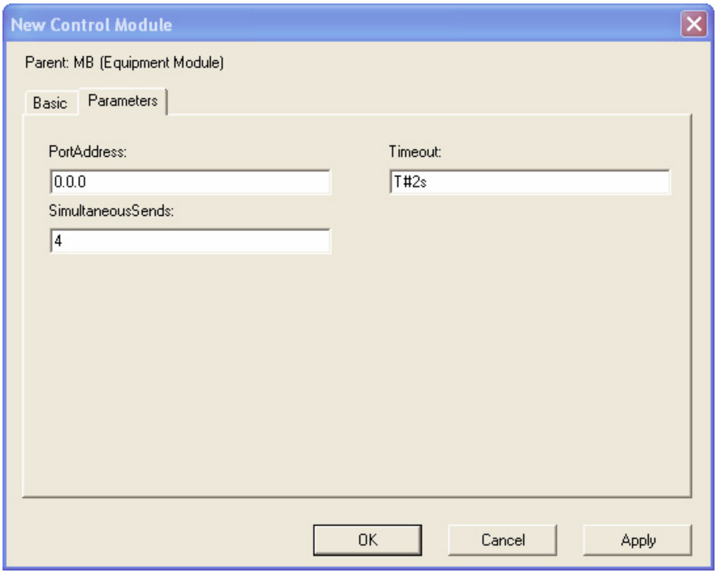
If the variables defined in the SCoD have the property 'Initial Value' then the values of the instantiated SCoD variables can be modified manually in Unity Pro before the PLC restart.

Input Output Diagram



Parameter Tab

The figure below shows the **Parameter tab** of the **Properties Control Module** window.



The following table describes the properties:

Property	Type	Description
SimultaneousSends	INT	Number of requests queued (serialized) in the Modbus Master.
PortAddress	STRING	Defines the physical port that the DFB will use to send the Modbus TCP/IP request in Modicon M340 PLCs.
Timeout	TIME	Defines the timeout value to be used in the DFB to activate the Fail output variable.

16.4 MBPortP: ModBus Port Premium

Overview

This section ModBusPortP, the Port function when using ModBus technology with Premium PLC.

What's in this Section?

This section contains the following topics:

Description

General

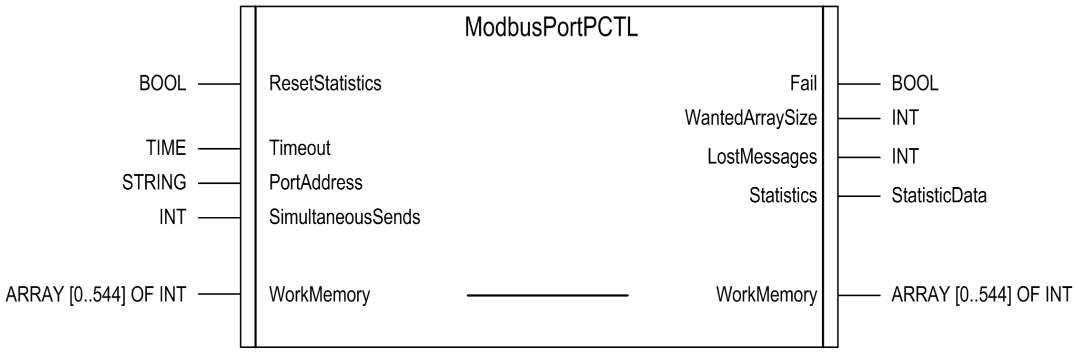
MBPortP FB is a Port *(see page 315)* function which is capable of serializing and managing requests sent to a physical medium working on serial Modbus.

The communications port block is implemented as listed below:

Software	**Implemented as**
Unity Pro	ModBusPortPCTL
UAG	SCoD ModbusPortP

Representation in FBD

The functional module that will be used in the program has the following aspect (at the FBD block level) after PLC generation in UNITY Pro:.



Inputs

Input Parameter Description

Parameter	Type	Description
ResetStatistics	BOOL	Described in CPortP Inputs. *(see page 339)*
TimeOut	TIME	Described in CPortP Inputs. *(see page 339)*
PortAddress	STRING	Described in EPortM Inputs. *(see page 416)*
SimultaneousSends	INT	Described in EPortQ Inputs. *(see page 430)*

Outputs

Output Parameter Description

Parameter	Type	Description
Fail	BOOL	Described in CPortP Outputs. *(see page 340)*
WantedArraySize	INT	Described in CPortP Outputs. *(see page 340)*
LostMessages	INT	Described in CPortP Outputs. *(see page 340)*
Statistics	StatisticsData	Described in CPortP Outputs. *(see page 340)*

Inputs/Outputs

WorkMemory (Array)

This Array is described in the CPortP Inputs/Outputs. *(see page 342)*

Representation as SCoD in UAG

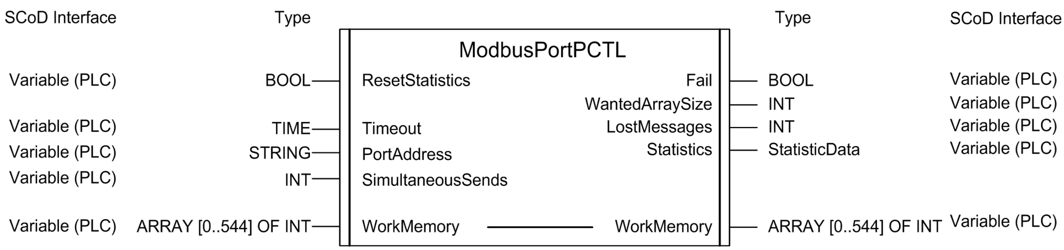
Overview

Use the DFB pins in the SCoD editor is explained in the input/output diagram below.

The variables defined in the SCoD editor have in principle the same name as the pin (exception is that an underscore in the pin name will be skipped for the variable name).

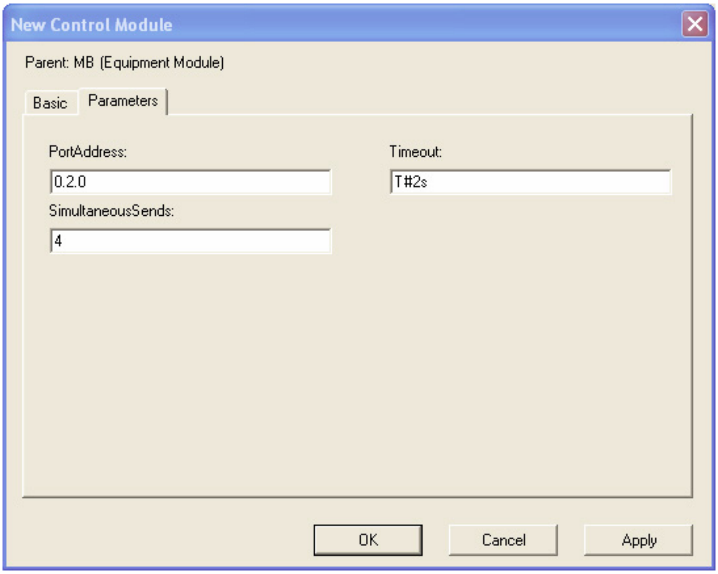
If the variables defined in the SCoD have the property 'Initial Value' then the values of the instantiated SCoD variables can be modified manually in Unity Pro before the PLC restart.

Input Output Diagram



Parameter Tab

The figure below shows the **Parameter tab** of the **Properties Control Module** window.



The following table describes the properties:

Property	Type	Description
PortAddress	STRING	Defines the physical port that the DFB will use to send the Modbus TCP/IP request in Modicon M340 PLCs.
Timeout	TIME	Defines the timeout value to be used in the DFB to activate the Fail output variable.
SimultaneousSends	INT	Number of requests queued (serialized) in the Modbus Master.

Tools

17

Overview

This chapter describes the SCoD Statistics.

⚠ WARNING

MISAPPLICATION OF MODULES

The modules in this section do not reflect any specific installation. Before adopting these modules for use in a specific application, the engineer must:

- conduct a safety analysis for the application and equipment installed
- verify that all modules are appropriate for the equipment or function in the installation
- supply appropriate parameters, particularly for limits
- check that all sensors and actuators are compatible with the modules selected
- thoroughly test all functions during verification and commissioning
- provide independent paths for critical control functions (emergency stop, over-limit conditions etc.) according to the safety analysis and applicable codes and regulations.

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

What's in this Chapter?

This chapter contains the following topics:

Statistics - Description

General

The Statistics block is implemented as listed below:

Software	Implemented as
Unity Pro	DFB StatisticsCounter
UAG	SCoD StatisticsCounter

Function Description

The statistics block enables to assess the quality and speed of s on a Modbus, Ethernet, or CANopen network.

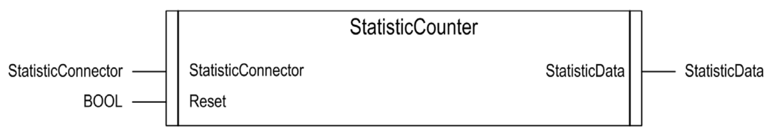
It enables calculations to be performed on the Client, Scanner, Port, and Gateway blocks from the entire communications library and on devices in Modbus communications.

The Statistics module main functions are summarized below:

Function	Description
Requests	The number of requests issued is calculated.
Times	Communication times are calculated (request time, time of last request, etc.)
Information	Communications information is calculated (maximum and minimum values, controller cycles, etc.)
Reset	The block allows the described totalizers to be reset.

Representation in FBD

The functional module used in the program has the following aspect (at the FBD block level) after PLC generation in UNITY Pro:



Statistic - Inputs

StatisticsConnector (StatisticsConnector)

Provides, to the DFB, the data from which to derive the statistics. Components that can provide statistics have an output of this type. It is possible to directly match the output variable of any component in order to evaluate statistics.

This table describes the StatisticsConnector type:

Name	Type	Description
Start	BOOL	Operation has started
EndOk	BOOL	Operation ended correctly
EndNOK	BOOL	Operation ended with an error
PartialTime	DINT	Partial time

Reset (Bool)

Enabling this input resets the output variables.

When this signal is true, the data on the Statistics output variable is reset to its default values (0).

The input is level-based, i.e., statistics will remain set to 0 as long as the input is held as true.

Statistic - Outputs

StatisticData (StatisticData)

Provides statistical data obtained from the DFB that is being analyzed.

This tables describes the StatisticData type:

Name	Type	Description
RequestsSended	DINT	Total number of sent requests.
RequestsOk	DINT	Total number of requests that ended correctly.
RequestsError	DINT	Total number of requests that ended in an error
MinTime	DINT	Minimum time required to end a request (msec)
AvgTime	REAL	Average time required to end a request (msec)
MaxTime	DINT	Maximum time required to end a request (msec)
LastTime	DINT	Time required to end the last request (msec)
CurrentTime	DINT	The time spent on processing the current request (0 if there is no request in progress)
RequestsSecond	DINT	Number of requests that can be sent per second (calculated with the AvgTime value).
LastCycleNumber	DINT	Number of controller cycles that the last request took to be executed.
CurrentCycleNumber	DINT	Number of controller cycles that the current request has taken so far (0 if there is no request in progress).
TimeOnQueue	DINT	Time that the last request has been waiting in the queue before being sent (only useful for client statistics).

Statistics - Representation as SCoD in UAG

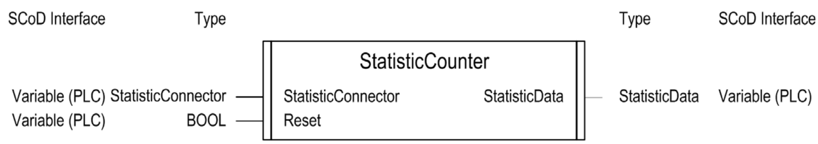
Overview

Use of the DFB pins in the SCoD editor is explained in the input/output diagram below.

The variables defined in the SCoD editor have in principle the same name as the pin (exception is that an underscore in the pin name will be skipped for the variable name).

If the variables defined in the SCoD have the property 'Initial Value' then the values of the instantiated SCoD variables can be modified manually in Unity Pro before the PLC restart.

Input Output Diagram



Ethernet Communication

18

Overview

This chapter describes the ethernet communcation based functions.

⚠ WARNING
MISAPPLICATION OF MODULES The modules in this section do not reflect any specific installation. Before adopting these modules for use in a specific application, the engineer must: ● conduct a safety analysis for the application and equipment installed ● verify that all modules are appropriate for the equipment or function in the installation ● supply appropriate parameters, particularly for limits ● check that all sensors and actuators are compatible with the modules selected ● thoroughly test all functions during verification and commissioning ● provide independent paths for critical control functions (emergency stop, over-limit conditions etc.) according to the safety analysis and applicable codes and regulations. **Failure to follow these instructions can result in death, serious injury, or equipment damage.**

What's in this Chapter?

This chapter contains the following sections:

18.1 EScanner: Ethernet Scanner

Overview

This section describes the EScanner SCoD, the Communication Scanner SCoD using Ethernet technology (Ethernet Messaging).

What's in this Section?

This section contains the following topics:

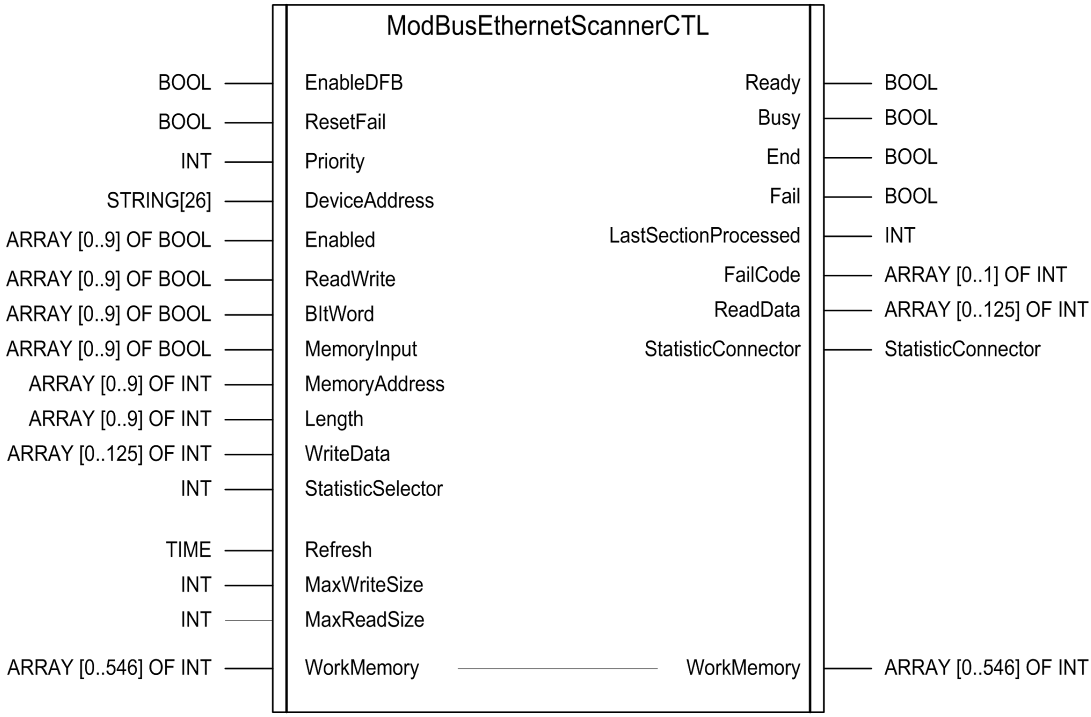
Description

Function Description

The EScanner is a Communications Scanner *(see page 312)* that periodically refreshes read or write requests for n registers issued to a single node on an Ethernet communications network.

Representation in FBD

The EScanner SCoD is represented with the ModBusEthernetScannerCTL DFB.



Inputs

General

Memory Address indexes are usually expressed in decimal values in device manuals. In order to express the index as a hexadecimal value, use the string “16#” before the index, i.e., 16#index.

Input Parameter Description

Parameter	Type	Description
EnableDFB	BOOL	Described in CScanner - Inputs *(see page 320)*
ResetFail	BOOL	Described in CScanner - Inputs *(see page 320)*
Priority	INT	Described in CScanner - Inputs *(see page 320)*
DeviceAddress	REAL	Device address. See the IP Addressing table below.
MemoryInput	ARRAY of INT	Each position in the array selects memory zones with false (40000 or %MW registers) or input zones with true (30000 or %IW registers).
MemoryAddress	ARRAY of INT	Each position in the array holds the memory address where the requested scanner operation will start. Any value is valid for the master, since the slave must validate the address based on its memory map.
Enabled	ARRAY of INT	Described in CScanner - Inputs *(see page 320)*
ReadWrite	ARRAY of INT	Described in CScanner - Inputs *(see page 320)*
BitWord	ARRAY of INT	Described in CScanner - Inputs *(see page 320)*
Length	ARRAY of INT	Described in CScanner - Inputs *(see page 320)*
WriteData	ARRAY of INT	Described in CScanner - Inputs *(see page 320)*
StatisticsSelector	INT	Described in CScanner - Inputs *(see page 320)*
Refresh	TIME	Described in CScanner - Inputs *(see page 320)*
MaxReadSize	INT	Described in CScanner - Inputs *(see page 320)*
MaxWriteSize	INT	Described in CScanner - Inputs *(see page 320)*

Depending on the platform, the following definitions apply for IP addressing:

Device	IP Addressing
M340	{IP}ID
Premium	{XWAY}.Slot.Channel.ID
Quantum	Slot.{IP}.ID

NOTE: Do not omit the punctuation marks.

Outputs

Output Parameter Description

Parameter	Type	Description
Ready	BOOL	Described in CScanner - Outputs *(see page 323)*
Busy	BOOL	Described in CScanner - Outputs *(see page 323)*
End	INT	Described in CScanner - Outputs *(see page 323)*
Fail	BOOL	Described in CScanner - Outputs *(see page 323)*
LastSectionProcessed	INT	Described in CScanner - Outputs *(see page 323)*
FailCode	INT[3]	Described in CScanner - Outputs *(see page 323)*
StatisticConnector	StatisticConnector	Described in CScanner - Outputs *(see page 323)*

Inputs/Outputs

Inputs/Outputs Parameter Description

Parameter	Type	Description
WorkMemory	Array	Described in CScanner - Inputs/Outputs *(see page 324)*

Representation as SCoD in UAG

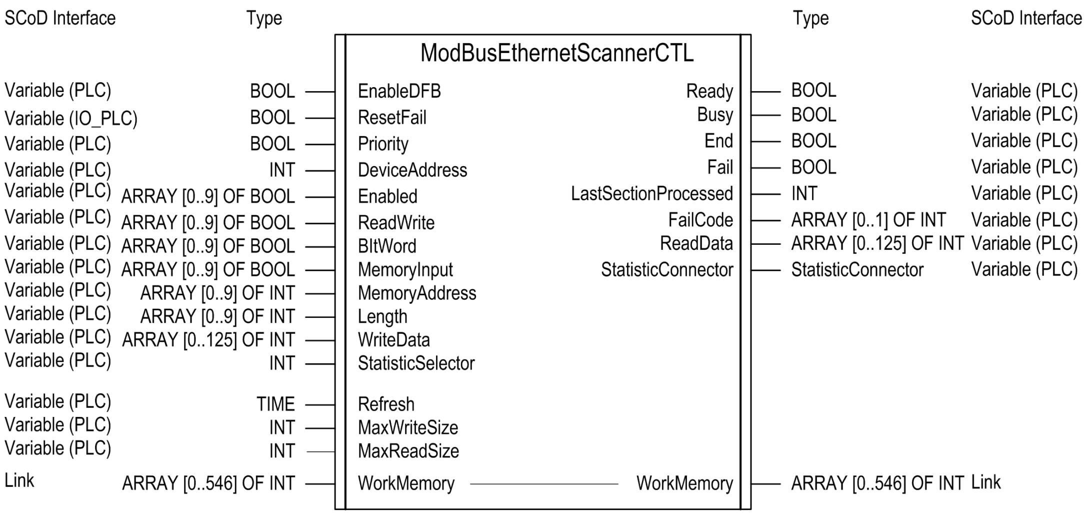
Overview

Use of the DFB pins in the SCoD editor is explained in the input/output diagram below.

The variables defined in the SCoD editor have in principle the same name as the pin (exception is that an underscore in the pin name will be skipped for the variable name).

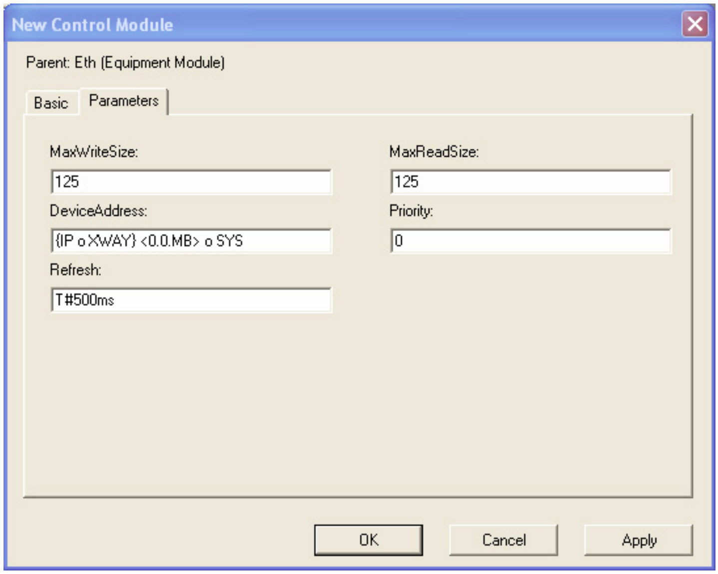
If the variables defined in the SCoD have the property 'Initial Value' then the values of the instantiated SCoD variables can be modified manually in Unity Pro before the PLC restart.

Input Output Diagram



Parameter Tab

The figure below shows the **Parameter tab** of the **Properties Control Module** window.



The following table describes the properties:

Property	**Type**	**Description**
MaxReadSize	INT	Configuration parameter that indicates how many words will be read in the request issued by this client. This parameter is used for calculating and managing the work memory area (WorkMemory).
MaxWriteSize	INT	Configuration parameter that indicates how many words will be read in the request issued by this client. This parameter is used for calculating and managing the work memory area (WorkMemory).
DeviceAddress	STRING [26]	Device address. See the “Ethernet Technology” chapter in this manual in order to see examples. *(see page 436)*
Priority	INT	Command priority. The lower the value, the higher the priority. (0 is the maximum priority)
Refresh	TIME	Time that the scanner will wait before it repeats an operation cycle.

18.2 EClient: Ethernet Client

Overview

This section describes the EClient SCoD, the Client SCoD with ethernet communication technology.

What's in this Section?

This section contains the following topics:

Description

Function Description

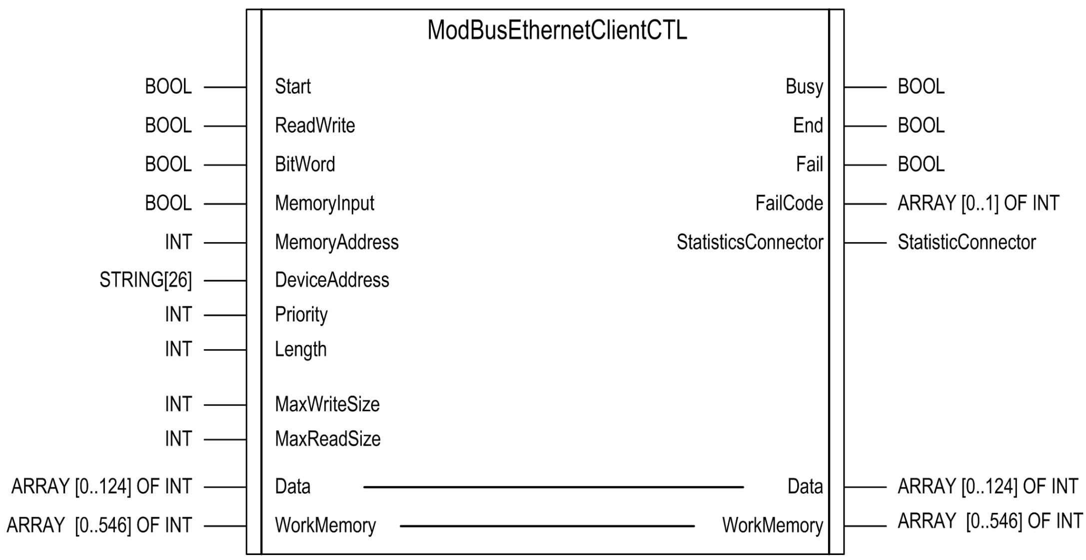
This Communication Client *(see page 314)* enables device data to be written or read to/from Ethernet technologies.

The module main functions are described below:

Function	Description
Read/Write	Enables select whether the parameter has read or write access.
Multiple register reading/writing	Enables several consecutive registers to be read (available only with ModBusEthernetClient and ModBusClientBasic).
Error management	Monitors transaction errors, identifying them on three levels in order to determine the source of the error.
Priorities	Enables to define priorities for systems with multiple client accesses.
Statistics	Obtains the transactions OK/NOK status, as well as their access times.

Representation in FBD

The EClient SCoD is represented with the ModBusEthernetClientCTL DFB.



Inputs

Input Parameter Description

Parameter	Type	Description
Start	BOOL	Described in the CClient - Inputs section. *(see page 330)*
ReadWrite	BOOL	Described in the CClient - Inputs section. *(see page 330)*
BitWord	BOOL	Described in the CClient - Inputs section. *(see page 330)*
MemoryInput	BOOL	Selects memory zones with false (40000 or %MW registers) or input zones with true (30000 or %IW registers).
DeviceAddress	STRING[26]	Device address. See the IP Addressing table below.
MemoryAddress	INT	Memory address where the requested operation will start. Any value is valid for the master, since the slave must validate the address based on its memory map. Values can be entered as hexadecimal or decimal values.
Priority	INT	Described in the CClient - Inputs section. *(see page 330)*
Length	INT	Described in the CClient - Inputs section. *(see page 330)*
MaxReadSize	INT	Described in the CClient - Inputs/Outputs section. *(see page 334)*
MaxWriteSize	INT	Described in the CClient - Inputs/Outputs section. *(see page 334)*

Depending on the platform, the following definitions apply:

Device	IP Addressing
M340	{IP}ID
Premium	{XWAY}.Slot.Channel.ID
Quantum	Slot.{IP}.ID

NOTE: Do not omit the punctuation marks.

Outputs

Output Parameter Description

Parameter	Type	Description
Busy	BOOL	Described in the CClient - Outputs section. *(see page 332)*
End	INT	Described in the CClient - Outputs section. *(see page 332)*
Fail	BOOL	Described in the CClient - Outputs section. *(see page 332)*
FailCode	INT[3]	Described in the CClient - Outputs section. *(see page 332)*
StatisticsConnector	StatisticsConnector	Described in the CClient - Outputs section. *(see page 332)*

Inputs/Outputs

Inputs/Outputs Parameter Description

Parameter	Type	Description
WorkMemory	Array	Described in the CClient - Inputs/Outputs section. *(see page 334)*
Data	Array	Described in the CClient - Inputs/Outputs section. *(see page 334)*

Representation as SCoD in UAG

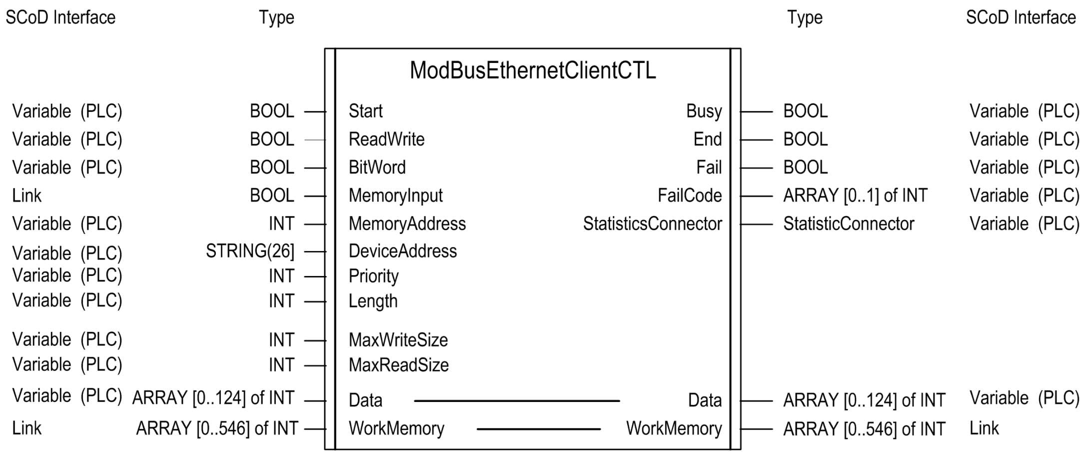
Overview

Use of the DFB pins in the SCoD editor is explained in the input/output diagram below.

The variables defined in the SCoD editor have in principle the same name as the pin (exception is that an underscore in the pin name will be skipped for the variable name).

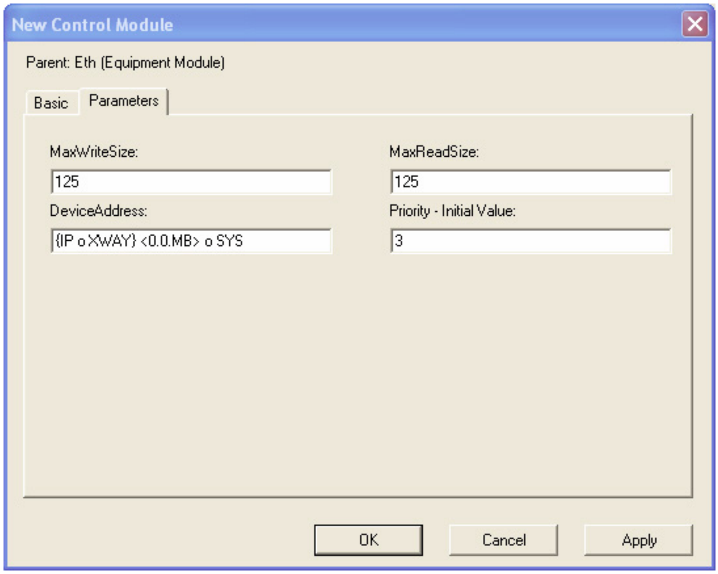
If the variables defined in the SCoD have the property ’Initial Value’ then the values of the instantiated SCoD variables can be modified manually in Unity Pro before the PLC restart.

Input Output Diagram



Parameter Tab

The figure below shows the **Parameter tab** of the **Properties Control Module** window.



The following table describes the properties:

Property	**Type**	**Description**
MaxReadSize	INT	Configuration parameter that indicates how many words will be read in the request issued by this client. This parameter is used for calculating and managing the work memory area (WorkMemory).
MaxWriteSize	INT	Configuration parameter that indicates how many words This parameter is used for calculating and managing the work memory area (WorkMemory).
DeviceAddress	STRING [26]	Device address. See the “Ethernet Technology” chapter in this manual in order to see examples. *(see page 436)*
Priority	INT	Command priority. The lower the value, the higher the priority. (0 is the maximum priority)

18.3 EGtwMB: Serial Modbus-Ethernet Gateway

Overview

This section describes the EGtwMB SCoD.

What's in this Section?

This section contains the following topics:

EGtwMB - Description

General

TheSerial Modbus-Ethernet Gateway block is implemented as listed below:

Software	Implemented as
Unity Pro	DFB ModBusGatewayCTL
UAG	SCoD EGtwMB

Function Description

The EthernetGatewayModbus gateway is the DFB that functions as a bridge between serial devices and Ethernet buses.

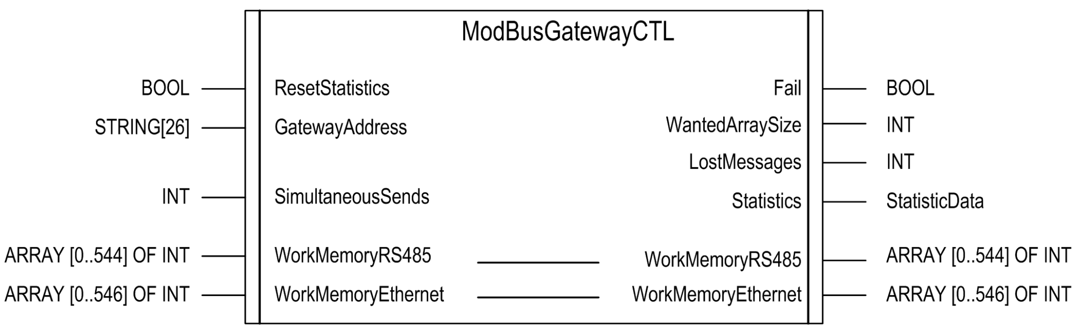
By using a gateway (TSXETG100), it is possible to establish communications for a CPU that only has Ethernet ports with devices that only implement an RS485 interface.

The module main functions are described below:

Function	Description
Request conversion	Enables read/write requests to be issued to devices on a 485 bus through an Ethernet port (available only for ModBusClientBasic and ModbusScanner).
Statistics	Obtains the transactions OK/NOK status, as well as their access times.

Representation in FBD

The functional module that will be used in the program has the following aspect (at the FBD block level) after PLC generation in UNITY Pro:



Inputs

Input Parameter Description

Parameter	Type	Description
Reset Statistics	BOOL	Activating this input enables communications. If the function is not enabled, initialize the scanner.
GatewayAddress	String [26]	IP address for the gateway where the serial Modbus devices are physically located. See IP addressing table below.
SimultaneousSends	Int	Indicates how many requests are in the gateway queue. The value ranges from 1 to 4. The more requests in the gateway queue, the better the system general performance, but the following features must be taken into account: • The maximum number of requests in the CPU buffer • The maximum number of requests for a gateway queue. • Priority management does not take into account requests that are queued in the hardware buffer, which makes it possible for a high-priority command to have to wait until lower-priority commands that are already in the queue are sent first. This results in a longer response time for high-priority commands.

Depending on the platform used and on the position in which the Ethernet port is located inside the PLC rack, IP addressing must be entered in different formats:

Device	IP Addressing (GatewayAddress)
M340	{IP}
Premium	{XWay network.XWay address}.Slot.Channel.
Premium Copro	{XWay network.XWay address}.0.0. In Premium range PLCs in which requests are issued through the CPU integrated Ethernet port, the channel will always be “zero.”
Quantum Copro	254.{IP}
Quantum	Slot.{IP}.ID

The Slot indicates the position occupied by the Ethernet port inside the PLC rack.

NOTICE

INVALID MODBUS SLAVES ADDRESSING

Follow the naming conventions specified above (periods, brackets, etc.) to ensure correct addressing of the requests to the serial Modbus slaves.

Failure to follow these instructions can result in equipment damage.

Outputs

Output Parameter Description

Parameter	Type	Description
Fail	BOOL	Indicates that an error has occurred while transmitting a request. The bit cannot be reset as soon as a request is issued without an error, it is set to 0. The error codes must be checked in each DFB belonging to devices or serial Modbus clients.
WantedArraySize	INT	Indicates the size necessary for the WorkMemoryRS485 array to function correctly. Since arrays start on 0, the array must be declared with the minimum size of [0..WantedArraySize-1]. If additional serial Modbus (RTU) clients or devices are added to the program, the size of this variable will increase, making it necessary to monitor the WorkMemoryRS485 Array. This variable is calculated during the first Scanning cycle so adding serial Modbus (RTU) clients or devices during “online” mode has 2 consquences: • The size of the Array will not increase. • The ModbusGateway DFB will not take into account requests from these new clients.
LostMessages	INT	Indicates the number of messages that have been lost (that the client has not received). A value ≠ 0 often comes from a CANClient instance is probably not being executed on every single cycle or a programming error. If messages are being lost on the CANopen bus timing between calls of different clients, hardware (wiring) problem should be checked.
Statistic	StatisticData	Holds a structure with statistical data on the operation of the EthernetGatewayModbus DFB. The information obtained in this data structure is the statistic for the requests managed by this EthernetGatewayModbus DFB. In other words, the requests sent by clients that are associated with this DFB by means of the WorkMemory485 Array.

This tables describes the StatisticData type:

Name	Type	Description
RequestsSended	DINT	Total number of sent requests.
RequestsOk	DINT	Total number of requests that ended correctly.

Name	Type	Description
RequestsError	DINT	Total number of requests that ended in an error
MinTime	DINT	Minimum time required to end a request (msec)
AvgTime	REAL	Average time required to end a request (msec)
MaxTime	DINT	Maximum time required to end a request (msec)
LastTime	DINT	Time required to end the last request (msec)
CurrentTime	DINT	The time spent on processing the current request (0 if there is no request in progress)
RequestsSecond	DINT	Number of requests that can be sent per second (calculated with the AvgTime value).
LastCycleNumber	DINT	Number of controller cycles that the last request took to be executed.
CurrentCycleNumber	DINT	Number of controller cycles that the current request has taken so far (0 if there is no request in progress).
TimeOnQueue	DINT	Time that the last request has been waiting in the queue before being sent (only useful for client statistics).

Inputs/Outputs

WorkMemoryRS485 Description

This Array holds the read/write requests of the clients/scanners.

The WorkMemoryRS485 Array is the work memory or link between the serial Modbus clients/scanners and an Ethernet port DFB.

During the first Scanning cycle, the devices reserve read/write %Mwords from this Array, based on the exchange words required by each DFB. The ModbusGateway DFB then counts them, and as a result specifies and knows how many pieces of data will be serialized.

NOTICE

UNEXPECTED EQUIPMENT OPERATION

The WorkMemoryRS485 variable must be adjusted previously to set:
- The size of the data exchange area between the device and client DFB
- The ModbusGateway DFB to the size needed for the elements that are being used

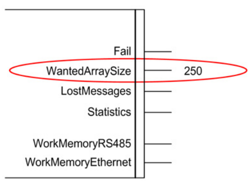
Failure to follow these instructions can result in equipment damage.

The system will be limited by fixed sizes or use memory that is not being used if this is not done.

Calculating the WorkMemoryRS485 Array Size

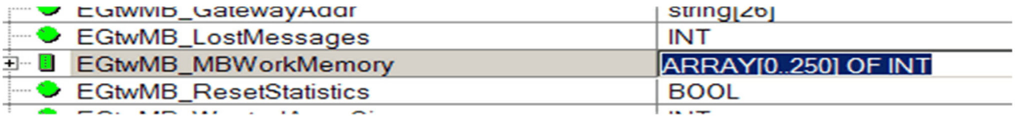
The necessary size for the array of the variable associated to WorkMemoryRS485 is automatically calculated by the DFBs of the port shown by the WantedArraySize output.

But it is necessary to check that the size of the array for the WorkMemoryRS485 variable of the ModbusGateway DFB, which has been calculated by the SGS Workbench code generator, has a size greater than or equal to the one that the WantedArraySize variable requires.



In order to check the value:

1. Execute the program with the calculated array
2. Check the value returned by the ModbusGateway DFB
3. Resize the array to the WantedArraySize value (the array must have a size of [0 .. WantedArraySize-1] as a minimum).



The DFB will rebuild the serial Modbus requests that are in WorkMemoryRS485 as Modbus TCP/IP requests, which will be copied to the WorkMemoryEthernet variable.

WorkMemoryEthernet (Array)

This variable holds the requests that are in the WorkMemoryRS485 in Modbus TCP/IP requests.

This variable must have a bigger size than the WorkMemoryRS485 variable, since each request holds more data, such as the destination IP address, etc.

NOTE: The WantedArraySize output variable only indicates the size that is required for the WorkMemoryRS485 variable. In order to know the appropriate size for the variable, check the WantedArraySize output variable of the Ethernet port DFB to which the ModbusGateway DFB is linked.

Representation as SCoD in UAG

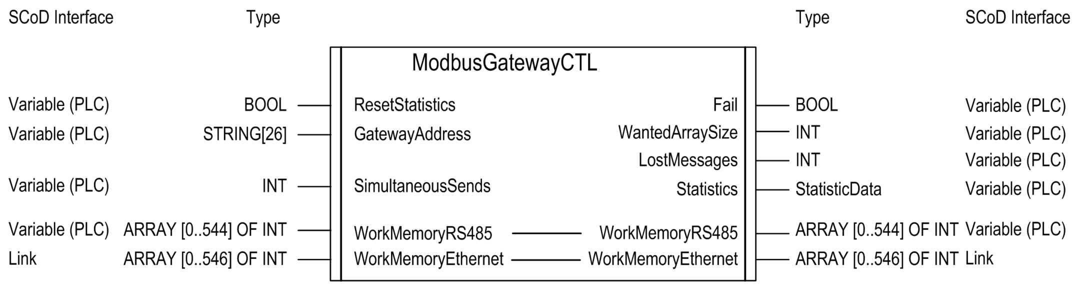
Overview

Use of the DFB pins in the SCoD editor is explained in the input/output diagram below.

The variables defined in the SCoD editor have in principle the same name as the pin (exception is that an underscore in the pin name will be skipped for the variable name).

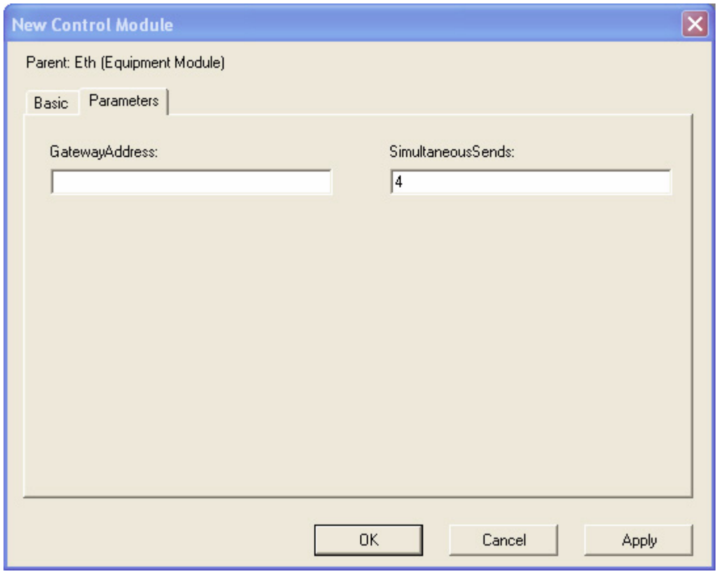
If the variables defined in the SCoD have the property 'Initial Value' then the values of the instantiated SCoD variables can be modified manually in Unity Pro before the PLC restart.

Input Output Diagram



Parameter Tab

The figure below shows the **Parameter tab** of the **Properties Control Module** window.



The following table describes the properties:

Property	Type	Description
GatewayAddress	String [26]	IP address for the gateway where the serial Modbus devices are physically located. See IP addressing table below.
SimultaneousSends	INT	Indicates how many requests are in the gateway queue.

18.4 EPortM: Ethernet Port Modicon

Overview

This section describes EPortM, the Port SCoD when using Ethernet technology with Modicon PLC.

What's in this Section?

This section contains the following topics:

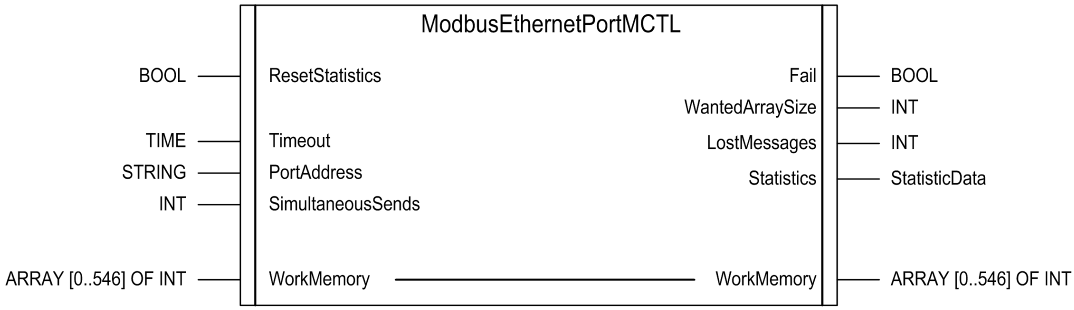
Description

General

EPortM is a Port *(see page 315)* function which is capable of serializing and managing requests sent to a physical medium working on Modbus TCP/IP.

Representation in FBD

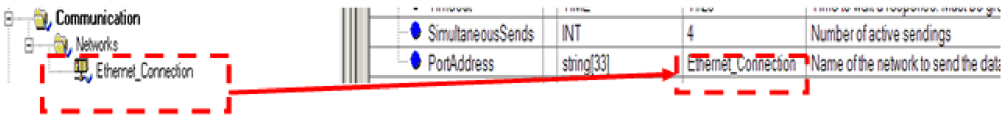
The functional module that will be used in the program has the following aspect (at the FBD block level) after PLC generation in UNITY Pro:



Inputs

Input Parameter Description

Parameter	Type	Description
Reset Statistics	BOOL	Described in CPortP Inputs. *(see page 339)*
TimeOut	TIME	Described in CPortP Inputs. *(see page 339)*
PortAddress	STRING	Defines the physical port that the DFB will use to send the Modbus TCP/IP request in Modicon M340 PLCs, to do so, write the name of the Ethernet channel through which communications will take place. (see graphic)
SimultaneousSends	INT	Described in EPortQ Inputs. *(see page 430)*



Outputs

Output Parameter Description

Parameter	Type	Description
Fail	BOOL	Described in CPortP Outputs. *(see page 340)*
WantedArraySize	INT	Described in CPortP Outputs. *(see page 340)*
LostMessages	INT	Described in CPortP Outputs. *(see page 340)*
Statistic	StatisticData	Described in CPortP Outputs. *(see page 340)*

Inputs/Outputs

WorkMemory (Array)

This Array is described in the CPortP Inputs/Outputs. *(see page 342)*

Representation as SCoD in UAG

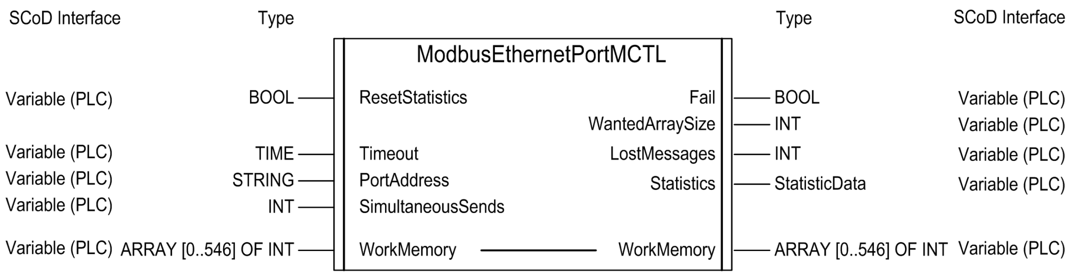
Overview

Use of the DFB pins in the SCoD editor is explained in the input/output diagram below.

The variables defined in the SCoD editor have in principle the same name as the pin (exception is that an underscore in the pin name will be skipped for the variable name).

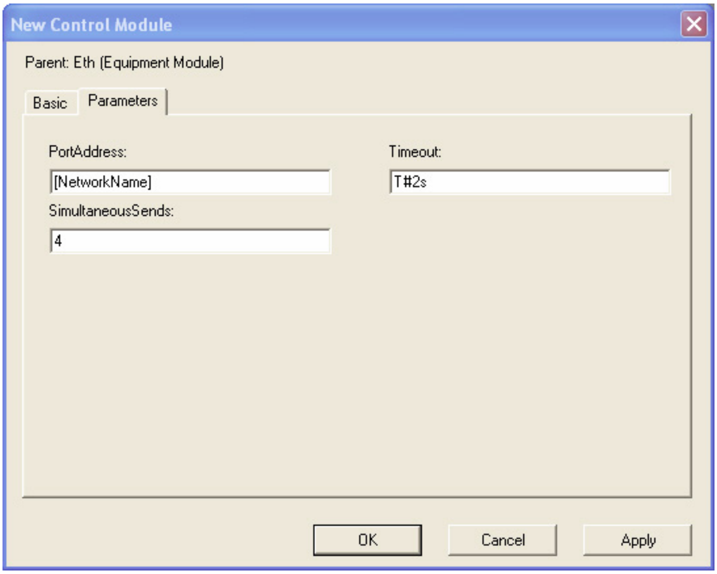
If the variables defined in the SCoD have the property 'Initial Value' then the values of the instantiated SCoD variables can be modified manually in Unity Pro before the PLC restart.

Input Output Diagram



Parameter Tab

The figure below shows the **Parameter tab** of the **Properties Control Module** window.



The following table describes the properties:

Property	Type	Description
SimultaneousSends	INT	Number of requests queued (serialized) in the Modbus Master.
PortAddress	STRING	Defines the physical port that the DFB will use to send the Modbus TCP/IP request in Modicon M340 PLCs.
Timeout	TIME	Defines the timeout value to be used in the DFB to activate the Fail output variable.

18.5 EPortP: Ethernet Port with Premium PLC

Overview

This section describes EPortP, the Port SCoD when using Ethernet technology with Premium PLC.

What's in this Section?

This section contains the following topics:

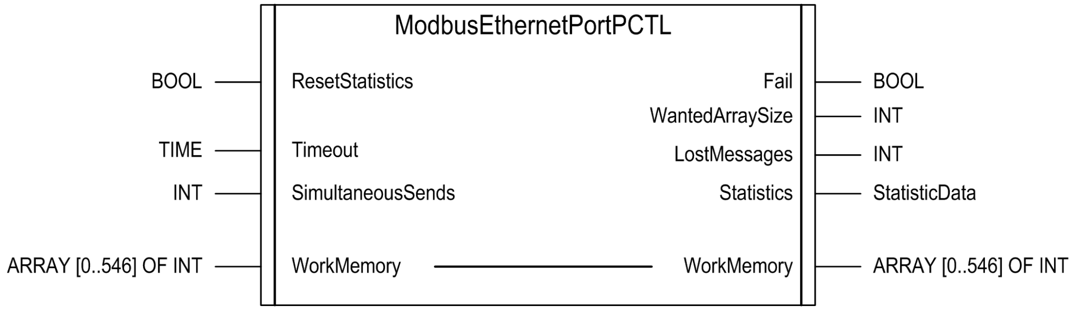
Description

General

EPortP is a Port *(see page 315)* function which is capable of serializing and managing requests sent to a physical medium working on Modbus TCP/IP.

Representation in FBD

The functional module that will be used in the program has the following aspect (at the FBD block level) after PLC generation in UNITY Pro:



Inputs

Input Parameter Description

Parameter	Type	Description
Reset Statistics	BOOL	Described in CPortP Inputs. *(see page 339)*
TimeOut	TIME	Described in CPortP Inputs. *(see page 339)*
SimultaneousSends	INT	Described in EPortQ Inputs. *(see page 430)*

Outputs

Output Parameter Description

Parameter	Type	Description
Fail	BOOL	Described in CPortP Outputs. *(see page 340)*
WantedArraySize	INT	Described in CPortP Outputs. *(see page 340)*
LostMessages	INT	Described in CPortP Outputs. *(see page 340)*
Statistic	StatisticData	Described in CPortP Outputs. *(see page 340)*

Inputs/Outputs

WorkMemory (Array)

This Array is described in the CPortP Inputs/Outputs. *(see page 342)*

Representation as SCoD in UAG

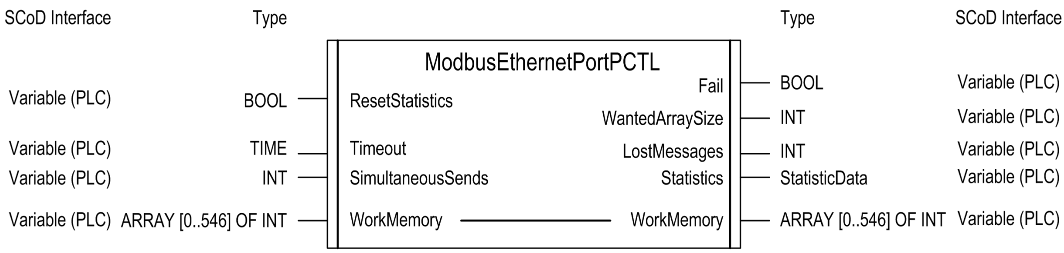
Overview

Use of the DFB pins in the SCoD editor is explained in the input/output diagram below.

The variables defined in the SCoD editor have in principle the same name as the pin (exception is that an underscore in the pin name will be skipped for the variable name).

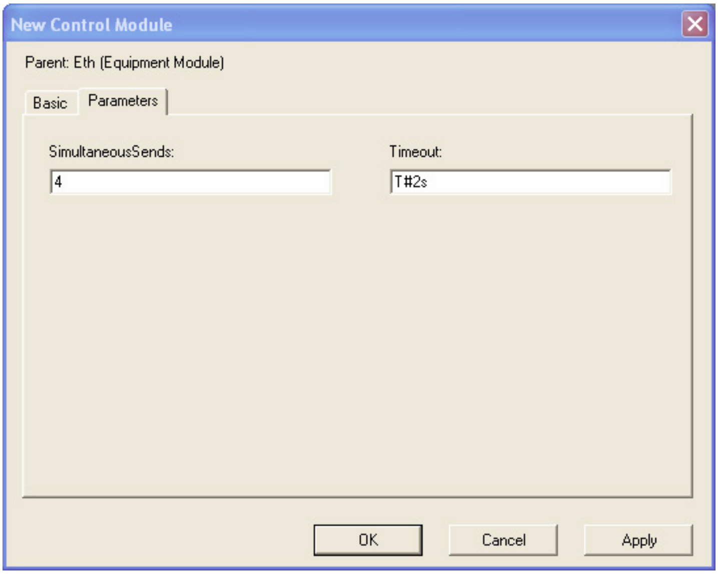
If the variables defined in the SCoD have the property ’Initial Value’ then the values of the instantiated SCoD variables can be modified manually in Unity Pro before the PLC restart.

Input Output Diagram



Parameter Tab

The figure below shows the **Parameter tab** of the **Properties Control Module** window.



The following table describes the properties:

Property	Type	Description
SimultaneousSends	INT	Number of requests queued (serialized) in the Modbus Master.
Timeout	TIME	Defines the timeout value to be used in the DFB to activate the Fail output variable.

18.6 EPortQ: Ethernet Port Quantum

Overview

This section describes EPortQ, the Port SCoD when using Ethernet technology with Quantum PLC.

What's in this Section?

This section contains the following topics:

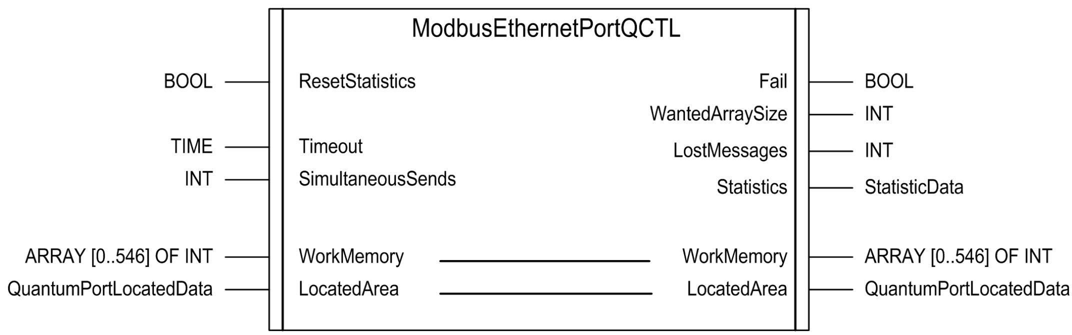
Description

General

EPortQ is a Port *(see page 315)* function which is capable of serializing and managing requests sent to a physical medium working on Modbus TCP/IP.

Representation in FBD

The functional module that will be used in the program has the following aspect (at the FBD block level) after PLC generation in UNITY Pro:



Inputs

Input Parameter Description

Parameter	Type	Description
Reset Statistics	BOOL	Described in CPortP Inputs. *(see page 339)*
TimeOut	TIME	Described in CPortP Inputs. *(see page 339)*
SimultaneousSends	INT	Indicates how many requests are in the gateway queue. The value ranges from 1 to 4. The more requests in the gateway queue, the better the system general performance, but the following features must be taken into account: ● The maximum number of requests in the CPU buffer ● The maximum number of requests for a gateway queue. ● Priority management does not take into account requests that are queued in the hardware buffer, which makes it possible for a high-priority command to have to wait until lower-priority commands that are already in the queue are sent first. This results in a longer response time for high-priority commands.

Outputs

Output Parameter Description

Parameter	Type	Description
Fail	BOOL	Described in CPortP Outputs. *(see page 340)*
WantedArraySize	INT	Described in CPortP Outputs. *(see page 340)*
LostMessages	INT	Described in CPortP Outputs. *(see page 340)*
Statistic	StatisticData	Described in CPortP Outputs. *(see page 340)*

Inputs/Outputs

WorkMemory (Array)

This Array is described in the CPortP Inputs/Outputs. *(see page 342)*

LocatedArea (QuantumPortLocatedData)

Due to internal requirements, the Quantum port needs some variables to be mapped.

This input is used to provide the port with a memory area that has the internal structure necessary for it to work. This internal data does not have to be checked and should not be modified.

It is mandatory to declare this variable (specify the variable address) in the variable table and to allocate an address to it; otherwise the port will not work.

The necessary data consists of 537 words. If, for example, the variable has been addressed on %MW100, this would be the equivalent of reserving a %MW100:537 array.

This area should not be used for any other purpose. By default, the SGS will generate a memory address that could cause an error or warning when the project is compiled.

This has been done to address the variable within the PLC working range. If this input is not mapped, error code 16#200E (Hex) will appear at level 0.

Representation as SCoD in UAG

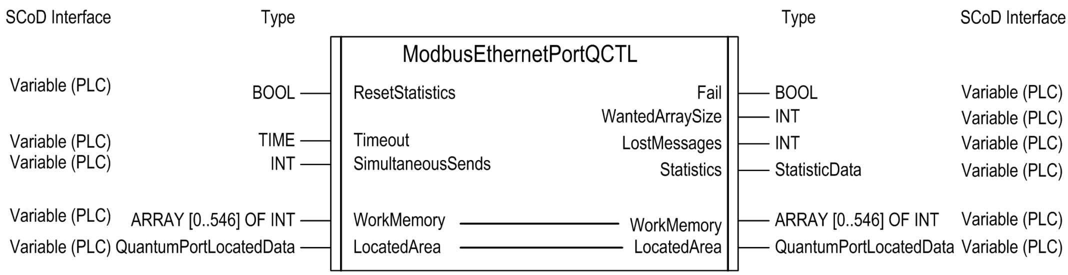
Overview

Use of the DFB pins in the SCoD editor is explained in the input/output diagram below.

The variables defined in the SCoD editor have in principle the same name as the pin (exception is that an underscore in the pin name will be skipped for the variable name).

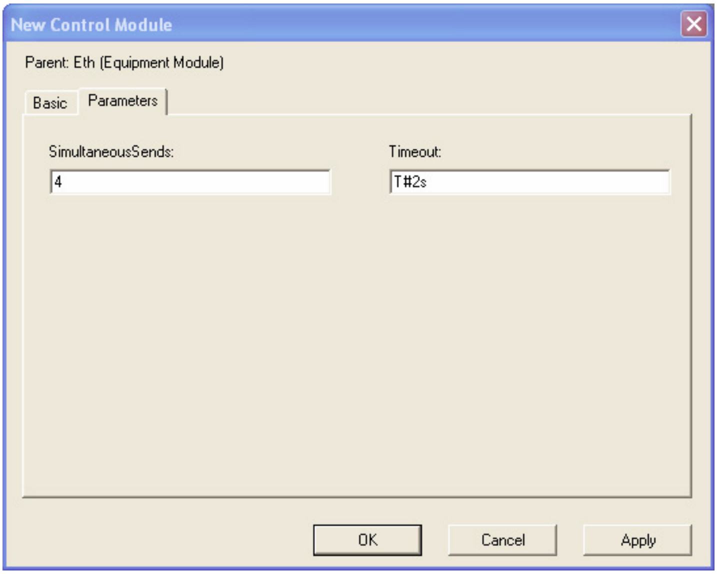
If the variables defined in the SCoD have the property 'Initial Value' then the values of the instantiated SCoD variables can be modified manually in Unity Pro before the PLC restart.

Input Output Diagram



Parameter Tab

The figure below shows the **Parameter tab** of the **Properties Control Module** window.



The following table describes the properties:

Property	Type	Description
SimultaneousSends	INT	Number of requests queued (serialized) in the Modbus Master.
Timeout	TIME	Defines the timeout value to be used in the DFB to activate the Fail output variable.

Communications Technologies

19

Overview

This chapter describes the different types of Communications Technologies.

⚠ WARNING

MISAPPLICATION OF MODULES

The modules in this section do not reflect any specific installation. Before adopting these modules for use in a specific application, the engineer must:

- conduct a safety analysis for the application and equipment installed
- verify that all modules are appropriate for the equipment or function in the installation
- supply appropriate parameters, particularly for limits
- check that all sensors and actuators are compatible with the modules selected
- thoroughly test all functions during verification and commissioning
- provide independent paths for critical control functions (emergency stop, over-limit conditions etc.) according to the safety analysis and applicable codes and regulations.

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

What's in this Chapter?

This chapter contains the following sections:

19.1 Ethernet Technology

Overview

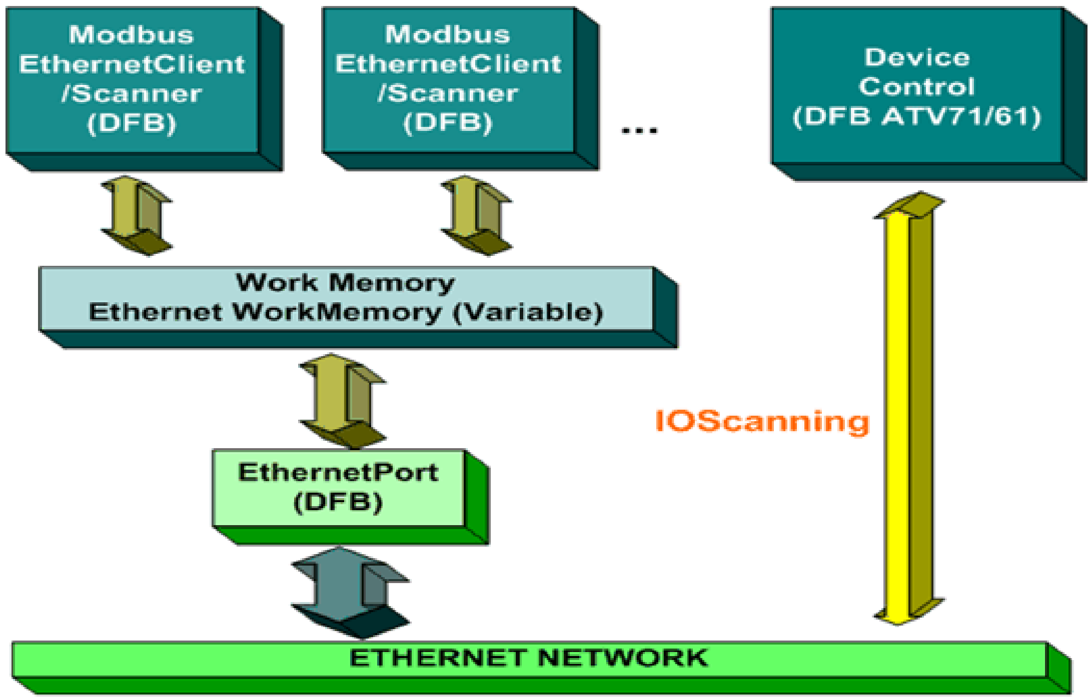
This section provides useful information regarding Ethernet Technology.

What's in this Section?

This section contains the following topics:

Communications Technologies - Ethernet

Ethernet Communications Architecture:



Description

This first part makes it possible to correctly address the EthernetClient DFB components that have to carry out non-cyclic messaging through the various EthernetPort DFB components.

Depending on the PLC platform to be used, IP addressing for Ethernet clients varies and must be handled separately. Independently of the PLC platform used, acyclic messaging requires an EthernetPort. IP addressing is implemented with the DeviceAddress input variable.

The EthernetPort DFB component carries out the Modbus TCP/IP request by means of Read_Var, Write_Var, or MBP_MSTR-type instructions in Quantum. Because of this, the clients message is sent through WorkMemory.

Depending on the platform, the following definitions apply:

Platform	IP Addressing, DeviceAddress(variable)
M340	{IP}ID
Premium	{XWAY}.Slot.Channel.ID
Quantum	Slot.{IP}.ID
Premium Copro	{XWAY}.0.0.ID
Quantum Copro	254.{IP}.ID

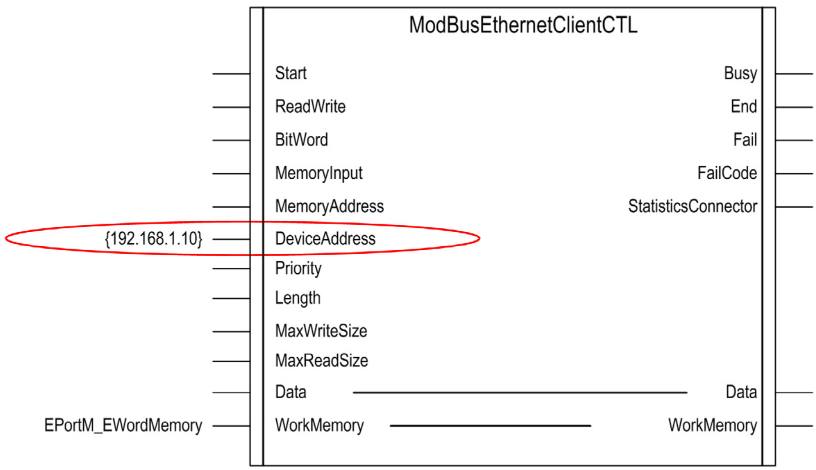
If the destination device is another CPU with an Ethernet port, the following table applies:

Platform	IP Addressing, DeviceAddress(variable)
M340	{IP}SYS
Premium	{XWAY}.SYS
Quantum	Slot.{IP}.SYS
Premium Copro	{XWAY}.SYS
Quantum Copro	254.{IP}.SYS

Addressing Example for the M340 Platform

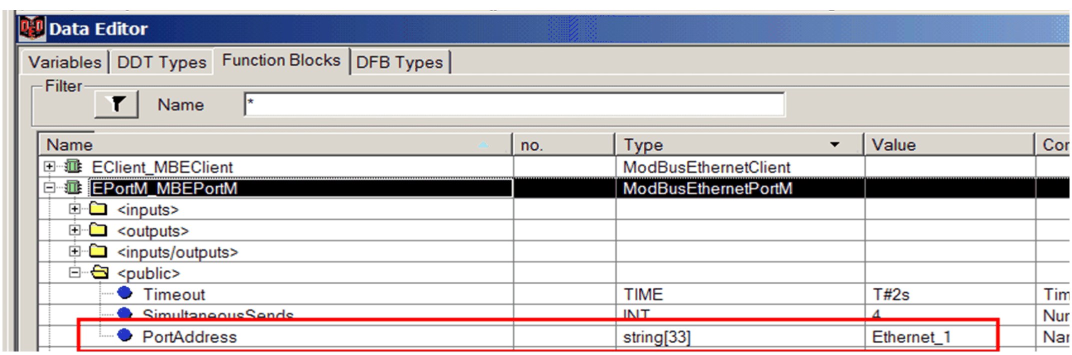
General

The current platform allows addressing with the name of the channel through which the Modbus TCP/IP requests are made. Addressing for the Ethernet client with M340 would be as in the following example (addressing for an ModBusEthernet Scanner DeviceAddress would be implemented the same way):



The ID ({IP}ID) is necessary based on the slave addressing, and ranges from 0 to 255.

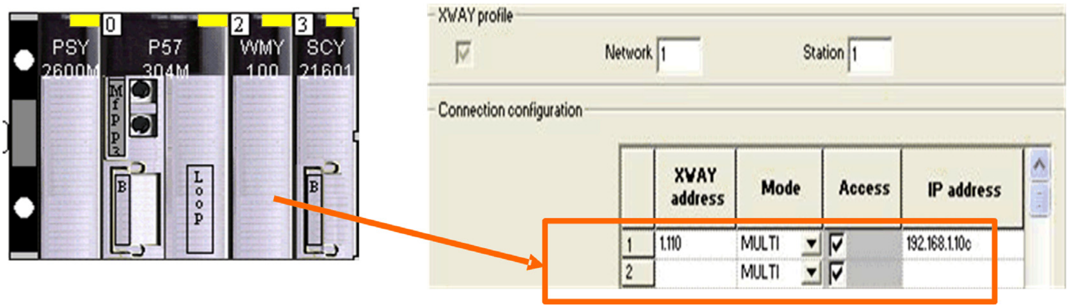
Regarding the Ethernet Port DFB component to/from which the client transmits/receives external requests with WorkMemory, the public variable must be addressed.



Addressing Example for the TSX Premium Platform

General

IP addressing for the Premium range is based on XWAY addressing, which is the service used on the PLC Ethernet ports.

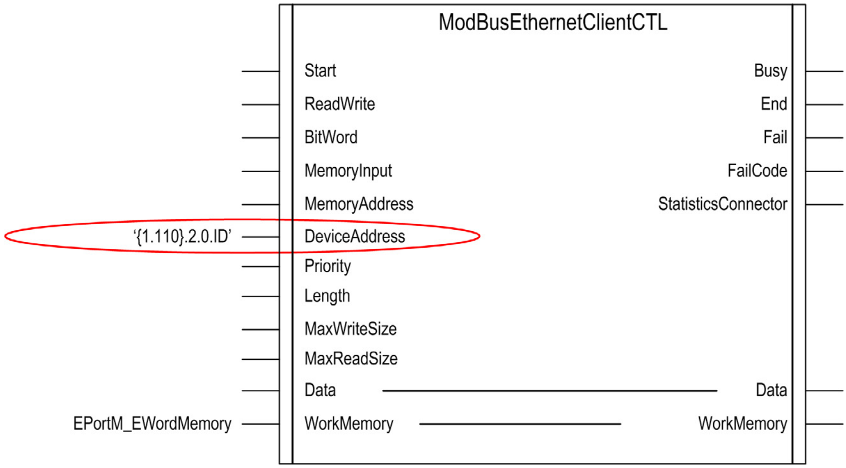


XWAY Address

An XWAY address is defined by using the Modbus TCP/IP messaging provided by the Premium Ethernet modules. An XWAY address is simply the definition of the Server device for the Ethernet module, as well as the identification of the type of protocol to be used on the Ethernet bus. The client (Premium) initiates the requests to the respective data server, and the server answers the request through the defined XWAY channel address.

An XWAY address consists of the network number and the station number (within that network) where the device for which communications should be established through Modbus TCP/IP is located (100..164 addresses).

Therefore, and as an example, an address of {1.110} is associated to the remote Modbus TCP/IP device with an address on the network (192.168.1.10 in this example). Communications are carried out through a module on Slot 2 and through channel 0.



The ID ({IP}ID) is necessary based on the slave addressing, and ranges from 0 to 255.

Addressing Example for the Quantum Platform

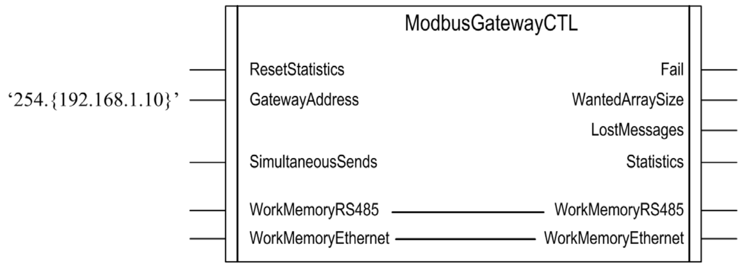
General

Quantum platforms do not have XWAY addressing, since the instructions they use for Modbus TCP/IP communications do not require this type of addressing.

Depending on the physical location of the Ethernet port for which communications are to be established.

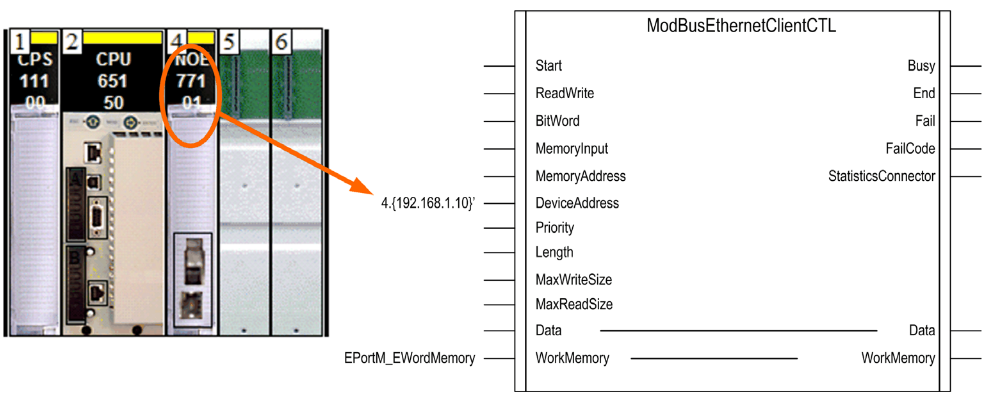
Port Located on the PLC CPU

If the Ethernet port through which communications will be carried out is the port located on the PLC CPU (not on an Ethernet communications NOE module), the address must have the header “254”.



Port Located on a NOE

If the Ethernet port with which communications will be carried out is a port located on a NOE (Quantum platform communications module), the corresponding Quantum Rack position must be indicated. Example: A NOE module located in Slot 4 of the Quantum PLC.



The ID ({IP}ID) is necessary based on the slave addressing, and ranges from 0 to 255.

19.2 Gateway Technology

Overview

This section provides useful information regarding Gateway Technology.

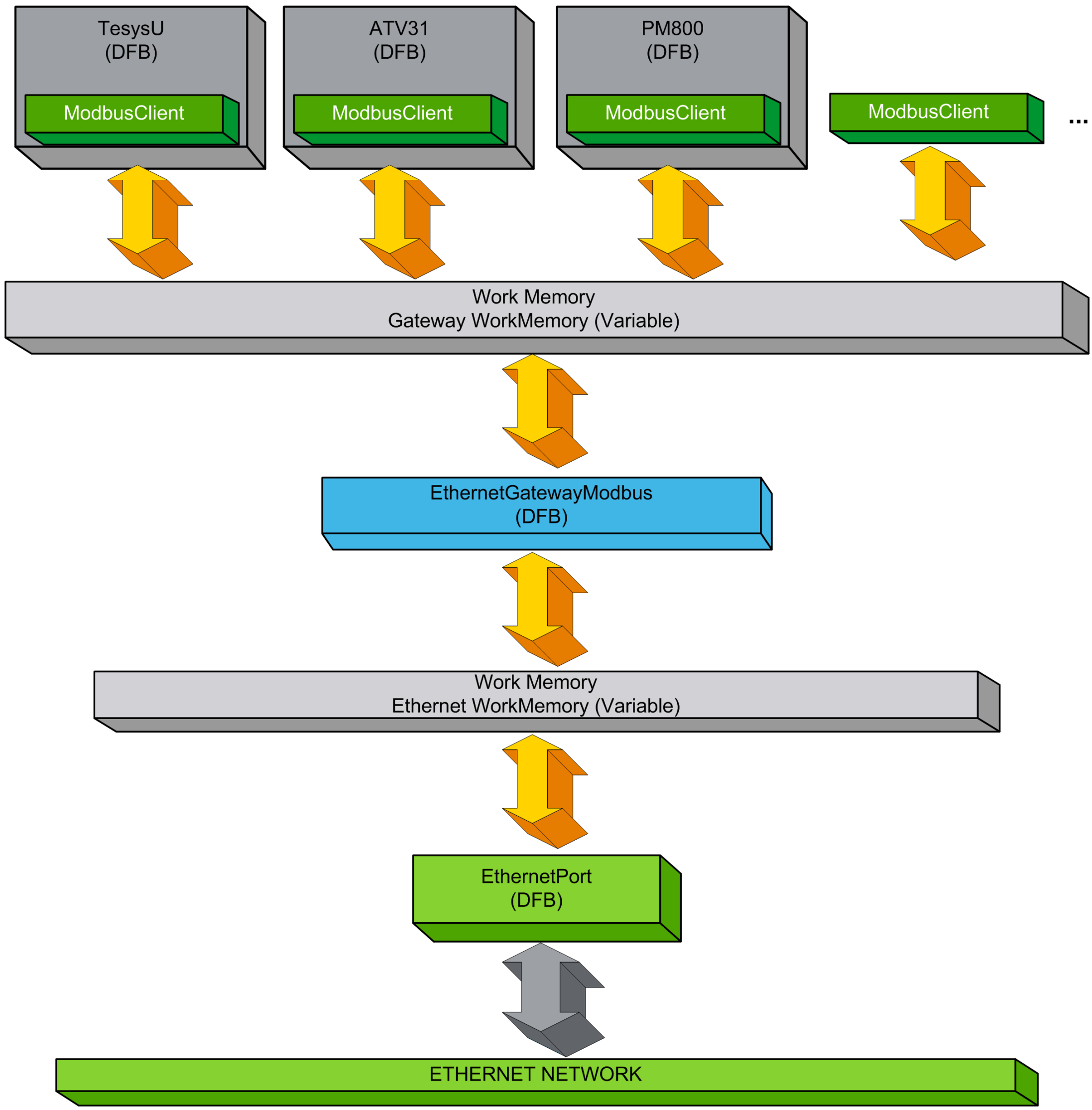
What's in this Section?

This section contains the following topics:

Communications Technologies - Gateway

Gateway Communications Architecture:

Diagram of the architecture used in communications implemented with a Serial Modbus-Modbus TCP/IP gateway:



Description

This part will make it possible to correctly address the ModbusClient/Scanner components that have to carry out non-cyclic messaging through Modbus Ethernet gateways and the various different EthernetPort instances.

Depending on the PLC platform to be used, IP addressing for Ethernet clients varies and must be handled separately. Independently of the PLC platform used, acyclic messaging requires an Ethernet gateway instance (EGtwMB). IP addressing is implemented with the DeviceAddress input variable.

The Ethernet Modbus TCP/IP-Serial Modbus gateway component receives requests from the Modbus slaves (clients/scanners) through WorkMemory RS485 and converts them into serial Modbus requests to the Ethernet port (EthernetPort).

The programmed EthernetPort instance issues the Modbus TCP/IP request by means of Read_Var, Write_Var, or MBP_MSTR-type instructions in Quantum. Because of this, the clients message is sent through EthernetWorkMemory.

Depending on the platform, the following definitions apply:

Platform	IP Addressing GatewayAddress(variable)	Client/Scanner Addressing ModbusAddress (variable)
M340	{IP}	Modbus slave number
Premium	{XWAY}.Slot.Channel	Modbus slave number
Quantum	Slot.{IP}	Modbus slave number

NOTE: Do not omit the punctuation marks.

Addressing Example for the M340 Platform

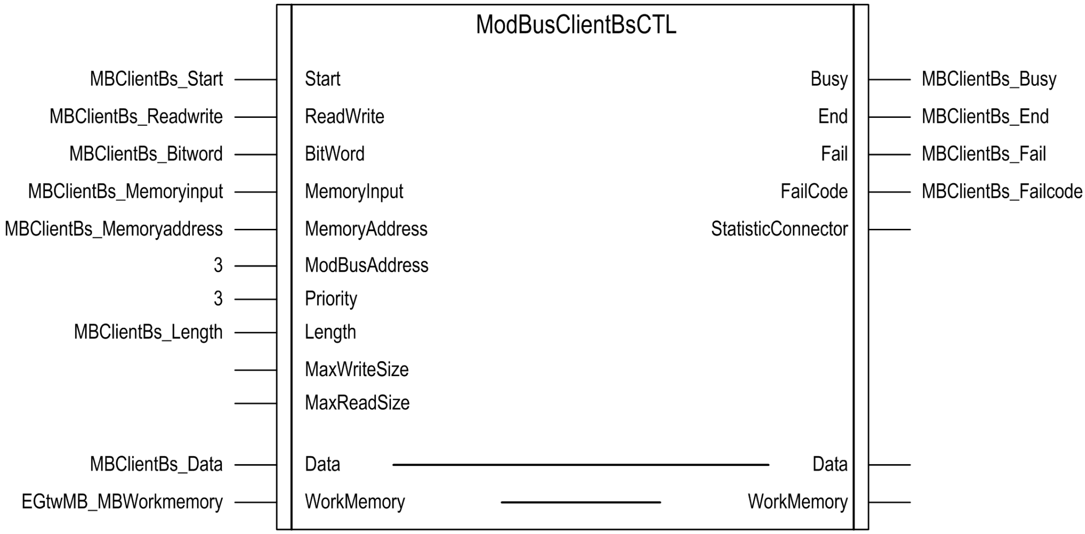
General

In this programming example, an M340 series PLC issues requests to a Modbus slave (slave number 3) through a Serial Modbus-Modbus TCP/IP gateway. This example has a client on Modbus (addressing in ModbusAddress and an MBScanner various WorkMemorys would be implemented the same way):

The name of the “EGtwMB_MBWorkMemory” variable must be the same in the “ModBusClientBsCTL” and “ModBusGatewayCTL.”

The name of the “EPortM_EWorkMemory” variable must be the same in the “ModBusGatewayCTL” and “ModBusEthernetPortPCTL”.

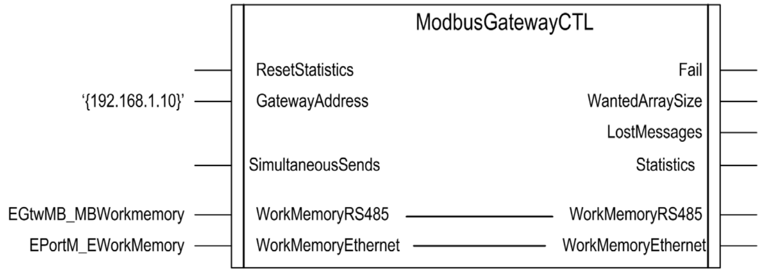
Modbus Slave with address 3



The result of the read operation once “MBClientBs_End” has been activated will be found in “MBClientBs_Data.” To carry out a write operation, the values to be written will be found in “MBClientBs_Data.”

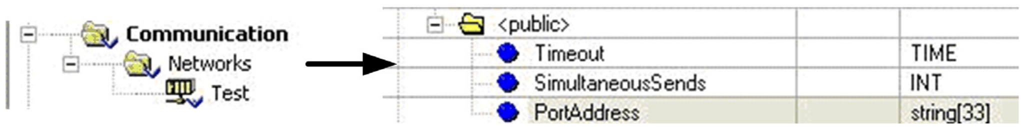
Modbus TCP/IP – Serial Modbus Gateway.

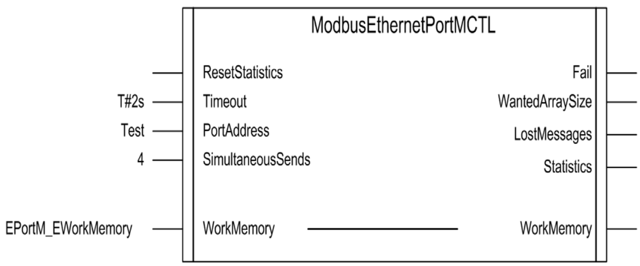
The IP address that must be entered into the Gateway Address input variable:



Ethernet port

The DFB has a input variable into which the name of the PLC configured channel (IP + services) must be provided. This is the channel through which the PLC will issue its requests to the slave:





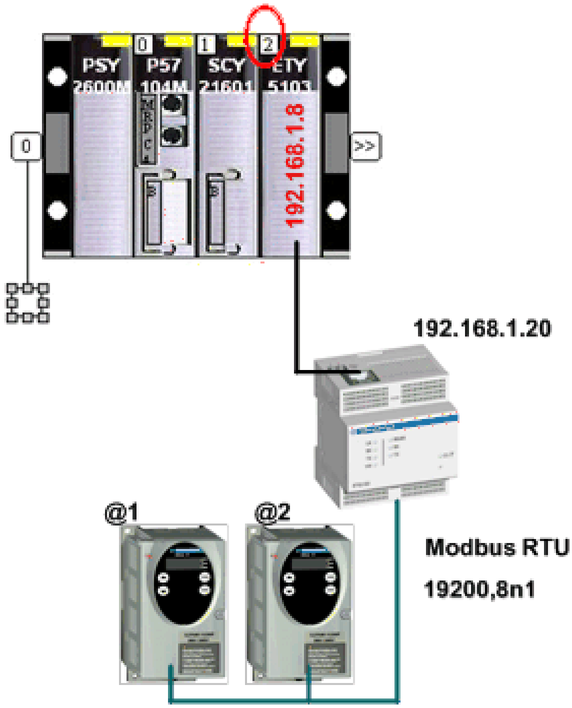
The name of the "EGtwMB_MBWorkMemory" variable must be the same in the "ModbusClientBasic" and "ModBusGateway."

The name of the "EPortM_EWorkMemory" variable must be the same in the "ModBusGateway" and "EthernetPor

Addressing Example for the TSX Premium Platform

General

The following is a practical example:

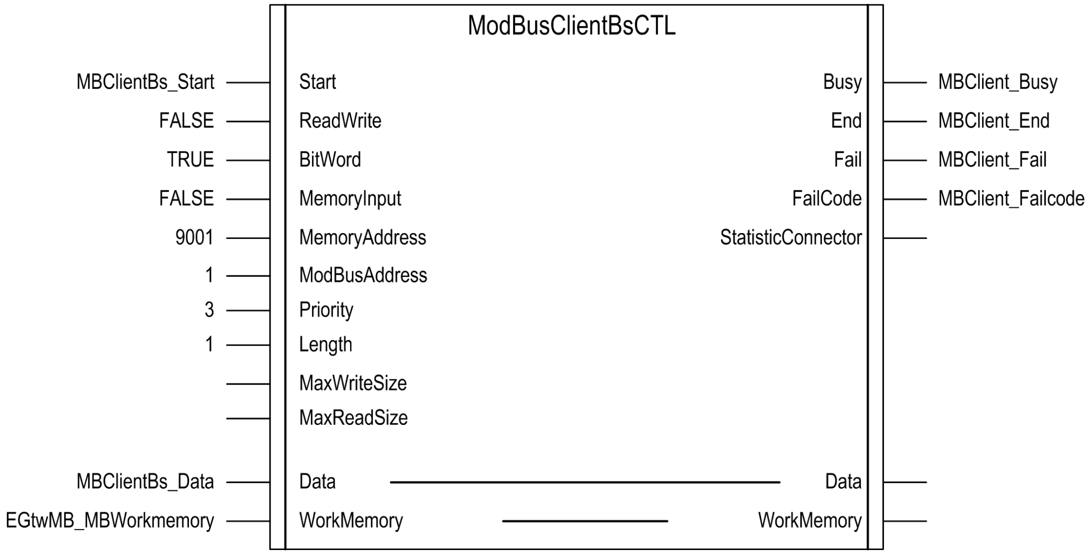


The same structure used in the previous section is used here. The objective is to establish communications with a Premium PLC towards an ATV31 Modbus slave (Modbus slave number 1) through a Modbus Ethernet-Serial Modbus gateway in order to read the Acceleration value (register 9001).

The first step is to define XWAY addressing in the PLC Ethernet module through which communication is done.

Modbus Slave: For the ATV31 speed driver with address 1

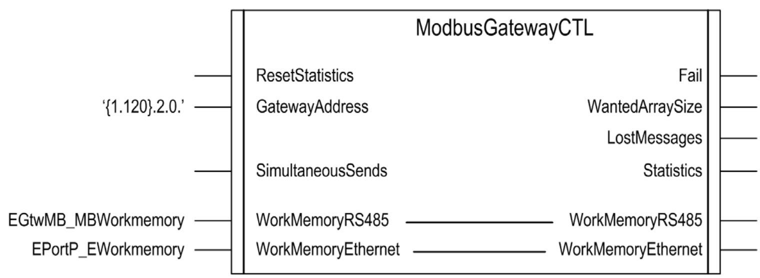
Therefore, and as an example, an address of {1.110} is associated to the remote Modbus TCP/IP device with an address on the network (192.168.1.10 in this example). Communications are carried out through a module on Slot 2 and through channel 0.



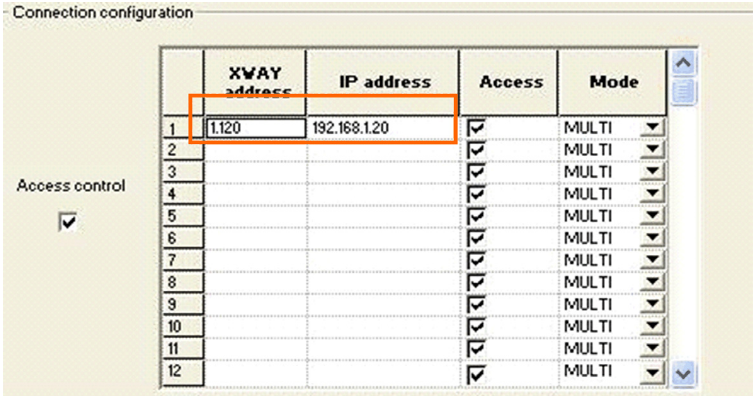
The result of the read operation once “MBClient_End” has been activated will be found in “MBClientBs_Data[0].”

Modbus TCP/IP – Serial Modbus Gateway

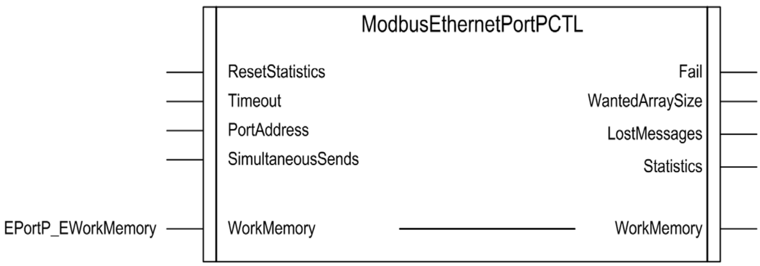
In this example, communications are carried out through the port of a Premium CPU ETY module; Slot 2, channel 0. Additionally, the XWAY address must be addressed correctly, as shown in the following example



The XWAY messaging address with which communications should take place must be edited in the PLC Ethernet port configuration:



Ethernet Port



The name of the “EGtwMB_MBWorkMemory” variable must be the same in the “ModBusClientBsCTL” and “ModBusGatewayCTL”.

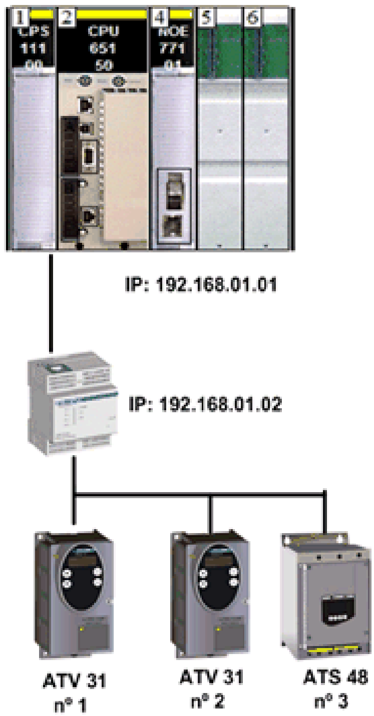
The name of the “EPort _EWorkMemory” variable must be the same in the “ModBusGatewayCTL” and “ModBusEthernetPortPCTL”.

The result of the read operation once “MBClient_End” has been activated will be found in “MBClientBs_Data[0].”

Addressing Example for the Quantum Platform

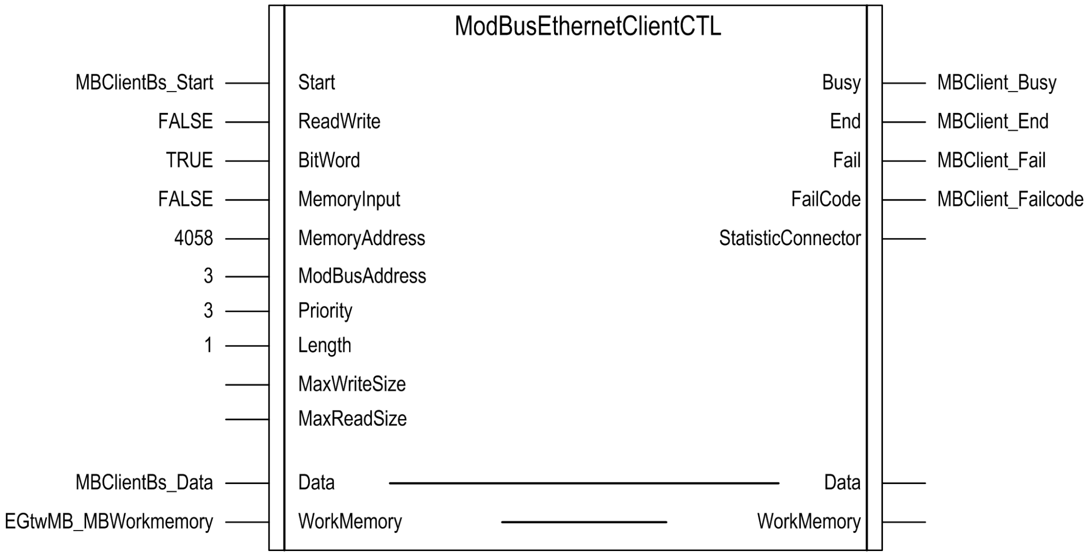
General

The following is a practical example:



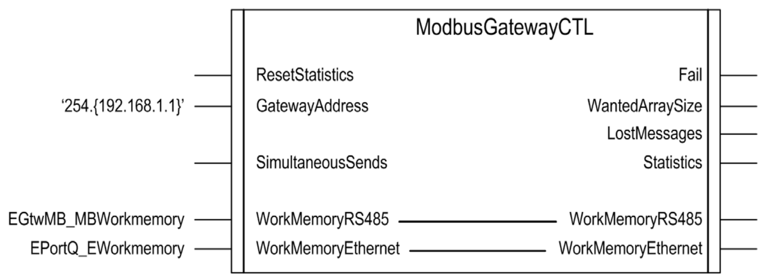
The objective is to establish communications with a Quantum PLC towards an ATS48 Modbus slave (Modbus slave number 3) through a Modbus Ethernet-Serial Modbus gateway in order to write to the cascade function activation register (register 4058).

Modbus Slave: ATS48 speed driver with address 3

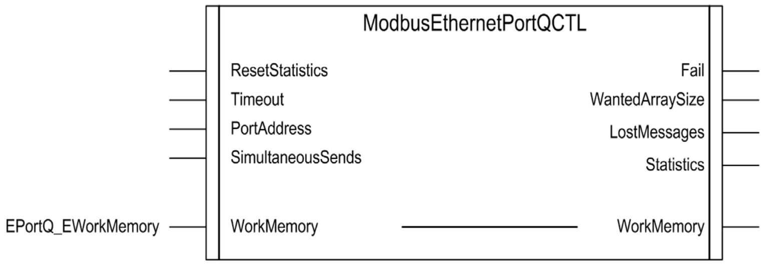


Modbus TCP/IP – Serial Modbus Gateway.

The Ethernet address must be addressed correctly. Communications will be carried out through the PLC integrated port. Because of this, the channel must be identified with a value of 254.



Ethernet Port



The name of the “EGtwMB_MBWorkMemory” variable must be the same in the “ModBusClientBsCTL” and “ModBusGatewayCTL.”

The name of the “EPort _EWorkMemory” variable must be the same in the “ModBusGatewayCTL” and “ModBusEthernetPortQCTL.”

19.3 CANOpen Technology

Overview

This section provides useful information regarding CANOpen Technology.

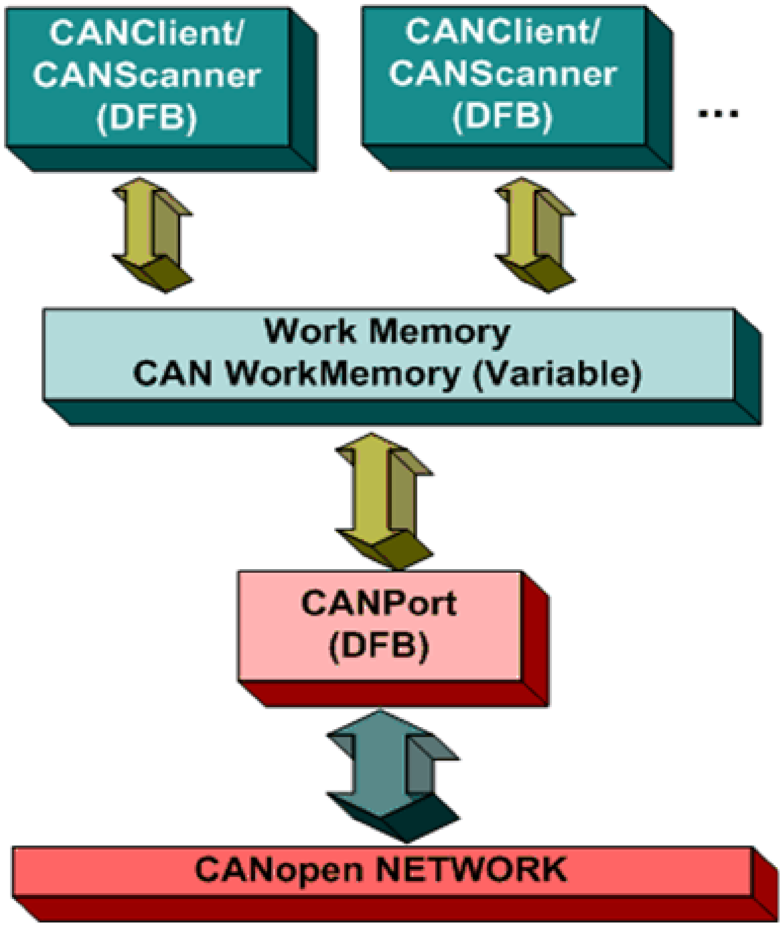
What's in this Section?

This section contains the following topics:

Communications Technologies - CANOpen

CANOpen Communications Architecture:

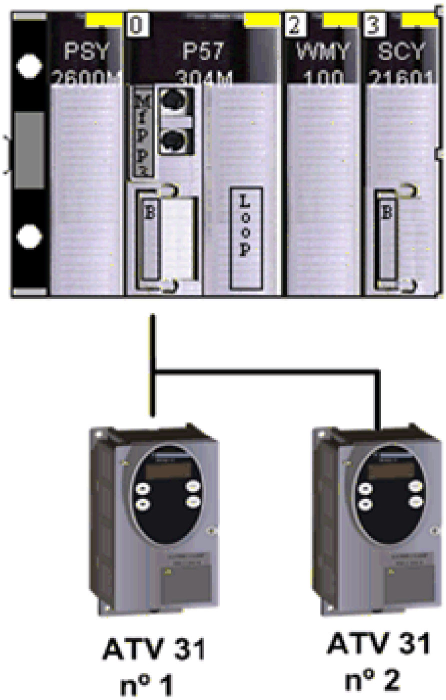
Diagram of CANopen communications architecture:



TSX Premium Platform Example

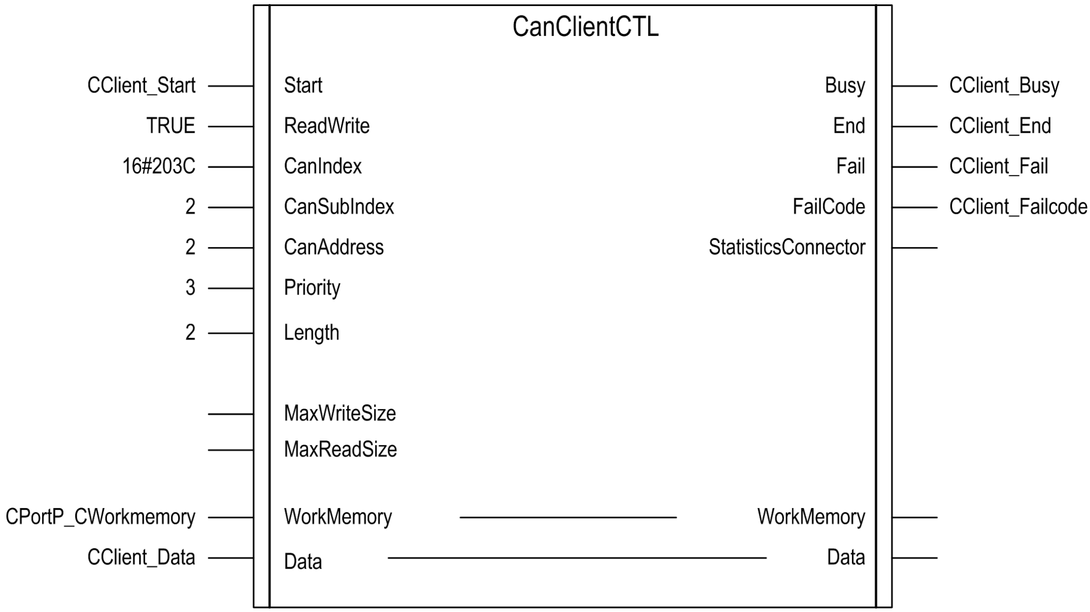
General

A Premium PLC is used in this example to write the value of the ACC (register 203C / 2 in CANopen) of the ATV31 speed driver with slave number 2.



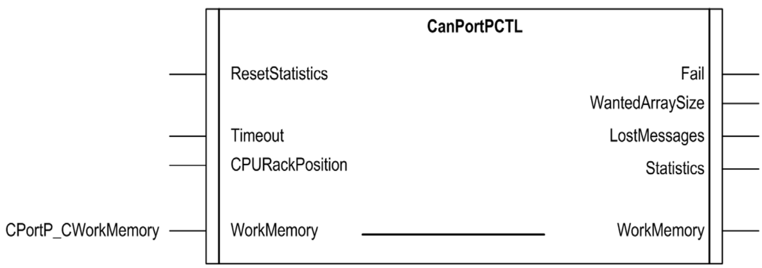
NOTE: This example can be applied to any CPU of the Modicon M340 and Premium families that can communicate on CANopen.

CANopen Client: Client with address 2



The first register of the CClient_Data Array holds the acceleration value that will be sent to the device every time that CClient_Start is TRUE and the CClient_End reading operation ends in TRUE.

CANopen Port



The name of the “CPortP_CWorkMemory” variable must be the same in the “CanClientCTL” and “CanPortPCTL”.

19.4 MODBUS Technology

Overview

This section provides useful information regarding MODBUS Technology.

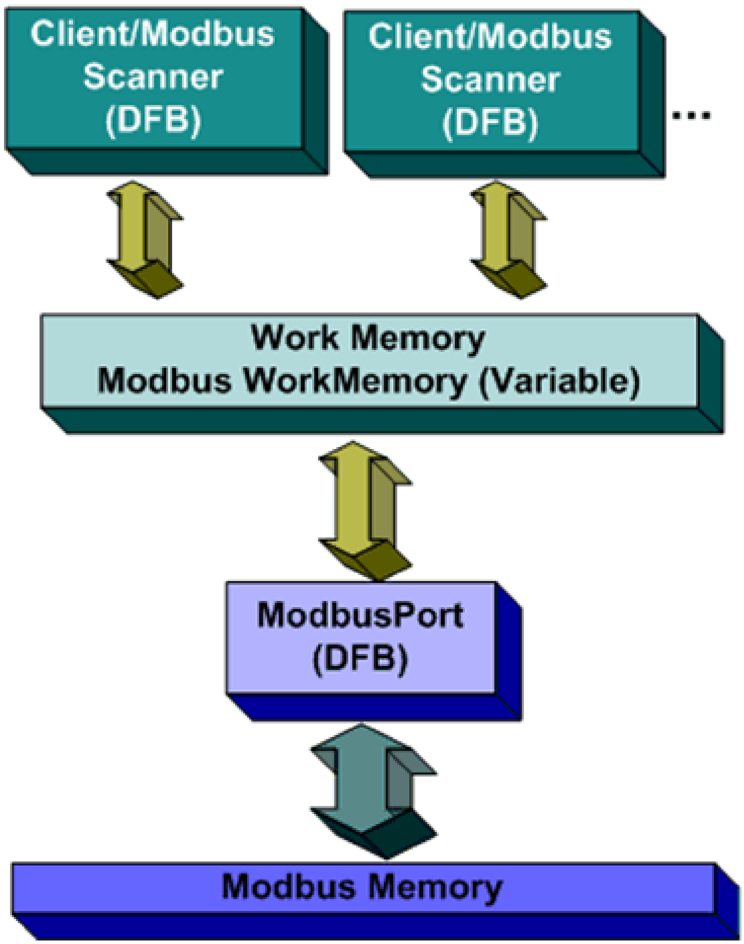
What's in this Section?

This section contains the following topics:

Communications Technologies - MODBUS

MODBUS Communications Architecture:

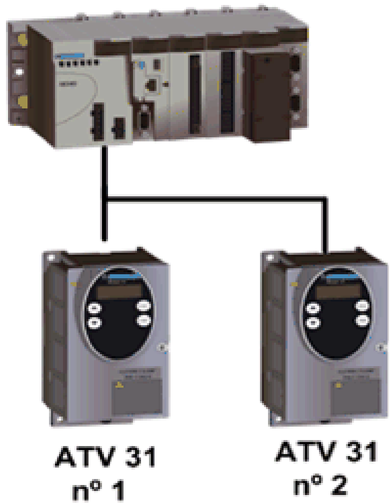
Diagram of Modbus communications architecture:



Modicon M340 Platform Example

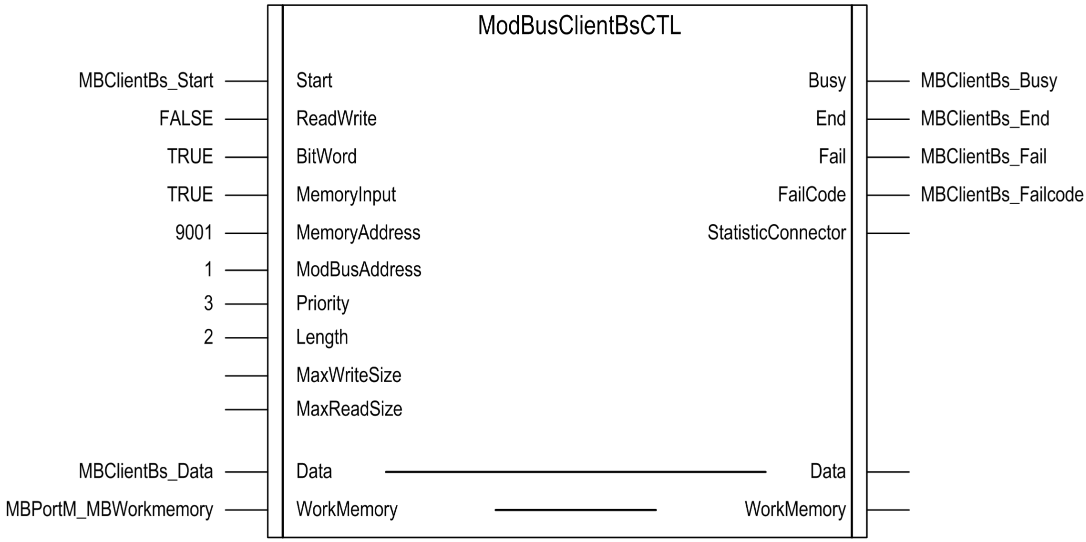
General

A Modicon M340 PLC is used in this example to read the value of the ACC and DEC (registers 9001 and 9002) of the ATV31 speed driver with slave number 1.



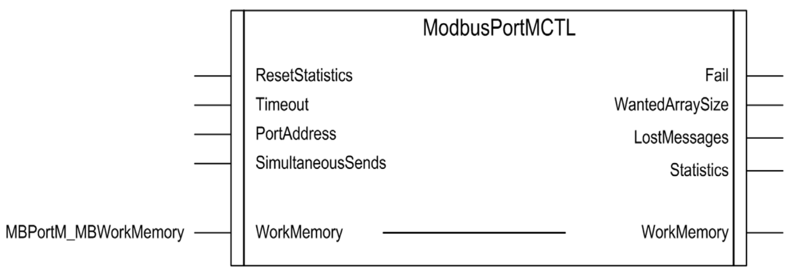
NOTE: This example can be applied to any CPU of the Modicon M340 and Premium families that can communicate on Modbus.

Modbus Client: Client with address 1



The first register of the MBClientBs_Data[0] Array holds the acceleration value, and the second register holds the deceleration read from the device every time that MBClientBs_Start is TRUE and the MBClient_End reading operation ends in TRUE.

Modbus Port



The name of the “MBPortM_MBWorkMemory” variable must be the same in the “ModBusClientBsCTL” and “ModBusPortMCTL”.

Index

G

I

M

T