Modicon M340

Motion Function Block Start-up Guide

(Original Document)

12/2018

35013563.09

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.

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.

About the Book

At a Glance

Document Scope

This manual presents, using a documented example, how to use motion function blocks (MFB) with Modicon M340 using EcoStruxure™ Control Expert. These blocks enable simplified management of servodrives and servo-amplifiers using the CANopen bus.

Expert knowledge of EcoStruxure™ Control Expert software is required in order to use MFBs with it, since their implementation requires use of its standard functions (data editor, IODDT, etc.).

Moreover, it is advisable to have expert knowledge of the specialist area of motion control before developing and commissioning an application involving implementation of axis movements.

Validity Note

This document is valid for EcoStruxure™ Control Expert 14.0 or later.

Related Documents

Title of documentation	Reference number
EcoStruxure™ Control Expert, Motion Function Blocks, Block Library	35010605 (English), 35010606 (French), 35010607 (German), 35010609 (Italian), 35010608 (Spanish), 35012310 (Chinese)
Premium using EcoStruxure™ Control Expert, Motion Function Blocks, Start-up Guide	35010601 (English), 35010602 (French), 35010603 (German), 35010600 (Italian), 35010604 (Spanish), 35012309 (Chinese)
Modicon M340, CANopen, Setup Manual	35013944 (English), 35013945 (French), 35013946 (German), 35013948 (Italian), 35013947 (Spanish), 35013949 (Chinese)

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

Part I

Start-up Guide for a Single Axis Application

Subject of this Part

This Part presents, in the form of a tutorial, an example of a motion control application implementing MFBs using Control Expert.

What Is in This Part?

This part contains the following chapters:

Chapter 1
Foreword

Subject of this Chapter

This chapter presents the specifications of the application as well as the methodology used in its development.

What Is in This Chapter?

This chapter contains the following topics:

General

Introduction

The MFB using Control Expert offer is a new motion control functionality. Using the CANopen bus, it provides you with simplified access to the basic functions on servodrives and variable speed drive (VSD).

This functionality, which may be accessed via the project browser, allows you to:

- declare and configure axes in Control Expert
- create motion control variables
- control the axes by using motion control elementary function blocks.

Specifications

The purpose of the proposed application is to:

- manage the operating modes of a linear axis using a **Lexium 05**-type servodrive.
- move the axis to the home position, carry out reversing movements or move the axis to various positions
- provide the possibility of interrupting the motion in progress with a Stop command.

All provisions shall be taken to perform fault diagnostics and acknowledgement.

Standards

The MFB library blocks comply with:

- PLCopen standard

Availability of Blocks on the Various Servodrives

Motion Function Blocks

The motion function blocks availability can be found in the following tables.

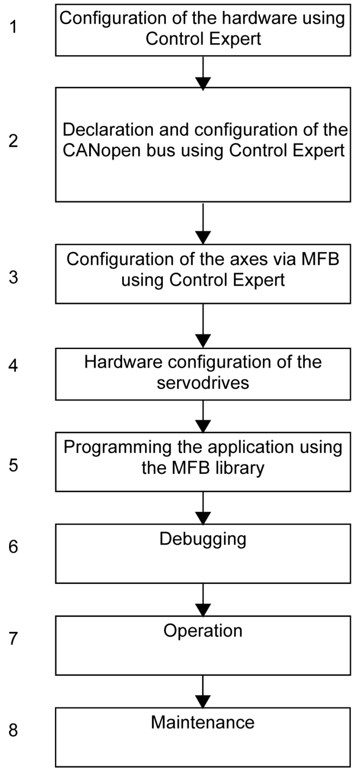
Type	Block name	ATV 31 ATV312 (7.)	ATV 32	ATV 71	Lexium 32, 32i	Lexium 05	Lexium 15 HP, MP, LP	IclA IFA, IFE, IFS
PLCopen motioncontrol V1.1	MC_ReadParameter	X	X	X	X	X	X	X
	MC_WriteParameter	X	X	X	X	X	X	X
	MC_ReadActualPosition				X	X	X	X
	MC_ReadActualVelocity (1.)	X	X	X	X	X	X	X
	MC_Reset	X	X	X	X	X	X	X
	MC_Stop	X	X	X	X	X	X	X
	MC_Power	X	X	X	X	X	X	X
	MC_MoveAbsolute				X	X	X	X
	MC_MoveRelative				X	X	X	
	MC_MoveAdditive				X	X		X
	MC_Home				X	X	X	X
	MC_MoveVelocity	X	X	X	X	X	X	X
	MC_ReadAxisError	X	X	X	X	X	X	X
	MC_ReadStatus	X	X	X	X	X	X	X
	MC_TorqueControl (1.)			X	X	X	X (3.)	
	MC_ReadActualTorque (1.)	X	X	X	X	X	X	
	MC_Jog (2.)				X	X	X (3.), except 15 LP	X
Parameter set save and restore functions for management of recipes or replacement of faulty servodrives	TE_UploadDriveParam	X	X	X	X(6.), except 32i	X	X	X
	TE_DownloadDriveParam	X	X	X	X(6.), except 32i	X	X	X

Type	Block name	ATV 31 ATV312 (7.)	ATV 32	ATV 71	Lexium 32, 32i	Lexium 05	Lexium 15 HP, MP, LP	IclA IFA, IFE, IFS
Advanced functions for the **Lexium**	Lxm_GearPos					X(4.)	X(5.)	
	Lxm_GearPosS				X	X(4.)	X(5.)	
	Lxm_UploadMTask						X	
	Lxm_DownloadMTask						X	
	Lxm_StartMTask				X		X	
System function	CAN_Handler	X	X	X	X	X	X	X
1. PLCopen V0.99 extension part 2 2. Not PLCopen standard 3. Only for firmware version >= 6.73 4. Only for firmware version >= 1.403 5. Only for firmware version >= 2.36 6. The parameter list is a Lexium32Advanced drive parameter list 7. Through an ATV 31 V1.7 CANopen Device configuration.								

Methodology

Overview

The flowchart below lists the various stages involved in installing the application:


1
Configuration of the hardware using Control Expert
2
Declaration and configuration of the CANopen bus using Control Expert
3
Configuration of the axes via MFB using Control Expert
4
Hardware configuration of the servodrives
5
Programming the application using the MFB library
6
Debugging
7
Operation
8
Maintenance

The table below details the tasks to be performed for each stage of the flowchart:

Step	Description
1	In Control Expert: ● create the project and select the processor
2	In Control Expert: ● open a CANopen bus configuration ● choose the CANopen slave in hardware catalog ● attribute a topological address to the new device ● check or set MFB function in the configuration window of device ● enable CANopen configuration ● check the accuracy of the configuration using the CANopen configuration tree structure in the project browser.
3	Create the axes in the project browser's **Motion** directory. Define the variables associated with these axes during their creation
4	With the PowerSuite software: ● connect to the device ● enter the required parameters for the correct operation of the CANopen communication (address, speed, etc.).
5	Program the motion sequences using the appropriate functions blocks from the MFB library. Associate the variables defined during creation of the axis with the MFB blocks.
6	Debug the axis using PowerSuite. In Control Expert: ● debug the program via the animation table ● use the data via the operator screens
7	manage the production recipes using the appropriate function blocks from the MFB library: ● create and back up the recipes ● transfer data from the recipes
8	Data backup and restore procedures.

Chapter 2
Application Configuration

Subject of this Chapter

This chapter describes the various stages involved in configuring the application.

What Is in This Chapter?

This chapter contains the following sections:

Section 2.1
Hardware and Software Environments

Subject of this Section

This sub-section describes the hardware and software environments used in the application.

What Is in This Section?

This section contains the following topics:

Application Architecture with a Lexium 05

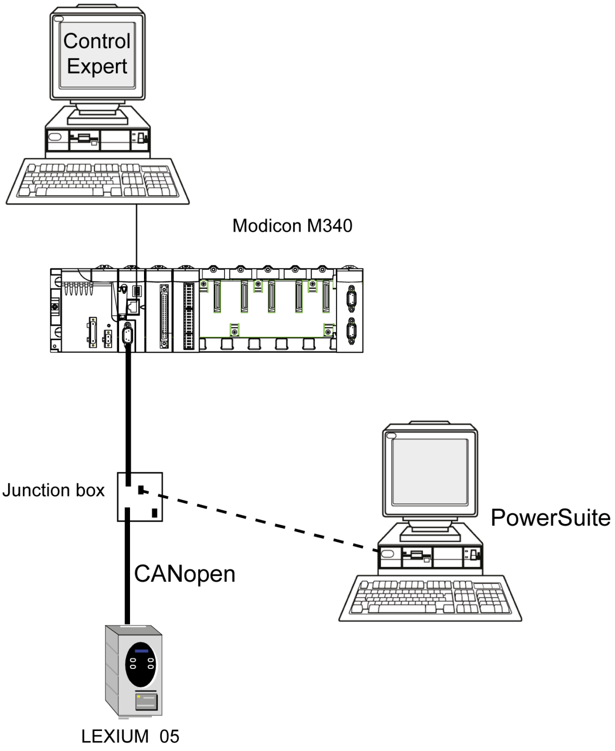
Overview

The proposed architecture is simple and designed to assimilate the implementation principles of motion control.

Other equipment can be added to this realistic architecture in order to manage several axes.

Illustration

The following figure shows the architecture used in the application that includes a **Lexium 05**.



Software Requirements

Overview

To implement the example, it is essential to have certain items of software on single PC. In particular, this will allow you to configure, set parameters for and operate the various devices used.

The software architecture is composed of:

- Control Expert, which is used to control the servodrive via the CANopen bus by programming movements
- Powersuite, which is used to set parameters and adjust the Lexium 05 servodrive

It is nonetheless possible to go without PowerSuite in certain cases by using the **Lexium 05** front panel user interface *(see page 52)*.

Versions

The following table lists the hardware and software versions used in the architecture *(see page 21)*, enabling the use of MFBs in Control Expert.

Hardware	Software version used in the example	Firmware Version
Modicon M340	Unity Pro V5.0	-
Lexium 05	PowerSuite for **Unity V5.0** V2.5, patch V2.2.0B	V1.403

NOTE: Unity Pro is the former name of Control Expert for version 13.1 or earlier.

Hardware Requirements

References of the Hardware Used

The following table lists the hardware used in the architecture *(see page 21)*, enabling implementation of **Lexium 05** MFBs in Control Expert.

Hardware	Reference
Modicon M340 PLC	BMX P34 2030
Modicon M340 power supply	BMX CPS 2000
Modicon M340 rack	BMX XBP 0800
CANopen junction box between the **Modicon M340** and **Lexium 05** servodrive	VW3CANTAP2
RJ45 programming cable with RS485/RS232 adapter between the junction box and servodrive	ACC2CRAAEF030
Lexium 05 servodrive	LXM05AD10M2
Lexium 05 motor	BSH0551T

NOTE: The terminating resistor is integrated in the **Lexium 05**.

Section 2.2
Configuration of the Application using Control Expert

Subject of this Section

This sub-section describes the hardware configuration using Control Expert.

What Is in This Section?

This section contains the following topics:

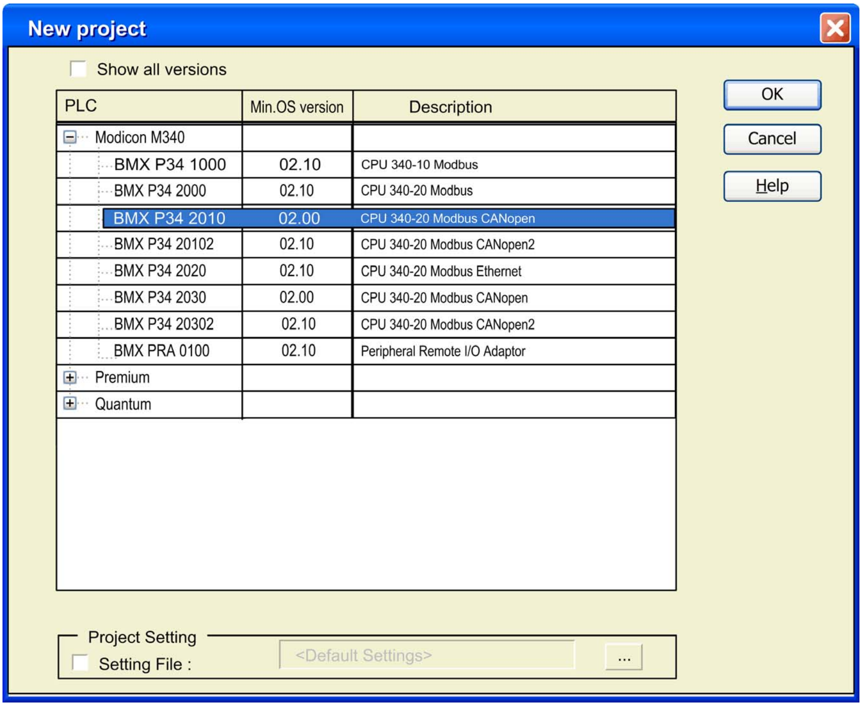
Creating the Project

At a Glance

Developing an application using Control Expert involves creating a project associated with a PLC.

Procedure for Creating a Project

The table below shows the procedure for creating the project using Control Expert.

<table>
<tr><th>Step</th><th>Action</th></tr>
<tr><td>1</td><td>Launch the Control Expert software,</td></tr>
<tr><td>2</td><td>Click on <b>File</b> then <b>New</b> then select a PLC,


New project
□ Show all versions
<table>
<tr><th>PLC</th><th>Min.OS version</th><th>Description</th></tr>
<tr><td>Modicon M340</td><td></td><td></td></tr>
<tr><td>BMX P34 1000</td><td>02.10</td><td>CPU 340-10 Modbus</td></tr>
<tr><td>BMX P34 2000</td><td>02.10</td><td>CPU 340-20 Modbus</td></tr>
<tr><td>BMX P34 2010</td><td>02.00</td><td>CPU 340-20 Modbus CANopen</td></tr>
<tr><td>BMX P34 20102</td><td>02.10</td><td>CPU 340-20 Modbus CANopen2</td></tr>
<tr><td>BMX P34 2020</td><td>02.10</td><td>CPU 340-20 Modbus Ethernet</td></tr>
<tr><td>BMX P34 2030</td><td>02.00</td><td>CPU 340-20 Modbus CANopen</td></tr>
<tr><td>BMX P34 20302</td><td>02.10</td><td>CPU 340-20 Modbus CANopen2</td></tr>
<tr><td>BMX PRA 0100</td><td>02.10</td><td>Peripheral Remote I/O Adaptor</td></tr>
<tr><td>Premium</td><td></td><td></td></tr>
<tr><td>Quantum</td><td></td><td></td></tr>
</table>
Project Setting
□ Setting File : <Default Settings> ...

</td></tr>
<tr><td>3</td><td>To see all PLC versions, click on the box Show all versions.</td></tr>
<tr><td>4</td><td>Select the processor you wish to use from those proposed.</td></tr>
<tr><td>5</td><td>To create a project with specific values of project settings, check the box <b>Settings File</b> and use the browser button to localize the .XSO file (Project Settings file). It is also possible to create a new one.
If the <b>Settings File</b> box is not checked , default values of project settings are used.</td></tr>
<tr><td>6</td><td>Confirm by clicking <b>OK</b>. The application inserts a rack and a power supply by default.</td></tr>
</table>

Master Task Configuration

General

The first operation you need to perform to create a program is to select the type of **Tasks**.

You are advised to program the servodrive movements using MFB blocks in the **MAST** task. This task must be scanned at regular intervals.

⚠ CAUTION

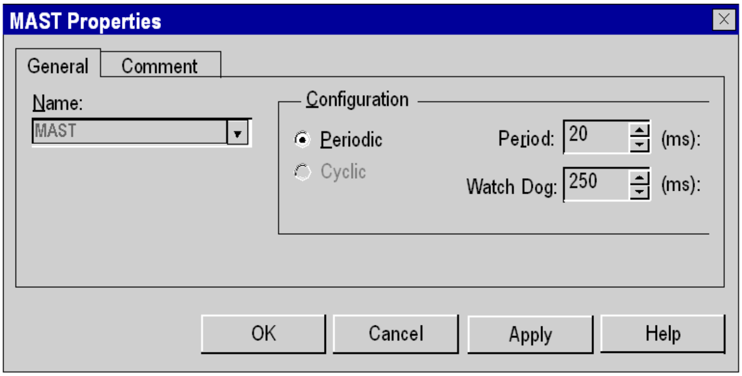
MFB BLOCKS UNEXPECTED BEHAVIOR

Do not mixe MAST and FAST tasks. It is possible to use the FAST task to program the MFBs.

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

Configuration

The following table describes the procedure for setting the parameters of the **MAST** task:

Step	Action
1	In the **Project Browser**, expand the **Program** directory. The **MAST** directory is displayed.
2	Right-click on the **MAST** directory and then execute the **Properties** command in the contextual menu.
3	Click on **Properties** and the following dialog box appears:  MAST Properties Name: MAST Configuration ● Periodic ○ Cyclic Period: 20 (ms): Watch Dog: 250 (ms):
4	Select the **Periodic** type of scanning.
5	Set the task period to 20.
6	Set the **Watchdog** value, which must be greater than the period value.
7	Click on **OK** to confirm the configuration.

Section 2.3
CANopen Bus Configuration

Subject of this Section

This section presents the CANopen bus configuration methodology.

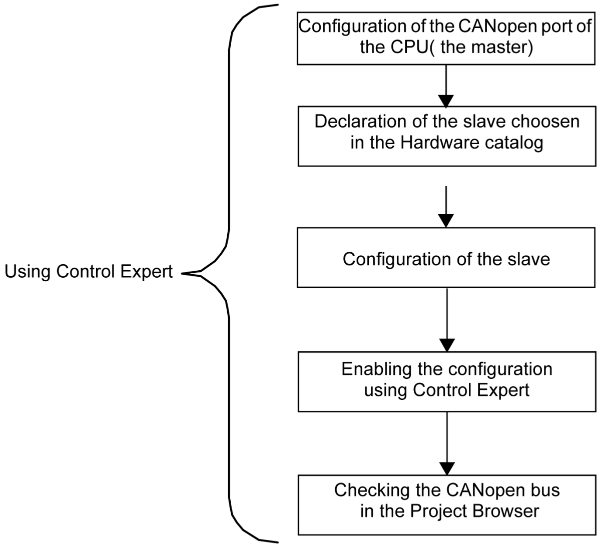
What Is in This Section?

This section contains the following topics:

Implementation Methodology for a CANopen Bus

Overview

The following flowchart shows the implementation methodology for a CANopen bus using Modicon M340.


Using Control Expert
Configuration of the CANopen port of the CPU(the master)
Declaration of the slave choosen in the Hardware catalog
Configuration of the slave
Enabling the configuration using Control Expert
Checking the CANopen bus in the Project Browser

Configuration of the CANopen port

At a Glance

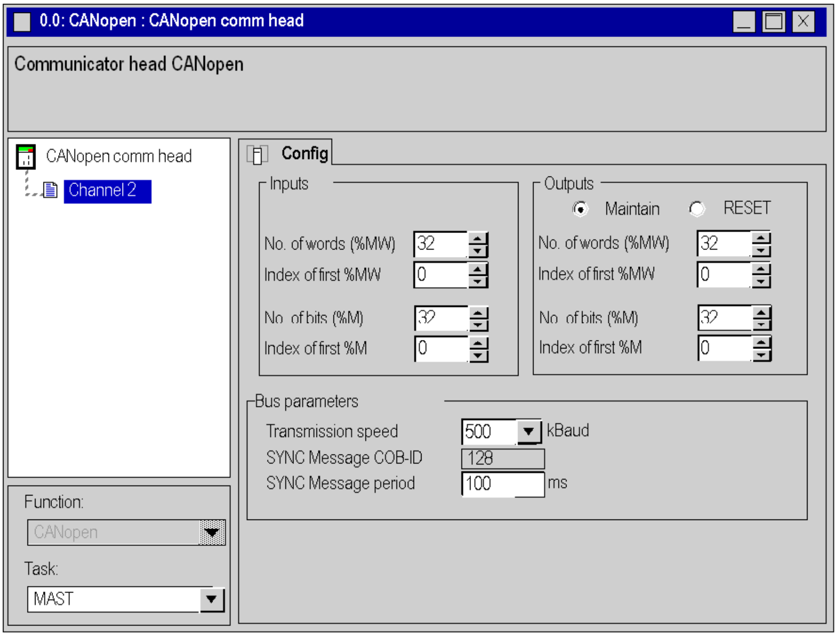
With Control Expert you can define the CANopen bus.

The CANopen bus master is a port integrated in the CPU.

First, the bus master must be configured.

How to Configure the CANopen Bus Master

This table describes the procedure to configure the CANopen port using Control Expert.

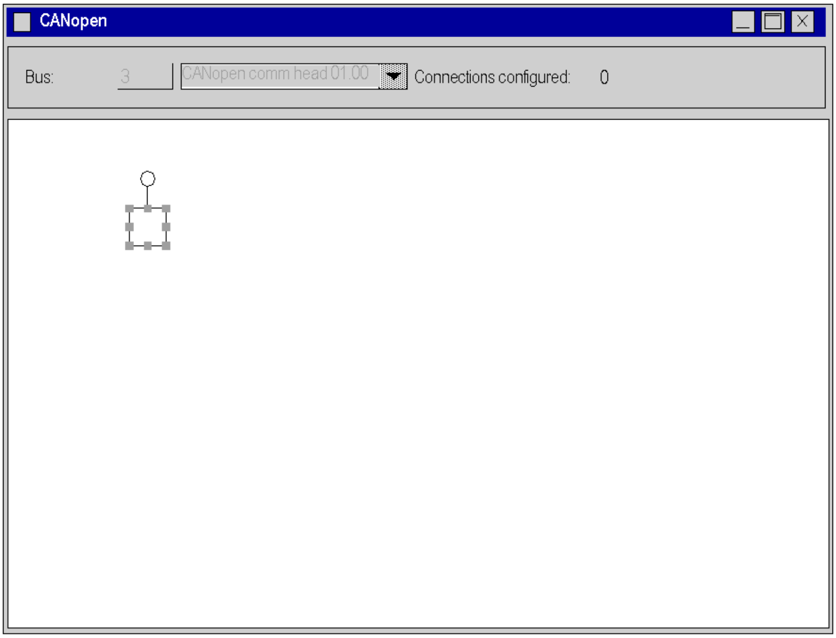
Step	Action
1	In the Control Expert **Project Browser**, fully expand the **Configuration** directory and then double-click on PLC bus.
2	Double-click on CANopen port of PLC. **Result:** The port configuration window appears:  0.0: CANopen : CANopen comm head Communicator head CANopen CANopen comm head Channel 2 Config Inputs No. of words (%MW) 32 Index of first %MW 0 No. of bits (%M) 32 Index of first %M 0 Outputs ● Maintain ○ RESET No. of words (%MW) 32 Index of first %MW 0 No. of bits (%M) 32 Index of first %M 0 Bus parameters Transmission speed 500 kBaud SYNC Message COB-ID 128 SYNC Message period 100 ms Function: CANopen Task: MAST

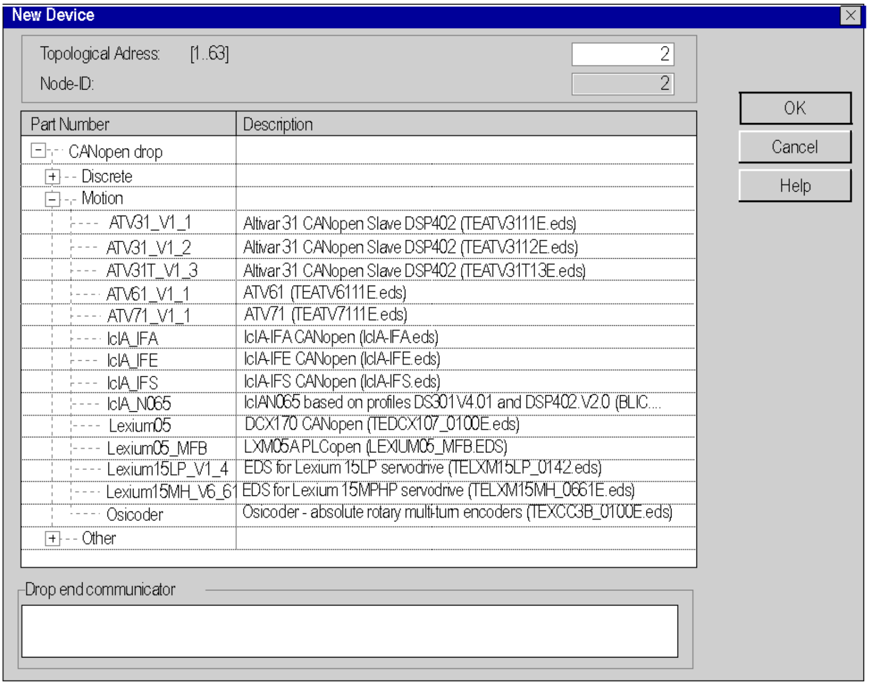
Step	Action
3	In the **Bus parameters** area, set 500 kBaud for the **transmission speed**. In the **Task** area, select **MAST**. In the **Outputs** area select **Reset** radio-button. (Strongly recommended)
4	Validate the configuration.
5	**Note:** We recommend using the IODDT T_COM_CO_BMX that corresponds to the CANopen port for the rest of the programming. Enter CAN for the **prefix name**. Close the window.

Configuration of the CANopen Slave

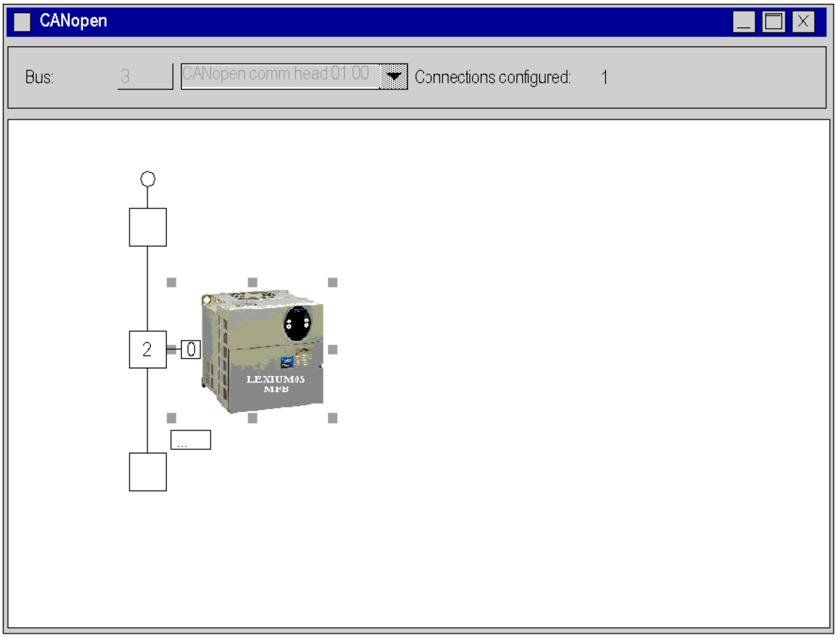
How to Configure the CANopen Slave

This table describes the procedure to configure the CANopen slave.

Step	Action
1	In the Control Expert **Project Browser**, fully expand the **Configuration** directory and then double-click on **CANopen**. **Result:** The CANopen window appears:  CANopen Bus: 3 CANopen comm head 01.00 Connections configured: 0

<table>
<tr><th>Step</th><th>Action</th></tr>
<tr><td>2</td><td>Select <b>Edit</b> → <b>New device</b>.
<b>Result:</b> The New Device window appears:


Topological Adress: [1..63] 2
Node-ID: 2
<table>
<tr><th>Part Number</th><th>Description</th></tr>
<tr><td>CANopen drop</td><td></td></tr>
<tr><td>Discrete</td><td></td></tr>
<tr><td>Motion</td><td></td></tr>
<tr><td>ATV31_V1_1</td><td>Altivar 31 CANopen Slave DSP402 (TEATV3111E.eds)</td></tr>
<tr><td>ATV31_V1_2</td><td>Altivar 31 CANopen Slave DSP402 (TEATV3112E.eds)</td></tr>
<tr><td>ATV31T_V1_3</td><td>Altivar 31 CANopen Slave DSP402 (TEATV31T13E.eds)</td></tr>
<tr><td>ATV61_V1_1</td><td>ATV61 (TEATV6111E.eds)</td></tr>
<tr><td>ATV71_V1_1</td><td>ATV71 (TEATV7111E.eds)</td></tr>
<tr><td>IclA_IFA</td><td>IclA-IFA CANopen (IclA-IFA.eds)</td></tr>
<tr><td>IclA_IFE</td><td>IclA-IFE CANopen (IclA-IFE.eds)</td></tr>
<tr><td>IclA_IFS</td><td>IclA-IFS CANopen (IclA-IFS.eds)</td></tr>
<tr><td>IclA_N065</td><td>IclAN065 based on profiles DS301 V4.01 and DSP402 V2.0 (BLIC....</td></tr>
<tr><td>Lexium05</td><td>DCX170 CANopen (TEDCX107_0100E.eds)</td></tr>
<tr><td>Lexium05_MFB</td><td>LXM05A PLCopen (LEXIUM05_MFB.EDS)</td></tr>
<tr><td>Lexium15LP_V1_4</td><td>EDS for Lexium 15LP servodrive (TELXM15LP_0142.eds)</td></tr>
<tr><td>Lexium15MH_V6_61</td><td>EDS for Lexium 15MPHP servodrive (TELXM15MH_0661E.eds)</td></tr>
<tr><td>Osicoder</td><td>Osicoder - absolute rotary multi-turn encoders (TEXCC3B_0100E.eds)</td></tr>
<tr><td>Other</td><td></td></tr>
</table>
Drop end communicator</td></tr>
<tr><td>3</td><td>Set 2 in Topological Address.
Choose Lexium05_MFB for the slave device.</td></tr>
</table>

Step	Action
4	Click on OK to confirm the choice. **Result:** The CANopen window appears with the new device selected:  CANopen Bus: 3 CANopen comm head 01.00 Connections configured: 1
5	Select **Edit** → **Open module**. If MFB has not already been selected, choose it in the Function area.
6	Select the tab **Error Control**. Verify that **Node Heartbeat Producer time** value is equal to 300ms.
7	You will be asked to validate your modifications when closing the Device and CANopen windows.

Checking the CANopen Bus Configuration

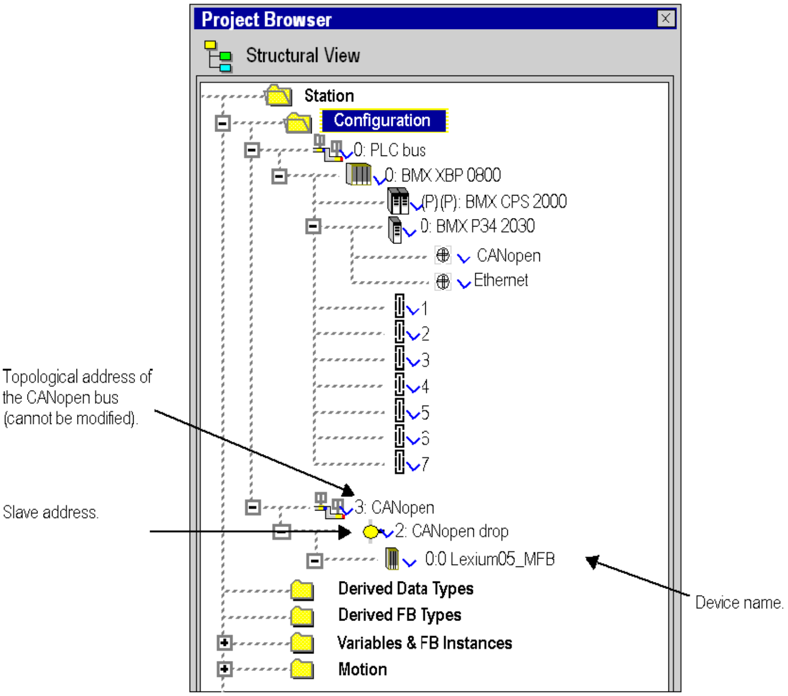
At a Glance

The CANopen bus is represented in the **Configuration** directory of the project browser.

After having selected and enabled the CANopen configuration, the CANopen slaves appear in the **Project Browser**.

The topological address of the CANopen bus is calculated automatically by Control Expert. This value cannot be modified.

The diagram below shows the CANopen bus with slave device from the tutorial example:



Section 2.4
Axis Configuration using the Motion Tree Manager

Subject of this Section

This sub-section describes the **Motion** directory added to Control Expert project browser. It also presents a procedure for creating the axis in this directory.

What Is in This Section?

This section contains the following topics:

Motion Directory

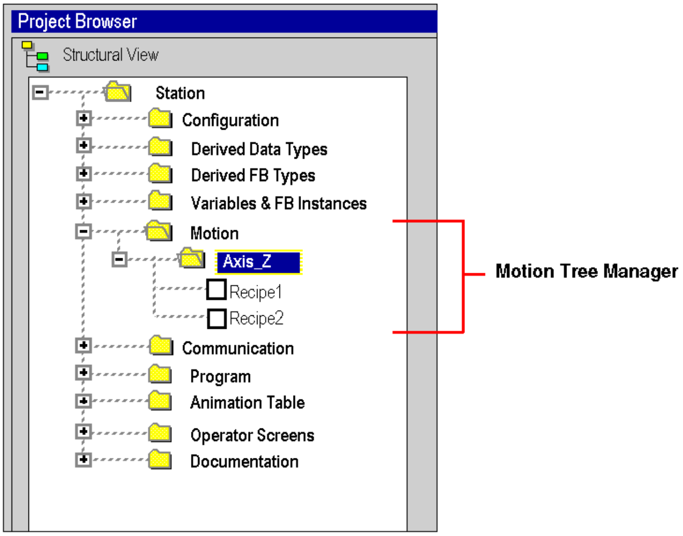
At a Glance

The **Motion** directory of the structural view of the project allows you to access the declaration and configuration of the servodrives.

When declaring a servodrive, various information is required, such as:

- the name given to the servodrive
- the type of servodrive
- the CANopen address of the servodrive
- the reference of the servodrive
- the version of the servodrive
- the input of variable names associated to the axis.

The following diagram shows an example of a tree structure for the **Motion** directory:



In this diagram, the name given to the servodrive is ‘Axis_Z’.

A recipe is linked, by default, each time an axis is created. It is possible to create several recipes *(see page 68)*.

Accessible Services

The **Motion** directory gives you access to the following services, which can be reached via the contextual menu:

Directory	Service
Motion	**New axis**: allows you to create a new axis.
Axis	**New recipe**: allows you to create a new recipe. **Delete**: allows you to delete an axis. **Properties**: allows you to access the axis properties.
Recipe	**Delete**: allows you to delete a recipe. **Properties**: allows you to access the recipe properties.

Axis Creation and Configuration

General

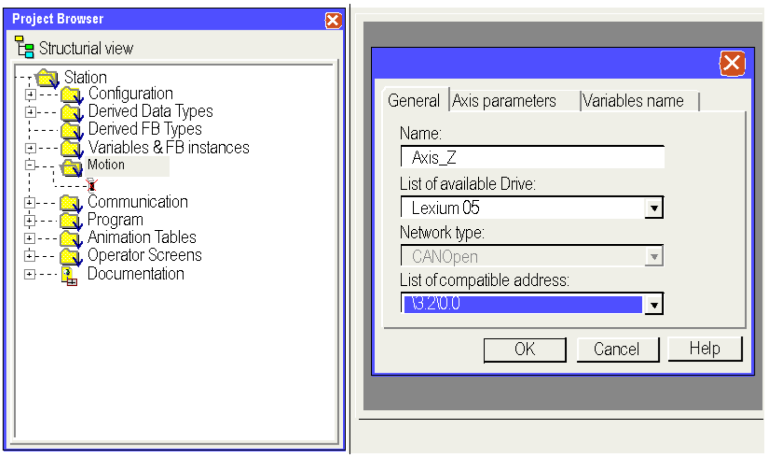
The **Motion** directory is used to declare an axis.

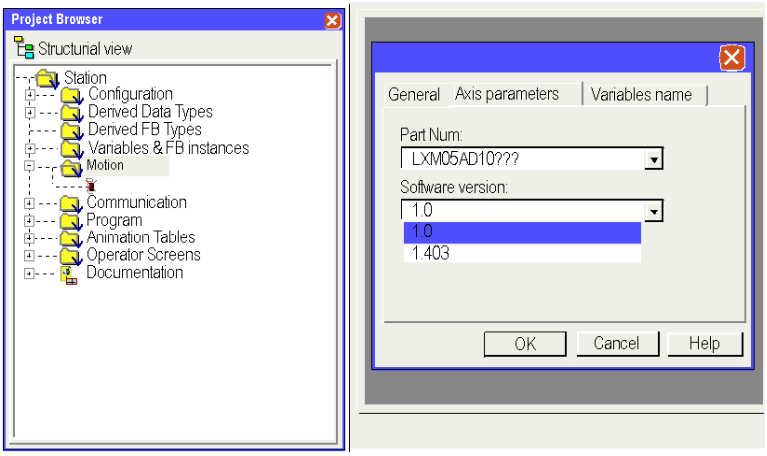
This creation allows you to simplify the management and programming of an axis using Control Expert.

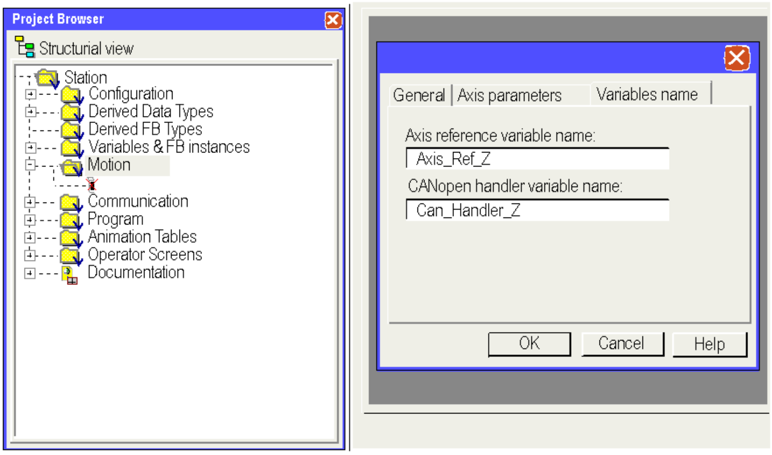
NOTE: For any modification to a device on the CANopen bus, those servodrives unaffected by the modification do not need to be reconfigured.

Creating an Axis

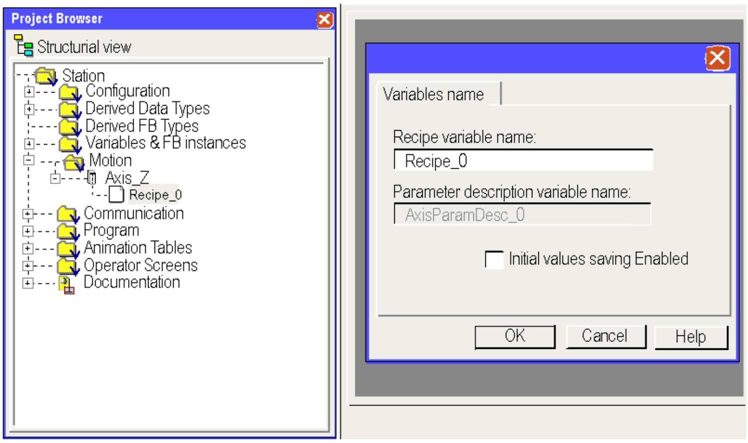
Carry out the following actions:

Step	Action
1	Right-click on the **Motion** directory and then execute the **New axis** command in the contextual menu.
2	Clicking on the **New axis** command will open a dialog box with three tabs.
3	In the **General** tab, ● enter: ❍ a name ● select: ❍ a servodrive from the list ❍ a compatible CANopen address **Note**: If the CANopen addresses have not as yet been defined, leave **<No link>** in the list. It is possible to continue development of the application if **<No link>** is assigned to a compatible CANopen address. When the CANopen addresses have been defined, select an adress in the compatible drive list. In this tab, the **Axis_Z** is configured as follows:  Project Browser Structurial view Station Configuration Derived Data Types Derived FB Types Variables & FB instances Motion Communication Program Animation Tables Operator Screens Documentation General Axis parameters Variables name OK Cancel Help Name: Axis_Z List of available Drive: Lexium 05 Network type: CANOpen List of compatible address: \3.2\0.0

Step	Action
4	In the **Axis Parameters** tab, select: ● the reference of the servodrive ● the minimum version of the servodrive's firmware In this tab, the **Axis_Z** is configured as follows:  Project Browser Structurial view Station Configuration Derived Data Types Derived FB Types Variables & FB instances Motion Communication Program Animation Tables Operator Screens Documentation General Axis parameters Variables name 1.0 1.403 OK Cancel Help Part Num: LXM05AD10??? Software version: 1.0 **Note:** You are advised to check for consistency between the version of the servodrive's firmware and the version declared in Control Expert. The version is used to define the drive parameters. During the servodrive init by the `CAN_HANDLER` MFB function block, the vesrsion is tested.

Step	Action
5	In the **Variables Name** tab, enter: - a name for the `Axis_Ref` type variable linked to the servodrive - a name for the `Can_Handler` type variable linked to the servodrive In this tab, the **Axis_Z** is configured as follows:  Axis reference variable name: Axis_Ref_Z CANopen handler variable name: Can_Handler_Z Project Browser Structurial view Station Configuration Derived Data Types Derived FB Types Variables & FB instances Motion Communication Program Animation Tables Operator Screens Documentation General Axis parameters Variables name OK Cancel Help
6	Click on **OK** to confirm the selections.

<table>
<tr><th>Step</th><th>Action</th></tr>
<tr><td>7</td><td>Right-click on the <b>Recipe_0</b> sub-directory and then select <b>Properties</b> in the contextual menu. It is then possible to modify the recipe and parameter variables created by default when creating the axis.
<b>Notes :</b> Tick the Initial Values saving Enabled checkbox allows to include the recipe in the application. This functionality is available for M340 V2.0 or later firmware versions, see the recipe variable. <i>(see page 43)</i>
In this window, the variables for the <b>Axis_Z</b> are named by default as follows:



Recipe variable name: Recipe_0
Parameter description variable name: AxisParamDesc_0
□ Initial values saving Enabled</td></tr>
<tr><td>8</td><td>Click on <b>OK</b> to confirm the configuration.</td></tr>
</table>

NOTE: It is possible to create several recipes for the same axis (there is one by default). Loading of the required recipe, depending on the request, is performed by the TE_DOWNLOADDRIVEPARAMETER *(see EcoStruxure™ Control Expert, Motion Function Blocks, Block Library)* block.

This MFB library block is used to:

- load parameters to a new servodrive if the previous one is faulty.
- load a new recipe to a servodrive during a production change, for example

You can delete the recipe if you can not use it.

NOTE: The memory size of unlocated data for the management of a recipe by servodrive type is around 2 Kwords.

The objects Axis_Ref, Can_Handler, AxisParamDesc and Recipe

At a Glance

For each axis creation, 1 function block and 3 variables are created:

- A `Can_Handler`-type function block automaticaly created by motion browser, which can be renamed using the axis directory
- An `Axis_Ref`-type variable which can be renamed using the axis directory
- A byte table type variable (ARRAY[....] OF BYTE) named by default `Recipe_`*`x`* (where *x* is a value) but which can be renamed using the `Recipe_`*`x`* directory
- An unsigned integer table type variable (ARRAY[....] OF UINT) named `AxisParamDesc_`*`x`* (where *x* is a value) and which may not be renamed

`Can_Handler`

This variable is an EFB. It is named after the CANopen manager variable.

It is declared in the **Function Block** tab during Axis Creation *(see page 38)*.

It must be used in the program as the instance of the `CAN_HANDLER` *(see page 59)* function block.

`Axis_Ref`

This variable is an `AXIS_REF`-type structured variable named after the axis reference variable.

It is declared in the **Variables Name** tab during Axis Creation *(see page 38)*.

It must be specified in the input parameter for each MFB block used by the axis.

`AxisParamDesc`

This variable is an unsigned integer table type variable (ARRAY[....] OF UNIT). It is automatically created during Axis Creation *(see page 38)*. It is named after the parameter description variable which can be seen in the `Recipe_`*`x`* properties of the axis.

This variable must be specified in the TE_UPLOADDRIVEPARAMETER *(see EcoStruxure™ Control Expert, Motion Function Blocks, Block Library)* and TE_DOWNLOADDRIVEPARAMETER *(see EcoStruxure™ Control Expert, Motion Function Blocks, Block Library)* blocks' `PARAMETERLIST` input parameter taken from the MFB library and useful for recipe creation or for replacing the axis if it is faulty.

This variable:

- cannot be modified,
- is identical if the axes declared in the application have the same references and firmware version.

Recipe

This variable is a byte table type variable (ARRAY[....] OF BYTE). It is automatically created during Axis Creation *(see page 38)*. It is named after the recipe variable which can be seen in the `Recipe_x` properties of the axis.

This variable must be specified in the TE_UPLOADDRIVEPARAMETER *(see EcoStruxure™ Control Expert, Motion Function Blocks, Block Library)* or TE_DOWNLOADDRIVEPARAMETER *(see EcoStruxure™ Control Expert, Motion Function Blocks, Block Library)* block's `PARAMETERSET` input parameter taken from the MFB library and useful for recipe creation or for replacing the axis if it is faulty.

The variable name may be modified using the `Recipe_x` properties of the axis.

The recipe can be included in the application :

The application can be updated with a storage in the inital values either with 'update Init Values with Current values' command or using the %S94 bit. Consequently, the STU or XEF file will include the values got from the drive after a TE_Upload calling . Finally, tick the 'Initial Values saving Enabled' checkbox to make this functionality available.

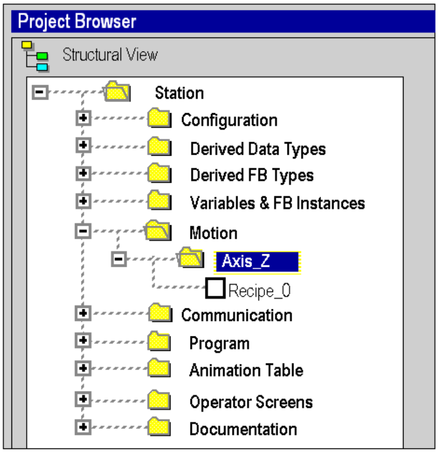
NOTE: By default, Initial Value saving Enabled checkbox is not ticked.

NOTE: Initial Values saving Enabled functionality is available for M340 V2.0 or later firmware versions.

Motion Directory Configuration Result

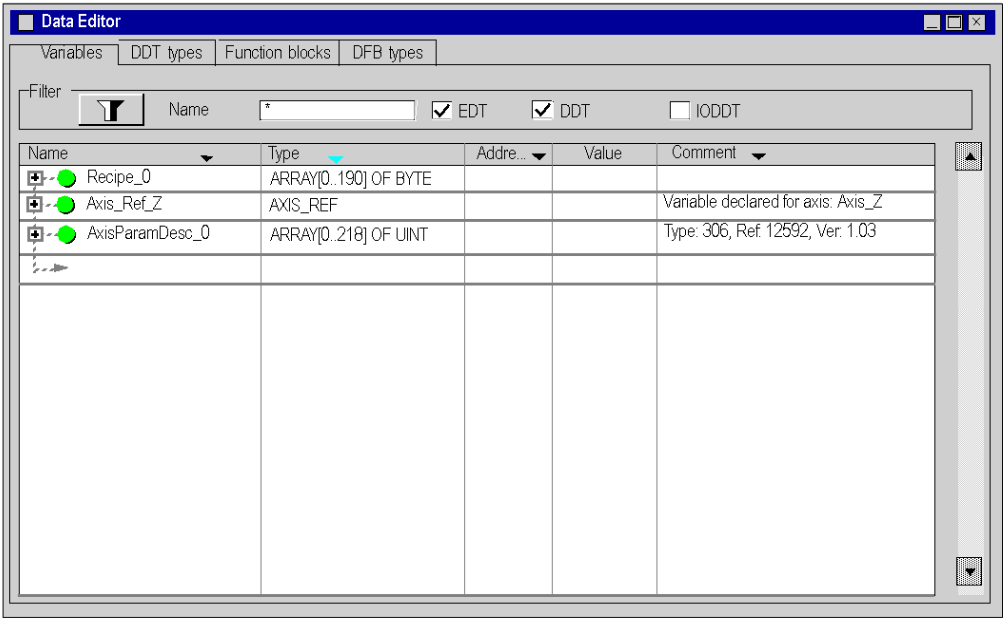
In the Project Browser

The following diagram shows the tree structure for the **Motion** directory after configuration:



In the Data Editor

The following screen shows the variables that are created in the data editor during the creation of the axes. To access this screen, double-click on the **Variables & FB instances** directory in the project browser:



Name: *   ☑ EDT   ☑ DDT   □ IODDT

Name	Type	Addre...	Value	Comment
Recipe_0	ARRAY[0..190] OF BYTE			
Axis_Ref_Z	AXIS_REF			Variable declared for axis: Axis_Z
AxisParamDesc_0	ARRAY[0..218] OF UINT			Type: 306, Ref. 12592, Ver. 1.03

The variable `Can_Handler_Z` may be accessed by clicking on the **Function blocks** tab.

Section 2.5
Configuring the Lexium 05

Aim of this Section

This section describes the basic servodrive configurations using PowerSuite for **Lexium 05** and the servodrive's front panel user interface.

What Is in This Section?

This section contains the following topics:

Configuring the Lexium 05 in PowerSuite

Overview

With PowerSuite, users can define installed device bases, and describe their associated configurations and communication settings.

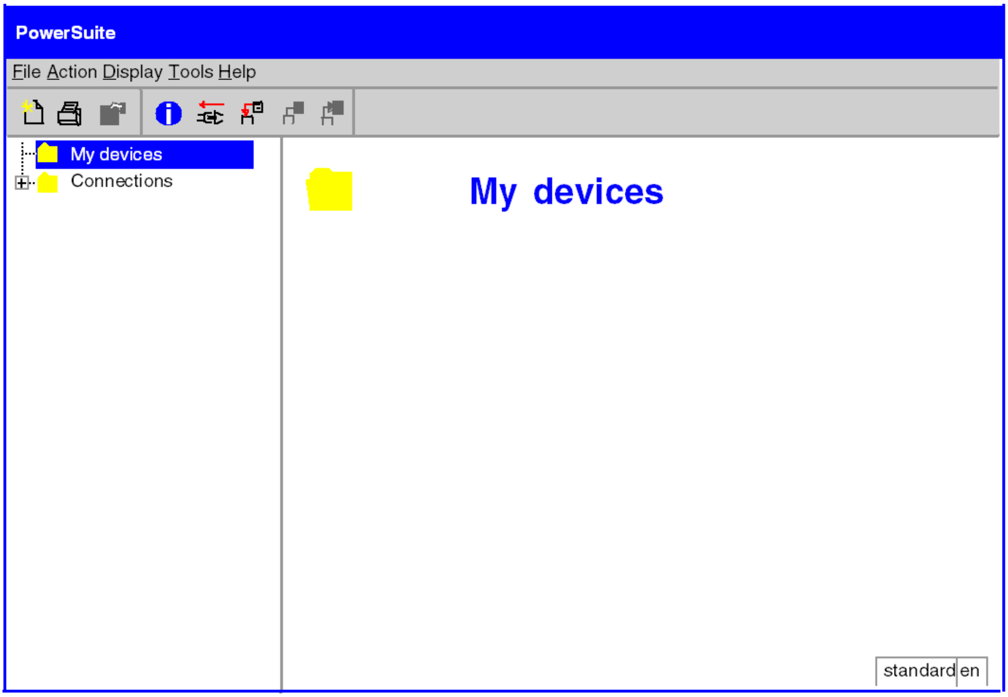
PowerSuite then gives access to a group of actions for editing or transferring the configurations and for connecting to the devices.

PowerSuite's navigation principle associates a configuration interface with each device type, making it possible to control, tune and monitor them.

NOTE: The required signals, i.e LIMN, LIMP, REF must be wired or deactived by the tuning software.

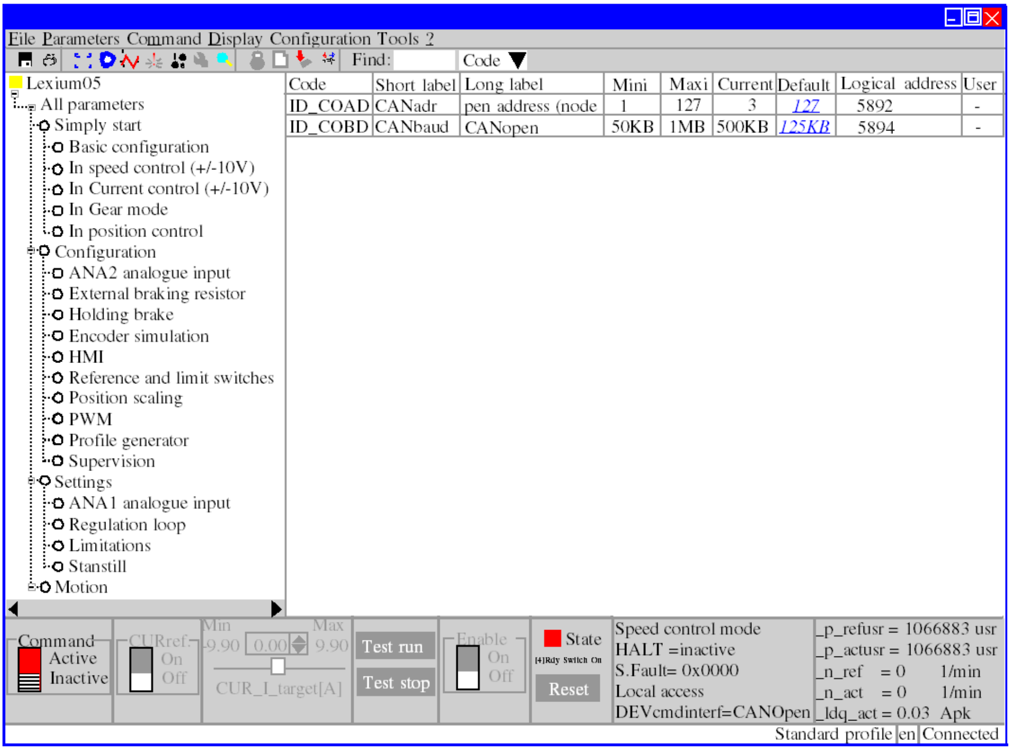
Connecting to the Lexium 05

This table describes the procedure for connecting to the **Lexium 05**:

Step	Action
1	Connect your PC, on which PowerSuite for **Lexium 05** is installed, to the **RJ45** connector on the servodrive to be configured.
2	Start PowerSuite for **Lexium 05**, **Result:** the following start-up screen is displayed:  PowerSuite File Action Display Tools Help My devices Connections My devices standard en
3	Choose **Action** and then **Connect**. **Result:** a text box is displayed.
4	Type a project name (Lexium05_MFB) and then click on **OK**. **Result:** a transfer confirmation window is displayed.
5	Press **Alt F** to start transferring data from the servodrive to the connected work station.

Basic Lexium 05 Configuration

This table describes the procedure for entering basic settings:

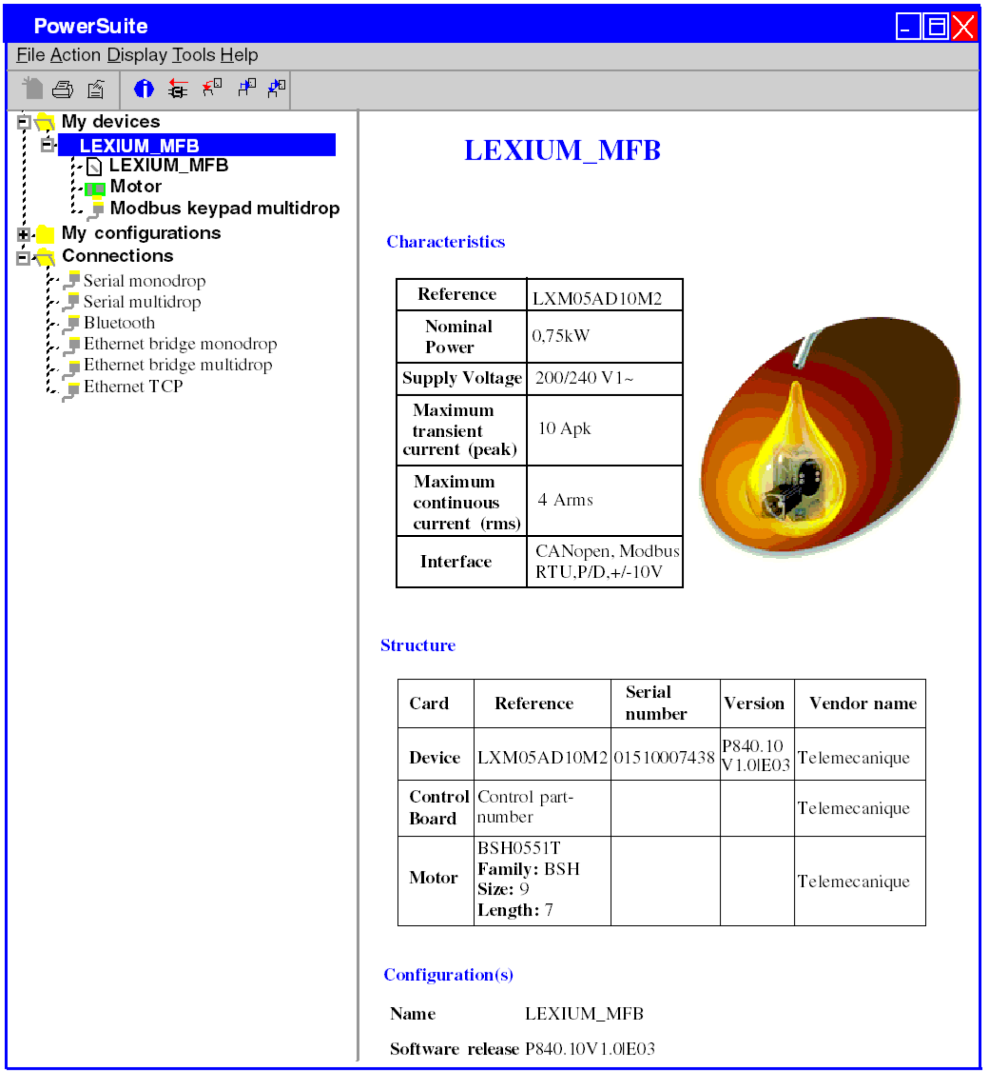
Step	Action
1	Following a connection and transfer of the device's configurations, PowerSuite displays a configuration screen in a new window that gives access to device control, tuning and monitoring functions. In the tree structure displayed, choose **CANopen** in the *Communication* directory. **Result:** the following window is displayed: 
2	Double-click on the value in the **ID_COAD** line, **Current Value** column, and type the **Lexium 05** CANopen address.
3	Double-click on the value in the **ID_COBD** line, **Current Value** column and choose the CANopen bus baud rate.
4	Save the CANopen settings to EEprom with the command **Configuration → Save to EEprom**. **Note:** it is possible to adjust the servodrive's settings with the same procedure.

Code	Short label	Long label	Mini	Maxi	Current	Default	Logical address	User
ID_COAD	CANadr	pen address (node	1	127	3	127	5892	-
ID_COBD	CANbaud	CANopen	50KB	1MB	500KB	125KB	5894	-

Speed control mode
HALT =inactive
S.Fault= 0x0000
Local access
DEVcmdinterf=CANOpen

_p_refusr = 1066883 usr
_p_actusr = 1066883 usr
_n_ref = 0 1/min
_n_act = 0 1/min
_Idq_act = 0.03 Apk

Standard profile | en | Connected

<table>
<tr><th>Step</th><th>Action</th></tr>
<tr><td>5</td><td>Once the settings have been adjusted, use the command <b>Configuration → Disconnect</b> to disconnect.
<b>Result:</b> the following screen is displayed, showing the data saved locally:


<h2>LEXIUM_MFB</h2>
<b>Characteristics</b>
<table>
<tr><td>Reference</td><td>LXM05AD10M2</td></tr>
<tr><td>Nominal Power</td><td>0,75kW</td></tr>
<tr><td>Supply Voltage</td><td>200/240 V1~</td></tr>
<tr><td>Maximum transient current (peak)</td><td>10 Apk</td></tr>
<tr><td>Maximum continuous current (rms)</td><td>4 Arms</td></tr>
<tr><td>Interface</td><td>CANopen, Modbus RTU,P/D,+/-10V</td></tr>
</table>
<b>Structure</b>
<table>
<tr><th>Card</th><th>Reference</th><th>Serial number</th><th>Version</th><th>Vendor name</th></tr>
<tr><td>Device</td><td>LXM05AD10M2</td><td>01510007438</td><td>P840.10 V1.0IE03</td><td>Telemecanique</td></tr>
<tr><td>Control Board</td><td>Control part-number</td><td></td><td></td><td>Telemecanique</td></tr>
<tr><td>Motor</td><td>BSH0551T
<b>Family:</b> BSH
<b>Size:</b> 9
<b>Length:</b> 7</td><td></td><td></td><td>Telemecanique</td></tr>
</table>
<b>Configuration(s)</b>
<table>
<tr><td>Name</td><td>LEXIUM_MFB</td></tr>
<tr><td>Software release</td><td>P840.10V1.0IE03</td></tr>
</table>
</td></tr>
<tr><td>6</td><td>The Lexium 05 must be turned off and then turned back on in order to apply the new settings.</td></tr>
</table>

Configuring the Lexium 05 with the User Interface

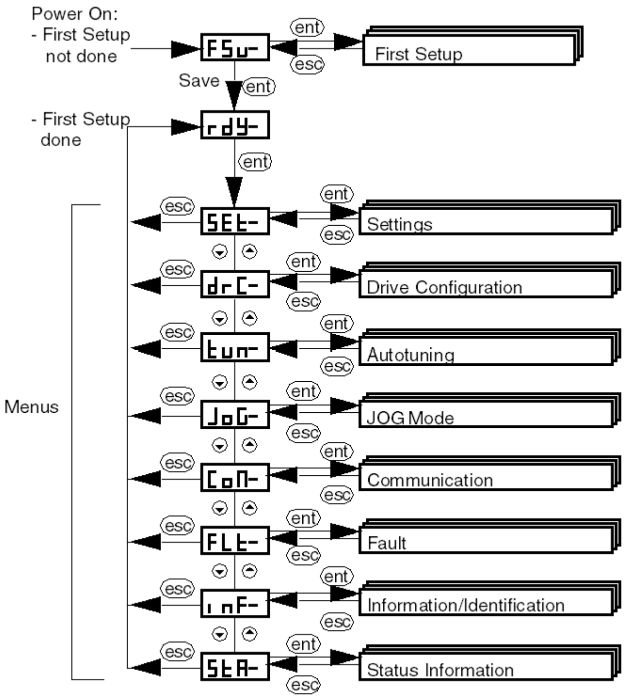
Overview

A user interface is integrated in the **Lexium 05**. With this interface, you can:

- put the device online
- configure the device
- carry out a diagnostic

Interface Menu Structure

The following graphic presents an overview of access to the interface’s main menus:



Basic Settings

The following table describes the procedure for entering basic settings (CANopen address and speed) with the interface.

Step	Action
1	Press the **ENT** button on the interface. **Result:** the **SET** (Setting) menu is displayed on the interface's status indicator.
2	Press the ▼ button several times to access the **COM** menu. **Result:** the **COM** (Communication) menu is displayed on the interface's status indicator.
3	Press the **ENT** button on the interface. **Result:** the **COAD** (CANopen Address) submenu is displayed on the interface's status indicator.
4	Press **ENT** again. **Result:** a value corresponding to the device's CANopen address is displayed.
5	Press the ▼ button to decrease, or the ▲ button to increase the CANopen address value. Press **ENT** when the desired CANopen address is displayed (3). **Result:** the value is confirmed and the **COAD** (CANopen Address) submenu is displayed again.
6	Press **ESC** once to return to the **COAD** submenu.
7	Press the ▼ button to access the **COBD** (CANopen Baud) submenu. Press **ENT**. **Result:** a value corresponding to the device's CANopen speed is displayed.
8	Press the ▲ button to decrease, or the ▼ button to increase the CANopen baud rate value. Press **ENT** when the desired CANopen speed is displayed (500). **Result:** the value is confirmed and the **COBD** (CANopen Baud) submenu is displayed again.
9	Press **ESC** several times to return to the main display (**RDY** by default).

Chapter 3
Application Programming

Subject of this Chapter

This chapter describes the various development phases of the application program.

What Is in This Chapter?

This chapter contains the following topics:

Declaration of Variables

At a Glance

In addition to the variables associated with the axis when it is created in the **Motion** directory, other variables must be declared.

They must be assigned to:

- Input or output parameters of the MFB blocs
- Operator Screen *(see page 77)* objects.

They allow you to use certain data and to control the axis with blocks from the MotionFunctionBlock library.

Declaration in the Data Editor

The table below summarizes the variables to be created in the data editor for the tutorial example:

Name	Type	Comment
`Cmd_Home_Z`	BOOL	Return axis to home position command
`Cmd_Mvt_Z`	BOOL	Move axis command
`Cmd_Run_Z`	BOOL	Run axis command
`Cmd_Stop_Z`	BOOL	Stop axis command
`Cmd_Reset_Z`	BOOL	Acknowledge axis command
`Cmd_Upload_Z`	BOOL	Save axis data in a recipe table command
`Cmd_Download_Z`	BOOL	Transfer data from recipe table to axis command
`Axis_OK_Z`	BOOL	Axis recognized by CANopen bus
`Position_Z`	DINT	Value of axis position
`Velocity_Z`	DINT	Value of axis speed
`Recipe_Z`	ARRAY[0..190] OF BYTE	Buffer variable for management of recipes
`CAN`	T_COM_CO_BMX	IODDT that manages CANOpen port

NOTE: the size of the recipe management table complies with that of the recipes created by the **Motion** directory.

Programming the Example

At a Glance

Just after declaration and parameter setting of the hardware, motion programming is the second development phase of the tutorial example.

Axis programming is divided up into:

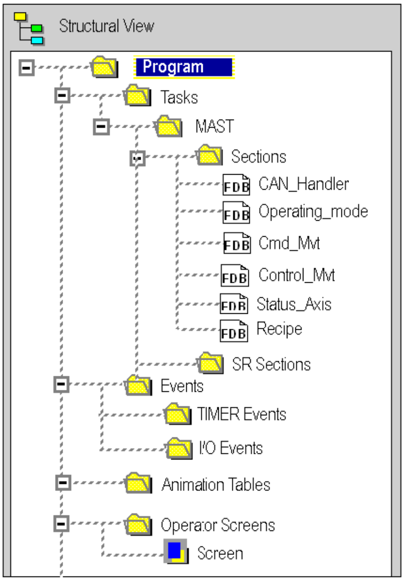
- declaration of variables
- an operator screen which is used to view and control the axis
- structured programming in several sections

Declaring the Sections

The table below presents a summary of the program sections to create

Section name	Language	Description
CAN_Handler *(see page 59)*	FBD	This section allows you to check that the parameters of the axis correspond to reality.
Operating_mode *(see page 62)*	FBD	This section allows you to power up the servodrives and to check the axes.
Cmd_Mvt *(see page 63)*	FBD	This section allows you to set a homing reference point for the axis and to then control it in absolute motion.
Control_Mvt *(see page 65)*	FBD	This section is used to determine the position and speed of the axis.
Status_Axes *(see page 66)*	FBD	This section is used to determine the status of the axis and to carry out diagnostics for an event.
Recipe *(see page 68)*	FBD	This section allows you to save or restore a servodrive's data.

The diagram below shows the program structure after the programming sections have been created:



The CAN_HANDLER Function Block

At a Glance

The use of the CAN_HANDLER *(see EcoStruxure™ Control Expert, Motion Function Blocks, Block Library)* **MFB** function block is **essential** and **mandatory** in the programming of the axis. The program section with this **MFB** function block must be associated with the same task of the CANopen bus master *(see page 29)*.

It allows you to check:

- the CANopen communication
- consistency between the software configuration and the connected physical device.

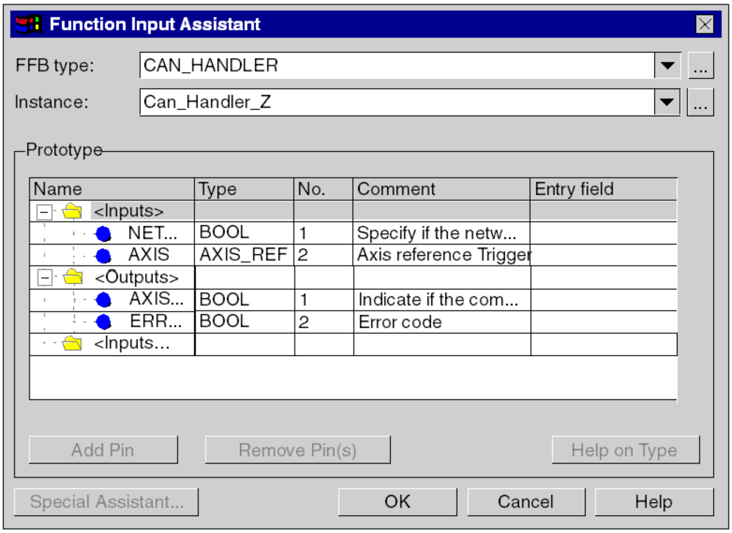
This block uses the two variables that belong to the axis' directory. The `Can_Handler_Z` variable must be used as instance and the `Axis_Ref_Z` variable must be assigned to the block's `AXIS` input parameter.

Inserting and Instantiating a Block

This table describes the procedure for inserting and select the instance of a block in a program section:

Step	Action
1	Right click in an empty field in the FBD section to display the contextual menu.
2	Execute the **FFB Input Assistant..** command in the contextual menu. **Result**: The Function Input Assistant opens.
3	Click on the **...** icon on the **FFB Type** line. **Result**: the **FFB Type Selection** window opens.
4	Expand **Libraries** → **MotionFunctionBlock** and click on **MFB**. **Result**: all of the blocks from the **MotionFunctionBlock** library are displayed on the right-hand side of the **FFB Type Selection** window.
5	Select the `CAN_HANDLER` block and confirm your choice by clicking on **OK**. **Result**: The **FFB Input Assistant..** window is displayed, set up by the `CAN_HANDLER` block.
6	Click on the **...** icon on the **Instance** line. **Result**: the **FB Instance Selection** window opens.

<table>
<tr><th>Step</th><th>Action</th></tr>
<tr><td>7</td><td>Select the `Can_Handler_Z` instance and confirm your choice by clicking on <b>OK</b>.
<b>Result</b>: The `Can_Handler_Z` variable is displayed in the <b>Instance</b> field:



<b>Function Input Assistant</b>

FFB type: CAN_HANDLER

Instance: Can_Handler_Z

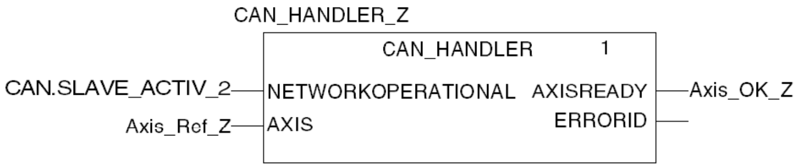
Prototype

<table>
<tr><th>Name</th><th>Type</th><th>No.</th><th>Comment</th><th>Entry field</th></tr>
<tr><td>&lt;Inputs&gt;</td><td></td><td></td><td></td><td></td></tr>
<tr><td>NET...</td><td>BOOL</td><td>1</td><td>Specify if the netw...</td><td></td></tr>
<tr><td>AXIS</td><td>AXIS_REF</td><td>2</td><td>Axis reference Trigger</td><td></td></tr>
<tr><td>&lt;Outputs&gt;</td><td></td><td></td><td></td><td></td></tr>
<tr><td>AXIS...</td><td>BOOL</td><td>1</td><td>Indicate if the com...</td><td></td></tr>
<tr><td>ERR...</td><td>BOOL</td><td>2</td><td>Error code</td><td></td></tr>
<tr><td>&lt;Inputs...</td><td></td><td></td><td></td><td></td></tr>
</table>

</td></tr>
<tr><td>8</td><td>Confirm the block configuration by clicking on <b>OK</b>.
<b>Result</b>: the FBD section is displayed again. A symbol is added to the mouse cursor.</td></tr>
<tr><td>9</td><td>Click on an empty field in the FBD section.
<b>Result</b>: the `CAN_HANDLER` block, instantiated by the `Can_Handler_Z` variable is inserted in the FBD section.</td></tr>
<tr><td>10</td><td>Specify the input and output parameters as defined in the contents.</td></tr>
</table>

Contents

The screen below shows the section result:



CAN.SLAVE_ACTIV_2 corresponds to the active slave bit created by the IODDT T_COM_CO_BMX.

The input parameter `NETWORKOPERATIONAL` must be assigned to a bit that validates the correct operation of the CANopen network.

The assignment of this parameter left to the discretion of the developer. It depends on the philosophy of the process and the way the bus is managed.

For example, this parameter may be connected to an object or to a T_COM_CO_BMX *(see Modicon M340, CANopen, User Manual)*-type IODDT equation.

Management of the Axis’ Operating and Stop Modes

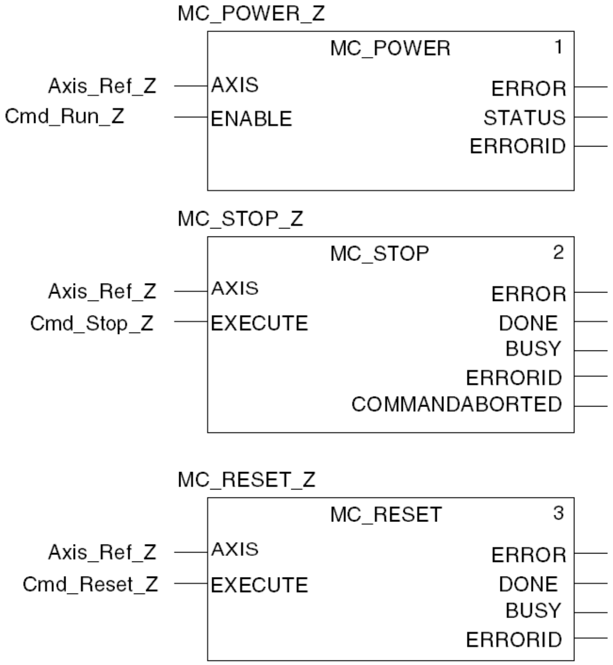
At a Glance

This section is made up of the following MFB blocks:

- MC_POWER *(see EcoStruxure™ Control Expert, Motion Function Blocks, Block Library)*, which is used to disable or enable the servodrives
- MC_STOP *(see EcoStruxure™ Control Expert, Motion Function Blocks, Block Library)*, which is used to stop any movement in progress
- MC_RESET *(see EcoStruxure™ Control Expert, Motion Function Blocks, Block Library)*, which is used to initialize the function blocks and to acknowledge servodrive faults.

Contents

The screen below shows a part of the section to develop:



The blocks are instantiated to variables input directly in the **Instance** zone of the **FFB Input Assistant** to facilitate subsequent diagnostics using the animation tables.

Motion Control

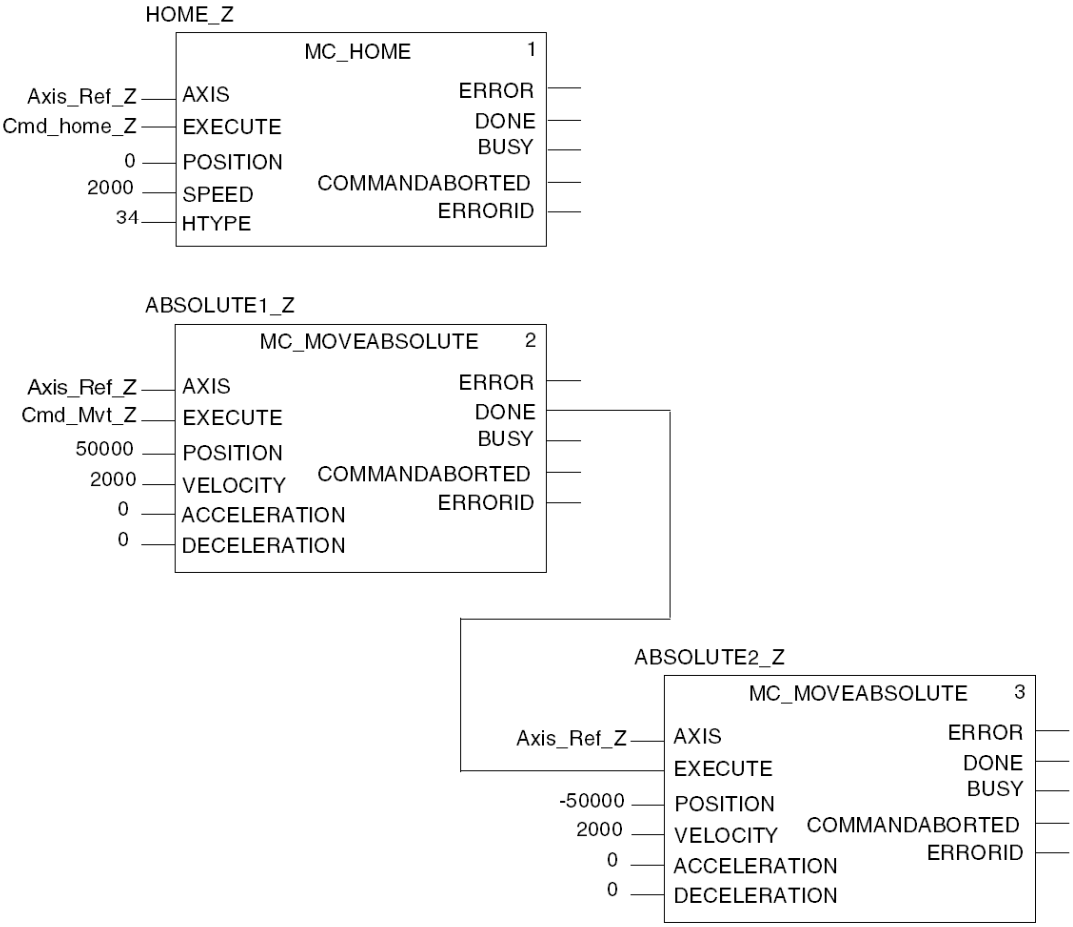
At a Glance

This programming section is made up of the following MFB blocks:

- MC_HOME *(see EcoStruxure™ Control Expert, Motion Function Blocks, Block Library)*, which allows a homing reference point to be set for the axis before then launching it in absolute motion
- MC_MOVEABSOLUTE *(see EcoStruxure™ Control Expert, Motion Function Blocks, Block Library)*, which allows the axis to make an absolute movement.

Contents

The screen below shows the part of the section:



For the tutorial example, the section is made up of a type of sequence of reversing movements.

The outward motion is conditioned by the `Cmd_Mvt_Z` bit from the operator screen *(see page 77)*.

The return motion is conditioned by the end of the outward motion.

The position unit is USR and the velocity unit is rpm.

The Homing type `HTYPE` value (34) corresponds to an homing within a single turn, positive direction of rotation.

Motion Monitoring

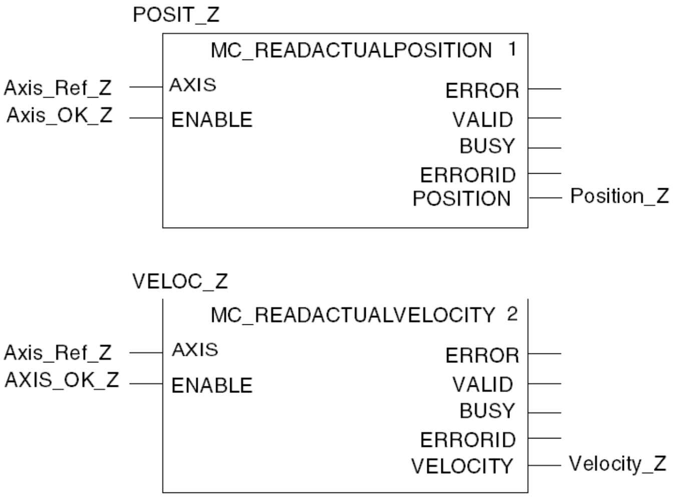
At a Glance

This section is made up of the MC_READACTUALPOSITION *(see EcoStruxure™ Control Expert, Motion Function Blocks, Block Library)* and MC_READACTUALVELOCITY *(see EcoStruxure™ Control Expert, Motion Function Blocks, Block Library)* MFB blocks.

These blocks are used to determine the exact position and speed of the axis.

Contents

The screen below shows a part of the section to develop:



Whilst the `Axis_OK_Z` bit is enabled, the position and speed values are continuously displayed on the operator screen *(see page 77)*.

Status and Axis Error Code Section

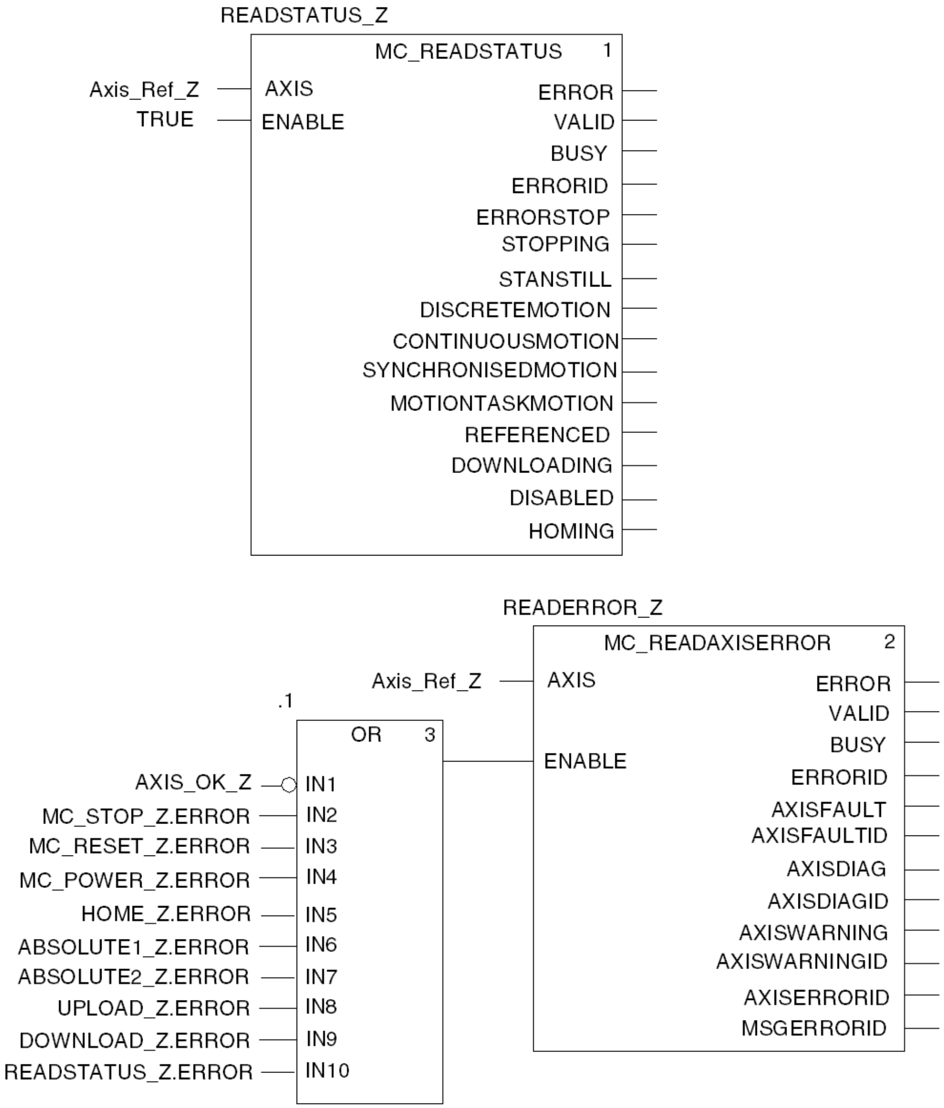
At a Glance

This section is made up of the following MFB blocks:

- MC_READSTATUS *(see EcoStruxure™ Control Expert, Motion Function Blocks, Block Library)*, which is used to determine the drive status *(see EcoStruxure™ Control Expert, Motion Function Blocks, Block Library)*
- MC_READAXISERROR *(see EcoStruxure™ Control Expert, Motion Function Blocks, Block Library)*, which is used to determine the error values according to the type of errors on the drive and to deduce their causes *(see EcoStruxure™ Control Expert, Motion Function Blocks, Block Library)*.

Contents

The screen below shows a part of the section:



The `UPLOAD_Z.ERROR` and `DOWNLOAD_Z.ERROR` variables must be added to the `OR` block after the recipe *(see page 68)* section has been created.

Backup and Transfer of the Servodrive Parameters

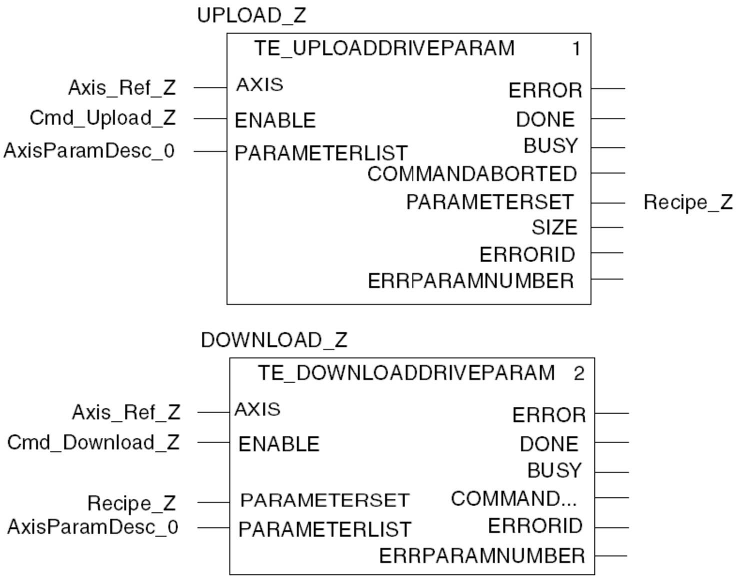
At a Glance

This programming section is made up of the following MFB blocks:

- TE_UPLOADDRIVEPARAM *(see EcoStruxure™ Control Expert, Motion Function Blocks, Block Library)*, which is used to back up the configuration of a servodrive in a data table
- TE_DOWNLOADDRIVEPARAM *(see EcoStruxure™ Control Expert, Motion Function Blocks, Block Library)*, which is used to transfer the data table parameters to a servodrive.

Contents

The screen below shows the Recipe section:



If `Cmd_Upload_Z` is enabled, the servodrive configuration is saved in the data table `Recipe_Z` (buffer variable for the recipes).

If `Cmd_Download_Z` is enabled, the servodrive configuration is restored by the data table `Recipe_Z`.

Transferring the Project between the Terminal and the PLC

At a Glance

Transferring a project allows you to copy the current project from the terminal to the current PLC’s memory (PLC that has its address selected).

Project Analysis and Generation

To perform analysis and generation of a project at the same time, carry out the following actions:

Step	Action
1	Activate the **Rebuild All Project** command in the **Build** menu. **Result:** the project is analyzed and generated by the software.
2	Any errors detected are displayed in the information window at the bottom of your screen.

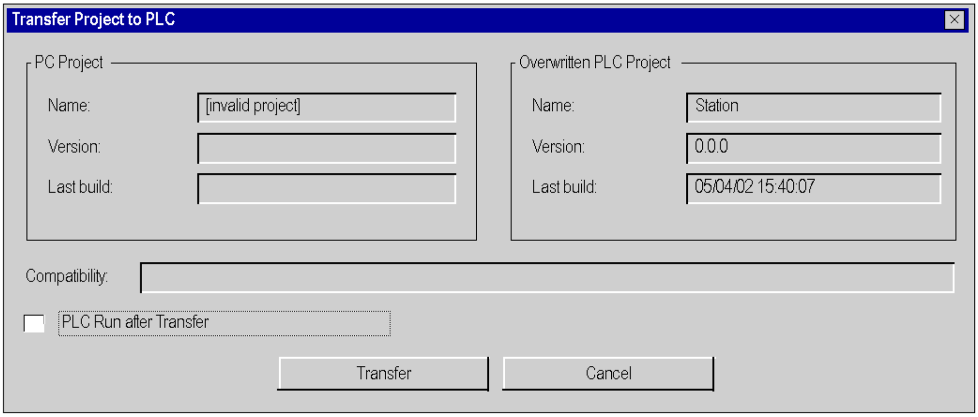
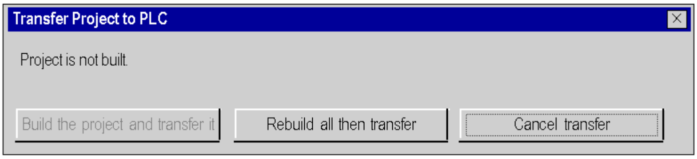
Project Backup

To back up the project, carry out the following actions:

Step	Action
1	Activate the **Save As** command in the **File** menu.
2	If necessary, select the directory to which the project will be saved (disk and path).
3	Enter the file name: **MFB_Lexium05**.
4	Confirm with **Save**. **Result:** the project is saved as **MFB_Lexium05.STU**.

Transferring the Project to the PLC

You must carry out the following actions to transfer the current project to a PLC:

Step	Action
1	Use the **PLC → Define the address** command. Enter **SYS** if you are using a **USB** media that is directly connected from the PC (terminal) to the PLC.
2	Switch to online mode using the **PLC → Connection** command.
3	Activate the **PLC → Transfer Project to PLC** command. **Result**: the screen used to transfer the project between the terminal and the PLC is displayed:  Transfer Project to PLC PC Project Name: [invalid project] Version: Last build: Overwritten PLC Project Name: Station Version: 0.0.0 Last build: 05/04/02 15:40:07 Compatibility: □ PLC Run after Transfer
4	Activate the **Transfer** command.
5	If the project has not been generated in advance, the screen below will be displayed allowing you to generate it before the transfer (**Rebuild All then Transfer**) or interrupt the transfer (**Cancel Transfer**).  Transfer Project to PLC Project is not built. Build the project and transfer it Rebuild all then transfer Cancel transfer
6	Transfer progress is displayed on screen. At any moment, you can interrupt the transfer by using the **Esc** key. In this case, the PLC project will be invalid. **Note**: In the event that the project is transferred to a Flash Eprom memory card, the transfer can take several minutes.

Chapter 4
Application Debugging

Subject of this Chapter

This chapter describes the possibilities for debugging the application using Control Expert and PowerSuite for **Lexium 05**.

What Is in This Chapter?

This chapter contains the following topics:

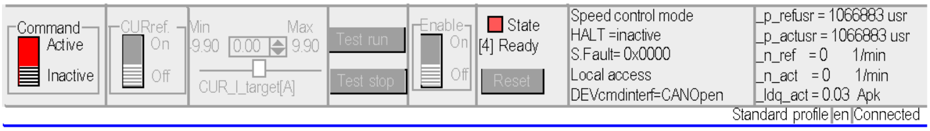
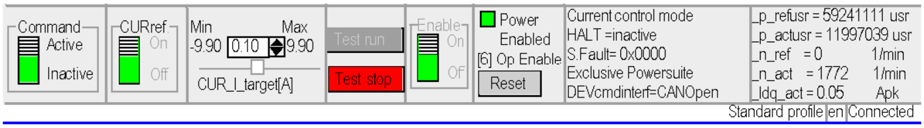
Tuning the Lexium 05 with PowerSuite

In Advance

We recommend tuning the axis kinematic before the program automatically starts it.

Tuning Example

The following table gives an example of kinematic tuning:

Step	Action
1	Connect *(see page 49)* to the **Lexium 05**.
2	After a connection and transfer of the device’s configurations, PowerSuite opens a new window with the configuration screen, which gives access to device control, tuning and monitoring functions. The following figure shows part of the new window. This lower window provides access to **Lexium 05** command functions:  Command Active Inactive CURref. On Off Min Max 9.90 0.00 9.90 CUR_I_target[A] Test run Test stop Enable On Off State Reset [4] Ready Speed control mode HALT =inactive S.Fault= 0x0000 Local access DEVcmdinterf=CANOpen _p_refusr = 1066883 usr _p_actusr = 1066883 usr _n_ref = 0 1/min _n_act = 0 1/min _Idq_act = 0.03 Apk Standard profile en Connected
3	Place the **Command** zone cursor on **Active**.
4	Place the **Enable** zone cursor on **On**.
5	Click the **Reset** button to clear any problems.
6	Click the **Test Run** button.
7	Enter the value 0,1 in the **CUR_I_target** zone.
8	Place the **CURref** zone cursor on **On**. **Result:** the motor runs and the sub-window is animated:  Command Active Inactive CURref On Off Min Max -9.90 0.10 9.90 CUR_I_target[A] Test run Test stop Enable On Off Reset Power Enabled [6] Op Enable Current control mode HALT =inactive S.Fault= 0x0000 Exclusive Powersuite DEVcmdinterf=CANOpen _p_refusr = 59241111 usr _p_actusr = 11997039 usr _n_ref = 0 1/min _n_act = 1772 1/min _Idq_act = 0.05 Apk Standard profile en Connected
9	Place the **Command** zone cursor on **Inactive** once tuning has been finalized.

Using Data via the Animation Tables

At a Glance

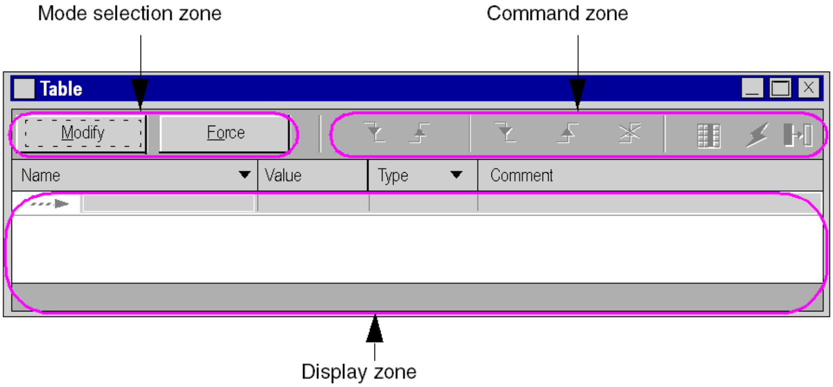
The animation table is the Control Expert' basic tool for viewing and forcing the status of variables.

NOTE: Control Expert also offers a graphic tool called **Operator Screens** which is designed to facilitate use of the application *(see page 77)*.

An animation table is divided into 3 areas that include:

- the **Mode** area
- the **Command** area
- the **Display** area

Animation table:



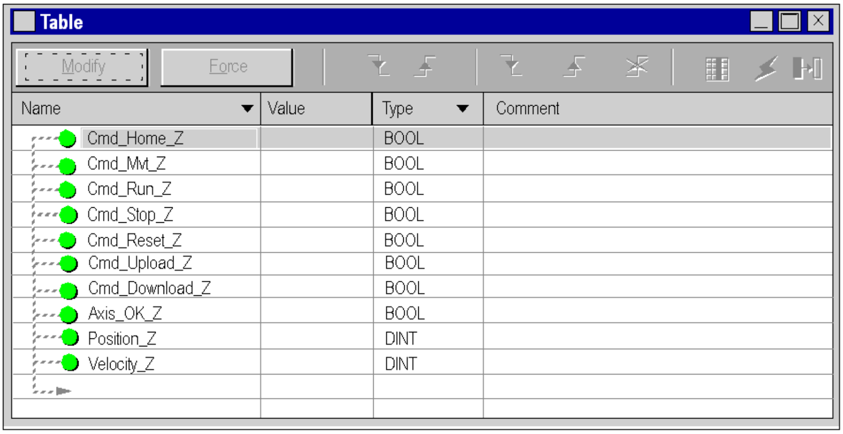
Creating an Animation Table

The table below presents the procedure for creating an animation table:

Step	Action
1	Right-click on the **Animation Tables** directory in the project browser. **Result**: the contextual menu is displayed.
2	Select **New Animation Table**. **Result**: a table properties window is displayed.
3	Click on OK to create the table, which is given a default name. **Result**: the animation table is displayed.

Adding Data to the Animation Table

The table below presents the procedure for creating data to view or force in the animation table:

Step	Action
1	In the **Table** window, click on the empty line in the **Name** column.
2	There are two possible ways of adding data: ● Enter the variable directly ● Click on the ⋯ icon to display the instance selection window in order to select the variable
3	Enter or select the respective variables. ● `Cmd_Home_Z` to issue an return axis to home position command ● `Cmd_Mvt_Z` to issue a move axis command ● `Cmd_Run_Z` to issue a run axis command ● `Cmd_Stop_Z` to issue a stop axis command ● `Cmd_Reset_Z` to issue an axis acknowledgement command ● `Cmd_Upload_Z` to issue a save axis data to a recipe table command ● `Cmd_Download_Z` to issue a transfer data from the recipe table to the axis command ● `Axis_OK_Z` to view the axis recognized by the CANopen bus ● `Position_Z` to determine the value of the axis position ● `Velocity_Z` to determine the value of the axis speed **Result**: the animation table looks like this. 

Table

Name	Value	Type	Comment
Cmd_Home_Z		BOOL	
Cmd_Mvt_Z		BOOL	
Cmd_Run_Z		BOOL	
Cmd_Stop_Z		BOOL	
Cmd_Reset_Z		BOOL	
Cmd_Upload_Z		BOOL	
Cmd_Download_Z		BOOL	
Axis_OK_Z		BOOL	
Position_Z		DINT	
Velocity_Z		DINT	

Program Debugging

At a Glance

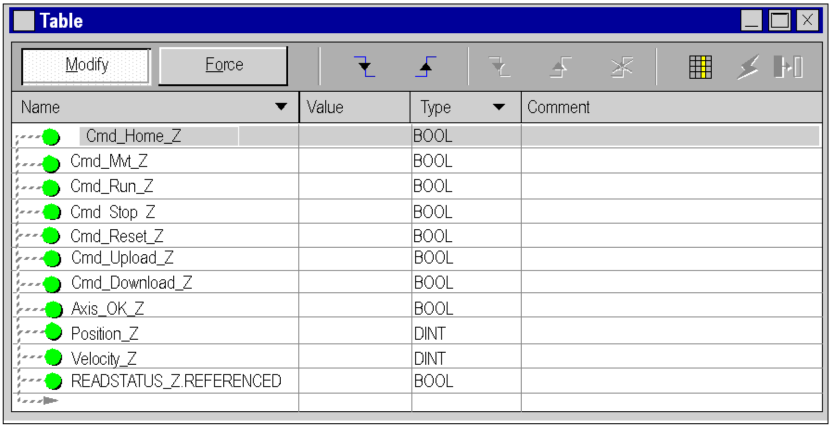
After transferring the program and running the axis using Powersuite for **Lexium 05**, the process is commissioned.

An animation table is a commissioning solution used to monitor, modify and/or force the values of variables.

The sets of parameters of the axis may be accessed and modified in Control Expert using the MFB messaging blocks MC_READPARAMETER *(see EcoStruxure™ Control Expert, Motion Function Blocks, Block Library)* and MC_WRITEPARAMETER *(see EcoStruxure™ Control Expert, Motion Function Blocks, Block Library)*.

Modification Mode

The following screen shows the animation table in modification mode:



Name	Value	Type	Comment
Cmd_Home_Z		BOOL	
Cmd_Mvt_Z		BOOL	
Cmd_Run_Z		BOOL	
Cmd_Stop_Z		BOOL	
Cmd_Reset_Z		BOOL	
Cmd_Upload_Z		BOOL	
Cmd_Download_Z		BOOL	
Axis_OK_Z		BOOL	
Position_Z		DINT	
Velocity_Z		DINT	
READSTATUS_Z.REFERENCED		BOOL	

This table is used to determine the status of the `MC_POWER` block’s input and output parameters.

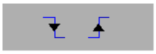
To access this mode, click on the **Modify** button in the mode selection zone.

NOTE: this operation may be assigned to other function blocks.

NOTE: the animation table is dynamic only in online mode (display of variable values).

Modifying Values

The tutorial example uses Boolean variables. To modify a Boolean value, carry out the following actions:

Step	Action
1	Use the mouse to select the Boolean variable you wish to modify.
2	Click on the  button corresponding to the desired value, or execute the **Set to 0** or **Set to 1** commands in the contextual menu.

Starting the System

The following table describes the procedure for starting the system used in the example:

Step	Action
1	Set the variable `Cmd_Run_Z` to 1. **Result:** the variable `Axis_OK_Z` changes to 1.
2	Set the variable `Cmd_Reset_Z` to 1.
3	Set the variable `Cmd_Home_Z` to 1. **Result:** the axis is referenced.
4	To rotate the axis, set the variable `Cmd_Mvt_Z` to 1. **Result:** the axis starts to turn and the values of the variables `Position_Z` and `Velocity_Z` are no longer set to 0.
5	To stop the axis from rotating: ● set the variable `Cmd_Stop_Z` to 1 ● set the variable `Cmd_Mvt_Z` to 0 **Result :** the axis stops rotating.
6	To start to rotate the axis again and complete the movement: ● set the variable `Cmd_Stop_Z` to 0 ● set the variable `Cmd_Mvt_Z` to 1 **Result:** the axis starts to rotate again and completes its movement.

Using Data via the Operator Screens

At a Glance

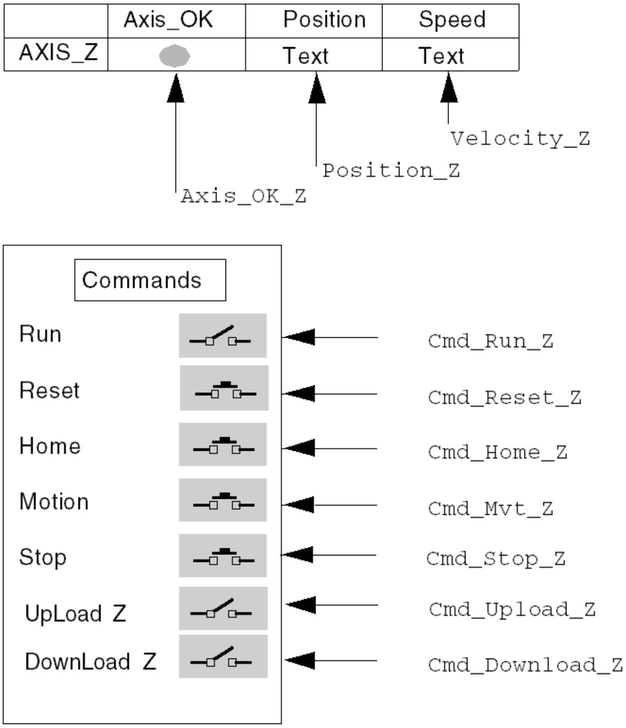
When a project is created without input cards, output cards or supervision, the Control Expert operator screen (associated with unlocated bits and words) allows to carry out initial debugging of the program.

In the tutorial example, the operator screen is used to:

- view data from the servodrives
- send commands to the servodrives

Representation

The representation below symbolizes the operating example which is used to control the axis and indicate the variables to be assigned to the objects (push button, LED and text):



Chapter 5
Operating the Application

Management of the Recipes

At a Glance

The TE_UPLOADDRIVEPARAM *(see EcoStruxure™ Control Expert, Motion Function Blocks, Block Library)* and TE_DOWNLOADDRIVEPARAM *(see EcoStruxure™ Control Expert, Motion Function Blocks, Block Library)* blocks are used to manage the production recipes.

An example of the procedure for creating and managing recipes is described in this section.

NOTE: for flexible machines, it is possible to manage several parameter recipes.

Creating and backing up the recipes

The table describes the procedure for creating recipes:

Step	Action
1	Create the recipes *(see page 38)* using the **Axis_Z** directory. **Result:** new recipe variables (`Recipe_0`, `Recipe_1`, etc.) are automatically created in the Data Editor *(see page 46)*.
2	Create a variable corresponding to the type of recipe variables. This variable is named in the `Recipe_Z` tutorial example. `Recipe_Z` acts as a buffer when backing up or transferring data. **Note**: it is essential to check **Allow dynamic arrays [ANY_ARRAY_XXX]** located in **Tools → Project options → Tab: Language extensions → Zone: Data type** to be able to use table type variables such as the recipes.
3	Configure the servodrive's parameters using Powersuite *(see page 48)*. These initial settings are used to configure a recipe.
4	Perform a backup of the parameters using the TE_UPLOADDRIVEPARAM *(see EcoStruxure™ Control Expert, Motion Function Blocks, Block Library)* block in the buffer variable `Recipe_Z`. The backup was successful if the bits of the MC_READSTATUS *(see EcoStruxure™ Control Expert, Motion Function Blocks, Block Library)* block are as follows: ● DOWNLOADING *(see EcoStruxure™ Control Expert, Motion Function Blocks, Block Library)* is set to 0 ● STANDSTILL *(see EcoStruxure™ Control Expert, Motion Function Blocks, Block Library)* is set to 1

Step	Action
5	Transfer the data backed up in the `Recipe_Z` buffer variable to the `Recipe_0` variable.

6	Repeat steps 3 and 4 to transfer the data backed up in the `Recipe_Z` buffer variable to the `Recipe_1` variable.
	The following programming presents a data transfer example based on the value of `PRODUCTION`:
	`IF UPLOAD_Z.DONE AND PRODUCTION=0 THEN`
	`Recipe_0:=Recipe_Z;`
	`END_IF;`
	`IF UPLOAD_Z.DONE AND PRODUCTION=1 THEN`
	`Recipe_1:=Recipe_Z;`
	`END_IF;`

Transfer Data from the Recipes

The table describes the procedure to transfer recipe data to the servodrive (for a production change, for example):

Step	Action

1	Reload the `Recipe_Z` buffer variable based on the value of `PRODUCTION` (type of production requested).
	`IF Cmd_Download_Z AND PRODUCTION=0 THEN`
	`Recipe_Z:=Recipe_0;`
	`END_IF;`
	`IF Cmd_Download_Z AND PRODUCTION=1 THEN`
	`Recipe_Z:=Recipe_1;`
	`END_IF;`

2	Transfer the parameter data, using the `Recipe_Z`buffer variable's TE_DOWNLOADDRIVEPARAM *(see EcoStruxure™ Control Expert, Motion Function Blocks, Block Library)* block, to the servodrive.
3	The transfer was successful if the bits of the MC_READSTATUS *(see EcoStruxure™ Control Expert, Motion Function Blocks, Block Library)* block are as follows: ● DOWNLOADING *(see EcoStruxure™ Control Expert, Motion Function Blocks, Block Library)* is set to 0 ● STANDSTILL *(see EcoStruxure™ Control Expert, Motion Function Blocks, Block Library)* is set to 1

Chapter 6
Application Maintenance

Subject of this Chapter

This chapter describes the procedure involved in replacing a servodrive after a fault has been diagnosed.

What Is in This Chapter?

This chapter contains the following topics:

Error Example

At a Glance

The MC_ReadAxisError function is used to recover system errors.

If an error or warning occurs, the block specifies a code by applying a value to the AXISFAULTID, AXISDIAGID and AXISWARNINGID output parameters.

Error Codes

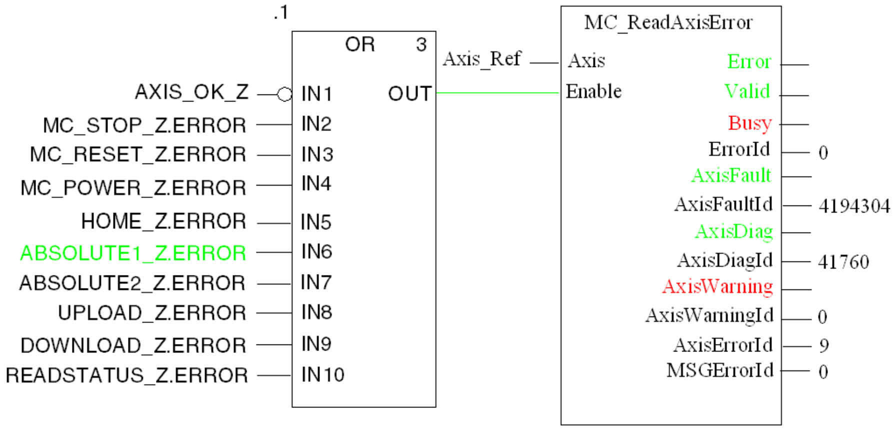
The following table shows the **Lexium 05** error codes:

	Lexium 05
AxisFaultId	SigLatched 301C:08
AxisDiagId	WarnLatched 301C:0C
AxisWarningId	StopFault 603F:0

NOTE: refer to the CANopen documentation for **Lexium 05** to identify the error.

Finding Errors

The table below describes a procedure for finding faults following an error or warning code.

Step	Action
1	The AxisFault output parameter equals 1. The AxisFaultId output parameter displays an error value. The graph below shows the error generated:  .1 OR 3 AXIS_OK_Z MC_STOP_Z.ERROR MC_RESET_Z.ERROR MC_POWER_Z.ERROR HOME_Z.ERROR ABSOLUTE1_Z.ERROR ABSOLUTE2_Z.ERROR UPLOAD_Z.ERROR DOWNLOAD_Z.ERROR READSTATUS_Z.ERROR IN1 IN2 IN3 IN4 IN5 IN6 IN7 IN8 IN9 IN10 OUT Axis_Ref MC_ReadAxisError Axis Enable Error Valid Busy ErrorId 0 AxisFault AxisFaultId 4194304 AxisDiag AxisDiagId 41760 AxisWarning AxisWarningId 0 AxisErrorId 9 MSGErrorId 0
2	Refer to the CANopen documentation of the **Lexium 05** and look for the code SigLatched 301C:08.
3	The AxisFaultID value is set to 4194304. This binary value means that bit 22 is set to one. Refer to the CANopen documentation of the **Lexium 05** and look for the code ‘SigLatched’ 301C:08. Bit 22 for ‘SigLatched’ designates a lag error.
4	Reduce the speed constants in absolute block or external load or acceleration.
5	Execute the `MC_Reset` block.

Replacing a Faulty Servodrive

At a Glance

If the servodrive fails, it may be necessary to swap it for an identical servodrive (reference). To do this, you are advised to save the adjustment parameters to a data table using the TE_UPLOAD-DRIVEPARAMETER *(see EcoStruxure™ Control Expert, Motion Function Blocks, Block Library)* block.

The TE_DOWNLOADDRIVEPARAM *(see page 68)* block then allows you to restore the saved data to a new servodrive.

Data Backup

The table below describes the procedure used to back up the servodrive's data using the TE_UPLOADDRIVEPARAMETER *(see EcoStruxure™ Control Expert, Motion Function Blocks, Block Library)* block:

Step	Action
1	Disable the Enable parameter, which belongs to the MC_POWER *(see EcoStruxure™ Control Expert, Motion Function Blocks, Block Library)* block. **Result**: the servodrive switches to Disable *(see EcoStruxure™ Control Expert, Motion Function Blocks, Block Library)* mode.
2	Enable the input parameter `Execute`. **Result**: the servodrive switches to Downloading *(see EcoStruxure™ Control Expert, Motion Function Blocks, Block Library)* mode. The data table assigned to the output parameter `PARAMETERSET` is filled in. **Note:** Please back up data to a *.DAT* file using **PLC → Transfer PLC data to the file** if the PLC has no memory card.

Restoring Data

The table below describes the procedure used to restore the servodrive’s data using the TE_DOWNLOADDRIVEPARAM *(see page 68)* block:

Step	Action
1	Disable the Enable parameter, which belongs to the MC_POWER *(see EcoStruxure™ Control Expert, Motion Function Blocks, Block Library)* block. **Result**: the servodrive switches to Disable *(see EcoStruxure™ Control Expert, Motion Function Blocks, Block Library)* mode.
2	Change the servodrive. The new servodrive must have the same references as the faulty servodrive. **Note**: make sure you take all the necessary precautions when changing the servodrive.
3	Configure the new servodrive with the basic parameters *(see page 48)* (CANopen address, speed) or using the keypad on the front panel.
4	Enable the block’s input parameter `Execute`. **Result**: the servodrive switches to Downloading *(see EcoStruxure™ Control Expert, Motion Function Blocks, Block Library)* mode. The data table assigned to the input parameter `PARAMETERSET` loads the input `PARAMETERLIST` which corresponds to the servodrive parameter.

Part II
Multi-Axis Application

Aim of this Part

This part describes the other hardware available for the Motion Function Blocks offer with a Modicon M340 running Control Expert.

The **Lexium 05** servodrive was used in the previous part to carry out an example. This part begins with a presentation of the following servodrives in a full architecture:

- **Lexium 32**
- **Lexium 15**
- **ATV 31**
- **ATV 32**
- **ATV 71**
- **IclA**

Following this presentation, configuration of each of the servodrives is described, detailing differences with the **Lexium 05** so as to carry out the same example.

What Is in This Part?

This part contains the following chapters:

Chapter 7
Foreword

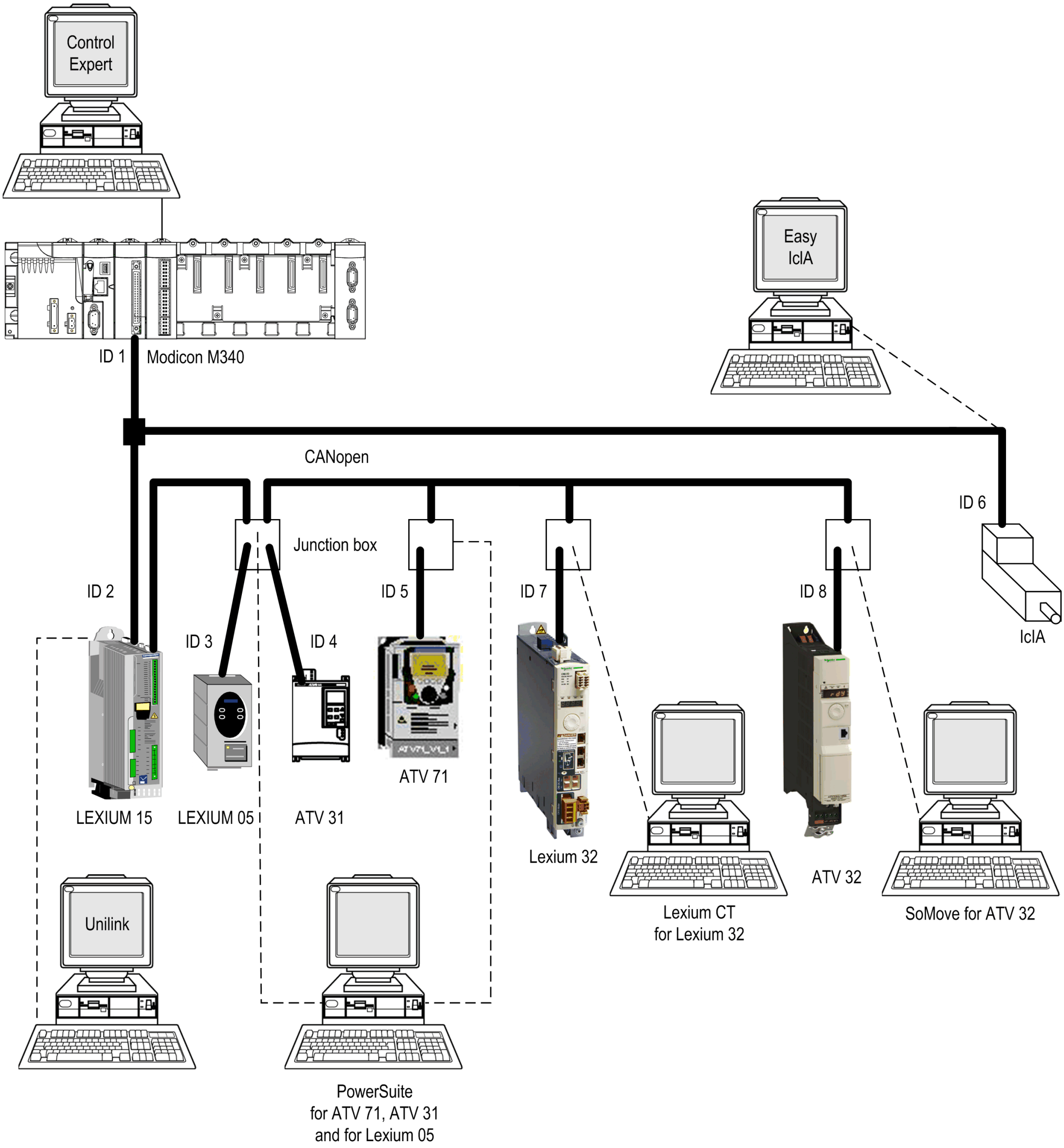
Application Architecture with All Servodrives

Overview

Following is a presentation of the usage of available hardware (servodrives), via an architecture, for implementing Motion Function Blocks in Control Expert.

Illustration

The following figure shows the architecture used in the application that includes all servodrives:



Chapter 8
Compatibility of Motion Applications with Control Expert Versions

Compatibility of XEF files

NOTE: Unity Pro is the former name of Control Expert for version 13.1 or earlier.

		Unity Pro/Control Expert Source version	
		V3.x/V4.0 M340 Proc < V2.0	≥ V4.0 M340 Proc ≥ V2.0
Unity Pro/Control Expert Target version	V3.x M340 < V2.0	Partially compatible in case of usage of Lexium15.	NC.
	≥ V4.0	PC.	FC.
NC : Not compatible. The motion parts are ignored during the import. PC : Partially compatible : the new axis type are ignored with an error message during the import : the application is imported by the sections using the drives that are in error. The new firmware version are downgraded to the highest available in the Unity Pro/Control Expert version with a warning during import, if the drives is present in the catalog for Mirano CPU. If not, the import is aborted. FC : Fully compatible.			

NOTE: 1. : The news EFB causes errors in the sections using them.

NOTE: 2. : Processor M340 ≥ V2.0: initial value saving enabled support.

Compatibility of STA files

NOTE: Unity Pro is the former name of Control Expert for version 13.1 or earlier.

		Unity Pro/Control Expert Source version		
		V3.x/V4.0 application without motion	V3.x/V4.0 with M340 < V2.0	≥ V4.0 with M340≥ V2.0
Unity Pro/Control Expert Target version	V3.x	FC	PC	NC
	≥ V4.0	FC	FC	FC
NC: Not compatible PC: Partially compatible: compatible only for applications with drive supported by the Unity Pro/Control Expert which is opening the application, in case of drives type or firmware versions evolutions. The application can be opened but can not be modified deeply. FC: Full compatible.				

Chapter 9
Lexium 32 Implementation for Motion Function Blocks

Aim of this Chapter

This chapter presents the implementation of Lexium 32 servodrives according to the methodology *(see page 17)* described in the quick start guide *(see page 11)* with a Lexium 05. It only details the differences and actions for Lexium 32.

What Is in This Chapter?

This chapter contains the following sections:

Section 9.1
Adapting the Application to the Lexium 32

Aim of this Section

This section presents adaptation of an application to the **Lexium 32** with an architecture, hardware and software requirements.

In this section Lexium 32 means else a Lexium 32 Advanced reference (LXM 32A...) else a Lexium 32 Modular reference (LXM 32 M...)

What Is in This Section?

This section contains the following topics:

Application Architecture with Lexium 32

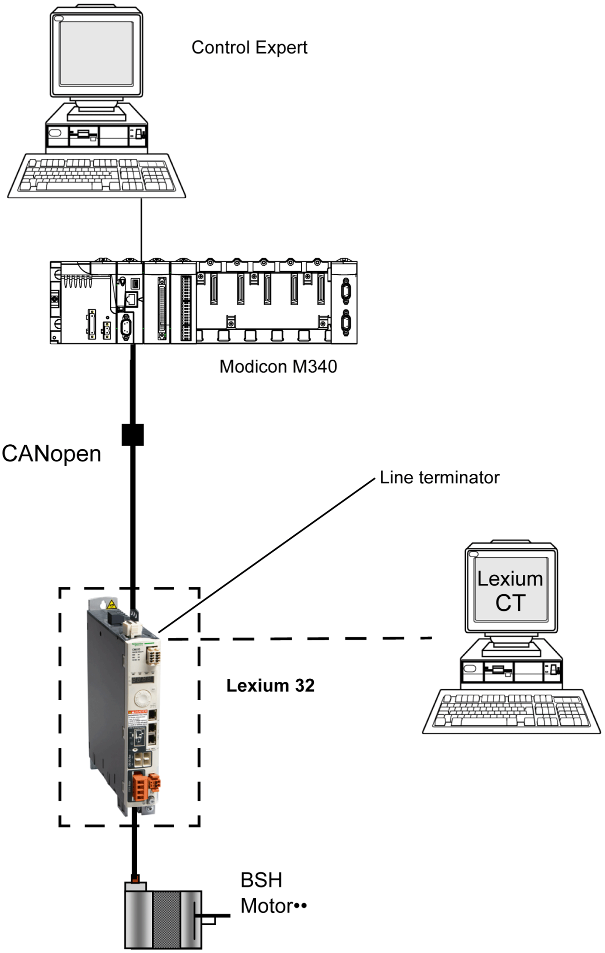
At a Glance

The proposed structure represents a simple structure which is designed to demonstrate motion control implementation principles.

This realistic structure may well be expanded upon with other devices in order to manage several axes.

Illustration

The figure below shows the structure used in the application:



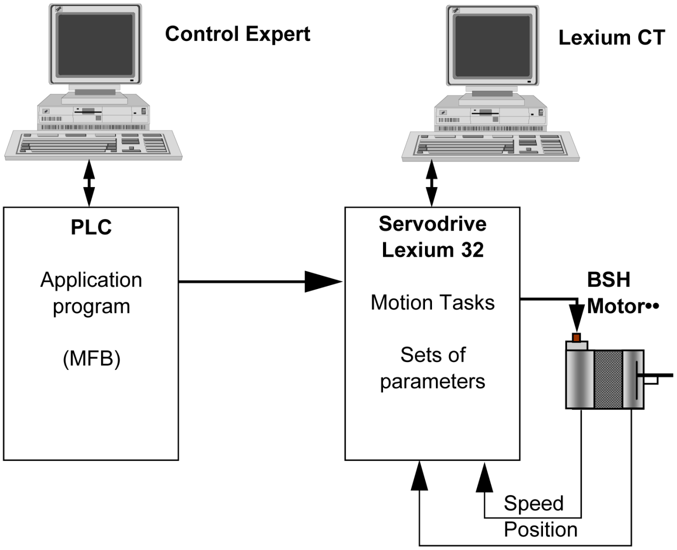
Software Requirements

Overview

Following the software requirements presented in the Quick Start Guide *(see page 22)*, Lexium CT is used for configuring and tuning the **Lexium 32**.

Functional Diagram for the Lexium 32

The following diagram shows the different functions performed by the PLC and the servodrive:



Versions

The following table lists the hardware and software versions used in the Architecture *(see page 115)*, enabling the use of MFBs in Control Expert.

Device	Software version used in the example	Version of firmware
Modicon M340	Unity Pro V5.0	>2.0
Lexium 32	Lexium CT V1.0	V1.x for Lexium 32 Advanced V1.y for Lexium 32 Modular

Hardware Requirements

References of the Hardware Used

The following table lists the hardware used in the architecture *(see page 95)*, enabling implementation of **Lexium 32** MFBs in Control Expert.

Hardware	Reference
Modicon M340 PLC	**BMX P34 20302**
Modicon M340 power supply	**BMX CPS 2000**
Modicon M340 rack	**BMX XBP 0800**
Lexium 32 Advanced	**LXM32AU90M2**
Lexium 32 connection cable to CANopen port of the PLC	**TCSCCN4F 3M3T/CAN**
CANopen Line terminator	TCSCAR013M120
Motor for **Lexium 32**	**BSH055••**

CANopen Bus Configuration Lexium 32

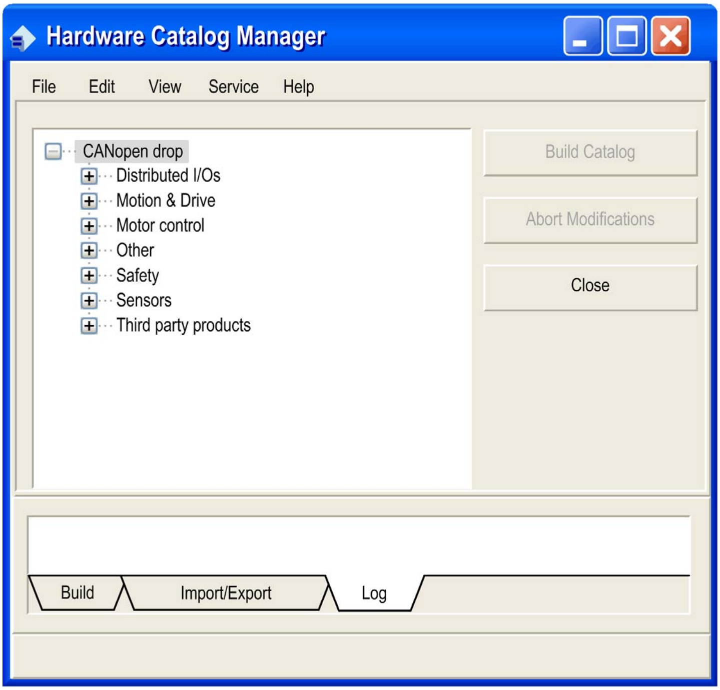
Overview

The implementation methodology for a CANopen bus using Modicon M340 is to:

- Upgrade the hardware catalog
- Configure *(see page 29)* the CANopen port of the CPU
- Declare the slave chosen from the hardware catalog (see paragraph below)
- Configure the slave
- Enable the configuration using Control Expert
- Check *(see page 34)* the CANopen bus in the Project browser

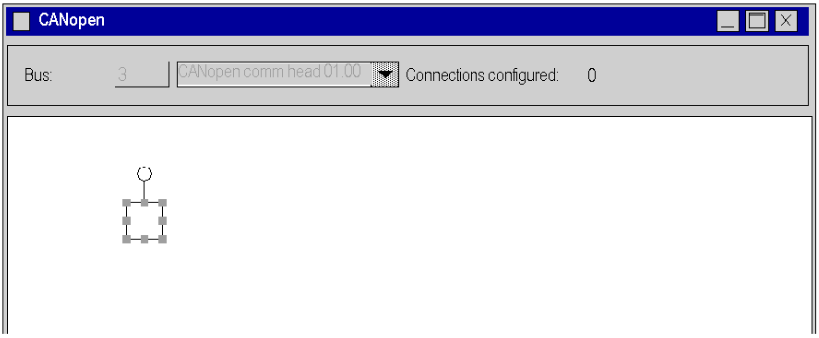
How to Upgrade the Hardware Catalog

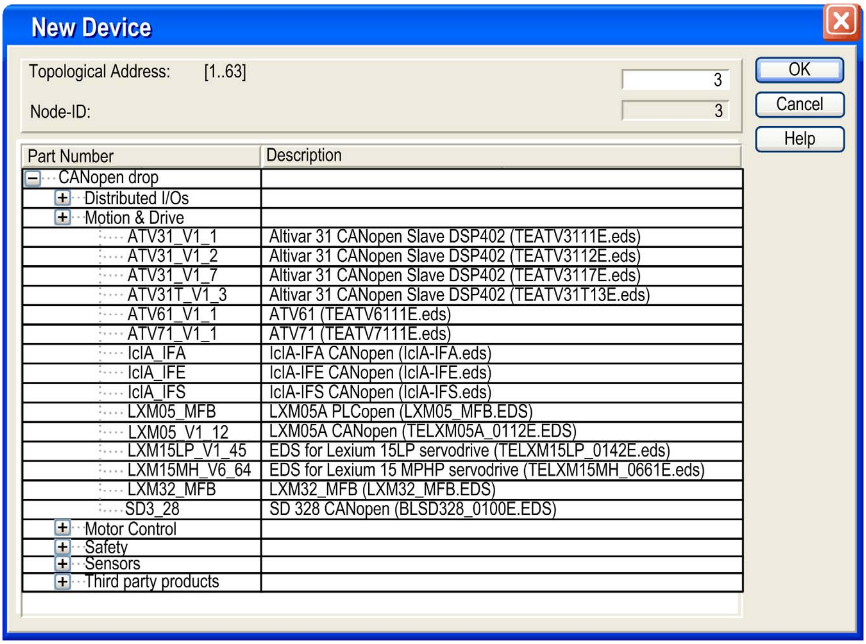
This table describes the procedure to configure the CANopen slave.

Step	Action
1	Open the Hardware Catalog Manager **Start → Programs → EcoStruxure Control Expert → Hardware Catalog Manager** **Result:**The Hardware Catalog Manager window appears:  Hardware Catalog Manager File Edit View Service Help CANopen drop Distributed I/Os Motion & Drive Motor control Other Safety Sensors Third party products Build Catalog Abort Modifications Close Build Import/Export Log
2	In the menu tab, click on File ==>Import User Devices, then import the LXM32_MFB.cpx file in the directory ...\ProgramData\Schneider Electric\ConfCatalog\Database\Motion (this file can be located in a hidden directory).

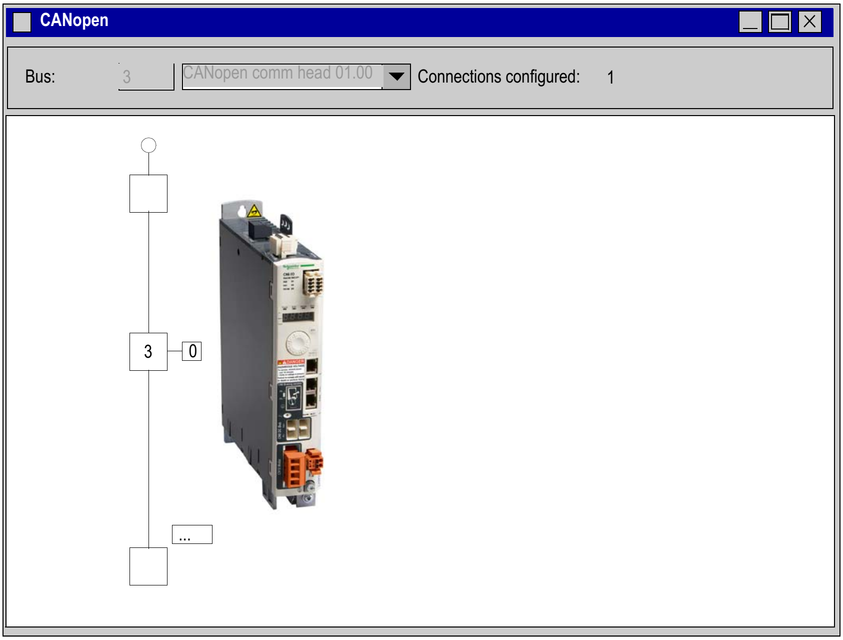
How to Configure the CANopen Slave

This table describes the procedure to configure the CANopen slave.

<table>
<tr><th>Step</th><th>Action</th></tr>
<tr><td>1</td><td>In the Control Expert <b>Project Browser</b>, fully expand the <b>Configuration</b> directory and then double-click on <b>CANopen</b>.
<b>Result:</b> The CANopen window appears:


Bus: 3 CANopen comm head 01.00 Connections configured: 0</td></tr>
<tr><td>2</td><td>Select <b>Edit</b> → <b>New device</b>.
<b>Result:</b> The New Device window appears:


Topological Address: [1..63] 3
Node-ID: 3
<table>
<tr><th>Part Number</th><th>Description</th></tr>
<tr><td>⊟ CANopen drop</td><td></td></tr>
<tr><td>⊞ Distributed I/Os</td><td></td></tr>
<tr><td>⊞ Motion &amp; Drive</td><td></td></tr>
<tr><td>ATV31_V1_1</td><td>Altivar 31 CANopen Slave DSP402 (TEATV3111E.eds)</td></tr>
<tr><td>ATV31_V1_2</td><td>Altivar 31 CANopen Slave DSP402 (TEATV3112E.eds)</td></tr>
<tr><td>ATV31_V1_7</td><td>Altivar 31 CANopen Slave DSP402 (TEATV3117E.eds)</td></tr>
<tr><td>ATV31T_V1_3</td><td>Altivar 31 CANopen Slave DSP402 (TEATV31T13E.eds)</td></tr>
<tr><td>ATV61_V1_1</td><td>ATV61 (TEATV6111E.eds)</td></tr>
<tr><td>ATV71_V1_1</td><td>ATV71 (TEATV7111E.eds)</td></tr>
<tr><td>IclA_IFA</td><td>IclA-IFA CANopen (IclA-IFA.eds)</td></tr>
<tr><td>IclA_IFE</td><td>IclA-IFE CANopen (IclA-IFE.eds)</td></tr>
<tr><td>IclA_IFS</td><td>IclA-IFS CANopen (IclA-IFS.eds)</td></tr>
<tr><td>LXM05_MFB</td><td>LXM05A PLCopen (LXM05_MFB.EDS)</td></tr>
<tr><td>LXM05_V1_12</td><td>LXM05A CANopen (TELXM05A_0112E.EDS)</td></tr>
<tr><td>LXM15LP_V1_45</td><td>EDS for Lexium 15LP servodrive (TELXM15LP_0142E.eds)</td></tr>
<tr><td>LXM15MH_V6_64</td><td>EDS for Lexium 15 MPHP servodrive (TELXM15MH_0661E.eds)</td></tr>
<tr><td>LXM32_MFB</td><td>LXM32_MFB (LXM32_MFB.EDS)</td></tr>
<tr><td>SD3_28</td><td>SD 328 CANopen (BLSD328_0100E.EDS)</td></tr>
<tr><td>⊞ Motor Control</td><td></td></tr>
<tr><td>⊞ Safety</td><td></td></tr>
<tr><td>⊞ Sensors</td><td></td></tr>
<tr><td>⊞ Third party products</td><td></td></tr>
</table></td></tr>
</table>

Step	Action
3	Set 3 in Topological Address. For the slave device choose Lexium 32.
4	Click on OK to confirm the choice. **Result:** The CANopen window appears with the new device selected:  CANopen Bus: 3 CANopen comm head 01.00 Connections configured: 1
5	Select **Edit** → **Open module**. If MFB has not already been selected, choose it in the Function area.
6	You will be asked to validate your modifications when closing the Device and CANopen windows.

Section 9.2
Configuring the Lexium 32

Basic Parameters for Lexium 32 using Lexium CT

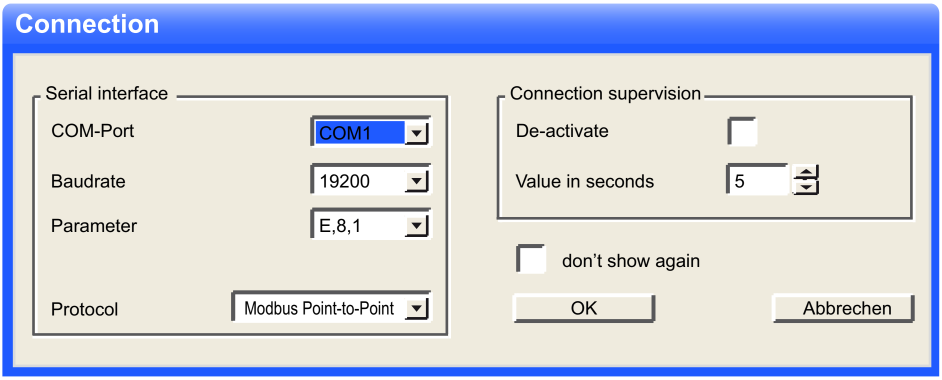
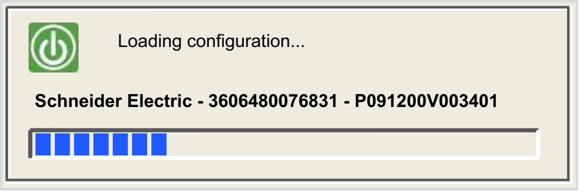
At a Glance

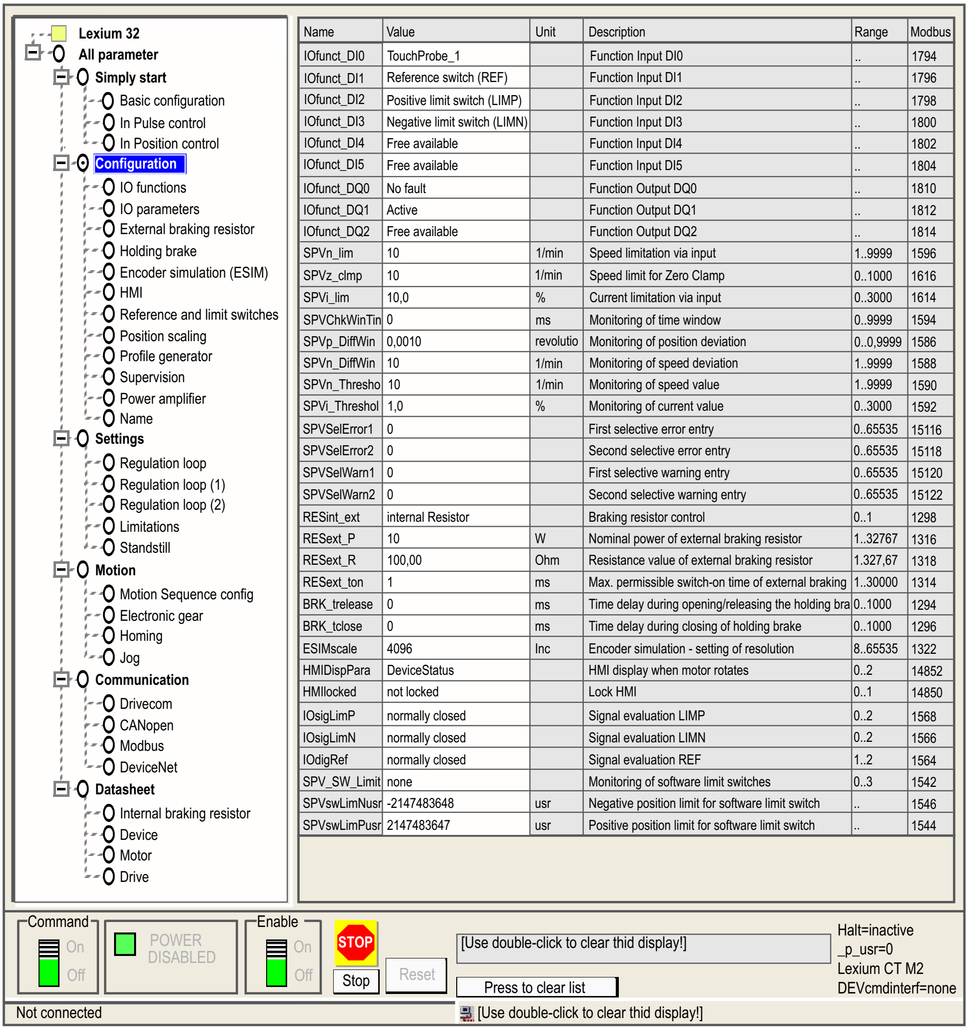
Lexium CT is a commissioning tool for axes intended for motion control applications.

Its graphic user interface provides a simple method for configuring the parameters of a **Lexium 32**-type servodrive.

Connecting to Lexium 32

This table describes the procedure for connecting to **Lexium 32**.

Step	Action
1	Start Lexium CT. Click on **Connection** and then select **ModbusSerialLine connection** connection. The Connection window is displayed:  Connection Serial interface COM-Port: COM1 Baudrate: 19200 Parameter: E,8,1 Protocol: Modbus Point-to-Point Connection supervision De-activate □ Value in seconds: 5 □ don't show again Select the COM-Port Validate by OK The following screen appears:  Loading configuration... **Schneider Electric - 3606480076831 - P091200V003401**

Step	Action
2	When configuration has been established, this general screen appears: 

Name	Value	Unit	Description	Range	Modbus
IOfunct_DI0	TouchProbe_1		Function Input DI0	..	1794
IOfunct_DI1	Reference switch (REF)		Function Input DI1	..	1796
IOfunct_DI2	Positive limit switch (LIMP)		Function Input DI2	..	1798
IOfunct_DI3	Negative limit switch (LIMN)		Function Input DI3	..	1800
IOfunct_DI4	Free available		Function Input DI4	..	1802
IOfunct_DI5	Free available		Function Input DI5	..	1804
IOfunct_DQ0	No fault		Function Output DQ0	..	1810
IOfunct_DQ1	Active		Function Output DQ1	..	1812
IOfunct_DQ2	Free available		Function Output DQ2	..	1814
SPVn_lim	10	1/min	Speed limitation via input	1..9999	1596
SPVz_clmp	10	1/min	Speed limit for Zero Clamp	0..1000	1616
SPVi_lim	10,0	%	Current limitation via input	0..3000	1614
SPVChkWinTin	0	ms	Monitoring of time window	0..9999	1594
SPVp_DiffWin	0,0010	revolutio	Monitoring of position deviation	0..0,9999	1586
SPVn_DiffWin	10	1/min	Monitoring of speed deviation	1..9999	1588
SPVn_Thresho	10	1/min	Monitoring of speed value	1..9999	1590
SPVi_Threshol	1,0	%	Monitoring of current value	0..3000	1592
SPVSelError1	0		First selective error entry	0..65535	15116
SPVSelError2	0		Second selective error entry	0..65535	15118
SPVSelWarn1	0		First selective warning entry	0..65535	15120
SPVSelWarn2	0		Second selective warning entry	0..65535	15122
RESint_ext	internal Resistor		Braking resistor control	0..1	1298
RESext_P	10	W	Nominal power of external braking resistor	1..32767	1316
RESext_R	100,00	Ohm	Resistance value of external braking resistor	1.327,67	1318
RESext_ton	1	ms	Max. permissible switch-on time of external braking	1..30000	1314
BRK_trelease	0	ms	Time delay during opening/releasing the holding bra	0..1000	1294
BRK_tclose	0	ms	Time delay during closing of holding brake	0..1000	1296
ESIMscale	4096	Inc	Encoder simulation - setting of resolution	8..65535	1322
HMIDispPara	DeviceStatus		HMI display when motor rotates	0..2	14852
HMIlocked	not locked		Lock HMI	0..1	14850
IOsigLimP	normally closed		Signal evaluation LIMP	0..2	1568
IOsigLimN	normally closed		Signal evaluation LIMN	0..2	1566
IOdigRef	normally closed		Signal evaluation REF	1..2	1564
SPV_SW_Limit	none		Monitoring of software limit switches	0..3	1542
SPVswLimNusr	-2147483648	usr	Negative position limit for software limit switch	..	1546
SPVswLimPusr	2147483647	usr	Positive position limit for software limit switch	..	1544

[Use double-click to clear thid display!]

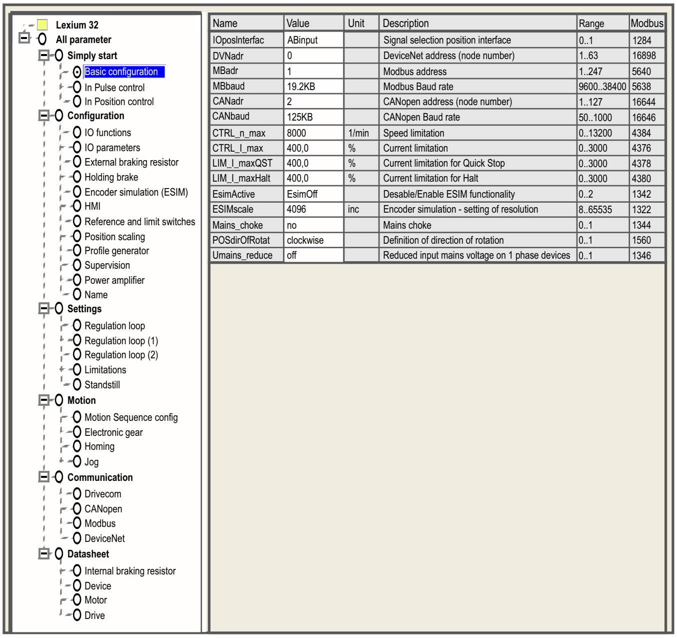
Halt=inactive
_p_usr=0
Lexium CT M2
DEVcmdinterf=none

Not connected

[Use double-click to clear thid display!]

Basic Parameters

This table describes the procedure for inputting the basic parameters:

<table>
<tr><th>Step</th><th>Action</th></tr>
<tr><td>1</td><td>Click on the <b>Basic Configuration</b>
The <b>Basic Configuration</b> window appears:

<table>
<tr><th>Name</th><th>Value</th><th>Unit</th><th>Description</th><th>Range</th><th>Modbus</th></tr>
<tr><td>IOposInterfac</td><td>ABinput</td><td></td><td>Signal selection position interface</td><td>0..1</td><td>1284</td></tr>
<tr><td>DVNadr</td><td>0</td><td></td><td>DeviceNet address (node number)</td><td>1..63</td><td>16898</td></tr>
<tr><td>MBadr</td><td>1</td><td></td><td>Modbus address</td><td>1..247</td><td>5640</td></tr>
<tr><td>MBbaud</td><td>19.2KB</td><td></td><td>Modbus Baud rate</td><td>9600..38400</td><td>5638</td></tr>
<tr><td>CANadr</td><td>2</td><td></td><td>CANopen address (node number)</td><td>1..127</td><td>16644</td></tr>
<tr><td>CANbaud</td><td>125KB</td><td></td><td>CANopen Baud rate</td><td>50..1000</td><td>16646</td></tr>
<tr><td>CTRL_n_max</td><td>8000</td><td>1/min</td><td>Speed limitation</td><td>0..13200</td><td>4384</td></tr>
<tr><td>CTRL_I_max</td><td>400,0</td><td>%</td><td>Current limitation</td><td>0..3000</td><td>4376</td></tr>
<tr><td>LIM_I_maxQST</td><td>400,0</td><td>%</td><td>Current limitation for Quick Stop</td><td>0..3000</td><td>4378</td></tr>
<tr><td>LIM_I_maxHalt</td><td>400,0</td><td>%</td><td>Current limitation for Halt</td><td>0..3000</td><td>4380</td></tr>
<tr><td>EsimActive</td><td>EsimOff</td><td></td><td>Desable/Enable ESIM functionality</td><td>0..2</td><td>1342</td></tr>
<tr><td>ESIMscale</td><td>4096</td><td>inc</td><td>Encoder simulation - setting of resolution</td><td>8..65535</td><td>1322</td></tr>
<tr><td>Mains_choke</td><td>no</td><td></td><td>Mains choke</td><td>0..1</td><td>1344</td></tr>
<tr><td>POSdirOfRotat</td><td>clockwise</td><td></td><td>Definition of direction of rotation</td><td>0..1</td><td>1560</td></tr>
<tr><td>Umains_reduce</td><td>off</td><td></td><td>Reduced input mains voltage on 1 phase devices</td><td>0..1</td><td>1346</td></tr>
</table>
This screen is used to set parameters for the servodrive’s CANopen address, the bus speed and the units used for acceleration, speed and position.</td></tr>
<tr><td>2</td><td>For the tutorial example, from this screen set or select the following:
● In the servodrive zone:
❍ The CANopen address to 2
❍ The baud rate of the bus to 500 Kbaud <i>(see Premium using EcoStruxure™ Control Expert, Motion Function Blocks, Start-up Guide)</i></td></tr>
<tr><td>3</td><td>Click <b>Items</b> → <b>Parameter</b> → <b>Save device parameters</b> in EEPROM to confirm the <b>SIMPLYSTART_BASICCONFIGURATION.</b>
<b>Result:</b> The <b>SIMPLYSTART_BASICCONFIGURATION</b> is saved and the main screen is displayed again.</td></tr>
<tr><td>4</td><td>Click on <b>Exit</b>.</td></tr>
</table>

NOTE: For information on how to adjust parameters correctly, please refer to the drive documentation

Section 9.3
Tuning the Lexium 32

Aim of this Section

This section gives an example of tuning the **Lexium 32** with Lexium CT.

What Is in This Section?

This section contains the following topics:

Tuning the Lexium 32

Operating modes

The various operating modes can be selected from the tabs in the Operating modes windows.

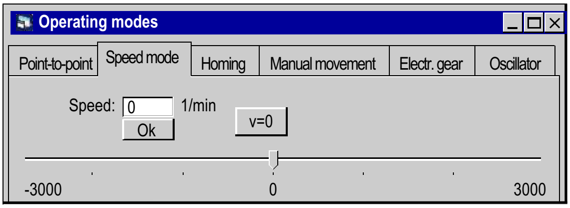
The windows is subdivided into two sections:

- Tabs for the selected operating mode and for setting specific parameters (top section)
- Display of status information (bottom section)

The user can switch between the tabs in the Operating Modes window without interfering with a currently active operating mode.

Profile Velocity

In the operating mode Profile Velocity, the drive accelerates to an adjustable target speed of rotation. You can set a motion profile with values for acceleration and deceleration ramps



Speed: 0 1/min

Debugging the Lexium 32

Pre-requisite

You are recommended to debug the axis dynamics before it is automatically started by the program.

Description

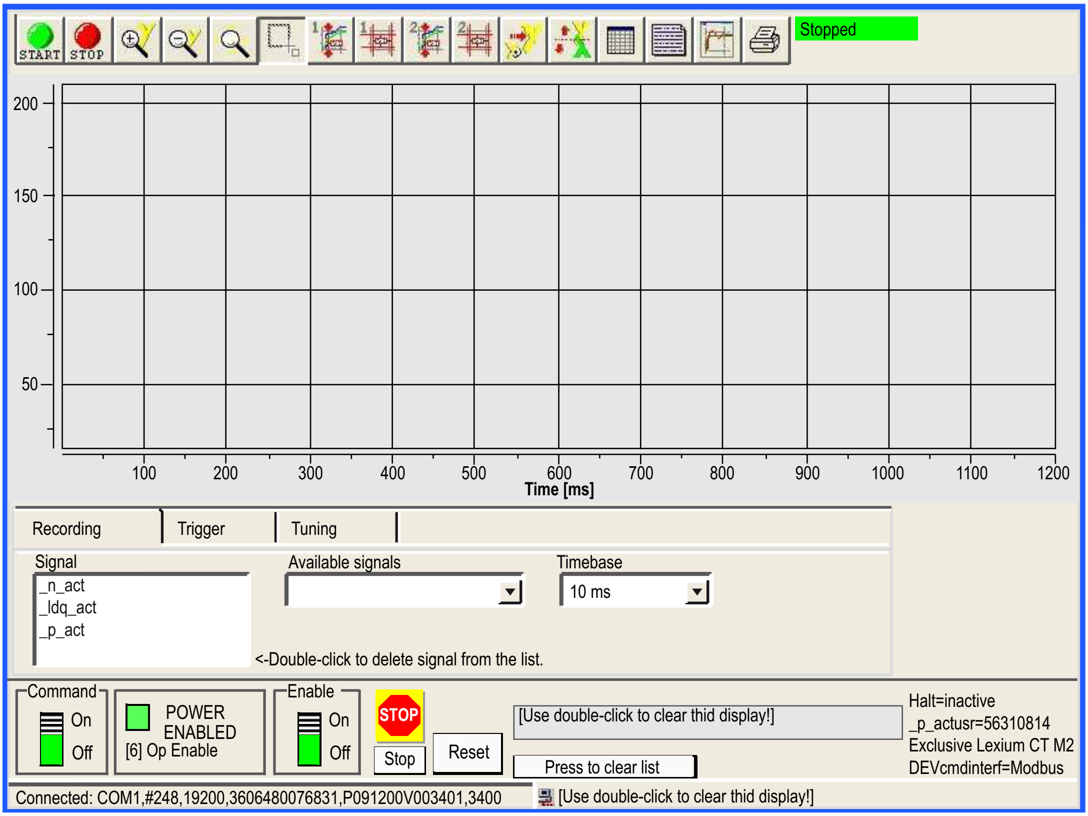
The commissioning software provides the “**Recording / Tuning**” function for visualizing internal device data during movements. The connected device stores the movement data to an internal memory for a defined recording period and then sends it to the PC. The PC processes the data and displays it in the form of charts or tables.

Recorded data can be saved on the PC, and can be archived or printed for documentation purposes.

Use the menu **Item” → Functions → Record / Tuning...”** to start the “record” function.

Illustration

The screen below can be accessed by clicking on the **Oscilloscope** tab:



Stopped

Signal
_n_act
_Idq_act
_p_act

Available signals

Timebase: 10 ms

<-Double-click to delete signal from the list.

[Use double-click to clear thid display!]

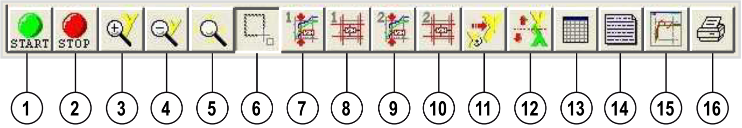
Halt=inactive
_p_actusr=56310814
Exclusive Lexium CT M2
DEVcmdinterf=Modbus

Connected: COM1,#248,19200,3606480076831,P091200V003401,3400

[Use double-click to clear thid display!]

Description Buttons

The buttons below can be accessed by clicking:

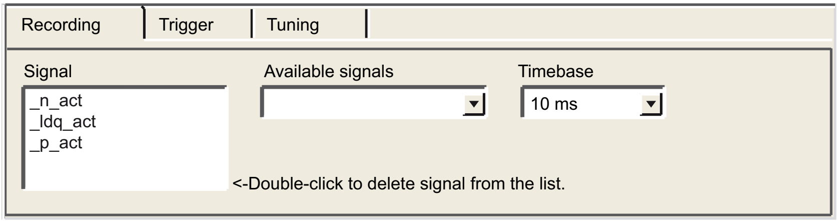


1. Start recording
2. Stop recording
3. Zoom in, y axis
4. Zoom out, y axis
5. Infinitely variable zoom, x axis and y axis
6. Zoom selected rectangle.
7. 1nd display of values for a specific time
8. Change displayed values for first display
9. 2nd display of values for a specific time.
10. Change displayed values for second display
11. Restore original display
12. Invert y axis
13. Display table of recorded values
14. Enter description
15. Show/hide configuration
16. Print recording

Recording

The desired parameters are selected in the “Available signals” input field. A maximum of 4 parameters can be selected. If a parameter is no longer required, it can be deselected by a double-click on the name of the parameter.

The desired recording increment is select in the “Timebase” input field. The smaller the “Time base”, the smaller the maximum recording time will be.



Signal

_n_act
_Idq_act
_p_act

Available signals

Timebase

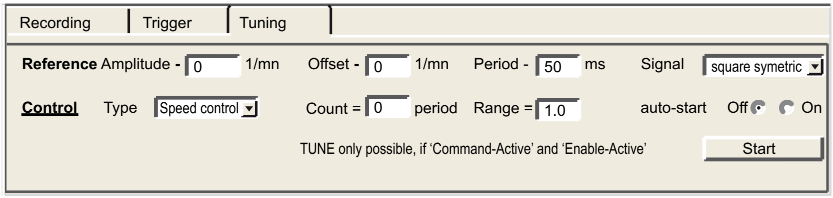
10 ms

<-Double-click to delete signal from the list.

Tuning

Tuning can only be started if the “Access” and “Enable” switches are set to “On”.

- The “Amplitude” field is used to set the maximum amplitude of the reference value
- The offset of the amplitude in positive or negative direction can be set in the “Offset” field.
- The duration of a period is set in the “Period” field.
- The signal type for the reference value is set in the “Signal ” dropdown list.
- The controller to be used is set in the “Type” dropdown list.
- The number of periods is specified in the “Count” field.
- The maximum number of revolutions that can be triggered by tuning can be set in the “Range” field. This value can, for example, help to avoid a movement to block.
- The “auto-start” radio buttons allow you to link the execution of the tuning movement and the start of recording. If the option is set to “Off” , the software displays a Start button. The Start button lets you trigger the tuning movement separately from starting the recording.
  NOTE: Settings that you may have made on the “Trigger” tab are lost if you set “auto-start” to “On”.



Reference Amplitude - 0 1/mn Offset - 0 1/mn Period - 50 ms Signal square symetric

Control Type Speed control Count = 0 period Range = 1.0 auto-start Off ● ○ On

TUNE only possible, if ‘Command-Active’ and ‘Enable-Active’

NOTE: For further information, please refer to the Lexium CT software user manual.

Chapter 10
Lexium 15MP/HP/LP Implementation for Motion Function Blocks

Aim of this Chapter

This chapter presents the implementation of Lexium 15MP/HP/LP servodrives according to the methodology *(see page 17)* described in the quick start guide *(see page 11)* with a Lexium 05. It only details the differences and actions for Lexium 15MP/HP/LP.

What Is in This Chapter?

This chapter contains the following sections:

Section 10.1
Adapting the Application to the Lexium 15MP/HP/LP

Aim of this Section

This section presents adaptation of the application to the **Lexium 15MP/HP/LP** with an architecture, and hardware and software requirements.

What Is in This Section?

This section contains the following topics:

Application Architecture with Lexium 15MP/HP/LP

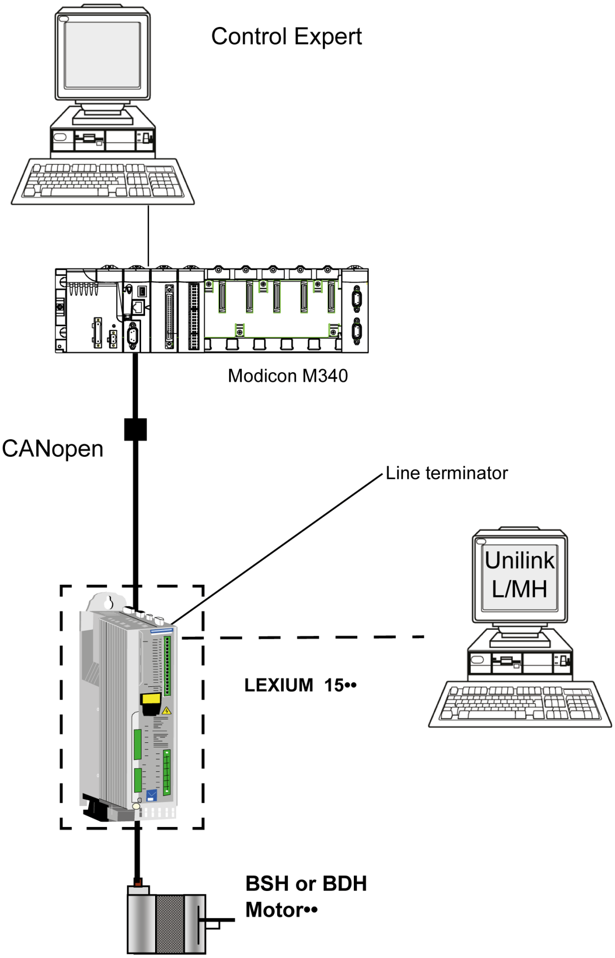
At a Glance

The proposed structure represents a simple structure which is designed to assimilate the motion control implementation principles.

This realistic structure may well be expanded upon with other devices in order to manage several axes.

Illustration

The figure below shows the structure used in the application:



Software Requirements

Overview

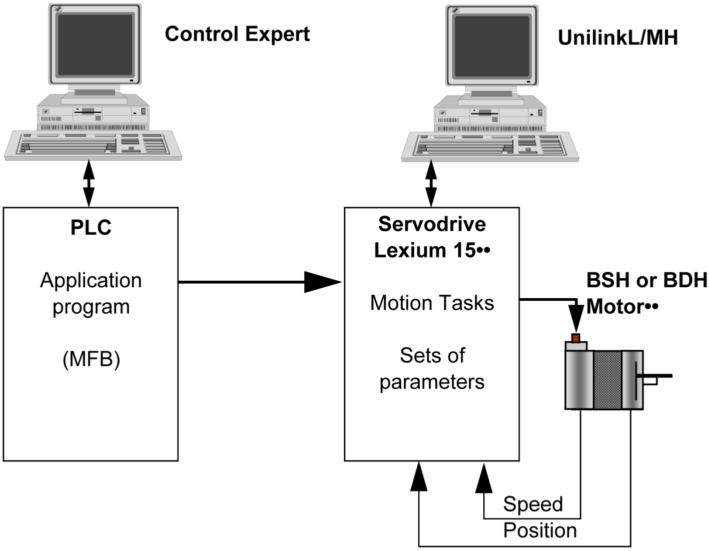
As regards the software requirements presented in the quick start guide *(see page 22)*, PowerSuite is used for configuring and tuning the **Lexium 05**.

PowerSuite for **Lexium 05** enables tuning of the axis and guarantees a simple method for configuring the parameters of a **Lexium 05** servodrive.

Unilink L/MH for **Lexium 15••** does the same, but for **Lexium 15••** servodrive.

Functional Diagram for the Lexium 15••

The following diagram shows the different functions performed by the PLC and the servodrive:



Versions

The following table lists the hardware and software versions used in the architecture *(see page 115)*, enabling the use of MFBs in Control Expert.

Device	Software version used in the example	Version of firmware
Modicon M340	Unity Pro V4.0	-
Lexium 15LP	Unilink V1.5	V1.45 only MFB Function V2.36 Managed by MTM
Lexium 15MH	Unilink V4.0	Compatible sInce V6.64

Hardware Requirements

References of the Hardware Used

The following table lists the hardware used in the architecture *(see page 115)*, enabling implementation of **Lexium 15MP** MFBs in Control Expert.

Hardware	Reference
Modicon M340 PLC	**BMX P34 2030**
Modicon M340 power supply	**BMX CPS 2000**
Modicon M340 rack	**BMX XBP 0800**
Lexium 15MP Servodrive	**LXM15MD28N4**
Lexium 15MP connection cable to CANopen port of the PLC	**TLA CD CBA •••**
CANopen connector for **Lexium 15MP**	**AM0 2CA 001 V000**
Motor for **Lexium 15MP**	**BPH055••**

The following table lists the hardware used in the architecture *(see page 115)*, enabling implementation of **Lexium 15LP** MFBs in Control Expert.

Hardware	Reference
Modicon M340 PLC	**BMX P34 2030**
Modicon M340 power supply	**BMX CPS 2000**
Modicon M340 rack	**BMX XBP 0800**
Lexium 15LP Servodrive	**LXM15LD13M3**
Lexium 15MP connection cable to CANopen port of the PLC	**TLA CD CBA •••**
CANopen connector for **Lexium 15LP**	**AM0 2CA 001 V000**
Motor for **Lexium 15LP**	**AKM 31E**

NOTE: the line terminator is an interrupter built into the **AM0 2CA 001 V000** CANopen connector.

Section 10.2
CANopen Bus Configuration Lexium 15MP/HP/LP

Configuration of the CANopen Slave on CANopen bus

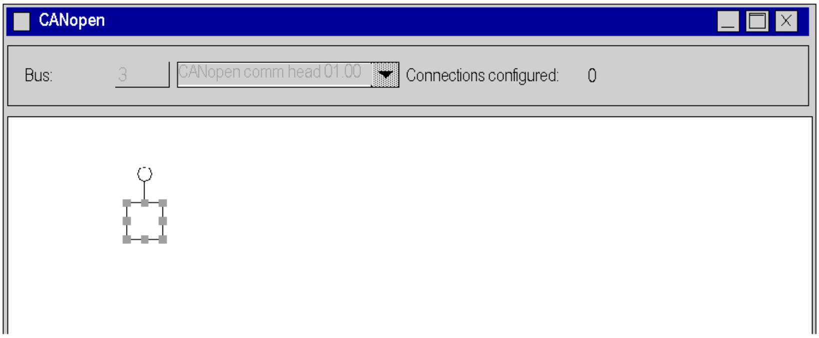
Overview

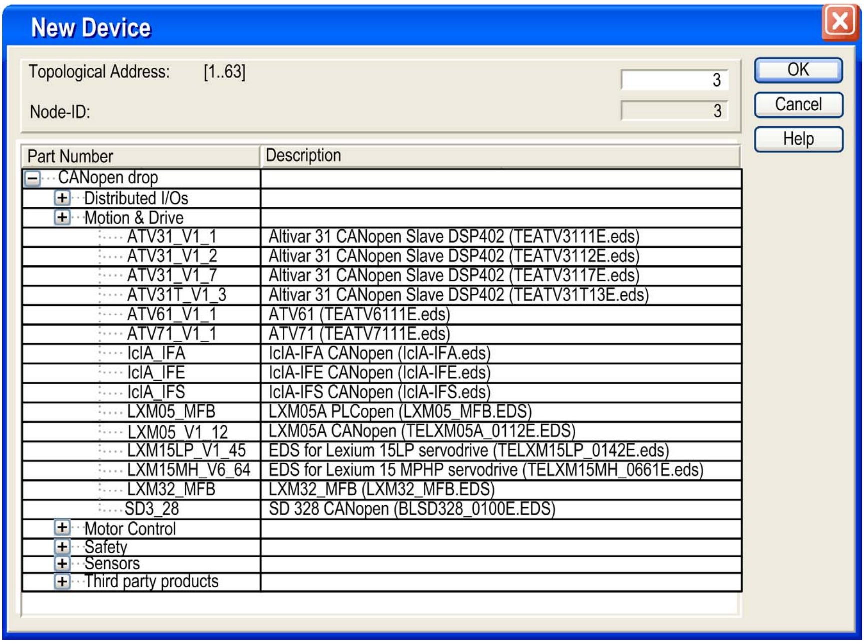
The implementation methodology for a CANopen bus using Modicon M340 is to:

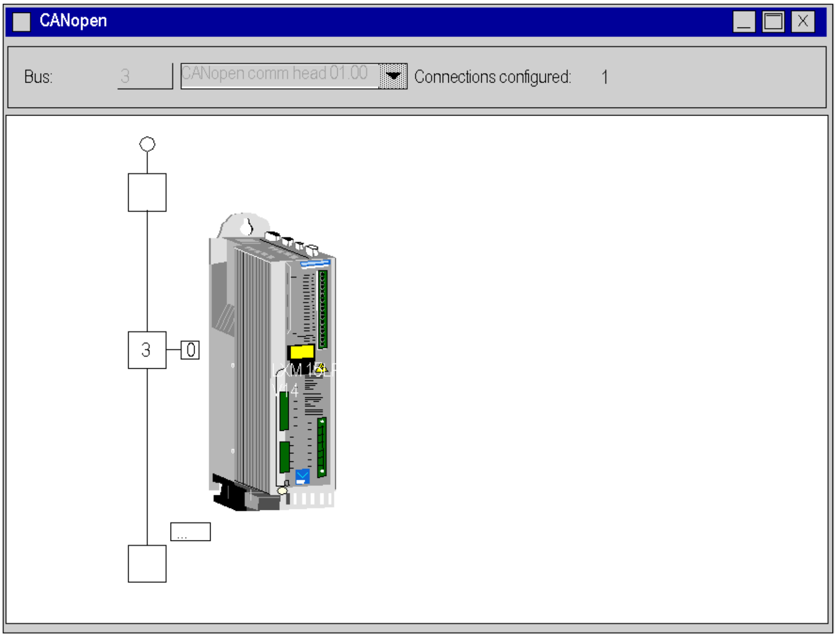
- configure *(see page 29)* the CANopen port of the CPU,
- declarate the slave chosen from the hardware catalog (see paragraph bellow),
- configure the slave,
- enable the configuration using Control Expert,
- check *(see page 34)* the CANopen bus in the Project browser.

How to Configure the CANopen Slave

This table describes the procedure to configure the CANopen slave.

Step	Action
1	In the Control Expert **Project Browser**, fully expand the **Configuration** directory and then double-click on **CANopen**. **Result:** The CANopen window appears:  Bus: 3 CANopen comm head 01.00 Connections configured: 0

<table>
<tr><th>Step</th><th>Action</th></tr>
<tr><td>2</td><td>Select <b>Edit</b> → <b>New device</b>.
<b>Result:</b> The New Device window appears:

New Device
Topological Address: [1..63] 3
Node-ID: 3
OK
Cancel
Help
<table>
<tr><th>Part Number</th><th>Description</th></tr>
<tr><td>⊟ CANopen drop</td><td></td></tr>
<tr><td>⊞ Distributed I/Os</td><td></td></tr>
<tr><td>⊞ Motion &amp; Drive</td><td></td></tr>
<tr><td>ATV31_V1_1</td><td>Altivar 31 CANopen Slave DSP402 (TEATV3111E.eds)</td></tr>
<tr><td>ATV31_V1_2</td><td>Altivar 31 CANopen Slave DSP402 (TEATV3112E.eds)</td></tr>
<tr><td>ATV31_V1_7</td><td>Altivar 31 CANopen Slave DSP402 (TEATV3117E.eds)</td></tr>
<tr><td>ATV31T_V1_3</td><td>Altivar 31 CANopen Slave DSP402 (TEATV31T13E.eds)</td></tr>
<tr><td>ATV61_V1_1</td><td>ATV61 (TEATV6111E.eds)</td></tr>
<tr><td>ATV71_V1_1</td><td>ATV71 (TEATV7111E.eds)</td></tr>
<tr><td>IclA_IFA</td><td>IclA-IFA CANopen (IclA-IFA.eds)</td></tr>
<tr><td>IclA_IFE</td><td>IclA-IFE CANopen (IclA-IFE.eds)</td></tr>
<tr><td>IclA_IFS</td><td>IclA-IFS CANopen (IclA-IFS.eds)</td></tr>
<tr><td>LXM05_MFB</td><td>LXM05A PLCopen (LXM05_MFB.EDS)</td></tr>
<tr><td>LXM05_V1_12</td><td>LXM05A CANopen (TELXM05A_0112E.EDS)</td></tr>
<tr><td>LXM15LP_V1_45</td><td>EDS for Lexium 15LP servodrive (TELXM15LP_0142E.eds)</td></tr>
<tr><td>LXM15MH_V6_64</td><td>EDS for Lexium 15 MPHP servodrive (TELXM15MH_0661E.eds)</td></tr>
<tr><td>LXM32_MFB</td><td>LXM32_MFB (LXM32_MFB.EDS)</td></tr>
<tr><td>SD3_28</td><td>SD 328 CANopen (BLSD328_0100E.EDS)</td></tr>
<tr><td>⊞ Motor Control</td><td></td></tr>
<tr><td>⊞ Safety</td><td></td></tr>
<tr><td>⊞ Sensors</td><td></td></tr>
<tr><td>⊞ Third party products</td><td></td></tr>
</table>
</td></tr>
<tr><td>3</td><td>Set 3 in Topological Address.
For the slave device choose Lexium15LP_V1_4 for a Lexium 15LP or Lexium15MH_V6_61for a Lexium 15MP.</td></tr>
</table>

Step	Action
4	Click on OK to confirm the choice. **Result:** The CANopen window appears with the new device selected:  CANopen Bus: 3 CANopen comm head 01.00 Connections configured: 1
5	Select **Edit → Open module**. If MFB has not already been selected, choose it in the Function area.
6	You will be asked to validate your modifications when closing the Device and CANopen windows.

Section 10.3
Configuring the Lexium 15MP/HP/LP

Aim of this Section

This section describes the basic servodrive configurations using **Unilink L/MH** for **Lexium 15MP/HP/LP**.

What Is in This Section?

This section contains the following topics:

Basic Parameters for Lexium 15MP using Unilink MH

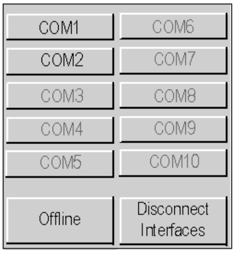
At a Glance

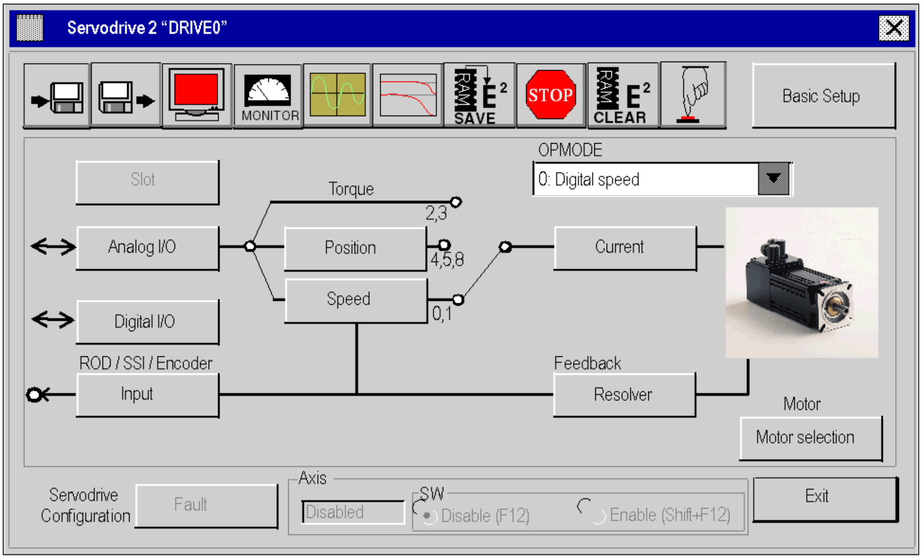
Unilink is a commissioning tool for axes intended for motion control applications.

Its graphic user interface provides a simple method for configuring the parameters of a **Lexium 15MP**-type servodrive.

Connecting to Lexium 15MP

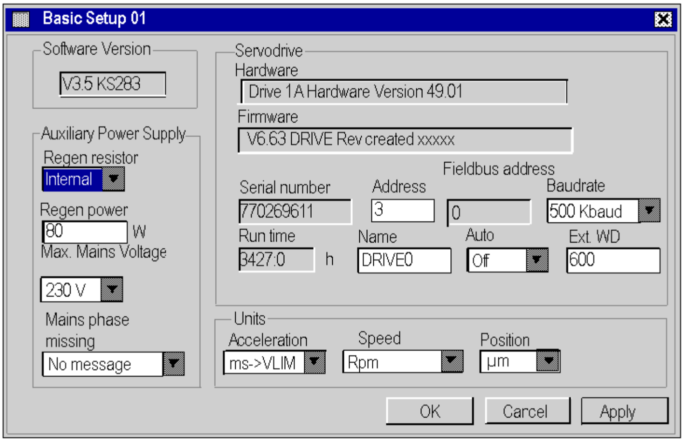
This table describes the procedure for connecting to **Lexium 15MP** :

Step	Action
1	Start Unilink MH via **Start** → **Program** → **Unilink** → **Unilink MH**. A communication window is displayed on main window of Unilink MH:  COM1 COM6 COM2 COM7 COM3 COM8 COM4 COM9 COM5 COM10 Offline Disconnect Interfaces If the port that you are using is available (i.e. is not being used by other devices or programs), the name COM1, COM2, COM3, COM4, COM5, COM6, COM7, COM 8, COM9, COM10 appears in black. Otherwise, it appears in grey.

Step	Action
2	Click on one of these communication ports (the port that you use on your PC) to transfer the values of the servodrive parameters to your PC. When communication has been established, this general screen appears:  Servodrive 2 "DRIVE0" MONITOR SAVE STOP CLEAR Basic Setup OPMODE 0: Digital speed Slot Torque 2,3 Analog I/O Position 4,5,8 Current Speed 0,1 Digital I/O ROD / SSI / Encoder Input Feedback Resolver Motor Motor selection Servodrive Configuration Fault Axis Disabled SW Disable (F12) Enable (Shift+F12) Exit

Basic Parameters

This table describes the procedure for inputting the basic parameters:

Step	Action
1	Click on the **Basic Setup** button in the general screen. The **Basic Setup** window appears:  Basic Setup 01 Software Version: V3.5 KS283 Auxiliary Power Supply Regen resistor: Internal Regen power: 80 W Max. Mains Voltage: 230 V Mains phase missing: No message Servodrive Hardware: Drive 1A Hardware Version 49.01 Firmware: V6.63 DRIVE Rev created xxxxx Serial number: 770269611 Address: 3 Fieldbus address: 0 Baudrate: 500 Kbaud Run time: 3427:0 h Name: DRIVE0 Auto: Off Ext. WD: 600 Units Acceleration: ms->VLIM Speed: Rpm Position: µm This screen is used to set parameters for the servodrive's CANopen address, the bus speed and the units used for acceleration, speed and position.
2	For the tutorial example, from this screen set or select the following: ● In the servodrive zone: ❍ the CANopen address to 2 ❍ the baud rate of the bus to 500 Kbaud *(see page 29)* ● In the Unit *(see EcoStruxure™ Control Expert, Motion Function Blocks, Block Library)* zone: ❍ the acceleration in ms->VLIM ❍ the speed in rpm ❍ the position in µm
3	Click on the **Motor Selection**, **Current**, **Resolver** buttons to declare the motor and the feedback parameters. **Note:** for information on how to declare the motor correctly, please refer to the motor documentation.
4	Click **OK** to confirm the basic configuration. **Result:** the basic setup is saved and the main screen is displayed again. **Note:** when certain ASCII parameters have been enabled, a window appears asking you to save changes to the servodrive's EEPROM memory. Click on **OK** to restart the servodrive and update the memory.
5	Click on **Exit**.

Basic Parameters for Lexium 15LP using Unilink L

At a Glance

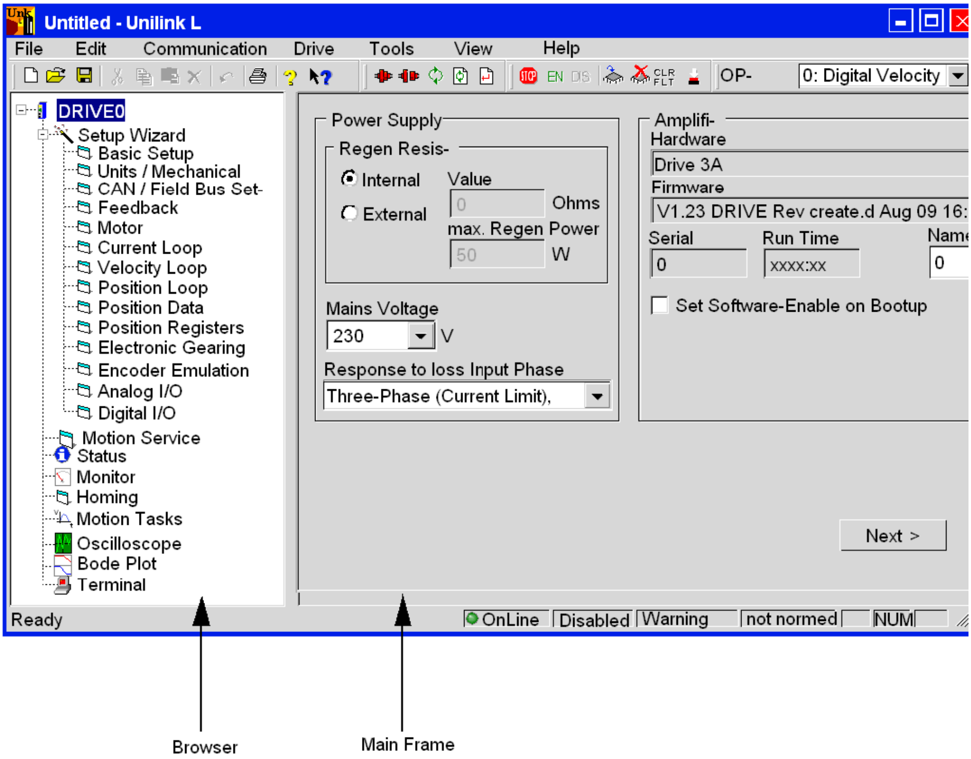
Unilink is a commissioning tool for axes intended for motion control applications.

Its graphic user interface provides a simple method for configuring the parameters of a **Lexium 15LP**-type servodrive.

Connecting to Lexium 15LP

This table describes the procedure for connecting to **Lexium 15LP** :

Step	Action
1	Start Unilink L via **Start** → **Program** → **Unilink** → **Unilink L**. **Result**: a window ask you if you would like to connect to the drive
2	Click on the **Yes** button. **Result**: a window to select the device appears.
3	Select **RS-232** and click on the **OK** button. **Result**: a window of RS-232 settings appears.
4	Set the serial port (COM1 to COM10), the Baud Rate (38400), the Timeout (2000ms).

Step	Action
5	Click on the **OK** button. **Result**: Unilink L software appears. 

Untitled - Unilink L

OP- 0: Digital Velocity

Power Supply

Regen Resis-

● Internal

○ External

Value 0 Ohms

max. Regen Power 50 W

Mains Voltage 230 V

Response to loss Input Phase: Three-Phase (Current Limit),

Amplifi-

Hardware: Drive 3A

Firmware: V1.23 DRIVE Rev create.d Aug 09 16:

Serial: 0

Run Time: xxxx:xx

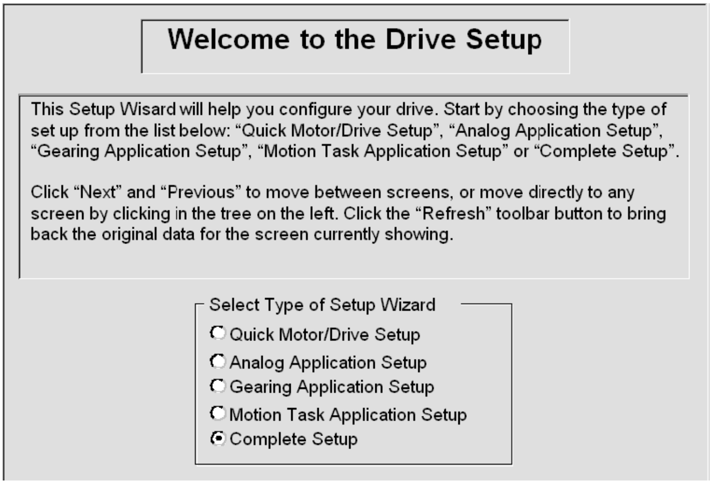
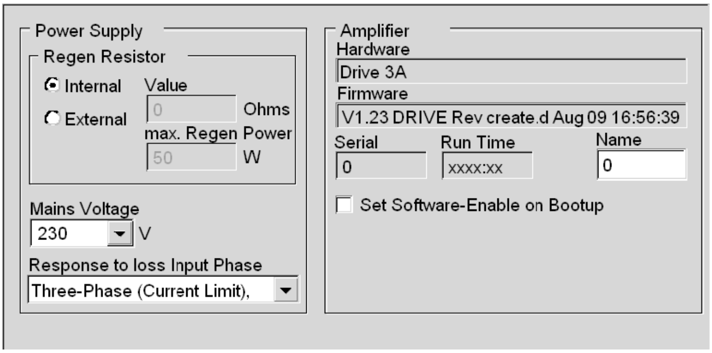
Nam: 0

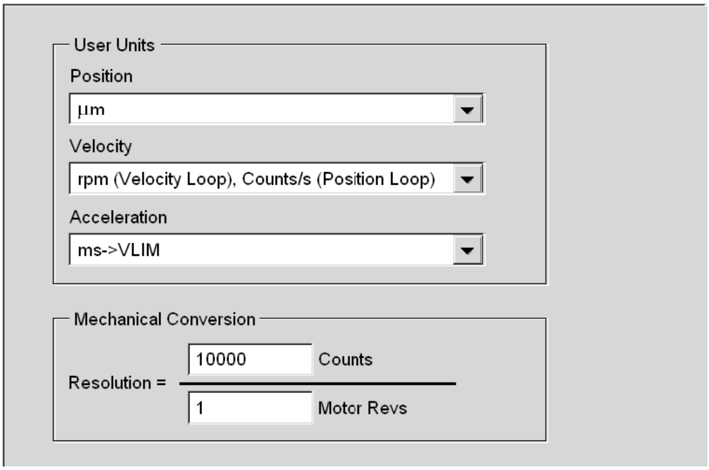
□ Set Software-Enable on Bootup

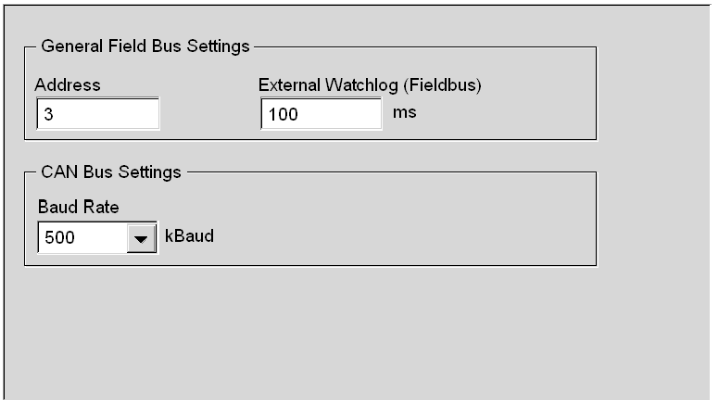
Ready | OnLine | Disabled | Warning | not normed | NUM

Basic Parameters

This table describes the procedure for inputting the basic parameters:

Step	Action
1	Click on the **Setup wizard** on the browser. **Result**:the **Drive Setup** screen in the main frame appears:  Welcome to the Drive Setup This Setup Wisard will help you configure your drive. Start by choosing the type of set up from the list below: “Quick Motor/Drive Setup”, “Analog Application Setup”, “Gearing Application Setup”, “Motion Task Application Setup” or “Complete Setup”. Click “Next” and “Previous” to move between screens, or move directly to any screen by clicking in the tree on the left. Click the “Refresh” toolbar button to bring back the original data for the screen currently showing. Select Type of Setup Wizard ○ Quick Motor/Drive Setup ○ Analog Application Setup ○ Gearing Application Setup ○ Motion Task Application Setup ● Complete Setup
2	Select the **Complete Setup** on the screen. **Result**: the browser with all configurations links appears.
3	Click on the **Basic Setup** on the browser. **Result**: the **Basic Setup** screen in the main frame appears:  Power Supply Regen Resistor ● Internal ○ External Value: 0 Ohms max. Regen Power: 50 W Mains Voltage: 230 V Response to loss Input Phase: Three-Phase (Current Limit), Amplifier Hardware: Drive 3A Firmware: V1.23 DRIVE Rev create.d Aug 09 16:56:39 Serial: 0 Run Time: xxxx:xx Name: 0 □ Set Software-Enable on Bootup This screen is used to set parameters of the power supply.

Step	Action
4	Click on the **Units/Mechanical** on the browser. The **Units/Mechanical** screen in the main frame appears:  User Units Position: µm Velocity: rpm (Velocity Loop), Counts/s (Position Loop) Acceleration: ms->VLIM Mechanical Conversion Resolution = 10000 Counts / 1 Motor Revs For the tutorial example, from this screen set or select the following: ● In the User Units zone: ❍ the acceleration in ms->VLIM ❍ the speed in rpm ❍ the position in µm

Step	Action
5	Click on the **CAN / Field Bus Settings** on the browser. The **CAN / Field Bus Settings** screen in the main frame appears:  General Field Bus Settings Address: 3 External Watchlog (Fieldbus): 100 ms CAN Bus Settings Baud Rate: 500 kBaud For the tutorial example, from this screen set or select the following: ● In the General Field Bus and CAN Bus Settings zones: ❍ the CANopen address to 3 ❍ the baud rate of the bus to 500 Kbauds
6	Click on the **Motor**, **Resolver** folders on the browser to declare the motor and the feedback parameters. **Note:** for information on how to declare the motor correctly, please refer to the motor documentation.
7	Save the parameters via **Drive** → **Save to EEPROM**. **Result:** the basic setup is saved and the main screen is displayed again.

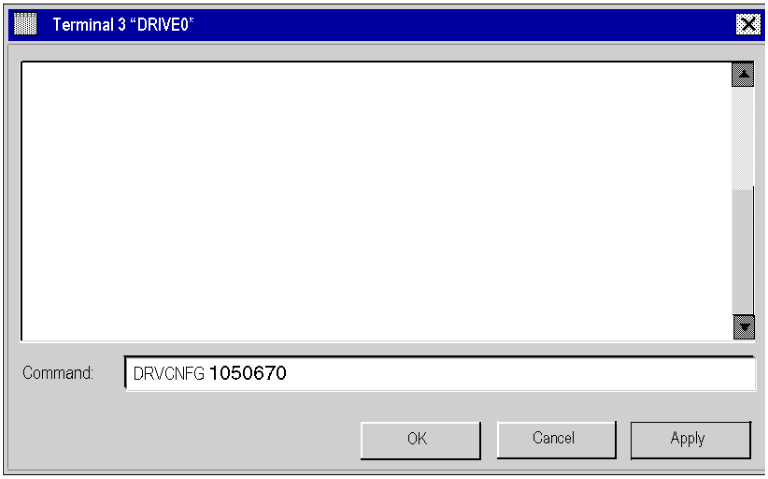
Specific Parameters for Lexium 15 MP/HP/LP using Unilink

At a Glance

Specific parameters are entered in addition to the basic *(see Premium using EcoStruxure™ Control Expert, Motion Function Blocks, Start-up Guide)*parameters. These specific parameters supplement the configuration of **Lexium 15 MP/HP/LP** by modifying certain ASCII codes using the **Terminal** window.

Specific Parameters

This table describes the procedure for inputting the specific parameters of **Lexium 15 MP/HP/LP**:

Step	Action
1	Click on the 🖥 **Terminal** icon on the general page. The **Terminal** window is displayed:  DRVCNFG 1050670 This screen is used to fully configure the connection point of a **Lexium 15MP/HP/LP**.
2	For Lexium 15 MP/HP i enter in the **Command** field: ● DRVCNFG 1050670 For Lexium 15 LP enter in the **Command** field: ● INPT2 x1.5 task time, or IN20Mode42 either **MAST** or **FAST**
3	Click on **Apply** to confirm the configuration of this ASCII parameter.

Step	Action
4	For Lexium 15 MP/HP repeat the steps by entering in the **Command** field: ● DRVCNFG2 64 ● INPT x1.5 task time **MAST** or **FAST** ● ENGAGE 1
5	Click on **OK** to confirm the last **Command** and return to the general page.
6	Click on the RAM E^2 SAVE **Save** icon on the general page to save the basic and specific parameters to the servodrive’s EEPROM memory.
7	Close the general window and click on **DIS** to disconnect from the servodrive.

Command

Enter the ASCII command here, with the corresponding parameters. Confirm the entry with **RETURN** or press the **APPLY** button to start the transmission.

⚠ CAUTION

UNEXPECTED APPLICATION BEHAVIOR

Before sending the ASCII command , ensure that is appropriate to the equipment.

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

Section 10.4
Tuning the Lexium 15MP/HP/LP

Debugging the axis

Pre-requisite

You are recommended to debug the axis dynamics before it is automatically started by the program.

Description

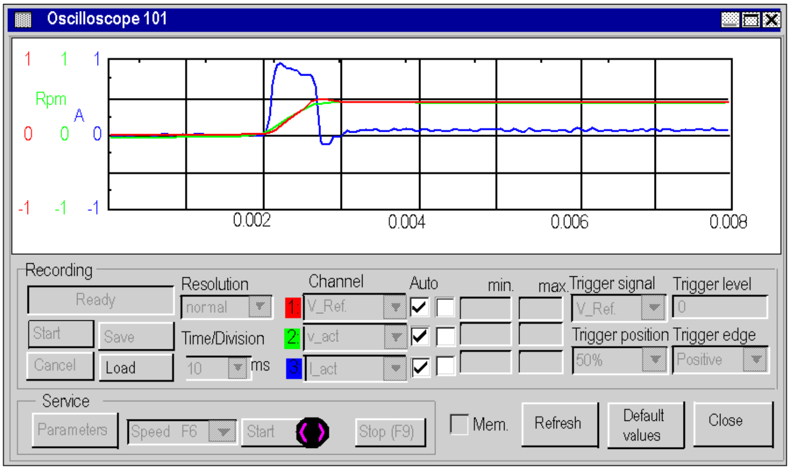
The oscilloscope is one way of carrying out the debug operation.

It allows you to:

- display up to three variables simultaneously, as a function of time
- save the recorded measurements to a data medium in CSV format (can be used with MS-Excel)
- load a CSV data file and restore the curves on the oscilloscope diagram
- use certain services

Illustration for Lexium 15MH

The screen below can be accessed by clicking on the **Unilink MH** menu's **Tools** → **Oscilloscope**:



Oscilloscope 101

Recording: Ready

Resolution: normal

Time/Division: 10 ms

Channel		Auto		min.	max.
1	V_Ref.	☑	□		
2	v_act	☑	□		
3	I_act	☑	□		

Trigger signal: V_Ref.

Trigger level: 0

Trigger position: 50%

Trigger edge: Positive

Service: Speed F6

□ Mem.

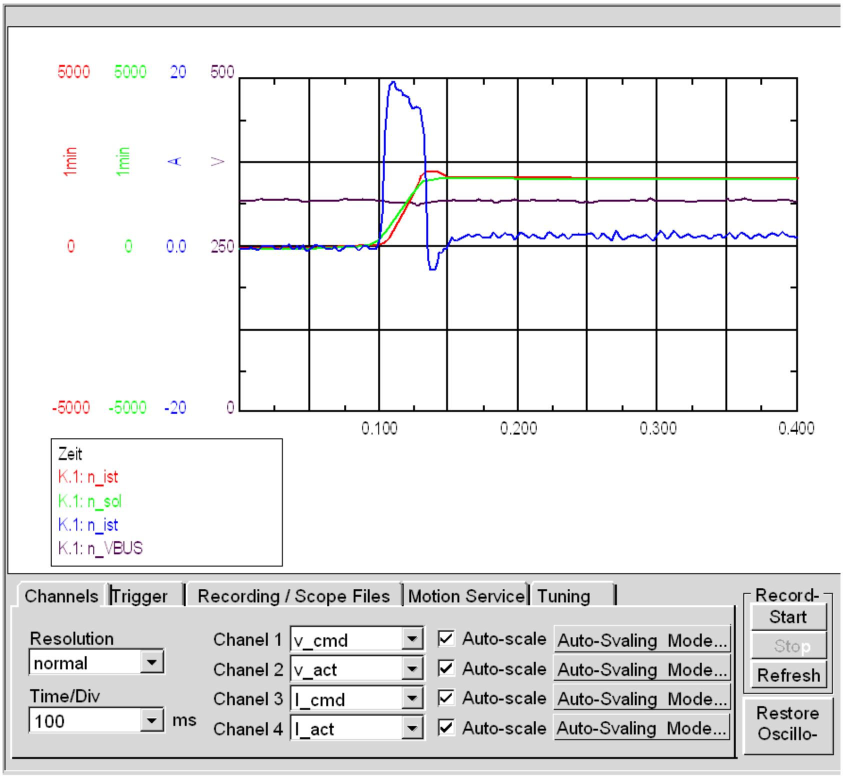
How to start service for Lexium 15MH

The table below explains how to use a service function with a Lexium 15MH:

Step	Action
1	On the field **Service**, select one of the service functions *(see page 134)* described below.
2	Click on the **Parameters** button.
3	Set the corresponding parameter.
4	Then start the function by using the **Start** button.
5	The function will continue to be performed until you click on the **Stop** button or press the function key **F9**.

Illustration for Lexium 15LP

The screen below can be accessed by clicking the folder **Oscilloscope** on the **Unilink L** browser's:



Resolution: normal

Time/Div: 100 ms

Chanel 1: v_cmd ☑ Auto-scale Auto-Svaling Mode...

Chanel 2: v_act ☑ Auto-scale Auto-Svaling Mode...

Chanel 3: I_cmd ☑ Auto-scale Auto-Svaling Mode...

Chanel 4: I_act ☑ Auto-scale Auto-Svaling Mode...

How to start service for Lexium 15LP

The table below explains how to use a service function with a Lexium 15LP:

Step	Action
1	Click on the **Motion Services** tab..
2	Select one of the service functions *(see page 134)* described below.
3	Click on the **Parameters** button.
4	Set the corresponding parameter.
5	Then start the function by using the **Start** button.
6	The function will continue to be performed until you click on the **Stop** button.

Service Functions

The table below explains how to use a service function:

Direct current	Apply a direct current to the motor with adjustable size and electrical field-vector angle. The changeover from speed control to current control is made automatically, commutation is made independently of the feedback (resolver or similar). The rotor locks onto a stator pole.
Speed	Operates the drive at constant speed. An internal digital setpoint is provided (speed is adjustable).
Torque	Operates the drive with constant current. An internal digital setpoint is provided (current is adjustable). The changeover from speed control to current control is made automatically, commutation is made independently of the feedback (resolver or similar).
Reversing mode	Operates the drive in reversing mode, with separately adjustable speed and reversing time for each direction of rotation.
Motion task	Starts the motion task that is selected in the screen page "Entry of service parameters".
Zero	Function used for feedback setting in conjunction with the positioning phase. This function can only be accessed in OMODE2.

NOTE: For further information, please refer to the Unilink software user manual.

NOTE: Once the parameters have been correctly set, you are advised to save them in EEPROM and to make a backup copy of them in a file.

Chapter 11
ATV 31 Implementation for Motion Function Blocks

Aim of this Chapter

This chapter presents the implementation of an ATV 31 servodrive according to the methodology *(see page 17)* described in the quick start guide *(see page 11)* with a Lexium 05. It only details the differences and actions for an ATV 31.

What Is in This Chapter?

This chapter contains the following sections:

Section 11.1
Adapting the Application to the ATV 31

Aim of this Section

This section presents adaptation of the application to the **ATV 31** with an architecture, and hardware and software requirements.

What Is in This Section?

This section contains the following topics:

Application Architecture with an ATV 31

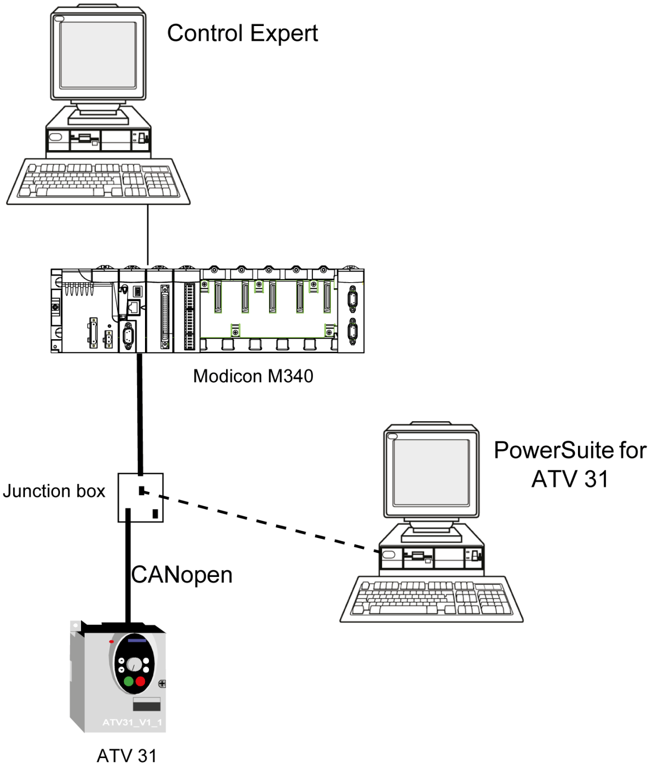
Overview

The proposed architecture is simple and designed to assimilate the implementation principles of motion control.

Other equipment can be added to this realistic architecture in order to manage several axes.

Illustration

The following figure shows the architecture used in the application that includes an **ATV 31**.



Software Requirements

Overview

As regards the software requirements presented in the quick start guide *(see page 11)*, PowerSuite is used for configuring and tuning the **ATV 31**.

Powersuite for **Lexium 05** enables tuning of the axis and guarantees a simple method for configuring the parameters of a **Lexium 05** servodrive.

PowerSuite for **ATV 31** does the same, but for an **ATV 31** servodrive.

It is possible to configure certain parameters withlout using PowerSuite by using the **ATV 31** front panel, user interface *(see page 147)*.

Versions

The following table lists the hardware and software versions used in the architecture *(see page 137)*, enabling the use of MFBs in Control Expert.

Hardware	Earliest version of software	Version of firmware
Modicon M340	Unity Pro V4.0	-
ATV 31	PowerSuite for **ATV 31** V2.00	V1.7 : Entry existing on Unity V3.1 + new MFB profile for V4.0

NOTE: ATV31 V1.7 is compatible with V1.2 functions.

Hardware Requirements

References of the Hardware Used

The following table lists the hardware used in the architecture *(see page 137)*, enabling implementation of **ATV 31** MFBs in Control Expert.

Hardware	Reference
Modicon M340 PLC	**BMX P34 2030**
Modicon M340 power supply	**BMX CPS 2000**
Modicon M340 rack	**BMX XBP 0800**
CANopen junction box between the **Modicon M340** and **ATV 31** servodrive	**VW3CANTAP2**
PC connection kit	**VW3A8106**
ATV 31 servodrive	**ATV31H037M2**

NOTE: The terminating resistor is integrated in the junction box and must be ON.

Section 11.2
CANopen Bus Configuration ATV 31

Configuration of the CANopen Slave (ATV 31) on CANopen bus

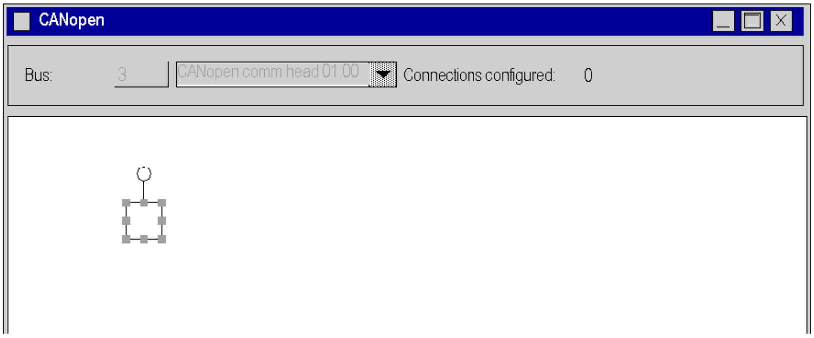
Overview

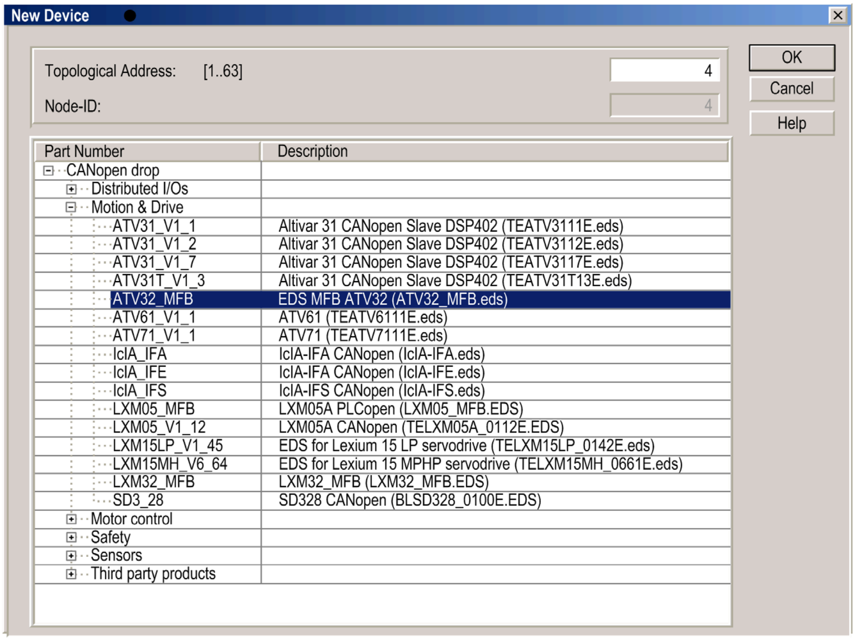
The implementation methodology for a CANopen bus using Modicon M340 is to:

- configure *(see page 29)* the CANopen port of the CPU,
- declarate the slave chosen from the hardware catalog (see paragraph bellow),
- configure the slave,
- enable the configuration using Control Expert,
- check *(see page 34)* the CANopen bus in the Project browser.

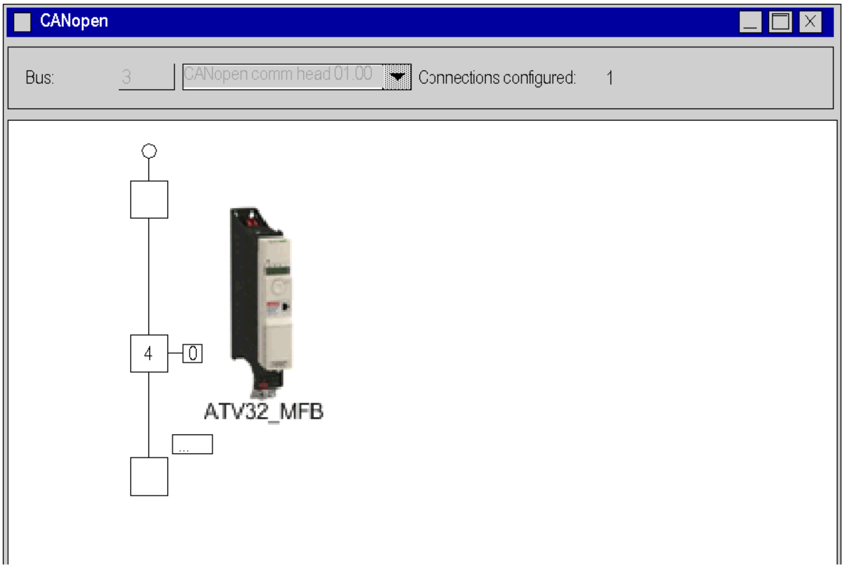
How to Configure the CANopen Slave

This table describes the procedure to configure the CANopen slave.

Step	Action
1	In the Control Expert **Project Browser**, fully expand the **Configuration** directory and then double-click on **CANopen**. **Result:** The CANopen window appears:  CANopen Bus: 3 CANopen comm head 01.00 Connections configured: 0

<table>
<tr><th>Step</th><th>Action</th></tr>
<tr><td>2</td><td>Select **Edit** → **New device**.
Result: The New Device window appears:


Topological Address: [1..63] 4
Node-ID: 4
<table>
<tr><th>Part Number</th><th>Description</th></tr>
<tr><td>CANopen drop</td><td></td></tr>
<tr><td>Distributed I/Os</td><td></td></tr>
<tr><td>Motion &amp; Drive</td><td></td></tr>
<tr><td>ATV31_V1_1</td><td>Altivar 31 CANopen Slave DSP402 (TEATV3111E.eds)</td></tr>
<tr><td>ATV31_V1_2</td><td>Altivar 31 CANopen Slave DSP402 (TEATV3112E.eds)</td></tr>
<tr><td>ATV31_V1_7</td><td>Altivar 31 CANopen Slave DSP402 (TEATV3117E.eds)</td></tr>
<tr><td>ATV31T_V1_3</td><td>Altivar 31 CANopen Slave DSP402 (TEATV31T13E.eds)</td></tr>
<tr><td>ATV32_MFB</td><td>EDS MFB ATV32 (ATV32_MFB.eds)</td></tr>
<tr><td>ATV61_V1_1</td><td>ATV61 (TEATV6111E.eds)</td></tr>
<tr><td>ATV71_V1_1</td><td>ATV71 (TEATV7111E.eds)</td></tr>
<tr><td>IclA_IFA</td><td>IclA-IFA CANopen (IclA-IFA.eds)</td></tr>
<tr><td>IclA_IFE</td><td>IclA-IFA CANopen (IclA-IFE.eds)</td></tr>
<tr><td>IclA_IFS</td><td>IclA-IFS CANopen (IclA-IFS.eds)</td></tr>
<tr><td>LXM05_MFB</td><td>LXM05A PLCopen (LXM05_MFB.EDS)</td></tr>
<tr><td>LXM05_V1_12</td><td>LXM05A CANopen (TELXM05A_0112E.EDS)</td></tr>
<tr><td>LXM15LP_V1_45</td><td>EDS for Lexium 15 LP servodrive (TELXM15LP_0142E.eds)</td></tr>
<tr><td>LXM15MH_V6_64</td><td>EDS for Lexium 15 MPHP servodrive (TELXM15MH_0661E.eds)</td></tr>
<tr><td>LXM32_MFB</td><td>LXM32_MFB (LXM32_MFB.EDS)</td></tr>
<tr><td>SD3_28</td><td>SD328 CANopen (BLSD328_0100E.EDS)</td></tr>
<tr><td>Motor control</td><td></td></tr>
<tr><td>Safety</td><td></td></tr>
<tr><td>Sensors</td><td></td></tr>
<tr><td>Third party products</td><td></td></tr>
</table></td></tr>
<tr><td>3</td><td>Set 4 in Topological Address.
For the slave device choose ATV31_V1_2.</td></tr>
</table>

Step	Action
4	Click on OK to confirm the choice. **Result:** The CANopen window appears with the new device selected:  CANopen Bus: 3 CANopen comm head 01.00 Connections configured: 1
5	Select **Edit → Open module**. If MFB has not already been selected, choose it in the Function area.
6	You will be asked to validate your modifications when closing the Device and CANopen windows.

Section 11.3
Configuring the ATV 31

Aim of this Section

This section describes the basic servodrive configurations using PowerSuite for **ATV 31** and the servodrive's front panel user interface.

What Is in This Section?

This section contains the following topics:

Configuring the ATV 31 in PowerSuite

Overview

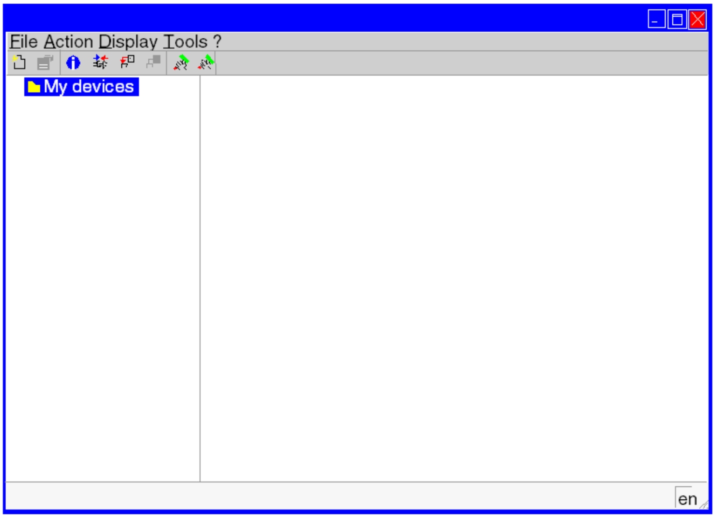
With PowerSuite, users can define installed device bases, and describe their associated configurations and communication settings.

PowerSuite then gives access to a group of actions for editing or transferring the configurations and for connecting to the devices.

PowerSuite’s navigation principle associates a configuration interface with each device type, making it possible to control, tune and monitor them.

Connecting to the ATV 31

This table describes the procedure for connecting to the **ATV 31**:

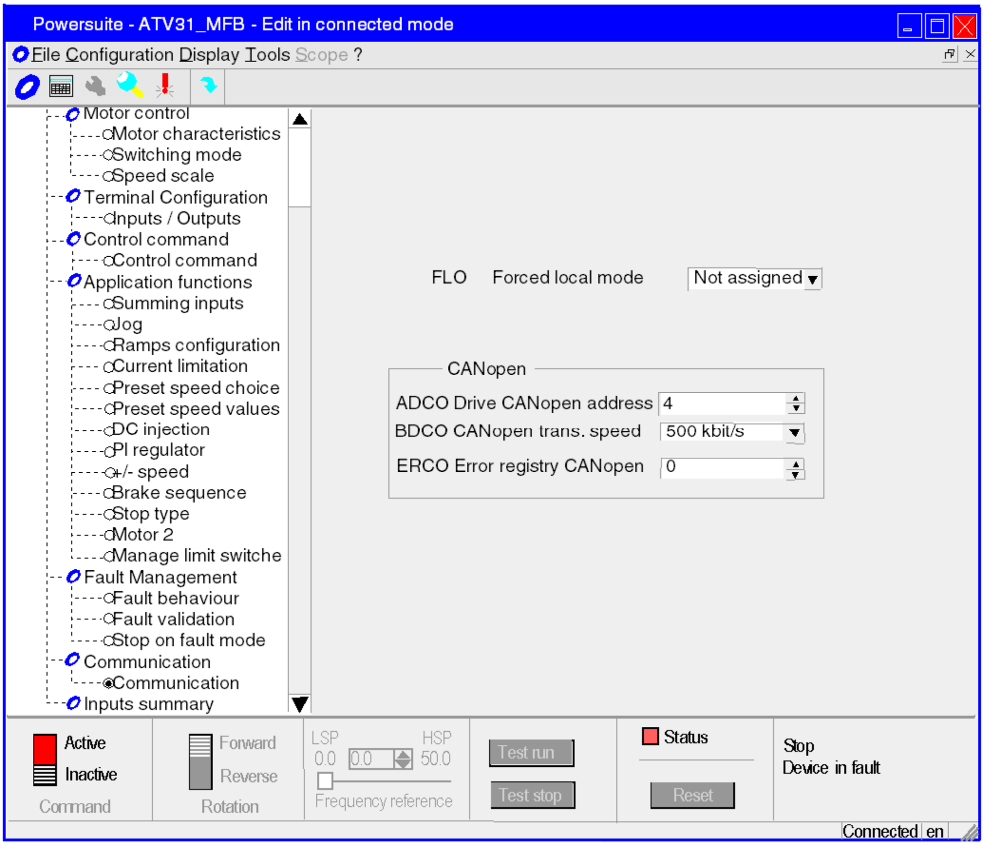
Step	Action
1	Connect your PC, on which PowerSuite for **ATV 31** is installed, to the **RJ45** connector on the servodrive to be configured.
2	Start PowerSuite for **ATV 31**, **Result:** the following start-up screen is displayed: 
3	Choose **Action** and then **Connect**. **Result:** a text box is displayed.
4	Type a project name (ATV31_MFB) and then click on **OK**. **Result:** a transfer confirmation window is displayed.
5	Press **Alt F** to start transferring data from the servodrive to the connected work station.

Basic ATV 31 Configuration

This table describes the procedure for entering basic settings:

<table>
<tr><th>Step</th><th>Action</th></tr>
<tr><td>1</td><td>

Following a connection and transfer of the device’s configurations, PowerSuite displays a configuration screen in a new window that gives access to device control, tuning and monitoring functions.
Use the command **Display → Configuration**.
In the tree structure displayed, choose **Communication** in the *Communication* directory.
Result: the following window is displayed:



Powersuite - ATV31_MFB - Edit in connected mode

FLO Forced local mode Not assigned

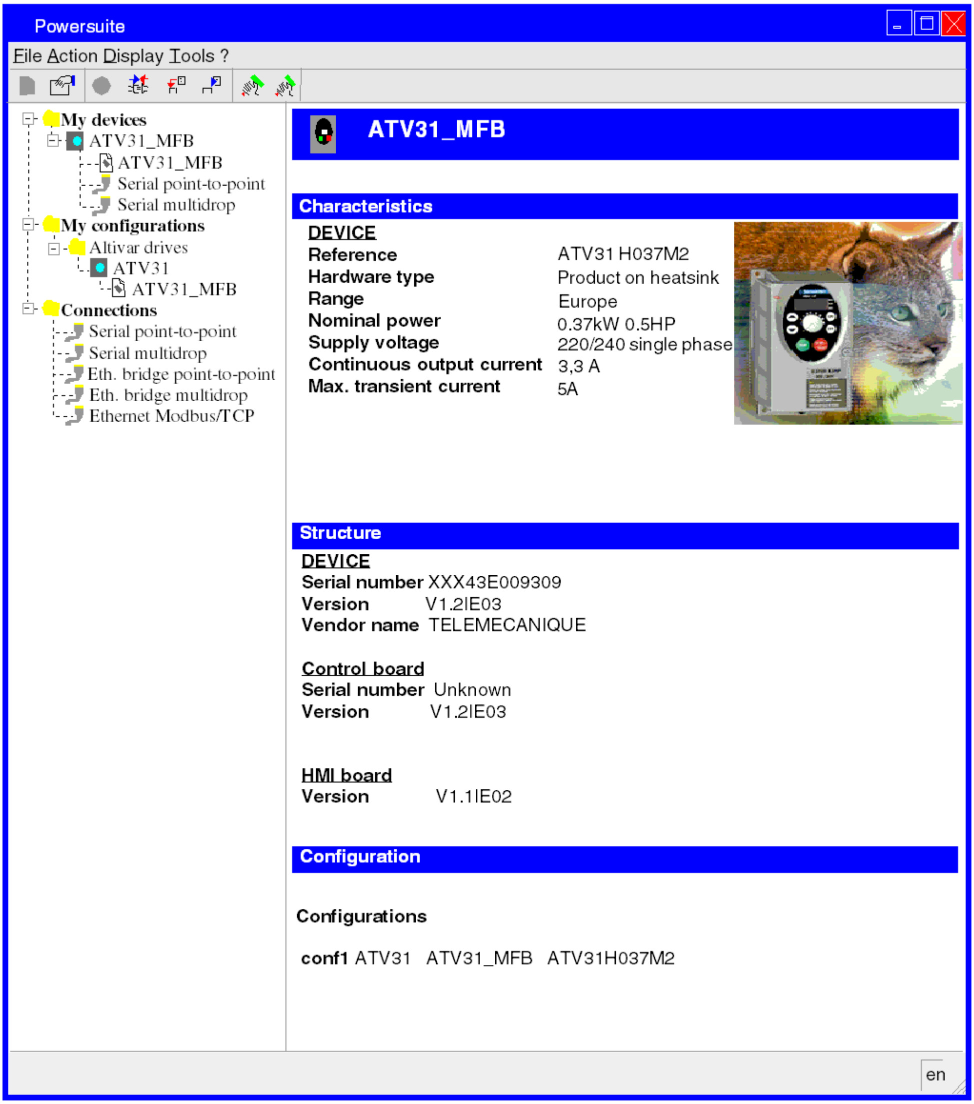
CANopen

ADCO Drive CANopen address 4

BDCO CANopen trans. speed 500 kbit/s

ERCO Error registry CANopen 0

</td></tr>
<tr><td>2</td><td>In the <b>ADCO</b> line, the CANopen address must be set to 4.</td></tr>
<tr><td>3</td><td>In the <b>BDCO</b> line, the CANopen bus speed must be set to 500.</td></tr>
</table>

Step	Action
4	Close the window to disconnect. **Note:** it is possible to adjust the servodrive’s settings with the same procedure. **Result:** the following screen is displayed, showing the data saved locally: 

ATV31_MFB

Characteristics

DEVICE

Reference ATV31 H037M2
Hardware type Product on heatsink
Range Europe
Nominal power 0.37kW 0.5HP
Supply voltage 220/240 single phase
Continuous output current 3,3 A
Max. transient current 5A

Structure

DEVICE
Serial number XXX43E009309
Version V1.2IE03
Vendor name TELEMECANIQUE

Control board
Serial number Unknown
Version V1.2IE03

HMI board
Version V1.1IE02

Configuration

Configurations

conf1 ATV31 ATV31_MFB ATV31H037M2

Configuring the ATV 31 with the User Interface

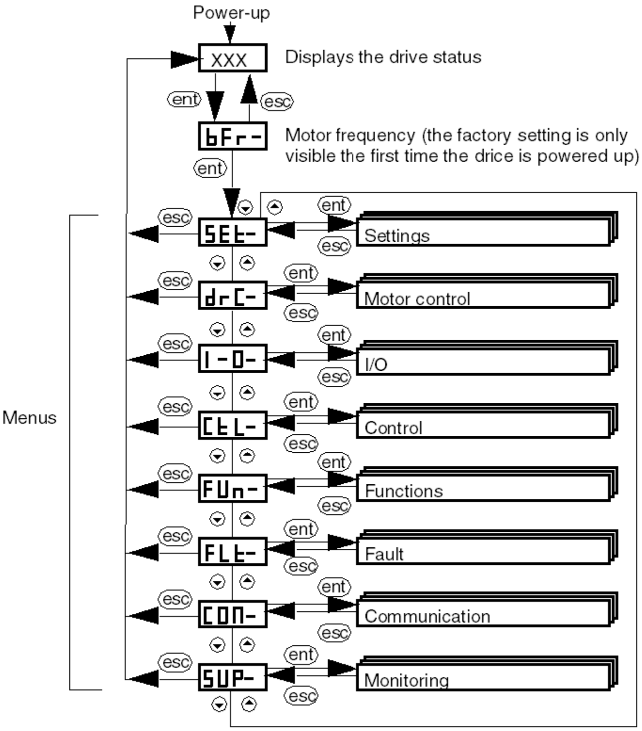
Overview

A user interface is integrated in the **ATV 31**. With this interface, you can:

- put the device online
- configure the device
- carry out a diagnostic

Interface Menu Structure

The following graphic presents an overview of access to the interface’s main menus:



Basic Settings

The following table describes the procedure for entering basic settings (CANopen address and speed) with the interface.

Step	Action
1	Press the **ENT** button on the interface. **Result:** the **SET** (Setting) menu is displayed on the interface’s status indicator.
2	Press the ▼ button several times to access the **COM** menu. **Result:** the **COM** (Communication) menu is displayed on the interface’s status indicator.
3	Press the **ENT** button on the interface. **Result:** the **COAD** (CANopen Address) submenu is displayed on the interface’s status indicator.
4	Press **ENT** again. **Result:** a value corresponding to the device’s CANopen address is displayed.
5	Press the ▼ button to decrease, or the ▲ button to increase the CANopen address value. Press **ENT** when the desired CANopen address is displayed (4). **Result:** the value is confirmed and the **COAD** (CANopen Address) submenu is displayed again.
6	Press the ▼ button to access the **COBD** (CANopen Baud) submenu. Press **ENT**. **Result:** a value corresponding to the device’s CANopen speed is displayed.
7	Press the ▼ button to increase, or the ▲ button to decrease the CANopen baud rate value. Press **ENT** when the desired CANopen speed is displayed (500). **Result:** the value is confirmed and the **COBD** (CANopen Baud) submenu is displayed again.
8	Press **ESC** several times to return to the main display (**RDY** by default).

Section 11.4
Tuning the ATV 31

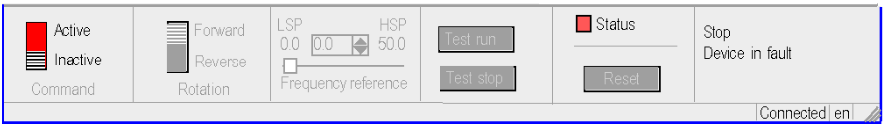
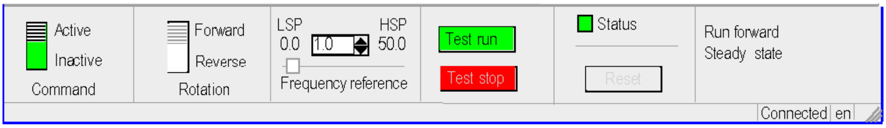
Tuning the ATV 31 with PowerSuite

In Advance

We recommend tuning the axis kinematic before the program automatically starts it.

Tuning Example

The following table gives an example of kinematic tuning:

Step	Action
1	Connect *(see page 144)* to the **ATV 31**.
2	After a connection and transfer of the device’s configurations, PowerSuite opens a new window with the configuration screen, which gives access to device control, tuning and monitoring functions. The following figure shows part of the new window. This lower window provides access to **ATV 31** command functions:  Active Inactive Command Forward Reverse Rotation LSP HSP 0.0 0.0 50.0 Frequency reference Test run Test stop Status Reset Stop Device in fault Connected en
3	Place the **Command** zone cursor on **Active**.
4	Click the **Reset** button to clear any problems (if status is red).
5	Enter the value 1 in the **Frequency reference** zone.
6	Click the **Test Run** button. **Result:** the motor runs and the sub-window is animated:  Active Inactive Command Forward Reverse Rotation LSP HSP 0.0 1.0 50.0 Frequency reference Test run Test stop Status Reset Run forward Steady state Connected en
7	Place the **Command** zone cursor on **Inactive** once tuning has been finalized.

Chapter 12
ATV 32 Implementation for Motion Function Blocks

Aim of this Chapter

This chapter presents the implementation of an ATV 32 servodrive according to the methodology *(see page 17)* described in the quick start guide *(see page 11)* with a Lexium 05. It only details the differences and actions for an ATV 32.

What Is in This Chapter?

This chapter contains the following sections:

Section 12.1
Adapting the Application to the ATV 32

Aim of this Section

This section presents adaptation of the application to the **ATV 32** with an architecture, and hardware and software requirements.

What Is in This Section?

This section contains the following topics:

Application Architecture with an ATV 32

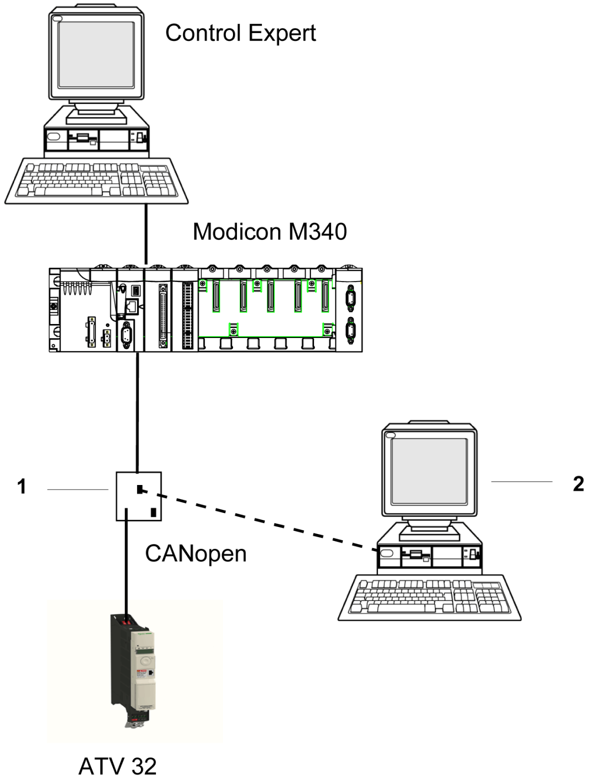
Overview

The proposed architecture is simple and designed to assimilate the implementation principles of motion control.

Other equipment can be added to this architecture to manage several axes.

Illustration

The following figure shows the architecture used in the application that includes an **ATV 32**.



1 Junction box
2 SoMove

Software Requirements

Overview

As regards the software requirements presented in the quick start guide *(see page 11)*, SoMove is used for configuring and tuning the **ATV 32**.

PowerSuite for **Lexium 05** enables a simple method for configuring the parameters of a **Lexium 05** servodrive.

SoMove does the same for an **ATV 32** servodrive.

Without using SoMove, it is possible for certain parameters to be configured by using the **ATV 32** front panel, the user interface *(see page 163)*.

NOTE: ATV 32 servodrive does not support the torque operating mode.

Versions

The following table lists the hardware and software versions used in the architecture *(see page 153)*, enabling the use of MFBs in Control Expert:

Hardware	Earliest version of software	Version of firmware
Modicon M340	Unity Pro V6.0 or higher	-
ATV 32	SoMove	V1.2

Hardware Requirements

References of the Hardware Used

The following table lists the hardware used in the architecture *(see page 153)*, enabling implementation of **ATV 32** MFBs in Control Expert:

Hardware	Reference
Modicon M340 PLC	**BMX P34 2030**
Modicon M340 power supply	**BMX CPS 2000**
Modicon M340 rack	**BMX XBP 0800**
CANopen junction box between the **Modicon M340** and **ATV 32** servodrive	**VW3CANTAP2**
PC connection kit	**VW3A8106**
ATV 32 servodrive	**ATV32 ••••••**

NOTE: The terminating resistor is integrated in the junction box and must be ON.

Section 12.2
CANopen Bus Configuration ATV 32

Configuration of the CANopen Slave (ATV 32) on CANopen Bus

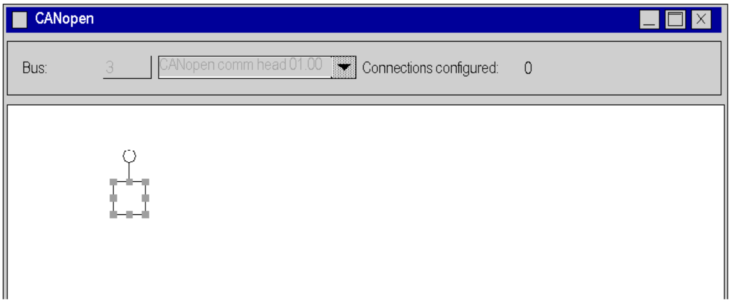
Overview

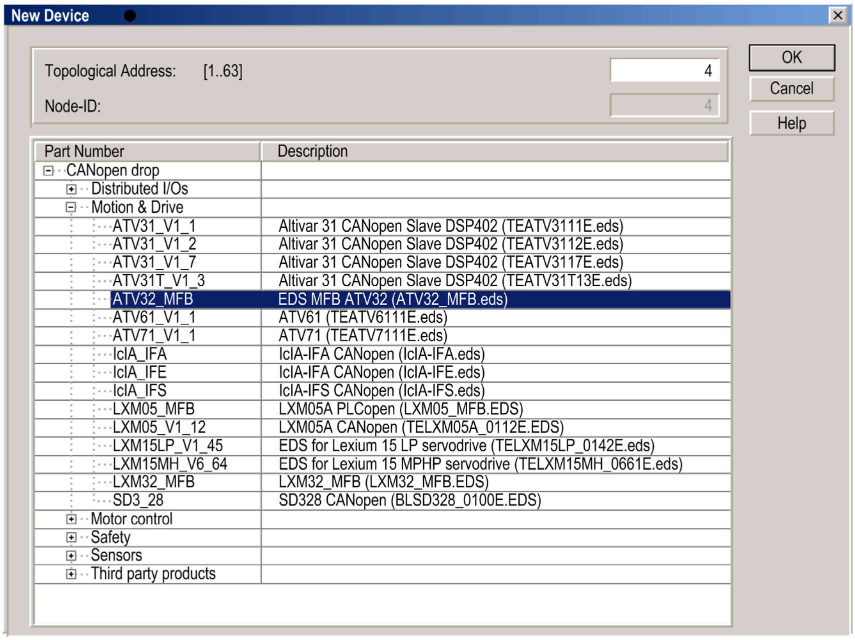
The implementation methodology for a CANopen bus using Modicon M340 is to:

- configure *(see page 29)* the CANopen port of the CPU
- declare the slave chosen from the hardware catalog (see paragraph bellow)
- configure the slave
- enable the configuration using Control Expert
- check *(see page 34)* the CANopen bus in the Project browser

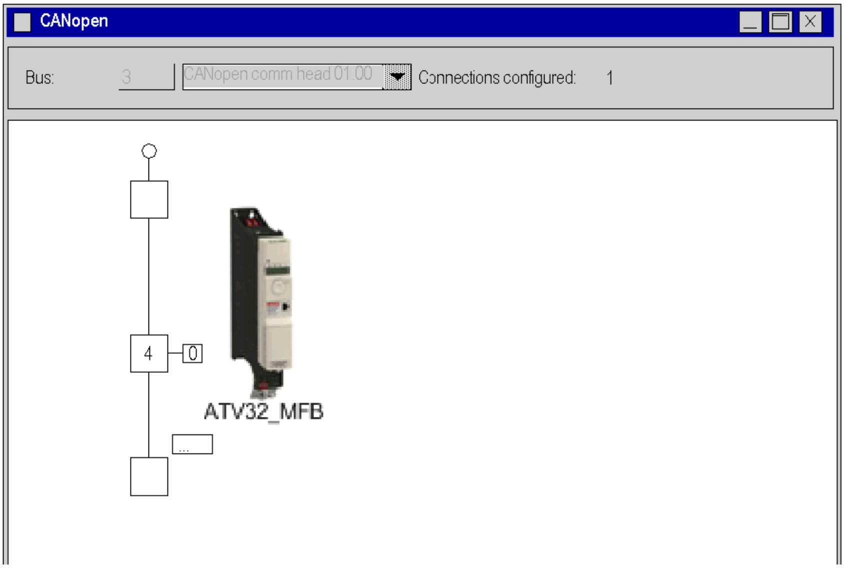
How to Configure the CANopen Slave

This table describes the procedure to configure the CANopen slave:

Step	Action
1	In the Control Expert **Project Browser**, fully expand the **Configuration** directory and then double-click on **CANopen**. **Result:** The CANopen window appears:  CANopen Bus: 3 CANopen comm head 01.00 Connections configured: 0

<table>
<tr><th>Step</th><th>Action</th></tr>
<tr><td>2</td><td>Select <b>Edit</b> → <b>New device</b>.
<b>Result:</b> The New Device window appears:

New Device
Topological Address: [1..63] 4
Node-ID: 4

<table>
<tr><th>Part Number</th><th>Description</th></tr>
<tr><td>CANopen drop</td><td></td></tr>
<tr><td>Distributed I/Os</td><td></td></tr>
<tr><td>Motion &amp; Drive</td><td></td></tr>
<tr><td>ATV31_V1_1</td><td>Altivar 31 CANopen Slave DSP402 (TEATV3111E.eds)</td></tr>
<tr><td>ATV31_V1_2</td><td>Altivar 31 CANopen Slave DSP402 (TEATV3112E.eds)</td></tr>
<tr><td>ATV31_V1_7</td><td>Altivar 31 CANopen Slave DSP402 (TEATV3117E.eds)</td></tr>
<tr><td>ATV31T_V1_3</td><td>Altivar 31 CANopen Slave DSP402 (TEATV31T13E.eds)</td></tr>
<tr><td>ATV32_MFB</td><td>EDS MFB ATV32 (ATV32_MFB.eds)</td></tr>
<tr><td>ATV61_V1_1</td><td>ATV61 (TEATV6111E.eds)</td></tr>
<tr><td>ATV71_V1_1</td><td>ATV71 (TEATV7111E.eds)</td></tr>
<tr><td>IclA_IFA</td><td>IclA-IFA CANopen (IclA-IFA.eds)</td></tr>
<tr><td>IclA_IFE</td><td>IclA-IFA CANopen (IclA-IFE.eds)</td></tr>
<tr><td>IclA_IFS</td><td>IclA-IFS CANopen (IclA-IFS.eds)</td></tr>
<tr><td>LXM05_MFB</td><td>LXM05A PLCopen (LXM05_MFB.EDS)</td></tr>
<tr><td>LXM05_V1_12</td><td>LXM05A CANopen (TELXM05A_0112E.EDS)</td></tr>
<tr><td>LXM15LP_V1_45</td><td>EDS for Lexium 15 LP servodrive (TELXM15LP_0142E.eds)</td></tr>
<tr><td>LXM15MH_V6_64</td><td>EDS for Lexium 15 MPHP servodrive (TELXM15MH_0661E.eds)</td></tr>
<tr><td>LXM32_MFB</td><td>LXM32_MFB (LXM32_MFB.EDS)</td></tr>
<tr><td>SD3_28</td><td>SD328 CANopen (BLSD328_0100E.EDS)</td></tr>
<tr><td>Motor control</td><td></td></tr>
<tr><td>Safety</td><td></td></tr>
<tr><td>Sensors</td><td></td></tr>
<tr><td>Third party products</td><td></td></tr>
</table>
</td></tr>
<tr><td>3</td><td>Set 4 in Topological Address.
For the slave device choose ATV32_V1_2.</td></tr>
</table>

Step	Action
4	Click on OK to confirm the choice. **Result:** The CANopen window appears with the new device selected:  Bus: 3 CANopen comm head 01.00 Connections configured: 1
5	Select **Edit** → **Open module**. If MFB has not already been selected, choose it in the Function area.
6	Validate your modifications when closing the Device and CANopen windows.

Section 12.3
Configuring the ATV 32

Aim of this Section

This section describes the basic servodrive configurations using SoMove and the servodrive’s front panel user interface.

What Is in This Section?

This section contains the following topics:

Configuring the ATV 32 with SoMove

Overview

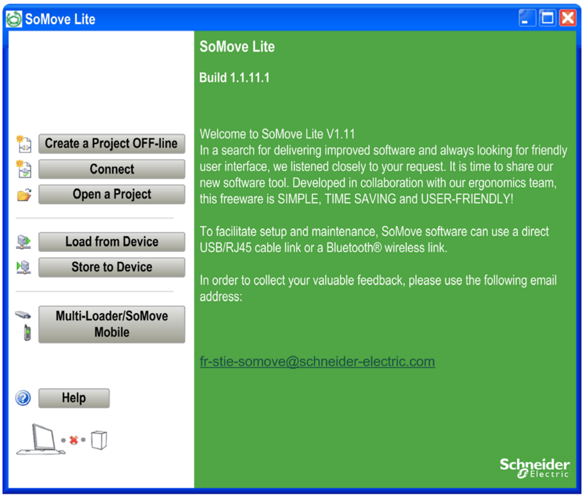
With SoMove, users can define installed device bases and describe their associated configurations and communication settings.

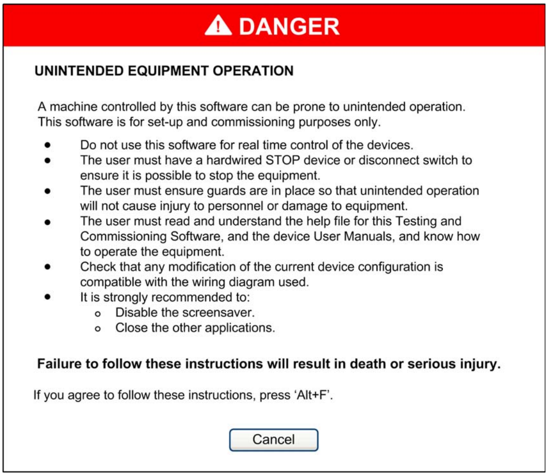
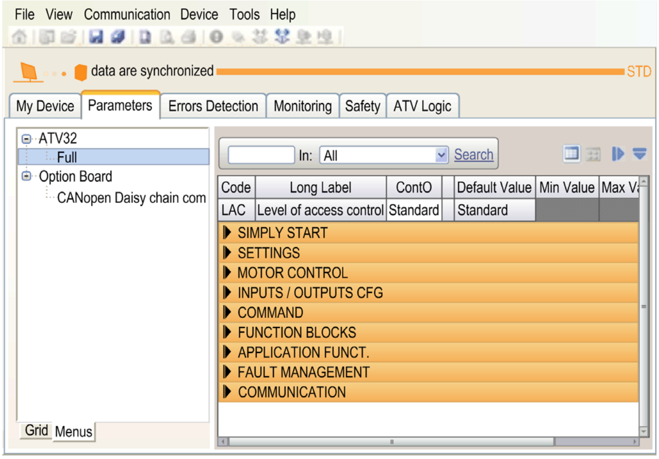
SoMove then gives access to a group of actions for editing or transferring the configurations and for connecting to the devices.

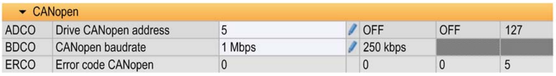
The SoMove navigation principle associates a configuration interface with each device type making it possible to control, tune and monitor them.

Connecting to the ATV 32

This table describes the procedure for connecting to the **ATV 32**:

Step	Action
1	Connect your PC, that has SoMove for ATV 32 installed, to the **RJ45** connector on the servodrive to be configured.
2	Start SoMove. **Result:** The following start-up screen is displayed:  SoMove Lite Create a Project OFF-line Connect Open a Project Load from Device Store to Device Multi-Loader/SoMove Mobile Help SoMove Lite Build 1.1.11.1 Welcome to SoMove Lite V1.11 In a search for delivering improved software and always looking for friendly user interface, we listened closely to your request. It is time to share our new software tool. Developed in collaboration with our ergonomics team, this freeware is SIMPLE, TIME SAVING and USER-FRIENDLY! To facilitate setup and maintenance, SoMove software can use a direct USB/RJ45 cable link or a Bluetooth® wireless link. In order to collect your valuable feedback, please use the following email address: fr-stie-somove@schneider-electric.com

Step	Action
3	Choose **Connect**. **Result:** The following screen is displayed:  ⚠ DANGER **UNINTENDED EQUIPMENT OPERATION** A machine controlled by this software can be prone to unintended operation. This software is for set-up and commissioning purposes only. • Do not use this software for real time control of the devices. • The user must have a hardwired STOP device or disconnect switch to ensure it is possible to stop the equipment. • The user must ensure guards are in place so that unintended operation will not cause injury to personnel or damage to equipment. • The user must read and understand the help file for this Testing and Commissioning Software, and the device User Manuals, and know how to operate the equipment. • Check that any modification of the current device configuration is compatible with the wiring diagram used. • It is strongly recommended to: ○ Disable the screensaver. ○ Close the other applications. **Failure to follow these instructions will result in death or serious injury.** If you agree to follow these instructions, press 'Alt+F'. Cancel
4	If you are agree to follow these instructions, press **Alt+F**. **Result:** The following screen is displayed:  File View Communication Device Tools Help data are synchronized STD My Device Parameters Errors Detection Monitoring Safety ATV Logic ATV32 Full Option Board CANopen Daisy chain com In: All Search Grid Menus Code | Long Label | ContO | Default Value | Min Value | Max V LAC | Level of access control | Standard | Standard SIMPLY START SETTINGS MOTOR CONTROL INPUTS / OUTPUTS CFG COMMAND FUNCTION BLOCKS APPLICATION FUNCT. FAULT MANAGEMENT COMMUNICATION

Step	Action
5	Open the **Communication tab** → **CANopen tab** **Result:** The following screen is displayed:  CANopen ADCO | Drive CANopen address | 5 | OFF | OFF | 127 BDCO | CANopen baudrate | 1 Mbps | 250 kbps | | ERCO | Error code CANopen | 0 | 0 | 0 | 5
6	In the ADCO line, set the CANopen address to 4
7	In the BCDO line, set the CANopen baud rate to 500 kbps.
8	Disconnect your workstation from the servodrive.
9	Save the project using **ATV32_MFB** as the project name.

Configuring the ATV 32 with the User Interface

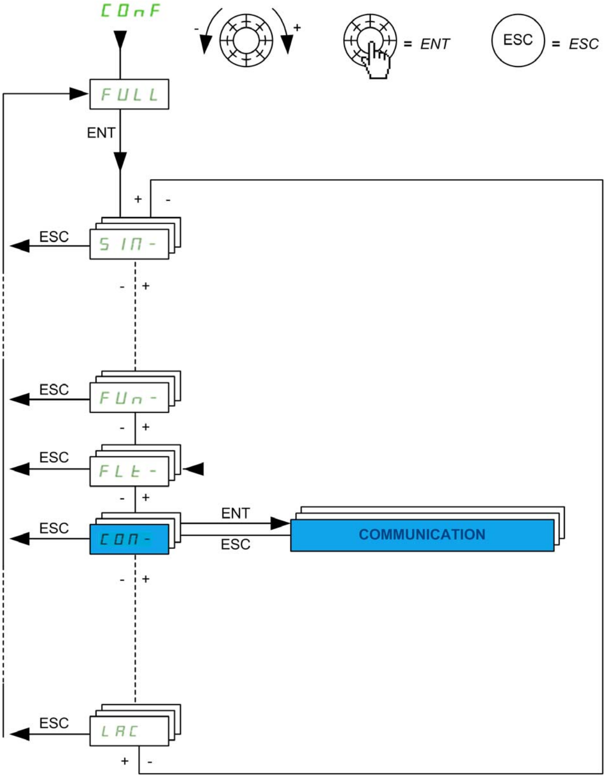
Overview

A user interface is integrated in the **ATV 32**. With this interface, you can:

- put the device online
- configure the device
- adjust the settings
- carry out a diagnostic

Interface Menu Structure

The following graphic shows how to the access to the Configuration menus using the Jog Dial to go to the `COnF` menu:



ENT Enter
ESC Escape

Basic CANopen Settings

This table gives the procedure for entering the basic CANopen address and speed settings through the User Interface:

Step	Action
1	Use the Jog dial to select **COnF**. **Result:** The **COnF** (CANopen configuration) menu is displayed.
2	Press the **ENT** key. **Result:** a rolling list of sub-menus is displayed.
3	Use the Jog dial to select **FULL**. **Result**: The **FULL** (non-preloaded parameters) menu is displayed.
4	Press the **ENT** key. **Result**: A rolling list of sub-menus is displayed.
5	Use the Jog dial to select **COM**. **Result**: The **COM** (Communication) menu is displayed.
6	Press the **ENT** key. **Result**: A rolling list of sub-menus is displayed.
7	Use the Jog dial to select **CnO**. **Result**: The **CnO** (CANopen) menu is displayed.
8	Press the **ENT** key. **Result**: A rolling list of parameters is displayed
9	Use the Jog dial to select **AdCO**. **Result**: The **AdCO** (CANopen address) parameter is displayed.
10	Press the **ENT** key. **Result**: A value corresponding to the default CANopen address is displayed.
11	Use the Jog dial to choose the CANopen address (4). **Result**: The selected CANopen address is displayed.
12	Press the **ENT** key. **Result**: The **AdCO** (CANopen address) parameter is displayed.
13	Use the Jog dial to select **bdCO**. **Result**: The **bdCO** (CANopen speed) parameter is displayed
14	Press the **ENT** key. **Result**: A value corresponding to the default CANopen speed is displayed.
15	Use the Jog dial to choose the CANopen speed (500). **Result**: the selected CANopen speed is displayed.
16	Press the **ENT** key. **Result**: The **bdCO** (CANopen speed) parameter is displayed.
17	Press **ESC** several times to return to the main menu.

Chapter 13
ATV 71 Implementation for Motion Function Blocks

Aim of this Chapter

This chapter presents the implementation of an ATV 71 servodrive according to the methodology *(see page 17)* described in the quick start guide *(see page 11)* with a Lexium 05. It only details the differences and actions for an ATV 71.

What Is in This Chapter?

This chapter contains the following sections:

Section 13.1
Adapting the Application to the ATV 71

Aim of this Section

This section presents adaptation of the application to the **ATV 71** with an architecture, and hardware and software requirements.

What Is in This Section?

This section contains the following topics:

Application Architecture with an ATV 71

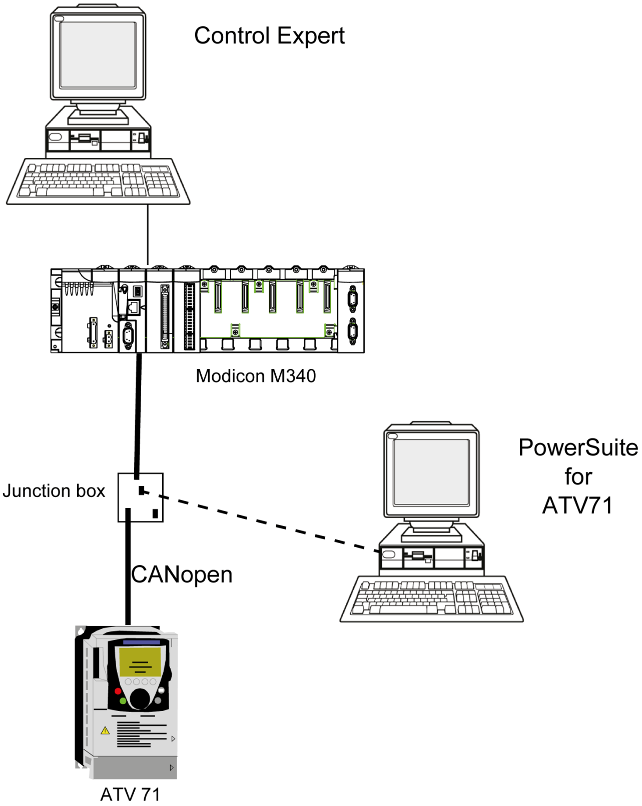
Overview

The proposed architecture is simple and designed to assimilate the implementation principles of motion control.

Other equipment can be added to this realistic architecture in order to manage several axes.

Illustration

The following figure shows the architecture used in the application that includes an **ATV 71**.



Software Requirements

Overview

As regards the software requirements presented in the quick start guide *(see page 11)*, PowerSuite is used for configuring and tuning the **ATV 71**.

PowerSuite for **Lexium 05** enables tuning of the axis and guarantees a simple method for configuring the parameters of a **Lexium 05** servodrive.

PowerSuite for **ATV 71** does the same for an **ATV 71** servodrive.

It is possible to configure certain permaters without PowerSuite cases by using the **ATV 71** front panel, user interface *(see page 178)*.

Versions

The following table lists the hardware and software versions used in the architecture *(see page 167)*, enabling the use of MFBs in Control Expert.

Hardware	Earliest version of software	Version of firmware
Modicon M340	Unity Pro V4.0	-
ATV 71	PowerSuite for **ATV 71** V2.00	Compatible since V1.1, V 1.7 managed by MTM

Hardware Requirements

References of the Hardware Used

The following table lists the hardware used in the architecture *(see page 167)*, enabling implementation of **ATV 71** MFBs in Control Expert.

Hardware	Reference
Modicon M340 PLC	**BMX P34 2030**
Modicon M340 power supply	**BMX CPS 2000**
Modicon M340 rack	**BMX XBP 0800**
CANopen junction box between the **Modicon M340** and **ATV 71** servodrive	**VW3CANTAP2**
RJ45 programming cable with RS485/RS232 adapter between the junction box and servodrive	**ACC2CRAAEF030**
ATV 71 servodrive	**ATV71H075N2Z**

NOTE: The terminating resistor is integrated in the junction box and must be ON.

Section 13.2
CANopen Bus Configuration ATV 71

Configuration of the CANopen Slave (ATV 71) on CANopen bus

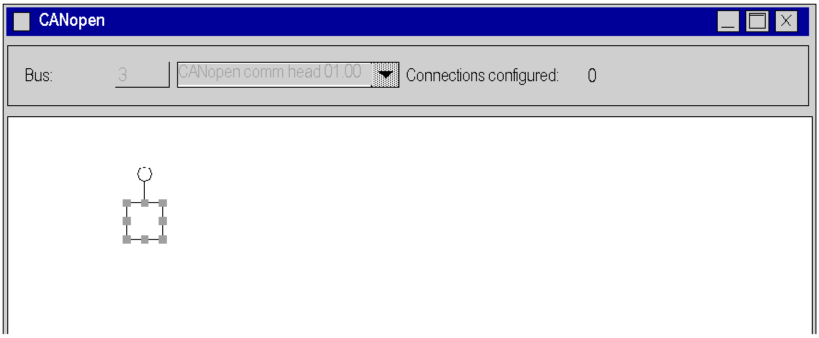
Overview

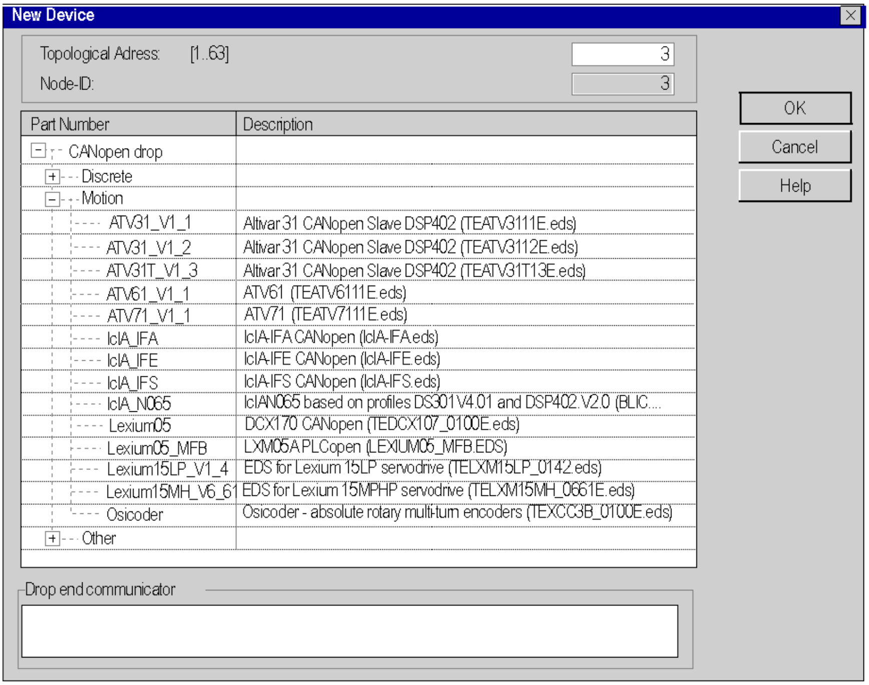
The implementation methodology for a CANopen bus using Modicon M340 is to:

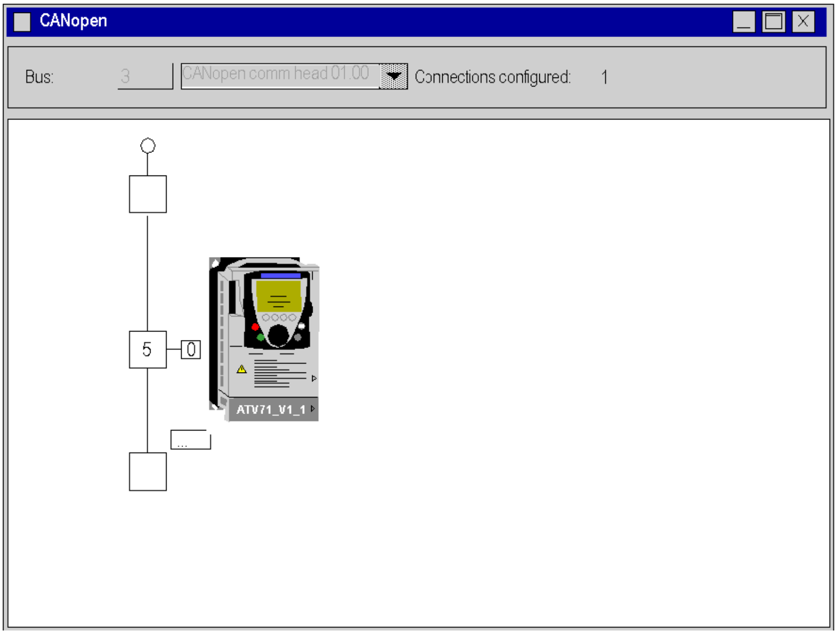
- configure *(see page 29)* the CANopen port of the CPU,
- declarate the slave chosen from the hardware catalog (see paragraph bellow),
- configure the slave,
- enable the configuration using Control Expert,
- check *(see page 34)* the CANopen bus in the Project browser.

How to Configure the CANopen Slave

This table describes the procedure to configure the CANopen slave.

Step	Action
1	In the Control Expert **Project Browser**, fully expand the **Configuration** directory and then double-click on **CANopen**. **Result:** The CANopen window appears:  Bus: 3 CANopen comm head 01.00 Connections configured: 0

<table>
<tr><th>Step</th><th>Action</th></tr>
<tr><td>2</td><td>Select <b>Edit</b> → <b>New device</b>.
<b>Result:</b> The New Device window appears:

New Device
Topological Adress: [1..63] 3
Node-ID: 3
<table>
<tr><th>Part Number</th><th>Description</th></tr>
<tr><td>CANopen drop</td><td></td></tr>
<tr><td>Discrete</td><td></td></tr>
<tr><td>Motion</td><td></td></tr>
<tr><td>ATV31_V1_1</td><td>Altivar 31 CANopen Slave DSP402 (TEATV3111E.eds)</td></tr>
<tr><td>ATV31_V1_2</td><td>Altivar 31 CANopen Slave DSP402 (TEATV3112E.eds)</td></tr>
<tr><td>ATV31T_V1_3</td><td>Altivar 31 CANopen Slave DSP402 (TEATV31T13E.eds)</td></tr>
<tr><td>ATV61_V1_1</td><td>ATV61 (TEATV6111E.eds)</td></tr>
<tr><td>ATV71_V1_1</td><td>ATV71 (TEATV7111E.eds)</td></tr>
<tr><td>IclA_IFA</td><td>IclA-IFA CANopen (IclA-IFA.eds)</td></tr>
<tr><td>IclA_IFE</td><td>IclA-IFE CANopen (IclA-IFE.eds)</td></tr>
<tr><td>IclA_IFS</td><td>IclA-IFS CANopen (IclA-IFS.eds)</td></tr>
<tr><td>IclA_N065</td><td>IclAN065 based on profiles DS301 V4.01 and DSP402 V2.0 (BLIC....</td></tr>
<tr><td>Lexium05</td><td>DCX170 CANopen (TEDCX107_0100E.eds)</td></tr>
<tr><td>Lexium05_MFB</td><td>LXM05A PLCopen (LEXIUM05_MFB.EDS)</td></tr>
<tr><td>Lexium15LP_V1_4</td><td>EDS for Lexium 15LP servodrive (TELXM15LP_0142.eds)</td></tr>
<tr><td>Lexium15MH_V6_61</td><td>EDS for Lexium 15MPHP servodrive (TELXM15MH_0661E.eds)</td></tr>
<tr><td>Osicoder</td><td>Osicoder - absolute rotary multi-turn encoders (TEXCC3B_0100E.eds)</td></tr>
<tr><td>Other</td><td></td></tr>
</table>
Drop end communicator
</td></tr>
<tr><td>3</td><td>Set 5 in Topological Address.
For the slave device choose ATV71_V1_1.</td></tr>
</table>

Step	Action
4	Click on OK to confirm the choice. **Result:** The CANopen window appears with the new device selected:  CANopen Bus: 3 CANopen comm head 01.00 Connections configured: 1
5	Select **Edit → Open module**. If MFB has not already been selected, choose it in the Function area.
6	You will be asked to validate your modifications when closing the Device and CANopen windows.

Section 13.3
Configuring the ATV 71

Aim of this Section

This section describes the basic servodrive configurations using PowerSuite for **ATV 71** and the servodrive’s front panel user interface.

What Is in This Section?

This section contains the following topics:

Configuring the ATV 71 in PowerSuite

Overview

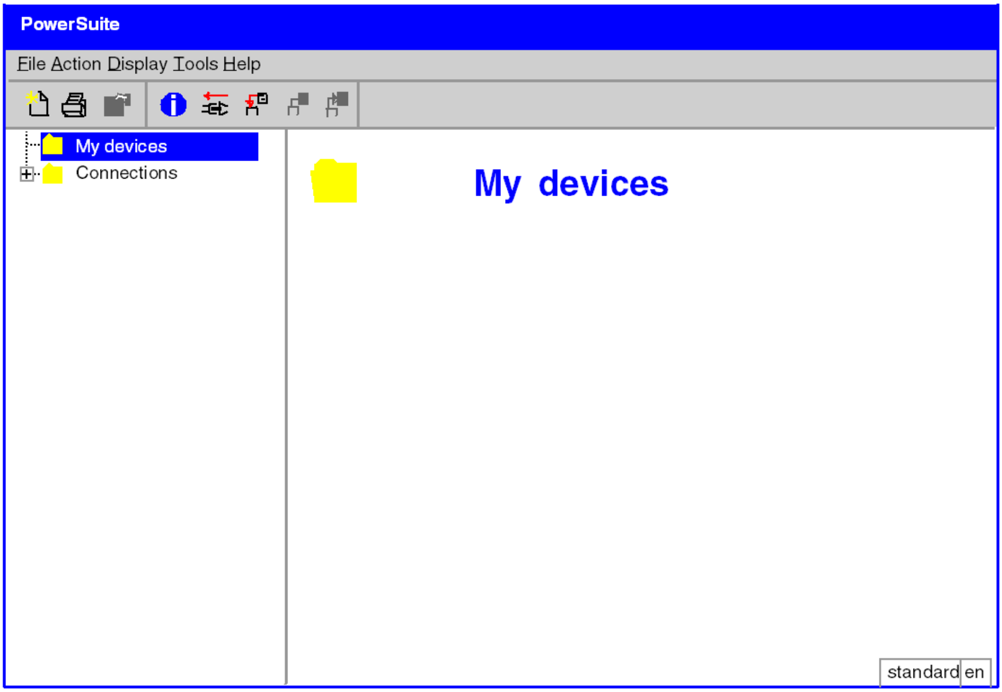
With PowerSuite, users can define installed device bases, and describe their associated configurations and communication settings.

PowerSuite then gives access to a group of actions for editing or transferring the configurations and for connecting to the devices.

PowerSuite's navigation principle associates a configuration interface with each device type, making it possible to control, tune and monitor them.

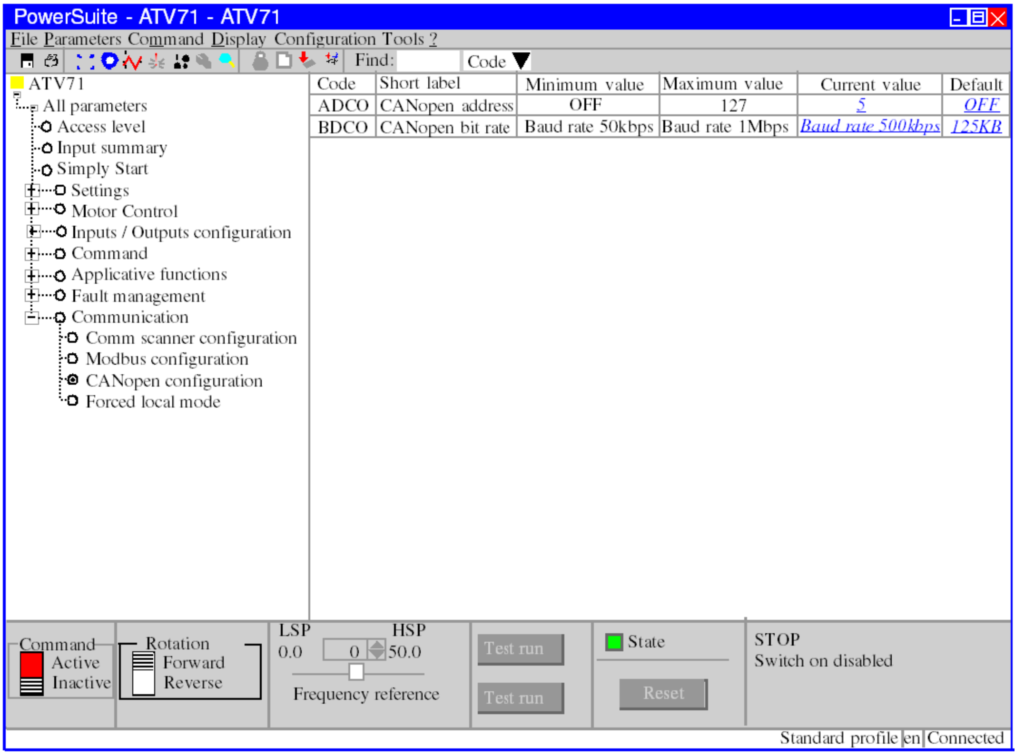
Connecting to the ATV 71

This table describes the procedure for connecting to the **ATV 71**:

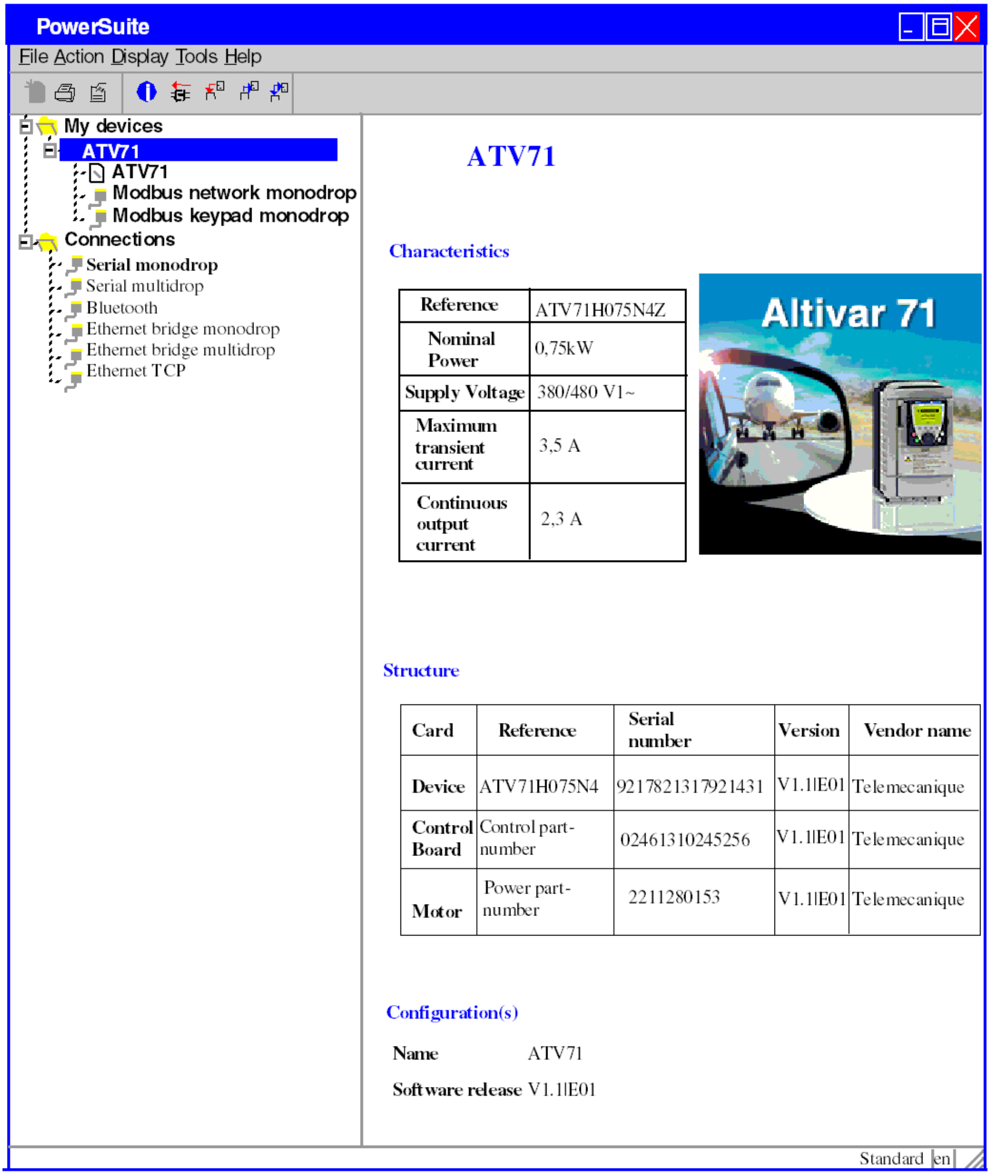
Step	Action
1	Connect your PC, on which PowerSuite for **ATV 71** is installed, to the **RJ45** connector on the servodrive to be configured.
2	Start PowerSuite for **ATV 71**, **Result:** the following start-up screen is displayed:  PowerSuite File Action Display Tools Help My devices Connections My devices standard en
3	Choose **Action** and then **Connect**. **Result:** a text box is displayed.
4	Type a project name (ATV71_MFB) and then click on **OK**. **Result:** a transfer confirmation window is displayed.
5	Press **Alt F** to start transferring data from the servodrive to the connected work station.

Basic ATV 71 Configuration

This table describes the procedure for entering basic settings:

<table>
<tr><th>Step</th><th>Action</th></tr>
<tr><td>1</td><td>Following a connection and transfer of the device’s configurations, PowerSuite displays a configuration screen in a new window that gives access to device control, tuning and monitoring functions.
In the tree structure displayed, choose <b>Communication</b> in the <i>Communication</i> directory.
<b>Result:</b> the following window is displayed:


<table>
<tr><th>Code</th><th>Short label</th><th>Minimum value</th><th>Maximum value</th><th>Current value</th><th>Default</th></tr>
<tr><td>ADCO</td><td>CANopen address</td><td>OFF</td><td>127</td><td>5</td><td>OFF</td></tr>
<tr><td>BDCO</td><td>CANopen bit rate</td><td>Baud rate 50kbps</td><td>Baud rate 1Mbps</td><td>Baud rate 500kbps</td><td>125KB</td></tr>
</table>
</td></tr>
<tr><td>2</td><td>In the <b>ADCO</b> line, the CANopen address must be set to 5.</td></tr>
<tr><td>3</td><td>In the <b>BDCO</b> line, the CANopen bus speed must be set to 500.
<b>Note:</b> it is possible to adjust the servodrive’s settings with the same procedure.</td></tr>
</table>

Step	Action
4	Once the settings have been adjusted, use the command **Configuration → Disconnect** to disconnect. **Result:** the following screen is displayed, showing the data saved locally: 

ATV71

Characteristics

Reference	ATV71H075N4Z
Nominal Power	0,75kW
Supply Voltage	380/480 V1~
Maximum transient current	3,5 A
Continuous output current	2,3 A

Structure

Card	Reference	Serial number	Version	Vendor name
Device	ATV71H075N4	9217821317921431	V1.1IE01	Telemecanique
Control Board	Control part-number	02461310245256	V1.1IE01	Telemecanique
Motor	Power part-number	2211280153	V1.1IE01	Telemecanique

Configuration(s)

Name ATV71

Software release V1.1IE01

Configuring the ATV 71 with the User Interface

Overview

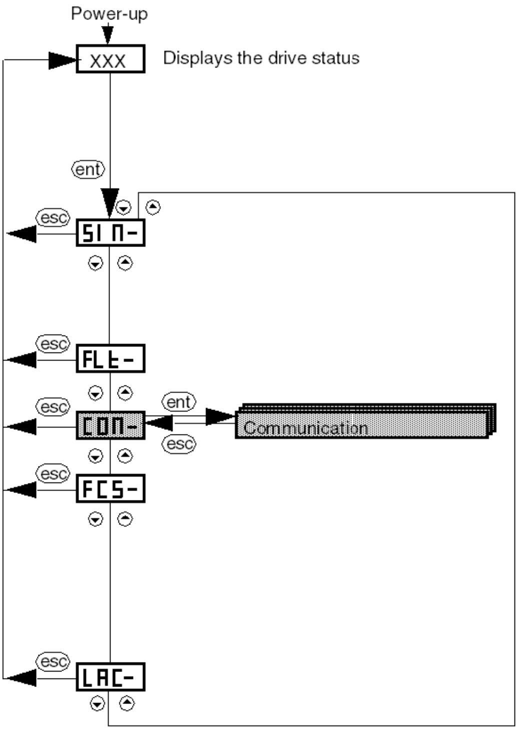
A user interface is integrated in the **ATV 71**. With this interface, you can:

- put the device online
- configure the device
- carry out a diagnostic

NOTE: There is a more user-friendly graphic display terminal, for instance for diagnosing faults.

Interface Menu Structure

The following graphic presents an overview of access to the interface's main menus:



Basic Settings

The following table describes the procedure for entering basic settings (CANopen address and speed) with the interface.

Step	Action
1	Press the **ENT** button on the interface. **Result:** the **SET** (Setting) menu is displayed on the interface’s status indicator.
2	Press the ▼ button several times to access the **COM** menu. **Result:** the **COM** (Communication) menu is displayed on the interface’s status indicator.
3	Press the **ENT** button on the interface. **Result:** the **COAD** (CANopen Address) submenu is displayed on the interface’s status indicator.
4	Press **ENT** again. **Result:** a value corresponding to the device’s CANopen address is displayed.
5	Press the ▼ button to decrease, or the ▲ button to increase the CANopen address value. Press **ENT** when the desired CANopen address is displayed (5). **Result:** the value is confirmed and the **COAD** (CANopen Address) submenu is displayed again.
6	Press the ▼ button to access the **COBD** (CANopen Baud) submenu. Press **ENT**. **Result:** a value corresponding to the device’s CANopen speed is displayed.
7	Press the ▼ button to increase, or the ▲ button to decrease the CANopen baud rate value. Press **ENT** when the desired CANopen speed is displayed (500). **Result:** the value is confirmed and the **COBD** (CANopen Baud) submenu is displayed again.
8	Press **ESC** several times to return to the main display (**RDY** by default).

Section 13.4
Tuning the ATV 71

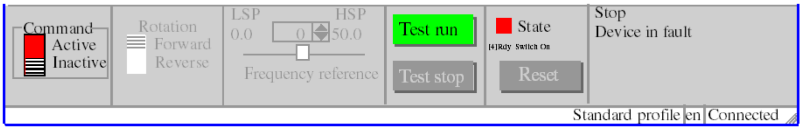
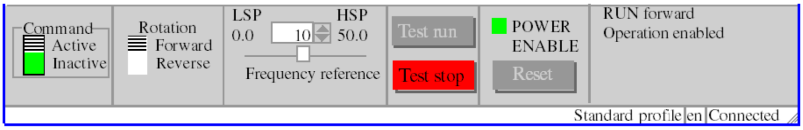
Tuning the ATV 71 with PowerSuite

In Advance

We recommend tuning the axis kinematic before the program automatically starts it.

Tuning Example

The following table gives an example of kinematic tuning:

Step	Action
1	Connect *(see page 175)* to the **ATV 71**.
2	After a connection and transfer of the device’s configurations, PowerSuite opens a new window with the configuration screen, which gives access to device control, tuning and monitoring functions. The following figure shows part of the new window. This lower window provides access to **ATV 71** command functions:  Command Active Inactive Rotation Forward Reverse LSP HSP 0.0 0 50.0 Frequency reference Test run Test stop State [4]Rdy Switch On Reset Stop Device in fault Standard profile en Connected
3	Place the **Command** zone cursor on **Active**.
4	Click the **Reset** button to clear any problems.
5	Enter the value 10 in the **Frequency reference** zone.
6	Click the **Test Run** button. **Result:** the motor runs and the sub-window is animated:  Command Active Inactive Rotation Forward Reverse LSP HSP 0.0 10 50.0 Frequency reference Test run Test stop POWER ENABLE Reset RUN forward Operation enabled Standard profile en Connected
7	Place the **Command** zone cursor on **Inactive** once tuning has been finalized.

Chapter 14
IclA Implementation for Motion Function Blocks

Aim of this Chapter

This chapter presents the implementation of an IclA servodrive according to the methodology *(see page 17)* described in the quick start guide *(see page 11)* with a Lexium 05. It only details the differences and actions for an IclA.

What Is in This Chapter?

This chapter contains the following sections:

Section 14.1
Adapting the Application to the IclA

Aim of this Section

This section presents adaptation of the application to the **IclA** with an architecture, and hardware and software requirements.

What Is in This Section?

This section contains the following topics:

Application Architecture with an IclA

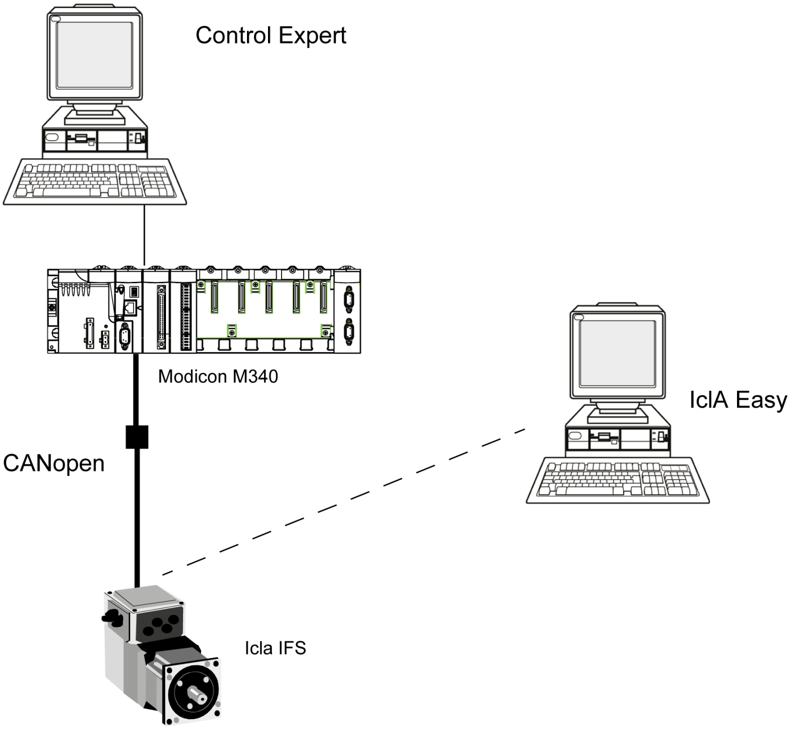
Overview

The proposed architecture is simple and designed to assimilate the implementation principles of motion control.

Other equipment can be added to this realistic architecture in order to manage several axes.

Illustration

The following figure shows the architecture used in the application that includes an **IclA IFS**.



Software Requirements

Overview

As regards the software requirements presented in the quick start guide *(see page 11)*, IclA Easy is used for configuring and tuning the **IclA**.

PowerSuite for **Lexium 05** enables tuning of the axis and guarantees a simple method for configuring the parameters of a **Lexium 05** servodrive.

IclA Easy does the same for an **IclA** servodrive.

It is necessary to configure certain parameters without Icla Easy by using the **IclA** switches *(see page 189)*, since this is the only way to configure these parameters.

Versions

The following table lists the hardware and software versions used in the architecture *(see page 183)*, enabling the use of MFBs in Control Expert.

Hardware	Earliest version of software	Version of firmware
Modicon M340	Unity Pro V4.0	-
IclA	**EasyIclA** V1.104	**IclA IFA** compatible since V1.1007 **IclA IFE** compatible since V1.1007 **IclA IFS** compatible since V1.1007

Hardware Requirements

References of the Hardware Used

The following table lists the hardware used in the architecture *(see page 183)*, enabling implementation of IclA MFBs in Control Expert.

Hardware	Reference
Modicon M340 PLC	BMX P34 2030
Modicon M340 power supply	BMX CPS 2000
Modicon M340 rack	BMX XBP 0800
CANopen SUB-D9-Way female connector (bended at 90° + additional SUB-D9-Way connector to connect a PC on the bus)	TSX CAN KCDF 90TP
CANopen preassembled cordset with moulded female SUB-D9-Way connectors at both end	TSX CAN CADD03
Dongle PCAN PS/2 for IclA Easy (parallel-to-CAN converter)	IPEH-002019
CANopen cable	TSX CAN CA50
IclA servodrive	IFS61/2-CAN-DS/-I-B54/0-001RPP41

NOTE: The terminating resistor is integrated in the IclA and must be ON *(see page 189)*.

Section 14.2
CANopen Bus Configuration IclA

Configuration of the CANopen Slave (IclA) on CANopen bus

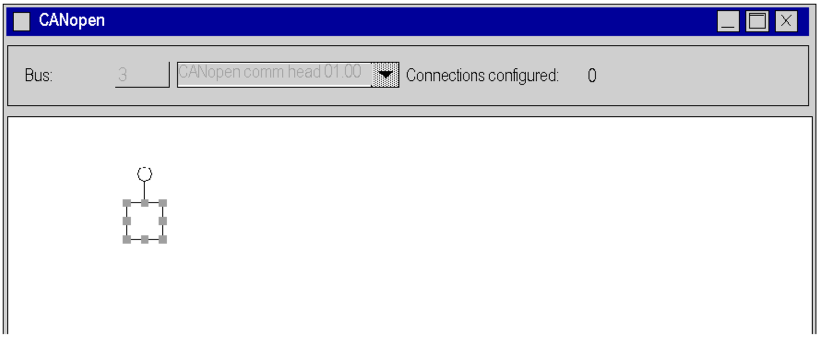
Overview

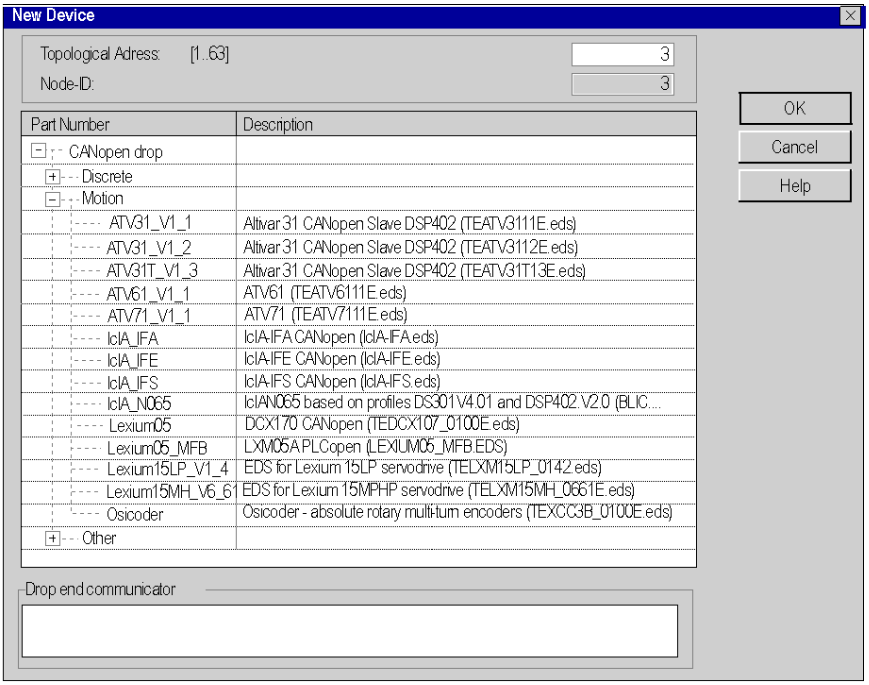
The implementation methodology for a CANopen bus using Modicon M340 is to:

- configure *(see page 29)* the CANopen port of the CPU,
- declarate the slave chosen from the hardware catalog (see paragraph bellow),
- configure the slave,
- enable the configuration using Control Expert,
- check *(see page 34)* the CANopen bus in the Project browser.

How to Configure the CANopen Slave

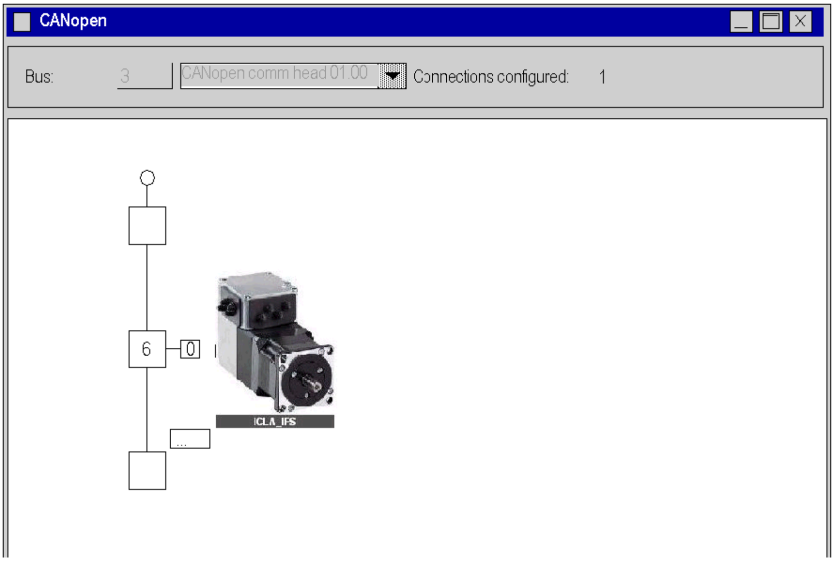
This table describes the procedure to configure the CANopen slave.

Step	Action
1	In the Control Expert **Project Browser**, fully expand the **Configuration** directory and then double-click on **CANopen**. **Result:** The CANopen window appears:  CANopen Bus: 3 CANopen comm head 01.00 Connections configured: 0

<table>
<tr><th>Step</th><th>Action</th></tr>
<tr><td>2</td><td>Select <b>Edit</b> → <b>New device</b>.
<b>Result:</b> The New Device window appears:


Topological Adress: [1..63] 3
Node-ID: 3

<table>
<tr><th>Part Number</th><th>Description</th></tr>
<tr><td>⊟ CANopen drop</td><td></td></tr>
<tr><td>⊞ Discrete</td><td></td></tr>
<tr><td>⊟ Motion</td><td></td></tr>
<tr><td>ATV31_V1_1</td><td>Altivar 31 CANopen Slave DSP402 (TEATV3111E.eds)</td></tr>
<tr><td>ATV31_V1_2</td><td>Altivar 31 CANopen Slave DSP402 (TEATV3112E.eds)</td></tr>
<tr><td>ATV31T_V1_3</td><td>Altivar 31 CANopen Slave DSP402 (TEATV31T13E.eds)</td></tr>
<tr><td>ATV61_V1_1</td><td>ATV61 (TEATV6111E.eds)</td></tr>
<tr><td>ATV71_V1_1</td><td>ATV71 (TEATV7111E.eds)</td></tr>
<tr><td>IclA_IFA</td><td>IclA-IFA CANopen (IclA-IFA.eds)</td></tr>
<tr><td>IclA_IFE</td><td>IclA-IFE CANopen (IclA-IFE.eds)</td></tr>
<tr><td>IclA_IFS</td><td>IclA-IFS CANopen (IclA-IFS.eds)</td></tr>
<tr><td>IclA_N065</td><td>IclAN065 based on profiles DS301 V4.01 and DSP402.V2.0 (BLIC....</td></tr>
<tr><td>Lexium05</td><td>DCX170 CANopen (TEDCX107_0100E.eds)</td></tr>
<tr><td>Lexium05_MFB</td><td>LXM05A PLCopen (LEXIUM05_MFB.EDS)</td></tr>
<tr><td>Lexium15LP_V1_4</td><td>EDS for Lexium 15LP servodrive (TELXM15LP_0142.eds)</td></tr>
<tr><td>Lexium15MH_V6_61</td><td>EDS for Lexium 15MPHP servodrive (TELXM15MH_0661E.eds)</td></tr>
<tr><td>Osicoder</td><td>Osicoder - absolute rotary multi-turn encoders (TEXCC3B_0100E.eds)</td></tr>
<tr><td>⊞ Other</td><td></td></tr>
</table>
</td></tr>
<tr><td>3</td><td>Set 6 in Topological Address.
For the slave device choose IclA_IFS.</td></tr>
</table>

Step	Action
4	Click on OK to confirm the choice. **Result:** The CANopen window appears with the new device selected:  CANopen Bus: 3 CANopen comm head 01.00 Connections configured: 1
5	Select **Edit → Open module**. If MFB has not already been selected, choose it in the Function area.
6	You will be asked to validate your modifications when closing the Device and CANopen windows.

Section 14.3
Configuring the IclA

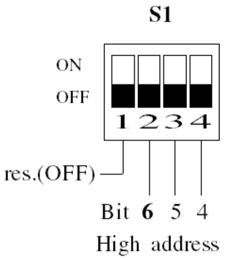
Configuring the IclA with DIP Switches

Overview

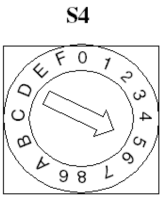
The address and baud rate are set with DIP switches on the **IclA IFX** drive.

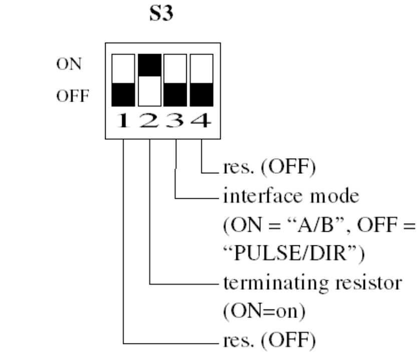
DIP Switches

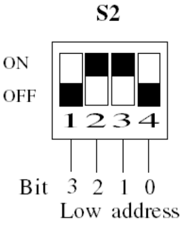
The following graphic presents the DIP switches inside the drive:



Hex	kBaud
0	-
1	50
2	100
3	125
4	250
5	500
6	800
7	1000
8..F	-







Basic Settings

The baud rate is set with the S4 switch on position 5 for a baud rate of 500.

The CANopen address is set with the S1 and S2 switches. Set S2.3 and S2.2 ON for the drive to have address 6. By default, as shown in the graphic above, all the switches on S1 and S2 are set ON except the first switch on S1, which gives address 127.

Set S3.2 ON to activate the terminating resistor.

Section 14.4
Tuning the IclA

Aim of this Section

This section gives an example of tuning the **IclA** with IclA Easy.

What Is in This Section?

This section contains the following topics:

Configuring the IclA in IclA Easy

Overview

With IclA Easy, users can define installed device bases, and describe their associated configurations and communication settings.

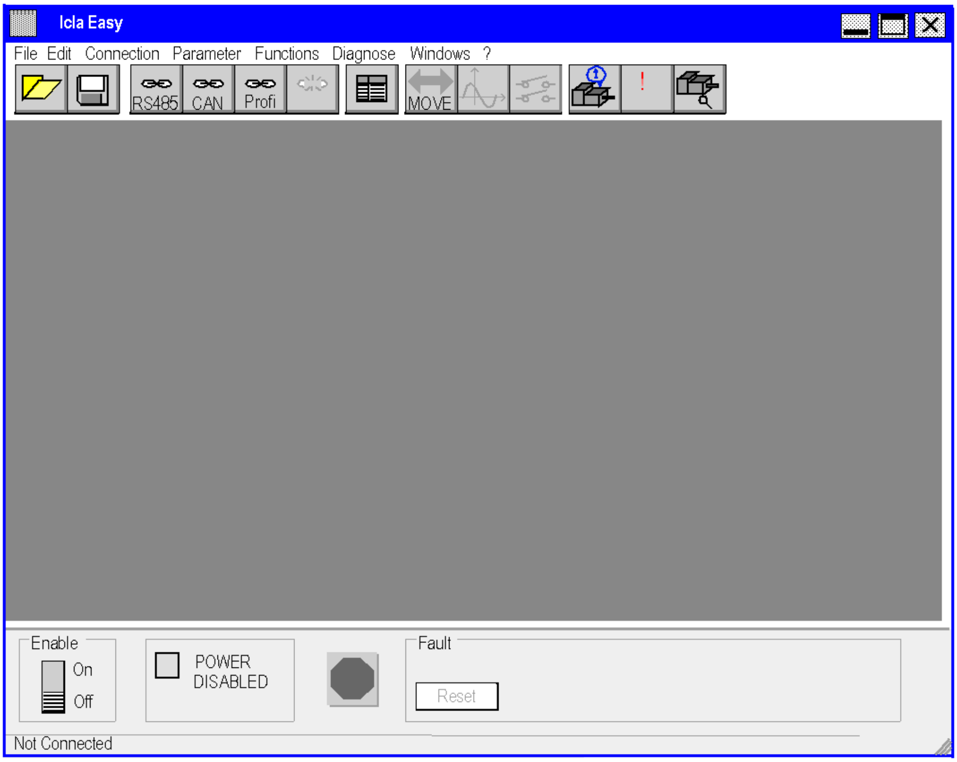
IclA Easy then gives access to a group of actions for editing or transferring the configurations and for connecting to the devices.

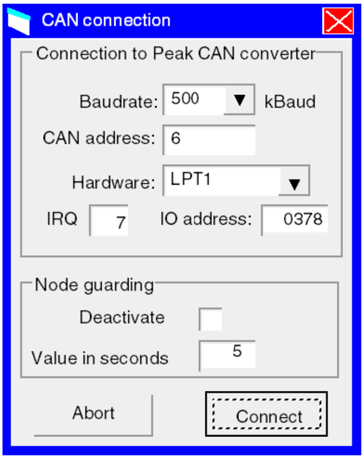
IclA Easy's navigation principle associates a configuration interface with each device type, making it possible to control, tune and monitor them.

NOTE: The required signals, i.e LIMN, LIMP, REF must be wired or deactived by the tuning software.

Connecting to the IclA

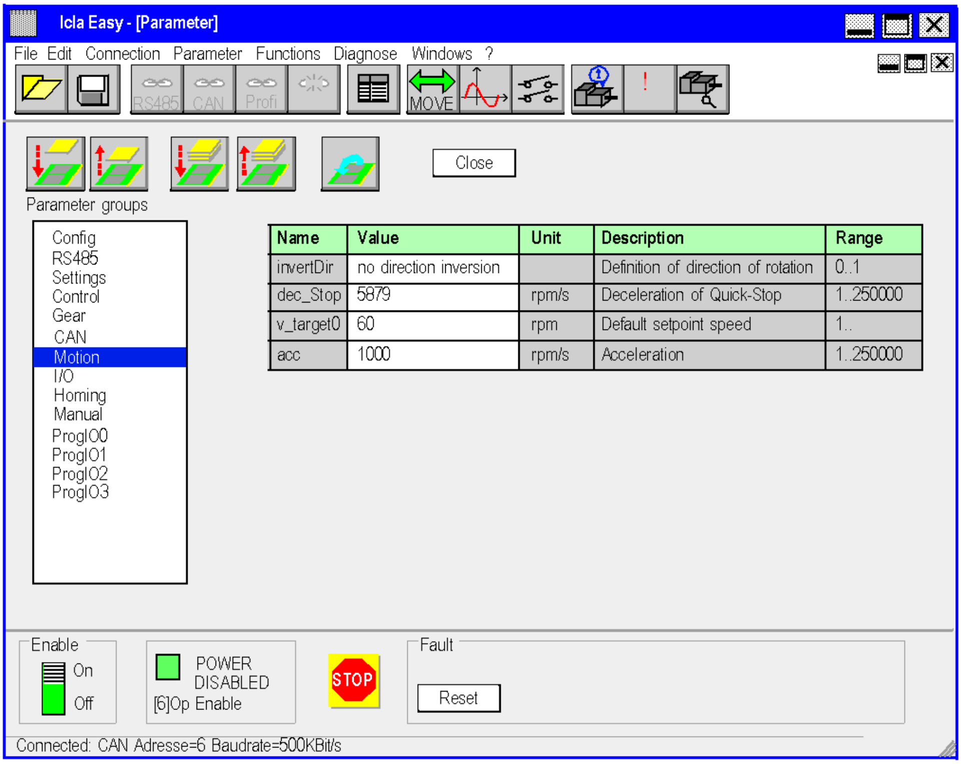
This table describes the procedure for connecting to the **IclA**:

Step	Action
1	Connect your PC, on which IclA Easy is installed, to the Dongle PCAN PS/2 connector on the servodrive to be configured.
2	Start IclA Easy for **IclA**. **Result:** the following start-up screen is displayed:  Not Connected

Step	Action
3	Choose the command **Connection** → **CAN Connection**. **Result:** a text box is displayed.  CAN connection Connection to Peak CAN converter Baudrate: 500 kBaud CAN address: 6 Hardware: LPT1 IRQ 7 IO address: 0378 Node guarding Deactivate □ Value in seconds 5
4	The **Baudrate** must be set to 500 Kbaud. The **CAN address** must be set to 6. The **Hardware** must be set to LPT1 (Dongle PCAN PS/2). **Result:** a data transfer from the servodrive to the connected work station is begun.

Basic IclA Configuration

An example is given to illustrate modification of the acceleration value. This table describes the procedure for entering this setting:

<table>
<tr><th>Step</th><th>Action</th></tr>
<tr><td>1</td><td>Following a connection and transfer of the device’s configurations, IclA Easy displays a screen that gives access to device control, tuning and monitoring functions.</td></tr>
<tr><td>2</td><td>Choose the <b>Motion</b> parameter in the <b>Parameter Groups</b>.
<b>Result:</b> the <b>Parameter</b> window is displayed.


<table>
<tr><th>Name</th><th>Value</th><th>Unit</th><th>Description</th><th>Range</th></tr>
<tr><td>invertDir</td><td>no direction inversion</td><td></td><td>Definition of direction of rotation</td><td>0..1</td></tr>
<tr><td>dec_Stop</td><td>5879</td><td>rpm/s</td><td>Deceleration of Quick-Stop</td><td>1..250000</td></tr>
<tr><td>v_target0</td><td>60</td><td>rpm</td><td>Default setpoint speed</td><td>1..</td></tr>
<tr><td>acc</td><td>1000</td><td>rpm/s</td><td>Acceleration</td><td>1..250000</td></tr>
</table>
🟩 POWER DISABLED
[6]Op Enable
Connected: CAN Adresse=6 Baudrate=500KBit/s</td></tr>
<tr><td>3</td><td>In the <b>acc</b> line, the acceleration can be set to 1000.</td></tr>
<tr><td>4</td><td>Save the CANopen settings to EEprom with the command <b>Parameter → Send parameter group to drive</b>.
<b>Note:</b> it is possible to adjust the servodrive’s settings with the same procedure.</td></tr>
<tr><td>5</td><td>Once the settings have been adjusted, use the command <b>File → Close</b> to disconnect.</td></tr>
</table>

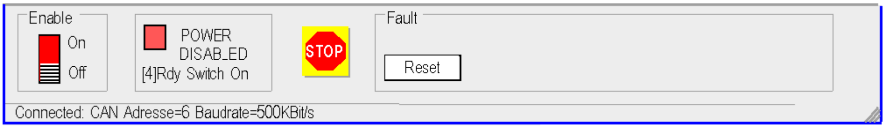
Tuning the IclA with IclA Easy

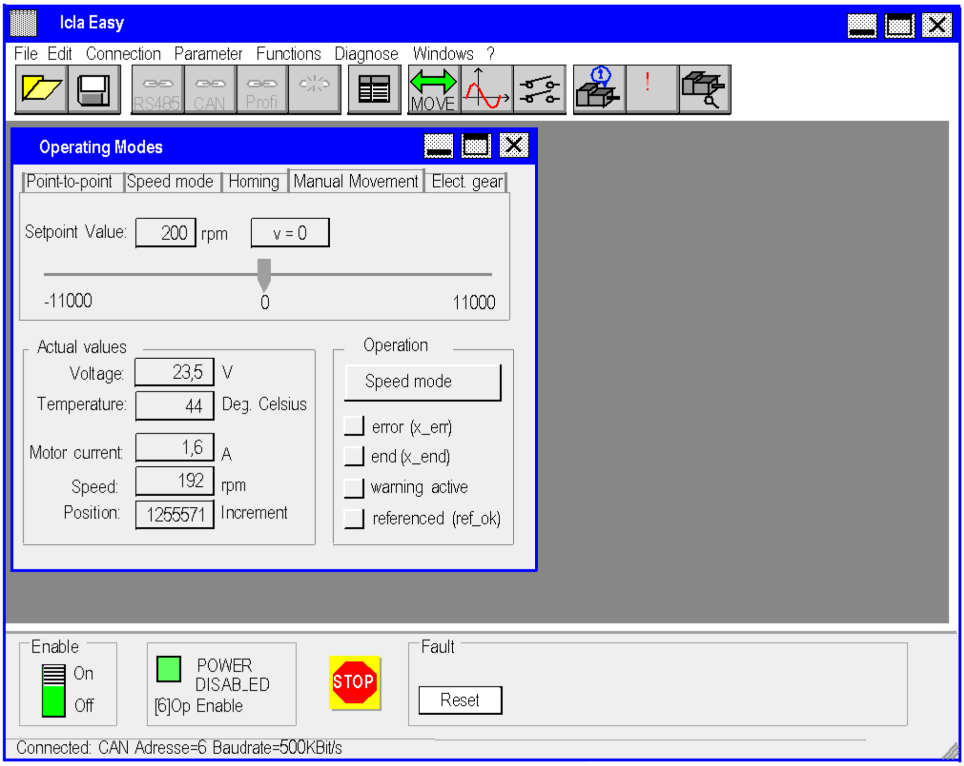
In Advance

We recommend tuning the axis kinematic before the program automatically starts it.

Tuning Example

The following table gives an example of kinematic tuning:

Step	Action
1	Connect *(see page 193)* to the **IclA**.
2	The following figure shows part of the new window. This lower window provides access to **IclA** command functions:  Enable On Off POWER DISAB_ED [4]Rdy Switch On STOP Fault Reset Connected: CAN Adresse=6 Baudrate=500KBit/s
3	Click the **Reset** button to clear any problems.
4	Place the **Enable** zone cursor on **ON**.
5	Choose the command **Functions** → **Operating modes**. **Result:** Operating modes windows is displayed.

Step	Action
6	Choose the **Speed mode** tab Enter the value 200 in the **Setpoint value** zone. **Result:** the motor runs and the sub-window is animated:  Icla Easy File Edit Connection Parameter Functions Diagnose Windows ? RS485 CAN Profi MOVE Operating Modes Point-to-point Speed mode Homing Manual Movement Elect. gear -11000 0 11000 Operation Speed mode STOP Reset Setpoint Value: 200 rpm v = 0 Actual values Voltage: 23,5 V Temperature: 44 Deg. Celsius Motor current: 1,6 A Speed: 192 rpm Position: 1255571 Increment ☐ error (x_err) ☐ end (x_end) ☐ warning active ☐ referenced (ref_ok) Enable On Off 🟩 POWER DISAB_ED [6]Op Enable Fault Connected: CAN Adresse=6 Baudrate=500KBit/s
7	Place the **Enable** zone cursor on **OFF** once tuning has been finalized.

Index