SoMachine

Altivar / PLCopen / CANopen Example Guide

06/2017

EIO0000002551.00

www.schneider-electric.com

Schneider Electric

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.

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, 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 an example how to implement an Altivar variable speed drive controlled via fieldbus CANopen.

Validity Note

This document has been updated for the release of SoMachine V4.3.

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

Step	Action
1	Go to the Schneider Electric home page *www.schneider-electric.com*.
2	In the **Search** box type the reference of a product or the name of a product range. ● Do not include blank spaces in the reference or product range. ● To get information on grouping similar modules, use asterisks (*).
3	If you entered a reference, go to the **Product Datasheets** search results and click on the reference that interests you. If you entered the name of a product range, go to the **Product Ranges** search results and click on the product range that interests you.
4	If more than one reference appears in the **Products** 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 data sheet.
6	To save or print a data sheet as a .pdf file, click **Download XXX product datasheet**.

The characteristics that are presented in this manual 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 manual and online information, use the online information as your reference.

Product Related Information

⚠ WARNING

LOSS OF CONTROL

- The designer of any control scheme must consider the potential failure modes of control paths and, for certain critical control functions, provide a means to achieve a safe state during and after a path failure. Examples of critical control functions are emergency stop and overtravel stop, power outage and restart.
- Separate or redundant control paths must be provided for critical control functions.
- System control paths may include communication links. Consideration must be given to the implications of unanticipated transmission delays or failures of the link.
- Observe all accident prevention regulations and local safety guidelines.[1]
- Each implementation of this equipment must be individually and thoroughly tested for proper operation before being placed into service.

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

[1] For additional information, refer to NEMA ICS 1.1 (latest edition), "Safety Guidelines for the Application, Installation, and Maintenance of Solid State Control" and to NEMA ICS 7.1 (latest edition), "Safety Standards for Construction and Guide for Selection, Installation and Operation of Adjustable-Speed Drive Systems" or their equivalent governing your particular location.

Before you attempt to provide a solution (machine or process) for a specific application using the POUs found in the library, you must consider, conduct and complete best practices. These practices include, but are not limited to, risk analysis, functional safety, component compatibility, testing and system validation as they relate to this library.

⚠ WARNING

IMPROPER USE OF POUS

- Perform a safety-related analysis for the application and the devices installed.
- Ensure that the POUs are compatible with the devices in the system and have no unintended effects on the proper functioning of the system.
- Use appropriate parameters, especially limit values, and observe machine wear and stop behavior.
- Verify that the sensors and actuators are compatible with the selected POUs.
- Thoroughly test all functions during verification and commissioning in all operation modes.
- Provide independent methods for critical control functions (emergency stop, conditions for limit values being exceeded, etc.) according to a safety-related analysis, respective rules, and regulations.

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

⚠ WARNING

UNINTENDED EQUIPMENT OPERATION

- Only use software approved by Schneider Electric for use with this equipment.
- Update your application program every time you change the physical hardware configuration.

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

Incomplete file transfers, such as data files, application files and/or firmware files, may have serious consequences for your machine or controller. If you remove power, or if there is a power outage or communication interruption during a file transfer, your machine may become inoperative, or your application may attempt to operate on a corrupted data file. If a an interruption occurs, reattempt the transfer. Be sure to include in your risk analysis the impact of corrupted data files.

⚠ WARNING

UNINTENDED EQUIPMENT OPERATION, DATA LOSS, OR FILE CORRUPTION

- Do not interrupt an ongoing data transfer.
- If the transfer is interrupted for any reason, re-initiate the transfer.
- Do not place your machine into service until the file transfer has completed successfully, unless you have accounted for corrupted files in your risk analysis and have taken appropriate steps to prevent any potentially serious consequences due to unsuccessful file transfers.

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

Related Documents

Document title	Reference
SoMachine Programming Guide	*EIO0000000067 (ENG)*;
SoMachine Motion Control Library Guide	*EIO0000002221 (ENG)*
Altivar Machine 320 - CANopen Manual	*NVE41309 (ENG)*
Altivar Machine 320 - Installation Manuall	*NVE41289 (ENG)*
Altivar Machine 340 - CANopen Manual	*NVE61655 (ENG)*
Altivar Machine 340 - Installation Manuall	*NVE61069 (ENG)*
Altivar Process 600 - CANopen Manual	*EVA64333 (ENG)*
Altivar Process, Variable Speed Drives ATV630, ATV650 - Installation Manuall	*EVA64301 (ENG)*
Altivar Process 900 - CANopen Manual	*NHA80945 (ENG)*
Altivar Process, Variable Speed Drives ATV930, ATV950 - Installation Manuall	*NHA80932 (ENG)*

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

Terminology Derived from Standards

The technical terms, terminology, symbols and the corresponding descriptions in this manual, or that appear in or on the products themselves, are generally derived from the terms or definitions of international standards.

In the area of functional safety systems, drives and general automation, this may include, but is not limited to, terms such as *safety*, *safety function*, *safe state*, *fault*, *fault reset*, *malfunction*, *failure*, *error*, *error message*, *dangerous*, etc.

Among others, these standards include:

Standard	Description
EN 61131-2:2007	Programmable controllers, part 2: Equipment requirements and tests.
ISO 13849-1:2008	Safety of machinery: Safety related parts of control systems. General principles for design.
EN 61496-1:2013	Safety of machinery: Electro-sensitive protective equipment. Part 1: General requirements and tests.
ISO 12100:2010	Safety of machinery - General principles for design - Risk assessment and risk reduction
EN 60204-1:2006	Safety of machinery - Electrical equipment of machines - Part 1: General requirements
EN 1088:2008 ISO 14119:2013	Safety of machinery - Interlocking devices associated with guards - Principles for design and selection
ISO 13850:2006	Safety of machinery - Emergency stop - Principles for design
EN/IEC 62061:2005	Safety of machinery - Functional safety of safety-related electrical, electronic, and electronic programmable control systems
IEC 61508-1:2010	Functional safety of electrical/electronic/programmable electronic safety-related systems: General requirements.
IEC 61508-2:2010	Functional safety of electrical/electronic/programmable electronic safety-related systems: Requirements for electrical/electronic/programmable electronic safety-related systems.
IEC 61508-3:2010	Functional safety of electrical/electronic/programmable electronic safety-related systems: Software requirements.
IEC 61784-3:2008	Digital data communication for measurement and control: Functional safety field buses.
2006/42/EC	Machinery Directive
2014/30/EU	Electromagnetic Compatibility Directive
2014/35/EU	Low Voltage Directive

In addition, terms used in the present document may tangentially be used as they are derived from other standards such as:

Standard	Description
IEC 60034 series	Rotating electrical machines
IEC 61800 series	Adjustable speed electrical power drive systems
IEC 61158 series	Digital data communications for measurement and control – Fieldbus for use in industrial control systems

Finally, the term *zone of operation* may be used in conjunction with the description of specific hazards, and is defined as it is for a *hazard zone* or *danger zone* in the *Machinery Directive* (*2006/42/EC*) and *ISO 12100:2010*.

NOTE: The aforementioned standards may or may not apply to the specific products cited in the present documentation. For more information concerning the individual standards applicable to the products described herein, see the characteristics tables for those product references.

Chapter 1
Example Description

Overview

Short Description

The SoMachine project AltivarPLCopenCANopenExample.project provides an application example which shows how to implement an Altivar variable speed drive controlled via fieldbus CANopen. The application example is based on a programming framework supporting general machine control. The monitor and control functions for the Altivar drives are implemented using the PLCopen function blocks from the library GMC Independent PLCopen MC.

System Requirements and Limitations

The application example has been created with the components listed in the table:

Component	Type and Version
Software	SoMachine V4.3
Controller	TM241CEC24T[1)]
Additional devices	● Altivar 320 ● Altivar 340 ● Altivar 630 ● Altivar 930 All Altivar drives are equipped with the communication module VW3A3608.
1) This example is based on the Modicon M241 Logic Controller, however, in principle, the concepts for the implementation of the Altivar drives on CANopen are the same for other SoMachine controllers. As a precondition, the controller must support the CANopen manager and the GMC Independent PLCopen MC library.	

Chapter 2
Project Information

Overview

This section provides information about the SoMachine project.

What Is in This Chapter?

This chapter contains the following sections:

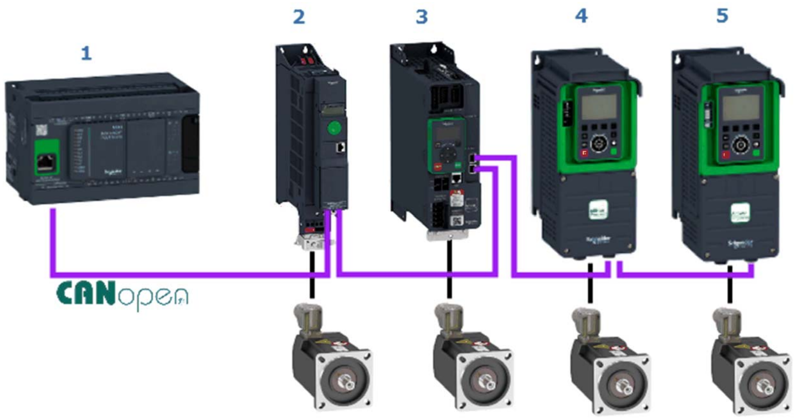
Section 2.1
Hardware Configuration

Overview of the Hardware Configuration

Overview of the Hardware Configuration

The project example implements a Modicon M241 Logic Controller and the four different Altivar variable speed drives. The Altivar drives are linked to the controller via CANopen fieldbus as CANopen slaves. The controller is the CANopen master and implements the logic to control and monitor the drives over the fieldbus.

The figure presents the layout of the network:



Item	Description
1	Modicon M241 Logic Controller
2	Altivar ATV320[1)]
3	Altivar ATV340[1)]
4	Altivar ATV630[1)]
5	Altivar ATV930[1)]
1) The Altivar drives are equipped with the communication module VW3A3608	

Hardware Configuration Procedure

Perform the following steps for the hardware configuration in the SoMachine project.

Step	Action	Comment
1	In the **Devices tree**, double-click the CAN interface and configure the parameter in the device editor.	Configure the transmission rate for the CANopen communication.
2	In the **Devices tree**, add the **CANopen Manager** under the CAN interface of the controller.	Execute the **Add Device …** command from the contextual menu of the CAN interface. In the **Add Device** dialog box, double-click the **CANopen Manager** to add it to the project.
3	Select the **CANopen Manager** node in the **Devices tree**.	The **Add Device** dialog box is updated and lists the devices which can be added.
4	Double-click the Altivar 320 to add it under the **CANopen Manager** node.	–
5	Double-click the Altivar 340 to add it under the **CANopen Manager** node.	–
6	Double-click the Altivar 6xx to add it under the **CANopen Manager** node.	–
7	Double-click the Altivar 9xx to add it under the **CANopen Manager** node.	–
8	Close the **Add Device** dialog box.	–
9	In the **Devices tree**, double-click the previously added device nodes. Adjust the CANopen address for each drive in the respective device editor.	Further configuration of the devices is not necessary for the described application example.

Section 2.2
Application

What Is in This Section?

This section contains the following topics:

Programming Framework

Overview of the Programming Framework

The application example is based on a programming framework which represents a generic implementation for machine control.

The programming framework consists of three categories of components:

Component	Description
MainMachine	The MainMachine provides the framework for the process of the machine control. It implements the typical steps like for example, initialization, reading and processing of the inputs, writing the outputs, and the operating mode selection.
Device/Module Handler	The Device/Module Handler implements the control logic of the specific module or device. It provides an interface which is processed in the single steps of the MainMachine. Further, each module implements a generic exception handler to publish module exceptions and to receive reactions requested because of external exceptions generated by other devices in the machine control.
Reaction Dispatcher	The Reaction Dispatcher monitors exceptions and distributes the reactions based on the rules specified in the Device/Module Handler.

For the interaction of the components inside the programming framework, different shared objects like for example, function blocks, structures, or interfaces are required.

These shared objects in the application example are contained in a folder with name **To be moved to Library**. To apply the introduced programming framework concept for other applications, outsource these objects to a library and reference it in your projects.

SR_Class2ReactionDispatcher Program

Overview of the SR_Class2ReactionDispatcher

The programming framework implements an exception/reaction handling. The exceptions are classified in 3 and the reactions are classified in four classes. The criticality is increasing starting with class 1. Each module handler implements a reaction handling covering each class.

The program SR_Class2ReactionDispatcher implements the logic to monitor the exceptions and to distribute reactions based on specified rules to the machine modules through the global data-structure GVL.g_stExceptionControlData.

The application example implements an instance of the function block FB_Class2ReactionExceptionDispatcher. This function block implements a simple reaction dispatcher with only one global rule for dispatching a reaction. Each exception triggers a reaction on each module depending on its class.

The table lists the resulting reactions for each exception class implemented in the application example.

Exception class	Reaction class	Reaction in module handler
0	0	No reaction
1	2	Stop the drive immediately, keep the power stage on
2	3	Stop the drive immediately, succeeding disable the power stage
3	4	Disable immediately the power stage

SR_Altivarxxx Programs

Overview

The programs SR_Altivar320, SR_Altivar340, SR_Altivar630, and SR_Altivar930 implement the control logic for the associated variable speed drive and provide the interface for the **MainMachine**. Internally an instance of the function block FB_HandlerAtvGmc is called.

The function block uses the library GMC Independent PLCopen MC to control and monitor the drive. The application example implements a selection of the mandatory and typically used motion control function blocks of this library. The libraries GMC Independent PLCopen MC and GMC Independent Altivar provide further function blocks for motion control and device configuration.

The FB_HandlerAtvGmc implements an instance of the function block FB_LocalExceptionHandler. With the use of this function block, the local exceptions are sent to and requested reactions are received from the global reaction dispatcher of the framework through the global data-structure GVL.g_stExceptionControlData.

The FB_HandlerAtvGmc is designed to be integrated in the programming framework environment. However, it can be used also in applications which are not based on the programming framework concept, introduced in the application example.

SR_ExceptionTranslation Program

Overview

The program SR_ExceptionTranslation implements the logic to convert the numeric error codes for each exception into a plain text message which can be displayed in a SoMachine visualization or on an HMI.

Internally an instance of the function block FB_UpdateExpandedExceptionList is called. This function block can reference an exception translator for each type of module.

Each exception is generated with a unique module ID and the device type ID to identify the source of the exception. The function block FB_UpdateExpandedExceptionList converts the error code of an exception into a plain text which is defined in the translator with the same device type ID.

In the application example for each module handler, as well as for the **MainMachine** an exception translator function block is implemented.

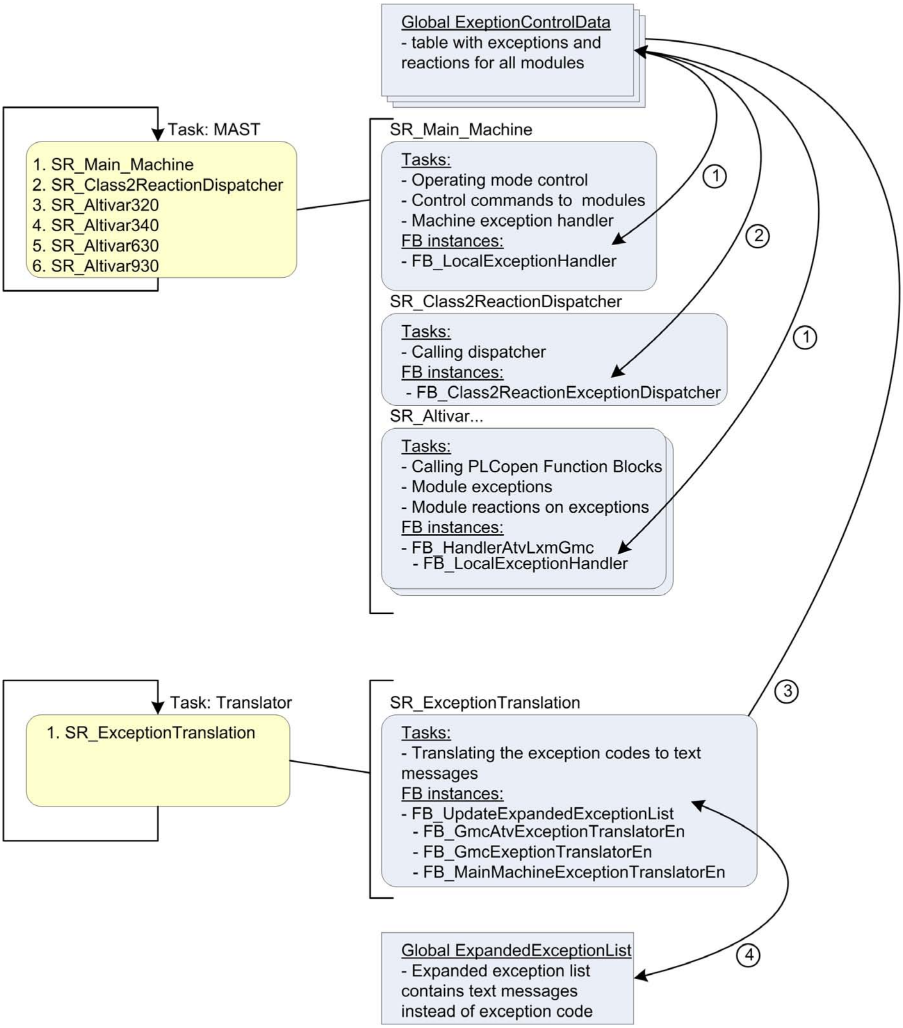
The SR_ExceptionTranslation is called in a separate task with lower priority and a greater task interval than the MAST task.

Task Configuration

Overview

The task configuration of the application example contains beside the default task **MAST** one additional task **Translator**.

The graphic provides an overview about the different tasks and the associated program calls.


Global ExeptionControlData
- table with exceptions and reactions for all modules
Task: MAST
1. SR_Main_Machine
2. SR_Class2ReactionDispatcher
3. SR_Altivar320
4. SR_Altivar340
5. SR_Altivar630
6. SR_Altivar930
SR_Main_Machine
Tasks:
- Operating mode control
- Control commands to modules
- Machine exception handler
FB instances:
- FB_LocalExceptionHandler
SR_Class2ReactionDispatcher
Tasks:
- Calling dispatcher
FB instances:
- FB_Class2ReactionExceptionDispatcher
SR_Altivar...
Tasks:
- Calling PLCopen Function Blocks
- Module exceptions
- Module reactions on exceptions
FB instances:
- FB_HandlerAtvLxmGmc
- FB_LocalExceptionHandler
1
2
1
3
4
Task: Translator
1. SR_ExceptionTranslation
SR_ExceptionTranslation
Tasks:
- Translating the exception codes to text messages
FB instances:
- FB_UpdateExpandedExceptionList
- FB_GmcAtvExceptionTranslatorEn
- FB_GmcExeptionTranslatorEn
- FB_MainMachineExceptionTranslatorEn
Global ExpandedExceptionList
- Expanded exception list contains text messages instead of exception code

Pos.	Description
1	Exceptions and reactions are exchanged through the global exception control data
2	The reaction dispatcher monitors the exceptions and distributes the reactions
3	Translator reads the exceptions from the global list
4	Translator creates an expanded exception list which provides plain text messages

Section 2.3
Visualization

Visualization Screen

Overview

The application example implements a SoMachine visualization to control and to monitor the application. The visualization is available as web visualization.

The web visualization offers you access to machine control functions over the network. To help prevent unauthorized access to your machine control, implement the following technical and organizational measurements for the system running your application.

⚠ WARNING

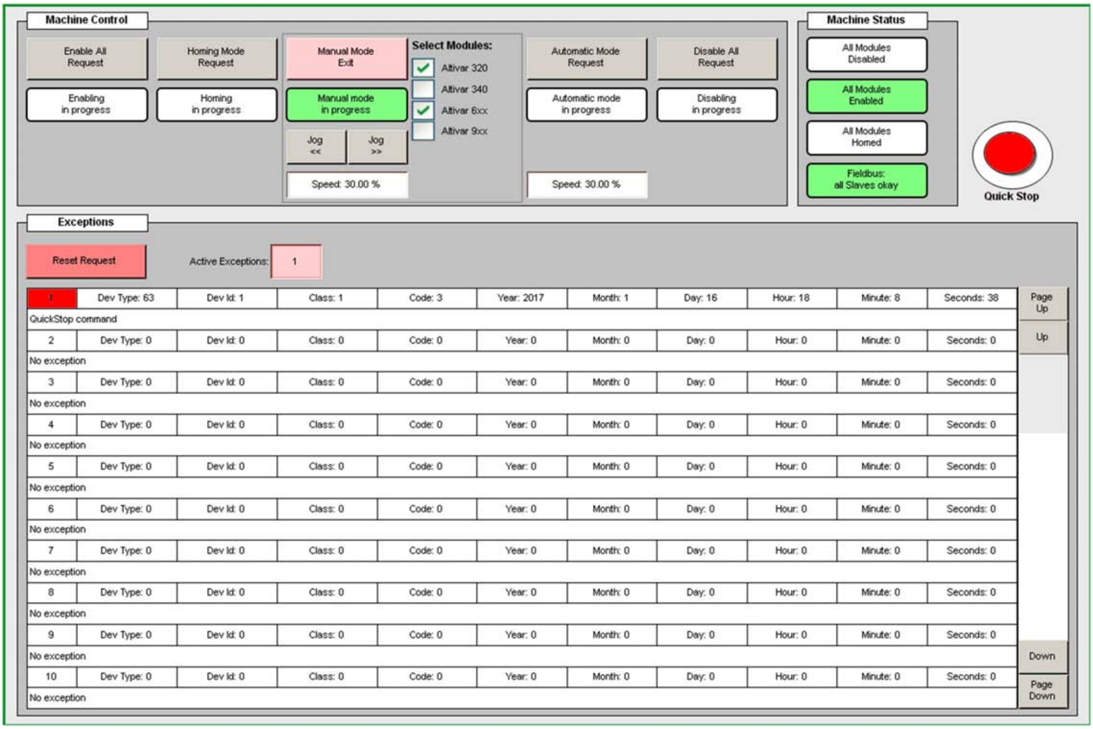
UNAUTHENTICATED, UNAUTHORIZED ACCESS

- Do not expose controllers and controller networks to public networks and the Internet as much as possible.
- Use additional security layers such as VPN for remote access and install firewall mechanisms.
- Restrict access to authorized personnel by activation and deployment of the user management of the controller and the visualization.
- Change default passwords at start-up and modify them frequently.
- Validate the effectiveness of these measurements regularly and frequently.

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

Visualization Screen - VisuMainMachine

A general overview and a main control panel are provided on the visualization screen **VisuMainMachine**.



Select Modules:

☑ Altivar 320
□ Altivar 340
☑ Altivar 6xx
□ Altivar 9xx

Speed: 30.00 %

Speed: 30.00 %

Active Exceptions: 1

1	Dev Type: 63	Dev Id: 1	Class: 1	Code: 3	Year: 2017	Month: 1	Day: 16	Hour: 18	Minute: 8	Seconds: 38
QuickStop command										
2	Dev Type: 0	Dev Id: 0	Class: 0	Code: 0	Year: 0	Month: 0	Day: 0	Hour: 0	Minute: 0	Seconds: 0
No exception										
3	Dev Type: 0	Dev Id: 0	Class: 0	Code: 0	Year: 0	Month: 0	Day: 0	Hour: 0	Minute: 0	Seconds: 0
No exception										
4	Dev Type: 0	Dev Id: 0	Class: 0	Code: 0	Year: 0	Month: 0	Day: 0	Hour: 0	Minute: 0	Seconds: 0
No exception										
5	Dev Type: 0	Dev Id: 0	Class: 0	Code: 0	Year: 0	Month: 0	Day: 0	Hour: 0	Minute: 0	Seconds: 0
No exception										
6	Dev Type: 0	Dev Id: 0	Class: 0	Code: 0	Year: 0	Month: 0	Day: 0	Hour: 0	Minute: 0	Seconds: 0
No exception										
7	Dev Type: 0	Dev Id: 0	Class: 0	Code: 0	Year: 0	Month: 0	Day: 0	Hour: 0	Minute: 0	Seconds: 0
No exception										
8	Dev Type: 0	Dev Id: 0	Class: 0	Code: 0	Year: 0	Month: 0	Day: 0	Hour: 0	Minute: 0	Seconds: 0
No exception										
9	Dev Type: 0	Dev Id: 0	Class: 0	Code: 0	Year: 0	Month: 0	Day: 0	Hour: 0	Minute: 0	Seconds: 0
No exception										
10	Dev Type: 0	Dev Id: 0	Class: 0	Code: 0	Year: 0	Month: 0	Day: 0	Hour: 0	Minute: 0	Seconds: 0
No exception										

In addition, for each Altivar module a separate visualization screen is implemented. These visualization screens provide extensive control functions which can be used during commissioning phase.

Section 2.4
Running the Application Example

What Is in This Section?

This section contains the following topics:

Preconditions for Running the Application Example

Overview

To run the application example, the following preconditions must be fulfilled:

- The application was downloaded and is running on the associated controller.
- The Altivar drives are configured for the control over the associated fieldbus. Further configuration of the drive is compatible with the connected motor and driven mechanical parts.
- A PC, laptop, or mobile device with a compatible web browser is connected to the same network as the controller so that the web visualization can be accessed.

Downloading the Application

Overview

To set the communication parameter of the controller, and to download the applications, perform these steps:

Step	Action	Comment
1	Connect your PC to the controller via Ethernet.	The controller and the PC must be connected to the same Ethernet network.
2	Open the project example in SoMachine and open the Logic Builder.	–
3	Double-click the **MyController** node in the **Devices tree**.	A network scan is performed automatically. Available controllers are provided in the **Controller selection** view of the device editor.
4	Select the target controller from the list.	–
5	Right-click the controller entry and execute the command **Process communication settings...** from the contextual menu.	–
6	Enter the **Communication parameter**.	–
7	Activate the option **Save settings permanently** and click the **OK** button.	**Result**: The communication parameters are stored on the controller. They are retained even if it is restarted. These parameters override the application settings.
8	Execute the **Multiple Download...** command from the **Online** menu.	–
9	In the **Multiple Download** dialog box, select the application, and click the **OK** button.	**Result**: The application is downloaded to the previously selected controller.

For more information on downloading an application and the communication settings, refer to the SoMachine Programming Guide *(see SoMachine, Programming Guide)*.

Operating the Application

Overview

The application examples implement a web visualization for monitoring and control purposes.

To open the web visualization, enter the following address in the address bar of a web browser:

```
http://<IP Address>:8080\webvisu.htm
```

This section provides a brief description of each function. In addition, refer to the visualization screen *(see page 26)*.

Start-up of the Drives

The application example is based on the programming framework which implements the essential operating modes. To operate the drives, follow this sequence:

1. **Enable**: Enables the power stage of all drives.
2. **Homing**: Referencing of all axes.
3. If all axes are enabled and referenced, you can operate the machine in manual or automatic mode.

NOTE: If no operating mode is in progress, the power stage of all drives can be powered off with the **Disable Request** command.

Manual Mode

The following operations are available:

- Click the **Manual Mode Request** button to enter the manual mode.
- Click the **Manual Mode Exit** button to leave the manual mode.
- Select the drives you want to operate in Jog-Mode.
- Select the speed for jog operation.
- Click the **JogFwd** or **JogRev** buttons to start a movement of connected motor in the according direction.

  NOTE: Only if manual mode is in progress the buttons are used to control the drives.

  The movement ends as soon you release the respective button.

Automatic Mode

The following operations are available:

- Click the **Automatic Mode Request** button to enter in automatic mode.
- Click the **Automatic Mode Exit** button to leave the automatic mode.
- Select the speed for automatic mode.
- Click the **Automatic Mode Start** button to start the automatic sequence.
- Click the **Automatic Mode Stop** button to stop the automatic sequence.
- While automatic mode is in progress, you can adjust the speed setpoint. Accordingly the speed of the drives is changed immediately.

Quick Stop

The visualization **VisuMainMachine** provides a **Quick Stop** button. If you click this button an exception of class 1 is generated. This exception requests a reaction of class 2 for all modules, which causes an immediately stop for all drives while the power stage stays enabled.

Exception List

The visualization **VisuMainMachine** provides an exception list in which all active exceptions are listed with the information about the source, the time, and the description of it. By clicking the **Reset** button all exceptions are reset, preconditioned the cause of the exception has been cleared before.

Glossary

A

application

A program including configuration data, symbols, and documentation.

C

CAN

(*controller area network*) A protocol (ISO 11898) for serial bus networks, designed for the interconnection of smart devices (from multiple manufacturers) in smart systems and for real-time industrial applications. Originally developed for use in automobiles, CAN is now used in a variety of industrial automation control environments.

CANopen

An open industry-standard communication protocol and device profile specification (EN 50325-4).

configuration

The arrangement and interconnection of hardware components within a system and the hardware and software parameters that determine the operating characteristics of the system.

controller

Automates industrial processes (also known as programmable logic controller or programmable controller).

E

Ethernet

A physical and data link layer technology for LANs, also known as IEE 802.3.

Ethernet is the most widely spread technology for local networks. Each PacDrive controller has an Ethernet port. The Ethernet standard defines layer 1 and 2 of the communication. Above the Ethernet, there are many different network protocols but only IP is used.

expansion bus

An electronic communication bus between expansion I/O modules and a controller.

F

function block

A programming unit that has 1 or more inputs and returns 1 or more outputs. FBs are called through an instance (function block copy with dedicated name and variables) and each instance has a persistent state (outputs and internal variables) from 1 call to the other.

Examples: timers, counters

I

I/O

(*input/output*)

M

MAST

A processor task that is run through its programming software. The MAST task has 2 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.

P

PLCopen

For more information, refer to *http://www.plcopen.org/*.

program

The component of an application that consists of compiled source code capable of being installed in the memory of a logic controller.

T

task

A group of sections and subroutines, executed cyclically or periodically for the MAST task or periodically for the FAST task.

A task possesses a level of priority and is linked to inputs and outputs of the controller. These I/O are refreshed in relation to the task.

A controller can have several tasks.

Index