Device and Process Library

a SoCollaborative library
Diagnostic Components

09/2009

EIO0000000254.01

www.schneider-electric.com

Schneider Electric

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

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

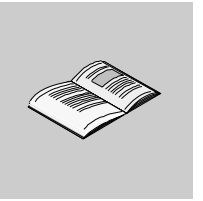
All pertinent state, regional, and local safety regulations must be observed when installing and using this product. For reasons of safety and to help ensure compliance with documented system data, only the manufacturer should perform repairs to components.

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

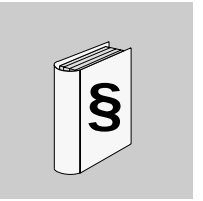
Failure to use Schneider Electric software or approved software with our hardware products may result in injury, harm, or improper operating results.

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

Table of Contents



Safety Information



Important Information

NOTICE

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

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

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

⚠ DANGER

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

⚠ WARNING

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

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

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

PLEASE NOTE

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

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

BEFORE YOU BEGIN

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

⚠ WARNING
UNGUARDED MACHINERY CAN CAUSE SERIOUS INJURY • Do not use this software and related automation equipment on equipment which does not have point-of-operation protection. • Do not reach into machinery during operation. **Failure to follow these instructions can result in death, serious injury, or equipment damage.**

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

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

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

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

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

START-UP AND TEST

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

⚠ CAUTION

EQUIPMENT OPERATION HAZARD

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

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

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

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

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

Before energizing equipment:

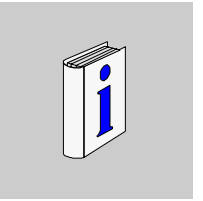
- Remove tools, meters, and debris from equipment.
- Close the equipment enclosure door.
- Remove ground from incoming power lines.
- Perform all start-up tests recommended by the manufacturer.

OPERATION AND ADJUSTMENTS

The following precautions are from the NEMA Standards Publication ICS 7.1-1995 (English version prevails):

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

About the Book



At a Glance

Document Scope

This document provides information on operating a project developed with the Unity Application Generator (UAG) and the DPL Diagnostic Components.

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

The DPL Diagnostic Components is intended for end users and system integrators.

To use the DPL Diagnostic Components, you must have a strong working knowledge of:

- Unity Application Generator V3.1 SP2 or V3.2
- Unity Pro V4.1 S, M, L, XL
- Vijeo Citect V7.0 or V7.1

Validity Note

This manual is valid for these product versions:

- Unity Application Generator V3.1 SP2 or V3.2
- Unity Pro V4.1 S, M, L, XL
- Vijeo Citect V7.0 or V7.1

Related Documents

Title of Documentation	Reference Number
Unity Application Generator 3.0 Basic User Manual	33002830
Unity Application Generator 3.0 Extended User Manual	33003669
SCoD Editor 3.0 User Manual	33002608

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

Product Related Information

⚠ WARNING

UNINTENDED EQUIPMENT OPERATION

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

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

⚠ WARNING

UNINTENDED EQUIPMENT OPERATION

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

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

User Comments

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

Introduction

1

Introduction

Overview

The DPL control system provides resources or components that are designed to perform hardware diagnostics on the various existing Schneider Electric automation platforms.

The control resources (included in the DPL platform for Unity (DPU)) and monitoring resources (included in the DPL platform for Vijeo Citect (DPC)) provide the commonly required functions, facilitating the development of control systems by incorporating hardware diagnostic functions and reducing the risk that errors will be reported.

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

To automate and simplify the implementation of control systems, these resources can be used in conjunction with tools to generate large amounts of code and for synchronizing the control and monitoring subsystems.

This document describes the DPL platform resources for hardware Diagnostics.

The document includes a detailed description of the DPU platform resources for Unity (DPU) that refer to diagnostics for the various SoCollaborative platforms.

For more detailed information, please consult the User Manual for DPL platform resources for Vijeo Citect (DPLVC).

General Overview of Resources for Diagnostics

2

Overview

This chapter describes the resources that were designed for hardware diagnostics.

What's in this Chapter?

This chapter contains the following topics:

DPU Functional Blocks for Diagnostics

Functional Blocks

This list shows the function blocks described in this document that are designed for hardware diagnostics.

CPUDIAG: This is a component for the general controller diagnostics. It is valid in any Unity platform (Premium, Quantum, and M340), communications diagnostics, battery status, display of MAST task scan times, reason for and time of last stop, etc.

This component is mandatory to use the associated Vijeo Citect Genies from this library.

FASTINFO: This is a supplementary component for CPUDIAG, intended for FAST task diagnostics.

It must be executed in a section of the FAST task *(see page 51)*.

AUX0INFO: This is a supplementary component for CPUDIAG, intended for AUX0 task diagnostics.

It must be executed in a section of the AUX0 task *(see page 55)*.

AUX1INFO: This is a supplementary component for CPUDIAG, intended for AUX1 task diagnostics.

It must be executed in a section of the AUX1 task *(see page 59)*.

AUX2INFO: This is a supplementary component for CPUDIAG, intended for AUX2 task diagnostics.

It must be executed in a section of the AUX2 task *(see page 63)*.

AUX3INFO: This is a supplementary component for CPUDIAG, intended for AUX3 task diagnostics.

It must be executed in a section of the AUX3 task *(see page 67)*.

DPU Component Library for Diagnostics

General

The function blocks mentioned previously (along with other standard function blocks from Unity libraries) are combined to generate DPU components that implement modules that enable the monitoring of hardware statuses.

Grouping Functional Blocks

DPU components are grouped under the Diagnostic Library for UAG and are made available to the user in order to facilitate the creation of automation projects with the Unity Application Generator tool.

In general, DPU components from the Diagnostic Library for UAG include these resources for the automatic generation of Unity projects:

- Variables
- Control logic

This list shows the component categories included in the Diagnostic Library for UAG:
General Status: CPUDIAG
Other Tasks: Remaining components

NOTE: The current version of UAG generates code only for the MAST task of the controller. The sections that were created for each component, with the exception of CPUDIAG, must be moved to the FAST, AUX0, AUX1, AUX2, or AUX3 tasks, as applicable. Otherwise the component information is incorrect.

DPC for Diagnostics

Introduction

This section describes the DPC for diagnostics (where xy corresponds to the new version and revision, x.y.).

Dynamic Symbols

This list describes Genies:
DPC_CITEC / DPC CPUDIAG_M340_xy: M340 CPU Diagnostics.
DPC_CITEC / DPC CPUDIAG_PREMIUM_xy: Premium CPU Diagnostics.
DPC_CITEC / DPC CPUDIAG_QUANTUM_xy: Quantum CPU Diagnostics.
DPC_CITEC / DPC CPUDIAG_M340_xy: Combines M340 CPU Diagnostics with the Vijeo Citect I/O Devices that communicate with the controller.
DPC_CITEC / DPC CPUDIAG_M340_xy: Combines Premium CPU Diagnostics with the Vijeo Citect I/O Devices that communicate with the controller.
DPC_CITEC / DPC CPUDIAG_M340_xy: Combines Quantum CPU Diagnostics with the Vijeo Citect I/O Devices that communicate with the controller.

Faceplates

Popup windows with SuperGenies syntax
CPUDIAG_M340_xy: M340 CPU Diagnostics.
CPUDIAG_QP_xy: Premium and Quantum CPU Diagnostics.

GENSTS - GENERAL STATUS

3

Overview

This chapter describes the GENSTS DFB.

⚠ WARNING

MISAPPLICATION OF MODULES

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

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

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

What's in this Chapter?

This chapter contains the following topics:

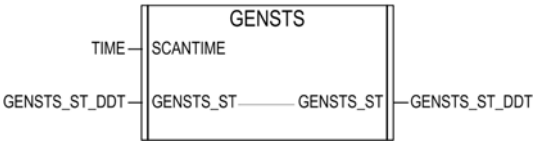
GENSTS - GENERAL STATUS

Overview

This DFB is part of the CPUDIAG component. It is also the essential DFB that must be included for Vijeo Citect to recognize the object.

FBD Representation

The function block used in the program has the following aspect (at the FBD level) after PLC generation in Unity Pro.



Inputs/Outputs

GENSTS_ST: This input/output provides the block with the data (GENSTS_ST_DDT) required for monitoring.

GENSTS_ST_DDT Structure Description

This table shows the GENSTS_ST_DDT structure:

Name	Type	Description
SYSBITW	WORD	system word
COUNTER	INT	This integer variable has a value from 0 to 100. It increases every second. When it reaches 100, the counter is reset to 0 and restarts. It is used to monitor the execution of the MAST task.

SYSBITW Description

This table shows SYSBITW (WORD) description:

Bit	Description
0	• This bit indicates a cold CPU boot. • The Unity %S0 system bit is mapped directly. • This bit has no effect on monitoring.
1	• This bit indicates a warm CPU boot. • The Unity %S0 system bit is mapped directly. • This bit has no effect on monitoring.
2	• This bit shows the I/O status of the controller. • This bit indicates the equivalent of the IO LED on the physical panel of the CPU. • This bit indicates an error when its value is 1. • This is the NOT mapping for the Unity %S10 bit. • This bit is displayed on the Vijeo Citect Genie.
3	• This bit indicates the watchdog overflow. • This bit is set to 1 when the watchdog is skipped. • This bit indicates the mapping for the Unity %S11 system bit. • This bit has no effect on monitoring.

SCoD Representation and Parameter Description

Overview

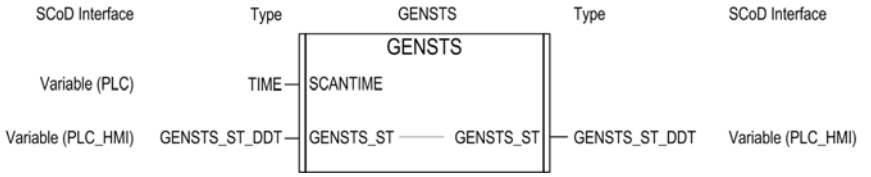
Use of the DFB pins in the SCoD editor is explained in the input/output diagram below.

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

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

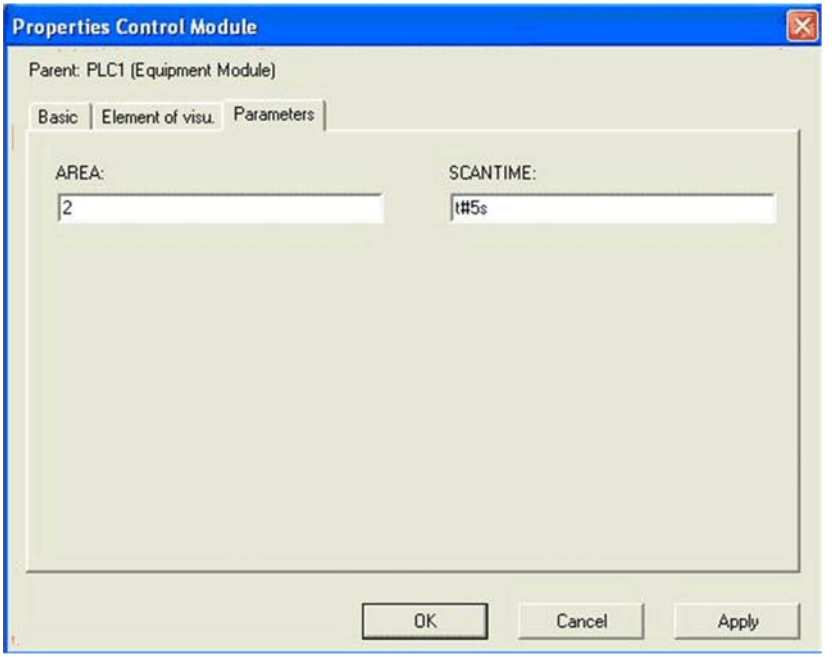
SCoD Representation

This figure represents the GENSTS SCoD:



Parameter Tab

This figure shows the Parameter Tab of the Properties Control Module:

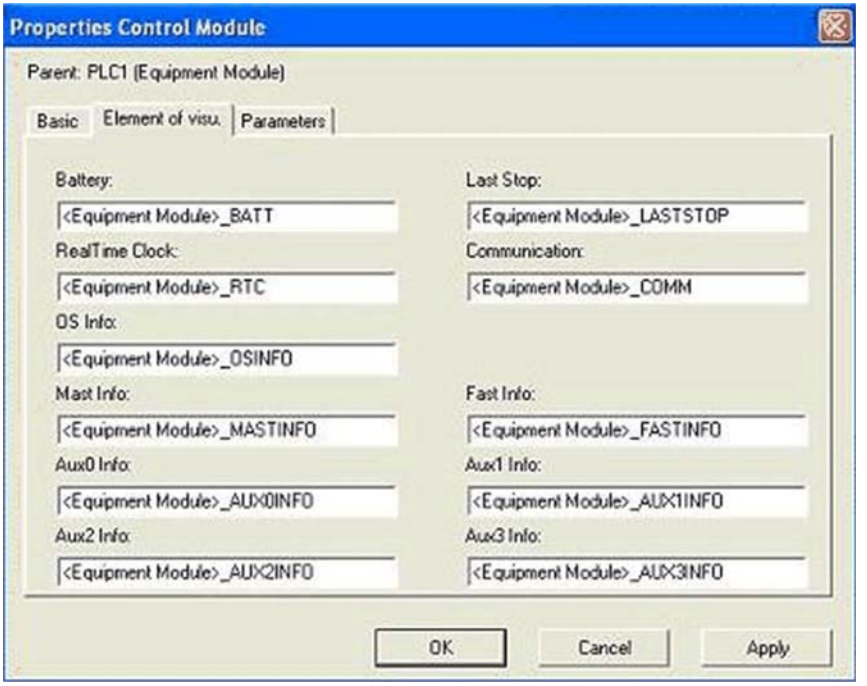


The following table describes the properties:

Property	Type	Description
AREA	BYTE	Area to which the object that you wish to instantiate belongs. Enables to control access to the functions provided by this object. More detailed in the “ ACCESS CONTROL” section of the Vijeo CItect documentation.
SCANTIME	TIME	Time (msec) that the alarm signals are kept active to be configured.

Visu. Tab

This figure shows the Visu. tab of the Properties Control module:



The following table describes the properties:

Property	Type	Description
Battery	SCoD BATT CM reference	Link to the CM Variable of the BATT SCoD
Last Stop	SCoD Last Stop reference	Link to the CM Variable of the LAST STOP SCoD
RealTime Clock	SCoD RealTime Clock reference	Link to the CM Variable of the RTC SCoD
Communication	SCoD Communication reference	Link to the CM Variable of the COMM SCoD
OS Info	SCoD OS Info reference	Link to the CM Variable of the OSINFO SCoD
MAST Info	SCoD MAST Info reference	Link to the CM Variable of the MASTINFO SCoD
FAST Info	SCoD FAST Info reference	Link to the CM Variable of the FASTINFO SCoD
Aux0 Info	SCoD Aux0 Info reference	Link to the CM Variable of the AUX0INFO SCoD
Aux1 Info	SCoD Aux1 Info reference	Link to the CM Variable of the AUX1INFO SCoD
Aux2 Info	SCoD Aux2 Info reference	Link to the CM Variable of the AUX2INFO SCoD
Aux3 Info	SCoD Aux3 Info reference	Link to the CM Variable of the AUX3INFO SCoD

OSINFO - OPERATING SYSTEM INFORMATION

4

Overview

This chapter describes the OSINFO DFB.

⚠ WARNING

MISAPPLICATION OF MODULES

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

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

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

What's in this Chapter?

This chapter contains the following topics:

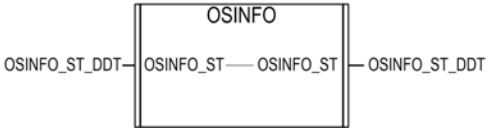
OSINFO - OPERATING SYSTEM INFORMATION

Overview

This DFB provides information regarding system operations.

FBD Representation

The function block used in the program has the following aspect (at the FBD level) after PLC generation in Unity Pro.



Inputs/Outputs

OSINFO_ST: This input/output provides the block with the data (OSINFO_ST_DDT) required for monitoring.

OSINFO_ST_DDT Structure Description

This table shows the OSINFO_ST_DDT structure:

Name	Type	Description
CPUVER	INT	This data holds the version of the controller. This data indicates the direct mapping of the %SW14 system word.
CPUPATCH	INT	This data holds the version of the patch for the processor. This data indicates the direct mapping of the %SW15 system word.
CPUFIRM	INT	This data holds the firmware revision number of the CPU. This data indicates the direct mapping of the %SW16 system word.

SCoD Representation

Overview

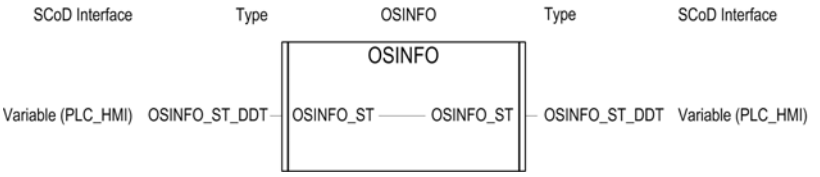
Use of the DFB pins in the SCoD editor is explained in the input/output diagram below.

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

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

SCoD Representation

This figure represents the OSINFO SCoD:



MASTINFO - MAST TASK INFORMATION

5

Overview

This chapter describes the MASTINFO DFB.

⚠ WARNING

MISAPPLICATION OF MODULES

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

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

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

What's in this Chapter?

This chapter contains the following topics:

MASTINFO - MAST TASK INFORMATION

Overview

This DFB is included in the UAG DPLDiagnoctic library. It is optional, and can therefore be deleted from the generated code.

FBD Representation

The function block used in the program has the following aspect (at the FBD level) after PLC generation in Unity Pro.

Inputs/Outputs

MASTINFO_ST: This input/output provides the block with the data (MASTINFO_ST_DDT) required for monitoring.

MASTINFO_ST_DDT Structure Description

This table shows the MASTINFO_ST_DDT structure:

Name	Type	Description
CURRTIME	INT	This data provides the duration of the last scan cycle of the controller. This data indicates the direct mapping of the %SW30 system word.
MAXTIME	INT	This data provides the duration of the longest cycle of the task since the last cold start. This data indicates the direct mapping of the %SW15 system word.
MINTIME	INT	This data provides the duration of the shortest cycle of the task since the last cold start. This data indicates the direct mapping of the %SW16 system word.
WDVALUE	INT	This data provides the value with which the MAST task Watchdog is configured.

SCoD Representation

Overview

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

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

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

SCoD Representation

This figure represents the MASTINFO SCoD:

LASTSTOP - LAST STOP INFORMATION

6

Overview

This chapter describes the LASTSTOP DFB.

⚠ WARNING

MISAPPLICATION OF MODULES

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

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

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

What's in this Chapter?

This chapter contains the following topics:

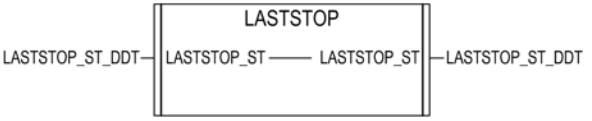
LASTSTOP - LAST STOP INFORMATION

Overview

This DFB is included in the CPUDIAG component. It is optional, and can therefore be deleted from the generated code.

FBD Representation

The function block used in the program has the following aspect (at the FBD level) after PLC generation in Unity Pro.



Inputs/Outputs

LASTSTOP_ST: This input/output provides the block with the data (LASTSTOP_ST_DDT) required for monitoring.

LASTSTOP_ST_DDT Structure Description

This table shows the LASTSTOP_ST_DDT structure:

Name	Type	Description
STOPSEC	INT	This data is represented in BCD format. This data holds the seconds that correspond to the time at which the last stop occurred.
STOPHM	INT	This data is represented in BCD format. This data shows the hours and minutes corresponding to the time of the last stop. (HHMM).
STOPMD	INT	This data is represented in BCD format. This data shows the month and date of the last stop. (MMDD).
STOPYEAR	INT	This data is represented in BCD format. This data shows the year of the last stop. (YYYY).
STOPDAY	INT	This data is represented in BCD format. This data shows the day of the week (high byte) of the last stop, as well as its cause (low byte).

High Byte Description

The high byte indicates the day of the week:

High Byte	Meaning
1	Monday
2	Tuesday
3	Wednesday
4	Thursday
5	Friday
6	Saturday
7	Sunday

Low Byte Description

The low byte describes the cause of the last stop:

Low Byte	Meaning
1	A terminal or dedicated input caused a transition from RUN to STOP.
2	A program interruption (a controller or SFC task overflow) caused the stop.
4	A power cutoff/outage or the manipulation of the memory card caused a stop.
5	A hardware interruption caused the stop.
6	A HALT command caused the stop.

SCoD Representation

Overview

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

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

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

SCoD Representation

This figure represents the LASTSTOP SCoD:

COMM - COMMUNICATIONS DIAGNOSTICS

7

Overview

This chapter describes the COMM DFB.

⚠ WARNING

MISAPPLICATION OF MODULES

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

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

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

What's in this Chapter?

This chapter contains the following topics:

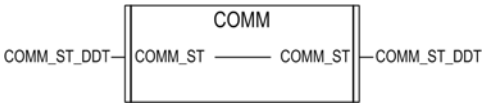
COMM - COMMUNICATIONS DIAGNOSTICS

Overview

This DFB is included in the CPUDIAG component. It is optional, and can therefore be deleted from the generated code.

FBD Representation

The function block used in the program has the following aspect (at the FBD level) after PLC generation in Unity Pro.



Inputs/Outputs

COMM_ST: This input/output provides the block with the data required for monitoring.

COMM_ST_DDT Structure Description

This table shows the COMM_ST_DDT structure:

Name	Type	Description
CURREQ	INT	This data provides the number of requests processed per SCAN cycle. This data indicates the direct mapping of the %SW87 system word.
MAXREQ	INT	This data provides the maximum number of requests that the CPU can manage per SCAN cycle. This data indicates the direct mapping of the %SW90 system word.
USEDCOMM	INT	This data provides the percentage corresponding to the current communications load on the controller that functions as a server, i.e., the load on the controller due to requests from other devices (PLCs, SCADAs, etc.).

SCoD Representation

Overview

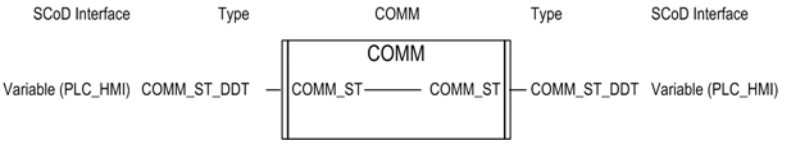
Use of the DFB pins in the SCoD editor is explained in the input/output diagram below.

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

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

SCoD Representation

This figure represents the COMM SCoD:



RTC - REAL TIME CLOCK

8

Overview

This chapter describes the RTC DFB.

⚠ WARNING

MISAPPLICATION OF MODULES

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

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

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

What's in this Chapter?

This chapter contains the following topics:

RTC - REAL TIME CLOCK

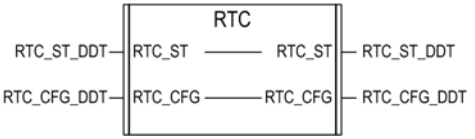
Overview

This DFB is part of the CPUDIAG component. It is optional, and can therefore be deleted from the generated code.

It enables the modification and display of the controller time.

FBD Representation

The function block used in the program has the following aspect (at the FBD level) after PLC generation in Unity Pro.



Inputs/Outputs

RTC_ST: This input/output provides the block with the read-only data (RTC_ST_DDT) required for monitoring and data exchange.

RTC_CFG: This input/output provides the block with the read/write data (RTC_CFG_DDT) required for monitoring and data exchange.

RTC_ST_DDT Structure Description

This table shows the RTC_ST_DDT structure:

Name	Type	Description	System Word
ACTDATOFWEEK	INT	This data provides the day of the week. For example: 1 = Monday 2 = Tuesday	%SW49
ACTSEC	INT	This data provides the seconds that correspond to the current time.	%SW50
ACTHOURMIN	INT	This data (BCD format) represents the hours and minutes (HHMM) that correspond to the current time. It is the direct mapping of the %SW51 system word.	%SW51
ACTMONTHDAY	INT	This data (BCD format) represents the current month and day (MMDD).	%SW52
ACTYEAR	INT	This data (BCD format) represents the current year (YYYY).	%SW53

NOTE: System words must be mapped directly.

RTC_CFG_DDT Structure Description

This table shows the RTC_CFG_DDT structure:

Name	Type	Description	System Word
RTCWRITEW	WORD	The word bit 0 updates the controller time. It is a write variable from SCADA that is activated from the Vijeo Citect Super Genie.	%SW50
NEWSEC	INT	This data provides the seconds for the time that is being adjusted from the monitoring system.	
NEWMINUTE	INT	This data provides the minutes for the time that is being adjusted from the monitoring system.	
NEWHOUR	INT	This data provides the hours for the time that is being adjusted from the monitoring system.	
NEWDATE	INT	This data provides the day of the month that is being adjusted from the monitoring system.	
NEWMONTH	INT	This data provides the number of the month for the date that is being adjusted from the monitoring system.	
NEWYEAR	INT	This data provides the number of the year for the date that is being adjusted from the monitoring system.	

SCoD Representation

Overview

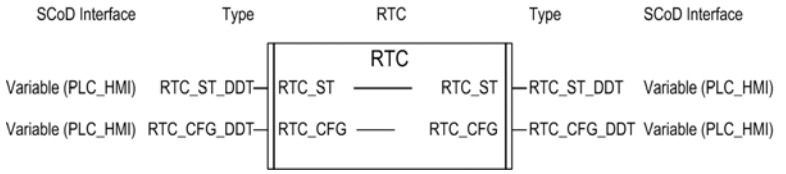
Use of the DFB pins in the SCoD editor is explained in the input/output diagram below.

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

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

SCoD Representation

This figure represents the RTC SCoD:



BATT - BATTERY DIAGNOSTICS

9

Overview

This chapter describes the BATT DFB.

⚠ WARNING

MISAPPLICATION OF MODULES

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

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

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

What's in this Chapter?

This chapter contains the following topics:

BATT - BATTERY DIAGNOSTICS

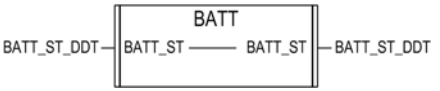
Overview

This DFB is part of the CPUDIAG component. It is optional, and can therefore be deleted from the generated code.

This function retrieves information about the battery status.

FBD Representation

The function block used in the program has the following aspect (at the FBD level) after PLC generation in Unity Pro.



Inputs/Outputs

BATT_ST: This input/output provides the block with the read-only data (BATT_ST_DDT) required for monitoring and data exchange.

BATT_ST_DDT Structure Description

This table shows the BATT_ST_DDT structure:

Name	Type	Description	System Word
SYSBITW	WORD	system word	%SW49

SYSBITW Description

This table describes the SYSBITW bits:

Bit	Description	System Word
0	This bit provides status of the data and program storage battery. **0** Working correctly. **1** Absence or out of service.	%S68
1	This bit provides status of the PCMCIA0 SLOT battery. **0** Battery normal. **1** Replace low battery.	%S67
2	This bit provides status of the PCMCIA1 SLOT battery. **0** Battery normal. **1** Replace low battery.	%S75

NOTE: This battery data is available for the Premium and Quantum automation platforms. It does not apply to the M340 platform, which does not have a battery.

SCoD Representation

Overview

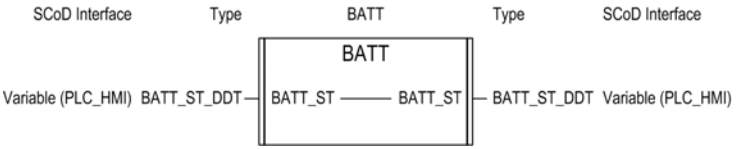
Use of the DFB pins in the SCoD editor is explained in the input/output diagram below.

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

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

SCoD Representation

This figure represents the BATT SCoD:



FASTINFO - FAST TASK DIAGNOSTICS

10

Overview

This chapter describes the FASTINFO DFB.

⚠ WARNING

MISAPPLICATION OF MODULES

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

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

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

What's in this Chapter?

This chapter contains the following topics:

FASTINFO - FAST TASK DIAGNOSTICS

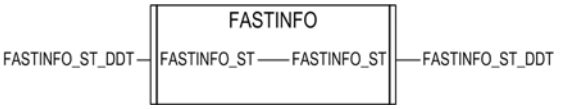
Overview

This DFB corresponds to the FASTINFO component. It must be executed in the FAST task of the controller.

The UAG generates code only for the MAST task. The generated component must be moved from the MAST task to the FAST task after the code has been generated.

FBD Representation

The function block used in the program has the following aspect (at the FBD level) after PLC generation in Unity Pro.



Inputs/Outputs

FASTINFO_ST_DDT: This input/output provides the block with the read-only data (FASTINFO_ST_DDT) required for monitoring and data exchange.

FASTINFO_ST_DDT Structure Description

This table shows the FASTINFO_ST_DDT structure:

Name	Type	Description	System Word
CURRTIME	INT	This data reports the duration of the last scan cycle of the controller.	%SW33
MAXTIME	INT	This data reports the duration of the longest cycle task since the last cold start.	%SW34
MINTIME	INT	This data reports the duration of the shortest task cycle since the last cold start.	%SW35
WDVALUE	INT	This data reports the value with which the Watchdog task is configured.	%SW11

NOTE: Times are in milliseconds.

SCoD Representation

Overview

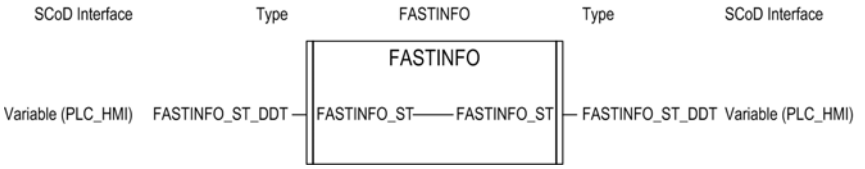
Use of the DFB pins in the SCoD editor is explained in the input/output diagram below.

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

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

SCoD Representation

This figure represents the GENSTS SCoD:



AUX0INFO - AUX0 TASK DIAGNOSTICS

11

Overview

This chapter describes the AUX0INFO DFB.

⚠ WARNING

MISAPPLICATION OF MODULES

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

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

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

What's in this Chapter?

This chapter contains the following topics:

AUX0INFO - AUX0 TASK DIAGNOSTICS

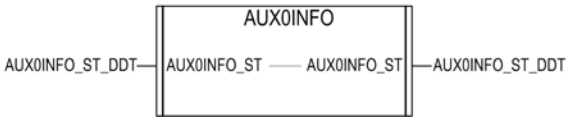
Overview

This DFB corresponds to the AUX0INFO component. It must be executed in the AUX0 task of the controller.

The UAG generates code only for the MAST task. The generated component must be moved from the MAST task to the AUX0 task after the code has been generated.

FBD Representation

The function block used in the program has the following aspect (at the FBD level) after PLC generation in Unity Pro.



Inputs/Outputs

AUX0INFO_ST_DDT: This input/output provides the block with the read-only data (AUX0INFO_ST_DDT) required for monitoring and data exchange.

AUX0INFO_ST_DDT Structure Description

This table shows the AUX0INFO_ST_DDT structure:

Name	Type	Description	System Word
CURRTIME	INT	This data reports the duration of the last scan cycle of the controller.	%SW36
MAXTIME	INT	This data reports the duration of the longest cycle task since the last cold start.	%SW37
MINTIME	INT	This data reports the duration of the shortest task cycle since the last cold start.	%SW38
WDVALUE	INT	This data reports the value with which the Watchdog task is configured.	%SW11

NOTE: Times are in milliseconds.

SCoD Representation

Overview

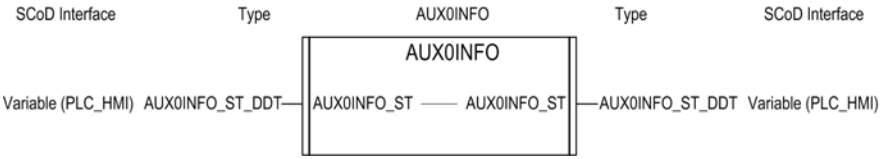
Use of the DFB pins in the SCoD editor is explained in the input/output diagram below.

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

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

SCoD Representation

This figure represents the GENSTS SCoD:



AUX1INFO - AUX1 TASK DIAGNOSTICS

12

Overview

This chapter describes the AUX1INFO DFB.

⚠ WARNING

MISAPPLICATION OF MODULES

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

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

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

What's in this Chapter?

This chapter contains the following topics:

AUX1INFO - AUX1 TASK DIAGNOSTICS

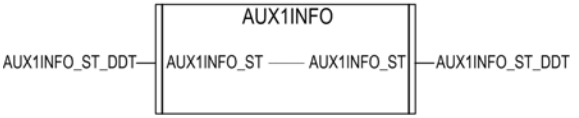
Overview

This DFB corresponds to the AUX1INFO component. It must be executed in the AUX1 task of the controller.

The UAG generates code only for the MAST task. The generated component must be moved from the MAST task to the AUX1 task after the code has been generated.

FBD Representation

The function block used in the program has the following aspect (at the FBD level) after PLC generation in Unity Pro.



Inputs/Outputs

AUX1INFO_ST_DDT: This input/output provides the block with the read-only data (AUX1INFO_ST_DDT) required for monitoring and data exchange.

AUX1INFO_ST_DDT Structure Description

This table shows the AUX1INFO_ST_DDT structure:

Name	Type	Description	System Word
CURRTIME	INT	This data reports the duration of the last scan cycle of the controller.	%SW39
MAXTIME	INT	This data reports the duration of the longest cycle task since the last cold start.	%SW40
MINTIME	INT	This data reports the duration of the shortest task cycle since the last cold start.	%SW41
WDVALUE	INT	This data reports the value with which the Watchdog task is configured.	%SW11

NOTE: Times are in milliseconds.

SCoD Representation

Overview

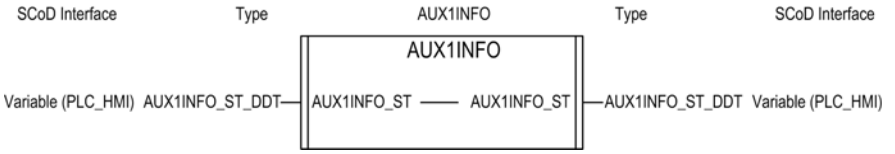
Use of the DFB pins in the SCoD editor is explained in the input/output diagram below.

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

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

SCoD Representation

This figure represents the AUX1INFO SCoD:



AUX2INFO - AUX2 TASK DIAGNOSTICS

13

Overview

This chapter describes the AUX2INFO DFB.

⚠ WARNING

MISAPPLICATION OF MODULES

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

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

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

What's in this Chapter?

This chapter contains the following topics:

AUX2INFO - AUX2 TASK DIAGNOSTICS

Overview

This DFB corresponds to the AUX2INFO component. It must be executed in the AUX2 task of the controller.

The UAG generates code only for the MAST task. The generated component must be moved from the MAST task to the AUX2 task after the code has been generated.

FBD Representation

The function block used in the program has the following aspect (at the FBD level) after PLC generation in Unity Pro.

Inputs/Outputs

AUX2INFO_ST_DDT: This input/output provides the block with the read-only data (AUX2INFO_ST_DDT) required for monitoring and data exchange.

AUX2INFO_ST_DDT Structure Description

This table shows the AUX2INFO_ST_DDT structure:

Name	Type	Description	System Word
CURRTIME	INT	This data reports the duration of the last scan cycle of the controller.	%SW42
MAXTIME	INT	This data reports the duration of the longest cycle task since the last cold start.	%SW43
MINTIME	INT	This data reports the duration of the shortest task cycle since the last cold start.	%SW44
WDVALUE	INT	This data reports the value with which the Watchdog task is configured.	%SW11

NOTE: Times are in milliseconds.

SCoD Representation

Overview

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

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

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

SCoD Representation

This figure represents the AUX2INFO SCoD:

AUX3INFO - AUX3 TASK DIAGNOSTICS

14

Overview

This chapter describes the AUX3INFO DFB.

⚠ WARNING

MISAPPLICATION OF MODULES

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

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

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

What's in this Chapter?

This chapter contains the following topics:

AUX3INFO - AUX3 TASK DIAGNOSTICS

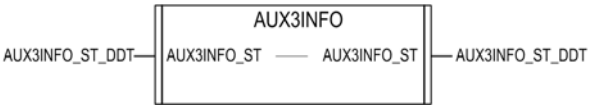
Overview

This DFB corresponds to the AUX3INFO component. It must be executed in the AUX3 task of the controller.

The UAG generates code only for the MAST task. The generated component must be moved from the MAST task to the AUX3 task after the code has been generated.

FBD Representation

The function block used in the program has the following aspect (at the FBD level) after PLC generation in Unity Pro.



Inputs/Outputs

AUX3INFO_ST_DDT: This input/output provides the block with the read-only data (AUX3INFO_ST_DDT) required for monitoring and data exchange.

AUX3INFO_ST_DDT Structure Description

This table shows the AUX3INFO_ST_DDT structure:

Name	Type	Description	System Word
CURRTIME	INT	This data reports the duration of the last scan cycle of the controller.	%SW45
MAXTIME	INT	This data reports the duration of the longest cycle task since the last cold start.	%SW46
MINTIME	INT	This data reports the duration of the shortest task cycle since the last cold start.	%SW47
WDVALUE	INT	This data reports the value with which the Watchdog task is configured.	%SW11

NOTE: Times are in milliseconds.

SCoD Representation

Overview

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

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

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

SCoD Representation

This figure represents the AUX3INFO SCoD: