

# Unity M580 Application Converter User Guide

Original instructions

06/2019

---

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

You agree not to reproduce, other than for your own personal, noncommercial use, all or part of this document on any medium whatsoever without permission of Schneider Electric, given in writing. You also agree not to establish any hypertext links to this document or its content. Schneider Electric does not grant any right or license for the personal and noncommercial use of the document or its content, except for a non-exclusive license to consult it on an "as is" basis, at your own risk. All other rights are reserved.

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.

© 2019 Schneider Electric. All rights reserved.

---

# Table of Contents

---



	<b>Safety Information</b> . . . . .	<b>5</b>
	<b>About the Book</b> . . . . .	<b>9</b>
<b>Chapter 1</b>	<b>About the Unity M580 Application Converter</b> . . . . .	<b>11</b>
	Unity M580 Application Converter . . . . .	<b>11</b>
<b>Chapter 2</b>	<b>Installation</b> . . . . .	<b>13</b>
	Installation . . . . .	<b>13</b>
<b>Chapter 3</b>	<b>UMAC Conversion Steps</b> . . . . .	<b>15</b>
	Select Screen . . . . .	<b>16</b>
	Prepare Screen . . . . .	<b>17</b>
	Analyze Screen . . . . .	<b>18</b>
	Convert Screen . . . . .	<b>19</b>
<b>Chapter 4</b>	<b>Hardware Management</b> . . . . .	<b>23</b>
4.1	Premium Hardware Configuration . . . . .	<b>24</b>
	Premium Hardware Migration . . . . .	<b>25</b>
	Retaining the Premium I/O . . . . .	<b>27</b>
4.2	Quantum Hardware Configuration . . . . .	<b>30</b>
	Quantum Hardware Migration . . . . .	<b>31</b>
	Retaining Local Quantum I/O . . . . .	<b>33</b>
	Retaining the Quantum S908 RIO . . . . .	<b>36</b>
	Retaining the Quantum EIO (Quantum and X80) . . . . .	<b>37</b>
<b>Chapter 5</b>	<b>Logic Management</b> . . . . .	<b>39</b>
	Retaining the Logic . . . . .	<b>40</b>
	Mismatched System Words and Bits . . . . .	<b>41</b>
	MAST Task Configuration . . . . .	<b>43</b>
	Missing Function Blocks . . . . .	<b>44</b>
	Unsupported Data Types . . . . .	<b>46</b>
	Address Alignment . . . . .	<b>47</b>
	Alignment in Multi-Dimensional Arrays and Data Structures . . . . .	<b>50</b>
	ADDR to ADDM Instructions . . . . .	<b>51</b>
	Ethernet I/O Scanner Translation to DTM . . . . .	<b>52</b>
	Animation Tables and Operator Screens . . . . .	<b>56</b>
	Retain Initial Values . . . . .	<b>57</b>
	Adjust Memory Allocation . . . . .	<b>59</b>

---

<b>Chapter 6</b>	<b>UMAC Reports</b> .....	<b>61</b>
	Analysis Report .....	62
	Benefit Estimation .....	63
	Conversion Report .....	64
<b>Chapter 7</b>	<b>Legacy DFBs and PLCSTAT</b> .....	<b>65</b>
	PLC Status DFB .....	65
<b>Chapter 8</b>	<b>Redundant Quantum Networks</b> .....	<b>69</b>
	Redundant Quantum Networks .....	69
<b>Appendices</b>	.....	<b>75</b>
<b>Appendix A</b>	<b>ST Sections</b> .....	<b>77</b>
	ST Sections Examples .....	78
	Detected Errors in ST Sections .....	83
<b>Glossary</b>	.....	<b>85</b>
<b>Index</b>	.....	<b>105</b>

---

# 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, service, or maintain it. The following special messages may appear throughout this documentation or on the equipment to warn of potential hazards or to call attention to information that clarifies or simplifies a procedure.



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



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

### **DANGER**

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

### **WARNING**

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

### **CAUTION**

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

### **NOTICE**

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

---

## PLEASE NOTE

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

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

## BEFORE YOU BEGIN

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

### WARNING

#### UNGUARDED EQUIPMENT

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

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

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

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

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

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

---

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

## START-UP AND TEST

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

### WARNING

#### EQUIPMENT OPERATION HAZARD

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

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

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

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

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

Before energizing equipment:

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

---

## OPERATION AND ADJUSTMENTS

The following precautions are from the NEMA Standards Publication ICS 7.1-1995 (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 describes the installation and use of the Unity M580 Application Converter (UMAC).

**NOTE:** The specific configuration settings contained in this guide are intended to be used for instructional purposes only. The settings required for your specific configuration may differ from the examples presented in this guide.

### Validity Note

This document is valid for an M580 system when used with Unity Pro XL 12.0 or later or EcoStruxure Control Expert 14.0 or later. (This document refers to the software as “Unity Pro.”)

For product compliance and environmental information (RoHS, REACH, PEP, EOL, etc.), go to [www.schneider-electric.com/green-premium](http://www.schneider-electric.com/green-premium).

The technical characteristics of the devices described in the present document also appear online. To access the information online:

Step	Action
1	Go to the Schneider Electric home page <a href="http://www.schneider-electric.com">www.schneider-electric.com</a> .
2	In the <b>Search</b> box type the reference of a product or the name of a product range. <ul style="list-style-type: none"><li>● Do not include blank spaces in the reference or product range.</li><li>● To get information on grouping similar modules, use asterisks ( * ).</li></ul>
3	If you entered a reference, go to the <b>Product Datasheets</b> search results and click on the reference that interests you. If you entered the name of a product range, go to the <b>Product Ranges</b> search results and click on the product range that interests you.
4	If more than one reference appears in the <b>Products</b> search results, click on the reference that interests you.
5	Depending on the size of your screen, you may need to scroll down to see the datasheet.
6	To save or print a datasheet as a .pdf file, click <b>Download XXX product datasheet</b> .

The characteristics that are presented in the present document should be the same as those characteristics that appear online. In line with our policy of constant improvement, we may revise content over time to improve clarity and accuracy. If you see a difference between the document and online information, use the online information as your reference.

## Related Documents

Title of Documentation	Reference Number
EcoStruxure™ Control Expert, Program Languages and Structure, Reference Manual	35006144 (English), 35006145 (French), 35006146 (German), 35013361 (Italian), 35006147 (Spanish), 35013362 (Chinese)
Modicon Quantum 140CRA31908 Adapter Module Installation and Configuration Guide	NVE78183 (English), NVE78184 (French), NVE78185 (German), NVE78186 (Italian), NVE78187 (Spanish), NVE78188 (Chinese)
Modicon M580, RIO Modules, Installation and Configuration Guide	EIO0000001584 (English), EIO0000001585 (French), EIO0000001586 (German), EIO0000001587 (Italian), EIO0000001588 (Spanish), EIO0000001589 (Chinese),
Modicon M580 BMENOC0301/11, Ethernet Communication Module, Installation and Configuration Guide	HRB62665 (English), HRB65311 (French), HRB65313 (German), HRB65314 (Italian), HRB65315 (Spanish), HRB65316 (Chinese)
Modicon M580 Hot Standby, System Planning Guide for Frequently Used Architectures	NHA58880 (English), NHA58881 (French), NHA58882 (German), NHA58883 (Italian), NHA58884 (Spanish), NHA58885 (Chinese)
Modicon M580, Hardware, Reference Manual	EIO0000001578 (English), EIO0000001579 (French), EIO0000001580 (German), EIO0000001582 (Italian), EIO0000001581 (Spanish), EIO0000001583 (Chinese)
Modicon M580 Standalone, System Planning Guide for Frequently Used Architectures	HRB62666 (English), HRB65318 (French), HRB65319 (German), HRB65320 (Italian), HRB65321 (Spanish), HRB65322 (Chinese)
Modicon M580 Standalone, System Planning Guide for Complex Topologies	NHA58892 (English), NHA58893 (French), NHA58894 (German), NHA58895 (Italian), NHA58896 (Spanish), NHA58897 (Chinese)

You can download these technical publications and other technical information from our website at <https://www.schneider-electric.com/en/download>

---

# Chapter 1

## About the Unity M580 Application Converter

---

### Unity M580 Application Converter

#### Introduction

Users of the Premium and Quantum lines of CPUs and I/O modules are transitioning to PlantStruxure Ethernet architectures that use M580 CPUs. The Unity M580 Application Converter (UMAC) converts Premium and Quantum applications for use in this new architecture.

#### WARNING

##### UNEXPECTED APPLICATION BEHAVIOR

- Perform a complete validation of the converted application in a simulated environment before you deploy a production system.
- Ensure that the converted application works as designed.

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

**NOTE:** The UMAC tool was developed for use with Unity Pro configuration software (up to version 13.1) and Control Expert configuration software (version 14.0 and later). This document refers to the software as “Unity Pro.”

#### Conversion Types

You can convert these types of Premium and Quantum projects for use in M580 Ethernet I/O architectures:

Source	Program Type
Premium	Premium application from Unity
	Premium application from PL7
Quantum	Quantum application from Unity
	Quantum application from a non-Unity project
	M580 (with Quantum Configuration) Unity To M580 Unity

**NOTE:** Refer to the description of the **Select** screen ([see page 16](#)).

#### Features by edition

The UMAC tool is available in LITE and STANDARD editions.

This table shows the features that are available for different editions of the UMAC tool:

Feature	LITE	STANDARD
Retain Premium I/O	X	X
Retain Local Quantum I/O	X	X
Retain Quantum S908 RIO (Quantum, S800 And SyMax)	X	X
Retain Quantum EIO (Quantum And X80 )	X	X
Retain Logic	X	X
Identify Mismatched System Words and Bits	X	X
CPU type selection	X	X
MAST task configuration (Periodic or Cyclic)	X	X
Hardware Configuration <ul style="list-style-type: none"> <li>● Replace Quantum I/O with X80 I/O</li> <li>● Replace 800 Series I/O with X80 I/O</li> </ul>		X
Select All (Recommended)		X
Replace Missing FFBs (PL7-3, PL7, Date and Time functions)		X
Mode: <ul style="list-style-type: none"> <li>● FLIP-FLOP</li> <li>● On The Fly (Premium only)</li> </ul>		X
Replace Unsupported Data Types (%MD, %KD, %MF and %KF)		X
Fix Alignment Constraints (32-bit data types starting on odd %MWs)		X
Fix Alignment in Multi-Dimensional Arrays and Data Structures		X
Replace ADDR instruction with ADDM		X
Translate Ethernet I/O scanner to DTM		X
NOC selection		X
Retain Animation Tables and Operator Screens		X
Retain Initial Values		X
Adjust Memory Allocation		X

**NOTE:** Refer to the description of features ([see page 19](#)) for the LITE and STANDARD editions of the UMAC tool.

---

# Chapter 2

## Installation

---

### Installation

#### System Requirements

Observe these software and hardware requirements to install the UMAC tool:

Type	Feature	Requirement
software	Unity Pro XL 12.0 or later	installed
	Windows 7 or later	<ul style="list-style-type: none"><li>● 32-bit Professional</li><li>● 64-bit Professional</li><li>● Ultimate</li></ul>
	Microsoft Excel	These software packages allow you to view spreadsheets and reports from the tool.
	PDF reader (for example, Acrobat Reader)	
	screen resolution (point size at 100%)	1024 x 768 pixels (minimum)
hardware	RAM	2GB (minimum)
	available hard drive space	1GB (minimum), 2GB (recommended)
	<b>NOTE:</b> Your operating system dictates the hardware requirements.	

#### Install the UMAC Tool

If you select the **LITE** edition to test the UMAC tool, you can activate the **STANDARD** edition of the tool later.

Install the UMAC tool:

Step	Action
1	Download the UMAC application (.zip format) from the appropriate website to your hard drive.
2	Unzip the downloaded file and open the folder inside.
3	Double-click <b>Setup.exe</b> .
4	Read the <b>Release Notes</b> and click <b>Next</b> .
5	Read the <b>EULA</b> , select <b>I accept</b> , and click <b>Next</b> .
6	Enter the <b>Customer Information</b> fields and click <b>Next</b> .
7	Assign a <b>Destination</b> folder (or accept the default).
8	Click <b>Install</b> to install the UMAC tool and License Manager.
9	Click <b>Finish</b> to complete the installation.

## Launch the UMAC Tool

Launch the UMAC tool for the first time:

Step	Action
1	Double-click the UMAC icon that appeared on your desktop after the installation.
2	Enter text fields with name and company.
3	Enter an email address that is associated with your Schneider Electric account.

You can now use the **LITE** edition of the UMAC tool. To use the enhanced features in the **STANDARD** edition of the tool, continue to the next set of instructions.

**NOTE:** Before you select an edition of the UMAC tool, refer to the descriptions of the **LITE** and **STANDARD** editions.

## Activate the STANDARD Edition License

Use an **Activation ID** to register and run the **STANDARD** edition of the UMAC tool:

Step	Action
1	Establish a connection to the internet.
2	Double-click the UMAC icon on your PC desktop, select the <b>About</b> menu item, and click <b>Upgrade License</b> .
3	Press the <b>Upgrade license</b> button.
4	Select <b>Activate new license</b> and click <b>Next</b> .
5	Select <b>By web</b> and click <b>Next</b> .
6	Enter the <b>Activation ID</b> that was provided at the time of purchase and click <b>Next</b> .
7	Enter your email address and click <b>Next</b> .
8	Wait for the server to activate the license and click <b>Finish</b> .

You can now use the **STANDARD** edition of the UMAC tool.

---

# Chapter 3

## UMAC Conversion Steps

---

### Introduction

This chapter describes the steps in the conversion when you use the Unity M580 Application Converter (UMAC).

To access these pages, press the **Start** button on the UMAC home page.

### What Is in This Chapter?

This chapter contains the following topics:

Topic	Page
Select Screen	16
Prepare Screen	17
Analyze Screen	18
Convert Screen	19

## Select Screen

### Select an Application for Conversion

Use the radio buttons to select a conversion:

Step	Action
1	Select a radio button: <ul style="list-style-type: none"><li>● Premium application from Unity</li><li>● Premium application from PL7</li><li>● Quantum application from Unity</li><li>● Quantum application from a non-Unity project</li><li>● M580 (with Quantum Configuration) Unity To M580 Unity</li></ul>
2	Press the <b>Next</b> button.



## Prepare Screen

### Introduction

Follow the instructions on the **Prepare** screen to prepare your application for the conversion.

The instructions on this page are specific to the choice you made on the **Select** screen (*see page 16*).

**NOTE:** Press the **Back** button at any time to return to the previous screen.

### File Formats

The UMAC tool converts only source projects that are in the .xef or .zef file formats.

You may have to open a legacy project in Unity and export it to one of those formats before you can convert the application.

### Rack Addresses (Premium Projects)

For Premium projects that have local I/O points on rack 0, physically move the I/O modules to another rack to retain the topological addresses in the converted M580 application.

### Continue

Press the **Next** button to proceed to the next screen.

# Analyze Screen

## Analyze the Project

Use the tools on the **Analyze** screen to analyze the project before the conversion:

Field	Button	Description
Source Application	Open	Locate and open the Unity project you want to convert (.zef or .xef).
Analyze	Analysis Report (.pdf)	A report in this format shows the items that the converter handles and the items that require additional work after the conversion.
	Analysis Report (.html)	A report in this format shows the items that the converter handles and the items that require additional work after the conversion.  <b>NOTE:</b> You can copy and paste from the .html format to Microsoft Word documents.
	Benefit Estimation	A spreadsheet shows the manual effort that was saved by using the converter.

**NOTE:** Press the **Back** button at any time to return to the previous screen. Pressing this button deletes analysis and reports.

## Open the Project

Locate and open the source file for the conversion:

Step	Action
1	Press the <b>Open</b> button.
2	Use standard operating system commands to drive to and open the .xef or .zef source file.
3	Wait for the analysis of the source file to finish and press the <b>OK</b> button.
4	Press the <b>Next</b> button to continue the conversion.

## Unity Pro Compatibility

The UMAC tool may not complete the analyze phase if the source .xef or .zef file was generated with an earlier version of Unity Pro. In this case, regenerate the source file:

Step	Action
1	Open the source project in Unity Pro XL 12.0 or later.
2	Generate a new file (.xef or .zef).
3	Repeat the conversion with the UMAC tool.

## Convert Screen

### Introduction

Use the **Convert** page to execute the conversion.

The conversion options that are available on this screen correspond to the edition of the UMAC tool that you have installed.

### Conversion Features

Use the tools on this screen to convert the project with any edition of the UMAC tool. These features are selected (checked) by default:

Feature	Description
<b>Retain Premium I/O</b>	<p>The converted M580 project retains the configuration of the local Premium I/O, including all local I/O from logical racks (racks 0 ... 7) and any attached extension racks.</p> <p><b>NOTE:</b></p> <ul style="list-style-type: none"> <li>● If you do not want to retain the Premium I/O, manually replace the Premium I/O with X80 I/O modules after the conversion.</li> <li>● Refer to the discussion of retaining the Premium I/O. (<a href="#">see page 27</a>)</li> </ul>
<b>Retain Local Quantum I/O</b>	<p>The converted M580 project retains the configuration of the local Quantum I/O as an Ethernet I/O drop by physically replacing the CPU with a 140CRA31200 module in the hardware configuration.</p> <p><b>NOTE:</b></p> <ul style="list-style-type: none"> <li>● If you do not retain the Quantum I/O, manually replace the Quantum I/O with X80 I/O modules after the conversion.</li> <li>● Refer to the discussion of retaining the local Quantum I/O (<a href="#">see page 33</a>).</li> </ul>
<b>Retain Quantum S908 RIO (Quantum, S800 And SyMax)</b>	<p>The converted M580 project retains the configuration of the drops on an S908 network by replacing the CPU with the 140CRA31908 adapter module. This makes the legacy local rack an EIO drop that can communicate with the S908 head 140CRP93-00.</p> <p><b>NOTE:</b></p> <ul style="list-style-type: none"> <li>● For this new I/O configuration, replace the Quantum CPU with a 140CRA31908 adapter module.</li> <li>● Refer to the discussion of retaining the Quantum S908 RIO (<a href="#">see page 36</a>).</li> </ul>
<b>Retain Quantum EIO (Quantum And X80)</b>	<p>The converted M580 project retains the configuration of the Quantum and X80 drops on an Ethernet I/O network.</p> <p><b>NOTE:</b></p> <ul style="list-style-type: none"> <li>● For this new I/O configuration, the network connects to the Device Network ports of the M580 P58-040 CPU.</li> <li>● Refer to the discussion of retaining the Quantum EIO (<a href="#">see page 37</a>).</li> </ul>
<b>Retain Logic</b>	<p>The converted M580 project retains the logic and variables from the source Premium or Quantum project.</p> <p><b>NOTE:</b></p> <ul style="list-style-type: none"> <li>● In this case, functions and function blocks from obsolete Premium and Quantum libraries are not retained in the <b>LITE</b> edition.</li> <li>● Refer to the discussion of retaining the logic (<a href="#">see page 40</a>).</li> </ul>
<b>Identify Mismatched System Words and Bits</b>	<p>For the Premium and Quantum system words and bits that do not correspond to those in M580 configurations, the UMAC tool inserts a conversion error into the ST sections of Unity Pro logic (<a href="#">see page 83</a>). All matching and non-matching occurrences are displayed in separate tables in the conversion report.</p> <p><b>NOTE:</b> Refer to the discussion of system words and bits identification (<a href="#">see page 41</a>).</p>
<b>CPU type selection</b>	<p>In the <b>CPU type selection</b> pull-down menu, make a selection from the list of CPUs that are available for your application.</p> <p><b>NOTE:</b> The list of available CPUs corresponds to the architecture of the converted project.</p>
<b>MAST task configuration</b>	<p>Configure the MAST task execution (<b>Periodic</b> or <b>Cyclic</b>).</p> <p>Refer to the discussion of the MAST task configuration options (<a href="#">see page 43</a>).</p>

If you have the **STANDARD** edition of the UMAC tool, you can use these additional conversion options. (These items are not available in the **LITE** edition of the UMAC tool):

Feature	Description
<b>Hardware Configuration</b>	Press the <b>Select drop(s)</b> button to configure the migrations of the Premium or Quantum racks in the source application: <ul style="list-style-type: none"> <li>● Change the Quantum configuration. (<a href="#">see page 30</a>)</li> <li>● Change the Premium configuration. (<a href="#">see page 27</a>)</li> </ul>
<b>Select All</b>	Select all features in this list (selected by default).
<b>Replace Missing FFBs</b>	The UMAC tool converts the Premium and Quantum instructions in the obsolete libraries to new DFBs that replicate the same functionality in the target M580 project. <b>NOTE:</b> Refer to the discussion of missing function blocks ( <a href="#">see page 44</a> ).
<b>FLIP-FLOP, On The Fly</b>	Use the radio buttons to select an address constraint strategy: <ul style="list-style-type: none"> <li>● FLIP-FLOP (<a href="#">see page 49</a>) (default for Quantum or Premium): The UMAC tool places the necessary PULL-PUSH statements into the ST sections (<a href="#">see page 81</a>) at the beginning and end of each task.</li> <li>● On The Fly (<a href="#">see page 48</a>) (Premium only): The UMAC tool inserts the necessary PULL-PUSH statements directly before and after they are required in logic. (The <b>On The Fly</b> radio button is not available for Quantum applications.)</li> </ul>
<b>Replace Unsupported Data Types</b>	The UMAC tool replaces the Premium-specific data types with those that M580 applications support. <b>NOTE:</b> Refer to the discussion of unsupported data types ( <a href="#">see page 46</a> ).
<b>Fix Alignment Constraints</b>	Assign Premium data types that are not supported in M580 to unlocated variables that are assigned to the same memory location in the converted M580 project. <b>NOTE:</b> Refer to the discussion of address alignment ( <a href="#">see page 47</a> ).
<b>Fix Alignment in Multi-Dimensional Arrays and Data Structures</b>	Some one- and two-dimensional arrays and simple data structures in the source Premium project contain 32- or 64-bit data types that are assigned to odd memory words. The UMAC tool reassigns these data types to unlocated variables in the converted M580 project and inserts the appropriate PUSH-PULL DFBs according to the selected strategy. <b>NOTE:</b> Refer to the discussion of alignment in multi-dimensional arrays and data structures ( <a href="#">see page 50</a> ).
<b>Replace ADDR instruction with ADDM</b>	The UMAC tool directly reassigns ADDR instructions in the source Premium project to ADDM instructions in the target M580 project. <b>NOTE:</b> Refer to the discussion of converting ADDR instructions to ADDM ( <a href="#">see page 51</a> ).
<b>Translate Ethernet I/O scanner to DTM</b>	The UMAC tools inserts direct assignments in logic between the legacy memory locations of the Ethernet I/O scanner table and the newly created DDT in the target M580 project. <b>NOTE:</b> Refer to the discussion for Ethernet I/O scanner translation to DTM ( <a href="#">see page 52</a> ).
<b>Retain Animation Tables and Operator Screens</b>	Animation tables and operator screens in Premium and Quantum projects appear in the converted M580 project. <b>NOTE:</b> <ul style="list-style-type: none"> <li>● In this case, data types that M580 projects do not support are assigned new variable names within the table or screen.</li> <li>● Refer to the discussion of animation tables and operator screens (<a href="#">see page 56</a>).</li> </ul>
<b>Retain Initial Values</b>	The UMAC tool retains the initial values assigned in the source application. <b>NOTE:</b> Refer to the discussion of keeping initial values ( <a href="#">see page 57</a> ).
<b>Adjust Memory Allocation</b>	Use the UMAC tool to adjust the assignment of objects in the M580 memory in the converted M580 project: <ul style="list-style-type: none"> <li>● Click the ellipses button (...) to configure the memory assignment.</li> <li>● If you do not select this feature, the converted M580 project uses the default memory assignments.</li> <li>● Refer to the discussion of memory allocation adjustment (<a href="#">see page 59</a>).</li> </ul>

## Conversion

These buttons are available in all editions of the UMAC tool:

Field	Button	Description
<b>Start Conversion</b>	<b>Convert</b>	Press this button and wait for the conversion to finish.
	<b>Save As ...</b>	Save the project.
<b>Conversion Report</b>	<b>Set Up Header/Footer</b>	Assign descriptions of the report that appear at the top (header) and bottom (footer) of the conversion report.
	<b>Generate Report</b>	Generate a report of the conversion.
	<b>View (.html)</b>	View the report in the HTML format.
	<b>View (.pdf)</b>	View the report in the PDF format.
—	<b>Back</b>	Return to the previous screen.
	<b>Finish</b>	Close the conversion program.

## Unity Pro Compatibility

The UMAC tool may not complete the conversion if the source .xef or .zef file was generated with an earlier version of Unity Pro. In this case:

Step	Action
1	Open the source project in Unity Pro XL 12.0 or later.
2	Generate a new file (.xef or .zef).
3	Repeat the conversion with the UMAC tool.



---

# Chapter 4

## Hardware Management

---

### Introduction

Use the UMAC tool to manage the migration of the Premium or Quantum hardware in your source project to an X80 project.

### What Is in This Chapter?

This chapter contains the following sections:

Section	Topic	Page
4.1	Premium Hardware Configuration	24
4.2	Quantum Hardware Configuration	30

# Section 4.1

## Premium Hardware Configuration

---

**What Is in This Section?**

This section contains the following topics:

Topic	Page
Premium Hardware Migration	25
Retaining the Premium I/O	27



## Premium Hardware Migration

### Access the Hardware Configuration

Access the **Hardware Configuration** options:

Step	Action
1	Go to the <b>Standard Conversion Options</b> on the <b>Convert</b> page.
2	Click the <b>Select drop(s)</b> button.
3	Make the appropriate selections in the <b>Premium Hardware Configuration</b> dialog box (below).

**NOTE:** The **Hardware Configuration** is available only in the **STANDARD** edition of the UMAC tool.

### Premium Configuration Change

Make these selections to migrate or replace your Premium racks and drops:

Rack	Selection	Description	M580 backplane(s) Allocation
Main Rack	Replace with X80 (Device DDT) in Local bus	Convert the local rack to an X80 local drop.	Local - 0,6
	No I/O replacement	M580 does not support a module in the drop.	
Extension Rack 1, 2, 3, ...	No I/O replacement	M580 does not support a module in the drop.	Local - 1, Local - 2, Local - 3, ...
	Retain as Premium I/O <b>NOTE:</b> Refer to the instructions to retain the Premium I/O hardware ( <a href="#">see page 33</a> ).	The Premium rack with Premium I/O connects to the X80/M580 rack with an extension cable or a module used as an extension rack.	
	Replace with X80 (Device DDT) in Local bus	The X80 rack with X80 I/O connects to the X80/M580 rack with an extension cable or a module used as an extension rack.	
	Replace with X80 (Device DDT) in EIO bus	The X80 rack with X80 I/O connects to the M580 CPU with an Ethernet/CRA module.	

**Validate the Configuration**

Assess the validity of each rack and drop in the hardware configuration by examining the corresponding icon:

Icon	Validity	Instruction
checkmark (green)	The entire configuration is valid.	Press <b>OK</b> to apply the changes and return to the <b>Convert</b> screen.
exclamation point (red)	The configuration is not valid.	Press <b>Cancel</b> and make the appropriate changes to your configuration.  <b>NOTE:</b> The conversion proceeds only when the entire configuration is valid.

Check the validity of a new configuration selection at the bottom of the screen (**Status of hardware configuration changes**).

## Retaining the Premium I/O

### Introduction

Every edition of the UMAC tool retains the Premium I/O configuration in the converted M580 project. This topic provides details about the **Retain Premium I/O** feature.

Access this feature from the list of features (*see page 19*) on the **Convert** screen.

### Topological Addresses

Every Premium application uses topological addresses to communicate with in-rack modules. By definition, a topological address includes the physical location of a specific logical rack, slot, and I/O point. For instance, the address %Q0.3.1 represents the output of channel 1 for the output module in slot 3 of rack 0.

There is some preparation required in the source Unity application before a conversion is performed. This is shown in the **Prepare** screen (*see page 17*) of the UMAC application. The UMAC tool inserts an M580 BMEXBP0800 rack into logical rack address 0. Move any Premium I/O modules in original rack 0 that you want to retain in another logical rack.

There are two types of Premium rack configurations:

- *single logical rack*: These configurations use one logical rack address. That address can refer to a pair of racks (with four, six, or eight slots) or a single 12-slot rack. (A 12-slot backplane has its own logical rack address that cannot be shared with another physical rack.) To prepare for the conversion of this configuration, add a single rack at logical rack address 1 and move the Premium I/O and power supply modules from original rack 0 to new rack 1. The Premium CPU and any communication modules can stay in logical rack 0. When you move the I/O from one rack to another, answer **YES** when you are prompted by the **Move Device** dialog box regarding the update of all references to variables in the program. A UMAC dialog box informs you that some I/O modules remain in local rack 0. In this case, complete the preparation phase before you continue.
- *multi-rack*: Extendable racks in multi-rack Premium configurations can have a maximum of eight logical rack addresses (0...7). Each logical address can contain two racks for a total of 16 physical racks, each with four, six, or eight slots. (See the note below.)

### Unsupported Modules

The number of supported Ethernet communication modules in your source application may exceed the number allowed in M580. To manage this discrepancy, adapt either the source or the target application in Unity Pro.

There are several Premium I/O modules that M580 applications do not support. For each unsupported module, the UMAC tool inserts an ST section that includes detected conversion errors.

**NOTE:** Refer to the description of ST sections and unsupported modules (*see page 83*).

M580 applications do not support these Premium modules:

Part	Description	
TSXCAY21	motion control module	2-channel axis control module
TSXCAY22	motion control module	2-channel axis control module
TSXCAY33	motion control module	3-channel axis control module
TSXCAY41	motion control module	4-channel axis control module
TSXCAY42	motion control module	4-channel axis control module
TSXCFY11	motion control module	1-channel stepper motor module
TSXCFY21	motion control module	2-channel stepper motor module
TSXCSY164	motion control module	16-axis N4 motion control module
TSXCSY164 Advanced	motion control module	16-axis N4 motion control module
TSXCSY84	motion control module	8-axis N4 motion control module
TSXCSY85	motion control module	8-axis N5 motion control module
XPSMC16	safety module	
XPSMC32	safety module	
XPSMF40	safety module	
TSXESY007	communication module	LES20 module
TSXETY100	communication module	EtherNet/IP module
TSXETY101	communication module	Ethernet module
TSXETY101.2	communication module	Ethernet module
TSXETY110	communication module	TCP/IP Ethway module
TSXETY120	communication module	Ethernet ETY120 module
TSXETY4103	communication module	Ethernet TCP/IP basic web server module
TSXETY5103	communication module	Ethernet TCP/IP configurable web server module
TSXIBX100	communication module	Interbus-S ISA module
TSXIBY100	communication module	Interbus-S module
TSXPBY100	communication module	Profibus DP module
TSXSAY100	communication module	AS-interface module
TSXSAY1000	communication module	AS-interface module V2
TSXSCY11601	communication module	Modbus board
TSXSCY21601	communication module	PCMCIA in-rack board
TSXWMY100	communication module	FactoryCast HMI web server module
TSXREY200	remote XBus module	2-channel electronic remote XBus
TSXDMY28RFK	discrete module	reflex module
TSXCCY128	counting module	electron CAM module
TSAXCTY2C	counting module	counter

### Exceptions

If you leave I/O modules in rack 0, the UMAC tool generates an error-detection message when the source file is opened and analyzed.

In this case, go back to the source application and move or delete those I/O modules. Then export a new .xef or .zef file.

# Section 4.2

## Quantum Hardware Configuration

---

**What Is in This Section?**

This section contains the following topics:

Topic	Page
Quantum Hardware Migration	31
Retaining Local Quantum I/O	33
Retaining the Quantum S908 RIO	36
Retaining the Quantum EIO (Quantum and X80)	37

## Quantum Hardware Migration

### Access the Hardware Configuration

Access the **Hardware Configuration** options:

Step	Action
1	Go to the <b>Standard Conversion Options</b> on the <b>Convert</b> page.
2	Click the <b>Select drop(s)</b> button.
3	Make the appropriate selections in the <b>Quantum Hardware Configuration</b> dialog box (below).

**NOTE:** The **Hardware Configuration** is available only in the **STANDARD** edition of the UMAC tool.

### Quantum Configuration Change

Make these selections to migrate or replace your Quantum racks and drops:

Configuration	Selection	Description
<b>Local Bus</b>	<b>Replace with Local X80 (Device DDT)</b>	Convert the local rack to an X80 local drop.
	<b>Maintain as Quantum EIO Drop</b>	Move the local rack to the EIO bus (Quantum drop) in the new M580 project.
<b>EIO Bus</b>	<b>Retain in EIO bus</b>	Keep the drop in the EIO bus.
	<b>Replace with EIO - X80 drop</b>	Replace the drop with an EIO X80 drop.
<b>RIO Bus</b>	<b>Retain in RIO bus</b>	Keep the drop in the RIO bus.
	<b>Maintain as Quantum EIO Drop</b>	Retain the drop in the EIO bus.
	<b>Replace with EIO - X80 drop</b>	Replace the drop with an EIO X80 drop.
<b>DIO Bus</b>	<b>No Replacement</b>	M580 does not support a module in the drop.
	<b>Maintain as Quantum EIO Drop</b>	Keep the drop in the EIO bus.
	<b>Replace with EIO - X80 drop</b>	Replace the drop with an EIO X80 drop.

When you choose to replace a legacy Quantum drop with an X80 drop, the UMAC software replaces the state RAM assignments that correspond to the legacy I/O modules with the Device DDT assignments that are required by X80 I/O modules. The UMAC software, therefore, adds ST sections to the converted application to show the direct mapping of the inputs (`_IN`) and outputs (`_OUT`) between the source application and the converted application.

**NOTE:**

- If topological addressing is used in the legacy application, aliasing is used to connect the topological addresses with the Device DDT assignments.
- Refer to the description of ST sections in the Unity Pro MAST task ([see page 77](#)).

Continue to these instructions:

- Maintain as Quantum EIO Drop ([see page 33](#))
- Retain the Quantum S908 RIO ([see page 36](#))
- Retain the Quantum EIO (Quantum and X80) ([see page 37](#))

**Validate the Configuration**

Assess the validity of each rack and drop in the hardware configuration by examining the corresponding icon:

Icon	Validity	Instruction
checkmark (green)	The entire configuration is valid.	Press <b>OK</b> to apply the changes and return to the <b>Convert</b> screen.
exclamation point (red)	The configuration is not valid.	Press <b>Cancel</b> and make the appropriate changes to your configuration. <b>NOTE:</b> The conversion proceeds only when the entire configuration is valid.

Check the validity of a new configuration selection at the bottom of the screen (**Status of hardware configuration changes**).



## Retaining Local Quantum I/O

### Introduction

This topic provides details about the **Retain Local Quantum I/O** feature, which is available in all editions of the UMAC tool.

Access this feature from the list of features (*see page 19*) on the **Convert** screen.

### State RAM Addresses

The vast majority of Quantum applications use state RAM addressing, which communicates with I/O points without regard to their physical locations. (For example, you can assign %M101 to channel *x* of an output module regardless of its location.) In most cases, therefore, you do not have to move the I/O modules from rack 0 in the Quantum source application to rack 0 in the M580 target application.

Create an EIO Drop from a Local Quantum Rack

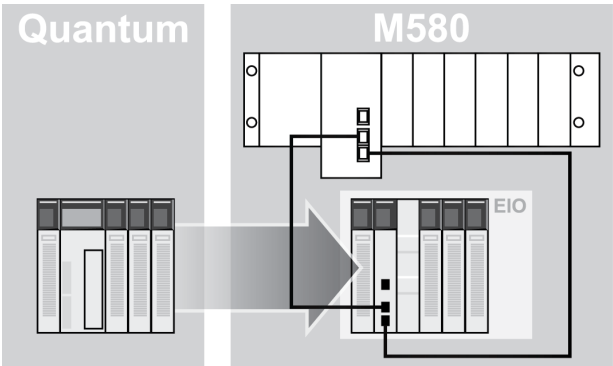
If the source configuration does not have an RIO S908 drop, the UMAC tool performs these actions to convert the configuration for a Quantum standalone rack to an M580 local rack:

- Add a BMEXBP1200 M580 base rack, BMXCPS4002 power supply, and BMEP586040 CPU to the configuration in the M580 target application.
- Replace the Quantum CPU with a 140CRA31200 adapter module.
- Configure the Quantum rack as an EIO drop in the M580 EIO network.

If the configuration has an RIO S908 drop, the UMAC tool performs these actions to convert the configuration for a Quantum standalone rack with RIO to an M580 local rack:

Step	Action
1	Add a BMEXBP1200 M580 base rack, BMXCPS4002 power supply, and BMEP586040 CPU to the configuration of your M580 application.
2	Replace the Quantum CPU with a 140CRA31908 adapter module.
3	Configure the Quantum rack as an EIO drop in the M580 EIO network that can communicate with the 140CRP93•00 S908 head module.
4	Physically replace the Quantum CPU with a 140CRA31908 adapter module that includes the S908 head module.

When the dual-slot Quantum CPU on the local drop is replaced by a single-slot 140CRA31200 adapter module, the local drop can become an EIO drop in the M580 system:



## Unsupported Modules

Unity M580 limits the number of supported Ethernet communication modules in the configuration. Your source application may exceed that number. Adapt either the source or target application in Unity Pro to manage this discrepancy.

There are several Quantum I/O modules that M580 applications do not support. For each unsupported module, the UMAC tool inserts an ST section into the MAST task that includes detected conversion errors.

**NOTE:** Refer to the description of ST sections and unsupported modules ([see page 83](#)).

M580 applications do not support these Quantum modules:

Part	Description	
140CRP31200	communication module	TSX Ethernet I/O head module
140CRP93•00	communication module	S908 RIO head module
140EIA92100	communication module	1-channel AS-I module
140NOC77100	communication module	EtherNet/IP module
140NOC77101	communication module	Ethernet module
140NOC78000	communication module	TSX Ethernet DIO head module
140NOC78100	communication module	TSX Ethernet control module
140NOE31100	communication module	SY/MAX Ethernet module (twisted pair)
140NOE35100	communication module	SY/MAX Ethernet module (fiber optic)
140NOE77100	communication module	Ethernet TCP/IP basic web server module
140NOE77101	communication module	Ethernet TCP/IP basic web server module
140NOE77110	communication module	Ethernet TCP/IP configurable web server module
140NOE77111	communication module	Ethernet TCP/IP configurable web server module
140NOG11100	communication module	A-line 1/SFB bus master module
140NOM2••00	communication module	MN1 Modbus Plus
140NRP95400	communication module	S908 fiber converter MM/ST
140NRP95401C	communication module	S908 fiber converter SM/LC
140NWM10000	communication module	FactoryCast HMI web server module
GENNOM	communication module	NOM type generic module
PTQPDPMV1	communication module	Profibus DP/DPV1 master module
140DCF07700	expert module	real-time DCF clock interface module
140HLI34000	expert module	HS-LTCH/INTPT-16 (interrupt module)
140XCP90000	expert module	battery backup
140MSB10100	motion control module	1-axis motion module (INC/ENC)
140MSC10100	motion control module	1-axis motion module (ENC/RES)
140MMS42501	motion control module	SERCOS
140MMS53502	motion control module	SERCOS

## Retaining the Quantum S908 RIO

### Introduction

This topic provides details about the **Retain Quantum S908 RIO** feature, which is available in all editions of the UMAC tool.

Access this feature from the list of features (*see page 19*) on the **Convert** screen.

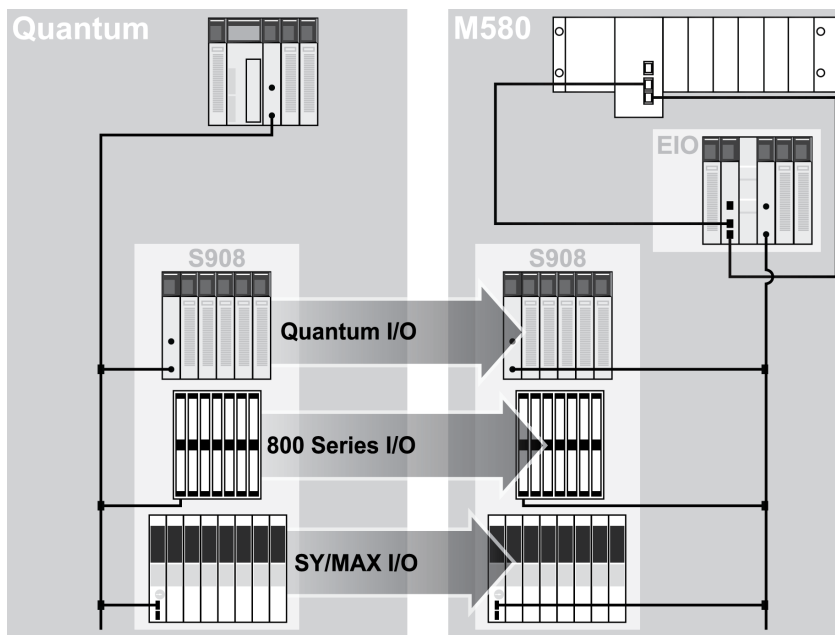
### Configuration

Some Quantum applications contain legacy S908 communications networks. Those networks include a 140CRP93•00 S908 head module on the local Quantum rack. To retain these networks, Schneider Electric developed an EIO drop module (140CRA31908) to facilitate communications between the M580 EIO network and the S908 network.

An S908 network typically connects I/O modules over a long distance for redundancy and includes these Modicon platforms:

- Quantum
- 800 series
- SY/MAX

In a converted Quantum application, the UMAC tool replaces the Quantum CPU with the 140CRA31908 module to create a drop on the M580 EIO network that communicates with the S908 head module:



## Retaining the Quantum EIO (Quantum and X80)

### Introduction

This topic provides details about the **Retain Quantum EIO** feature, which is available in all editions of the UMAC tool to retain the Quantum and X80 EIO drops in the source Quantum application. Access this feature from the list of features (*see page 19*) on the **Convert** screen.

### Configuration

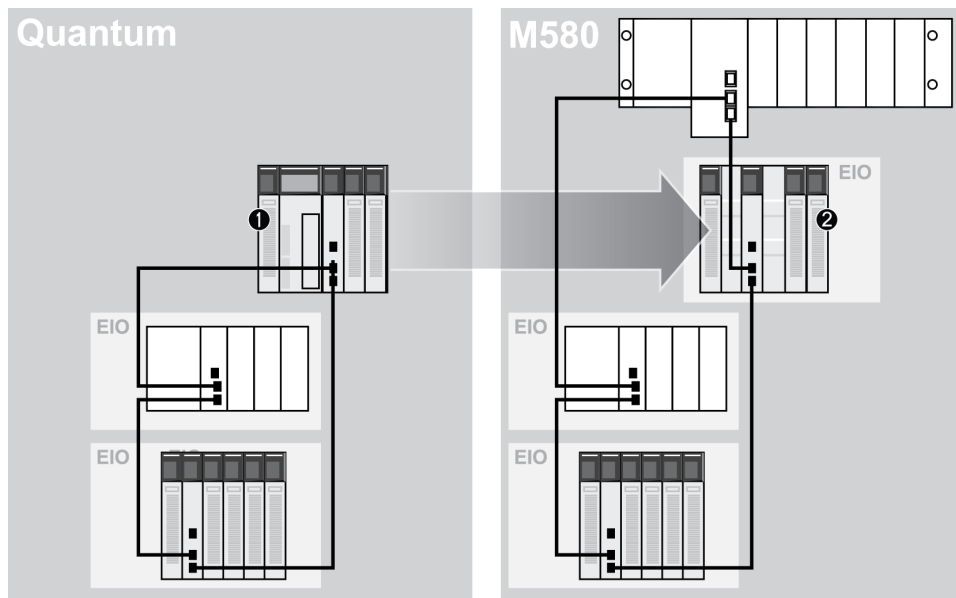
When you use the UMAC tool, notice that some Quantum source applications contain a Quantum or X80 EIO communications network. In those cases, the UMAC tool configures the 140CRP31200 EIO head module, which is installed on the local rack, so the target M580 network can retain the EIO communications network.

The UMAC tool performs these operations to convert the configuration of the Quantum local rack to an M580 configuration:

Step	Action
1	Add a BMEXBP1200 M580 base rack, BMXCPS4002 power supply, and BMEP586040 CPU to the configuration of the M580 target application.
2	Replace the CPU and the EIO head module in the local Quantum rack with a 140CRA31200.
3	Configure the Quantum rack as an EIO drop in the M580 EIO network. This drop includes all of the I/O modules formerly in Quantum local racks.

**Example**

In this example, the CPU and the CRP module in the Quantum source hardware configuration are replaced by a 140CRA31200 adapter module in the M580 target configuration, thereby retaining the EIO drop in the M580 architecture:



- 1 This local rack in a Quantum source configuration includes a CPU and a 140CRP31200 module.
- 2 This Quantum rack in an M580 configuration includes a 140CRA31200 module.

---

# Chapter 5

## Logic Management

---

### Introduction

Use the information in this chapter to manage the migration of the logic in your source project to an X80 project with the UMAC tool.

### What Is in This Chapter?

This chapter contains the following topics:

Topic	Page
Retaining the Logic	40
Mismatched System Words and Bits	41
MAST Task Configuration	43
Missing Function Blocks	44
Unsupported Data Types	46
Address Alignment	47
Alignment in Multi-Dimensional Arrays and Data Structures	50
ADDR to ADDM Instructions	51
Ethernet I/O Scanner Translation to DTM	52
Animation Tables and Operator Screens	56
Retain Initial Values	57
Adjust Memory Allocation	59

## Retaining the Logic

### Introduction

This topic provides details about the **Retain Logic** feature, which is available in all editions of the UMAC tool.

Access this feature from the list of features (*see page 19*) on the **Convert** screen.

### Logic Sections

The logic sections in Premium and Quantum applications use native IEC-61131-3 languages and LL984 sections. The UMAC tool retains that logic for each task, section, SR section, and event in the target application.

Because M580 handles only AUX0 and AUX1 events, the AUX2 and AUX3 events in Premium applications are added as sections at the end of the MAST task.



## Mismatched System Words and Bits

### Introduction

This topic provides details about the **Identify Mismatched System Words and Bits** feature, which is available in all editions of the UMAC tool.

Access this feature from the list of features (*see page 19*) on the **Convert** screen.

### About System Words and Bits

Each PLC family uses system words and bits to (for example) monitor the health, performance, and communications in the PLC network. These words and bits can be assigned to variables or used directly in logic, DFBs, animation tables, and operator screens.

The functions of some words and bits do not change across PLC platforms, but sometimes the function of a bit or word in a Premium or Quantum source application is different from its function in an M580 application.

The UMAC tool compares the descriptions of the system words and bits that are used in the logic of the source (Premium or Quantum) application with the descriptions of the physical words and bits in the target (M580) application:

mismatch	For any mismatch in the descriptions for system words and bits, the match is tabulated in the <b>Evaluation and Conversion Report</b> . In the case of at least one mismatch, the UMAC tool inserts a conversion error into the ST sections of Unity Pro logic to generate a detected-error message when you build or analyze the M580 application.  <b>NOTE:</b> Refer to the instructions for managing detected errors in ST section logic ( <i>see page 83</i> ).
match	System words and bits that match are shown in the <b>Evaluation and Conversion Report</b> .

Matching or mismatched system words or bits are assigned to variables or used within DFBs, animation tables, or operator screens, the occurrence is tabulated in the **Evaluation and Conversion Report**.

For each match or mismatch in the **Evaluation and Conversion Report**, the number of occurrences appear in the **Benefit Estimation** spreadsheet. Default values are inserted in the spreadsheet to calculate the amount of time you saved by using the UMAC tool (as opposed to manual conversion).

## WARNING

### UNEXPECTED APPLICATION BEHAVIOR

- Resolve and remove each conversion error (`convError`) that the UMAC tool identifies.
- Perform a complete validation of the converted application in a simulated environment before you deploy a production system.
- Ensure that the converted application works as designed.

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

**Example: Match**

System bit %S13 performs the same function (first cycle after RUN) across all Schneider Electric PLC families. When this bit is used in logic it does not generate an error, and its use is tabulated in the **Evaluation Report** at **System Bits and Words Info: Logic Tasks/Sections**:

System Bits and Words Info:Logic Tasks/Sections : Match List						
Task	Section Name	Object	Tag Name	Occurences	Function Name : Description in Source environment	Functional Name :Description in Destination environment (M580)
MAST	One	%S13		1	1STSCANRUN : First cycle after switching to RUN	1STSCANRUN : First cycle after switching to RUN

**Example: Mismatch**

System bit %S68 (processor battery fault) is not used by M580 applications. Any occurrence of that bit in the source application is followed by a conversion error instruction. You, therefore, cannot build or analyze the M580 project in Unity. The use of this bit is tabulated in the **Evaluation Report** at **System Bits and Words Info: Logic Tasks/Sections**:

System Bits and Words Info:Logic Tasks/Sections : Mismatch List						
Task	Section Name	Object	Tag Name	Occurences	Function Name : Description in Source environment	Functional Name :Description in Destination environment (M580)
MAST	One	%S118		1	REMIOERR : General remote I/O fault (Fipio)	<i>Not Supported on M580 platform.</i>

This is the corresponding code for the %S13 and %S68 system bits in an ST section of the MAST task in the converted M580 application:

```
IF %S13 THEN
    Reset_ALL := TRUE;
END_IF;

IF %S68 THEN
    Processor_Battery_Out := TRUE {convError('Bit / Word meaning has
changed. Please make required adaptation and changes')};
END_IF;
```

## MAST Task Configuration

### MAST Configuration Options

You can access the **MAST Task Configuration** feature from the **Convert** screen in every edition of the UMAC tool.

Choose an option to configure the MAST task schedule:

- **Cyclic:** The task cycles for solving logic, processing inputs, and updating outputs are sequential (one after the other). After the outputs are updated, the system performs its own specific processing and the next scan starts without delay. The cycle is checked by the watchdog timer.

**NOTE:** The **Cyclic** option is not available for M580 Hot Standby CPUs.

- **Periodic:** Logic is solved, inputs are processed, and outputs are updated over a defined period (1 ... 255 ms). The countdown begins from the defined period when the scan starts. When the scan finishes before a timeout, a new cycle is launched. The PLC uses any time remaining in the scan for internal processing. If the scan does not finish within the defined period, the PLC sets the task overrun bit (%S19) to 1.

**NOTE:** Use the **Periodic** option for M580 Hot Standby CPUs.

## Missing Function Blocks

### Introduction

This topic provides details about the **Replace Missing FFBs** feature, which is available for source applications in the **STANDARD** edition of the UMAC tool.

During the conversion, the UMAC tool finds commonly used legacy functions and function blocks (FFBs) in a Premium or Quantum source application and replaces them in the M580 target application with derived function blocks (DFBs). These new DFBs were developed specifically to correspond to each supported legacy FFB, and they have the same functionality as the obsolete FFBs that they replace. Users, therefore, do not have to create alternative blocks.

Access this feature from the list of features (*see page 19*) on the **Convert** screen.

### Replacement Process

The UMAC tool performs these updates automatically:

Stage	Description
1	The tool identifies and locates missing FFBs.
2	The tool replaces each obsolete FFB with a new corresponding DFB.
3	The tool adapts the logic syntax and variable declarations, as required by the application.

### Obsolete Library

The UMAC tool replaces these FFBs from the Unity **Obsolete Library** with DFBs that have the *same name*:

Quantum FFBs			
GET_3X	GET_4X	PUT_4X	IEC_BMDI_M
Premium FFBs			
PL7_3_TIMER	PL7_TON	ADD_DT_PL7	DT_ARINT_TO_STRING
PL7_COUNTER	PL7_TOF	SUB_DT_PL7	DAY_OF_WEEK
PL7_MONOSTABLE	PL7_TP	DATE_DINT_TO_STRING	FTON
PL7_REGISTER_32	ADD_TOD_PL7	TIME_DINT_TO_STRING	FTOF
PL7_REGISTER_255	SUB_TOD_PL7	TOD_DINT_TO_STRING	FTP

## Asynchronous Function Blocks in Premium Projects

The Premium operating system executes some function blocks (including timers and counters) on each scan without directly calling them in each scan. The UMAC tool calls these “asynchronous” function blocks.

The execution of asynchronous function blocks by the operating system is independent of the CPU scan. To replicate that behavior in M580, the UMAC tool inserts an ST section ([see page 77](#)) with calls to the asynchronous DFBs. This allows these DFBs to be refreshed in each scan without a call to the SR section they are in.

To avoid some behavior deviation, for each such asynchronous function block instance that is encountered in the source Premium application, the UMAC tool inserts in the M580 converted application a blank call to this DFB in a supplementary section that is inserted at the end of the MAST task. As a result, the DFB (and its internal data and outputs) is called or refreshed on each scan.

The UMAC tool supports an asynchronous call to any of these function blocks:

PL7_3_TIMER
PL7_COUNTER
PL7_MONOSTABLE
PL7_REGISTER_32
PL7_REGISTER_255
PL7_TOF
PL7_TON
PL7_TP

# Unsupported Data Types

## Introduction

This topic provides details about the **Replace Unsupported Data Types** feature, which is available in the **STANDARD** edition of the UMAC tool.

Access this feature from the list of features (*see page 19*) on the **Convert** screen.

## About Unsupported Data Types

M580 applications do not support some of the data types that Premium does, including 32-bit constants %KD and %KF and 32-bit memory words %MD and %MF.

The UMAC tool creates a new variable with a name based on the unsupported data types description and assigns it to the corresponding %MW or %KW descriptor in the converted target M580 application.

## Example

If a Premium source application has %KD100 assigned to a value 500,000 in the **Data Editor**, the UMAC tool creates a variable called KD100 and assigns it to addresses %KW100 and %KW101. The assigned value of 500,000 also gets assigned to KD100 in the M580 target application. Examine the evolution of the ST section in this example:

Application	ST Section
Premium (source)	My_DINT_Constant := %KD100
M580 (target)	My_DINT_Constant := KD100

**NOTE:** Refer to the description of ST sections in the Unity Pro MAST task (*see page 77*).

**Exception:** When any of these unsupported data types are assigned to odd memory words (such as %MD101), the UMAC tool creates a new unlocated variable that is based on the descriptor and its memory location. The converter then creates PUSH-PULL statements within the logic to emulate the Premium source application as defined in the address alignment description (*see page 47*).

## Address Alignment

### Introduction

This topic provides details about the **Fix Alignment Constraints** feature, which is available in the **STANDARD** edition of the UMAC tool.

Access this feature from the list of features (*see page 19*) on the **Convert** screen.

### About Address Alignment

Unlike Premium and Quantum applications, M580 applications do not allow data types that typically require 32 bits of memory (like REAL or DINT) to start on odd-numbered memory addresses. Therefore, the UMAC tool adapts those addresses for use in the M580 target application to maintain the even and odd address assignment that is used in peer-to-peer communications between (for example) a PLC and a SCADA system or HMI:

- **On The Fly strategy** (Premium only): The UMAC tool inserts PUSH-PULL instructions in the logic immediately before and after they are needed.
- **FLIP-FLOP strategy** (default for Premium or Quantum): The UMAC tool inserts ST sections with PUSH-PULL instructions (*see page 81*) at the beginning and end of the task.

### Unity Instructions

These tables show the Unity PUSH-PULL instructions for simple data types and arrays.

Simple data types:

PULL DFB	Type	PUSH DFB
INTS_AS_DINT	DINT	DINT_AS_INTS
INTS_AS_REAL	REAL	REAL_AS_INTS
INTS_AS_UDINT	UDINT	UDINT_AS_INTS
INTS_AS_DWORD	DWORD	DWORD_AS_INTS
INTS_AS_TIME	TIME	TIME_AS_INTS
INTS_AS_TOD	TOD	TOD_AS_INTS
INTS_AS_DATE	DATE	DATE_AS_INTS
INTS_AS_DT	DT	DT_AS_INTS

Arrays:

PULL DFB	Type	PUSH DFB
COPY_ARINT_AS_ARDINT	ARRAY OF DINT	COPY_ARDINT_ARINTS
COPY_ARINT_AS_ARREAL	ARRAY OF REAL	COPY_ARREAL_ARINTS
COPY_ARINT_AS_ARUDINT	ARRAY OF UDINT	COPY_ARUDINT_ARINTS
COPY_ARINT_AS_ARDWORD	ARRAY OF DWORD	COPY_ARDWORD_ARINTS
COPY_ARINT_AS_ARTIME	ARRAY OF TIME	COPY_ARTIME_ARINTS
COPY_ARINT_AS_ARTOD	ARRAY OF TOD	COPY_ARTOD_ARINTS
COPY_ARINT_AS_ARDATE	ARRAY OF DATE	COPY_ARDATE_ARINTS
COPY_ARINT_AS_ARDT	ARRAY OF DT	COPY_ARDT_ARINTS

The UMAC tool creates single instances of these DFBs within the target project. The names of those instances suggest their functions. For example:

- To pull a DINT, the tool creates the instance Pull\_DINT of type INTS\_AS\_DINT.
- To push a DINT, the tool creates the instance Push\_DINT of type DINT\_AS\_INTS.
- For projects with more than one task, you can assign separate instances of PUSH and PULL DFBs. That is, create a new instance for each task by prefixing the task name with the DFB name (for example, MAST\_Pull\_DINT or FAST\_Push\_REAL).

Constraints

Observe these guidelines for Premium and Quantum source applications:

Configuration	Description
Premium	Premium applications do not have constraints regarding the physical memory addresses that are assigned to 32- and 64-bit data types, including DATE, DINT, DT, DWORD, REAL, TIME, TOD, UDINT. You can therefore assign any physical memory address (even or odd) to these data types. (For example, you can assign a DINT variable to %MW100 or %MD101.) Conversely, in an M580 application, you cannot assign a 32- or 64-bit single variable or array to an odd memory address.
Quantum	Quantum applications do not have constraints regarding the assignment of 32- and 64-bit data types. You can assign the data types DATE, DINT, DT, DWORD, REAL, TIME, TOD, UDINT to any %MW physical address.

On the Fly Strategy

**NOTE:** The On The Fly strategy is not available for Quantum source applications.

The UMAC tool takes these actions when it encounters simple type variables that are subject to alignment constraint in a Premium source application:

Step	Action
1	The UMAC tool creates unlocated variables with names that are based on defined variable names from the source application. <b>Example:</b> When My_REAL_01 is assigned to %MW101 in the source application, it becomes an unlocated variable called My_REAL_01 in the target application.
2	The UMAC tool turns unsupported data types (like %MD and %KF) into unlocated variables that are based on their physical memory locations. (Refer to the discussion of unsupported data types <a href="#">(see page 46)</a> .)
3	The UMAC tool creates the DFBs that push a single unlocated 32-bit data type into two located INTs.
4	The UMAC tool creates the DFBs that pull two 16-bit located INTs into a single unlocated 32-bit data type.
5	The UMAC tool creates the DFBs that push a single unlocated 64-bit data type into four 16-bit located INTs.
6	The UMAC tool creates the DFBs that pull four 16-bit located INTs into a single unlocated 64-bit data type.



These additional considerations apply to Premium source applications:

- The UMAC tool adds PULL instructions to the logic to assign the correct values to unlocated 32- or 64-bit variables for use as an operand.
- The UMAC tool adds PUSH instructions to the logic to store unlocated 32- or 64-bit variables in located memory when they are used as resultants.

### FLIP-FLOP Strategy

The UMAC tool performs several operations in the M580 target application to maintain the even and odd address assignment in the source Premium or Quantum application. This address assignment supports peer-to-peer communications between (for example) a PLC and a SCADA system or HMI.

**NOTE:** The **FLIP-FLOP** strategy is selected by default for Premium and Quantum applications.

This table shows the steps when the FLIP-FLOP method is used to convert a source application:

Step	Action
1	The UMAC tool creates unlocated variables with names that are based on defined variable names from the source application. <b>Example:</b> When <code>My_REAL_01</code> is assigned to <code>%MW101</code> in the source application, it becomes an unlocated variable called <code>My_REAL_01</code> in the target application.
2	The UMAC tool creates the DFBs that push a single unlocated 32-bit data type into two 16-bit located INTs.
3	The UMAC tool creates the DFBs that pull two 16-bit located INTs into a single unlocated 32-bit data type.
4	The UMAC tool creates the DFBs that push a single unlocated 64-bit data type into four 16-bit located INTs.
5	The UMAC tool creates the DFBs that pull four 16-bit located INTs into a single unlocated 64-bit data type.
6	The UMAC tool creates the PULL instructions in the new ST section at the beginning of each task to assign the correct value to the unlocated 32- or 64-bit variable for use within the logic.
7	The UMAC tool adds PUSH instructions to the new ST section at the end of each task to store unlocated 32- or 64-bit variables in located memory when the value is changed because of a function block.

**NOTE:** Refer to the description of ST sections in the Unity Pro logic ([see page 78](#)).

## Alignment in Multi-Dimensional Arrays and Data Structures

### Introduction

This topic provides details about the **Fix Alignment in Multi-Dimensional Arrays and Data Structures** feature, which is available in the **STANDARD** edition of the UMAC tool.

Access this feature from the list of features (*see page 19*) on the **Convert** screen.

### Alignment

The address alignment description (*see page 47*) applies only to simple data types and one-dimensional arrays of simple data types. This selection (**Fix Alignment in Multi-Dimensional Arrays and Data Structures**) allows the UMAC tool to perform the same functions on two-dimensional arrays and simple derived data types (DDTs) that have alignment constraints.

### Exceptions

Data Structure	Description
one-dimensional arrays	For one-dimensional arrays that have address constraints, the entire array is assigned to an unlocated array. Values are moved between the unlocated array and the odd memory assignments of the 32- or 64-bit data types by using the PUSH-PULL DFBs in either the FLIP-FLOP strategy or On-The-Fly strategies.
two-dimensional arrays	Two-dimensional arrays may have 32- or 64-bit elements that are assigned to odd memory locations. There may also be even memory word assignments in the same array. In this case, the UMAC tool creates unlocated variables for only those elements that are assigned to odd memory locations. As with one-dimensional arrays, the values at the odd memory locations are moved using PUSH-PULL DFBs in the FLIP-FLOP strategy or On-The-Fly strategy.
simple data structures	For simple data structures, the assignment of a single element to an odd memory location means the entire data structure is assigned to an unlocated array of INTs (regardless of the number of 32- or 64-bit elements that are assigned to even memory locations). These elements are then moved between the original assigned memory locations and the unassigned array of INTs using PUSH-PULL DFBs with the FLIP-FLOP strategy or On-The-Fly strategy.

**NOTE:** Refer to the descriptions of the **FLIP-FLOP strategy** and **On The Fly strategy**.

## ADDR to ADDM Instructions

### Introduction

This topic provides details about the **Replace ADDR instruction with ADDM** feature, which is available in the **STANDARD** edition of the UMAC tool.

Access this feature from the list of features (*see page 19*) on the **Convert** screen.

### About ADDR and ADDM Instructions

The Premium family uses ADDR instructions to convert character strings into addresses that can be used directly by the communication functions. Since M580 does not support ADDR instructions, the UMAC replaces the string “ADDR” in the Premium source application instruction with “ADDM” in the M580 target application instruction.

This ADDR-to-ADDM conversion occurs in each supported IEC 61131-3 language (FBD, IL, LD, ST).

## Ethernet I/O Scanner Translation to DTM

### Introduction

This topic provides details about the **Ethernet I/O Scanner Translation into DTM** feature, which is available in the **STANDARD** edition of the UMAC tool.

Access this feature from the list of features (*see page 19*) on the **Convert** screen.

The UMAC tool can handle any source project that includes Ethernet I/O scanner tables. An `IOScannerAlert` ST section is inserted in the target M580 application. Double-check the correctness of memory mapping from the source project to the target project and delete the section.

**NOTE:** Refer to the description of ST sections in the Unity Pro MAST task (*see page 77*).

### WARNING

#### UNINTENDED EQUIPMENT OPERATION

Verify that the memory mapping in the target application is complete.

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

### Legacy Ethernet I/O Scanner Operation

The Premium and Quantum families support the I/O scanner for Modbus TCP/IP communications with distributed I/O. This is done through the Ethernet coprocessor on equipped CPUs or through a dedicated Ethernet communications module (like Premium ETY or Quantum NOE modules).

The Ethernet I/O scanner table contains entries like **IP address**, **Repetitive rate (ms)**, **RD Master Object**, **RD Ref Slave**, **WR Master Object**, **WR Ref Slave**, and source/target write addresses. While M580 does not support the Ethernet I/O scanner table format, it does rely on a Device Type Manager (DTM) within the Field Device Tool (FDT) technology.

**NOTE:** Refer to the description of new ST sections that identify unsupported modules (*see page 83*).

### Methodology

The UMAC tool applies each entry in the Ethernet I/O scanner table to a request in a Modbus device within a hardware-specific DTM device list. Entries within an Ethernet I/O scanner table for a Premium or Quantum CPU with an Ethernet coprocessor are converted into Modbus devices within the M580 6040 CPU DTM device list. Likewise, entries for Premium ETY or Quantum NOE modules are converted into Modbus devices within the BMENOC0301 DTM device list. Multiple entries in the Ethernet I/O scanner table that are assigned to the same IP address receive separate requests within the same Modbus device.

Modbus devices within a DTM are assigned to a specific Device DDT. These DDTs are not assigned to memory words as in the source Premium or Quantum Ethernet I/O scanner table. The UMAC tool generates ST sections in the MAST task for direct assignment of Device DDT I/O elements. One element is placed at the beginning of the MAST task to assign read arrays in the Device DDT to the register address locations. Another element is placed at the end of the MAST task to assign located registers to write arrays in the Device DDT.

## Overflowing Scan Lines and Words

When a Premium or Quantum Ethernet coprocessor is assigned more scan lines or words than a corresponding M580 CPU coprocessor can handle, the maximum number of scan lines or words are assigned to that M580 CPU coprocessor with the remainder assigned to an additional BMENOC0301. This occurs when there are more than 112 scan lines in the I/O scanner table for the Premium or Quantum CPU coprocessor.

## Scanline Support for Premium

This table shows the number of scanlines that each Premium module supports. It also shows the types of supported connections:

Premium	Reference	Connection	Scanlines
CPU	TSXP571634M	ETY port	TCP/IP 10/100 regular connection
	TSXP572634M	ETY port	TCP/IP 10/100 regular connection
	TSXP573634M	ETY port	TCP/IP 10/100 regular connection
	TSXP574634M	built-in	TCP/IP 10/100 extended connection
	TSXP575634M	built-in	TCP/IP 10/100 extended connection
	TSXP576634M	built-in	TCP/IP 10/100 extended connection
communications	TSXEY5103	TCP/IP 10/100 regular connection	64
	TSXEY4103	TCP/IP 10/100 regular connection	64

## Scanline Support for Quantum

This table shows the number of scanlines that each Quantum module supports. It also shows the types of supported connections:

Quantum	Reference	Connection	Scanlines
CPU	140CPU65150	TCP/IP 10/100 extended connection	128
	140CPU65160	TCP/IP 10/100 regular connection	128
	140CPU65260	TCP/IP 10/100 regular connection	128
	140CPU65860	TCP/IP 10/100 extended connection	128
communications	140NOE77100 v2.2 (or earlier)	TCP/IP 10/100	64
	140NOE77100 v3.0 (or later)	TCP/IP 10/100	128
	140NOE77101	TCP/IP 10/100 regular connection	128
	140NOE77110	TCP/IP 10/100 FactoryCast	128
	140NOE77111	TCP/IP 10/100 regular connection	128

### Scanline Support for M580

This table shows the number of scanlines that are supported for each M580 module. It also shows the types of supported connections:

M580	Reference	Scanlines
CPU	BME•58•0•0	112
	BMENOC3•1.•	112

### Example

Use the tables below to compare the I/O scanner table in the source application to the communication DTM in the target application.

**NOTE:** These instructions assume that you are familiar with the Unity Pro programming software.

**Premium and Quantum:** View the I/O scanner table before the conversion:

Step	Action
1	To view the configured networks in the <b>Project Browser</b> for the source Unity Pro project, expand <b>Communication → Networks</b> . <b>NOTE:</b> For this example, imagine these two configured communications networks: <code>Ethernet_CPU</code> , <code>Ethernet_NOE</code>
2	Double-click the <b>Ethernet_CPU</b> network to access the <b>I/O Scanner</b> tab and view the two scan lines that are assigned to the same IP address. <b>NOTE:</b> The M580 target application associates those scan lines with each target DTM as defined by the source application.

**M580:** View the DTM table after the conversion:

Step	Action
1	Open the <b>DTM Browser</b> in the M580 Unity Pro project and expand the <b>Distributed Bus</b> to see that the target M580 application includes target DTMs that correspond to the scan lines for the <code>Ethernet_CPU</code> network in the source application.
2	In the M580 application, expand the device list for the <code>BMEP58_ECPU_EXT</code> DTM to see that it includes the two devices.
3	Notice the two requests for device <code>MBDevice1_10_169_69_10</code> are now the first two entries in the I/O scanner table. The address assignments and block lengths are assigned in the Request Setting tab of the <code>MBDevice1_10_169_69_10</code> in the <code>BMEP58_ECPU_EXT</code> DTM.

**NOTE:** The IP addresses come from the I/O scanner table, but the subnet mask and gateway are default parameters based on the local PC network interface card.

### IP Parameter Adjustment

The UMAC conversion retains only the IP address for each Ethernet I/O scanner entry. Therefore, adjust the **Subnet Mask** and **Gateway** in the **Address Setting** tab of the CPU or NOC DTM for each Modbus device:

Step	Action
1	Set the DHCP for this device in the drop-down list to <b>Enabled</b> .
2	Adjust the <b>Subnet Mask</b> and <b>Gateway</b> .
3	A pencil icon appears next to the altered field after you press the tab key. If a red apostrophe appears, the entry is not consistent with the subnet or gateway of the CPU/NOC. (Adjust the <b>Subnet Mask</b> and <b>Gateway</b> parameters accordingly.)
4	Set the DHCP for this device drop-down list to <b>Disabled</b> .
5	Click <b>Apply</b> to accept the changes.

## Animation Tables and Operator Screens

### Introduction

This topic provides details about the **Retain existing Animation Tables and Operator Screen** feature, which is available in the **STANDARD** edition of the UMAC tool.

Access this feature from the list of features (*see page 19*) on the **Convert** screen.

### About Animation Tables and Operator Screens

The UMAC tool brings animation tables and operator screens from the Premium or Quantum source application into the M580 target application.

Unsupported data types and address alignment constraints that require the UMAC tool to generate new variables replace the previous variables or unsupported data types in the table or screen.



## Retain Initial Values

### Introduction

This topic provides details about the **Retain Initial Values** feature, which is available in the **STANDARD** edition of the UMAC tool.

Access this feature from the list of features (*see page 19*) on the **Convert** screen.

### New ST Section

The UMAC tool adds an ST section called ReadMe\_First to the converted project. This new section includes a conversion error message that reports the initial values that were modified in the conversion. To successfully build the program, delete the ReadMe\_First section.

**NOTE:** Refer to the instructions for managing detected errors in ST section logic (*see page 83*).

### WARNING

#### UNINTENDED OPERATION AT START-UP

Delete the ReadMe\_First ST section before you build the Unity Pro program.

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

### Retaining Initial Values from Source Projects

The UMAC tool retains the initial values from the Premium or Quantum source applications and uses them in the converted M580 target application. These include simple data types, one- and two-dimensional arrays, and derived data types (DDTs).

PLC platforms can have initial values set directly to constant memory locations, located and unlocated single data type, arrays, or structures. A variable or object with an initial value gets that value on a cold start. Initial values allow a given machine to be reset to its initial conditions to prepare for a restart.

The UMAC tool handles all forms of variables with initial values. These include those unsupported data types that are in Premium but unsupported in M580, such as %KD and %KF. In these instances, unlocated variable names are created and PULL statements are used to get initial values, as explained in the description of unsupported data types (*see page 46*):

Type	Address	Value
DINT	%KD10	16#1111_2222
DINT	%KD15	16#3333_4444
REAL	%KF20	1111.2222
REAL	%KF25	3333.4444
INT	%KW30	1234
INT	%KW35	5678

Another example might have an alignment constraint issue where a 32-bit or 64-bit datatype is located on an odd memory word and requires the creation of an unlocated variable with a PULL statement that transfers the initial value to the correct memory location for use by the project.

### Example

This is an example of a variable section that could be included in a Quantum or Premium source project:

Name	Type	Address	Value
VAR1	UDINT	%MW101	454363

The UMAC tool unlocates the variable from the memory word, which makes VAR1 an unlocated variable with an associated initial value that is redundant. To offset this, these variables are added to the converted output:

- Two variables of type INT for the parent data type DINT (32 bits), in this case MW101 and MW102.
- These newly generated variables occupy two consecutive addresses starting at the address associated with the parent variable:
  - MW101 at %MW101
  - MW102 at %MW102

The initial 32-bit UDINT value 454363 (6EEDB hex) is split into two consecutive words. In this case, consider the first word (0x0006) to be the most significant bit (MSB). Consider the second word 0xEEDB to be the least significant bit (LSB):

Variable	Address	Value
MW101 (MSB)	%MW101	0xEEDB
MW102 (LSB)	%MW102	0x0006

The UMAC tool adds these variables to the variable list in the target M580 application. You can view these variables in the elementary variable window.

## Adjust Memory Allocation

### Introduction

This topic provides details about the **Adjust Memory Allocation** feature. This feature is available for the BMEP586040 CPU only when you use the **STANDARD** edition of the UMAC tool.

Access this feature from the list of features (*see page 19*) on the **Convert** screen.

### Adjustment

Adjust the memory allocation:

Step	Action
1	Select <b>Adjust Memory Allocation</b> to adjust the amount of memory that is allotted to each located memory type based on a percentage above the required amount the M580 target application.
2	Click the <b>Adjust</b> button to open the dialog box that displays the different memory types along with the maximum allowed memory locations for the BMEP586040.
3	Adjust the <b>Additional Buffer</b> to change the amount of memory that is configured for a particular memory type. Set this value to 0 to make the <b>To Be Configured</b> value equal to the <b>Required Memory</b> value in the application. You can also set this value to any value that maximizes the <b>To Be Configured</b> value to equal that of the maximum value of the BMEP586040 memory type.
4	Click <b>OK</b> to close the dialog box.

### Examples

The UMAC tool has calculated that the converted M580 project requires 4096 memory words (%MW4096). The buffer is set to its default of 20% so the CPU is configured for 4920 memory words (%MW4920). The %M, %I, and %IW locations use the same process. The **Additional Buffer** is greyed out and a small amount is configured when the maximum used offset is 0.

### Exceptions

For Quantum projects, use %M and %MW values that are in multiples of 128 to successfully build and analyze the Unity project.



---

# Chapter 6

## UMAC Reports

---

### Introduction

This chapter describes the reports that the UMAC tool generates during the conversion process. The **Analysis** report and **Benefit Estimation** report access the legacy project before the conversion. The **Conversion** report is generated after the conversion to show what was converted and what was not.

### What Is in This Chapter?

This chapter contains the following topics:

Topic	Page
Analysis Report	62
Benefit Estimation	63
Conversion Report	64

## Analysis Report

### About the Report

After you open an application in the UMAC tool, you can generate an analysis report. The report is a list of issues that may require your attention before the conversion, such as:

- **Alignment Constraint:** These tables show the number of single and structured variables that are affected by an alignment constraint issue in the application.
- **Hardware Configuration Detail:** M580 applications do not support the hardware elements in this table.
- **System Bits and Words Info:** These tables show the matches and mismatches in the conversion.
- **Initial Value Report:** This table shows the number of variables that have an initial value.
- **FFBs:** This table shows the missing FFBs that are handled by the UMAC tool.

The report also contains the configuration from the source application.

### Generating the Report

Press the **Generate Report** button to view the conversion report. The report is available in these formats:

- **PDF:** Open this report in a PDF reader.
- **HTML:** Open this file in a browser.

Save the report (**File** → **Save As**).

## Benefit Estimation

### Introduction

The **Benefit Estimation** spreadsheet provides an estimate of the man-hours and money that are saved when the application is converted with the UMAC tool.

This report accounts for the number of occurrences that each converter feature handles and multiplies it by a default number of man-hours listed. It then multiplies that total by an hourly rate to generate an estimated savings that could be realized by using the converter. The **Summary** provides a summary of all the worksheets in the **Benefit Estimation** spreadsheet. You can adjust the duration (hours) per feature occurrence and the rate per hour.

### Exceptions

Install Microsoft Excel on the PC that runs the UMAC software.

## Conversion Report

### Setting Up the Report

Before you generate the conversion report, use the **Set Up Header/Footer** button to change the header and footer information on the page. Press this button and enter your information in the **Report header** and **Report footer** fields.

Example:

header	Customer	Company	Your Corporation
		Plant/Site	Springfield Plant 1
		PLC Name	package_machine_3
footer	Actor	Company	My Integration Partners, Inc.

### Generating the Report

Press the **Generate Report** button to view the conversion report. The report is available in these formats:

- **.pdf**: Open this report in a .pdf reader.
- **.html**: Open this file in a browser.



---

# Chapter 7

## Legacy DFBs and PLCSTAT

---

### PLC Status DFB

#### Introduction

Legacy Concept Quantum applications support PLCSTAT, a function block that collects RIO S908 system diagnostics. This function block references system bits and words that are not supported in M580.

A new DFB has therefore been developed for use in M580 applications. The new PLCSTAT DFB is delivered with Control Expert, but it is not part of the installed library. It is located in the Control Expert **Extras** folder on your PC.

The DFB is generic in its imported form. It supports both standalone and redundant architectures but is customized for specific applications. Open the ST section within the DFB and follow the customization instructions.

**NOTE:** Refer to the description of ST sections in the Unity Pro MAST task ([see page 77](#)).

The DFB has an input/output called HSBY\_DDDT. For standalone applications, the UMAC tool user creates a dummy Device DDT of type T\_M\_ECPU\_HSBY. This dummy DDDT is assigned to the HSBY\_DDDT input/output. For redundant applications, the input/output is assigned to the device DDT HSBY\_DDDT that is created when you add a redundant CPU to the configuration.

**NOTE:** Refer to the *EcoStruxure™ Control Expert, UnityLL984, Block Library* for detailed descriptions of the L9\_STAT and L9\_MRTM EFBs.

#### Examples

To use the PLCSTAT DFB, import it into your M580 project from the Control Expert **Extras** folder. (The DIOSTATE, PLCSTATE, and RIOSTATE DDDTs are imported at the same time.) A generic conversion error instruction in the ST section section instructs you to assign the standalone or redundant behavior in the logic.

Insert comment tags for each system type to adjust the logic within the ST section of the DFB.

For standalone projects, comment out the conversion error (convError) instruction along with any logic statements **between** `Second Choice` and `End Second Choice`:

```
(*-- ##### SecondChoice: PLC_STAT.word2 for a Hot Standby Application
##### --*)

(*-- IEC bit 15: set to 1 if hot standby mode --*)
PLC_STAT.word2 := 16#8000;

(*-- IEC bit 14: always 0 --*)
```

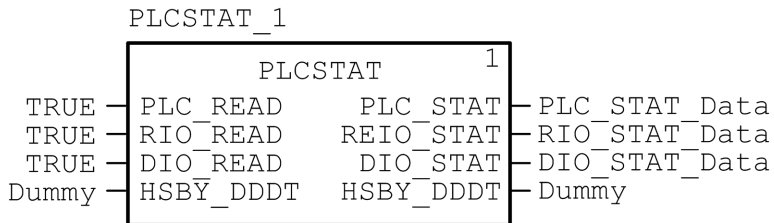
```
(*-- IEC bit 13: IP or IP+1 = TODO --*)
(*-- IEC bit 12: info validity = always valid --*)
PLC_STAT.word2 := PLC_STAT.word2 or 16#1000;
(*-- IEC bit 11,10,9: not used, always 0 --*)
(*-- IEC bit 8: copro fimware mismatch = not significant, always 0
(no copro) --*)
(*-- IEC bit 7: all firmware (CPU, copro, CRP) mismatch --*)
if (HSBY_DDDT.FW_MISMATCH) then
    PLC_STAT.word2 := PLC_STAT.word2 or 16#0080;
end_if;
(*-- IEC bit 6: CPU-sync link status --*)
if (NOT(    HSBY_DDDT.LOCAL_HSBY_STS.HSBY_LINK_ERROR
    OR HSBY_DDDT.REMOTE_HSBY_STS.HSBY_LINK_ERROR)) then
    PLC_STAT.word2 := PLC_STAT.word2 or 16#0040;
end_if;
(*-- IEC bit 5: unit A or B --*)
if (HSBY_DDDT.LOCAL_HSBY_STS.PLC_B) then
    PLC_STAT.word2 := PLC_STAT.word2 or 16#20;
end_if;
(*-- IEC bit 5: unit A or B --*)
if (HSBY_DDDT.LOCAL_HSBY_STS.PLC_B) then
    PLC_STAT.word2 := PLC_STAT.word2 or 16#20;
end_if;
(*-- IEC bit 4: Application mismatch --*)
if (    HSBY_DDDT.APP_MISMATCH
    OR HSBY_DDDT.LOGIC_MISMATCH
    OR HSBY_DDDT.OFFLINE_BUILD_MISMATCH) then
    PLC_STAT.word2 := PLC_STAT.word2 or 16#0010;
end_if;
(*-- IEC bits 3, 2: remote system state --*)
if (    HSBY_DDDT.REMOTE_STS_VALID
    AND HSBY_DDDT.REMOTE_HSBY_STS.RUN_PRIMARY) then
    PLC_STAT.word2 := PLC_STAT.word2 or 16#0008; (*-- 10 = primary
--*)
```

```

elseif (      HSBY_DDDT.REMOTE_STS_VALID
          AND HSBY_DDDT.REMOTE_HSBY_STS.RUN_STANDBY) then
    PLC_STAT.word2 := PLC_STAT.word2 or 16#0008; (*-- 10 = primary
--*)
elseif (      HSBY_DDDT.REMOTE_STS_VALID
          AND HSBY_DDDT.REMOTE_HSBY_STS.RUN_STANDBY) then
    PLC_STAT.word2 := PLC_STAT.word2 or 16#000C; (*-- 11 = standby --*)
else
    PLC_STAT.word2 := PLC_STAT.word2 or 16#0004; (*-- 01 = offline --*)
end_if;
(*-- IEC bits 1, 0: local system state --*)
if (HSBY_DDDT.LOCAL_HSBY_STS.RUN_PRIMARY) then
    PLC_STAT.word2 := PLC_STAT.word2 or 16#0002; (*-- 10 = primary --*)
elseif (HSBY_DDDT.LOCAL_HSBY_STS.RUN_STANDBY) then
    PLC_STAT.word2 := PLC_STAT.word2 or 16#0003; (*-- 11 = standby --*)
else
    PLC_STAT.word2 := PLC_STAT.word2 or 16#0001; (*-- 01 = offline --*)
end_if;
(*-- ##### End Second Choice: PLC_STAT.word2 for a Hot Standby
Application          ##### --*)

```

This is the PLCSTAT DFB for standalone systems in an FBD section:



For redundant projects, comment out the conversion error (convError) statement and any logic statements between First Choice and End First Choice:

```

(* --
#####
##### -- *);

```

```
(* -- ##### PLC_STAT.word2 contains Hot Standby status if application
is Hot Standby typed, ##### -- *);

(* -- ##### otherwise in an Standalone Application, this word do not
have significative ##### -- *);

(* -- ##### value. In this case it is set to a zero value.
##### -- *);

(* -- ##### User of the Converter have to choose between the 2
implementations and keep one ##### -- *);

(* -- ##### (leaving in comment the non-used portion, so available
for a further change) ##### -- *);

{ConvError('PLCSTAT DFB type: PLC_STAT.word2 discussion: Choose one
implementation depending of targeted application, Standalone or Hot
Standby.')}

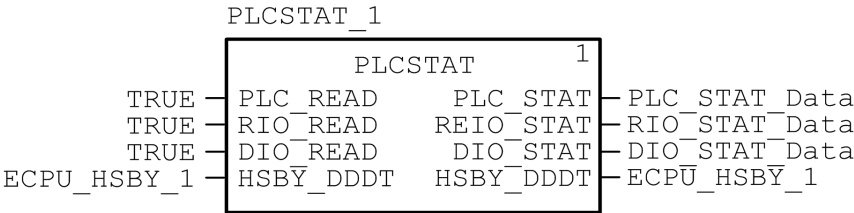
(*-- ##### First Choice: PLC_STAT.word2 for a Standalone Application
##### --*)

(*-- Hot standby Status it is not relevant in a Standalone Application
--*)

PLC_STAT.word2 := 0;

(*-- ##### End First Choice: PLC_STAT.word2 for a Standalone
Application #####
```

This is the PLCSTAT DFB for redundant systems in an FBD section:



This table describes the inputs and outputs of the PLCSTAT DFB:

Pin	Description	
PLC_READ	input	BOOL
RIO_READ	input	BOOL
DIO_READ	input	BOOL
HSBY_DDDT	input/output	HSBY_DDDT is assigned to the redundant DTM that is created when a redundant processor is selected.
PLC_STAT	output	DDT (1-word array)
REIO_STAT	output	DDT (160-word array)
DIO_STAT	output	DDT (106-word array)

---

# Chapter 8

## Redundant Quantum Networks

---

### Redundant Quantum Networks

#### Introduction

All UMAC conversions to M580 are made using the same local M580 configuration. It contains the BME rack, power supply, and BMEP586040 CPU. Converting a redundant Quantum application to M580 is identical to standalone systems except that the user replaces the BMEP586040 CPU in the target application with a BMEH586040 CPU.

A redundant system can include RIO networks, EIO networks, or both. The UMAC tool, therefore, creates two EIO drops made from the two local racks of the source redundant Quantum hardware configuration. The primary and standby CPUs are replaced with the appropriate communication module.

You can connect S908 drops to redundant Quantum networks. The redundancy is retained when you migrate the network to an M580 architecture because the 140CRA31908 EIO adapter module assumes the redundancy tasks of the Quantum CPUs.

**NOTE:** Refer to the *Modicon M580 Hot Standby, System Planning Guide for Frequently Used Architectures*.

#### Process Overview

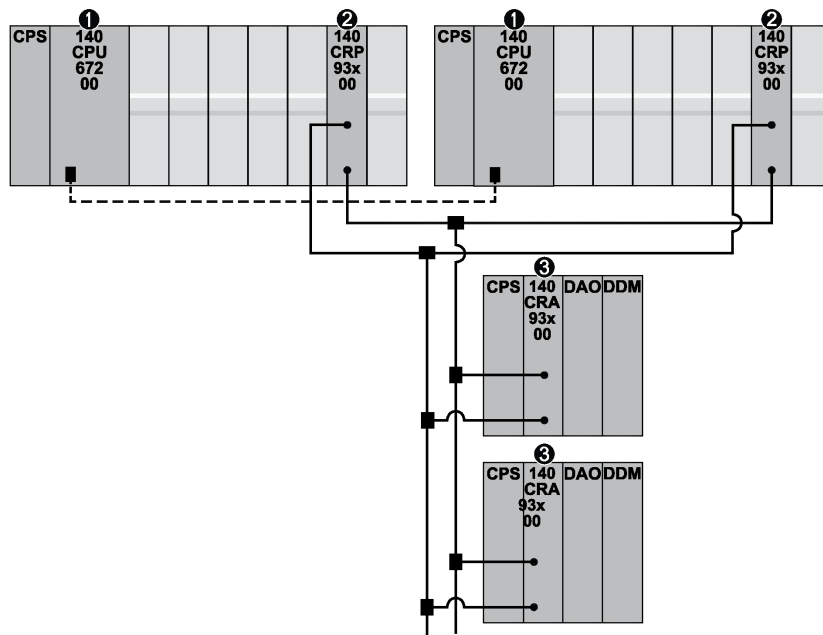
These are the general stages of the process when the UMAC tool migrates a redundant Quantum architecture to a redundant M580 architecture:

Stage	Description
1	<b>Generate a Redundant Quantum RIO Network:</b> Create a hardware configuration that includes Quantum S908 RIO drops.
2	<b>Generate a Redundant Quantum EIO Network:</b> Build a redundant Quantum network that includes both Quantum EIO drops and X80 EIO drops.
3	<b>Complete the Migration:</b> Physically replace the Quantum CPUs in the above example networks with 140CRA31908 adapter modules and add redundant M580 CPUs in local racks.

These stages are described in more detail below.

## Generate a Redundant Quantum RIO Network

This redundant Quantum RIO network is connected to S908 RIO drops. The 140CRP93•00 communications module facilitates S908 communications with the I/O modules in the S908, SY/MAX, and 800 Series RIO drops:



- 1 Quantum primary and standby CPUs on local racks with a fiber-optic link
- 2 140CRP93•00 communications modules with redundant connections to the S908 RIO drops
- 3 140CRA93•00 adapter module on an S908 drop

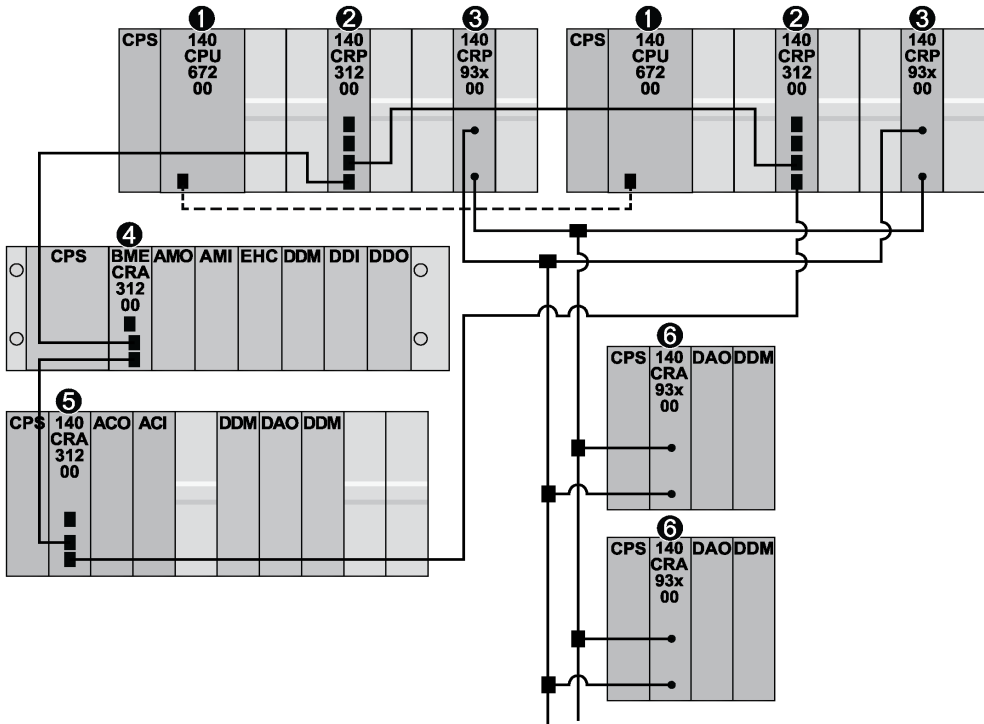
**NOTE:** The configuration of each Quantum CPU is identical. For information about redundant Quantum networks, refer to the *Modicon Quantum Hot Standby System User Manual*.

### Generate a Redundant Quantum EIO Network

Add X80 EIO drops to the redundant network (above):

Step	Action
1	Add a 140CRP31200 communications module to each local rack in the redundant Quantum network.
2	Add a Quantum X80 EIO drop to the main ring.
3	Add an M580 X80 EIO drop to the main ring.
4	Build the Unity Pro application and download it to the Quantum CPUs.

**Result:** The 140CRP31200 modules (not the CPUs) connect the local rack to the Quantum main ring to facilitate Quantum EIO communications with the X80 EIO drops:



- 1 Quantum primary and standby CPUs on local racks with a fiber-optic link
- 2 140CRP31200 communications modules
- 3 140CRP93•00 adapter modules with redundant connections to the S908 RIO drops
- 4 BMECRA31200 adapter module on an M580 X80 EIO drop
- 5 140CRA31200 adapter module on a Quantum X80 EIO drop
- 6 140CRA93•00 adapter module on an S908 RIO drop

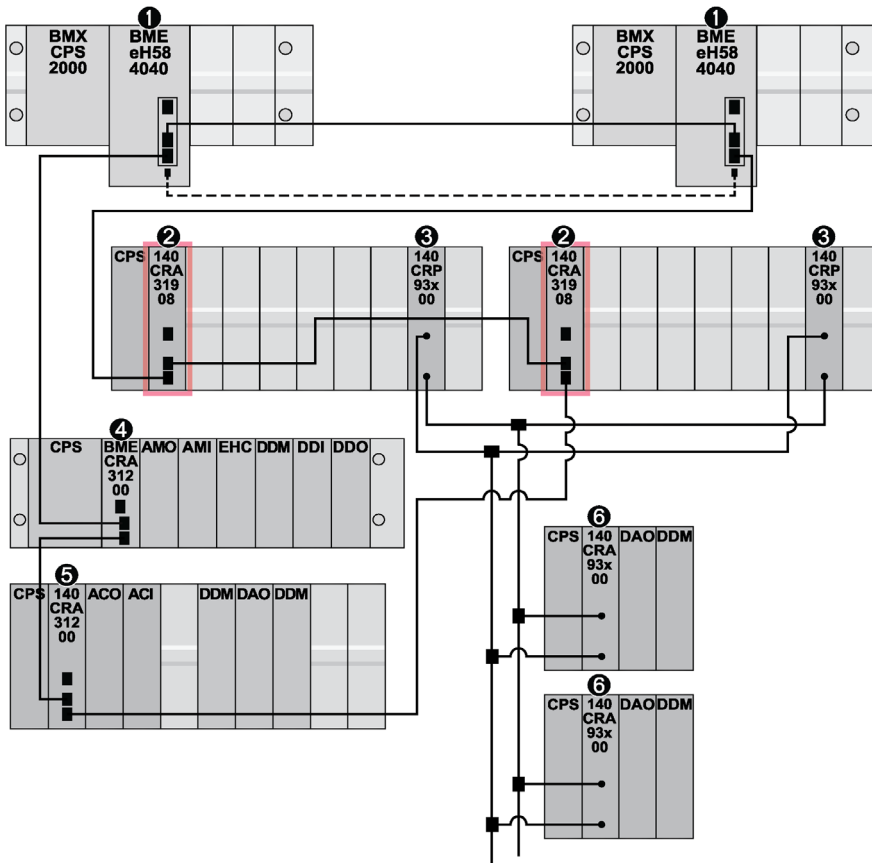
**Complete the Migration**

Complete the migration of the redundant Quantum network (shown above) to an M580 architecture:

Step	Action
1	Remove the Quantum CPUs from the racks in the reverse order of their installation.
2	Place the 140CRA31908 adapter modules in the slots from which you removed the CPUs.
3	Add redundant M580 CPUs in the local racks. <b>NOTE:</b> The configurations of the redundant M580 CPUs is identical.
4	Connect the main ring to the M580 CPUs.
5	Remove the 140CRP31200 communications modules.
6	Modify your Unity Pro application as necessary.
7	Download the Unity Pro application to the M580 CPUs.



**Result:** A 140CRA31908 EIO adapter module (not a Quantum CPU) connects to the main ring. The M580 CPUs now manage the network and the 140CRA31908 modules:



- 1 M580 primary and standby CPUs on local racks with a fiber-optic link
- 2 140CRA31908 EIO adapter modules
- 3 140CRP93-00 communications modules
- 4 BMECRA31200 adapter module on an M580 X80 EIO drop
- 5 140CRA31200 adapter module on a Quantum X80 EIO drop
- 6 140CRA93-00 adapter module on an S908 RIO drop

## Reference Topics

Refer to these discussions of redundant networks in the *Modicon Quantum 40CRA31908 EIO Adapter Module Installation and Configuration Guide*:

- switchover
- master 140CRA31908 module selection
- EIO support



---

# Appendices

---





---

# Appendix A

## ST Sections

---

**What Is in This Chapter?**

This chapter contains the following topics:

Topic	Page
ST Sections Examples	78
Detected Errors in ST Sections	83

## ST Sections Examples

### Introduction

In some cases, the UMAC software adds ST (structured text) sections to the converted Unity Pro application.

### Input I/O Direct Assignment

This sample code represents the input I/O direct assignment in an ST section:

```
(*ST program generation for module 140DAI34000*)
```

```
%I1 := WRITE_INPUT_EBOOL (INP :=  
PLC0_d0_r0_s2_DAI1602.DIS_CH_IN[0].VALUE);  
%I2 := WRITE_INPUT_EBOOL (INP :=  
PLC0_d0_r0_s2_DAI1602.DIS_CH_IN[1].VALUE);  
%I3 := WRITE_INPUT_EBOOL (INP :=  
PLC0_d0_r0_s2_DAI1602.DIS_CH_IN[2].VALUE);  
%I4 := WRITE_INPUT_EBOOL (INP :=  
PLC0_d0_r0_s2_DAI1602.DIS_CH_IN[3].VALUE);  
%I5 := WRITE_INPUT_EBOOL (INP :=  
PLC0_d0_r0_s2_DAI1602.DIS_CH_IN[4].VALUE);  
%I6 := WRITE_INPUT_EBOOL (INP :=  
PLC0_d0_r0_s2_DAI1602.DIS_CH_IN[5].VALUE);  
%I7 := WRITE_INPUT_EBOOL (INP :=  
PLC0_d0_r0_s2_DAI1602.DIS_CH_IN[6].VALUE);  
%I8 := WRITE_INPUT_EBOOL (INP :=  
PLC0_d0_r0_s2_DAI1602.DIS_CH_IN[7].VALUE);  
%I9 := WRITE_INPUT_EBOOL (INP :=  
PLC0_d0_r0_s2_DAI1602.DIS_CH_IN[8].VALUE);  
%I10 := WRITE_INPUT_EBOOL (INP :=  
PLC0_d0_r0_s2_DAI1602.DIS_CH_IN[9].VALUE);  
%I11 := WRITE_INPUT_EBOOL (INP :=  
PLC0_d0_r0_s2_DAI1602.DIS_CH_IN[10].VALUE);  
%I12 := WRITE_INPUT_EBOOL (INP :=  
PLC0_d0_r0_s2_DAI1602.DIS_CH_IN[11].VALUE);  
%I13 := WRITE_INPUT_EBOOL (INP :=  
PLC0_d0_r0_s2_DAI1602.DIS_CH_IN[12].VALUE);  
%I14 := WRITE_INPUT_EBOOL (INP :=  
PLC0_d0_r0_s2_DAI1602.DIS_CH_IN[13].VALUE);  
%I15 := WRITE_INPUT_EBOOL (INP :=  
PLC0_d0_r0_s2_DAI1602.DIS_CH_IN[14].VALUE);  
%I16 := WRITE_INPUT_EBOOL (INP :=  
PLC0_d0_r0_s2_DAI1602.DIS_CH_IN[15].VALUE);
```

## (\*ST program generation for module 140DAI44000\*)

```
%I17 := WRITE_INPUT_EBOOL (INP :=  
PLC0_d0_r0_s3_DAI1603.DIS_CH_IN[0].VALUE);  
  
%I18 := WRITE_INPUT_EBOOL (INP :=  
PLC0_d0_r0_s3_DAI1603.DIS_CH_IN[1].VALUE);  
  
%I19 := WRITE_INPUT_EBOOL (INP :=  
PLC0_d0_r0_s3_DAI1603.DIS_CH_IN[2].VALUE);  
  
%I20 := WRITE_INPUT_EBOOL (INP :=  
PLC0_d0_r0_s3_DAI1603.DIS_CH_IN[3].VALUE);  
  
%I21 := WRITE_INPUT_EBOOL (INP :=  
PLC0_d0_r0_s3_DAI1603.DIS_CH_IN[4].VALUE);  
  
%I22 := WRITE_INPUT_EBOOL (INP :=  
PLC0_d0_r0_s3_DAI1603.DIS_CH_IN[5].VALUE);  
  
%I23 := WRITE_INPUT_EBOOL (INP :=  
PLC0_d0_r0_s3_DAI1603.DIS_CH_IN[6].VALUE);  
  
%I24 := WRITE_INPUT_EBOOL (INP :=  
PLC0_d0_r0_s3_DAI1603.DIS_CH_IN[7].VALUE);  
  
%I25 := WRITE_INPUT_EBOOL (INP :=  
PLC0_d0_r0_s3_DAI1603.DIS_CH_IN[8].VALUE);  
  
%I26 := WRITE_INPUT_EBOOL (INP :=  
PLC0_d0_r0_s3_DAI1603.DIS_CH_IN[9].VALUE);  
  
%I27 := WRITE_INPUT_EBOOL (INP :=  
PLC0_d0_r0_s3_DAI1603.DIS_CH_IN[10].VALUE);  
  
%I28 := WRITE_INPUT_EBOOL (INP :=  
PLC0_d0_r0_s3_DAI1603.DIS_CH_IN[11].VALUE);  
  
%I29 := WRITE_INPUT_EBOOL (INP :=  
PLC0_d0_r0_s3_DAI1603.DIS_CH_IN[12].VALUE);  
  
%I30 := WRITE_INPUT_EBOOL (INP :=  
PLC0_d0_r0_s3_DAI1603.DIS_CH_IN[13].VALUE);  
  
%I31 := WRITE_INPUT_EBOOL (INP :=  
PLC0_d0_r0_s3_DAI1603.DIS_CH_IN[14].VALUE);  
  
%I32 := WRITE_INPUT_EBOOL (INP :=  
PLC0_d0_r0_s3_DAI1603.DIS_CH_IN[15].VALUE);
```

## Output I/O Direct Assignment

This sample code represents the output I/O direct assignment in an ST section:

```
(*ST program generation for module 140DAO84000*)
PLC0_d0_r1_s5_DAO1605.DIS_CH_OUT[0].VALUE := %M1;
PLC0_d0_r1_s5_DAO1605.DIS_CH_OUT[1].VALUE := %M2;
PLC0_d0_r1_s5_DAO1605.DIS_CH_OUT[2].VALUE := %M3;
PLC0_d0_r1_s5_DAO1605.DIS_CH_OUT[3].VALUE := %M4;
PLC0_d0_r1_s5_DAO1605.DIS_CH_OUT[4].VALUE := %M5;
PLC0_d0_r1_s5_DAO1605.DIS_CH_OUT[5].VALUE := %M6;
PLC0_d0_r1_s5_DAO1605.DIS_CH_OUT[6].VALUE := %M7;
PLC0_d0_r1_s5_DAO1605.DIS_CH_OUT[7].VALUE := %M8;
PLC0_d0_r1_s5_DAO1605.DIS_CH_OUT[8].VALUE := %M9;
PLC0_d0_r1_s5_DAO1605.DIS_CH_OUT[9].VALUE := %M10;
PLC0_d0_r1_s5_DAO1605.DIS_CH_OUT[10].VALUE := %M11;
PLC0_d0_r1_s5_DAO1605.DIS_CH_OUT[11].VALUE := %M12;
PLC0_d0_r1_s5_DAO1605.DIS_CH_OUT[12].VALUE := %M13;
PLC0_d0_r1_s5_DAO1605.DIS_CH_OUT[13].VALUE := %M14;
PLC0_d0_r1_s5_DAO1605.DIS_CH_OUT[14].VALUE := %M15;
PLC0_d0_r1_s5_DAO1605.DIS_CH_OUT[15].VALUE := %M16;
```

(\*ST program generation for module 140DAO84010\*)

```
PLC0_d0_r1_s6_DAO1605.DIS_CH_OUT[0].VALUE := %M17;
PLC0_d0_r1_s6_DAO1605.DIS_CH_OUT[1].VALUE := %M18;
PLC0_d0_r1_s6_DAO1605.DIS_CH_OUT[2].VALUE := %M19;
PLC0_d0_r1_s6_DAO1605.DIS_CH_OUT[3].VALUE := %M20;
PLC0_d0_r1_s6_DAO1605.DIS_CH_OUT[4].VALUE := %M21;
PLC0_d0_r1_s6_DAO1605.DIS_CH_OUT[5].VALUE := %M22;
PLC0_d0_r1_s6_DAO1605.DIS_CH_OUT[6].VALUE := %M23;
PLC0_d0_r1_s6_DAO1605.DIS_CH_OUT[7].VALUE := %M24;
PLC0_d0_r1_s6_DAO1605.DIS_CH_OUT[8].VALUE := %M25;
PLC0_d0_r1_s6_DAO1605.DIS_CH_OUT[9].VALUE := %M26;
PLC0_d0_r1_s6_DAO1605.DIS_CH_OUT[10].VALUE := %M27;
PLC0_d0_r1_s6_DAO1605.DIS_CH_OUT[11].VALUE := %M28;
PLC0_d0_r1_s6_DAO1605.DIS_CH_OUT[12].VALUE := %M29;
PLC0_d0_r1_s6_DAO1605.DIS_CH_OUT[13].VALUE := %M30;
PLC0_d0_r1_s6_DAO1605.DIS_CH_OUT[14].VALUE := %M31;
PLC0_d0_r1_s6_DAO1605.DIS_CH_OUT[15].VALUE := %M32;
```



## PULL Section

This code is an example of a Pull section at the beginning of the MAST task:

```
Pull_Struct_ANALOG_BW(IN := %MW005930:10, OUT => LT6210Scada);
Pull_Struct_ANALOG_BW(IN := %MW005380:10, OUT => LT5212Scada);
Pull_Struct_ANALOG_BW(IN := %MW005390:10, OUT => LT5222Scada);
Pull_Struct_ANALOG_BW(IN := %MW005910:10, OUT => PT6115Scada);
Pull_Struct_ANALOG_BW(IN := %MW005920:10, OUT => TT6120Scada);
Pull_Struct_ANALOG_BW(IN := %MW005990:10, OUT => PT9405AScada);
Pull_Struct_ANALOG_BW(IN := %MW005900:10, OUT => CT9698Scada);
Pull_Struct_ANALOG_BW(IN := %MW005890:10, OUT => OT6111Scada);
Pull_Struct_ANALOG_BW(IN := %MW005970:10, OUT => WT9313Scada);
Pull_Struct_ANALOG_BW(IN := %MW006010:10, OUT => PT9405BScada);
Pull_Struct_ANALOG_BW(IN := %MW005980:10, OUT => FT9404AScada);
Pull_Struct_ANALOG_BW(IN := %MW006000:10, OUT => FT9404BScada);
Pull_Struct_ANALOG_BW(IN := %MW005770:10, OUT => TT5304Scada);
Pull_Struct_ANALOG_BW(IN := %MW006260:10, OUT => PT5431Scada);
Pull_Struct_ANALOG_BW(IN := %MW006280:10, OUT => PT5441Scada);
Pull_Struct_ANALOG_BW(IN := %MW005810:10, OUT => JT8640Scada);
Pull_Struct_ANALOG_BW(IN := %MW006300:10, OUT => PT5451Scada);
Pull_Struct_ANALOG_BW(IN := %MW005880:10, OUT => PT8319Scada);
Pull_Struct_ANALOG_BW(IN := %MW005850:10, OUT => CT9699Scada);
Pull_Struct_ANALOG_BW(IN := %MW005860:10, OUT => OT6310Scada);
Pull_Struct_ANALOG_BW(IN := %MW005750:10, OUT => OT5301Scada);
Pull_Struct_ANALOG_BW(IN := %MW005820:10, OUT => FT4141Scada);
Pull_Struct_ANALOG_BW(IN := %MW005780:10, OUT => BT5310Scada);
Pull_Struct_ANALOG_BW(IN := %MW005870:10, OUT => BT6325Scada);
Pull_Struct_ANALOG_BW(IN := %MW005790:10, OUT => PT5306Scada);
Pull_Struct_ANALOG_BW(IN := %MW005760:10, OUT => FT5302Scada);
Pull_Struct_ANALOG_BW(IN := %MW006160:10, OUT => FT5311Scada);
Pull_Struct_ANALOG_BW(IN := %MW005800:10, OUT => FT9175Scada);
Pull_Struct_ANALOG_BW(IN := %MW006220:10, OUT => FT9110Scada);
Pull_Struct_ANALOG_BW(IN := %MW006290:10, OUT => FT5450Scada);
Pull_Struct_ANALOG_BW(IN := %MW006230:10, OUT => FT9111Scada);
Pull_Struct_ANALOG_BW(IN := %MW005430:10, OUT => FT7740Scada);
Pull_Struct_ANALOG_BW(IN := %MW005420:10, OUT => LT7110Scada);
Pull_Struct_ANALOG_BW(IN := %MW006240:10, OUT => FT9112Scada);
Pull_Struct_ANALOG_BW(IN := %MW006250:10, OUT => FT5430Scada);
Pull_Struct_ANALOG_BW(IN := %MW006270:10, OUT => FT5440Scada);
```

## Push Section


This code is an example of a Push section at the end of the MAST task:

```
Push_Struct_ANALOG_BW(IN := LT5212Scada, OUT => %MW005380:10);
Push_Struct_ANALOG_BW(IN := LT5222Scada, OUT => %MW005390:10);
Push_Struct_ANALOG_BW(IN := FT6480Scada, OUT => %MW005410:10);
Push_Struct_ANALOG_BW(IN := AT5330Scada, OUT => %MW005400:10);
Push_Struct_ANALOG_BW(IN := PT6115Scada, OUT => %MW005910:10);
Push_Struct_ANALOG_BW(IN := CT9698Scada, OUT => %MW005900:10);
Push_Struct_ANALOG_BW(IN := TT6120Scada, OUT => %MW005920:10);
Push_Struct_ANALOG_BW(IN := PT9405AScada, OUT => %MW005990:10);
Push_Struct_ANALOG_BW(IN := WT9313Scada, OUT => %MW005970:10);
Push_Struct_ANALOG_BW(IN := WT9313Scada, OUT =>
%MW005970:10);Push_Struct_ANALOG_BW(IN := PT9405BScada, OUT =>
%MW006010:10);

Push_Struct_ANALOG_BW(IN := OT6111Scada, OUT => %MW005890:10);
Push_Struct_ANALOG_BW(IN := FT9404AScada, OUT => %MW005980:10);
Push_Struct_ANALOG_BW(IN := FT9404BScada, OUT => %MW006000:10);
Push_Struct_ANALOG_BW(IN := FT9175Scada, OUT => %MW005800:10);
Push_Struct_ANALOG_BW(IN := TT5304Scada, OUT => %MW005770:10);
Push_Struct_ANALOG_BW(IN := FT9110Scada, OUT => %MW006220:10);
Push_Struct_ANALOG_BW(IN := FT9111Scada, OUT => %MW006230:10);
Push_Struct_ANALOG_BW(IN := FT9112Scada, OUT => %MW006240:10);
Push_Struct_ANALOG_BW(IN := FT5430Scada, OUT => %MW006250:10);
Push_Struct_ANALOG_BW(IN := PT5431Scada, OUT => %MW006260:10);
Push_Struct_ANALOG_BW(IN := FT5440Scada, OUT => %MW006270:10);
Push_Struct_ANALOG_BW(IN := PT5441Scada, OUT => %MW006280:10);
Push_Struct_ANALOG_BW(IN := JT8640Scada, OUT => %MW005810:10);
Push_Struct_ANALOG_BW(IN := FT5450Scada, OUT => %MW006290:10);
Push_Struct_ANALOG_BW(IN := PT5451Scada, OUT => %MW006300:10);
Push_Struct_ANALOG_BW(IN := PT8319Scada, OUT => %MW005880:10);
Push_Struct_ANALOG_BW(IN := CT9699Scada, OUT => %MW005850:10);
Push_Struct_ANALOG_BW(IN := OT6310Scada, OUT => %MW005860:10);
Push_Struct_ANALOG_BW(IN := OT5301Scada, OUT => %MW005750:10);
Push_Struct_ANALOG_BW(IN := FT4141Scada, OUT => %MW005820:10);
Push_Struct_ANALOG_BW(IN := BT5310Scada, OUT => %MW005780:10);
Push_Struct_ANALOG_BW(IN := BT6325Scada, OUT => %MW005870:10);
Push_Struct_ANALOG_BW(IN := PT5306Scada, OUT => %MW005790:10);
```

## Detected Errors in ST Sections

### Addressing Detected Errors

 **WARNING**

**UNINTENDED OPERATION AT START-UP**  
Delete the `ReadMe_First` and `convError` ST sections before you build the Unity Pro program.  
**Failure to follow these instructions can result in death, serious injury, or equipment damage.**

Some ST sections report conditions that may generate errors when you analyze or build your Unity Pro project. Use these instructions to address that logic:

Step	Action
1	Open the converted Unity Pro application.
2	View the new ST sections by expanding the <b>Project Browser</b> navigation tree: <b>Project → Program → Tasks → Sections</b>
3	Double-click an individual ST section to read its contents.
4	For ST sections that contain conversion errors ( <code>convError</code> ), address the logic on a case-by-case basis.
5	Delete the ST section.

### Unsupported Modules

For each unsupported module, the UMAC tool inserts an ST section called **Unsupported\_Modules** into the MAST task.

Refer to the lists of unsupported Quantum modules (*see page 35*) and unsupported Premium modules (*see page 27*).





## !

**%I**

According to the CEI standard, %I indicates a language object of type discrete IN.

**%IW**

According to the CEI standard, %IW indicates a language object of type analog IN.

**%M**

According to the CEI standard, %M indicates a language object of type memory bit.

**%MW**

According to the CEI standard, %MW indicates a language object of type memory word.

**%Q**

According to the CEI standard, %Q indicates a language object of type discrete OUT.

**%QW**

According to the CEI standard, %QW indicates a language object of type analog OUT.

**%SW**

According to the CEI standard, %SW indicates a language object of type system word.

## A

### **adapter**

An adapter is the target of real-time I/O data connection requests from scanners. It cannot send or receive real-time I/O data unless it is configured to do so by a scanner, and it does not store or originate the data communications parameters necessary to establish the connection. An adapter accepts explicit message requests (connected and unconnected) from other devices.

### **advanced mode**

In Control Expert, advanced mode is a selection that displays expert-level configuration properties that help define Ethernet connections. Because these properties should be edited only by people with a good understanding of EtherNet/IP communication protocols, they can be hidden or displayed, depending upon the qualifications of the specific user.

### **applicative time stamping**

Use the applicative time stamping solution to access time stamp event buffers with a SCADA system that does not support the OPC DA interface. In this case, function blocks in the Control Expert PLC application read events in the buffer and formats them to be sent to the SCADA system.

**architecture**

Architecture describes a framework for the specification of a network that is constructed of these components:

- physical components and their functional organization and configuration
- operational principles and procedures
- data formats used in its operation

**ARRAY**

An **ARRAY** is a table containing elements of a single type. This is the syntax: **ARRAY** [**<limits>**] **OF** **<Type>**

**Example:** **ARRAY** [1..2] **OF** **BOOL** is a one-dimensional table with two elements of type **BOOL**.

**ARRAY** [1..10, 1..20] **OF** **INT** is a two-dimensional table with 10x20 elements of type **INT**.

**ART**

(*application response time*) The time a CPU application takes to react to a given input. **ART** is measured from the time a physical signal in the CPU turns on and triggers a write command until the remote output turns on to signify that the data has been received.

**AUX**

An (**AUX**) task is an optional, periodic processor task that is run through its programming software. The **AUX** task is used to execute a part of the application requiring a low priority. This task is executed only if the **MAST** and **FAST** tasks have nothing to execute. The **AUX** task has two sections:

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

**B****BCD**

(*binary-coded decimal*) Binary encoding of decimal numbers.

**BOOL**

(*boolean type*) This is the basic data type in computing. A **BOOL** variable can have either of these values: 0 (**FALSE**) or 1 (**TRUE**).

A bit extracted from a word is of type **BOOL**, for example: %MW10.4.

**BOOTP**

(*bootstrap protocol*) A UDP network protocol that can be used by a network client to automatically obtain an IP address from a server. The client identifies itself to the server using its MAC address. The server, which maintains a pre-configured table of client device MAC addresses and associated IP addresses, sends the client its defined IP address. The **BOOTP** service utilizes UDP ports 67 and 68.

**broadcast**

A message sent to all devices in a broadcast domain.

## C

### **CCOTF**

(*change configuration on the fly*) A feature of Control Expert that allows a module hardware change in the system configuration while the system is operating. This change does not impact active operations.

### **CIP™**

(*common industrial protocol*) A comprehensive suite of messages and services for the collection of manufacturing automation applications (control, safety, synchronization, motion, configuration and information). CIP allows users to integrate these manufacturing applications with enterprise-level Ethernet networks and the internet. CIP is the core protocol of EtherNet/IP.

### **class 1 connection**

A CIP transport class 1 connection used for I/O data transmission via implicit messaging between EtherNet/IP devices.

### **class 3 connection**

A CIP transport class 3 connection used for explicit messaging between EtherNet/IP devices.

### **connected messaging**

In EtherNet/IP, connected messaging uses a CIP connection for communication. A connected message is a logical relationship between two or more application objects on different nodes. The connection establishes a virtual circuit in advance for a particular purpose, such as frequent explicit messages or real-time I/O data transfers.

### **connection**

A virtual circuit between two or more network devices, created prior to the transmission of data. After a connection is established, a series of data is transmitted over the same communication path, without the need to include routing information, including source and destination address, with each piece of data.

### **connection originator**

The EtherNet/IP network node that initiates a connection request for I/O data transfer or explicit messaging.

### **connectionless**

Describes communication between two network devices, whereby data is sent without prior arrangement between the two devices. Each piece of transmitted data also includes routing information, including source and destination address.

### **control network**

An Ethernet-based network containing PACs, SCADA systems, an NTP server, PCs, AMS, switches, etc. Two kinds of topologies are supported:

- flat: All modules and devices in this network belong to same subnet.
- 2 levels: The network is split into an operation network and an inter-controller network. These two networks can be physically independent, but are generally linked by a routing device.

**CPU**

(*central processing unit*) The CPU, also known as the processor or controller, is the brain of an industrial manufacturing process. It automates a process as opposed to relay control systems. CPUs are computers suited to survive the harsh conditions of an industrial environment.

**D****DDT**

(*derived data type*) A derived data type is a set of elements with the same type (`ARRAY`) or with different types (structure).

**determinism**

For a defined application and architecture, you can predict that the delay between an event (change of value of an input) and the corresponding change of a controller output is a finite time  $t$ , smaller than the deadline required by your process.

**Device DDT (DDDT)**

A Device DDT is a DDT predefined by the manufacturer and not modifiable by user. It contains the I/O language elements of an I/O module.

**device network**

An Ethernet-based network within a remote I/O network that contains both remote I/O and distributed I/O devices. Devices connected on this network follow specific rules to allow remote I/O determinism.

**device network**

An Ethernet-based network within an RIO network that contains both RIO and distributed equipment. Devices connected on this network follow specific rules to allow RIO determinism.

**DFB**

(*derived function block*) DFB types are function blocks that can be defined by the user in ST, IL, LD or FBD language.

Using these DFB types in an application makes it possible to:

- simplify the design and entry of the program
- make the program easier to read
- make it easier to debug
- reduce the amount of code generated

**DHCP**

(*dynamic host configuration protocol*) An extension of the BOOTP communications protocol that provides for the automatic assignment of IP addressing settings, including IP address, subnet mask, gateway IP address, and DNS server names. DHCP does not require the maintenance of a table identifying each network device. The client identifies itself to the DHCP server using either its MAC address, or a uniquely assigned device identifier. The DHCP service utilizes UDP ports 67 and 68.



**DIO**

(*distributed I/O*) Also known as distributed equipment. DRSs use DIO ports to connect distributed equipment.

**DIO cloud**

A group of distributed equipment that is not required to support RSTP. DIO clouds require only a single (non-ring) copper wire connection. They can be connected to some of the copper ports on DRSs, or they can be connected directly to the CPU or Ethernet communications modules in the *local rack*. DIO clouds **cannot** be connected to *sub-rings*.

**DIO network**

A network containing distributed equipment, in which I/O scanning is performed by a CPU with DIO scanner service on the local rack. DIO network traffic is delivered after RIO traffic, which takes priority in an RIO network.

**distributed equipment**

Any Ethernet device (Schneider Electric device, PC, servers, or third-party devices) that supports exchange with a CPU or other Ethernet I/O scanner service.

**DNS**

(*domain name server/service*) A service that translates an alpha-numeric domain name into an IP address, the unique identifier of a device on the network.

**domain name**

An alpha-numeric string that identifies a device on the internet, and which appears as the primary component of a web site's uniform resource locator (URL). For example, the domain name *schneider-electric.com* is the primary component of the URL *www.schneider-electric.com*.

Each domain name is assigned as part of the domain name system, and is associated with an IP address.

Also called a host name.

**DRS**

(*dual-ring switch*) A ConneXium extended managed switch that has been configured to operate on an Ethernet network. Predefined configuration files are provided by Schneider Electric to downloaded to a DRS to support the special features of the main ring / sub-ring architecture.

**DSCP**

(*differentiated service code points*) This 6-bit field is in the header of an IP packet to classify and prioritize traffic.

**DST**

(*daylight saving time*) DST is also called *summer time* and is a practice consisting of adjusting forward the clock near the start of spring and adjusting it backward near the start of autumn.

DT

(*date and time*) The DT type, encoded in BCD in a 64-bit format, contains this information:

- the year encoded in a 16-bit field
- the month encoded in an 8-bit field
- the day encoded in an 8-bit field
- the time encoded in an 8-bit field
- the minutes encoded in an 8-bit field
- the seconds encoded in an 8-bit field

**NOTE:** The eight least significant bits are not used.

The DT type is entered in this format:

**DT#**<Year>-<Month>-<Day>-<Hour>:<Minutes>:<Seconds>

This table shows the upper/lower limits of each field:

Field	Limits	Comment
Year	[1990,2099]	Year
Month	[01,12]	The leading 0 is displayed; it can be omitted during data entry.
Day	[01,31]	For months 01/03/05/07/08/10/12
	[01,30]	For months 04/06/09/11
	[01,29]	For month 02 (leap years)
	[01,28]	For month 02 (non-leap years)
Hour	[00,23]	The leading 0 is displayed; it can be omitted during data entry.
Minute	[00,59]	The leading 0 is displayed; it can be omitted during data entry.
Second	[00,59]	The leading 0 is displayed; it can be omitted during data entry.

DTM

(*device type manager*) A DTM is a device driver running on the host PC. It provides a unified structure for accessing device parameters, configuring and operating the devices, and troubleshooting devices. DTMs can range from a simple graphical user interface (GUI) for setting device parameters to a highly sophisticated application capable of performing complex real-time calculations for diagnosis and maintenance purposes. In the context of a DTM, a device can be a communications module or a remote device on the network.

See FDT.

E

EDS

(*electronic data sheet*) EDS are simple text files that describe the configuration capabilities of a device. EDS files are generated and maintained by the manufacturer of the device.

**EF**

(*elementary function*) This is a block used in a program which performs a predefined logical function.

A function does not have any information on the internal state. Several calls to the same function using the same input parameters will return the same output values. You will find information on the graphic form of the function call in the [*functional block (instance)*]. Unlike a call to a function block, function calls include only an output which is not named and whose name is identical to that of the function. In FBD, each call is indicated by a unique [number] via the graphic block. This number is managed automatically and cannot be modified.

Position and configure these functions in your program to execute your application.

You can also develop other functions using the SDKC development kit.

**EFB**

(*elementary function block*) This is a block used in a program which performs a predefined logical function.

EFBs have states and internal parameters. Even if the inputs are identical, the output values may differ. For example, a counter has an output indicating that the preselection value has been reached. This output is set to 1 when the current value is equal to the preselection value.

**EIO network**

(*Ethernet I/O*) An Ethernet-based network that contains three types of devices:

- local rack
- X80 remote drop (using a BM•CRA312•0 adapter module), or a BMENOS0300 network option switch module
- ConneXium extended dual-ring switch (DRS)

**NOTE:** Distributed equipment may also participate in an Ethernet I/O network via connection to DRSs or the service port of X80 remote modules.

**EN**

**EN** stands for **EN**able; it is an optional block input. When the **EN** input is enabled, an **ENO** output is set automatically.

If **EN** = 0, the block is not enabled; its internal program is not executed, and **ENO** is set to 0.

If **EN** = 1, the block's internal program is run and **ENO** is set to 1. If a runtime error is detected, **ENO** is set to 0.

If the **EN** input is not connected, it is set automatically to 1.

**ENO**

**ENO** stands for **Error NOT**ification; this is the output associated with the optional input **EN**.

If **ENO** is set to 0 (either because **EN** = 0 or if a runtime error is detected):

- The status of the function block outputs remains the same as it was during the previous scanning cycle that executed correctly.
- The output(s) of the function, as well as the procedures, are set to 0.

**Ethernet**

A 10 Mb/s, 100 Mb/s, or 1 Gb/s, CSMA/CD, frame-based LAN that can run over copper twisted pair or fiber optic cable, or wireless. The IEEE standard 802.3 defines the rules for configuring a wired Ethernet network; the IEEE standard 802.11 defines the rules for configuring a wireless Ethernet network. Common forms include 10BASE-T, 100BASE-TX, and 1000BASE-T, which can utilize category 5e copper twisted pair cables and RJ45 modular connectors.

**Ethernet DIO scanner service**

This embedded DIO scanner service of M580 CPUs manages distributed equipment on an M580 device network.

**Ethernet I/O scanner service**

This embedded Ethernet I/O scanner service of M580 CPUs manages distributed equipment **and** RIO drops on an M580 device network.

**EtherNet/IP™**

A network communication protocol for industrial automation applications that combines the standard internet transmission protocols of TCP/IP and UDP with the application layer common industrial protocol (CIP) to support both high speed data exchange and industrial control. EtherNet/IP employs electronic data sheets (EDS) to classify each network device and its functionality.

**explicit messaging**

TCP/IP-based messaging for Modbus TCP and EtherNet/IP. It is used for point-to-point, client/server messages that include both data, typically unscheduled information between a client and a server, and routing information. In EtherNet/IP, explicit messaging is considered class 3 type messaging, and can be connection-based or connectionless.

**explicit messaging client**

(*explicit messaging client class*) The device class defined by the ODVA for EtherNet/IP nodes that only support explicit messaging as a client. HMI and SCADA systems are common examples of this device class.

**F****FAST**

A FAST task is an optional, periodic processor task that identifies high priority, multiple scan requests, which is run through its programming software. A FAST task can schedule selected I/O modules to have their logic solved more than once per scan. The FAST task has two sections:

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

**FBD**

(*function block diagram*) An IEC 61131-3 graphical programming language that works like a flowchart. By adding simple logical blocks (AND, OR, etc.), each function or function block in the program is represented in this graphical format. For each block, the inputs are on the left and the outputs on the right. Block outputs can be linked to inputs of other blocks to create complex expressions.

**FDR**

(*fast device replacement*) A service that uses configuration software to replace an inoperable product.

**FDT**

(*field device tool*) The technology that harmonizes communication between field devices and the system host.

**FTP**

(*file transfer protocol*) A protocol that copies a file from one host to another over a TCP/IP-based network, such as the internet. FTP uses a client-server architecture as well as separate control and data connections between the client and server.

**full duplex**

The ability of two networked devices to independently and simultaneously communicate with each other in both directions.

**function block diagram**

See FBD.

## G

**gateway**

A gateway device interconnects two different networks, sometimes through different network protocols. When it connects networks based on different protocols, a gateway converts a datagram from one protocol stack into the other. When used to connect two IP-based networks, a gateway (also called a router) has two separate IP addresses, one on each network.

**GPS**

(*global positioning system*) The GPS standard consists of a space-based positioning, navigation, and timing signals delivered worldwide for civil and military use. Standard positioning service performance depends on satellite broadcast signal parameters, GPS constellation design, the number of satellites in sight, and various environmental parameters.

## H

### **harsh environment**

Resistance to hydrocarbons, industrial oils, detergents and solder chips. Relative humidity up to 100%, saline atmosphere, significant temperature variations, operating temperature between -10°C and +70°C, or in mobile installations. For hardened (H) devices, the relative humidity is up to 95% and the operating temperature is between -25°C and +70°C.

### **HART**

(*highway addressable remote transducer*) A bi-directional communication protocol for sending and receiving digital information across analog wires between a control or monitoring system and smart devices.

HART is the global standard for providing data access between host systems and intelligent field instruments. A host can be any software application from a technician's hand-held device or laptop to a plant's process control, asset management, or other system using any control platform.

### **high-capacity daisy chain loop**

Often referred to as HCDL, a high-capacity daisy chain loop uses dual-ring switches (DRSs) to connect device sub-rings (containing RIO drops or distributed equipment) and/or DIO clouds to the Ethernet RIO network.

### **HMI**

(*human machine interface*) System that allows interaction between a human and a machine.

### **Hot Standby**

A Hot Standby system uses a primary PAC (PLC) and a standby PAC. The two PAC racks have identical hardware and software configurations. The standby PAC monitors the current system status of the primary PAC. If the primary PAC becomes inoperable, high-availability control is maintained when the standby PAC takes control of the system.

### **HTTP**

(*hypertext transfer protocol*) A networking protocol for distributed and collaborative information systems. HTTP is the basis of data communication for the web.

## I

### **I/O scanner**

An Ethernet service that continuously polls I/O modules to collect data, status, event, and diagnostics information. This process monitors inputs and controls outputs. This service supports both RIO and DIO logic scanning.

### **IEC 61131-3**

International standard: programmable logic controllers

Part 3: programming languages

**IGMP**

*(internet group management protocol)* This internet standard for multicasting allows a host to subscribe to a particular multicast group.

**IL**

*(instruction list)* An IEC 61131-3 programming language that contains a series of basic instructions. It is very close to assembly language used to program processors. Each instruction is made up of an instruction code and an operand.

**implicit messaging**

UDP/IP-based class 1 connected messaging for EtherNet/IP. Implicit messaging maintains an open connection for the scheduled transfer of control data between a producer and consumer. Because an open connection is maintained, each message contains primarily data, without the overhead of object information, plus a connection identifier.

**INT**

*(INTEger)* (encoded in 16 bits) The upper/lower limits are as follows:  $-(2 \text{ to the power of } 15)$  to  $(2 \text{ to the power of } 15) - 1$ .

Example: -32768, 32767, 2#1111110001001001, 16#9FA4.

**inter-controller network**

An Ethernet-based network that is part of the control network, and provides data exchange between controllers and engineering tools (programming, asset management system (AMS)).

**IODDT**

*(input/output derived data type)* A structured data type representing a module, or a channel of a CPU. Each application expert module possesses its own IODDTs.

**IP address**

The 32-bit identifier, consisting of both a network address and a host address assigned to a device connected to a TCP/IP network.

**IPsec**

*(internet protocol security)* An open set of protocol standards that make IP communication sessions private and secure for traffic between modules using IPsec, developed by the internet engineering task force (IETF). The IPsec authentication and encryption algorithms require user-defined cryptographic keys that process each communications packet in an IPsec session.

**isolated DIO network**

An Ethernet-based network containing distributed equipment that does not participate in an RIO network.

**L****LD**

*(ladder diagram)* An IEC 61131-3 programming language that represents instructions to be executed as graphical diagrams very similar to electrical diagrams (contacts, coils, etc.).

**literal value of an integer**

A literal value of an integer is used to enter integer values in the decimal system. Values may be preceded by the "+" and "-" signs. Underscore signs (\_) separating numbers are not significant.

Example:

-12, 0, 123\_456, +986

**local rack**

An M580 rack containing the CPU and a power supply. A local rack consists of one or two racks: the main rack and the extended rack, which belongs to the same family as the main rack. The extended rack is optional.

**local slave**

The functionality offered by Schneider Electric EtherNet/IP communication modules that allows a scanner to take the role of an adapter. The local slave enables the module to publish data via implicit messaging connections. Local slave is typically used in peer-to-peer exchanges between PACs.

## M

**M580 Ethernet I/O device**

An Ethernet device that provides automatic network recovery and deterministic RIO performance. The time it takes to resolve an RIO logic scan can be calculated, and the system can recover quickly from a communication disruption. M580 Ethernet I/O devices include:

- local rack (including a CPU with Ethernet I/O scanner service)
- RIO drop (including an X80 adapter module)
- DRS switch with a predefined configuraton

**main ring**

The main ring of an Ethernet RIO network. The ring contains RIO modules and a local rack (containing a CPU with Ethernet I/O scanner service) and a power supply module.

**MAST**

A master (MAST) task is a deterministic processor task that is run through its programming software. The MAST task schedules the RIO module logic to be solved in every I/O scan. The MAST task has two sections:

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

**MB/TCP**

(*Modbus over TCP protocol*) This is a Modbus variant used for communications over TCP/IP networks.

**MIB**

(*management information base*) A virtual database used for managing the objects in a communications network. See SNMP.



**Modbus**

Modbus is an application layer messaging protocol. Modbus provides client and server communications between devices connected on different types of buses or networks. Modbus offers many services specified by function codes.

**multicast**

A special form of broadcast where copies of the packet are delivered to only a specified subset of network destinations. Implicit messaging typically uses multicast format for communications in an EtherNet/IP network.

## N

**network**

There are two meanings:

- In a ladder diagram:  
A network is a set of interconnected graphic elements. The scope of a network is local, concerning the organizational unit (section) of the program containing the network.
- With expert communication modules:  
A network is a set of stations that intercommunicate. The term *network* is also used to define a group interconnected graphic elements. This group then makes up part of a program that may comprise a group of networks.

**network time service**

Use this service to synchronize computer clocks over the Internet to record events (sequence events), synchronize events (trigger simultaneous events), or synchronize alarms and I/O (time stamp alarms).

**NIM**

(*network interface module*) A NIM resides in the first position on an STB island (leftmost on the physical setup). The NIM provides the interface between the I/O modules and the fieldbus master. It is the only module on the island that is fieldbus-dependent — a different NIM is available for each fieldbus.

**NTP**

(*network time protocol*) Protocol for synchronizing computer system clocks. The protocol uses a jitter buffer to resist the effects of variable latency.

## O

**O->T**

(*originator to target*) See originator and target.

**ODVA**

(*Open DeviceNet Vendors Association*) The ODVA supports network technologies that are based on CIP.

## OFS

(*OPC Factory Server*) OFS enables real-time SCADA communications with the Control Expert family of PLCs. OFS utilizes the standard OPC data access protocol.

## OPC DA

(*OLE for Process Control Data Access*) The Data Access Specification is the most commonly implemented of the OPC standards that provide specifications for real-time data communications between clients and servers.

## operation network

An Ethernet-based network containing operator tools (SCADA, client PC, printers, batch tools, EMS, etc.). Controllers are connected directly or through routing of the inter-controller network. This network is part of the control network.

## originator

In EtherNet/IP, a device is considered the originator when it initiates a CIP connection for implicit or explicit messaging communications or when it initiates a message request for un-connected explicit messaging.

# P

## PAC

*programmable automation controller*. The PAC is the brain of an industrial manufacturing process. It automates a process as opposed to relay control systems. PACs are computers suited to survive the harsh conditions of an industrial environment.

## port 502

Port 502 of the TCP/IP stack is the well-known port that is reserved for Modbus TCP communications.

## port mirroring

In this mode, data traffic that is related to the source port on a network switch is copied to another destination port. This allows a connected management tool to monitor and analyze the traffic.

## PTP

(*precision time protocol*) Use this protocol to synchronize clocks throughout a computer network. On a local area network, PTP achieves clock accuracy in the sub-microsecond range, making it suitable for measurement and control systems.

# Q

## QoS

(*quality of service*) The practice of assigning different priorities to traffic types for the purpose of regulating data flow on the network. In an industrial network, QoS is used to provide a predictable level of network performance.

## R

### **rack optimized connection**

Data from multiple I/O modules are consolidated in a single data packet to be presented to the scanner in an implicit message in an EtherNet/IP network.

### **ready device**

Ethernet ready device that provides additional services to the EtherNet/IP or Modbus module, such as: single parameter entry, bus editor declaration, system transfer, deterministic scanning capacity, alert message for modifications, and shared user rights between Control Expert and the device DTM.

### **RIO drop**

One of the three types of RIO modules in an Ethernet RIO network. An RIO drop is an M580 rack of I/O modules that are connected to an Ethernet RIO network and managed by an Ethernet RIO adapter module. A drop can be a single rack or a main rack with an extended rack.

### **RIO network**

An Ethernet-based network that contains 3 types of RIO devices: a local rack, an RIO drop, and a ConneXium extended dual-ring switch (DRS). Distributed equipment may also participate in an RIO network via connection to DRSs or BMENOS0300 network option switch modules.

### **RPI**

*(requested packet interval)* The time period between cyclic data transmissions requested by the scanner. EtherNet/IP devices publish data at the rate specified by the RPI assigned to them by the scanner, and they receive message requests from the scanner at each RPI.

### **RSTP**

*(rapid spanning tree protocol)* Allows a network design to include spare (redundant) links to provide automatic backup paths if an active link stops working, without the need for loops or manual enabling/disabling of backup links.

## S

### **S908 RIO**

A Quantum RIO system using coaxial cabling and terminators.

### **SCADA**

*(supervisory control and data acquisition)* SCADA systems are computer systems that control and monitor industrial, infrastructure, or facility-based processes (examples: transmitting electricity, transporting gas and oil in pipelines, and distributing water).

### **scanner**

A scanner acts as the originator of I/O connection requests for implicit messaging in EtherNet/IP, and message requests for Modbus TCP.

**scanner class device**

A scanner class device is defined by the ODVA as an EtherNet/IP node capable of originating exchanges of I/O with other nodes in the network.

**service port**

A dedicated Ethernet port on the M580 RIO modules. The port may support these major functions (depending on the module type):

- port mirroring: for diagnostic use
- access: for connecting HMI/Control Expert/ConneXview to the CPU
- extended: to extend the device network to another subnet
- disabled: disables the port, no traffic is forwarded in this mode

**SFC**

(*sequential function chart*) An IEC 61131-3 programming language that is used to graphically represent in a structured manner the operation of a sequential CPU. This graphical description of the CPU's sequential behavior and of the various resulting situations is created using simple graphic symbols.

**SFP**

(*small form-factor pluggable*). The SFP transceiver acts as an interface between a module and fiber optic cables.

**simple daisy chain loop**

Often referred to as SDCL, a simple daisy chain loop contains RIO modules only (no distributed equipment). This topology consists of a local rack (containing a CPU with Ethernet I/O scanner service), and one or more RIO drops (each drop containing an RIO adapter module).

**SMTP**

(*simple mail transfer protocol*) An email notification service that allows controller-based projects to report alarms or events. The controller monitors the system and can automatically create an email message alert with data, alarms, and/or events. Mail recipients can be either local or remote.

**SNMP**

(*simple network management protocol*) Protocol used in network management systems to monitor network-attached devices. The protocol is part of the internet protocol suite (IP) as defined by the internet engineering task force (IETF), which consists of network management guidelines, including an application layer protocol, a database schema, and a set of data objects.

**SNTP**

(*simple network time protocol*) See NTP.

**SOE**

(*sequence of events*) The process of determining the order of events in an industrial system and correlating those events to a real-time clock.

**ST**

(*structured text*) An IEC 61131-3 programming language that presents structured literal language and is a developed language similar to computer programming languages. It can be used to organize a series of instructions.

**sub-ring**

An Ethernet-based network with a loop attached to the main ring, via a dual-ring switch (DRS) or BMENOS0300 network option switch module on the main ring. This network contains RIO or distributed equipment.

**subnet mask**

The 32-bit value used to hide (or mask) the network portion of the IP address and thereby reveal the host address of a device on a network using the IP protocol.

**switch**

A multi-port device used to segment the network and limit the likelihood of collisions. Packets are filtered or forwarded based upon their source and destination addresses. Switches are capable of full-duplex operation and provide full network bandwidth to each port. A switch can have different input/output speeds (for example, 10, 100 or 1000Mbps). Switches are considered OSI layer 2 (data link layer) devices.

## T

**T->O**

(*target to originator*) See target and originator.

**target**

In EtherNet/IP, a device is considered the target when it is the recipient of a connection request for implicit or explicit messaging communications, or when it is the recipient of a message request for un-connected explicit messaging.

**TCP**

(*transmission control protocol*) A key protocol of the internet protocol suite that supports connection-oriented communications, by establishing the connection necessary to transmit an ordered sequence of data over the same communication path.

**TCP/IP**

Also known as *internet protocol suite*, TCP/IP is a collection of protocols used to conduct transactions on a network. The suite takes its name from two commonly used protocols: transmission control protocol and internet protocol. TCP/IP is a connection-oriented protocol that is used by Modbus TCP and EtherNet/IP for explicit messaging.

**TFTP**

(*trivial file transfer protocol*) A simplified version of *file transfer protocol* (FTP), TFTP uses a client-server architecture to make connections between two devices. From a TFTP client, individual files can be uploaded to or downloaded from the server, using the user datagram protocol (UDP) for transporting data.

**TIME\_OF\_DAY**

See TOD.

## TOD

(*time of day*) The TOD type, encoded in BCD in a 32-bit format, contains this information:

- the hour encoded in an 8-bit field
- the minutes encoded in an 8-bit field
- the seconds encoded in an 8-bit field

**NOTE:** The eight least significant bits are not used.

The TOD type is entered in this format: xxxxxxxx: **TOD#**<Hour>:<Minutes>:<Seconds>

This table shows the upper/lower limits of each field:

Field	Limits	Comment
Hour	[00,23]	The leading 0 is displayed; it can be omitted during data entry.
Minute	[00,59]	The leading 0 is displayed; it can be omitted during data entry.
Second	[00,59]	The leading 0 is displayed; it can be omitted during data entry.

Example: TOD#23:59:45.

## TR

(*transparent ready*) Web-enabled power distribution equipment, including medium- and low-voltage switch gear, switchboards, panel boards, motor control centers, and unit substations. Transparent Ready equipment allows you to access metering and equipment status from any PC on the network, using a standard web browser.

## trap

A trap is an event directed by an SNMP agent that indicates one of these events:

- A change has occurred in the status of an agent.
- An unauthorized SNMP manager device has attempted to get data from (or change data on) an SNMP agent.

# U

## UDP

(*user datagram protocol*) A transport layer protocol that supports connectionless communications. Applications running on networked nodes can use UDP to send datagrams to one another. Unlike TCP, UDP does not include preliminary communication to establish data paths or provide data ordering and checking. However, by avoiding the overhead required to provide these features, UDP is faster than TCP. UDP may be the preferred protocol for time-sensitive applications, where dropped datagrams are preferable to delayed datagrams. UDP is the primary transport for implicit messaging in EtherNet/IP.

## UMAS

(*Unified Messaging Application Services*) UMAS is a proprietary system protocol that manages communications between Control Expert and a controller.

**UTC**

(*coordinated universal time*) Primary time standard used to regulate clocks and time worldwide (close to former GMT time standard).

**V****variable**

Memory entity of type `BOOL`, `WORD`, `DWORD`, etc., whose contents can be modified by the program currently running.

**VLAN**

(*virtual local area network*) A local area network (LAN) that extends beyond a single LAN to a group of LAN segments. A VLAN is a logical entity that is created and configured uniquely using applicable software.







## P

PLCSTAT, *65*

## U

Unity M580 Application Converter

Analyze screen, *18*

Convert screen, *19*

Prepare screen, *17*

Select screen, *16*