

(for software release 1.00.00.20)

Edition 03.2026



Despite the care taken in the preparation of this document, NUM cannot guarantee the accuracy of the information it contains and cannot be held responsible for any errors therein, nor for any damage which might result from the use or application of the document.

The physical, technical and functional characteristics of the hardware and software products and the services described in this document are subject to modification and cannot under any circumstances be regarded as contractual.

The programming examples described in this manual are intended for guidance only. They must be specially adapted before they can be used in programs with an industrial application, according to the automated system used and the safety levels required.

All rights reserved. No part of this manual may be copied or reproduced in any form or by any means whatsoever, including photographic or magnetic processes. The transcription on an electronic machine of all or part of the contents is forbidden.

© Copyright NUM 2026 software

This software is the property of NUM. Each memorized copy of this software sold confers upon the purchaser a non-exclusive licence strictly limited to the use of the said copy. No copy or other form of duplication of this product is authorized.

Summary

1	Foreword	13
1.1	Use of symbols in the manual	13
1.2	Document revisions	13
1.3	How to read the manual	14
1.4	NUM Agencies	16
2	General	21
2.1	General application	21
2.2	FlexiumPro system configuration	22
2.2.1	FlexiumPro 6, FlexiumPro 8 and FlexiumPro 68 systems	22
2.2.2	FlexiumPro Multi-Panel with FS184i and MP07	24
2.2.3	FlexiumPro and FS122 panel	25
2.2.4	FlexiumPro and Office PC	25
2.3	FlexiumPro Tools	26
2.3.1	Pre-Requisites	26
3	FlexiumPro Tools installation and configuration	31
3.1	FlexiumPro Tools installation	31
3.2	Create a project	32
3.3	Building the device tree	34
3.4	NCK advanced editors	37
3.4.1	MachineConfig TAB: NCK - Settings	37
3.4.1.1	MachineConfig TAB: NCK - Programming	40
3.4.1.2	MachineConfig TAB: NCK - Memory	41
3.4.2	NCK TAB: Axes Calibrations	42
3.4.2.1	NCK TAB: Axes calibration - Axes	42
3.4.2.2	NCK TAB: Axes calibration - Interaxes	43
3.4.3	NCK TAB: Parameters editor	44
3.4.4	NCK TAB: Status & Information	44
3.5	NCK configuration	45
3.5.1	Handwheels	46
3.5.2	MachineConfig TAB: Channel - Configuration	47
3.5.3	MachineConfig TAB: Channel - Settings	48
3.5.4	MachineConfig TAB: Axes - Settings	49
3.5.5	MachineConfig TAB: Axes - Kinematics	50
3.5.6	MachineConfig TAB: Axes - Travels	50
3.5.7	MachineConfig TAB: Axes - Servo-System	51
3.5.8	MachineConfig TAB: Axes - EGB	51
3.5.9	Spindle declaration	52
3.5.10	MachineConfig TAB : Spindles - Settings	52
3.5.11	MachineConfig TAB: Spindles - Servo-systems	53
3.5.12	MachineConfig TAB: Spindles - Rigid tapping	54
3.6	I/OS Bus	55
3.6.1	EtherCAT settings	55
3.6.2	EtherCAT scanning	55
3.6.2.1	Defining a communication channel	56
3.6.2.2	Browse for EtherCAT Mac Address	56
3.6.2.3	Scanning the bus configuration	57
3.6.3	Checking the bus configuration	58

3.7	CANOpen settings	59
3.7.1	Add CANbus	59
3.7.2	CANbus settings	59
3.7.2.1	Baudrate.....	60
3.7.2.2	Addresses	60
4	PLC Programming	65
4.1	General	65
4.1.1	Tasks	65
4.1.2	POUs.....	66
4.1.3	Variables	66
4.1.4	Languages	67
4.1.4.1	Affectation	67
4.1.4.2	Operations (List is not exhaustive).....	67
4.1.4.3	Conversion.....	68
4.1.4.4	Set - Reset	68
4.1.4.5	Test	68
4.1.4.6	Loops	68
4.1.4.7	Case.....	69
4.2	CNC <-> PLC Exchange Zone.....	70
4.2.1	Structure RCNC	71
4.2.1.1	Structure RCNC.Axis[.].	72
4.2.1.2	Structure RCNC.Axis[.].DigitalAxis	74
4.2.1.3	Structure RCNC.Axis[.].SAMX	75
4.2.1.4	Structure RCNC.Channel[.].	76
4.2.1.5	Structure RCNC.Channel[.].DecodeM	81
4.2.1.6	Structure RCNC.Channel[.].DecodeM.General	81
4.2.1.7	Structure RCNC.Channel[.].DecodeM.Spindle[...]	82
4.2.1.8	Method RCNC.Channel[.].ReadEParameter(.....	82
4.2.1.9	Method RCNC.Channel[.].ReadMaximumDynamicTravelLimits(.....	83
4.2.1.10	Method RCNC.Channel[.].ReadMinimumDynamicTravelLimits(.....	83
4.2.1.11	Method RCNC.Channel[.].ReadPosition(.....	83
4.2.1.12	Array RCNC.E10000[.].	83
4.2.1.13	Array RCNC.E30000[.].	84
4.2.1.14	Array RCNC.E33000[.].	84
4.2.1.15	Array RCNC.E36000[.].	84
4.2.1.16	Array RCNC.E37000[.].	84
4.2.1.17	Structure RCNC.General	85
4.2.1.18	Structure RCNC.IO	88
4.2.1.19	Method RCNC.GetParameterType(.....	89
4.2.1.20	Method RCNC.ReadParameter(.....	89
4.2.1.21	Structure RCNC.Spindle[.].	89
4.2.1.22	Structure RCNC.Spindle[.].Load	90
4.2.2	Structure Simulation	91
4.2.2.1	Structure Simulation.RCNC	91
4.2.2.2	Structure Simulation.WCNC	91
4.2.2.3	Structure Simulation.WCNC.Axis[.].	91
4.2.2.4	Structure Simulation.WCNC.Axis[.].DigitalAxis	92
4.2.2.5	Structure Simulation.WCNC.Axis[.].SAMX	93
4.2.3	Structure WCNC	93
4.2.3.1	Structure WCNC.Axis[.].	94
4.2.3.2	Structure WCNC.Axis[.].DigitalAxis	95
4.2.3.3	Method WCNC.Axis[.].DigitalAxis.MotorBrakeTest	95
4.2.3.4	Structure WCNC.Channel[.].	96
4.2.3.5	Structure WCNC.Channel[.].Spindle[...]	99
4.2.3.6	Method WCNC.Channel[.].WriteEParameter(.....	100
4.2.3.7	Method WCNC.Channel[.].WriteMaximumDynamicTravelLimit(.....	100
4.2.3.8	Method WCNC.Channel[.].WriteMaximumDynamicTravelLimits(.....	100
4.2.3.9	Method WCNC.Channel[.].WriteMinimumDynamicTravelLimit(.....	101
4.2.3.10	Method WCNC.Channel[.].WriteMinimumDynamicTravelLimits(.....	101
4.2.3.11	Array WCNC.E20000[.].	102

4.2.3.12	Array WCNC.E40000[.]	102
4.2.3.13	Array WCNC.E43000[.]	102
4.2.3.14	Array WCNC.E46000[.]	103
4.2.3.15	Array WCNC.E47000[.]	103
4.2.3.16	Structure WCNC.General	104
4.2.3.17	Structure WCNC.General.LockIncJog	108
4.2.3.18	Structure WCNC.General.LockSpecificMode	108
4.2.3.19	Structure WCNC.Handwheel[.]	108
4.2.3.20	Structure WCNC.IO	109
4.2.3.21	Structure WCNC.IO.AnalogOutput[.]	109
4.2.3.22	Structure WCNC.IO.DigitalOutput[.]	109
4.2.3.23	Structure WCNC.Spindle[.]	110
4.2.3.24	Structure WCNC.Spindle[.] .Load	110
4.2.3.25	Method WCNC.WriteParameter()	111
4.3	Miscellaneous Read Data	112
4.4	Methods and Functions	113
4.4.1	Method LoadToolNumber()	113
4.4.1.1	Syntax	113
4.4.1.2	Parameters	113
4.4.1.3	Result return codes	113
4.4.2	Method ToolDyn()	114
4.4.2.1	Syntax	114
4.4.2.2	Parameters	114
4.4.2.3	Result return codes	114
4.4.2.4	Example	115
4.4.3	Probing	116
4.4.3.1	Method TouchProbe.ChannelsCfg()	116
4.4.3.2	Method TouchProbe.HardwareCfg()	117
4.4.3.3	Method TouchProbe.InputReading()	118
4.4.3.4	Structure RCNC.ProbeInput[.]	118
4.4.3.5	Example	119
4.4.4	Messages and Warnings	120
4.4.4.1	Text Management in the Tools	120
4.4.4.2	HMI	121
4.4.5	PLC Message Management	121
4.4.5.1	Define the characteristics of the receipt classes (once)	121
4.4.5.2	Activate the messages when necessary	121
4.4.5.3	Checking which messages are active	122
4.4.5.4	Checking a reaction class	122
4.4.5.5	Checking a receipt class	122
4.4.5.6	Cleaning a class of messages	122
4.4.6	Operator's message in different languages	123
4.4.7	Warnings	123
4.5	Analog axes via EtherCAT	125
4.5.1	General configuration	125
4.5.2	Interface module configuration	127
4.5.2.1	Configuration for TTL interface (CTMT 5101)	127
4.5.2.2	Configuration for SSI interface (EL 5002)	132
4.5.2.3	Configuration for EnDat interface	134
4.5.3	PLC program	136
4.5.4	Parameterization	137
5	FlexiumPro functions.....	141
5.1	Automatic homing	141
5.1.1	Homing mode	142
5.1.1.1	Example of an homing program for multi channel	142
5.1.1.2	Automatic Homing Procedure in Homing Modes	143
5.1.2	Call by function G199: AUTO or MDI Mode	144
5.1.3	Case of axes synchronized by machine parameters	144
5.1.3.1	Case of several slave axes	144

5.2	Auxiliary Channels	145
5.2.1	Operating auxiliary axes	145
5.2.1.1	Reset.....	145
5.2.1.2	Auxiliary Axes	145
5.2.1.3	Programming	145
5.2.1.4	ISO Program	146
5.2.2	PLC exchange zone.....	146
5.3	Independent Channels.....	147
5.4	Duplicated and synchronized axes	148
5.4.1	Principle	148
5.4.2	Implementation	148
5.4.2.1	Defining Axes Couplings.....	149
5.4.2.2	Enabling axes duplication or synchronization.....	150
5.4.2.3	Synchronization	150
5.4.2.4	Enabling by Machine Parameters or E964xx.....	151
5.4.2.5	Enabling by External Parameters	152
5.4.3	Tuning	153
5.4.4	Restrictions	154
5.5	Tandem function	155
5.6	Spindle synchronization	156
5.6.1	Parameters.....	156
5.6.2	Activating the synchronization	157
5.6.3	Setting procedure	157
5.6.4	Example	158
5.7	Spindle commutation	159
5.7.1	Axis / Spindle commutation	159
5.7.1.1	General	159
5.7.1.2	PLC operation	160
5.7.1.3	Programming	160
5.7.1.4	Example	160
5.8	Delayed Power down in case of error.....	161
5.8.1	Example	162
5.8.1.1	CNC channel program	162
5.8.1.2	Auxiliary channel program	162
5.9	Acceleration profiles	163
5.9.1	Declaring the type of acceleration	166
5.9.2	Constraints related to trapezoidal or parabolic	166
5.9.2.1	Sequenced thread cutting (G38).....	166
5.9.2.2	5-axes polynomial interpolation	166
5.10	Backtrack and resume	167
5.10.1	Auto recall	167
5.11	Smart Servo Balance.....	168
5.11.1	Adjustment	168
5.11.2	Settings	169
5.11.3	Verification: A correct setting can be verified with a Ball-Bar test.....	169
5.12	Electronic Gear box	170
5.12.1	General	170
5.12.2	Setup.....	171
5.12.2.1	Machine parameters	171
5.12.2.2	Leader axis	171
5.12.2.3	E parameters and PLC signals	172
5.12.2.4	Reset.....	172
5.12.2.5	Emergency.....	173
5.13	Five axes operation.....	174
5.13.1	General	174
5.13.2	Targeted kinematic chains	174
5.13.2.1	Five-axes kinematic chain	175
5.13.2.2	General settings.....	177

5.14	Kinematic measuring cycle248 with 3D touch probe	180
5.14.1	General	180
6	FlexiumPro Tools - Instruments.....	185
6.1	General	185
6.2	Drive Test points	186
6.3	Oscilloscope	187
6.3.1	Stimulus	187
7	Machine parameters setup.....	191
7.1	Introduction	191
7.2	General configuration	191
7.2.1	Data tables.....	191
7.2.2	Related parameters	192
7.3	Axes and Spindle declaration	193
7.3.1	Data tables.....	193
7.3.1.1	Channels definition	193
7.3.1.2	Axis and Spindle per channel	193
7.3.1.3	Axes and Spindles definition in each channel	194
7.3.1.4	Handwheels definition.....	194
7.3.2	Related parameters	195
7.4	Axis and Spindle characteristics.....	196
7.4.1	Data tables.....	196
7.4.1.1	Axis data tables	196
7.4.1.2	Spindle data tables	197
7.4.2	Related parameters	198
7.5	Axis and Spindle dynamic.....	199
7.5.1	Related parameters	199
7.6	NCK controlled analog axes	200
7.6.1	Related parameters	200

A	Appendix.....	205
A.1	Axis calibration.....	205
A.1.1	General	205
A.1.2	Minimum and maximum travel points not corrected	206
A.1.3	Minimum and maximum travel points corrected	206
A.2	Inter-axis calibration.....	207
A.2.1	Description	207
A.2.1.1	General	207
A.2.1.2	Special case of correction in a single point	207
A.2.2	Examples of Inter-axis Compensation Calibration	208
A.2.2.1	Correction of a perpendicular alignment fault	208
A.2.2.2	Correction of spindle flexion	208
A.2.2.3	Correction of sag of the cross piece of a gantry machine	209
A.2.2.4	Correction of an axis expansion	209
A.2.3	Tools for Inter-axis Compensation Calibration	209
A.2.4	Records of the Corrections on the Axes	210
A.3	Dynamic Inter-axis Calibration	211
A.3.1	Addresses of the Correction Parameters	211
A.3.2	Correction Tables	211
A.3.3	Writing and Enabling the Correction Tables	213
A.3.3.1	Conditions for Writing Parameters E81xxx and E82xxx	213
A.3.3.2	Conditions for Writing Parameters E940xx	213
A.3.3.3	Procedure for Writing and Enabling the Correction Tables	213
A.4	Example of MP04 programming	214
A.5	Example of MP06 / MP07 programming	216
A.6	Kinematic structures managed by the RTCP and iPLANE functions	220
A.7	Summary of the reference postures of five axis kinematics	221
A.8	E80x parameters used by RTCP and by iPLANE functions.....	229
A.9	Time charts	230
A.9.1	General	230
A.9.2	Order of execution of S, T and M functions	231
A.9.3	Details on M functions	232
A.9.4	Details on E parameters	235
A.9.5	Axes clamp	236
A.9.5.1	Management by .WCNC.Channel[x].EndExtMovement	236
A.9.5.2	Management by Axis.Stop	237
A.9.6	Spindle indexing.....	238
A.9.6.1	Indexing at constant deceleration	238
A.9.6.2	Indexing at constant deceleration not chosen	239

Page deliberately empty.

Page deliberately empty.

Summary

1	Foreword.....	13
1.1	Use of symbols in the manual.....	13
1.2	Document revisions	13
1.3	How to read the manual.....	14
1.4	NUM Agencies.....	16

1

2

3

4

5

6

7

A

Page deliberately empty.

1 Foreword

Use of the manual

The Commissioning Manual gives information on how to customize the FlexiumPro system for a particular machine. This manual is for use by qualified personnel only.

To reduce the risk due to an improper use of the product, do not perform any servicing other than that contained in this manual unless you are qualified to do so.

To ensure proper use of these products, please always read this manual carefully and retain for future reference. Should the unit require maintenance, please contact a NUM authorized service center only.

1.1 Use of symbols in the manual

Symbol	Meaning of the symbol
	Important notes on product use.
	You may find this symbol whenever a wrong operation may damage the product.

1.2 Document revisions

Date	Index	Reason for revision	See chapter
10 - 2024	00	Document creation For software release 1.00.00.00	
03 - 2026	01	For software release 1.00.00.20	

1.3 How to read the manual

The machine Commissioning Manual gives information on how to set the parameters to customise the FlexiumPro for a particular machine.

Chapter	Description
1	Foreword - Use of symbols in the manual - Document revisions - How to read the manual - NUM Agencies
2	General - General application - FlexiumPro system configuration - FlexiumPro system configuration - FlexiumPro Tools
3	FlexiumPro Tools installation and configuration - FlexiumPro Tools installation - Create a project - Building the device tree - NCK advanced editors - NCK configuration - I/OS Bus - CANOpen settings
4	PLC Programming - General - CNC <-> PLC Exchange Zone - Miscellaneous Read Data - Methods and Functions - Analog axes via EtherCAT
5	FlexiumPro functions - Automatic homing - Auxiliary Channels - Independent Channels - Duplicated and synchronized axes - Tandem function - Spindle synchronization - Spindle commutation - Delayed Power down in case of error - Acceleration profiles - Backtrack and resume - Smart Servo Balance - Electronic Gear box - Five axes operation - Kinematic measuring cycle248 with 3D touch probe
6	FlexiumPro Tools - Instruments - General - Drive Test points - Oscilloscope

Chapter	Description
7	Machine parameters setup - Introduction - General configuration - Axes and Spindle declaration - Axis and Spindle characteristics - Axis and Spindle dynamic - NCK controlled analog axes
A	Appendix - Axis calibration - Inter-axis calibration - Dynamic Inter-axis Calibration - Example of MP04 programming - Example of MP06 / MP07 programming - Kinematic structures managed by the RTCP and iPLANE functions - Summary of the reference postures of five axis kinematics - E80x parameters used by RTCP and by iPLANE functions - Time charts

1

2

3

4

5

6

7

A

1.4 NUM Agencies

The list of NUM agencies is given on <http://www.num.com>.

Page deliberately empty.

Page deliberately empty.

Summary

2	General	21
2.1	General application	21
2.2	FlexiumPro system configuration	22
2.2.1	FlexiumPro 6, FlexiumPro 8 and FlexiumPro 68 systems	22
2.2.2	FlexiumPro Multi-Panel with FS184i and MP07	24
2.2.3	FlexiumPro and FS122 panel	25
2.2.4	FlexiumPro and Office PC	25
2.3	FlexiumPro Tools	26
2.3.1	Pre-Requisites	26

1

2

3

4

5

6

7

A

Page deliberately empty.

2 General

2.1 General application

A FlexiumPro application is made of different components.

Some of them are mandatory in any system structure:

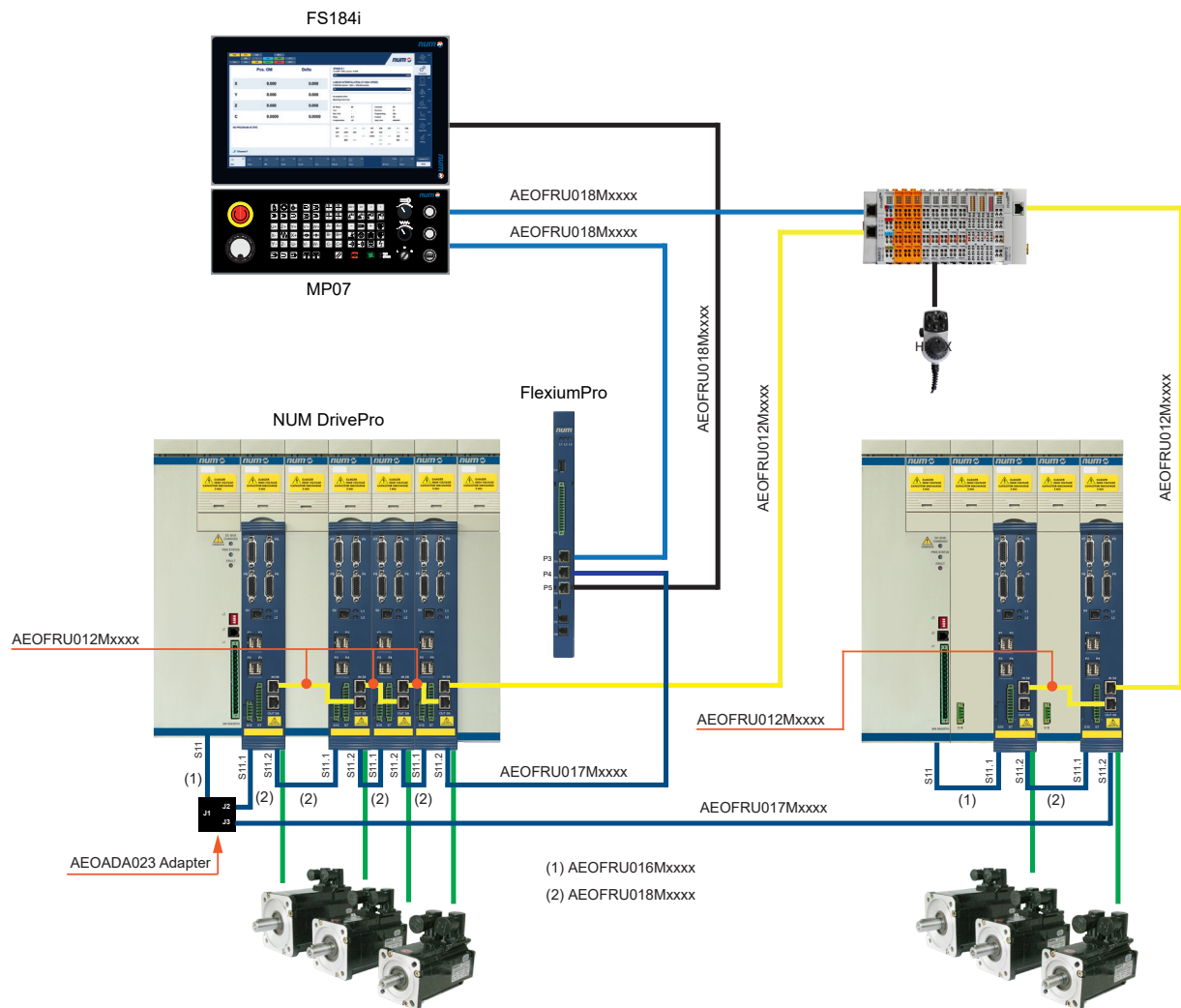
- One FlexiumPro RTK mostly in charge of trajectory and axes control.
- A PLC: It is integrated in the FlexiumPro RTK.
The PLC executes the machine functions and optionally the safety functions.
- Digital drives: they are connected to the RTK; NUM DrivePro for FlexiumPro.
- I/Os Fieldbus on EtherCAT or CANOpen.

Others are optional:

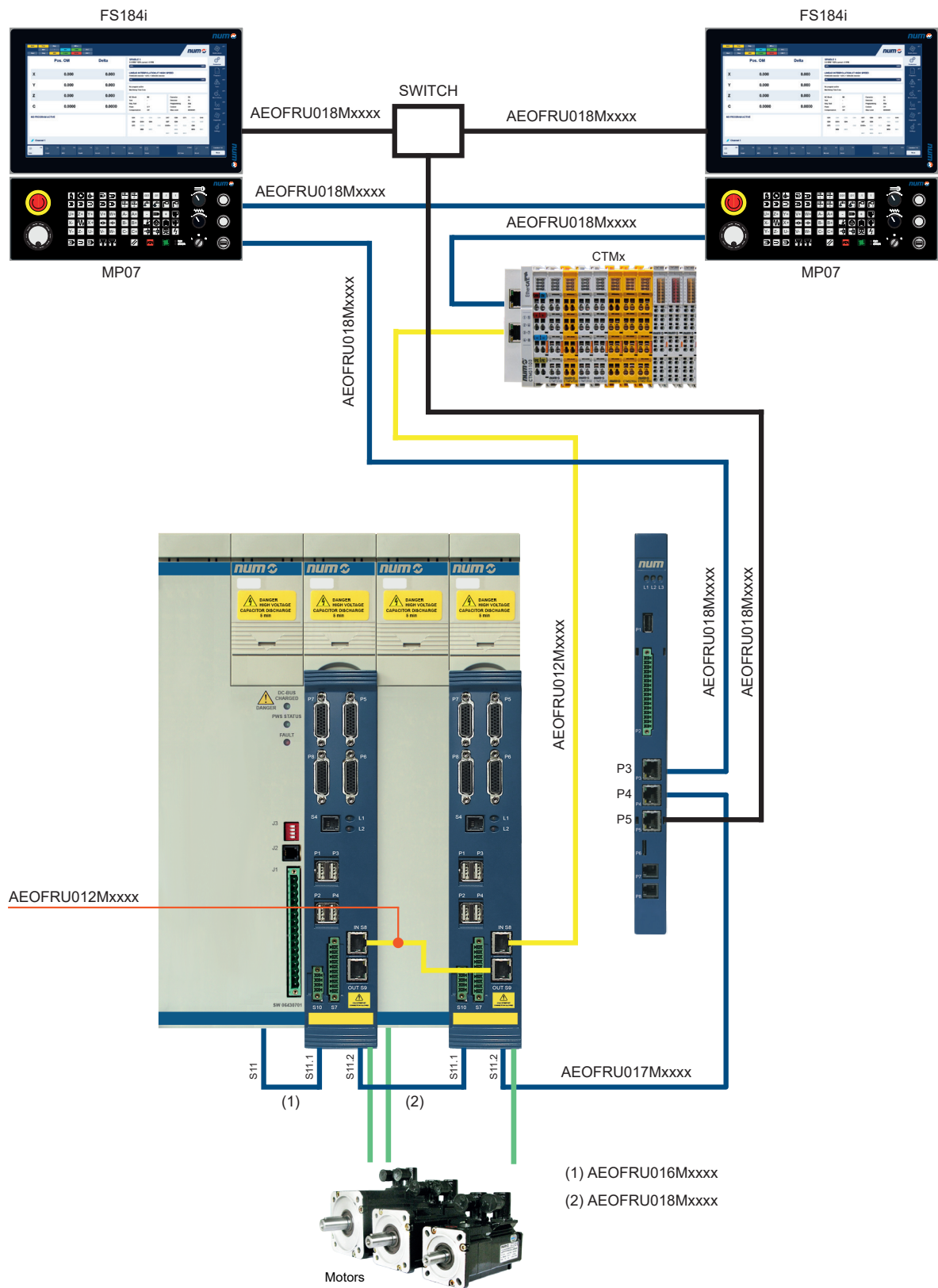
- Operator's panel: independent (FSxxx) or non existent.
- The operator's panel is used to manage the HMI and run the machine.
- Machine Panel: MP06, MP07, MP08.
- Portables devices (Handwheel, ...)
- Safety devices (PLC, I/Os, NUM-SAMX)

2.2 FlexiumPro system configuration

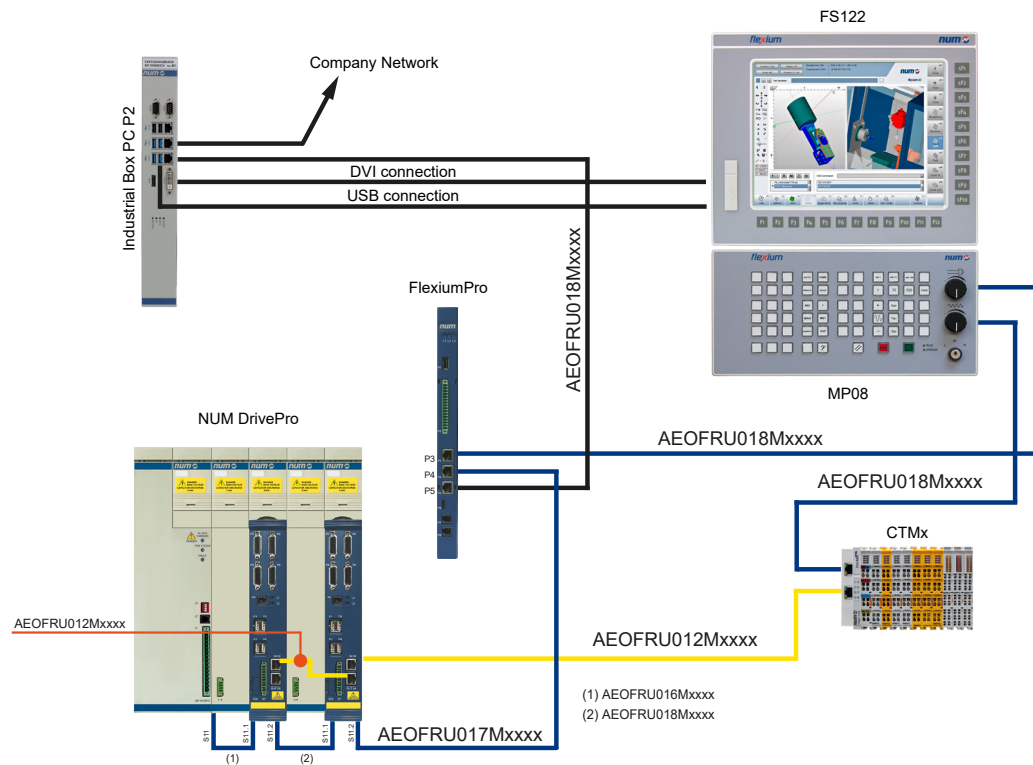
2.2.1 FlexiumPro 6, FlexiumPro 8 and FlexiumPro 68 systems



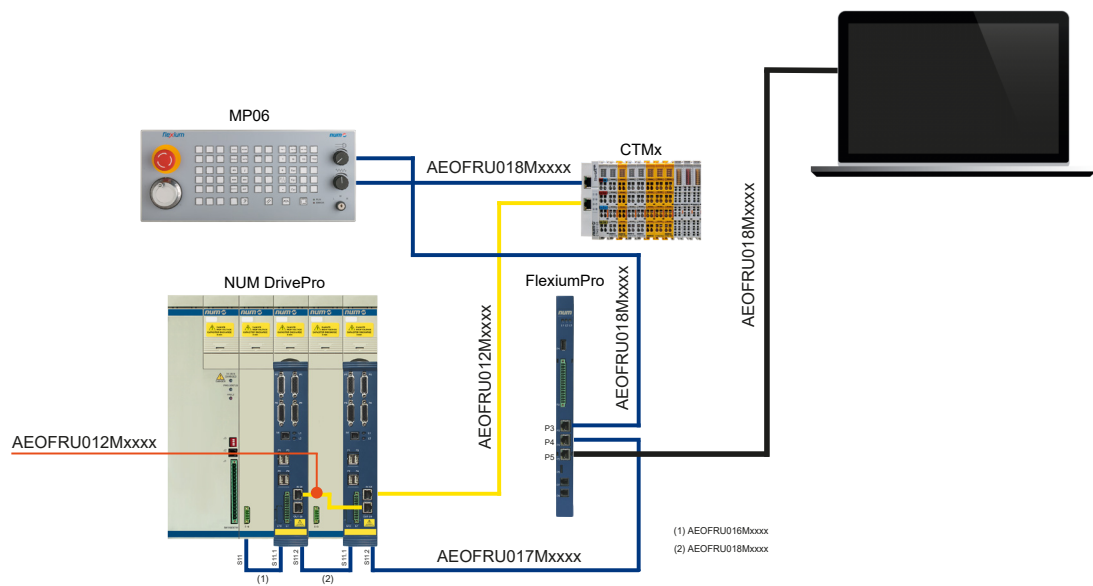
2.2.2 FlexiumPro Multi-Panel with FS184i and MP07



2.2.3 FlexiumPro and FS122 panel



2.2.4 FlexiumPro and Office PC



2.3 FlexiumPro Tools

- Commissioning of the machine will be done with the FlexiumPro Tools suite.
- FlexiumPro Tools will create a binary Application file from a project file.
- A project includes at least:
- The physical description of the machine structure with the different components (devices).
 - The Machine parameters used by the NCK as well as calibration tables
 - Part program files are not directly handled by FlexiumPro Tools but can be stored inside the project.
- The drives parameters – filter settings, tests points,...
- The PLC program.
- As well as some optional components:
- The safety PLC program.
- Optional visualization tables.

2.3.1 Pre-Requisites

- Before creating a project certain information are required:
- Name of the FlexiumPro RTK
- I/O structure : number and type of bus – I/Os groups and functions – additional components
- Number and type of drives, motors, encoders.

Page deliberately empty.

Page deliberately empty.

Summary

3	FlexiumPro Tools installation and configuration.....	31
3.1	FlexiumPro Tools installation.....	31
3.2	Create a project	32
3.3	Building the device tree	34
3.4	NCK advanced editors.....	37
3.4.1	MachineConfig TAB: NCK - Settings.....	37
3.4.1.1	MachineConfig TAB: NCK - Programming.....	40
3.4.1.2	MachineConfig TAB: NCK - Memory	41
3.4.2	NCK TAB: Axes Calibrations	42
3.4.2.1	NCK TAB: Axes calibration - Axes.....	42
3.4.2.2	NCK TAB: Axes calibration - Interaxes	43
3.4.3	NCK TAB: Parameters editor.....	44
3.4.4	NCK TAB: Status & Information.....	44
3.5	NCK configuration	45
3.5.1	Handwheels	46
3.5.2	MachineConfig TAB: Channel - Configuration	47
3.5.3	MachineConfig TAB: Channel - Settings.....	48
3.5.4	MachineConfig TAB: Axes - Settings	49
3.5.5	MachineConfig TAB: Axes - Kinematics.....	50
3.5.6	MachineConfig TAB: Axes - Travels.....	50
3.5.7	MachineConfig TAB: Axes - Servo-System	51
3.5.8	MachineConfig TAB: Axes - EGB.....	51
3.5.9	Spindle declaration	52
3.5.10	MachineConfig TAB : Spindles - Settings	52
3.5.11	MachineConfig TAB: Spindles - Servo-systems.....	53
3.5.12	MachineConfig TAB: Spindles - Rigid tapping	54
3.6	I/OS Bus	55
3.6.1	EtherCAT settings.....	55
3.6.2	EtherCAT scanning.....	55
3.6.2.1	Defining a communication channel.....	56
3.6.2.2	Browse for EtherCAT Mac Address	56
3.6.2.3	Scanning the bus configuration	57
3.6.3	Checking the bus configuration	58
3.7	CANOpen settings.....	59
3.7.1	Add CANbus	59
3.7.2	CANbus settings.....	59
3.7.2.1	Baudrate.....	60
3.7.2.2	Addresses	60

Page deliberately empty.

3 FlexiumPro Tools installation and configuration

3.1 FlexiumPro Tools installation

FlexiumPro Tools come pre-installed on each FS panel; it can also be downloaded from a 'ftp server'. Credential to login will be given by your local NUM office.

All official versions of FlexiumPro Tools are available as well as some beta versions. Do not download or install a beta version unless asked to do so by NUM.

Installing a new version does not require to uninstall the previous. They can all coexist, it is indeed recommended to keep installed all versions you have been working with.

To install the desired software (FlexiumPro Tools or Flexium HMI) click on the corresponding item.

1

2

3

4

5

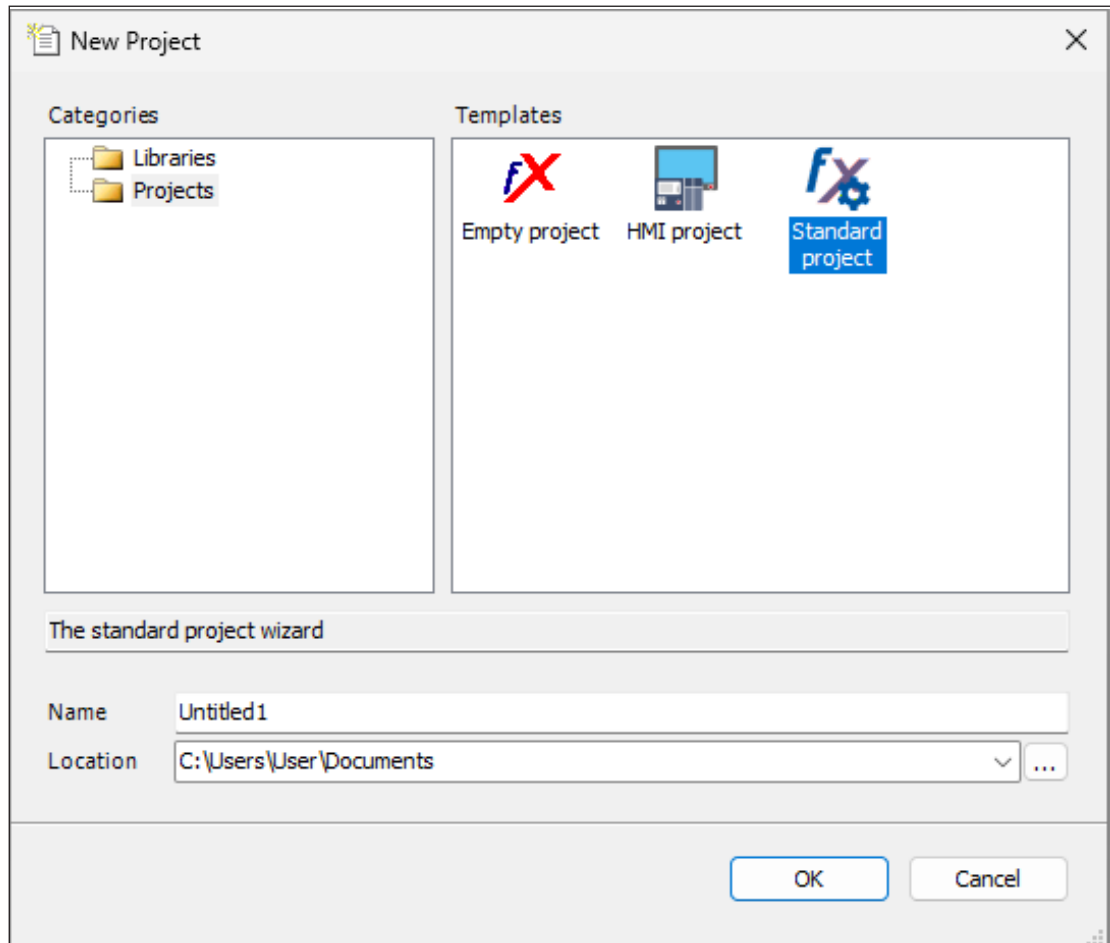
6

7

A

3.2 Create a project

- To create a new project, open the menu 'File'. The easiest way to begin is to use the wizard.



- Using the 'next' button select the different component required in your application.
It is recommended to check the box '*Always use the newest devices version*' unless you want to work on system with an older RTS. In all case it is necessary that the devices version match the RTS version installed on Flexium.



Be careful to select the correct version of the components.
The choices made here have an impact on the rest of the project.

You can rename the component according to your application.

The final configuration may look like:

RTK

RTS name: NUM_FlexiumPro_RTK

Version: 1.0.0.20

Source code download: Implicit on program download and online change

PLC program

PLC programing language: Structured Text (ST)

POU name: PLC_PRG

Task name: BusCycle

Cycle time: 10 ms

1

2

3

4

5

6

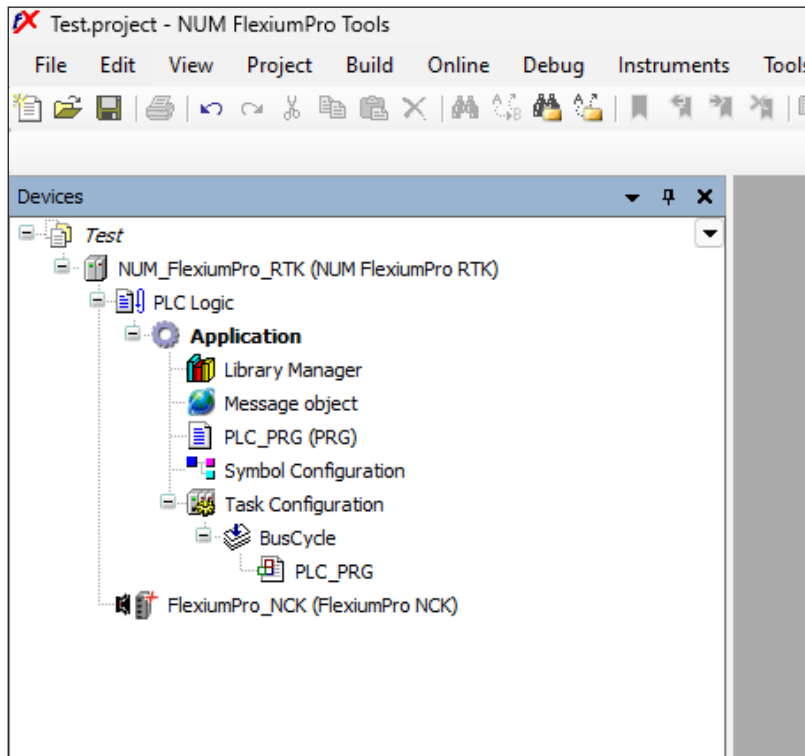
7

A

3.3 Building the device tree

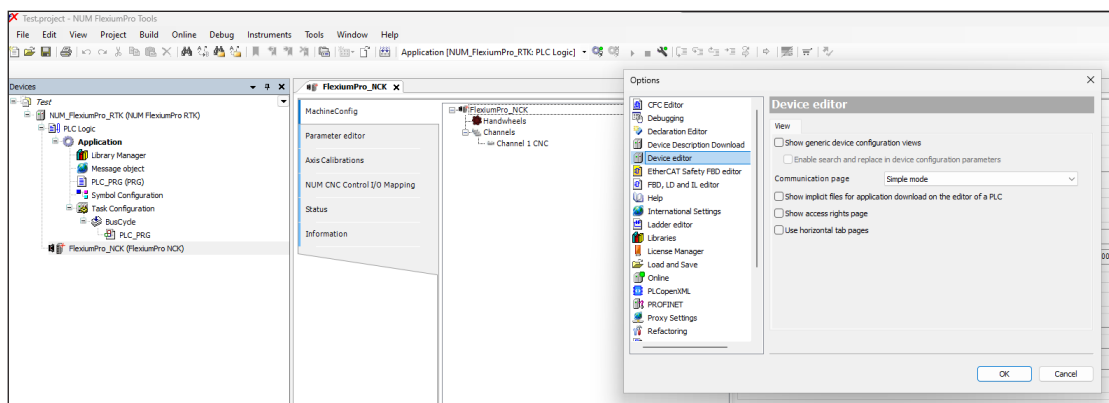
The next step is to create the machine structure. The left side of the screen should display a window 'Devices' similar to the one below.

- If this window does not exist open it via the menu 'View'.



At this stage, you can already perform an initial system check and activate the communication between Flexium and FlexiumPro Tools.

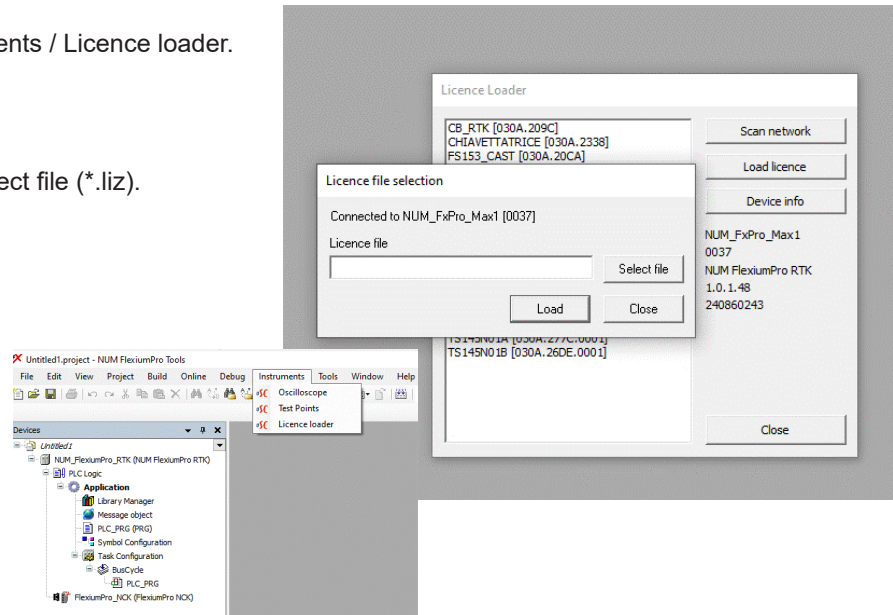
By default FlexiumPro Tools proposes vertically aligned tab pages (1) for better use of wide screens. You can retrieve the previous horizontal structure by the menu Tools/Options / Device editor (2) choice (3). The following screen copy will be generally displayed with horizontal tab structure.



You need now to store the *Licence file* to the system.

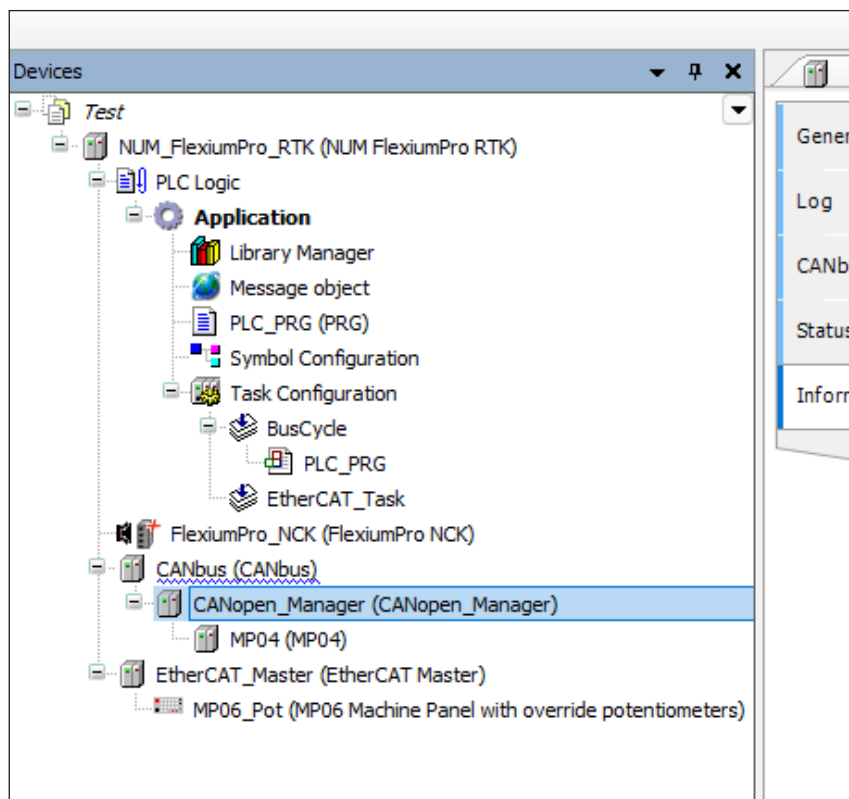
The licence file is necessary to define the software configuration. It is unique per Flexium system. It is delivered on an USB stick in the NCK box. If you have misplaced this stick you can get the file from NUM by specifying the serial number of the NCK.

- Select Instruments / Licence loader.
- Scan Network.
- Load Licence.
- Select the correct file (*.liz).
- Load.
- Close.



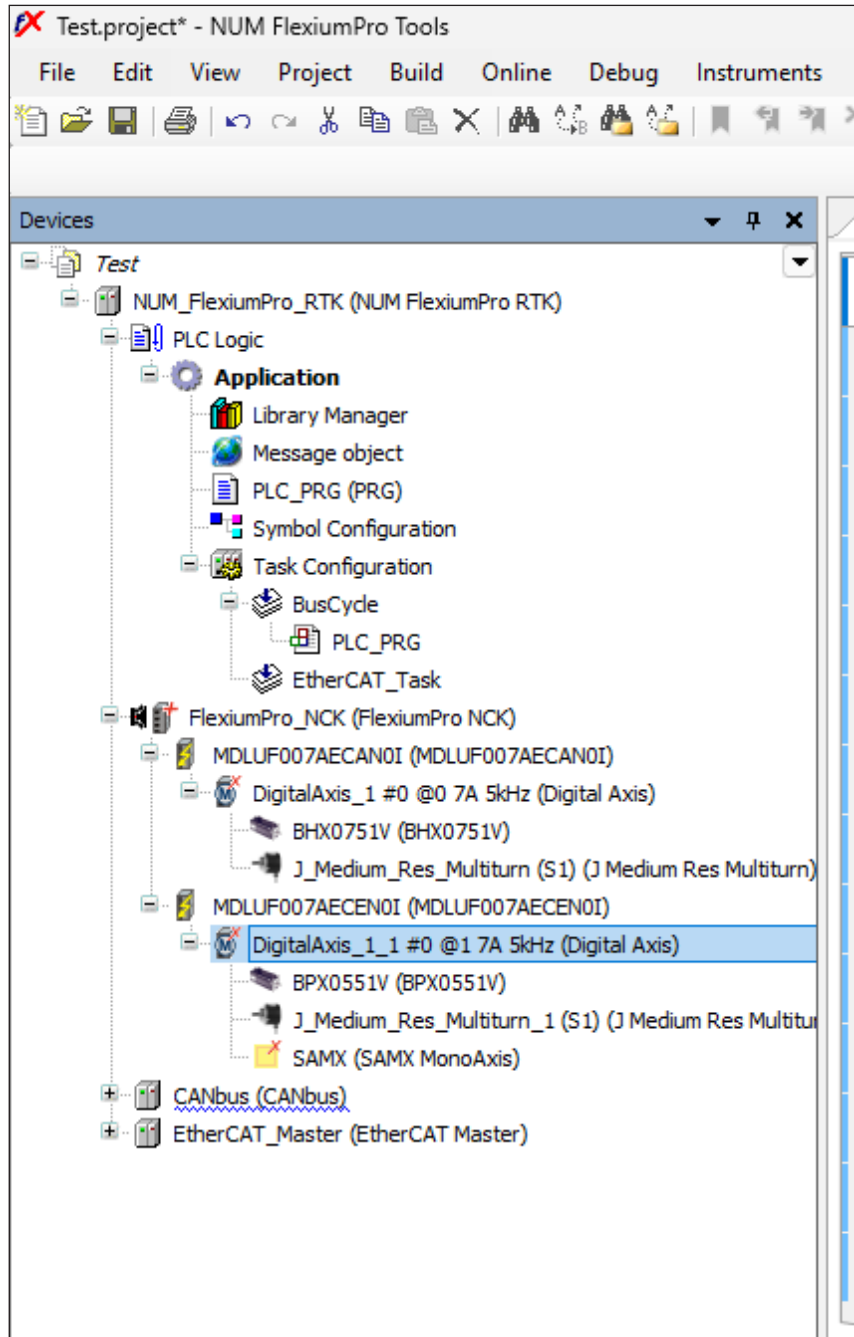
You can now add the different components with a right click on specific items and the command 'Add device'.

- Under NUM_FlexiumPro_RTK add Fieldbus components (see example) Click on a device to add a child components – the window Add device can remain open during the operation.



- Under *FlexiumPro_NCK* add Drives, motors and encoders.

The system will only present the set of motors compatibles with the drive selected. This is the reason why PWM frequency selected at this moment.

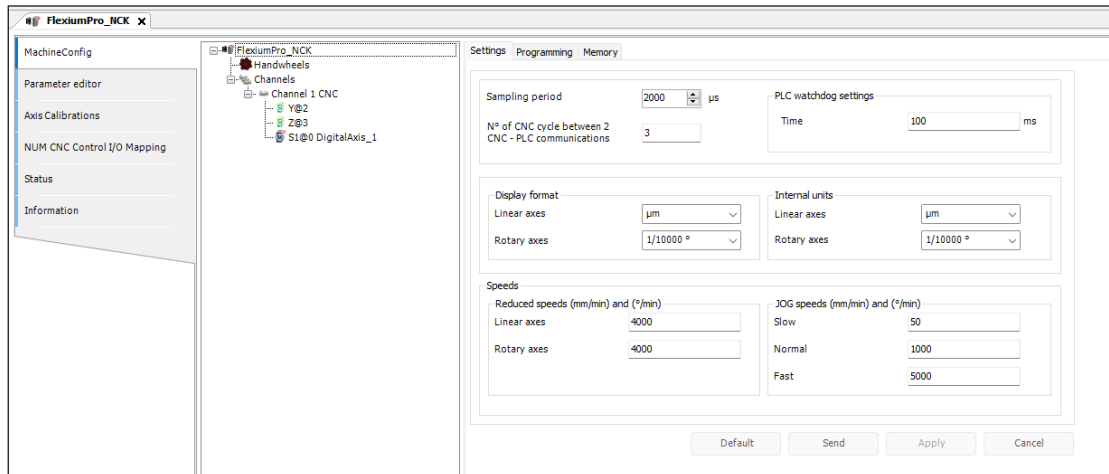


Should you have Safety drives (MDLUF with SAMX), please refer to NUM-SAMX Safety module for NUM DrivePro Instruction manual (M00069) for specific information.

3.4 NCK advanced editors

3.4.1 MachineConfig TAB: NCK - Settings

<i>Axes scan task</i>	P50 N0 : keep the default value when possible.
<i>NC <> PLC exchange rate</i>	P99 N0 : idem
<i>Internal and display units</i>	P4
<i>Jog and reduced speeds</i>	P31



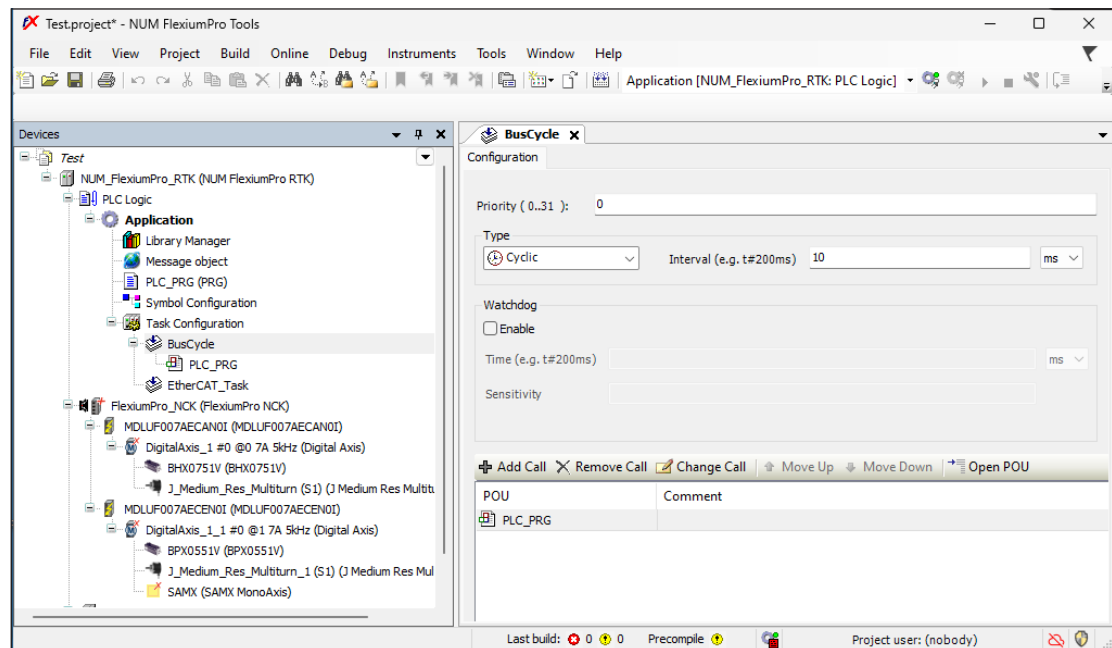
Every P99 N0 * P50 there is an exchange of data between the NCK and the PLC.

PLC watchdog setting

The watchdog reaction time should be set, in case of plc scan time exceeded the configured time the watchdog relay will open the contact.

Exchange data between NCK<->PLC (compatibility mode)

In Task Configuration is possible to set the PLC task as Cyclic like FlexiumPlus



As the NCK and PLC tasks are asynchronous the data exchanged are saved in a temporary buffer until the next run of the BusCycle task (see 4.1) to prevent the processing of erroneous data.

Obviously it is recommended to have exchanges at a rate faster than the PLC scan.

Example

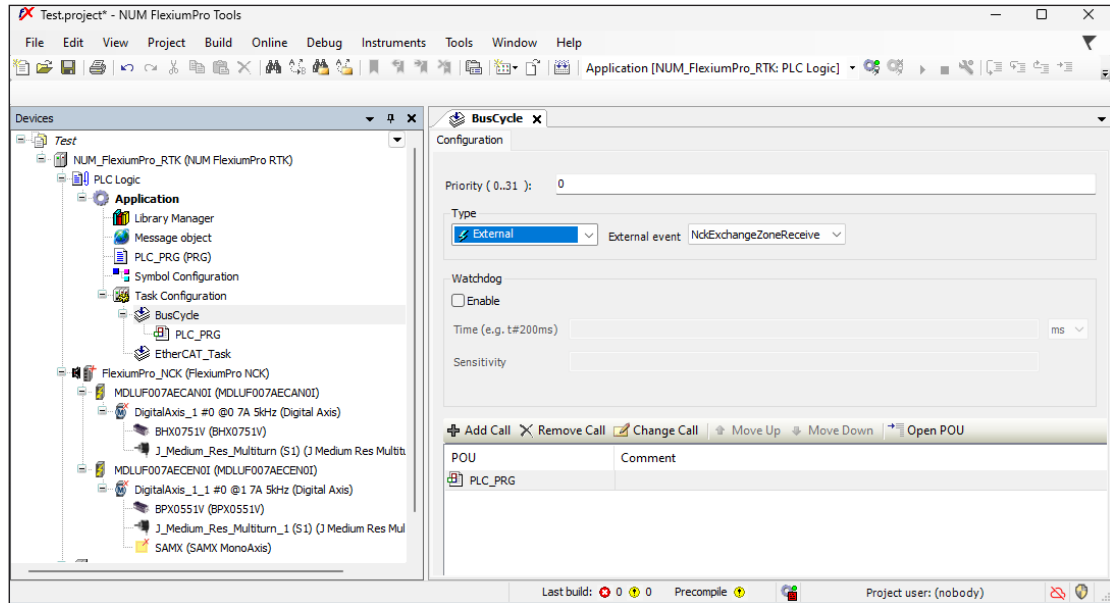
- P99 N0 = 3
- P50 = 2ms
- PLC BusCycle period = 10 ms

There is an exchange between NCK and PLC temporary buffer every 6ms

- The PLC transfers this buffer into its work memory just before launching the BusCycle task every 10 ms. It also updates the I/Os at this time. Therefore the PLC will always work with coherent data;
- As the PLC and NCK clocks are not synchronized it would be hazardous to select 10ms for the exchange between NCK and PLC. 6ms (P99N0 = 3) is a good compromise 8ms will work as well; 4 or 2ms will increase the transfer load due to unnecessary exchanges.

New synchro exchange data between NCK<->PLC (Recommended)

FlexiumPro PLC cycle time is the multiple of interpolator, that means at every $P50 \cdot P99N0$ (after exchange data) an external event from NCK will start the PLC cycle.



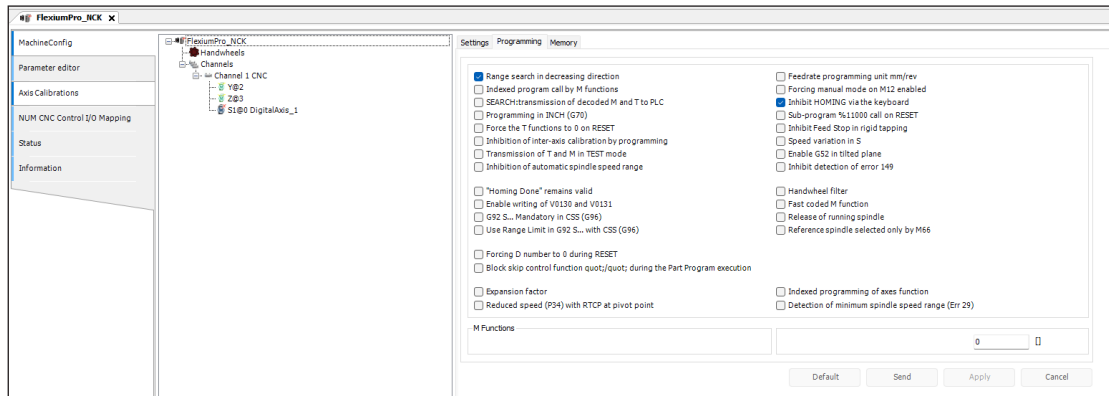
This new PLC cycle time setting and synchro exchange data between NCK<->PLC offer several advantages in terms of efficiency, no timing lost for asynchronous NCK tasks and PLC tasks.

Example

- $P99 N0 = 3$
- $P50 = 2\text{ms}$
- $\text{PLC BusCycle period} = P50 \cdot P99N2 = 6\text{ms}$

3.4.1.1 MachineConfig TAB: NCK - Programming

Default NC configuration	P7
Subroutine calls by M code	P35
M function for workpiece change	P37 N1



M function for workpiece change is used by the NCK to calculate the part machining time in case of looped-back programs. Refer to the machine parameters manual for detailed informations about the P7 settings.

Refer to the FlexiumPro Extended Programming Manual M00075 for information about Indexed Programming of Axes.



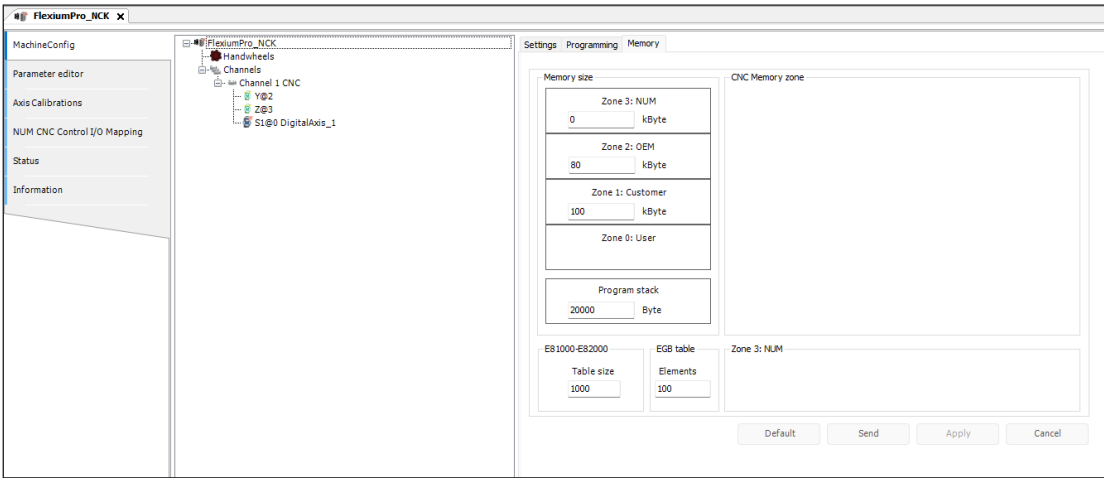
Warning: The tab NCK-Programming not completed in version 1.00.00.10



Pages in preparation, the functions will be completed in next version.

3.4.1.2 MachineConfig TAB: NCK - Memory

Sizes of the different memory areas	P95
Stack size and number of E81/82 parameters	P58
The section memory zone permits to create the content of User and OEM zones by selecting different subroutines. They will be compiled and stored in the NCK when going online. Please note that the files become part of the project, therefore if the source changes you will have to reload the new version. It is also possible to store data in the NUM zone. Files for the NUM zone (*.xm3) will be provided as binary encrypted files.	



Pages in preparation, the functions will be completed in next version.

3.4.2 NCK TAB: Axes Calibrations

3.4.2.1 NCK TAB: Axes calibration - Axes

- This tab allows for entering axes calibration. The data are entered per axis.
- There is a maximum of 2719 points per NCK unit.
- The data are entered in metric (mm- floating point).

<i>P</i>	Axis position
<i>C</i>	Correction in positive direction
<i>D</i>	Correction in negative direction

Data can also be imported from a file (plain text - extension *.axca). File format is as follows:

```

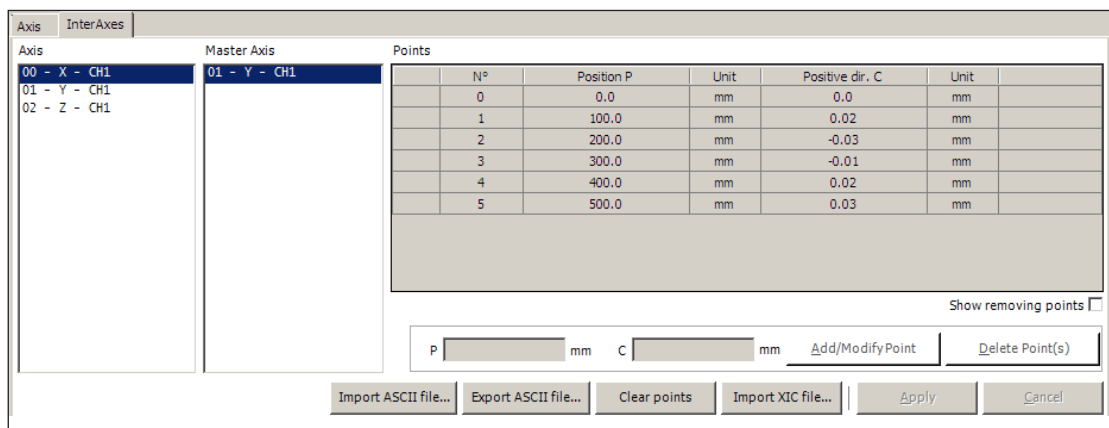
1 MainAxis:CH0 X@0;
2 NoP:4;
3 M;0.0;C;0.002;D;0.003
4 M;100.0;C;-0.001;D;0.002
5 M;500.0;C;0.0;D;-0.001
6 M;1000.0;C;0.005;D;0.001
7 MainAxis:CH0 Y@1;
8 NoP:1;
9 M;0.0;C;0.0;D;0.0
10 MainAxis:CH0 Z@2;
11 NoP:1;
12 M;0.0;C;0.0;D;0.0
13

```

- NoP : Number of point for the particular axis.

3.4.2.2 NCK TAB: Axes calibration - Interaxes

- The data are entered per couple of Slave axis vs Master axis.
- They give the correction on the slave according to the master position.
- There is a maximum of 2708 correction points for the full machine.
- The data are entered in metric (mm - floating point).



Data can be exported/imported to/from an ascii file (extension axic) with the following format.

```

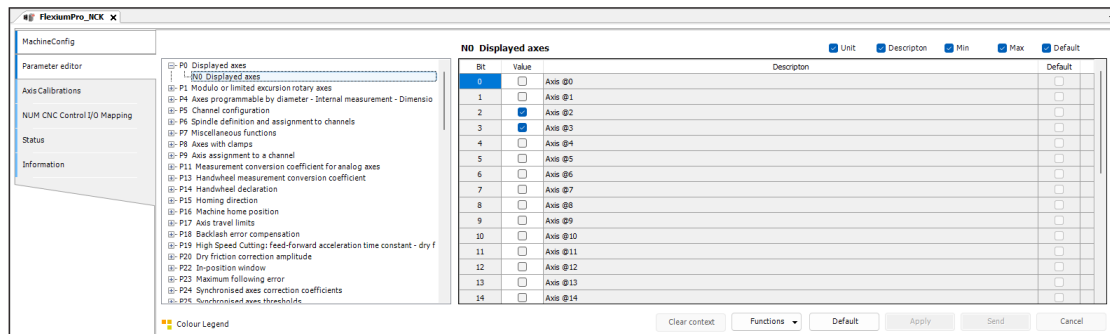
1 MainAxis:CH0 X@0,CorrectionAxis:@1;
2 NoP:6;
3 P;0.0;C;0.0
4 P;100.0;C;0.02
5 P;200.0;C;-0.03
6 P;300.0;C;-0.01
7 P;400.0;C;0.02
8 P;500.0;C;0.03
9

```

3.4.3 NCK TAB: Parameters editor

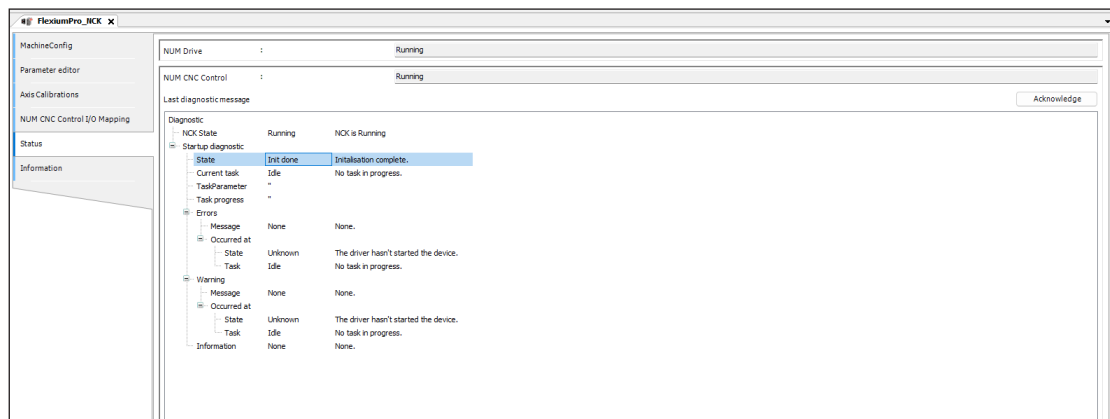
This editor allows to enter directly NCK parameters.

Details about NCK parameters are available in FlexiumPro Parameters manual M00068.



3.4.4 NCK TAB: Status & Information

These editors return information about operation and version numbers. They are seldom required during commissioning.

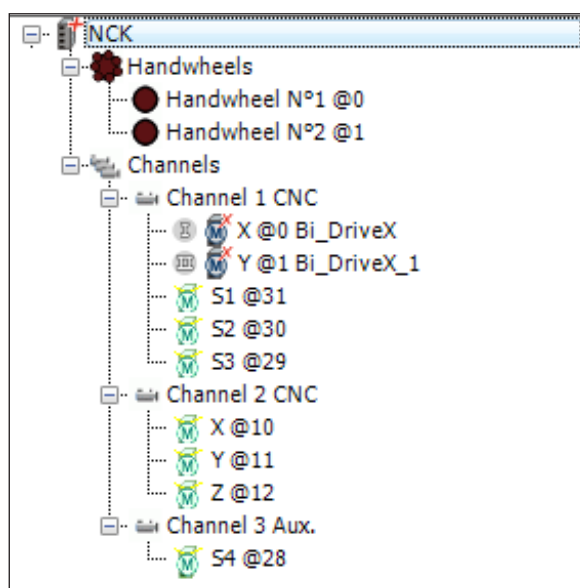


3.5 NCK configuration

NCK configuration consists of the following:

- Define the number of handwheels and their configuration.
- Each spindle also gets a logical number freely defined (1 to 32). These spindles will be managed in the part program by this logical number.
- Create the different channels and specify if they are NC channels or Auxiliary channels.
- For each channel define its type, all spindles and all axes by their physical address (address of the drive). N.B.
- FlexiumPro authorizes up to four spindles per channel.
- The four spindles of FlexiumPro will be in this order the main and the three auxiliary (see FlexiumPro Programming Manual M00073).
- Each spindle of FlexiumPro requires a logical address (unique per NCK). This logical address independent of all other addressing is required for part programming.
- The group 'Spindles and axes pool' groups all axes and spindles that are not linked to a channel.

The complete configuration may look like:

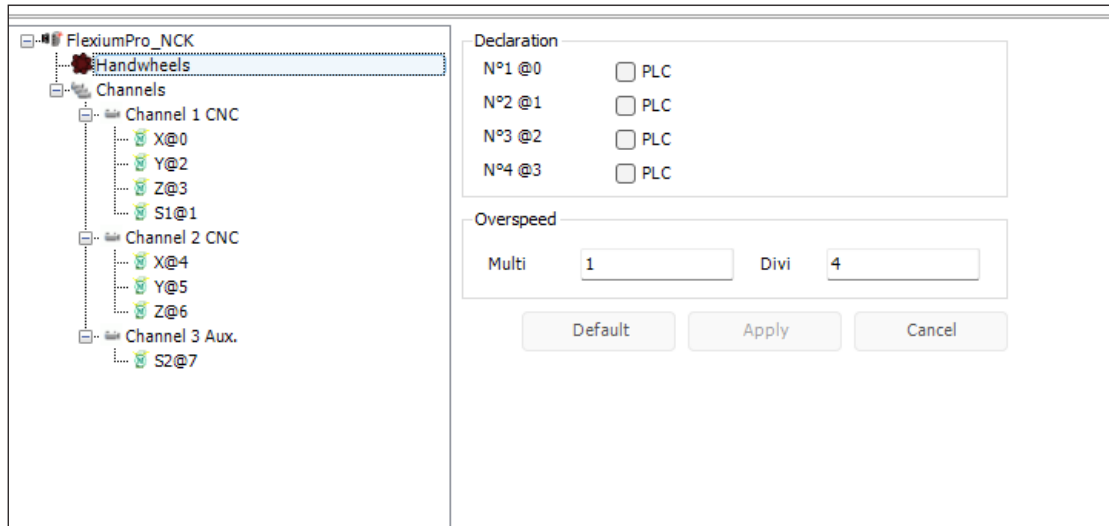


3.5.1 Handwheels

This editor permits to define up to 4 handwheels total.

The handwheel can be connected on the PLC (connected via a PLC counter module or a machine panel as indicated in the FlexiumPro Installation manual M00070).

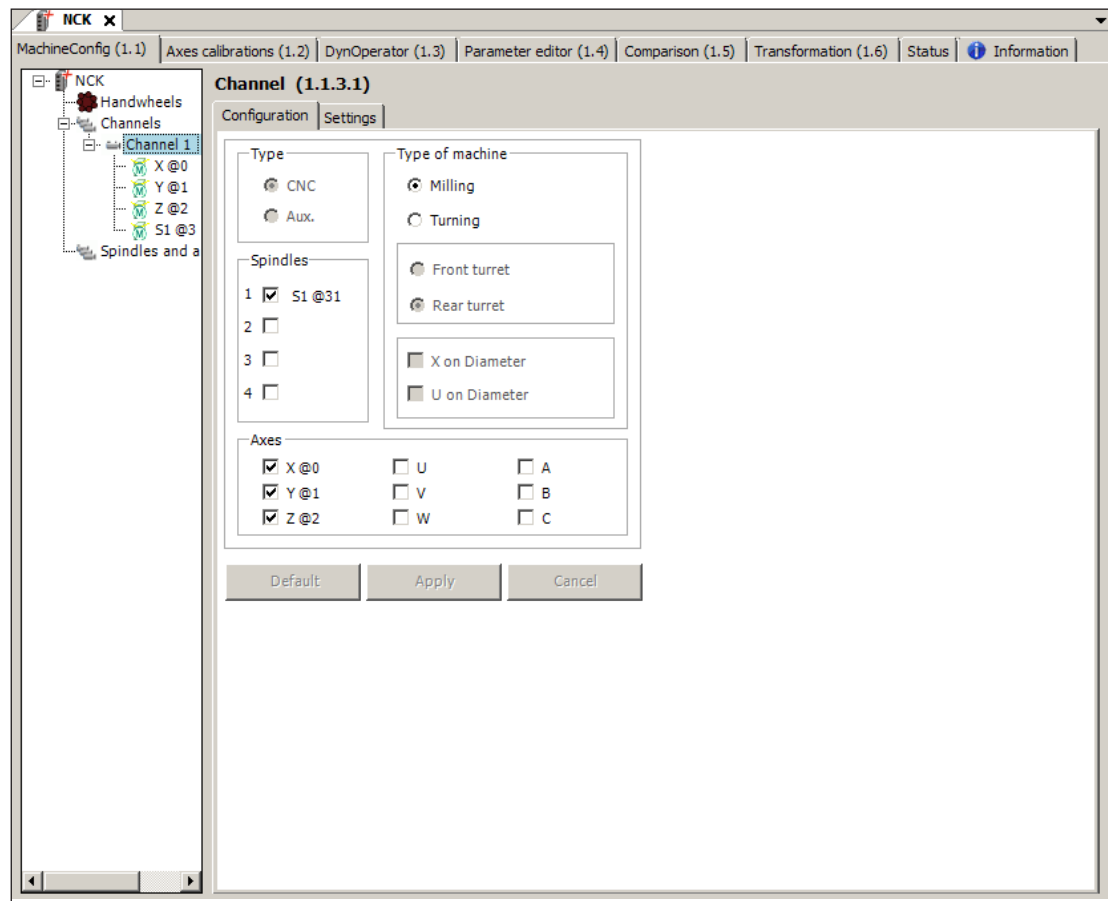
The numbering of the handwheels is totally independent of axes or spindles numbering.



3.5.2 MachineConfig TAB: Channel - Configuration

In addition to defining axes and spindles of each channel, this tab is used to define the type of channel (NCK or Auxiliary) as well as its type (Milling/Turning) and the specificities of the turning type.

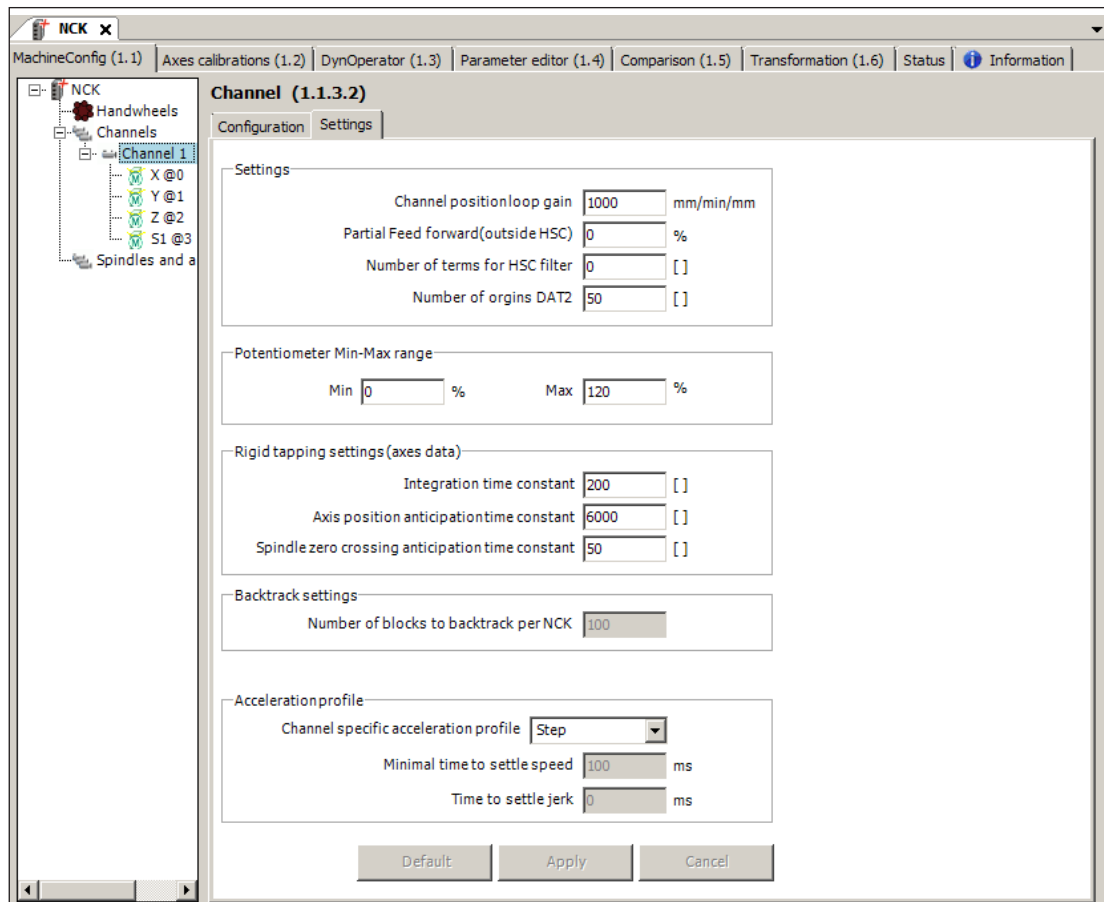
The possibilities are function of the options defined in the option tab of the NCK editor.



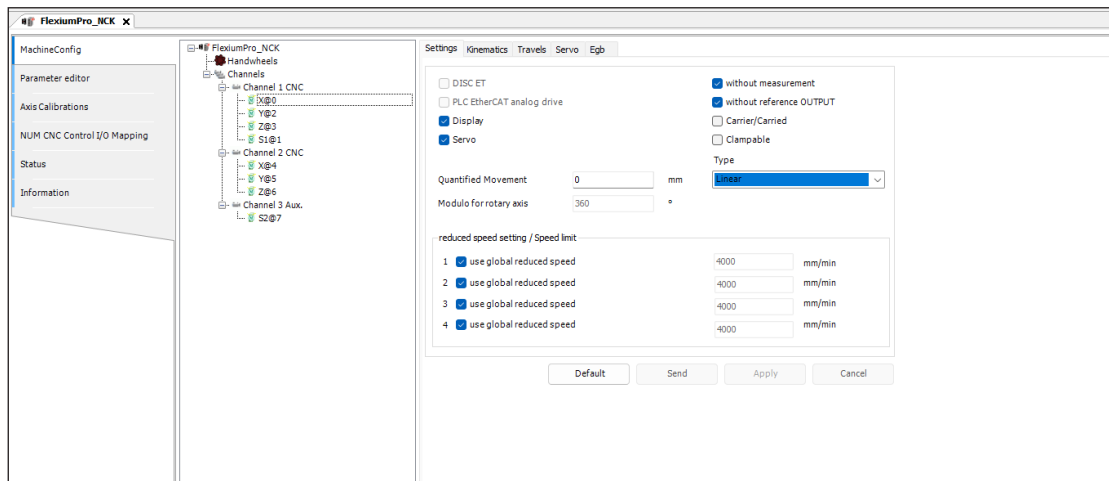
An auxiliary channel can only execute program %9998 and does not provide the full set of functions (see chapter 5.2).

3.5.3 MachineConfig TAB: Channel - Settings

Permits to define the time constant (gain) of each individual channel as well as feed forward, overrides and rigid tapping parameters.



3.5.4 MachineConfig TAB: Axes - Settings



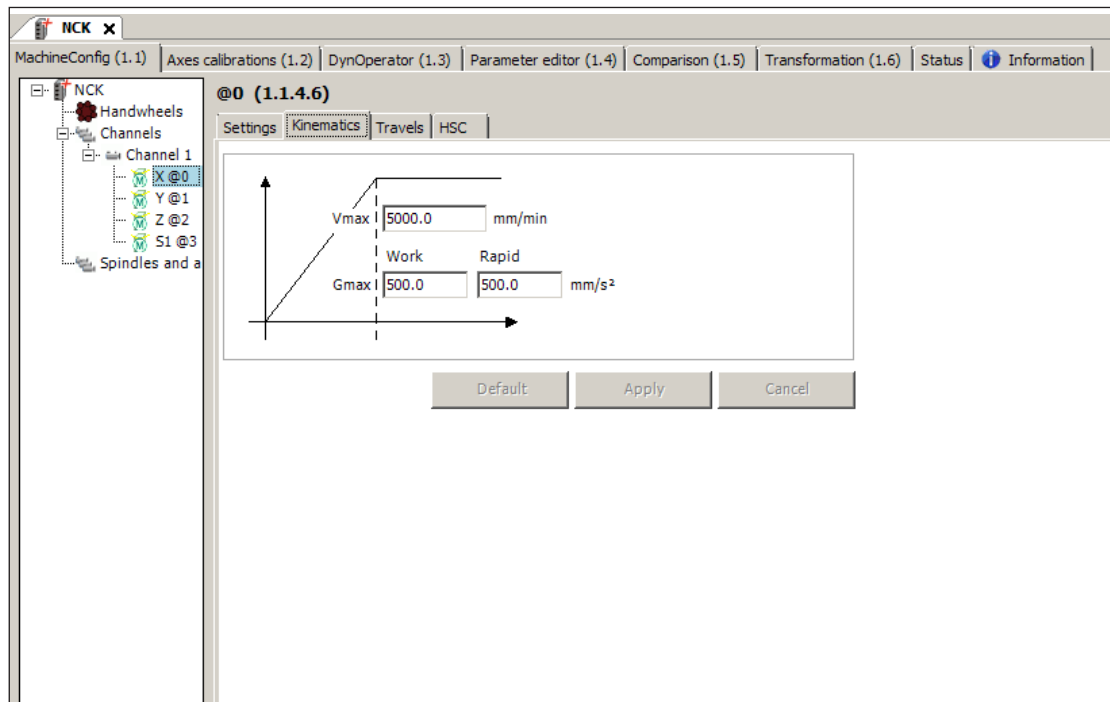
The check box Disc ET is filled automatically according to the type of drive connected.

Main axes settings

<i>Servo</i>	Means that the axis is linked to an interpolator.
<i>With reference output</i>	There is a command sent to the drive. This is valid for Digital axes and Analog axes.
<i>Carrier/Carried</i>	Defines couples of parallel linear axes. (see FlexiumPro Programming Manual M00073)
<i>Clampable</i>	Axes with a mechanical clamping. (see M10/M11 in FlexiumPro Programming Manual M00073).
<i>Rotary</i>	Axes can perform an infinite number of rotations.
<i>Rotary limited excursion</i>	The number of turns is limited (can be less than one)
<i>Quantified movement</i>	Valid for linear axes as well as rotary. The positioning of such axis is on discrete values.
<i>Use global reduced speed</i>	Four safety speed are defined per channel. When the respective checkbox is not marked a particular value can be entered. By default the limit speed will be the one defined for the whole NCK.
<i>PLC EtherCAT analog drive</i>	The respective axis will be controlled by the PLC via the EtherCAT bus.

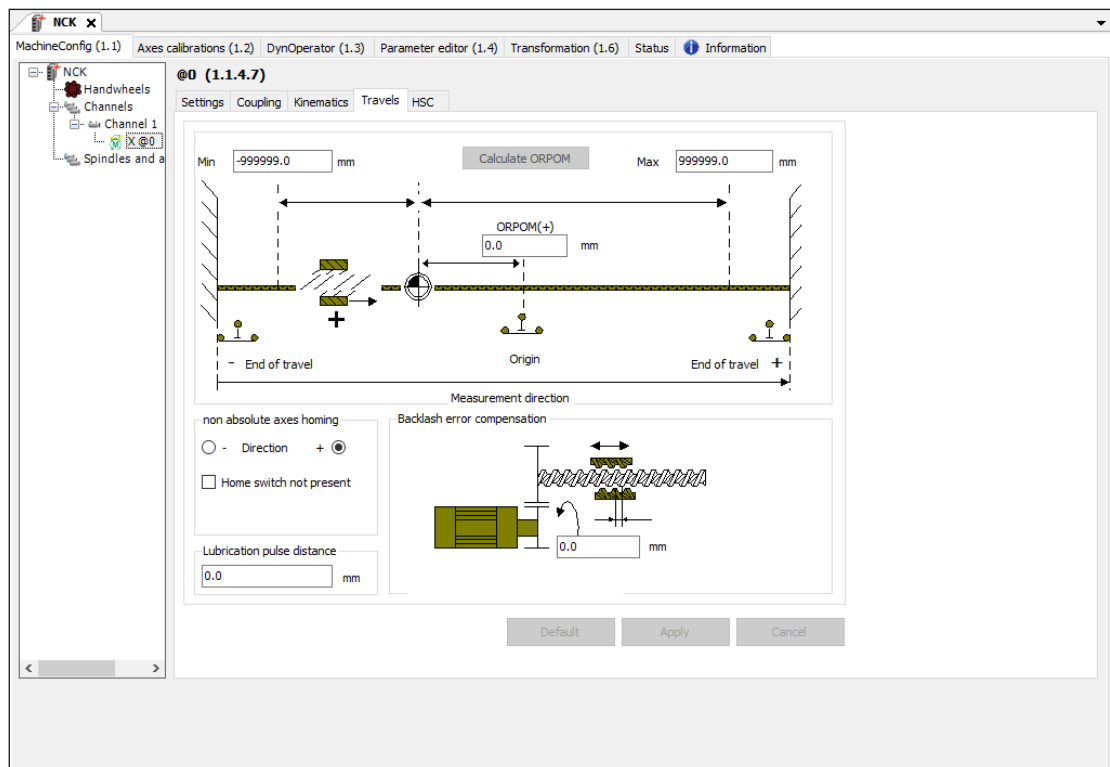
3.5.5 MachineConfig TAB: Axes - Kinematics

For each axis define the maximum speed as well as acceleration in G0 and G1.



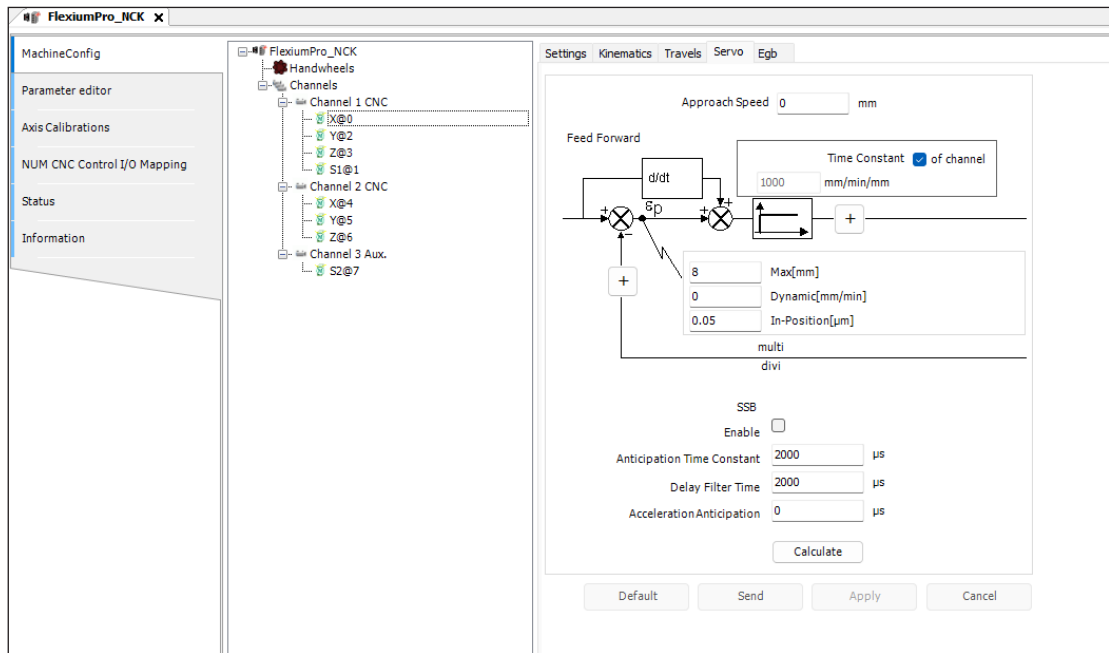
3.5.6 MachineConfig TAB: Axes - Travels

Defines travels, backlash and homing characteristics. In the PLC exchange area for lubrication once a certain travel has been accomplished, the distance is defined on this screen.



3.5.7 MachineConfig TAB: Axes - Servo-System

P22	In position window.
P23	Maximum following error. This value is not updated automatically. It should be modified according the machine builder desires.
P33	Approach speed: Highest step of speed between two blocks.
P57	Dynamic movement control.

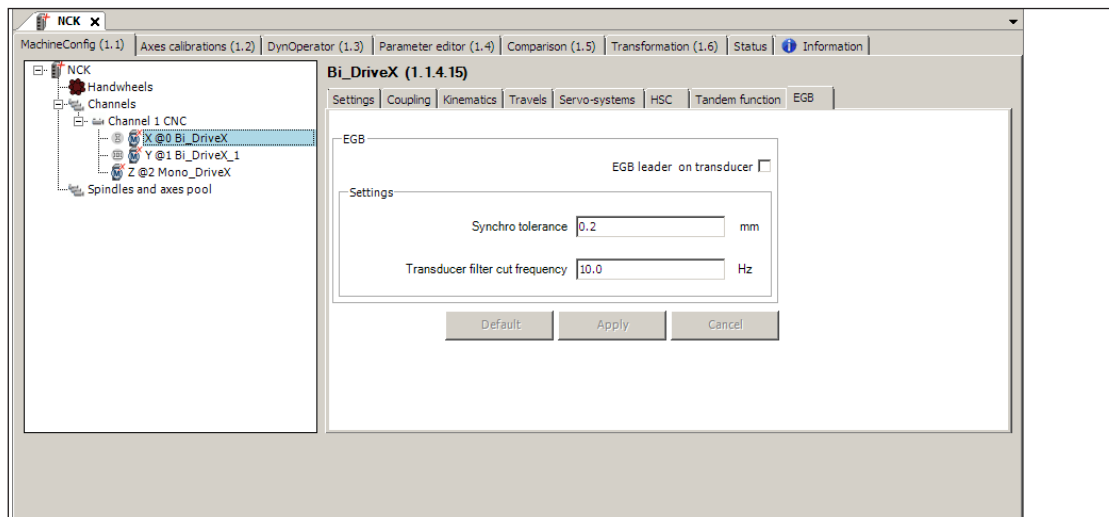


Position loop gain of channel : Unchecking this box allows for entering a specific gain on the particular axis.

3.5.8 MachineConfig TAB: Axes - EGB

This tab is linked to the MLEGB feature (Multi Level Electronic Gear Box).

It defines if the leader axis is running on the transducer (sensor) or according to the position reference issued by the interpolator.



3.5.9 Spindle declaration

The spindles are defined in the channel configuration.

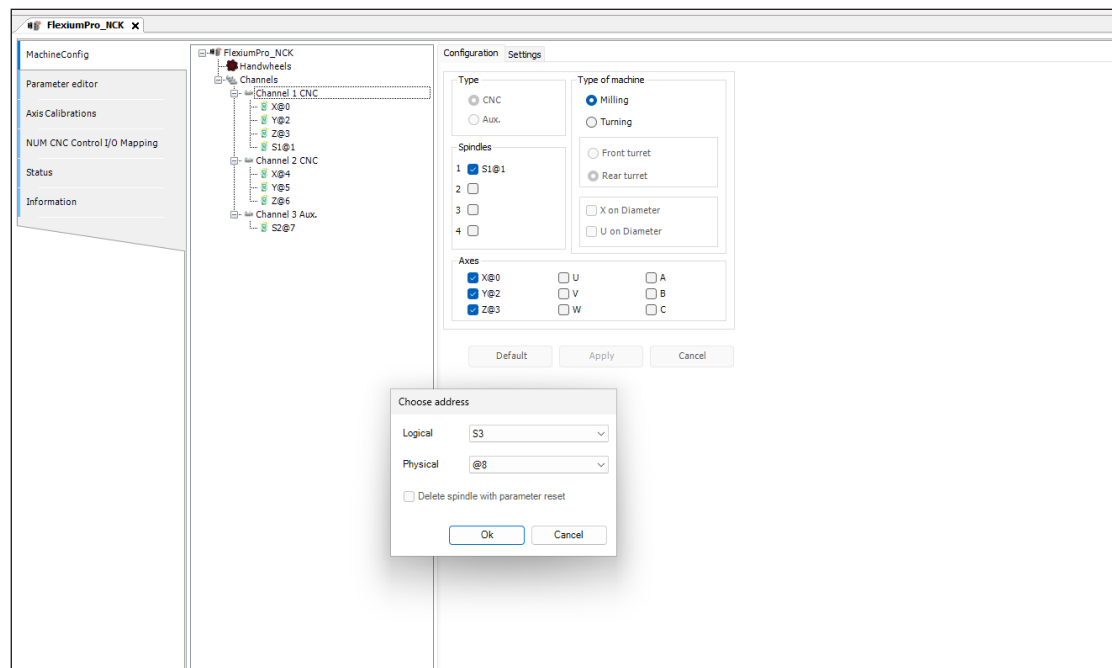
The spindle at position 1 becomes the main spindle of the channel, the others are auxiliary.

For information about the different possibilities please refer to the FlexiumPro Programming Manual M00073. This power-up affectation can be changed by part program at any time.

For each spindle, you have to define its logical address as well as its physical address.

The logical address (1 to 32) is used to select a spindle for programming. This address is not related to any other address and can be freely selected.

The physical address relates to the drive controlling the spindle. Any address from 0 to 31 not already used for an axis can be used.



3.5.10 MachineConfig TAB : Spindles - Settings

Declare the settings for each spindle.

Indexing at constant acceleration (deceleration) is the preferred choice.

The other possibility is indexing with deceleration according to a ramp on the reference.

Sensor coupled on the motor shaft will take the selected range in account.

As a general manner the spindle position increases in M04

The spindle resolution is linked to the encoder resolution in case of analog controlled spindle. When the spindle is controlled by a digital drive this resolution can be optimized by setting the encoder V311/V312 parameters.

The resolution should be calculated in such a way that at max speed the count difference between two scan does not exceed 60000. From 4.1.20 it is now possible to control the spindle in manual mode.

The speed is defined on this editor and the minimum override will be 0% whatever the minimum value is defined in the parameters. The spindle control will be possible if a range is selected. There is no check of concordance between actual range and programmed range.

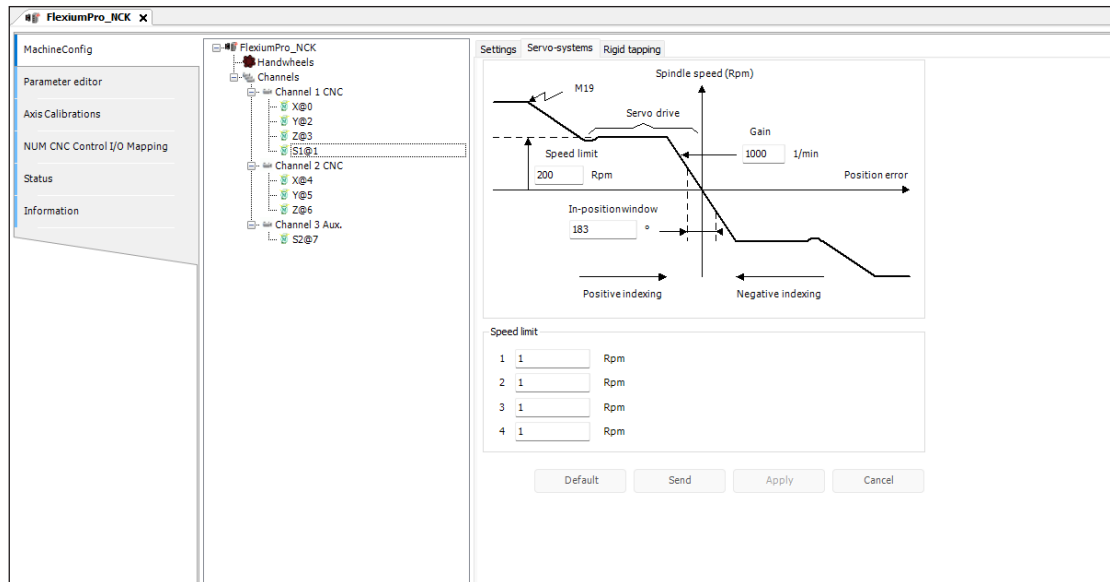
This editor also permits to define the spindle acceleration as well as the spindle origin.

3.5.11 MachineConfig TAB: Spindles - Servo-systems

This editor permits to set the parameters related to spindle indexing.

Refer to the FlexiumPro Parameters manual M00068 for additional information about the signification of the different parameters.

It is also possible to define up to four Spindle speed limits (for safety). The selection of a particular limit is done by the PLC and can be controlled by a drive fitted with the SAMX safety board.

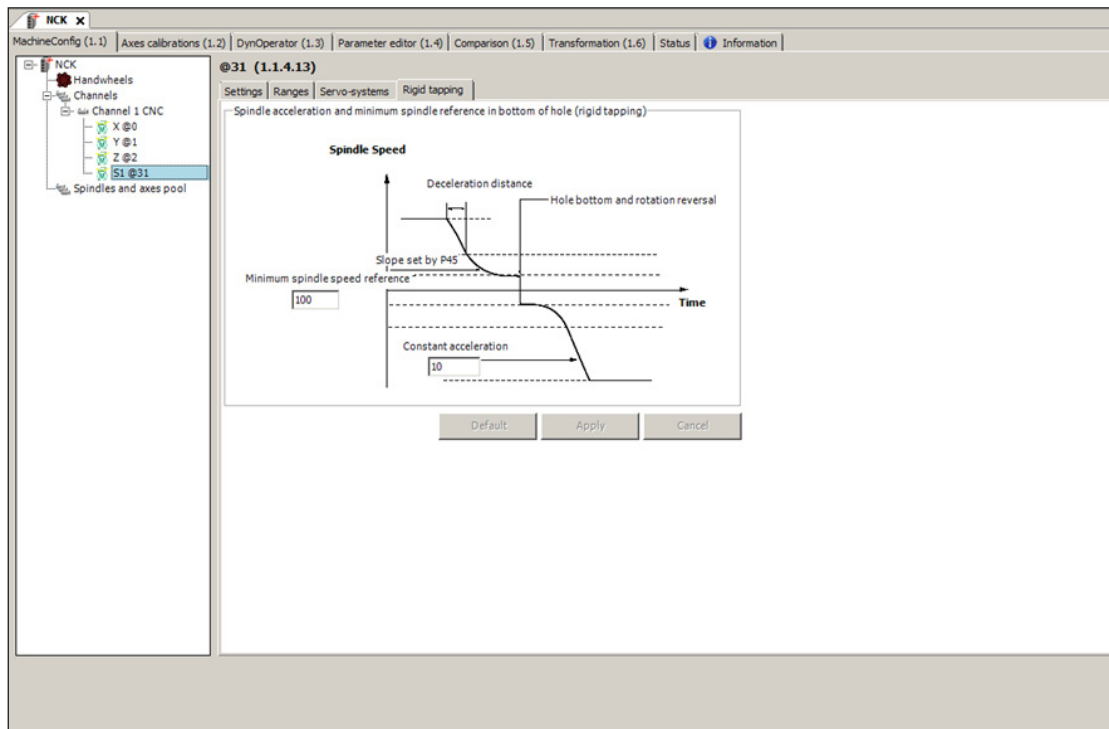


3.5.12 MachineConfig TAB: Spindles - Rigid tapping

The standard servo parameters (integral gain, Anticipation and zero anticipation) generally apply for rigid tapping; therefore they are not repeated here.

This editor allows for setting the deceleration at the bottom of the tapped hole (rev/s²). It should be set so that there is no risk of position overshoot.

The minimum spindle speed reference (given in 1/10000 of the max speed of the range) must be set in order to have a minimal speed (for immediate inversion) but no possibility of spindle stopped; as the axis is synchronized to the spindle a spindle stopped will stop the motion and the rigid tapping cycle will be in dead lock.



3.6 I/OS Bus

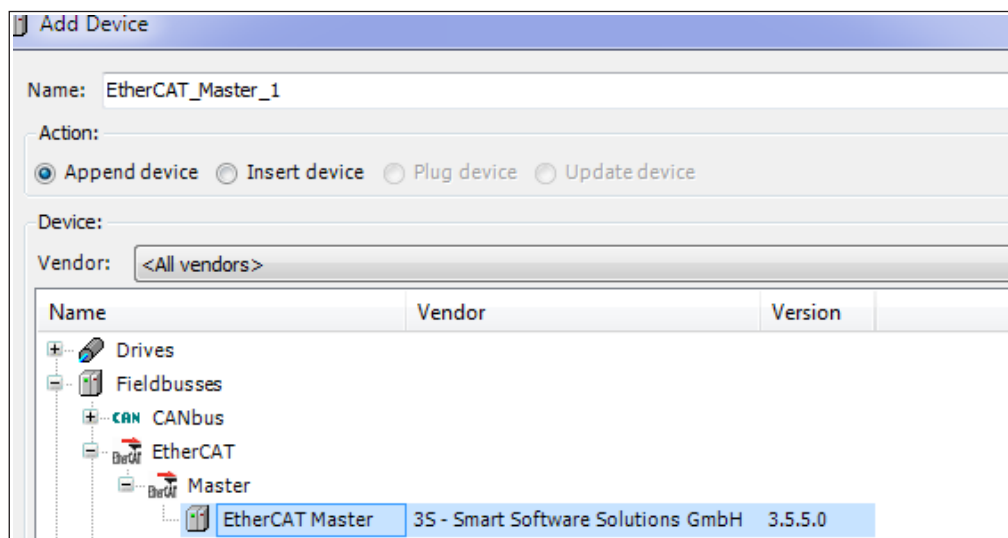
FlexiumPro provide two different types of I/O busses EtherCAT and CANOpen. Details on wiring and hardware set-up are available in the FlexiumPro Installation manual M00070.



Please note that EtherCAT bus is required for Integrated Safety.

3.6.1 EtherCAT settings

In the *Device tree*, right click under *Flexium RTS*; select *Add device* then add an EtherCAT master.



The minimal setting for the communication is the definition of the *MAC address* in the EtherCAT master editor. It can be entered manually, or determined by scanning as defined below.

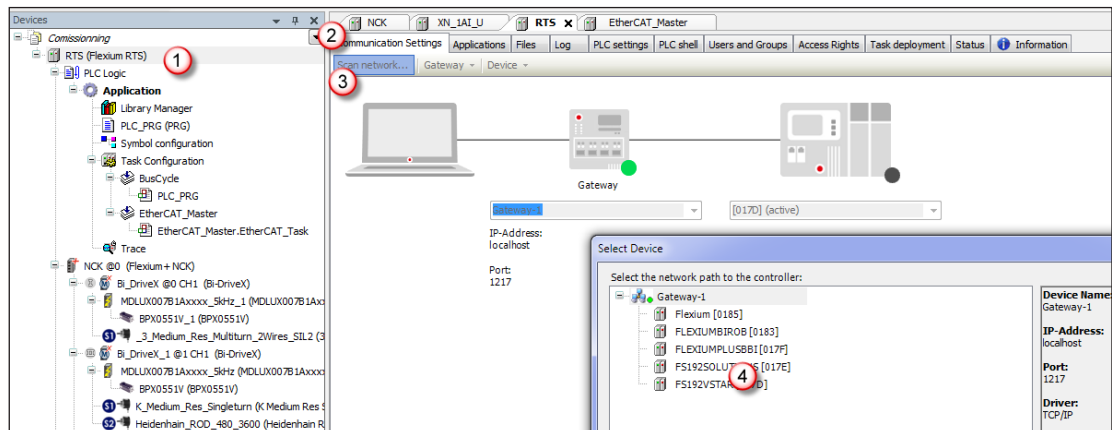
3.6.2 EtherCAT scanning

Scanning requires the following steps:

- Define a communication channel between FXTools and PLC
- Browse for EtherCAT MAC Address
- Scan the bus configuration.

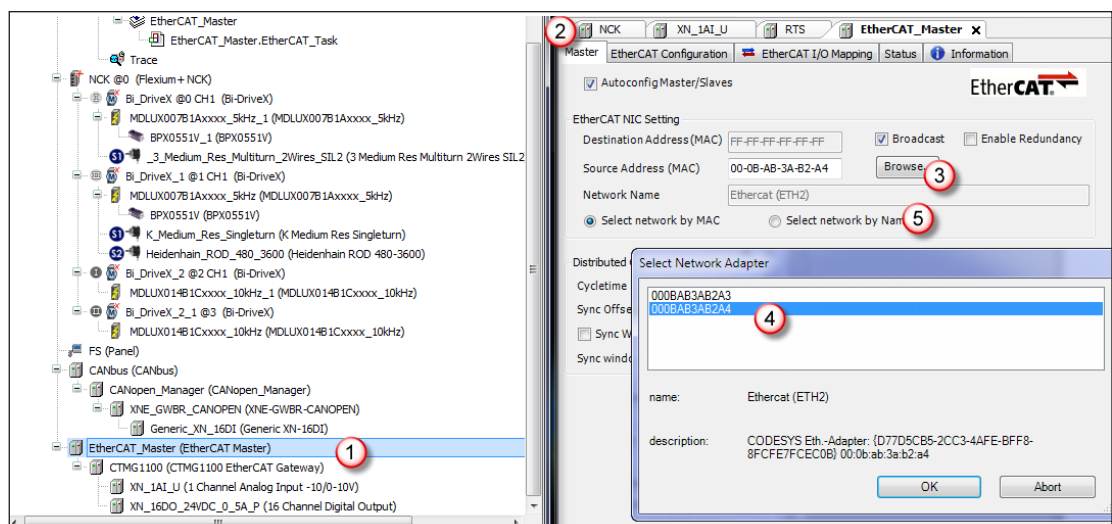
3.6.2.1 Defining a communication channel

- Double click on *Flexium RTS* (1)
- Select the *Tab communication settings* (2)
- Select *Scan Network* (3)
- Double click on the *selected target* (4).



3.6.2.2 Browse for EtherCAT Mac Address

- Double click on *EtherCAT Master* (1)
- Select the *Tab Master – Press Browse* (2) (3)
- On the new window opened select the *port* where the *EtherCAT bus* is connected (4)
- Click on **Select Network by Name** (5). This selection will make the application hardware independent.

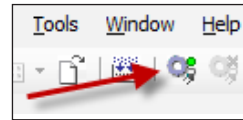


3.6.2.3 Scanning the bus configuration

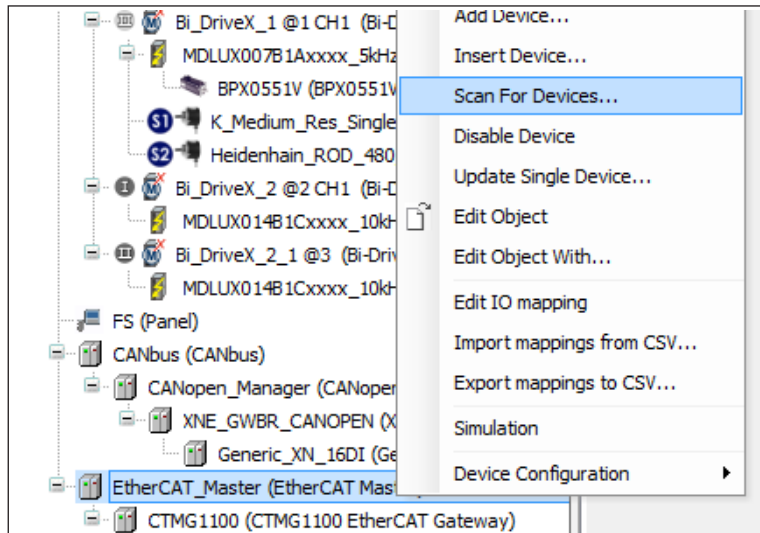
It is possible to define the bus configuration manually by the command 'Add Device'. However, when possible it is easier to make an automatic recognition.

This is done in two steps:

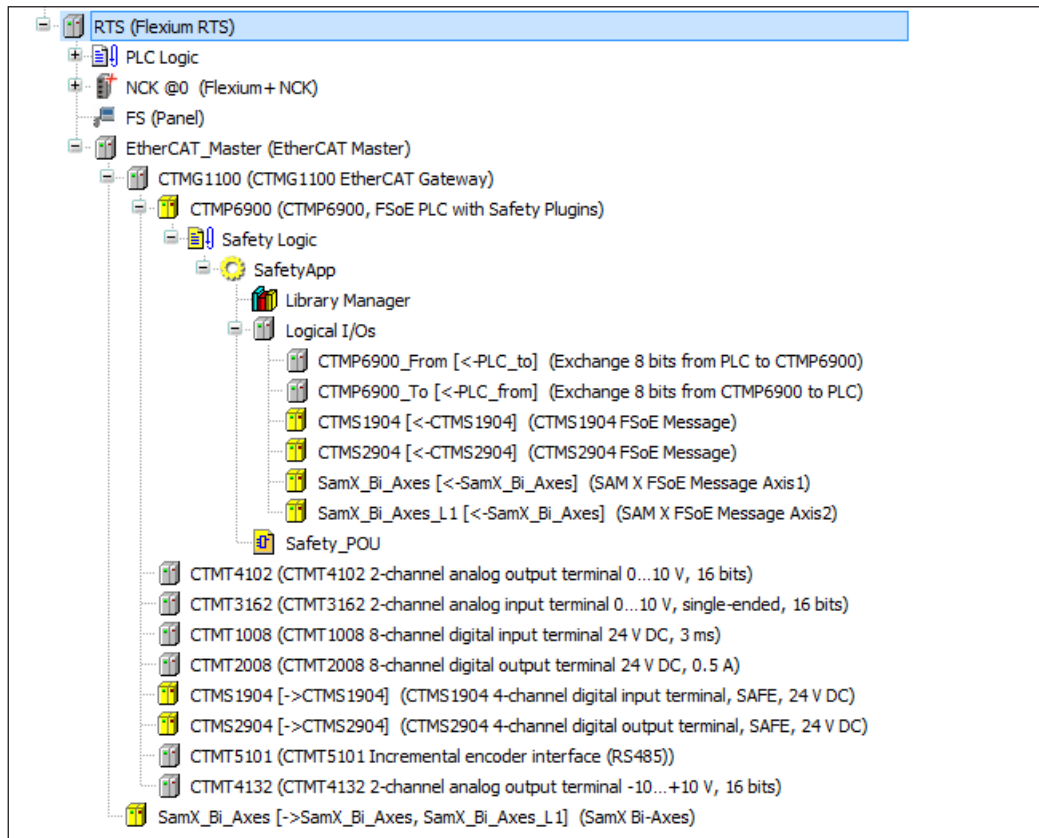
- Login with the PLC (*ALT + F8*) or *Menu OnLine* or



- Once *OnLine*, a right click on the *EtherCAT Master* will propose to scan the configuration.



An example of configuration is given below: (FlexiumPro with NUM SAMX)



3.6.3 Checking the bus configuration

Like for any IO system, it is necessary to check the functionality of the bus.

The libraries allow for extensive checking. A very minimal configuration is shown below. It will return the number of modules found as well as the status of the first three. This should obviously be extended to all modules present.



This example is given as a guideline only without warranty from NUM. It must be adapted to your specific configuration and verified extensively.

Expression	Type	Value	Comment
pSlave	POINTER TO ETCSlave	16#00000000	
NbSlavesOK	INT	9	Number of modules on the bus
Status_Node	ETC_SLAVE_STATE	ETC_SLAVE_OPERATIONAL	Status Gateway CTMG1100
Status_I0_7	ETC_SLAVE_STATE	ETC_SLAVE_OPERATIONAL	Status first set of Inputs
Status_Q0_7	ETC_SLAVE_STATE	ETC_SLAVE_OPERATIONAL	Status first set of Outputs

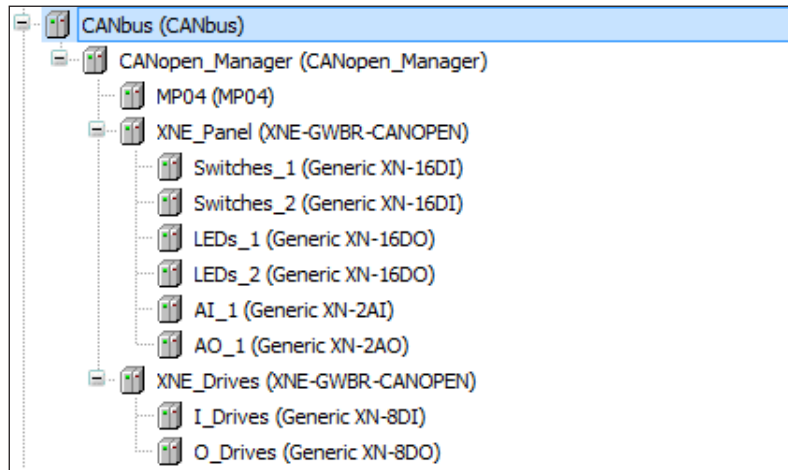
```
1 // Counts the number of slaves
2 // Required for update of wState
3
4 pSlave[16#00000000] := EtherCAT_Master.FirstSlave;
5 NbSlavesOK[9] := 0; // Number of slaves found
6
7 WHILE pSlave[16#00000000] <> 0 DO
8   pSlave^();
9   IF pSlave^.wState[???] = ETC_SLAVE_STATE.ETC_SLAVE_OPERATIONAL THEN
10     NbSlavesOK[9] := NbSlavesOK[9] + 1;
11   END_IF
12   pSlave[16#00000000] := pSlave^.NextInstance;
13 END_WHILE
14
15 // Check status of the first three modules including Gateway
16 // Correct status should be ETC_SLAVE_OPERATIONAL
17 Status_Node[ETC_SLAVE] := CTMG1100.wState[ETC_SLAVE];
18 Status_I0_7[ETC_SLAVE] := CTMT1008.wState[ETC_SLAVE];
19 Status_Q0_7[ETC_SLAVE] := CTMT2008.wState[ETC_SLAVE];
20
21 ETC_SLAVE_STATE.ETC_SLAVE_OPERATIONAL
22 'ETC_SLAVE_STATE.ETC_SLAVE_OPERATIONAL' represents raw value '8'
23
24 RETURN
```

3.7 CANOpen settings

3.7.1 Add CANbus

- Right click on 'Flexium RTS' and then 'Add Device'
- Select Fieldbuses and double click on CANbus
- Keep the Window Add Device open and click on CANBus in the Device tree
- Select CANOpen Manager then double click on CANOpen Manager
- Keep building your I/O configuration via the window Add Device.

A typical structure can be as below:



If a MP04 is used, only 125000 or 500000 bits/s are possible.

3.7.2 CANbus settings

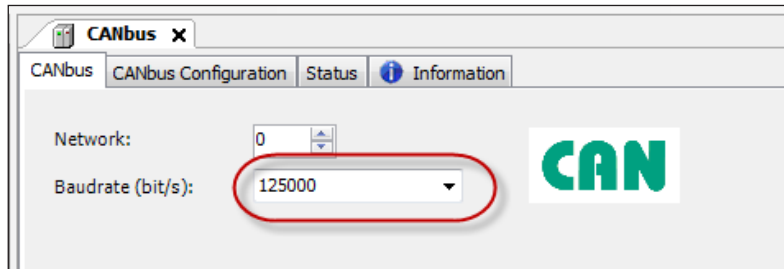


Unlike EtherCAT CANOpen requires a bit rate setting as well as modules addressing.

3.7.2.1 Baudrate

The baud rate is generally defined by dip switches on each module. The bus will not work correctly if one single module is set with a different value than the others.

To set the master baud rate double click on *CANbus* in the device tree.

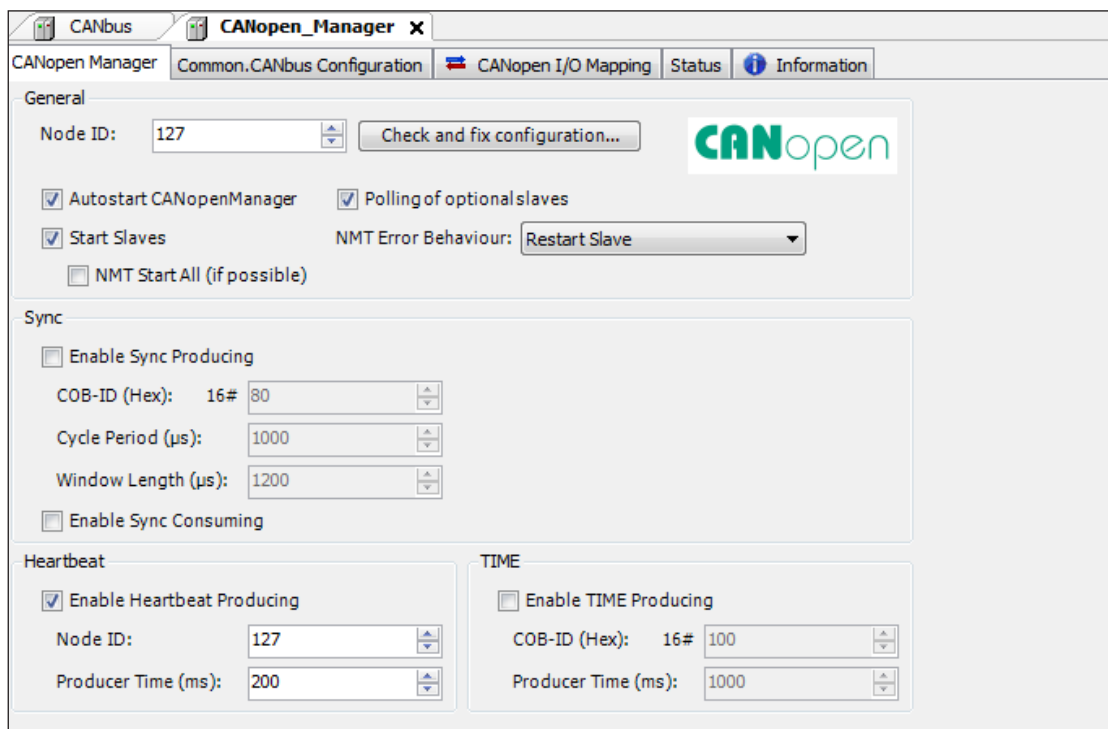


- If a MP04 is used, only 125000 or 500000 bits/s are possible.
- In general the network number does not need to be changed.

3.7.2.2 Addresses

CANOpen Manager needs an address (*Node ID*); the default 127 is generally good.

Checking *Enable Heartbeat* producing can be used for a peripheral to detect that the bus has been broken and therefore enter in a safe state.



As well each Gateway also requires a node address. The address set in the device tree should match the physical address defined on the gateway itself (dip switches).

The screenshot displays the CANopen Manager configuration window. The 'General' tab is active, showing the following settings:

- Node ID:** 1
- SDO Channels:** 1/4 active
- Enable Expert Settings:** ☒
- Optional Device:** ☐
- Autoconfig PDO Mapping:** ☒
- Create all SDOs:** ☐
- No initialisation:** ☐
- Reset Node:** Sub:002
- Enable Sync Producing:** ☐

The **Nodeguarding** section shows:

- Enable Nodeguarding:** ☐
- Guard Time (ms):** 0
- Life Time Factor:** 0

The **Heartbeat** section shows:

- Enable Heartbeat Producing:** ☒
- Producer Time (ms):** 200
- Heartbeat Consuming:** 0/1 active

The **Emergency** section shows:

- Enable Emergency:** ☒
- COB-ID:** \$NODEID+16#80

The **TIME** section shows:

- Enable TIME Producing:** ☐
- COB-ID (Hex): 16#** 100
- Enable TIME Consuming:** ☐

The **Checks at Startup** section shows:

- Check Vendor ID:** ☐
- Check Product Number:** ☐
- Check Revision Number:** ☐



As for the EtherCAT bus it is mandatory to control the bus functionality using for example the Heartbeat signal.

Page deliberately empty.

Summary

4	PLC Programming	65
4.1	General	65
4.1.1	Tasks	65
4.1.2	POUs	66
4.1.3	Variables	66
4.1.4	Languages	67
4.1.4.1	Affectation	67
4.1.4.2	Operations (List is not exhaustive)	67
4.1.4.3	Conversion	68
4.1.4.4	Set - Reset	68
4.1.4.5	Test	68
4.1.4.6	Loops	68
4.1.4.7	Case	69
4.2	CNC <=> PLC Exchange Zone	70
4.2.1	Structure RCNC	71
4.2.1.1	Structure RCNC.Axis[.].	72
4.2.1.2	Structure RCNC.Axis[.].DigitalAxis.	74
4.2.1.3	Structure RCNC.Axis[.].SAMX.	75
4.2.1.4	Structure RCNC.Channel[.].	76
4.2.1.5	Structure RCNC.Channel[.].DecodeM.	81
4.2.1.6	Structure RCNC.Channel[.].DecodeM.General.	81
4.2.1.7	Structure RCNC.Channel[.].DecodeM.Spindle[...].	82
4.2.1.8	Method RCNC.Channel[.].ReadEParameter(.....	82
4.2.1.9	Method RCNC.Channel[.].ReadMaximumDynamicTravelLimits(.....	83
4.2.1.10	Method RCNC.Channel[.].ReadMinimumDynamicTravelLimits(.....	83
4.2.1.11	Method RCNC.Channel[.].ReadPosition(.....	83
4.2.1.12	Array RCNC.E10000[.].	83
4.2.1.13	Array RCNC.E30000[.].	84
4.2.1.14	Array RCNC.E33000[.].	84
4.2.1.15	Array RCNC.E36000[.].	84
4.2.1.16	Array RCNC.E37000[.].	84
4.2.1.17	Structure RCNC.General.	85
4.2.1.18	Structure RCNC.IO.	88
4.2.1.19	Method RCNC.GetParameterType(.....	89
4.2.1.20	Method RCNC.ReadParameter(.....	89
4.2.1.21	Structure RCNC.Spindle[.].	89
4.2.1.22	Structure RCNC.Spindle[.].Load.	90
4.2.2	Structure Simulation.	91
4.2.2.1	Structure Simulation.RCNC.	91
4.2.2.2	Structure Simulation.WCNC.	91
4.2.2.3	Structure Simulation.WCNC.Axis[.].	91
4.2.2.4	Structure Simulation.WCNC.Axis[.].DigitalAxis.	92
4.2.2.5	Structure Simulation.WCNC.Axis[.].SAMX.	93
4.2.3	Structure WCNC.	93
4.2.3.1	Structure WCNC.Axis[.]	94
4.2.3.2	Structure WCNC.Axis[.].DigitalAxis.	95
4.2.3.3	Method WCNC.Axis[.].DigitalAxis.MotorBrakeTest	95
4.2.3.4	Structure WCNC.Channel[.].	96
4.2.3.5	Structure WCNC.Channel[.].Spindle[...].	99
4.2.3.6	Method WCNC.Channel[.].WriteEParameter(.....	100

4.2.3.7	Method WCNC.Channel[...].WriteMaximumDynamicTravelLimit(	100
4.2.3.8	Method WCNC.Channel[...].WriteMaximumDynamicTravelLimits(	100
4.2.3.9	Method WCNC.Channel[...].WriteMinimumDynamicTravelLimit(	101
4.2.3.10	Method WCNC.Channel[...].WriteMinimumDynamicTravelLimits(	101
4.2.3.11	Array WCNC.E20000[...]	102
4.2.3.12	Array WCNC.E40000[...]	102
4.2.3.13	Array WCNC.E43000[...]	102
4.2.3.14	Array WCNC.E46000[...]	103
4.2.3.15	Array WCNC.E47000[...]	103
4.2.3.16	Structure WCNC.General.	104
4.2.3.17	Structure WCNC.General.LockIncJog.	108
4.2.3.18	Structure WCNC.General.LockSpecificMode.	108
4.2.3.19	Structure WCNC.Handwheel[...].	108
4.2.3.20	Structure WCNC.IO.	109
4.2.3.21	Structure WCNC.IO.AnalogOutput[...].	109
4.2.3.22	Structure WCNC.IO.DigitalOutput[...].	109
4.2.3.23	Structure WCNC.Spindle[...].	110
4.2.3.24	Structure WCNC.Spindle[...].Load.	110
4.2.3.25	Method WCNC.WriteParameter(	111
4.3	Miscellaneous Read Data	112
4.4	Methods and Functions	113
4.4.1	Method LoadToolNumber(	113
4.4.1.1	Syntax	113
4.4.1.2	Parameters	113
4.4.1.3	Result return codes	113
4.4.2	Method ToolDyn(	114
4.4.2.1	Syntax	114
4.4.2.2	Parameters	114
4.4.2.3	Result return codes	114
4.4.2.4	Example	115
4.4.3	Probing	116
4.4.3.1	Method TouchProbe.ChannelsCfg(	116
4.4.3.2	Method TouchProbe.HardwareCfg(	117
4.4.3.3	Method TouchProbe.InputReading(	118
4.4.3.4	Structure RCNC.ProbeInput[...].	118
4.4.3.5	Example	119
4.4.4	Messages and Warnings	120
4.4.4.1	Text Management in the Tools	120
4.4.4.2	HMI	121
4.4.5	PLC Message Management	121
4.4.5.1	Define the characteristics of the receipt classes (once)	121
4.4.5.2	Activate the messages when necessary	121
4.4.5.3	Checking which messages are active	122
4.4.5.4	Checking a reaction class	122
4.4.5.5	Checking a receipt class	122
4.4.5.6	Cleaning a class of messages	122
4.4.6	Operator's message in different languages	123
4.4.7	Warnings	123
4.5	Analog axes via EtherCAT	125
4.5.1	General configuration	125
4.5.2	Interface module configuration	127
4.5.2.1	Configuration for TTL interface (CTMT 5101)	127
4.5.2.2	Configuration for SSI interface (EL 5002)	132
4.5.2.3	Configuration for EnDat interface	134
4.5.3	PLC program	136
4.5.4	Parameterization	137

4 PLC Programming

The PLC handles the exchange with the NCK as well as the machine environment.

The exchange with the NCK is made via a dedicated memory zone exchanged every $P99N0 \times P50 \mu s$ (see machine parameters manual).

4.1 General

The PLC programming conforms to the standard 61131-3. The manual will detail below the necessary knowledge to create a PLC program.

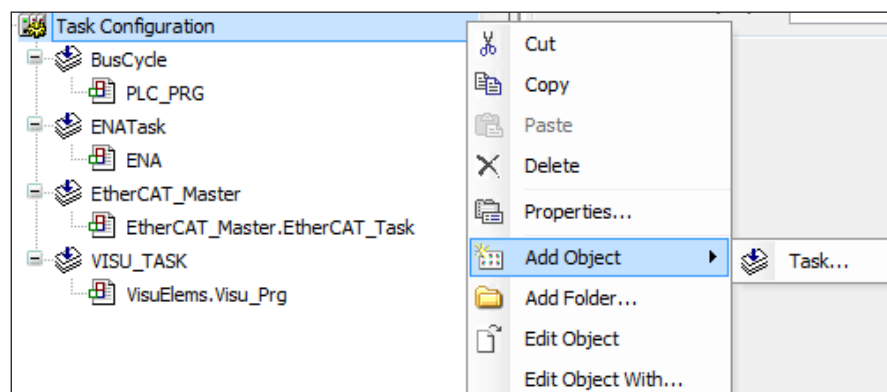
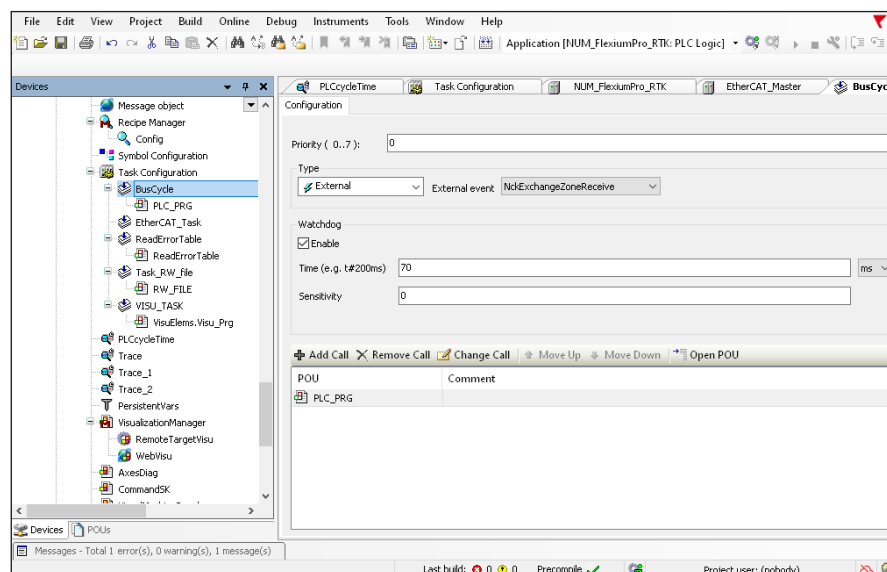
For additional info refer to the help of FlexiumPro Tools as well as the literature about this standard which is widely available.

4.1.1 Tasks

A blank project created by FlexiumPro Tools contains one task:

Bus_Cycle which runs the code written in *PLC_PRG*. *Bus_Cycle* presents the following characteristics:

The plc cycle task is linked to $P50 \times P99N0$ by specific setting in the Task configuration. It is possible to add additional tasks (up to 100) by the command *Add Object* in *Task Configuration*:



4.1.2 POU

POUs (Program Organizational Units) are modules in which PLC code is written.
The number of POU is not limited.

POUs can be declared under *Application* in the device tree. They will be available for the particular application; they can also be declared in the POU window, in which case they will be available for all applications in the project.

Each POU can be written in a particular language which will be explained in the chapters [4.1.4](#).

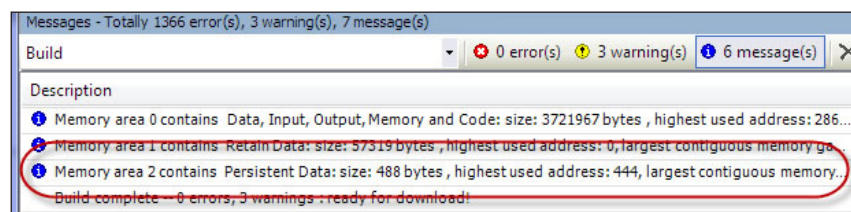
4.1.3 Variables

A PLC programs deals with different types of variables. The 61131-3 standard describes a large set of variables among them the most commonly used are:

Keyword	Size (Bits)	Range
BOOL	8	0,1, TRUE, FALSE
BYTE	8	0 .. 255
WORD	16	0 .. 65535
DWORD	32	0 .. 4 294 967 295
LWORD	64	0 .. 2 ⁶⁴ -1
SINT	8	-128 .. 127
USINT	8	0 .. 255
INT	16	-32768 .. 32767
UINT	16	0 .. 65535
DINT	32	-2147483648 .. 2147483647
UDINT	32	0 .. 4 294 967 295
LINT	64	-2 ⁶³ .. 2 ⁶³ -1
STRING	Variable	from 1 to 255 characters
REAL	32	1.401e-45 to 3.403 e+38
LREAL	64	2.225e-308 to 1.797e+308

Flexium handles two types of variables:

- Standard and Retain Persistent.
Standard variables are initialized (cleared) at power up and on Full download, they are only saved during an Online change.
- Retain Persistent variables are saved on any situation except Reset Origin.
Retain Persistent variables are backed up on NVRAM when the system provides a CAN board (about 57Kb available); in other cases they are saved in the file *Application.ret* which is created and updated during system shutdown.
- The size of Retain Persistent Variable segment can be shown after compilation in the Message window.



4.1.4 Languages

The following languages (IEC standard IEC 61131) are provided via dedicated editors:

LD	Ladder Logic Diagram (LD)
ST	Structured Text
SFC	Sequential Function Chart
FBD	Function Block Diagram
IL	Instruction List (IL)

The most commonly used are *Ladder Logic* and *Structured Text*. For detailed information please refer to the Help of FlexiumPro Tools.

In the rest of the manual examples will be given in Structured Text. The main instructions are given below.

Please note that in ST language, an instructions always end with the character *semicolon*.



Even though the variables shown are always A, B and C there is no relation between them in the examples below (in particular the types are different).

4.1.4.1 Affectation

```
// Affectation :

A := B;           // The content of B is stored into A
```

4.1.4.2 Operations (List is not exhaustive)

```
// Operations

A := B + C;       // Addition Result in A

A := B OR C;      // Logical OR - Result in A

A := SHL(B,N);    // Shift Left of B by n positions - result in A

A := SIN(B)       // Trigonometric - Sine
```

4.1.4.3 Conversion

```
//      Conversion

A := BYTE_TO_BCD (B)    // B converted from Byte to BCD and stored into A
```

4.1.4.4 Set - Reset

```
//      Set - Reset

A S= (B=4)              // If B = 4 then A is set - Otherwise nothing occurs

A R= (B=2)              // If B = 2 then A is reset - Otherwise nothing occurs
```

4.1.4.5 Test

```
// Test

IF (A=B) THEN

    // List of operations to execute if A=B

ELSE    // (Optional)

    // List of operations to execute otherwise

END_IF
```

4.1.4.6 Loops

```
// Loop (may not be executed if condition is false from start)
WHILE (A=B) DO

    // List of operations to execute as long as A = B

END_WHILE

//Other kind of loop (executed at least once)
REPEAT

    // List of operations to execute

UNTIL (A=B)
```

4.1.4.7 Case

```
CASE A OF
  1, 5: B1 := TRUE;
      B3 := FALSE;    // List of instructions if A equals 1 or

  2   : B2 := FALSE;    // List of instructions if A = 2

  10..20: BOOL1 := TRUE; // List of instructions if A equals 10 to 20

ELSE
  B1 := NOT B2;        // List of insns in all other cases
  B3 := B1 OR B4;

END_CASE;
```

Even though the variables shown are always A, B and C they are different and of different types with each example.

1

2

3

4

5

6

7

A

4.2 CNC <-> PLC Exchange Zone

The exchange zone contains the following area.

This structure can change according to the version.

Name	Type	Description
CommunicationReady	Prop.	See chapter 4.3
CycleCounterNCK	Prop.	See chapter 4.3
CycleCounterPLC	Prop.	See chapter 4.3
LoadToolNumber	Method	See chapter 4.4.1
RCNC	Struct	See chapter 4.2.1
Simulation	Struct	See chapter 4.2.2
ToolDyn	Method	See chapter 4.4.2
TouchProbe	Struct	See chapter 4.4.3
WCNC	Struct	See chapter 4.2.3
SycEParameterExchange	Prop.	

When this flag is true, the exchange data between NCK and PLC for Exxxxx is synchronized; if this flag is false, NCK does not wait for the exchange data to be completed, but continues with the next block.

The different tables below list all exchange variables.

- **P**: identifies a flag whose action is triggered on the raising edge (P as Pulse)
- **S**: identifies a flag or a variable whose action is active on the status (S as status).

The following variable must necessarily be handled in the PLC otherwise no movement will be possible.

```
FlexiumPro_NCK.WCNC.General.FeedEnable:= TRUE;  
FlexiumPro_NCK.WCNC.General.NoStopBlockEnd:= TRUE;  
FlexiumPro_NCK.WCNC.Channel[x].AckMFunction:= TRUE;  
FlexiumPro_NCK.WCNC.Channel[x].ChannelEnable:= TRUE;  
FlexiumPro_NCK.WCNC.Channel[x].EndExtMovement:= TRUE;  
FlexiumPro_NCK.WCNC.Channel[x].FeedEnable:= TRUE;  
FlexiumPro_NCK.WCNC.General.Mode:= 7;
```

Conventions

The following chapter uses the following conventions:

START	Refers to the start command.
CYCLE START	Refers to the start feedback.
STOP	Refers to the stop command.
CYCLE STOP	Refers to the stop feedback.
RESET	Refers to the reset command.
CYCLE RESET (pulse)	Refers to the reset feedback.

The use of the general or channel specific variable depends about the context.

In the examples given below, the term NCK represent the name of the object NCK, it can be different according to your application. In particular for each new project, it is generally *FlexiumPro_NCK*.

4.2.1 Structure **RCNC**.

Name	Type	Description
Axis[.].	Array of Struct	See Chapter 4.2.1.1
Channel[.].	Array of Struct	See Chapter 4.2.1.4
E10000[x]	Array of BOOL	See Chapter 4.2.1.12
E30000[x]	Array of LREAL	See Chapter 4.2.1.13
E33000[x]	Array of USINT	See Chapter 4.2.1.14
E36000[x]	Array of UINT	See Chapter 4.2.1.15
E37000[x]	Array of BYTE	See Chapter 4.2.1.16
General.	Struct	See Chapter 4.2.1.17
IO.	Struct	See Chapter 4.2.1.18
GetParameterType	Method	See Chapter 4.2.1.19
PLCString	STRING	Memory location where the PLC can read a string of up to 35 characters written by the NCK via the function _SETSPLC (refer to the FlexiumPro Extended programming manual M00075).
ProbeInput[x].	Struct	See Chapter 4.4.3.4
ReadParameter	Method	See Chapter 4.2.1.20
Spindle[.].	Array of Struct	See Chapter 4.2.1.21

4.2.1.1 Structure `RCNC.Axis[.].`

[.] Represents the axis physical address (0 to 31)

Syntax example: `A:= FlexiumPro_NCK.RCNC.Axis[4].Direction;`

Name	Type	Description
AssignedToChannel	INT	Indicates the which channel number uses the axis. In case of axis not in a channel the value is -1
Auxiliary	BOOL	Indicates if the axis is an auxiliary axis or CNC axis type
Clamp	BOOL	Indicates that the axis n° X can be clamped The axes that can be clamped are defined by parameter P8 and can be modified by E913xx. Refer to annex 10.6 for timings.
ControlledByOpDyn	BOOL	The axis movement is controlled by dynamic operators
DigitalAxis.	Struct	See chapter 4.2.1.2
Direction	BOOL	Set to True when the axis move is in the positive direction set to FALSE otherwise. The direction signal report the actual direction and memorize the last direction movement, to check the actual direction movement read this value with Moving signal.
DriveSimulation	BOOL	This axis is linked to a simulated drive.
EgbActFollower	BOOL	Linked to MLEGB function: Indicates that the axis is an active follower.
EgbEmeFollower	BOOL	Linked to MLEGB function: Indicates that the axis is a follower axis in Emergency.
EgbFwerSynczed	BOOL	Linked to MLEGB function: Indicates that the axis is a follower synchronized on his master.
FollowingTangency	BOOL	The axis follow a tangential path controlled by G748
HomeForced	BOOL	Indicates that the homing of this axis has been performed by a keyboard action. It may therefore be out of position.
LubricationPulse	BOOL	Indicates the time for a lubrication has occurred
MovementBlockedNeg	BOOL	The movement is blocked due to a possible collision in negative direction
MovementBlockedPos	BOOL	the movement is inhibited due to a possible collision in positive direction

Name	Type	Description
Moving	BOOL	This variable has two different behaviors according to the status clamping or not clamping. - For not clampable axes or clampable axes with M11 active, this flag will be set to TRUE as soon as the axis has started to move. When a motion is requested the motion is supposed started even if : - <i>FlexiumPro_NCK.WCNC.Channel[x].FeedOverride:= 0 or</i> - <i>FlexiumPro_NCK.WCNC.Channel[x].FeedEnable:= FALSE or</i> - <i>FlexiumPro_NCK.WCNC.Axis[x].Stop:= True.</i> - For clampable axes with M10 active each time an axis will start or stop moving in the transition between two blocks, the system will position the “<i>flags Moving</i>” according to the next status and wait for acknowledge from the PLC. (with <i>MFunctionAcknowledge</i>). (with <i>MFunctionAcknowledge</i>, Flexium+ compatibility mode) The new mode for clamping axis is using the plc signal <i>.WCNC.Channel[x].EndExtMovement</i>
NckControlSuspended	BOOL	Indicates that Torque or reference control are managed by FlexiumPro Tools irrespective of the PLC requests.
NotHomed	BOOL	Homing on the axis has been completed or not This flag is set to FALSE when homing is completed.
Rotating	BOOL	Linked to MLEGB function: Indicates that this axis is in permanent rotation.
RotatingSpeedReached	BOOL	Linked to MLEGB function: Indicates that this axis is in permanent rotation and has reached its speed.
SAMX.	Struct	See chapter 4.2.1.3
ServoCntrlDisabled	BOOL	Indicates that the axis is not servo controlled either by E910xx or the PLC flag DisableServoCntrl. The PLC servo control disable has priority over E910xx.
SpindleMode	BOOL	With this flag is true the NCK is asking to Drive the speed mode

1

2

3

4

5

6

7

A

4.2.1.2 Structure `RCNC.Axis[.].DigitalAxis`.

[..] Represents the axis physical address (0 to 31)

Syntax example: `A:= FlexiumPro_NCK.RCNC.Axis[8].DigitalAxis.DriveOk;`

Name	Type	Description
BrakeRelay	BOOL	Set to TRUE when the brake is released (energized). (brake management by the drive)
BrakeTestRunning	BOOL	Set to TRUE when the brake test is running.
BumpSuppressor	BOOL	Set to TRUE when the BumpSuppressor is active.
ConsWPMActive	BOOL	Set to TRUE when WPM software is connected in console mode.
ENUMCNTRLMODE	- Delta pos - Speed - Current	The value reports the drive operates status: Delta pos → position mode Speed → speed mode Current → current mode
DCBusPresent	BOOL	Set to TRUE when high voltage DC Bus is present.
DEMxDigitalOutput	BOOL	Set to TRUE by a DEMX (drive Embedded Macros – optional) application program.
DigitalAxisI2T	USINT	Percentage of the Drive I2T value.
DriveInRunState	BOOL	Set to TRUE when Drive firmware in execution.
DriveOK	BOOL	Set to TRUE when the drive is operational. (No alarm present)
EPS_EPSMActiv	BOOL	Set to TRUE when encoder Self-calibration is in progress.
MainsOn	BOOL	Set to TRUE when the main power supply is active.
MechPower	REAL	Mechanical power at motor shaft: Positive value means power to load Negative value means power from load.
MotorI2T	USINT	Percentage of I2T value of the motor.
MotorLoad	UINT	Percentage of actual torque current related to available torque current.
MotorMoving	BOOL	Set to TRUE when Motor speed is higher than +/- V41 or V42 drive parameters.
ParameterSet	USINT	Parameter set number. If 99 means default parameter set.
Parking	BOOL	Drive in parking state.
Prog01	BOOL	Status of programmable output 1 (see V140 to V145 drive parameter).
Prog02	BOOL	Status of programmable output 2 (see V150 to V155 drive parameter).
PWMCutOff	BOOL	Set to TRUE when NUM-STOX or NUM-SAMX intervention.
ReferenceEnabled	BOOL	Set to TRUE when the Reference (speed) command is enabled.

Name	Type	Description
CncDriveCommOk	BOOL	Set to TRUE when NCK <-> Drive communication is running. If this flag is false, all other Status and Extended Status information are meaningless.
SpeedReached	BOOL	Set to TRUE when the requested speed is reached. (Refer to V40 on M00066 manual)
TorqueEnabled	BOOL	Set to TRUE when the Drive torque is enabled.
Warnings	BOOL	Set to TRUE when the drive issues a Warning.



Status and ExtendStatus signals are guaranteed in their value only when the flag CncDriveCommOk is true.

4.2.1.3 Structure **RCNC.Axis[.].SAMX.**

[..] Represents the axis physical address (0 to 31)

Name	Type	Function description
EmergencyQuickStop	BOOL	Emergency stop requested by SAMX.
EmergencyStop-Status	BOOL	Emergency braking requested by SAMX.
SAMXOk	BOOL	SAMX operative.

4.2.1.4 Structure `RCNC.Channel [..]`.

[..] Represents the index number (1 to 32)

Syntax example: `B:= FlexiumPro_NCK.RCNC.Channel[2].AxesNotInPosition;`

Name	Type	Description
AxesPhysicalAddress[x]	Array of INT	Array of 9 integer corresponding to the relation between logical address and physical address. AxesPhysicalAddress[0] = 1 means logical axis X of the channel is linked to @1 drive. If the value is -1 means no physical address linked, no logical axis present in channel.
AxesNotInPosition	BOOL	When set to TRUE, this flag indicates that an axis of the channel is out of the positioning window even though it should be. The flag will return to FALSE as soon as the axis will be back in the said window. (see P22)
AxesRecall	BOOL	This flag is relevant in independent channels mode only. In STOP mode as well as at the end of SEARCH, indicates that axis recall is enabled.
BackForwardOnPath	BOOL	Backward/forward movement on path. When set to TRUE, this flag indicates that a backward or forward on path sequence is in progress in the particular channel.
BlockSkip	BOOL	This flag is relevant in Independent Channels mode only. When this flag is set to TRUE, program blocks starting with "/xx" are ignored, according to the setting of the block skip level.
ChannelEnable	BOOL	Status of Channel Enable
ChannelEnableSuspended	BOOL	It is possible to suspend Channel Enable via FXServer.
ChannelType	Enumerator	The return flag is an enumerator: ENUMCHANNELTYPE.AuxiliaryChannel ENUMCHANNELTYPE.CNCChannel ENUMCHANNELTYPE.NotUsed
CncError	BOOL	This flag indicates that an error has been detected on the particular channel.
CncWarning	WORD	Indicates which (if any) warning number is currently present on the channel.
CycleNumber	USINT	This flag indicates that a G code cycle is in execution. Used to read the machining cycle number from %10000 to %10255 (0 for %10000 to 255 for %10255).
DecodeM.	Struct	<i>Structure RCNC.Channel[..].DecodeM.</i> See chapter... 4.2.1.5

Name	Type	Description
EmergencyRetract	BOOL	This flag is relevant in independent channels mode only. When set to TRUE, it indicates the execution of an emergency retract program. Such program can be executed on a voluntary action or activated automatically.
Error	UINT	Error Number for Drive error at channel level handled. This variable contains the channel (generally servo) error number detected by the CNC. (Errors 10, 18, 20, 24, 32, 33, 35, 36, 39-71, 79, 81, 150-158, 160, 161, 200-203, 209-241, 245, 248, 249, 254, 300-331, 424). This variable only returns the channel drives errors and not the part programming error (e.g. Error 1 : Programming format will not be returned). A value of 0 indicates that there is no error currently active.
FeedProgrammed	DWORD	Value of the feed programmed in this channel.
Intervention	BOOL	This flag is relevant in Independent Channels mode only. This flag returns the status ' <i>Intervention</i> '. Intervention occurs after a ' <i>STOP</i> ' the operator will request this mode by action on <i>WCNC</i> . <i>AxisRecall</i> . On the first action on this flag 'Intervention' status is set and the manual commands are authorized. Intervention is also enabled by the M12 function if this one is set as 'Programmed Intervention' (see P7 N1.1).
LoadedToolNumber	DWORD	Tool number loaded in spindle.
M01	BOOL	This flag is relevant in Independent Channels mode only. It indicates the status of the Optional Stop validation. When this flag is set to TRUE all M01 in the channel will be processed like M00 (Stop).

1

2

3

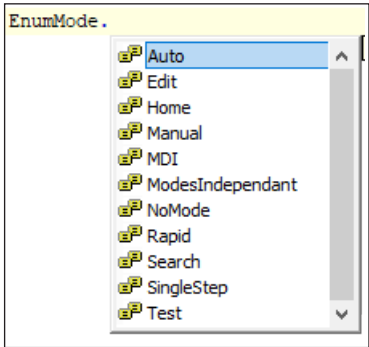
4

5

6

7

A

Name	Type	Description
MFunctionAck	WORD	Encoded M Function with acknowledge This variable is used to read the encoded M functions with acknowledge. Such functions are considered post move and non-modal. Only one non-modal encoded M function per block is allowed. When such an M code is programmed, this variable will reflect the value for at least ONE plc cycle. During this time it is possible to act by clearing the flag ...<i>WCNC.Channel[i].AckMFunction</i>. As long as it will remain FALSE the part program execution will be halted and the M code will remain in the variable. Part program execution will resume when <i>AckMFunction</i> returns to TRUE. When <i>Fast Coded M</i> function (P7 N2.5) is enabled two consecutive M codes can be issued in sequence, when it is not two consecutive M codes will be separated by a value 0 for at least one PLC cycle. M codes can also be issued with delay respective to the position. Refer to the FlexiumPro Extended programming manual M00075.
MFunctionNoAck	WORD	Encoded M Function without acknowledge This variable is used to read the “on the fly” M functions Mxxx to M899. The lowest number for such a function is defined by machine parameter P37 N0 and is higher than 199 in any case. These functions are considered pre-move and modal. As program execution is not halted when such a function is issued care should be taken not to program two consecutively in which case the first might be overwritten without being processed by the PLC.
Mode	INT	This flag is relevant in independent channels mode only. The value of the variable is the image of the current CNC mode for the particular channel: The current mode is defined by EnumMode. 

Name	Type	Description
ProgramActive	BOOL	This flag is relevant in Independent Channels mode only. It indicates that a part program is being executed in AUTO, SINGLE, MDI or DRYRUN mode. - Set to TRUE on the first START command - Set to FALSE on M00, M01 active, M02; and RESET.
ProgramNumber	WORD	Indicates the program number currently active in the channel (valid in the case of independent channels).
ProgramStop	BOOL	This flag is relevant in Independent Channels mode only. When set to TRUE, it indicates a program stop caused by M00 or enabled M01.
Rapid	BOOL	When set to TRUE, this flag indicates that the axes of the channel are currently moving in G00.
ReadEParameter	Method	See Chapter 4.2.1.8
ReadMaximumDynamicTravelLimits	Method	See Chapter 4.2.1.9
ReadMinimumDynamicTravelLimits	Method	See Chapter 4.2.1.10
ReadPosition	Method	See Chapter 4.2.1.11
Reset	BOOL	This flag is relevant in independent channels mode only. This flag indicates that a RESET is being executed. RESET can be triggered by a PLC request, an Operator action or the end of a part program (M02). A RESET returns all functions to their original status except certain E parameters (refer to the FlexiumPro Programming manual M00073 for behavior of the E parameters). If the Reset flag is permanently set to TRUE, it may indicate a malfunction of the PLC program. (in Particular <i>MFunctionAcknowledge</i> to FALSE)
Spindle[x].PhysicalAddress	Array of INT	Indicate the physical address of the particular spindle in the channel (x= 0 to 3). This physical address is required to know which drive to control.
SpindlePhysicalAddressMeasure	INT	Indicates the physical address of the spindle whose measure serves as reference (e.g. for G95). Such spindle is activated by the M66 command. As there is only one reference spindle per channel this variable is valid for the main spindle.
Start	BOOL	This flag is relevant in independent channels mode only. This flag is the image of the field 'START' in the status window. It is set when a command is in progress.

Name	Type	Description
Stop	BOOL	This flag is relevant in independent channels mode only. This flag is the image of the field 'STOP' in the status window. It indicates that a command is on Hold because a stop has been requested.
ToolLifeError	BOOL	Linked with the Tool Management function. This flag will be set as soon as the life of the tool in spindle life reaches the error threshold level and will remain set as soon as some tools in error remain.
ToolLifeWarning	BOOL	Linked with the Tool Management function. This flag will be set as soon as the life of the tool in spindle life reaches the warning level and will remain set as soon as some tools in warning remain.
ToolNumber	DWORD	Tool Number programmed in the channel Contains the tool number (decimal values from 0 to 99999999). T functions are pre-move and modal. After issuing the T function, the system does not wait for an acknowledgement.
TProgrammed	BOOL	A new T value has been programmed in this channel.
TreadCutting	BOOL	Thread cutting in progress. Indicates execution of a thread cutting cycle: - G31 (thread chasing), - G33 (thread cutting) or G38 (sequenced thread cutting), - G84 (rigid tapping). Set by execution of function G31, G33, G38 or G84. Reset by cancellation of the function. It is generally used to inhibit the spindle override by the PLC during such cycles. (Gxx refer to the FlexiumPro Programming manual M00073)

4.2.1.5 Structure `RCNC.Channel[.].DecodeM.`

This structure regroups all decoded M functions of a channel in two sub-structures:

- General purpose M functions
- M functions related to the spindle (array).

As for the coded ML function, it is possible to halt program execution on arrival of a new value by clearing the flag `WCNC.Channel[i].AckMFunction`. Contrary to the coded M function these *DecodeM* functions are generally modal so the flag remain SET as long as the function is active.

Name	Type	Description
General.	Struct	RDecodedMFunctionGeneral 4.2.1.6
Spindle.	Struct	ARRAY [0..3] OF RDecodedMFunctionSpindle 4.2.1.7

4.2.1.6 Structure `RCNC.Channel[.].DecodeM.General.`

Regroups all general purpose decoded M functions of the channel.

Syntax example: **B:= FlexiumPro_NCK.RCNC.Channel[4].DecodeM.General.M08;**

Name	Type	Description
M00	BOOL	Reserved
M01	BOOL	Reserved
M02	BOOL	Reserved
M06	BOOL	Tool change.
M07	BOOL	Coolant 1 ON.
M08	BOOL	Coolant 2 ON.
M09	BOOL	All coolants OFF.
M10	BOOL	Indicates operation with clampable axes clamped when necessary.
M11	BOOL	Indicates operation with clampable axes always unclamped.
M12	BOOL	Mode M12 active. (Programmable intervention see FlexiumPro Programming manual M00073)
M48	BOOL	Overrides active.
M49	BOOL	Overrides locked to 100%.
M997	BOOL	Consecutive execution of the part program blocks forced. See M997 in the FlexiumPro Programming manual M00073.
M998	BOOL	MDI and subroutine call by PLC authorized. See M998 in the FlexiumPro Programming manual M00073.
M999	BOOL	MDI and subroutine call by PLC inhibited. See M999 in the FlexiumPro Programming manual M00073.

4.2.1.7 Structure [RCNC.Channel\[.\].DecodeM.Spindle\[...\]](#).

- [...] Represents the Channel number (1 to 32)
- [...] Represent the spindle number in the channel (0 to 3) (Main, Aux1, Aux2, Aux3)

Syntax example: **B:= FlexiumPro_NCK.RCNC.Channel[4].DecodeM.Spindle[0].M03;**

Name	Type	Description
M03	BOOL	Spindle CW.
M04	BOOL	Spindle CCW.
M05	BOOL	Spindle stopped.
M19	BOOL	Spindle indexing active.
M40	BOOL	Spindle speed range 1 requested.
M41	BOOL	Spindle speed range 2 requested.
M42	BOOL	Spindle speed range 3 requested.
M43	BOOL	Spindle speed range 4 requested.
M44	BOOL	Spindle speed range 5 requested.
M45	BOOL	Spindle speed range 6 requested.
M61	BOOL	When set to TRUE indicates that this particular spindle is released from the channel.

4.2.1.8 Method [RCNC.Channel\[.\].ReadEParameter\(](#)

[.] Represents the index number (1 to 32)

	Name	Type	Description
	EParameter	UDINT	Starting E parameter index to read
	numberToRead	UINT	Number of E parameter to read out the values
	Values	Array of CNCVariablesValue	Return values in array
	Result	UDINT	RTS_IEC_RESULT

Extended NCK access (ENA) licence needed.

4.2.1.9 Method [RCNC.Channel\[..\].ReadMaximumDynamicTravelLimits\(](#)

[..] Represents the index number (1 to 32)

	Name	Type	Description
	Values	Array of LREAL	Array destination of the values
	Result	UDINT	RTS_IEC_RESULT

Extended NCK access (ENA) licence needed.

4.2.1.10 Method [RCNC.Channel\[..\].ReadMinimumDynamicTravelLimits\(](#)

[..] Represents the index number (1 to 32)

	Name	Type	Description
	Values	Array of LREAL	Array destination of the values
	Result	UDINT	RTS_IEC_RESULT

Extended NCK access (ENA) licence needed.

4.2.1.11 Method [RCNC.Channel\[..\].ReadPosition\(](#)

[..] Represents the index number (1 to 32)

	Name	Type	Description
	Values	Array of LREAL	Array destination of the values
	Result	UDINT	RTS_IEC_RESULT

Extended NCK access (ENA) licence needed.

4.2.1.12 Array [RCNC.E10000\[..\]](#)

Syntax example: **C:= FlexiumPro_NCK.RCNC.E10000[1];**

Name	Description
E10000[xy]	External parameters E100xy (xy being a number from 00 to 31) These parameters, Parameters used to report Boolean information to the PLC program. They can be read or written by part program and dynamic operators. The function of these parameters is defined by the machine builder. Format: BOOL (0/1).

4.2.1.13 Array [RCNC.E30000\[.\]](#)

Syntax example: **C:= FlexiumPro_NCK.RCNC.E30000[4];**

Name	Description
E30000[xyz]	External parameters E30xyz (xyz being a number from 000 to 255) These parameters are used to report a LREAL information to the PLC program. They can be read or written by part program and dynamic operators. The function of these parameters is defined by the machine builder. Format: LREAL.

4.2.1.14 Array [RCNC.E33000\[..\]](#)

Syntax example: **C:= FlexiumPro_NCK.RCNC.E33000[4];**

Name	Description
E33000[xyz]	External parameters E30xyz (xyz being a number from 000 to 999) These parameters are used to report an USINT information to the PLC program. They can be read or written by part program and dynamic operators for the first 10. The function of these parameters is defined by the machine builder. Format: USINT (0.. 255).

4.2.1.15 Array [RCNC.E36000\[.\]](#)

Syntax example: **C:= FlexiumPro_NCK.RCNC.E36000[4];**

Name	Description
E36000[xyz]	External parameters E36xyz (xyz being a number from 000 to 999) These parameters are used to report an UINT information to the PLC program. They can be read or written by part program and dynamic operators for the first 3. The function of these parameters is defined by the machine builder. Format: UINT (0..65535).

4.2.1.16 Array [RCNC.E37000\[.\]](#)

Syntax example: **C:= FlexiumPro_NCK.RCNC.E37000[8];**

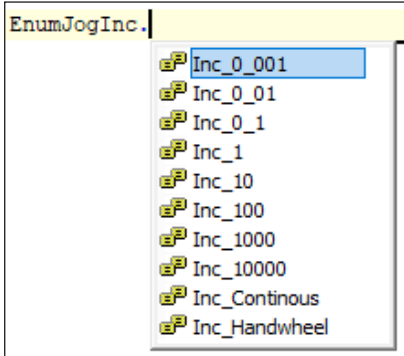
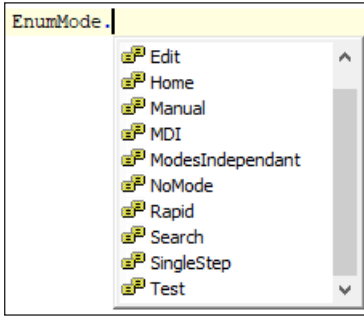
Name	Description
E37000[xyz]	External parameters E37xyz (xyz being a number from 000 to 127) These parameters are used to report Byte information to the PLC program. They can only be written by dynamic operators. The function of these parameters is defined by the machine builder. Format: BYTE (0.. 255).

4.2.1.17 Structure **RCNC.General**.

This structure is intended to report General information on the machine status.

Syntax example: **C:= FlexiumPro_NCK.RCNC.General.Mode;**

Name	Type	Description
AxisRecall	BOOL	General axis recall. Set or reset by a PLC request. (<i>WCNC.General.AxisRecall</i>). In STOP mode and at the end of SEARCH, indicates that axis recall is enabled.
BlockChangeOccurred[x]	Array of BOOL	Indicates that an early block change has been issued. For information about Early Block Change, please refer to the FlexiumPro Extended Programming Manual M00075.
BlockSkip	BOOL	Status of the Block skip Image of the <i>"/" field</i> of the CNC status window. When this flag is set, program blocks starting with <i>"/xx"</i> are ignored, according to the setting of the block skip level.
CncError	BOOL	CNC error Image of field <i>"CN??"</i> in the CNC status window. This flag indicates a machine error or a part program error. The machine error number is contained in <i>"RCNC.General.MachineError"</i> . Reset by the RESET key on the panel or <i>"WCNC.General.Reset" = TRUE</i> .
CncReady	BOOL	CNC ready Indicates that the CNC is able to handle the machine. Set to TRUE at Power On and after a rest assuming the condition below are met. Set to FALSE by detection of an excessive following error, encoder fault, out of synchronization and in general any fault that can prevent a safe functioning.
CommunicationCycleTime	UINT	PLC cycle time defined by P50*P99N0
DripFeed	BOOL	Drip Feed mode status ready Indicates that the CNC is ready to operate in drip feed mode (loading part program during execution).
DripFeedBufferReady	BOOL	The dripfeed buffer contains data therefore START can be activated.
EmergencyRetract	BOOL	General emergency retract in progress Set to indicate the execution of an emergency retract program. Such program can be executed on a voluntary action or activated automatically. Reset by detection of M00 or M02.
IndependentGroup	BOOL	Common channels/independent channels switch over. The change of state is taken into account only on a general reset (<i>WCNC.General.Reset</i>) requested by the PLC. IndependentGroup = 0: Common channels. IndependentGroup = 1: Independent channels.

Name	Type	Description
Intervention	BOOL	This flag returns the status '<i>Intervention</i>' Intervention occurs after a '<i>STOP</i>' the operator will request this mode by action on <i>WCNC.AxisRecall</i>. On the first action on this flag '<i>Intervention</i>' status is set and the manual commands are authorized. This flag is also set by the function M12 when declared as Programmed Intervention (see parameter P7 N1.1).
JogIncrement	INT	Current jog increment The value of this variable is the image of the current jog increment: The value is of type EnumJogInc. 
M01	BOOL	Status of optional program stop Image of field M01 of the CNC status window. If the bit is set, the next M01 in the part program will be processed as an M00 and generate a stop.
MachineError	WORD	Machine Error Number This variable contains the machine (generally servo) error number detected by the CNC. (Errors 10, 18, 20, 24, 32, 33, 35, 36, 39-71, 79, 81, 150-158, 160, 161, 200-203, 209-241, 245, 248, 249, 254, 300-331, 424). This variable only returns the machine errors and not the part programming error (e.g. Error 1 : Programming format will not be returned). A value of 0 indicates that there is no machine error currently active.
Mode	INT	Current Mode The value of the variable is the image of the current CNC mode: The current mode is defined by EnumMode.  In Independent Channels mode, the returned value is 16 (not in enum)

Name	Type	Description
NMAuto	BOOL	N/M AUTO functionality Set to indicate that the N/M functionality (2/3, 3/5, etc.) is enabled.
OnlineCollision	BOOL	Indicates that a collision has been detected and of what type. For information about On Line Collision
ProgramActive	BOOL	Program active Indicates that a part program is being executed in AUTO, SINGLE, MDI or DRYRUN mode. - Set to TRUE on the first START command - Set to FALSE on M00, M01 active, M02; and RESET.
ProgramNumber	WORD	Current Part Program Number This variable contains the number of the current (active) part program. - The value -1 indicates there is no part program active. - The value 0 indicates the selection of the drip feed mode.
ProgramStop	BOOL	Program stop Set to indicate a program stop caused by M00 or enabled M01. Reset by the START key on the machine operator panel. (<i>WCNC.General.Start</i>).
Reset	BOOL	CNC reset This flag indicates that a RESET is being executed. RESET can be triggered by a PLC request, an Operator action or the end of a part program (M02). A RESET returns all functions to their original status except certain E parameters (refer to the FlexiumPro Programming manual M00073 for behavior of the E parameters). If the Reset flag is permanently set to TRUE, it may indicate a malfunction of the PLC program (in Particular <i>MFunctionAcknowledge</i> to FALSE).
Start	BOOL	Cycle START This flag is the image of the field 'START' in the status window. It is set when a command is in progress.
Stop	BOOL	Cycle STOP This flag is the image of the field 'STOP' in the status window. It indicates that a command is on Hold because a stop has been requested.

1

2

3

4

5

6

7

A

4.2.1.18 Structure **RCNC.IO**.

This structure corresponds to the analog and digital IOs located on the RTK unit P2 connector (See FlexiumPro Installation Manual M00070).

Please note that as the index starts at 0 the above value actually define the range and not the max number.

Syntax example: **C:= FlexiumPro_NCK.RCNC.IO.AnalogInput[2];**

Name	Type	Description
AnalogInput[x].Value	WORD	Value of the analogue input, 12 bits ADC 0-10V, located on the RTK unit P2 connector. [x] = 1 to 2 according to the input group. The data is also available in the parameters E44100 to E44103.
AnalogOutput[x].Value	WORD	Word Value of the analog output issued on the RTK unit P2 connector. [x] = 1 to 2 according to the group. Caution. This variable allows for reading the analog output value written by the PLC or CNC. For writing a value please refer to the structure <i>WCNC.IO</i> .
DigitalInput[x].Value	BOOL	Value of the digital input byte located on the RTK unit P2 connector. [x] = 1 to 4 according to the input group. The data is also available in the parameters E44000 to E44003.
DigitalOutput[x].Value	BOOL	Value of the digital output issued on RTK unit P2 connector. [x] = 1 to 4 according to the group. The data are also available in the parameters E34010 to E34013. Caution. This variable allows for reading the digital output value written by the PLC or CNC. For writing a value please refer to the structure <i>WCNC.IO</i> .

4.2.1.19 Method [RCNC.GetParameterType\(\)](#)

	Name	Type	Description
	Parameter	DWORD	NCK Parameter number
	NNumber	DWORD	Number of the DWord of the parameter
	Result	UDINT	RTS_IEC_RESULT
	GetParameterType	BYTE	NCKParameterType. TYPE0_BIN_BYTE TYPE1_DEC_WORD TYPE10_UNSIGNED_LREAL TYPE2_DEC_DWORD TYPE3_DEC_SINT TYPE4_DEC_UDINT TYPE5_DEC_UINT TYPE6_BIN_UDINT TYPE7_BIN_UINT TYPE9_SIGNED_LREAL UNKNOWN

Extended NCK access (ENA) licence needed.

4.2.1.20 Method [RCNC.ReadParameter\(\)](#)

	Name	Type	Description
	Parameter	DWORD	Number of CNC parameter to read
	NNumber	DWORD	Number of the DWord of the parameter
	pValue	POINTER TO BYTE	Pointer to value used for read out value
	Size	DWORD	Type size of NCK parameter
	Result	UDINT	RTS_IEC_RESULT

Extended NCK access (ENA) licence needed.

4.2.1.21 Structure [RCNC.Spindle\[.\].](#)

[.] This structure is an array of 32 structures corresponding to the physical addresses of the 32 possible spindles.

Syntax example: **B:= FlexiumPro_NCK.RCNC.Spindle[2].Speed;**

Name	Type	Description
AssignedToChannel	INT	Spindle assigned to channel number (-1 = not assigned)
AssignedToRank	INT	Spindle assigned to rank number (-1 not in rank)
HomeForced	BOOL	Indicates that this spindle has been homed by a keyboard action. It can therefore be out of position.
Load	Struct	See chapter 4.2.1.22

Name	Type	Description
LogicalAddress	INT	Spindle logical address
NotHomed	BOOL	Indicates that this spindle has not yet been homed or has lost this status (e.g. Excessive speed).
PositionReached	BOOL	This flag is managed in spindle indexing or spindle synchronization request. When set to TRUE, it indicates that the spindle is in position or synchronized. It will return to FALSE after leaving the required position, in case of oscillations and by cancellation of the M19 function.
Speed	WORD	This variable returns the spindle speed reference. The value is unsigned. The maximum speed on the selected gear corresponding to 32767.
SpeedProgrammed	UDINT	Spindle speed programmed by user
SpeedReached	BOOL	This flag is set to TRUE when the spindle equals the programmed speed considering a tolerance window given by E90764 to E90795 [respectively for spindle @0 to @31 plus the offset E90732 to E90763 [respectively for spindle @0 to @31]. These parameters are expressed in 1/256 of the programmed speed. They can be written and are initialized at 13 (about 5%).
Stopped	BOOL	This flag is set to TRUE when the spindle speed is lower than the parameter E90732 to E90763 [respectively for spindle @0 to @31]. This parameter in rpm is initialized at 10.

4.2.1.22 Structure **RCNC.Spindle[...].Load**.

Name	Type	Description
Valid	BOOL	If this signal is true the value is available, for instance for index not defined as spindle
Value	UINT	Value in % of the nominal current of the motor

4.2.2 Structure **Simulation.**

Name	Type	Description
RCNC.	Struct	See chapter 4.2.2.1
WCNC.	Struct	See chapter 4.2.2.2

4.2.2.1 Structure **Simulation.RCNC.**

Name	Type	Description
Axis[x].SimulRefPos	UINT	Return the position value of the specific Axis in Real.
General. Simulation	BOOL	The flag is true when the system is in simulation mode (digital twin).

4.2.2.2 Structure **Simulation.WCNC.**

Name	Type	Description
AXIS[..].	Struct	See chapter 4.2.2.3

4.2.2.3 Structure **Simulation.WCNC.Axis[..].**

[..] Represents the index number (0 to 31)

Name	Type	Description
DigitalAxis.	Struct	See chapter 4.2.2.4
SAMX.	Struct	See chapter 4.2.2.5

4.2.2.4 Structure [Simulation.WCNC.Axis\[..\].DigitalAxis.](#)

The structure [Simulation.WCNC.Axis\[..\].DigitalAxis](#) handled by PLC assign to [RNC.Axis\[..\].DigitalAxis](#) the values only when the Drive is in simulaiton (drive is not really present).

Useful for digital twin, plc developing without physical drives hardware or missing drive on real machine.

[..] Represents the index number (0 to 31)

Name	Type	Description
BrakeRelay	BOOL	Set to TRUE when the brake is released (energized). (brake management by the drive)
BrakeTestRunning	BOOL	Set to TRUE when the brake test is running.
BumpSuppressor	BOOL	Set to TRUE when the BumpSuppressor is active.
ConsWPMActive	BOOL	Set to TRUE when WPM software is connected in console mode.
ControlMode	BOOL	Set to FALSE when the MDLUF drives operates in Position mode, set to TRUE for Speed mode.
DCBusPresent	BOOL	Set to TRUE when high voltage DC Bus is present.
DEMxDigitalOutput	BOOL	Set to TRUE by a DEMX (drive Embedded Macros – optional) application program.
DigitalAxisI2T	USINT	Percentage of the Drive I2T value.
DriveInRunState	BOOL	Set to TRUE when Drive firmware in execution.
DriveOk	BOOL	Set to TRUE when the drive is operational. (No alarm present)
EPS_EPSMActiv	BOOL	Set to TRUE when encoder Self-calibration is in progress.
MainsOn	BOOL	Set to TRUE when the main power supply is active.
MechPower	REAL	Mechanical power at motor shaft: Positive value means power to load Negative value means power from load.
MotorI2T	USINT	Percentage of I2T value of the motor.
MotorLoad	UINT	Percentage of actual torque current related to available torque current.
MotorMoving	BOOL	Set to TRUE when Motor speed is higher than +/- V41 or V42 drive parameters.
ParameterSet	USINT	Parameter set number. If 99 means default parameter set.
Parking	BOOL	Drive in parking state.
Prog01	BOOL	Status of programmable output 1 (see V140 to V145 drive parameter).
Prog02	BOOL	Status of programmable output 2 (see V150 to V155 drive parameter).
PWMCutOff	BOOL	Set to TRUE when NUM-STOX or NUM-SAMX intervention.
ReferenceEnabled	BOOL	Set to TRUE when the Reference (speed) command is enabled.
CncDriveCommOk	BOOL	Set to TRUE when NCK <-> Drive communication is running. If this flag is false, all other Status and Extended Status information are meaningless.
SpeedReached	BOOL	Set to TRUE when the requested speed is reached. (Refer to V40 on M00066 manual)
TorqueEnabled	BOOL	Set to TRUE when the Drive torque enabled.
Warnings	BOOL	Set to TRUE when the drive issues a Warning.

4.2.2.5 Structure [Simulation.WCNC.Axis\[..\].SAMX](#).

[..] Represents the index number (0 to 31)

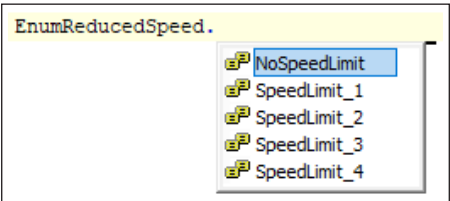
Name	Type	Description
EmergencyQuickStop	BOOL	Emergency stop requested by SAMX.
EmergencyStopStatus	BOOL	Emergency braking requested by SAMX.
SAMXOk	BOOL	Communication Drive - SAMX ok

4.2.3 Structure [WCNC](#).

Name	Type	Description
Axis[..]	Struct	See chapter 4.2.3.1
Channel[..]	Struct	See chapter 4.2.3.4
E20000[x]	Array	See chapter 4.2.3.11
E40000[x]	Array	See chapter 4.2.3.12
E43000[x]	Array	See chapter 4.2.3.13
E46000[x]	Array	See chapter 4.2.3.14
E47000[x]	Array	See chapter 4.2.3.15
General	Struct	See chapter 4.2.3.16
Handwheel[..]	Struct	See chapter 4.2.3.19
IO[..]	Struct	See chapter 4.2.3.20
PLCString	Prop.	String(36) retrieved by the CNC ISO function _GETSPLC
Spindle[..]	Struct	See chapter 4.2.3.23
WriteParameter	Method	See chepter 4.2.3.25

4.2.3.1 Structure [WCNC.Axis\[..\]](#)

[..] Represents the axis physical address (0 to 31)

Name	Mode	Description
DigitalAxis	Struct	See chapter 4.2.3.2
DisableServoCntrl	S	This flag when set inhibits the link with the interpolator Equivalent to parameter P120 N2 or E910xx. 1. This bit cannot be forced before the NCK is fully started; it works on the transition FALSE->TRUE and TRUE->FALSE 2. It should be used when the axis is stopped otherwise the axis will stop immediately while other axes moving together keep moving.
JogNeg	S	In Manual or Home mode setting this flag to TRUE requests a movement in the negative direction on the selected axis. In incremental JOG the movement is started on the positive edge of the signal and continues even is the flag is not maintained to TRUE. In handwheel mode, this flag should be set to TRUE to authorize the move in negative direction. The quantity of movement itself is defined by the handwheel rotation.
JogPos	S	In Manual or Home mode setting this flag to TRUE requests a movement in the positive direction on the selected axis. In incremental JOG the movement is started on the positive edge of the signal and continues even is the flag is not maintained to TRUE. In handwheel mode, this flag should be set to TRUE to authorize the move in positive direction. The quantity of movement itself is defined by the handwheel rotation.
ReducedSpeed	S	Force a speed limit among four possible on the corresponding axis. See P34 for axes reduced speeds and P43 for reduced spindle speeds. The different possibilities of reduced speeds are: Please note that the limitation is applied at the tool center point level. Therefore in case of 5 axes machining with RTCP, it is possible that the speed of one or more linear axes exceeds the respective P34 value. 
Stop	S	In machining mode or JOG mode, setting this flag to TRUE stops the corresponding axis or prevents it to start moving. Other axes can continue to move except if they are involved in an interpolation with one axes whose Stop bit is TRUE. In such a case all movement are in idle (equivalent to override =0).

4.2.3.2 Structure `WCNC.Axis[..].DigitalAxis`.

[..] Represents the axis physical address (0 to 31)

Syntax example: `FlexiumPro_NCK.WCNC.Axis[7].DigitalAxis.TorqueEnable:= TRUE;`

Name	Mode	Description
CurrentReduction	S	The current reduction function is used to reduce the current on the digital axes and spindles. Current reduction is coded on a byte. Let's call this value 'a' . - For a = [16#0, 16#80 to 16#FF] : No current reduction. - For a = [16#01 to 16#7F] - $I_{red} = I_{nom}(128 - \text{value})/128$ - $I_{nom} = V50$ or $V57$ or $V1050$ or $V1057$ When operating in anti-backlash configuration, this current reduction has no effect on the preload currents.
LoadParameterSet	Method	Load parameters in the drive by set number.
MotorBreakTest	Method	See chapter 4.2.3.3
RampDisable	S	When set to TRUE there is no acceleration / deceleration control by the drive itself. When set to FALSE the acceleration/ deceleration ramp depends about the drive settings (V30.14).
ReferenceEnable	S	When set to TRUE this flag authorizes the Speed command activation on the drive selected. Speed command activation is not possible as long as the torque is not active.
ResetError	Method	Reset the Drive error
SetToHalt	Method	Set the Drive to halt
SetToRun	Method	Set the drive to run
TorqueEnable	S	When set to TRUE this flag authorizes Torque activation on the drive selected.
DisableBySetExch	S	When this flag is true the Drive disable the braking ramp at torque disable by plc. Use this flag for star-delta switching on the fly.

4.2.3.3 Method `WCNC.Axis[..].DigitalAxis.MotorBrakeTest`

	Name	Type	Description
	PosDisplacementAmplitude	LREAL	Distance of displacement in positive direction expressed in CNC unit
	NegDisplacementAmplitude	LREAL	Distance of displacement in negative direction expressed in CNC unit
	Result	UDINT	RTS_IEC_RESULT

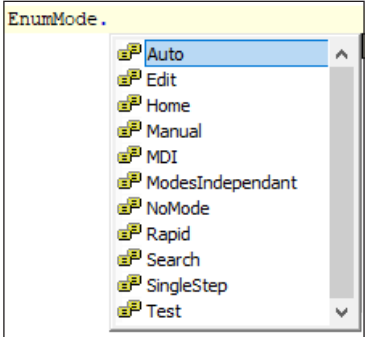
4.2.3.4 Structure `WCNC.Channel[.].`

[.] Represents the channel number (1 to 32)

Syntax example: `FlexiumPro_NCK.WCNC.Channel[2].AckMFunction:= TRUE;`

Name	Mode	Description
AckMFunction	S	M function acknowledge. This flag is checked at the end of an M function processing. The execution will be on hold until it is set to TRUE. As a RESET (RAZ) reinitializes several M codes it will also be checked at this time and therefore should be TRUE after power-up This flag should be TRUE in normal operation.
AxisRecall	P	Axis Recall in independent channels mode. This request is taken into account in 'CYCLE STOP' condition and when all axis jogs commands are released. It is a toggle command between 'Intervention' and 'Axes recall'. The first push activates 'Intervention' mode. The manual movements are enabled in both directions. The second push activates 'AxisRecall' mode. The manual movements are only permitted in the direction allowing each axis to return the position it had before 'Intervention'. As long as all axes are not in their initial position, it will not be possible to quit the 'CYCLE STOP' status.
BacktrackBackward	P	When set to TRUE, requests a backward movement on the path The backward movement will stop when this flag is set to FALSE
BacktrackForward	P	When set to TRUE, requests a forward movement on the programmed path after an intervention and a prior backtrack backward. This flag is available from version 4.0.20 on.
BlockSkip	P	This flag is used in Independent channels mode. A positive edge will toggle the status of the <i>Block skip (/n)</i> . In common channels mode, it is the BlockSkip flag in the structure 'WCNC.General.' which is tested.
ChannelEnable	S	Channel enable. When this flag is set to TRUE it enables the use of the axes channel. The flag is only taken into account on a RESET. The main use is for machine setup purpose by selectively enabling each channel for easier testing. This flag should be TRUE in normal operation.

Name	Mode	Description
EmergencyReq	P	When this signal is active the nominal speed of the spindles and axes in the channel, is cleared without deceleration ramp. The lag error and the position window standstill monitoring are disabled. CNC error 246 Ext1 Emergency Req is displayed and the NCK goes in emergency. As long as <i>FlexiumPro_NCK.WCNC.Channel[x].EmergencyReq</i> is active the real axis position is displayed in the Pos. OP in the Prod Page To clear error 246.1 a Reset is required once this signal goes low again.
EmergencyRetract	P	This flag is only relevant in independent channels mode. Emergency retraction request in independent channels mode. On a positive edge of this flag during part program execution, the current block will be interrupted and a jump to the last emergency retraction sequence declared (G75) executed. If no emergency retraction sequence was specified, a STOP will be generated. In common channels mode, it's the Emergency retract flag in the structure '<i>WCNC.General</i>'. that must be managed.
EndExtMovement	S	End of external movement. When this flag is FALSE during program execution it will prevent the final completion of the current block (CYCLE START stays ON). The execution will resume automatically when the flag will be ON again. This flag should be TRUE in normal operation.
FeedEnable	S	Feed authorization for the channel. This flag must be ON to authorize movement on the channel. This flag should be TRUE in normal operation.
FeedOverride	S	Value of the Feed Rate potentiometer for the particular channel. The value ranges from 0 to 255, the applied value (%) depend from the potentiometers range defined P67 N0-N1: 0= P67N0, 255= P67N1 and it is applied if M48 active.
M01	P	This flag is used in Independent channels mode. A positive edge will toggle the status of the Optional Stop (M01). In common channels mode, it is the M01 flag in the structure '<i>General</i>' which is tested.
ManualSpeed	S	Selection of manual speed and Handwheel increment by the PLC. The following values are available: <pre> EnumManSpeed. FastJog_HandWheel_10 FastJog_HandWheel_1000 NormalJog_HandWheel_1 SlowJog_HandWheel_100 </pre> The speed values <i>FastJog</i>, <i>NormalJog</i> and <i>SlowJog</i> are defined respectively in machine parameters P31 N0, N1 and N2. This variable is used in independent channels. For common channel mode use the flag at level <i>WCNC.General</i>.

Name	Mode	Description
Mode	S	Mode requested on independent axes channel. This variable will only be considered if the flag <code>WCNC.General.LockMode</code> is SET. The following CNC modes can be requested and the following values are available:  The activation will be made on the rising edge of the flag <code>WCNC.Channel[.].Reset</code>.
PlcAxisMode	S	Activates AUTO or SINGLE mode on PLC axes channel. When set to FALSE: AUTO mode enabled (on the next block if program in progress). When set to TRUE: SINGLE mode enabled immediately. This flag is only meaningful for PLC channels and if the channel is enabled.
PlcCallSubroutine	P	Subroutine call by the PLC. When this flag is set to TRUE during execution of a part program, setting a branch to subroutine %9999.x (where x is the channel number) will be executed at the end of the current block.. No confirmation is sent by the system, the subroutine itself can send an acknowledge (M code or else) to the PLC to confirm the correct execution.
PlcInterrupt	P	Block interrupt. Associated with the G10 function, this flag when set to TRUE will generate an early termination of the current block (again assuming it contains a G10).
ProgramNumber	S	Part Program Number selection in case of independent channels. Used to select the part program number to activate. The activation will be made on the rising edge of the flag <code>"WCNC.Channel[.].Reset"</code>. 0: No program number requested by the PLC. 1 to 9999: Program number specified by the PLC.
Rapid	S	When set to TRUE during program execution this flag will enable movement at the highest possible speed. The speed returns to the programmed value as soon as the flag is set to FALSE again. The override remains active (if M48) This flag has no effect on the manual movements.
Reset	P	Reset request in PLC or independent channels mode. Request to reset in NCK in the default power up configuration with the exception of some E parameters (see FlexiumPro Programming Manual M00073). It will be taken into account only if there is no movement on the axes.

Name	Mode	Description
Spindle[...].	Struct	Array [0..3] of WCNC.Channel[...].Spindle[...] (see chapter 4.2.3.5) for manual control of the spindle direction. Spindles are considered by their rank (Main, Aux1.. Aux3) in the channel.
Start	P	Start request in PLC or independent channels mode. Starts part program or command line execution in AUTO, SINGLE, MDI, DRYRUN, SEARCH, and TEST modes. This signal is ignored unless the channel is valid.
Stop	P	Stop request in PLC or independent channels mode. Immediately stops program or command execution in AUTO, SINGLE, MDI, DRYRUN as well as incremental JOG modes. Machining is restarted by activating "Start". This flags stop the execution of the movements controlled by the CNC. For cycles controlled by the PLC (e.g. Tool changer sequence), the programmer needs to take all appropriate actions.
WriteEParameter	Method	See chapter 4.2.3.6
WriteMaximumDynamicTravelLimit	Method	See chapter 4.2.3.7
WriteMaximumDynamicTravelLimits	Method	See chapter 4.2.3.8
WriteMinimumDynamicTravelLimit	Method	See chapter 4.2.3.9
WriteMinimumDynamicTravelLimits	Method	See chapter 4.2.3.10

4.2.3.5 Structure **WCNC.Channel[...].Spindle[...].**

[..] Represents the channel number (1 to 32)

[...] Represent the spindle number in the channel (0 to 3) (Main, Aux1, Aux2, Aux3)

Syntax example: **FlexiumPro_NCK.WCNC.Channel[2].Spindle[0].JogM3:= TRUE;**

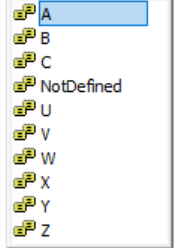
Name	Mode	Description
JogM3	S	In manual mode, this flag requests the particular spindle of the channel to rotate in M03 direction.
JogM4	S	In manual mode, this flag requests the particular spindle of the channel to rotate in M04 direction.

4.2.3.6 Method [WCNC.Channel\[..\].WriteEParameter\(\)](#)

	Name	Type	Description
	Eparameter	UDINT	Starting E parameter index to write
	numberToWrite	UINT	Number of E parameter to write
	Values	Array of CNCVariablesValue	Array of value to write in the Exxx
	Result	UDINT	RTS_IEC_RESULT

License Extend NCK Access is needed (FPSW282124).

4.2.3.7 Method [WCNC.Channel\[..\].WriteMaximumDynamicTravelLimit\(\)](#)

	Name	Mode	Description
	Axis	UINT	Axis:= EnumAxisName. 
	Value	LREAL	Value to write
	Result	UDINT	RTS_IEC_RESULT

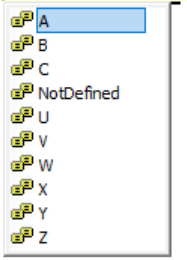
License Extend NCK Access is needed (FPSW282124).

4.2.3.8 Method [WCNC.Channel\[..\].WriteMaximumDynamicTravelLimits\(\)](#)

	Name	Type	Description
	X	LREAL	Value to write for specific axis
	Y	LREAL	Value to write for specific axis
	Z	LREAL	Value to write for specific axis
	U	LREAL	Value to write for specific axis
	V	LREAL	Value to write for specific axis
	W	LREAL	Value to write for specific axis
	A	LREAL	Value to write for specific axis
	B	LREAL	Value to write for specific axis
	C	LREAL	Value to write for specific axis
	Result	UDINT	RTS_IEC_RESULT

License Extend NCK Access is needed (FPSW282124).

4.2.3.9 Method [WCNC.Channel\[...\].WriteMinimumDynamicTravelLimit\(\)](#)

	Name	Type	Description
	Extended NCK access (ENA) licence needed Axis	UINT	<code>Axis:= EnumAxisName.</code> 
	Value	LREAL	Value to write
	Result	UDINT	RTS_IEC_RESLT

License Extend NCK Access is needed (FPSW282124).

4.2.3.10 Method [WCNC.Channel\[...\].WriteMinimumDynamicTravelLimits\(\)](#)

	Name	Type	Description
	X	LREAL	Value to write for specific axis
	Y	LREAL	Value to write for specific axis
	Z	LREAL	Value to write for specific axis
	U	LREAL	Value to write for specific axis
	V	LREAL	Value to write for specific axis
	W	LREAL	Value to write for specific axis
	A	LREAL	Value to write for specific axis
	B	LREAL	Value to write for specific axis
	C	LREAL	Value to write for specific axis
	Result	UDINT	RTS_IEC_RESULT

Extended NCK access (ENA) licence needed

4.2.3.11 Array [WCNC.E20000\[.\]](#)

[..] Represents the index number (0 to 31)

Syntax example: **FlexiumPro_NCK.WCNC.E20000[5]:= TRUE;**

Name	Mode	Description
E20000[xy]	S	External parameters E200xy (xy being a number from 00 to 31). These parameters, used to make machine status data available to the part program, are written by the PLC application and read by the part program. Format: BOOL (0/1).

4.2.3.12 Array [WCNC.E40000\[.\]](#)

[..] Represents the index number (0 to 255)

Syntax example: **FlexiumPro_NCK.WCNC.E40000[2]:= 56.245;**

Name	Mode	Description
E40000[xyz]	S	External parameters E40xyz (xyz being a number from 000 to 255). These parameters, used to transfer signed numerical values like dimensions, offset, are written by the PLC and read by the part program. Format: LREAL (64bits).

4.2.3.13 Array [WCNC.E43000\[.\]](#)

[..] Represents the index number (0 to 999)

Syntax example: **FlexiumPro_NCK.WCNC.E43000[1]:= 22;**

Name	Mode	Description
E43000[xyz]	S	External parameters E43xyz (xyz being a number from 000 to 999). These parameters, used to transfer PLC data on bytes, are written by the PLC and read by the part program. The first 10 parameters can be read by Dynamic Operators. Format: USINT (8 Bits).

4.2.3.14 Array [WCNC.E46000\[..\]](#)

[..] Represents the index number (0 to 999)

Syntax example: **FlexiumPro_NCK.WCNC.E46000[458]:= 89;**

Name	Mode	Description
E46000[xyz]	S	External parameters E46xyz (xyz being a number from 000 to 999). These parameters, used to transfer PLC data on words, are written the PLC and read by the part program. The first 2 parameters can be read by Dynamic Operators. Format: UINT (16 bits).

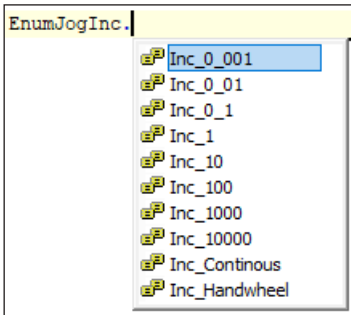
4.2.3.15 Array [WCNC.E47000\[..\]](#)

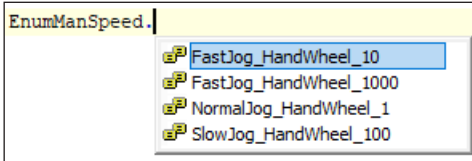
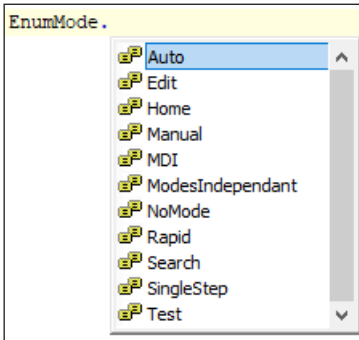
[..] Represents the index number (0 to 127)

Syntax example: **FlexiumPro_NCK.WCNC.E47000[8]:= 25;**

Name	Mode	Description
E47000[xyz]	S	External parameters E47xyz (xyz being a number from 000 to 127). These parameters, used to make PLC information available to the NCK, are written by the PLC application and read by the part program (including by Dynamic Operators). Format: Byte (8 bits).

Name	Mode	Description
AxisRecall	P	Axis recall selection. This request is taken into account in 'CYCLE STOP' condition and when all axis jogs commands are released. It is a toggle command between 'Intervention' and 'Axes recall'. The first push activates 'Intervention' mode. The manual movements are enabled in all directions. The second push activates 'AxisRecall' mode. The manual movements are only permitted in the direction allowing each axis to return the position it had before 'Intervention'. As long as all axes are not in their initial position, it will not be possible to quit the 'STOP' status.
BackTrackAuto	S	When set to TRUE, requests an automatic axis recall after an intervention. The return path follows the intervention path (by default for the first 10 elements). The return sequence stops when the first point is reached or when the command is released.
BlockChangeCtrl[x]	P	Early block Change (EBC) command. Array [0..31] of boolean to allow an early block change via G777 lx. For information about Early Block change please refer to the FlexiumPro Extended Programming Manual M00075.
Blockskip	P	Toggles the <i>Block Skip</i> validation (see FlexiumPro Programming manual M00073). According to the setting of P7 this command can be taken into account or not during program execution. This flag is associated with <i>BlockSkipLevel</i>.
BlockSkipLevel	S	This 32 bit value defines the 32 classes of Block skip. When blockskip is set, then all blocks starting with "n,m,j-k" being one of the classes set in this word, will be ignored.
CancelLagError	P	This flag is used to clear the lag value when an axis is in excess following error status (Error 4[x] ext 3). Once the error has been cancelled the system goes automatically into Axis Recall operation.
DifferentialAxisOffset	S	When SET this flag enables to generate a differential offset between axes with a handwheel. For details refer to Dynamic Operator 24 in the FlexiumPro Extended Programming Manual M00075.
EmergencyRetract	P	Emergency retract request. This request is taken into account in AUTO, SINGLE and DRYRUN Modes. When the first interruptible block is encountered, the system branches to the last emergency retract sequence declared in the part program by the function G75. If no emergency retract sequence was defined, this command generates a 'STOP'.

Name	Mode	Description
FeedEnable	S	When SET this flag enables axes movement in all modes (unless such movements are forbidden otherwise). When clear, no axis movement can occur. The state of this signal is displayed by a "STOP" in the HMI status window.  This flag only inhibits Servo axes movements. Auxiliary movements (e.g. Tool changer arm) are controlled by the PLC and may be able to move suddenly. This flag should be TRUE in normal operation.
InchUnit	S	Selection of the inch units (metric or inch system). When this flag is set the display and dimension entry are in inches and display in inches, otherwise they are in mm.
IndependentGroup	S	Common channels/independent channels switchover. The change of state is taken into account only on a general reset (WCNC.General.Reset) requested by the PLC. IndependentGroup = 0: Common channels. IndependentGroup = 1: Independent channels.
JogIncrement	S	JOG increment selection. The variable value corresponds to the jog increment requested. The jog increment is in mm or inches depending on the value of variable <i>InchUnit</i>. Possible values: (numbers correspond to NCK resolution (Axis increment)). 
LockBlockSkipLevel	S	When set this flags allows the PLC for changing the block skip level and prevents the HMI of doing so.
LockDynToolOffset	S	Enables Tool Wear offset modification by PLC. When this flag is set, the tool wear offsets modifications are authorized by the PLC and forbidden by the HMI.
LockInkJog.	Struct	See chapter 4.2.3.17
LockJog	S	Enables JOG increment selection by the PLC. When this flag is set, the selection of jog increments is authorized by the PLC and forbidden by the HMI.

Name	Mode	Description
Lockmode	S	Enables Mode selection by the PLC. When this flag is set, mode selection is authorized by the PLC and forbidden by the HMI.
LockSpecificMode.	Struct	See chapter 4.2.3.18
M01	P	Toggles the Optional Stop (M01) validation. Each pulse inverts the state, enabling or inhibiting the optional stop depending on the previous state.
Manuelspeed	S	Selection of manual speed and Handwheel increment by the PLC. The following values are available:  The different speed values are defined by machine parameter P31 N0, N1 and N2.
Mode	S	Mode requested by the PLC. This variable will only be considered if the flag <i>WCNC.General.LockMode</i> is SET. The following CNC modes can be requested: The following values are available. 
NMAuto	S	N/M AUTO functionality. (see FlexiumPro Extended Programming Manual M00075) Set to enable the <i>NMAuto</i> (2/3, 3/5, etc.) functionality. This command is taken into account when 'CYCLE START' is Off.
NMAutoFeedOverride	S	Value of the Feed Rate potentiometer for the NMAuto. The value ranges from 0 to 255 (255 = 100%)
NoStopBlockEnd	S	No stop at end of block. When set, this flag enables a 'START' command in AUTO, SINGLE, MDI, RAPID modes as well as block sequencing in the AUTO and RAPID. When clear the 'START' will not be possible and if a program was running it will stop at the end of the current block until the flag is SET again and a 'START' command is issued. This flag should be TRUE in normal operation.

Name	Mode	Description
ProgramNumber	S	Program Number requested by the PLC. Used to set the active part program or request machining in drip feed mode. The activation is made on the rising edge of the flag <i>"WCNC.General.Reset"</i>. Possible values : 0: No part program activation requested. 1 to 9999 : Program number activation requested. -1 : Deactivation of the current part program -2: Drip feed mode requested.
ReloadDynOp	S	This flag can be used during the development of a dynamic operators in C application. It reloads the compiled C code routine without requesting a reboot of the CNC. Of course the C application must be developed in a way that a reboot is not necessary.
Reset	P	Request to reset in NCK in the default power up configuration with the exception of some E parameters (see FlexiumPro Programming Manual M00073). It will be taken into account only if there is no movement on the axes.
Start	P	<i>START</i> request. Starts part program or command line execution in AUTO, SINGLE, MDI, DRYRUN, SEARCH, and TEST modes. If a part program has been previously selected.
Stop	P	<i>STOP</i> request. Immediately stops program or command execution in AUTO, SINGLE, MDI, DRYRUN as well as incremental JOG modes. Machining is restarted by activating <i>"Start"</i>. This flags stop the execution of the movements controlled by the CNC. For cycles controlled by the PLC (e.g. Tool changer sequence), the programmer needs to take all appropriate actions.
StoreEnable	S	This flag is used to store the context (program number, line number, percentage of execution in the block, ... on Reset in order to facilitate a block search. The context is saved in NVRAM giving the possibility to restart even after a shut down. The fact that the store is made only when the flag is set allows the programmer to select additional conditions (e.g. Mode) without losing a previously saved context.
SynchAxisPowerOn	S	This flag informs the NCK that the drives are energized and able to move the axes. Its function is to enable the synchronization of Gantry axes as well as synchronized spindles.

4.2.3.17 Structure [WCNC.General.LockIncJog.](#)

When set to TRUE, Inhibits the selection of some Jog Increment values by the HMI.

Name	Mode	Description
Handwheel	S	Inhibits Handwheel operating mode
Inc_0_001	S	Inhibits Handwheel operating mode
Inc_0_01	S	Inhibits increment 0.01 selection
Inc_0_1	S	Inhibits increment 0.1 selection
Ind_1	S	Inhibits increment 1 selection
Inc_10	S	Inhibits increment 10 selection
Inc_100	S	Inhibits increment 100 selection
Inc_1000	S	Inhibits increment 1000 selection
Inc_10000	S	Inhibits increment 10000 selection
Inc_Cont	S	Inhibits continuous selection

4.2.3.18 Structure [WCNC.General.LockSpecificMode.](#)

When set to TRUE, Inhibits the selection of some operating mode by the HMI. Possible choices are:

Name	Mode	Description
Auto	S	Inhibits Auto operating mode selection
Edit	S	Inhibits Edit operating mode selection
Home	S	Inhibits Home operating mode selection
Manual	S	Inhibits Manual operating mode selection
MDI	S	Inhibits MDI operating mode selection
Rapid	S	Inhibits Rapid operating mode selection
Search	S	Inhibits Search operating mode selection
SingleStep	S	Inhibits Single operating mode selection
Test	S	Inhibits Test operating mode selection

4.2.3.19 Structure [WCNC.Handwheel\[..\].](#)

[..] Represents the handwheel number (1 to 4)

Syntax example: **FlexiumPro_NCK.WCNC.Handwheel[2].Axis:= 6;**

Name	Mode	Description
Axis	S	This byte variable select which axis (0 to 31) will be controlled by a particular handwheel [1 to 4].
Pulse	S	This word variable defines the number of counts read at the handwheel port. The CNC will calculate the increase. This variable is required for PLC handwheels (CAN or EtherCAT).

4.2.3.20 Structure [WCNC.IO.](#)

This structure is used to provide digital and analog values on the CNC front P2 connector.

As these ports can also be managed by the CNC itself semaphores are required to manage access rights (Lock).

Name	Type	Description
AnalogOutput[x].	Struct	Word Value of the analog output x (x = 1 or 2) The value (-10V to +10V) is expressed by absolute value on <i>b14 to b0 and the sign on b15</i> . It is output on RTK connector P2 and can be read by the Part program in E34100 and E34101. As the CNC is also able to write to this port, the flags <i>WCNC.IO.AnalogOutput[x].Lock</i> must be managed to prevent conflicts. See chapter 4.2.3.21
DigitalOutput[x].	Struct	Byte value of the digital set of outputs x (x = 1 to 4) The output value could range from 0 to 255 on RTK connector P2 and could be read by the Part program in E34010 to E34013. As the CNC is also able to write to this port, the flags <i>WCNC.IO.DigitalOutput[x].Lock</i> must be managed to prevent conflicts. See chapter 4.2.3.22

4.2.3.21 Structure [WCNC.IO.AnalogOutput\[..\].](#)

[..] Represents the AnalogOutput number (1 or 2)

Name	Mode	Description
Lock	S	When this flag is set to TRUE the NCK is forbidden to write on the analog output number x (x = 0 or 1). The respective output is then available to the PLC to write via the variable <i>AnalogOutput[x]</i> .
Value	S	Value in mv

4.2.3.22 Structure [WCNC.IO.DigitalOutput\[..\].](#)

[..] Represents the DigitalOutput number (1 to 4)

Name	Mode	Description
Lock	S	When this flag is set to TRUE the NCK is forbidden to write on the digital Output number x (x = 0 or 1). The respective output is then available to the PLC to write via the variable <i>DigitalOutput[x]</i> .
Value	S	Bit mask set the output

4.2.3.23 Structure [WCNC.Spindle\[.\].](#)

[.] Represents the physical spindle address (0 to 31)

Name	Mode	Description
Gear	S	At the Reset the value in this signal is assumed as currently gear inserted
IndexReq	S	Spindle Indexing Request When set to TRUE, this flag requests the spindle to index. It is only taken into account when <i>PLCSpeedControl</i> is TRUE.
Load.	Struct	Structure used to display the spindle load in the HMI. See chapter 4.2.3.24
Lock	S	Spindle Lock When set to TRUE, this flag will stop the spindle if it is running or prevent any rotation by disabling the reference. This flag is effective for any kind of spindle including slave spindles, contrary to the Stop flag below.
Override	S	Spindle speed override value The value ranges from 0 to 255, the applied value (%) depend from the potentiometers range defined P67 N16/N17: 0= P67N16, 255= P67N17 and it is applied if M48 active.
PLCSpeedControl	S	Spindle control When set to TRUE, this flag enables the spindle control by the PLC. The speed value (and direction) is given to the spindle drive by the variable " <i>Spindle[x].Speed</i> ".
Speed	S	Spindle speed value This variable is used to send the spindle servo-drive speed command in binary code on 15 bits. The spindle direction is managed by bit 15.
Stop	S	Spindle Stop Spindle stop requested by the PLC. When set to TRUE, this flag request the spindle to stop or remain stopped. This flag is effective in all cases except for slave synchronized spindles.

4.2.3.24 Structure [WCNC.Spindle\[.\].Load.](#)

[.] Represents the spindle physical address (0 to 31)

Name	Mode	Description
ByPLC	S	When this flag is set to TRUE the spindle load displayed corresponds to the value given by the PLC " <i>Load.Value</i> ". When set to FALSE the spindle load displayed comes from the drive.
Value	S	Load value displayed coming from the PLC (0 to 150).

4.2.3.25 Method [WCNC.WriteParameter\(\)](#)

	Name	Type	Description
	Parameter	DWORD	Parameter number to write
	NNumber	DWORD	Dword Number of the parameter to write
	pValue	POINTER TO BYTE	Pointer to variable of the value to wriet
	Size	DWORD	Size of the parameter
	Result	UDINT	RTS_IEC_RESULT

License Extend NCK Access is needed (FPSW282124)

1

2

3

4

5

6

7

A

4.3 Miscellaneous Read Data

Name	Description
CommunicationReady	Communication between NCK and PLC is operational.
CycleCounterNCK ⁽¹⁾	Number of NCK scan cycles (P50).
CycleCounterPLC ⁽¹⁾	Number of PLC scan cycles.
⁽¹⁾ These counters are made available for debug purpose. They are generally not required in a machine application.	

4.4 Methods and Functions

4.4.1 Method **LoadToolNumber**(

- Required Tool (= T code)
- Tool in spindle (tool in spindle after M06)

NC Block	N0
Tool	T100
Req. Tool	T30
Plane	X-Y
Compensation	Off

The required tool information is displayed immediately when a block with a T code is executed.

To display the tool in spindle, the CNC should be made aware of this value via the method *FlexiumPro_NCK.LoadToolNumber*.

4.4.1.1 Syntax

Answer := *FlexiumPro_NCK.LoadToolNumber* (parameters)

4.4.1.2 Parameters

	Name	Type	Description
	ToolNumber	DWORD	DWORD Toolnumber
	Channel	BYTE	Channel
	Result	BYTE	1=in Process, 0=OK, -1=Busy, -2=Error, -3=ToolMgr Used (P7 N3 Bit 3)

4.4.1.3 Result return codes

0	OK
1	In process
-1	Busy
-2	Error
-3	Tool manager is activated

4.4.2 Method ToolDyn()

Tool wear compensation by PLC without tool manager.

This function is intended to modify or clear to tool correction (L/R or X/Z) by a PLC command.

This requires temporarily inhibiting the tool wear correction by the HMI.



The value correction is given in Internal Units even though the type in LREAL.

4.4.2.1 Syntax

Answer := FlexiumPro_NCK.ToolDyn (parameters).

4.4.2.2 Parameters

	Name	Type	Description
	Correction	LREAL	Value of the wear offset change
	Command	INT	<code>EnumToolDynCommand.</code>
	ToolNumber	BYTE	Number of the Tool
	Result	BYTE	1=in Process, 0=OK, -1=Busy, -2=Error, -3=ToolMgr Used (P7 N3 Bit 3)

4.4.2.3 Result return codes

0	OK
1	In progress
-1	Busy
-2	Error
-3	Tool manager is activated

4.4.2.4 Example

```

CASE byStep_Tool_Dyn OF
    0:      IF Button_Pressed[3,6] THEN                // Command
            byP1 := byDNbr;                            // Latches the values
            lrP2 := INT_TO_LREAL(iCorrValue);
            FlexiumPro_NCK.WCNC.General.LockDynToolOffset := TRUE;    // Prevents HMI actions
            byStep_Tool_Dyn := 1;
        END_IF

    1: iAnswer := FlexiumPro_NCK.ToolDyn (ToolNumber := byP1,
        Correction := lrP2,
        Command := EnumToolDynCommand.IncrementCorrection_X_L);

        IF iAnswer = 0 THEN
            byStep_Tool_Dyn := 2;                        // Invokes method done
        END_IF

        IF iAnswer = -2 THEN                             // In case of error
            byStep_Tool_Dyn := 255;
        END_IF

    2: FlexiumPro_NCK.WCNC.General.LockDynToolOffset := FALSE; // Done - Release HMI action
        IF NOT Button_Pressed[3,6] THEN                // Prevents double action
            byStep_Tool_Dyn := 0;
        END_IF

        ELSE;                                           //In case of error
END_CASE

```

1

2

3

4

5

6

7

A

4.4.3 Probing

The probing includes three methods:

FlexiumPro_NCK.TouchProbe.ChannelsCfg	Links Interrupts inputs to channels. See chapter 4.4.3.1
FlexiumPro_NCK.TouchProbe.HardwareCfg	Configure the hardware (edges, filtering etc.). See chapter 4.4.3.2
FlexiumPro_NCK.TouchProbe.InputReading	Reads the probing input in the PLC. See chapter 4.4.3.3

Return values for the methods: RTS_IEC_RESULT

The status of probing inputs can be accessed by part program via the parameters E20100 to E20103.

FlexiumPro_NCK.RCNC.ProbeInput[x].Value ⁽¹⁾	Input status.
FlexiumPro_NCK.RCNC.ProbeInput[x].Counter ⁽¹⁾	Interrupt counter.

⁽¹⁾ [x] from 1 to 4, it corresponds to DigIn1 to DigIn4 on RTK P2 connector.
(See FlexiumPro Installation manual M00070)

4.4.3.1 Method [TouchProbe.ChannelsCfg](#)(

	Name	Type	Description
	LineNumber	BYTE	IT line Number
	Channels[x]	ARRAY [1..32] OF BOOL	Associated Channels
	Result	BYTE	

With this function each interrupt input can be linked to one or several channels.

- LineNumber = 1 corresponds to input DigIn1 on RTK P2 connector
- LineNumber = 2 corresponds to input DigIn2 on RTK P2 connector
- LineNumber = 3 corresponds to input DigIn3 on RTK P2 connector
- LineNumber = 4 corresponds to input DigIn4 on RTK P2 connector

4.4.3.2 Method `TouchProbe.HardwareCfg()`

	Name	Type	Description
	LineNumber	BYTE	IT line Number
	Valid	BOOL	Enable method
	Edge	INT	EnumTPEdge.  Falling  Rising  RisingAndFalling
	FilterFallingEdge	BOOL	Enable or disable filter for falling edge
	FilterRisingEdge	BOOL	Enable or disable filter for rising edge
	FilterTimeFallingEdge	INT	EnumTPTTime.  Time_05  Time_1  Time_4  Time_8 Filtering times are in ms.
	FilterTimeRisingEdge	INT	EnumTPTTime.  Time_05  Time_1  Time_4  Time_8 Filtering times are in ms.
	DelayTime	REAL	0 to 15.75ms by step of 0.25ms
	Result	UDINT	Return values: 0 OK 1 In progress -1 Busy -2 Error

DelayTime is used with some new types of probe of higher precision but inducing a constant delay before putting out the information.

The filter are activated if the corresponding flag are TRUE (FilterFallingEdge or FilterRisingEdge), but if those flags are put = FALSE, this doesn't mean that any filter is applied, but a default filter of 63µs is however active.

4.4.3.3 Method [TouchProbe.InputReading\(\)](#)

	Name	Type	Description
	LineNumber	BYTE	Line number interrogated
	Interrupt	POINTER TO BOOL	Memory of interruption occurred
	State	POINTER TO BOOL	State of the input
	Result	UDINT	Return values: 0 OK 1 In progress -1 Busy -2 Error

This method returns the status of the interrupt input and informs if an interrupt has occurred.

As reading the status will reset the flag '*Interrupt*', it is necessary to save it in another variable for further processing.

4.4.3.4 Structure [RCNC.ProbeInput\[..\]](#).

[..] This structure permits to check the status of the probing inputs of the NCK as well as counting the interrupts triggered:

Syntax example: **C:= FlexiumPro_NCK.RCNC.ProbeInput[1].value;**

Name	Type	Description
Channel[x]	ARRAY OF BOOL	Array of Boolean returning the channels on which the probe is associated.
Counter	BYTE	Byte variable returning the number of interrupts occurred on input 0 or 1. This counter must be read frequently enough as it rolls over after 255.
Enabled	BOOL	Boolean variable returning the status of the probe activation.
Value	BOOL	Boolean variable returning the status of the probing input 0 or 1.

4.4.3.5 Example

```

IF      NOT xProbingLioneInit THEN
    // Channel configuration
    Channel[1] := TRUE;
    Channel[2] := TRUE;
    FlexiumPro_NCK.TouchProbe.ChannelsCfg(LineNumber := 2, Channels := Channel);

    //interrupt line 1 is affected to channel 1 and 3

    IF      iProbingAnswer = 0 THEN
        xProbingLineInit := TRUE;
    END_IF

ELSIF   NOT xProbingHdwInit THEN
    // Hardware configuration
    iProbingHdwAnswer := FlexiumPro_NCK.TouchProbe.HardwareCfg(LineNumber:=2, Valid :=TRUE,
        Edge := EnumTPEdge.Falling, FilterFallingEdge:= TRUE,
        FilterRisingEdge := TRUE,
        FilterTimeFallingEdge := enumTPTime.Time_8,
        FilterTimeRisingEdge := enumTPTime.Time_8,
        DelayTime := 0);

    // Line 1 configured
    IF      iProbingHdwAnswer = 0 THEN
        xProbingHdwInit := TRUE;
    END_IF
END_IF

// Check if IT received and bit status
iStatusAnswer := FlexiumPro_NCK.TouchProbe.InputReading      (LineNumber := 2,
                                                                Interrupt := ADR(xGot_IT),
                                                                State      := ADR(xStatus_IT));

IF      iStatusAnswer := 0 THEN
    xIT_received S= xGot_IT;
    // As reading the statu reset the flag xGot_IT (interrupt has occurred)
    // it might be neccessary to latch the value for further use
END_IF

```

1

2

3

4

5

6

7

A

4.4.4 Messages and Warnings

The PLC can manage machine specific messages used as operator guidance, warnings or error messages.

These messages can initiate various reactions, defined as reaction classes.

8 different reaction classes are defined, each one having:

- A specific action on the machine
- An association with specific icon
- The possibility to be logged or not.

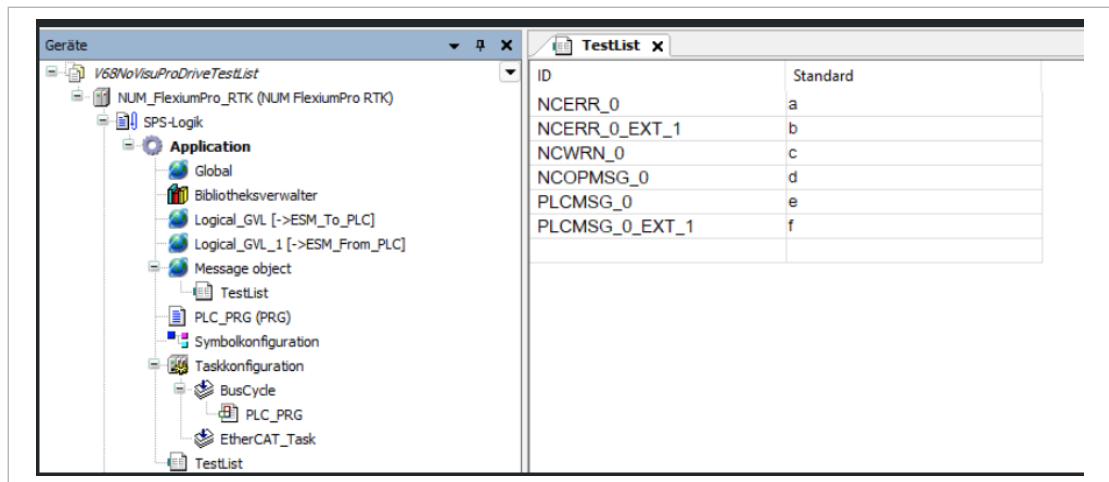
A second attribute of the message is the Receipt class which defines the condition to acknowledge a message

- Automatically when the condition is not present anymore
- On a specific action (Reset, E-Stop, Button etc. for example)

All messages of the same Receipt class have the same behavior.

4.4.4.1 Text Management in the Tools

In FlexiumPro all texts are located in the SD card and handled by FlexiumProTools, the system use the Codesys TextList linekd to specific object in in device tree (Message object).



The ID in TextList the specific key connects the text to kind of message:

Key value	Message type
NCERR_X	NCK error number
NCERR_X_EXT_Y	NCK errpr number and Extension number
NCWRN_X	NCK warning number
NCOPMSG_X	NCK operator message number
PLCMSG_X	PLC message number
PLCMSG_X_EXT_Y	PLC message number and Extension

4.4.4.2 HMI

The HMI use a Object from FXProServer to read the Text in a SD card and display the it. There will no text in the Setup of the Installations.

4.4.5 PLC Message Management

4.4.5.1 Define the characteristics of the receipt classes (once)

In the example below:

- *T#5s* : Means the message will self reset five seconds after the cause is gone.
- *T#0s* : The message will not self reset. To clear this message the complete receipt class should be cleaned.

```

IF NOT ml_init THEN

    //Receipt Class 1
    NUM_FlexiumPro_RTK.PlcMsg.SettingReceiptClass(ReceiptClass:=1,SelfReceiptTime:= T#0.0S);

    //Receipt Class 2
    NUM_FlexiumPro_RTK.PlcMsg.SettingReceiptClass(ReceiptClass:=2,SelfReceiptTime:= T#0.0S);

    //Receipt Class 3
    NUM_FlexiumPro_RTK.PlcMsg.SettingReceiptClass(ReceiptClass:=3,SelfReceiptTime:= T#0.5S);

    //Receipt Class 4
    NUM_FlexiumPro_RTK.PlcMsg.SettingReceiptClass(ReceiptClass:=4,SelfReceiptTime:= T#0.5S);

    ml_init := TRUE;
END_IF

```

4.4.5.2 Activate the messages when necessary

The function “*Insert*” is used to activate a message as well as defining the reaction and acknowledge classes.

FlexiumRTS.PlcMsg.Insert (IN, Number, ReactioClass, ReceiptClass)

Channel	Channel number
Components	Component in enum ErrorComponent like NCK_PLC_ERROR, PLC_CANOpen_Device etc.
ComponentAddress	number of component
ExtNumber	Extension number of the error
IN (Bool)	If TRUE the message is activated
Number (UINT)	Number of the message to activate (1 to 65535)
ReactionClass (BYTE)	Reaction class of the message (1-8)
ReceiptClass (BYTE)	Receipt class of the message (1-4)

In the example below, the message 'NbMsg' will be displayed when *xMsg1* becomes *TRUE*;

```
NUM_FlexiumPro_RTK.PlcMsg.Insert(IN:=PLCERROR[nr_err].Active,  
                                Number:=nr_err,  
                                ExtNumber:=PLCERROR[nr_err].ExtNumber,  
                                Components:=PLCERROR[nr_err].Components,  
                                ComponentsAddress:=PLCERROR[nr_err].ComponentsAddress,  
                                ReceiptClass:=PLCERROR[nr_err].CLASSTYP,  
                                ReactionClass:=PLCERROR[nr_err].REACTION);
```

4.4.5.3 Checking which messages are active

The method '*PLCMsg.IsActive*' permits to check if a particular message is active by returning its reaction class.

```
TestMsg 1 := RTS.PlcMsg.IsActive (Number := 103);
```

Message 103 is active and is from reaction class 1

4.4.5.4 Checking a reaction class

It is possible to test if some messages of a particular reaction class are active, for example to take an appropriate action.

```
xReactionClass1 FALSE := RTS.PlcMsg.Q_Reaction[1] FALSE;
```

e.g. There is currently no message of reaction class 1.

4.4.5.5 Checking a receipt class

In the same way it is possible to check the presence of some messages of a particular receipt class (for example to activate a type of acknowledge).

```
xReceiptClass := NUM_FlexiumPro_RTK.PlcMsg.Q_Receipt[1,chnr];
```

e.g.: There is currently at least one message of receipt class 1 and channel number [chnr].

4.4.5.6 Cleaning a class of messages

The method '*DeleteReceiptClass*' allows for erasing all messages of a particular class (except the messages whose condition is still *TRUE*).

```
NUM_FlexiumPro_RTK.PlcMsg.DeleteReceiptClass(IN:xDelete,ReceiptClass:=1,chnr);
```

4.4.6 Operator's message in different languages

A new syntax is defined to display messages in different languages on the HMI message window.

This new syntax is introduced by \$. (*\$ sign followed by a point*).

The previous syntax (*\$ alone*) remains for compatibility.

It is possible to include several variables in the message. They are entered as parameters separated by a *semi colon* “;”.

The value of the variable will respectively replace %1; %2, ...in the text string.

There is no limit to the number of variables. The only limit is the length of the string displayed : 255.

The management of the different languages is made on the HMI side.

The different languages descriptions have to be defined in proper files by the OEM and installed into proper directories of the HMI PC. These files are text files with a message description index (identifier number) for each message.

Messages concatenation remains possible like for \$ messages.

The syntax is \$.+ .

Syntax example: **\$.10;E80000** where:

- \$. = new syntax (similar to E. and EW.)
- 10 = message number to display
- ;E80000 = the value of the second argument will replace the %1 token in the message text

The message files contains lines like : **10 The value of MYVARIABLE is %1 mm** where:

- 10 is the message number
- %1 will be replaced by the variable invoked in the \$. Command

The variables can be parameters, symbolic or parametric expression or even strings (between single quotes) : \$.20;E80000/E80001;L9;[Num];12 .

4.4.7 Warnings

Warnings are generally defined by the NC firmware itself. It is however also possible to generate them by a part program command.

- Warning numbers up to 4999 are reserved to NUM.
- OEM specific warning (>= 5000) are not self cleared.
- It is possible to display a warning with parameters.

The following commands are available:

Warning format

Refer to the file *CN_Warning_xx* in the Flexium HMI language directory

Ex: **5010 Warning display test number %1**

%1 will be replaced by the first parameter passed in the EW. command.

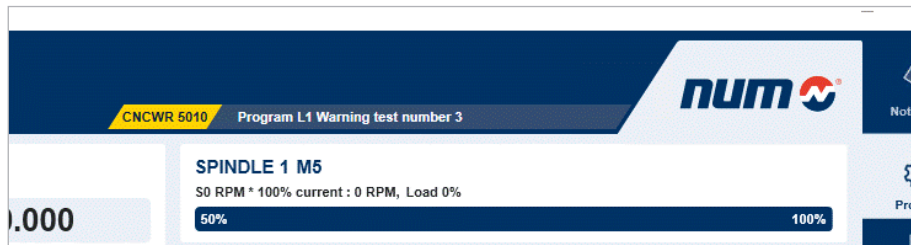
It is possible to have up to 4 parameters.

EW.nnnn/*/*

Display warning *number nnnn* with *parameters nnnn* is the warning number.

* represents a *single character* passed as a parameter to the display
(warning text permitting)

Ex : **EW.5010/3** results in the display of:



EW.-nnnn : Clears the warning message.

Like for operators messages, warnings can be displayed in different languages by entering the path-name in the section "*CncPlusWarningFilei*" of *language.ini*.

Warning numbers < 1000 are general and numbers >= 1000 are channel specific.

0-1999 Reserved for NUM internal use (NCK-Drives warnings).

2000-5000 Reserved for NUM Macro.

>= 5000 OEM Macro and free use.

4.5 Analog axes via EtherCAT

FlexiumPro allows for controlling analog axes via EtherCAT.

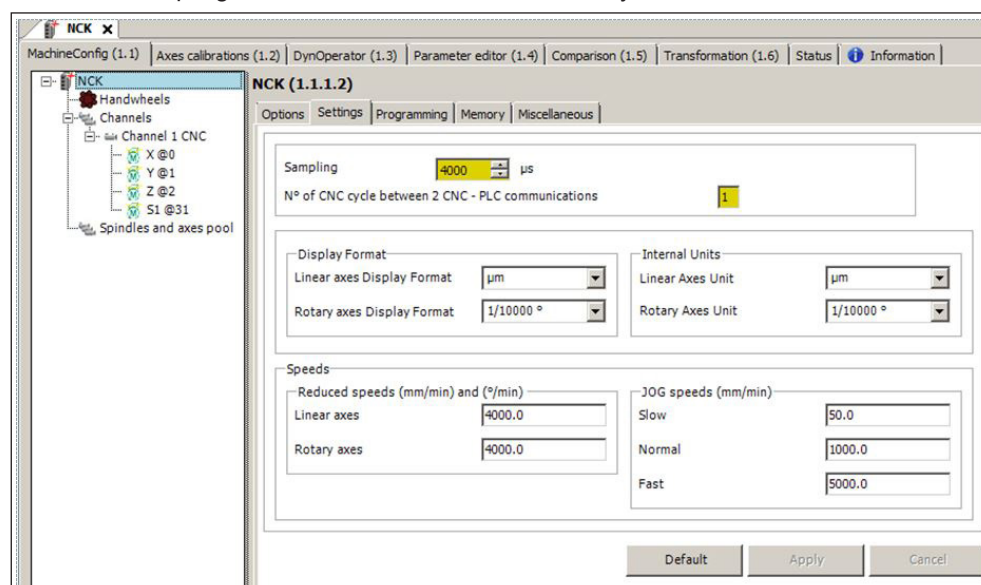
The measurement can be TTL SSI or EnDat

This feature is an equivalent to the analog axes directly connected on the NCK (X11 / X12) and extends the possibility up to 32 axes/spindles.

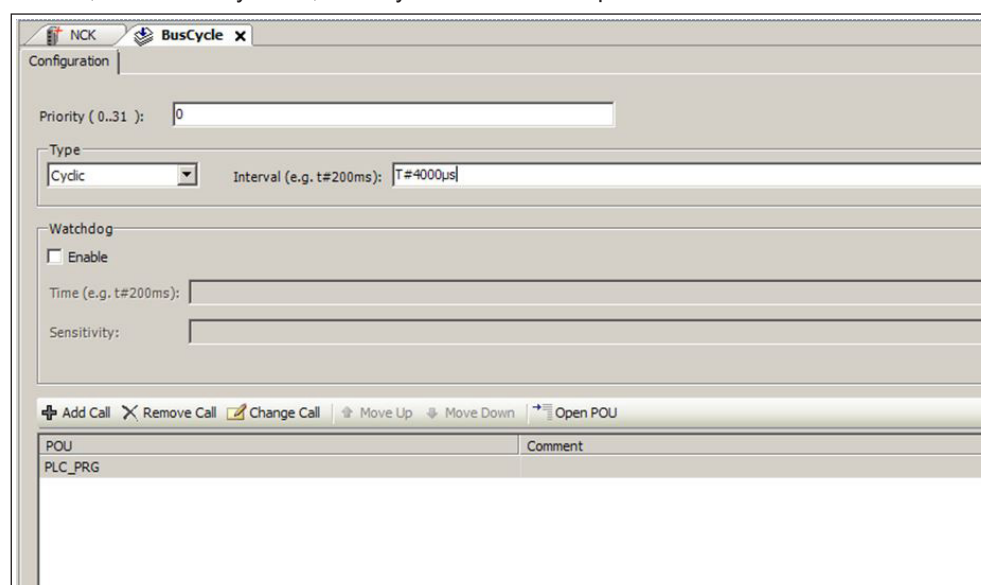
Each axis requires half of an EtherCAT Analog output module (CTMT 4132) and an EtherCAT measurement interface (CTMT5101 for TTL, EL5002 for SSI or EL5032 for EnDat).

4.5.1 General configuration

- Set the sampling time at 4ms and number of CNC cycles between ... at 1.



- Set, if not already done, Bus Cycle Task at 4000 μs .

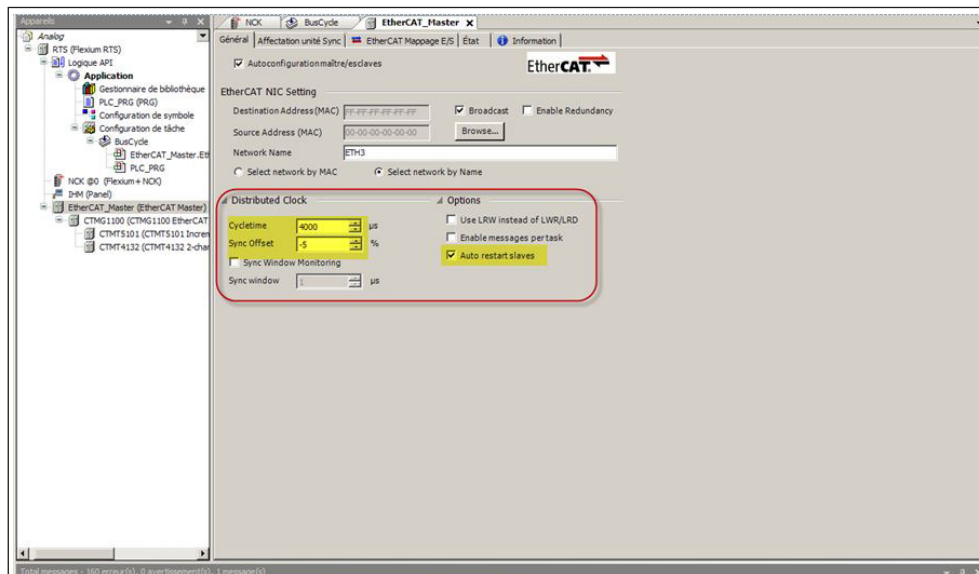


- In EtherCATMaster editor tab General configure as shown.

Distributed clock

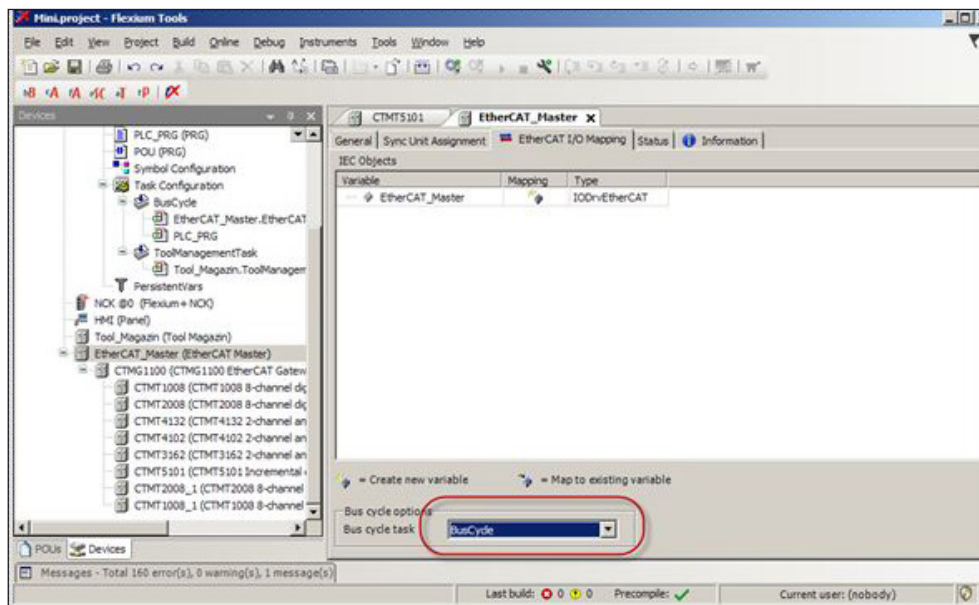
Cycle time = 4000µs

Sync offset = 5%.



- In the EtherCATMaster editor, tab EtherCAT I/O Mapping.

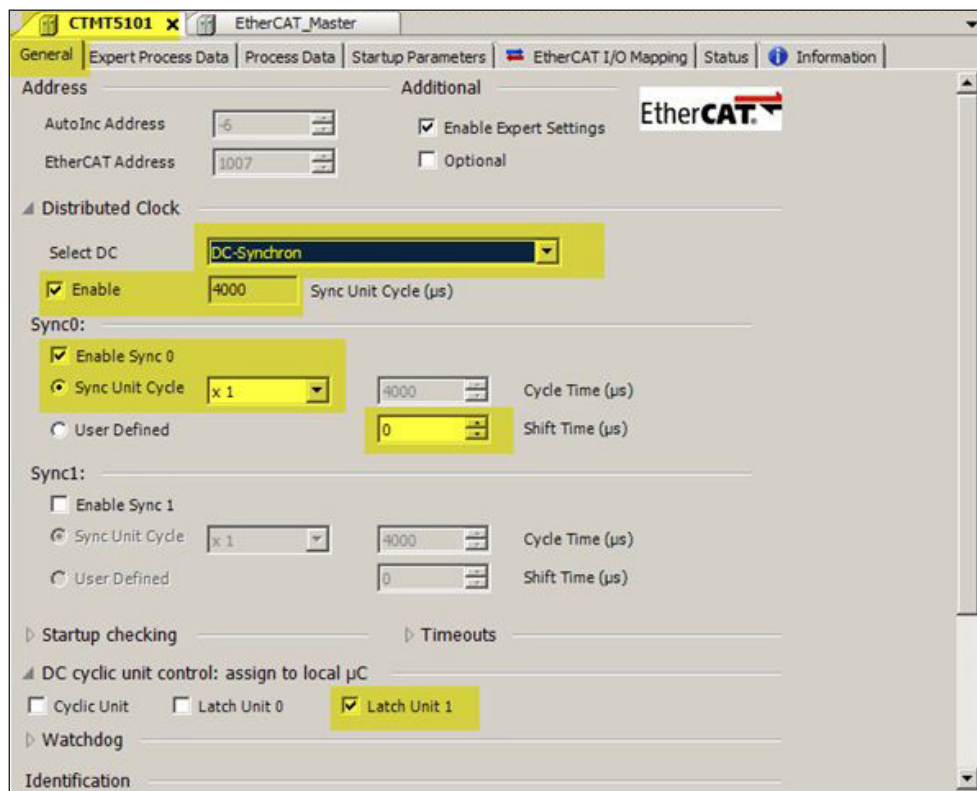
Select Bus Cycle for the Bus cycle task.



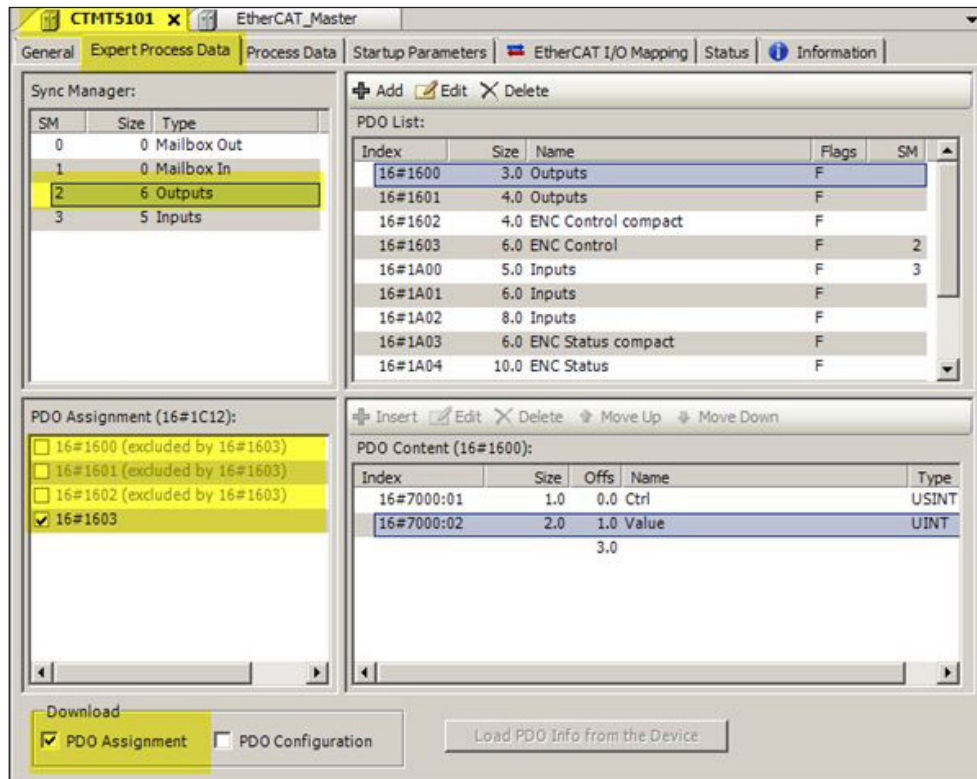
4.5.2 Interface module configuration

4.5.2.1 Configuration for TTL interface (CTMT 5101)

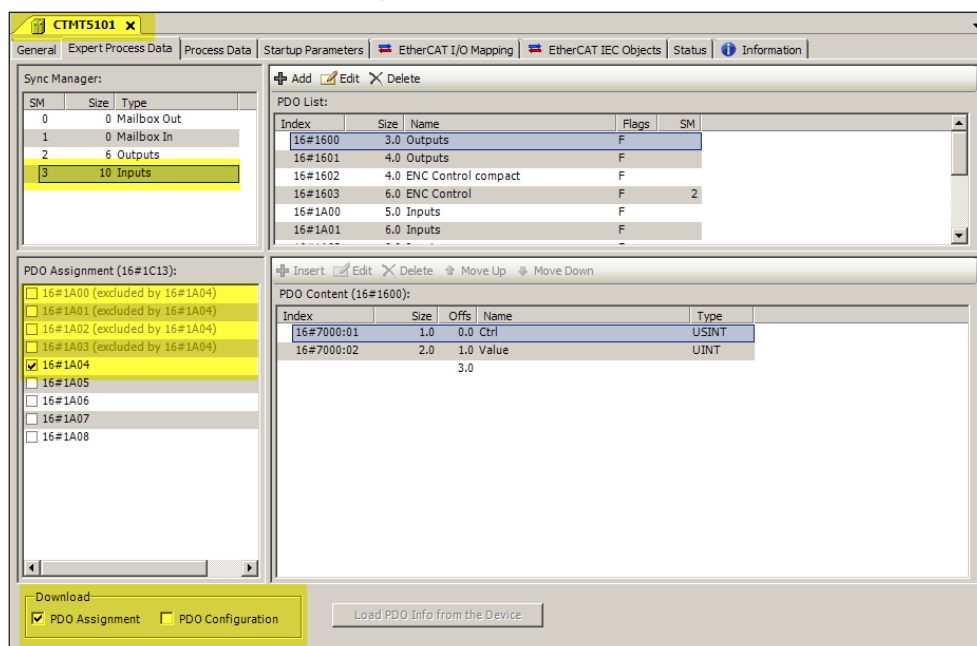
- In the CTMT 5101 editor, tab *General* validate as shown:
 - Distributed clock: *Select DC → DC synchron Enable.*
 - Sync 0:
 - Enable Sync0
 - Sync unit cycle = x1
 - Shift Time = 0.
 - DC Cycle unit control : *Latch Unit 1.*



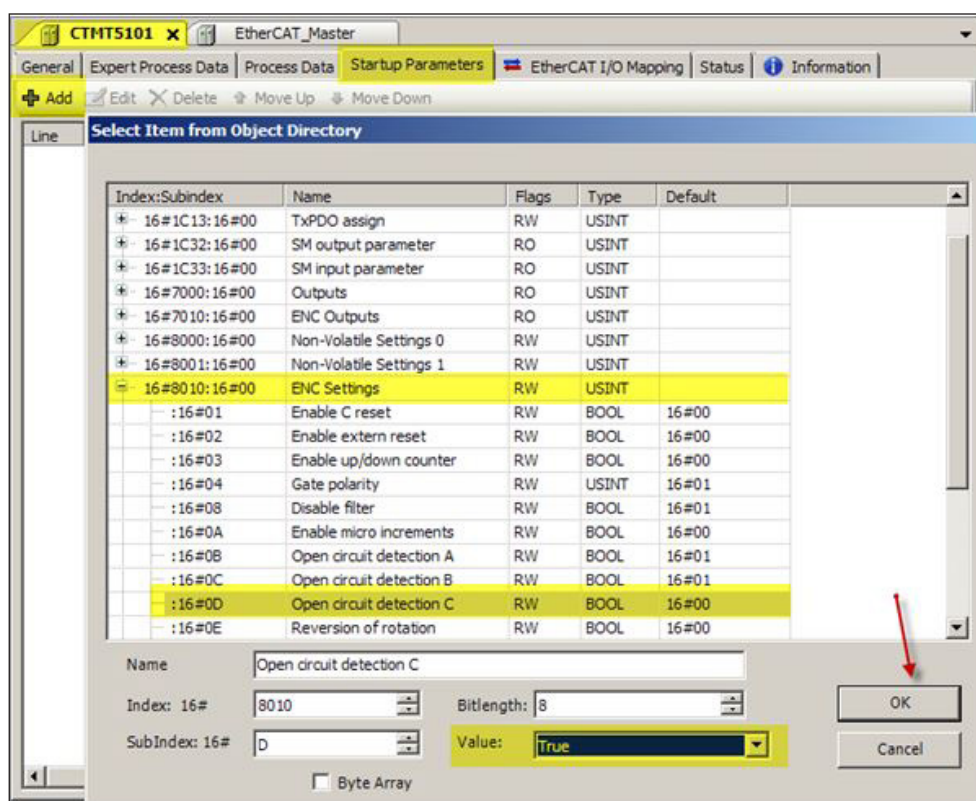
- In the CTMT 5101 editor, tab *Expert Process Data*:
 - Select *Outputs*.
 - In PDO assignment *unselect 16#1600* and *select 16#1603*.
 - Check “*Download PDO assignment*” as well.



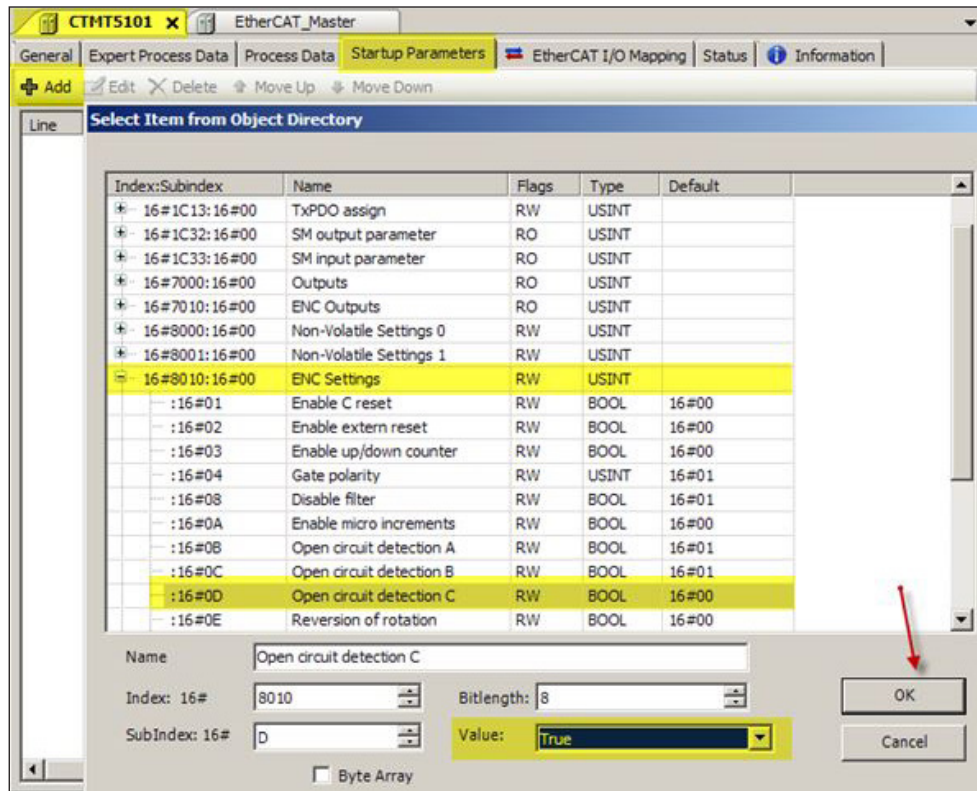
- In the CTMT 5101 editor, tab *Expert Process Data*:
 - Select *Inputs*.
 - In PDO assignment *unselect 16#1A00* and *select 16#1A04*.
 - Check “*Download PDO assignment*” as well.



- In the CTMT 5101 editor, tab *Startup parameters*:
 - Press + to add.
 - On line 16#8010.
 - Open the list (+).
 - Select "Open circuit detection C".
 - Push OK to add.



- In the CTMT 5101 editor, tab *Startup parameters*:
 - On line 16#8010.
 - Open the list (+).
 - Select “Open circuit detection C”.
 - Select value *True*.
 - Push *OK* to add.



- You should get this:

CTMT5101 x EtherCAT_Master							
General Expert Process Data Process Data Startup Parameters EtherCAT I/O Mapping Status Information							
Add Edit Delete Move Up Move Down							
Line	Index:Subindex	Name	Value	Bitlength	Abort if error	Jump to line if error	N
1	16#8010:16#04	Gate polarity	Disable gate	8	☐	☐	0
2	16#8010:16#0D	Open circuit detection C	True	8	☐	☐	0

- Define the I/O mapping for each axis (example shown for X axis)

Variable	Mapping	Channel	Address	Type	Unit	Description
X_Control_Enable_Latch_C		Control_Enable latch C	%QX12.0	BIT		Control
X_Set_Counter		Control_Enable latch extern on po...	%QX12.1	BIT		Control
X_Set_counter_Value		Control_Set counter	%QX12.2	BIT		Control
X_Latch_C_Valid		Control_Enable latch extern on ne...	%QX12.3	BIT		Control
X_Set_Counter_Done		Set counter value	%QD4	UDINT		Set cou
X_Input_Status		Status_Latch C valid	%IX12.0	BIT		Status_
X_Open_Circuit		Status_Latch extern valid	%IX12.1	BIT		Status_
X_Home_Switch		Status_Set counter done	%IX12.2	BIT		Status_
		Status_Counter underflow	%IX12.3	BIT		Status_
		Status_Counter overflow	%IX12.4	BIT		Status_
		Status_Status of input status	%IX12.5	BIT		Status_
		Status_Open circuit	%IX12.6	BIT		Status_
		Status_Extrapolation stall	%IX12.7	BIT		Status_
		Status_Status of input A	%IX13.0	BIT		Status_
		Status_Status of input B	%IX13.1	BIT		Status_
		Status_Status of input C	%IX13.2	BIT		Status_
		Status_Status of input gate	%IX13.3	BIT		Status_
		Status_Status of extern latch	%IX13.4	BIT		Status_
		Status_Sync Error	%IX13.5	BIT		Status_
		Status_TxPDO State	%IX13.6	BIT		Status_
		Status_TxPDO Toggle	%IX13.7	BIT		Status_
X_Counter_Value		Counter value	%ID4	UDINT		Counte
X_Counter_Latch		Latch value	%ID5	UDINT		Latch v

- Define as well the mapping for the analog module.

Variable	Mapping	Channel	Address	Type	Unit	Description
X_Analog		Output	%QW1	INT		Output
		Output	%QW2	INT		Output

4.5.2.2 Configuration for SSI interface (EL 5002)

The configuration of the device is limited to the tab “General” and the I/O assignment.

- In the CTMT 5102 editor, tab *General* validate as shown:

- Enable *Expert Settings*.
- Distributed clock:
Select DC: *DC synchron Enable*
- Sync 0:
Enable Sync0
Shift Time = 0
- DC Cycle unit control:
Latch Unit 1.

The screenshot shows the configuration window for the EL5002 device in the CTMT 5102 editor. The window has a title bar with tabs for BusCycle, CTMT5101, CTMT4132, PLC_PRG, and EL5002. The main area is divided into sections for configuration:

- Address:** AutoInc Address is set to -3, and EtherCAT Address is set to 1004.
- Additional:** ☒ Enable Expert Settings, ☐ Optional.
- Distributed Clock:** Select DC is set to DC-Synchron, ☒ Enable, and Sync Unit Cycle (µs) is set to 4000.
- Sync0:** ☒ Enable Sync 0, ☒ Sync Unit Cycle (x 1), Cycle Time (µs) is 4000, and Shift Time (µs) is 0.
- Sync1:** ☒ Enable Sync 1, ☒ Sync Unit Cycle (x 1), Cycle Time (µs) is 4000, and Shift Time (µs) is 15.
- Startup checking:** ☐ Startup checking.
- Timeouts:** ☐ Timeouts.
- DC cyclic unit control: assign to local µC:** ☐ Cyclic Unit, ☐ Latch Unit 0, ☒ Latch Unit 1.
- Watchdog:** ☐ Watchdog.

- Example for Y axis:

PLC_PRG GVL_PLC_PRG EtherCAT_Master CTMT5101

Find Filter Show all

Variable	Mapping	Channel	Address	Type
Y_SSI_DataError		Status__Data error	%IX16.0	BIT
Y_SSI_FrameError		Status__Frame error	%IX16.1	BIT
		Status__Power failure	%IX16.2	BIT
		Status__Sync error	%IX17.5	BIT
		Status__TxPDO State	%IX17.6	BIT
		Status__TxPDO Toggle	%IX17.7	BIT
Y_SSI_Counter		Counter value	%ID5	UDINT
		Status__Data error	%IX24.0	BIT
		Status__Frame error	%IX24.1	BIT
		Status__Power failure	%IX24.2	BIT
		Status__Sync error	%IX25.5	BIT
		Status__TxPDO State	%IX25.6	BIT
		Status__TxPDO Toggle	%IX25.7	BIT
		Counter value	%ID7	UDINT

4.5.2.3 Configuration for EnDat interface

Like for the SSI interface, the configuration of the device is limited to the tab “General” and the I/O assignment.

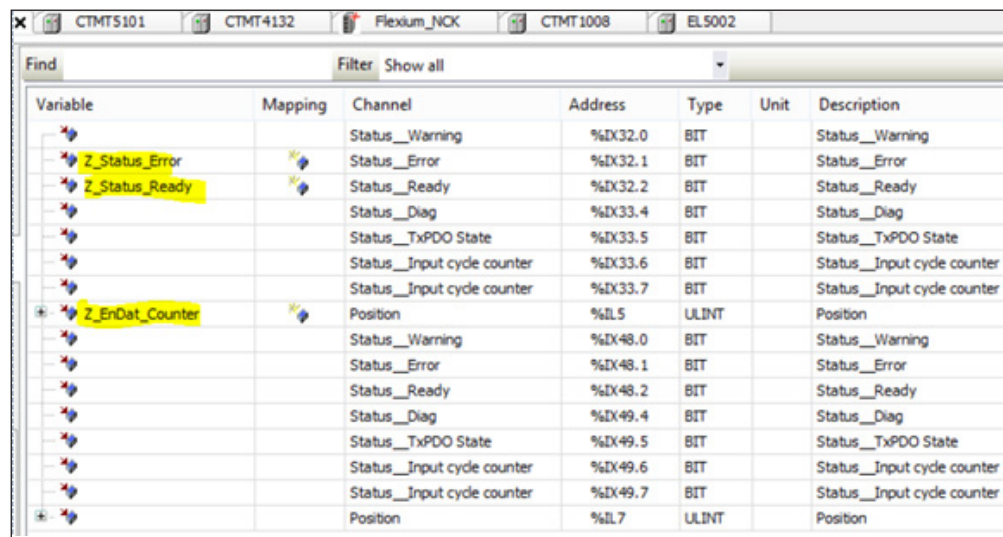
- In the EL 5032 editor, tab *General* validate as shown:

- Enable *Expert Settings*.
- Distributed clock:
Select DC: *DC synchron Enable*
- Sync 0:
Enable Sync0
Sync unit cycle = x1
Shift Time = -30
- DC Cycle unit control:
Latch Unit 1.

The screenshot shows the configuration window for the EL5032 device. The 'General' tab is active. The 'Address' section shows 'AutoInc Address' set to -4 and 'EtherCAT Address' set to 1005. The 'Additional' section has 'Enable Expert Settings' checked. The 'Distributed Clock' section shows 'Select DC' set to 'DC-Synchron' and 'Enable' checked with a 'Sync Unit Cycle' of 4000. The 'Sync0' section shows 'Enable Sync 0' checked, 'Sync Unit Cycle' set to 'x 1' with a 'Cycle Time' of 4000, and 'Shift Time' set to -30. The 'Sync1' section shows 'Enable Sync 1' unchecked, 'Sync Unit Cycle' set to 'x 1' with a 'Cycle Time' of 4000, and 'Shift Time' set to 0. The 'Startup checking' and 'Timeouts' sections are collapsed. The 'DC cyclic unit control: assign to local µC' section shows 'Cyclic Unit' unchecked, 'Latch Unit 0' unchecked, and 'Latch Unit 1' checked. The 'Watchdog' section is collapsed.

Section	Parameter	Value
Address	AutoInc Address	-4
	EtherCAT Address	1005
Additional	Enable Expert Settings	Checked
	Optional	Unchecked
Distributed Clock	Select DC	DC-Synchron
	Enable	Checked
Sync0	Enable Sync 0	Checked
	Sync Unit Cycle	x 1
	Shift Time (µs)	-30
Sync1	Enable Sync 1	Unchecked
	Sync Unit Cycle	x 1
	Shift Time (µs)	0
DC cyclic unit control: assign to local µC	Cyclic Unit	Unchecked
	Latch Unit 1	Checked

- Example for Z axis:



Variable	Mapping	Channel	Address	Type	Unit	Description
		Status__Warning	%IX32.0	BIT		Status__Warning
Z_Status_Error		Status__Error	%IX32.1	BIT		Status__Error
Z_Status_Ready		Status__Ready	%IX32.2	BIT		Status__Ready
		Status__Diag	%IX33.4	BIT		Status__Diag
		Status__TxPDO State	%IX33.5	BIT		Status__TxPDO State
		Status__Input cycle counter	%IX33.6	BIT		Status__Input cycle counter
		Status__Input cycle counter	%IX33.7	BIT		Status__Input cycle counter
Z_Endat_Counter		Position	%IL5	ULINT		Position
		Status__Warning	%IX48.0	BIT		Status__Warning
		Status__Error	%IX48.1	BIT		Status__Error
		Status__Ready	%IX48.2	BIT		Status__Ready
		Status__Diag	%IX49.4	BIT		Status__Diag
		Status__TxPDO State	%IX49.5	BIT		Status__TxPDO State
		Status__Input cycle counter	%IX49.6	BIT		Status__Input cycle counter
		Status__Input cycle counter	%IX49.7	BIT		Status__Input cycle counter
		Position	%IL7	ULINT		Position

4.5.3 PLC program

Once the configuration is finalized it is only required to execute a particular routine for each axis. They are 3 different routines according to the type of measurement.

```
// Example of Procedure for incremental encoder

NCK.PLCAAI (Addr      := X_addr,                      // Physical address of this axis
           LatchCValid := X_Latch_C_Valid,
           SetCounterDone := X_Set_Counter_Done,
           InputStatus  := X_Input_Status,
           OpenCircuit  := X_Open_Circuit,
           CounterValue  := X_Counter_Value,
           CounterLatch  := X_Counter_Latch,
           FCO           := X_Home_Switch,            // Home switch status
           EthCatRd      := EtherCAT_Operational,     // Check bus operational

           ControlEnableLatch => X_Control_Enable_Latch_C,
           SetCounter         => X_Set_Counter,
           SetCounterValue    => X_Set_counter_Value,
           AnalogValue        => X_Analog,             // Analog value written to module 4132
           FirstPLCRunEnd    := X_FirstPLCRunEnd      // Set to true after first succesful run
);

// Example for SSI encoder (for Y axis)

NCK.PLCAAI_SSI( Addr      := Y_Addr,
               CounterError := Y_SSI_DataError OR Y_SSY_FrameError,
               CounterValue := Y_SSI_Counter,
               EthCatRd     := EtherCAT_Operational,
               AnalogValue  => Y_AnalogValue,
               FirstPLCRunEnd := Y_FirstPLCRunEnd
);

// Example for EnDat encoder

NCK.PLCAAI_EnDat(Addr      := Z_Addr,
                CounterError := NOT Z_Status_Ready OR Z_Status_Error,
                CounterValue := Z_EnDatCounter,
                EthCatRd     := EtherCAT_Operational,
                PosPerRev    := 33554432,             // To adjust according to the encoder characteristics
                LinePerRev   := 16384,                // To adjust according to the encoder characteristics
                AnalogValue  => Z_AnalogValue,
                FirstPLCRunEnd := Z_FirstPLCRunEnd
);
```



It is absolutely necessary to check that the bus is working properly as explained in chapter 3.9.2.4. In the following example the variable *EthCatRdy* reflects this correct functionality.



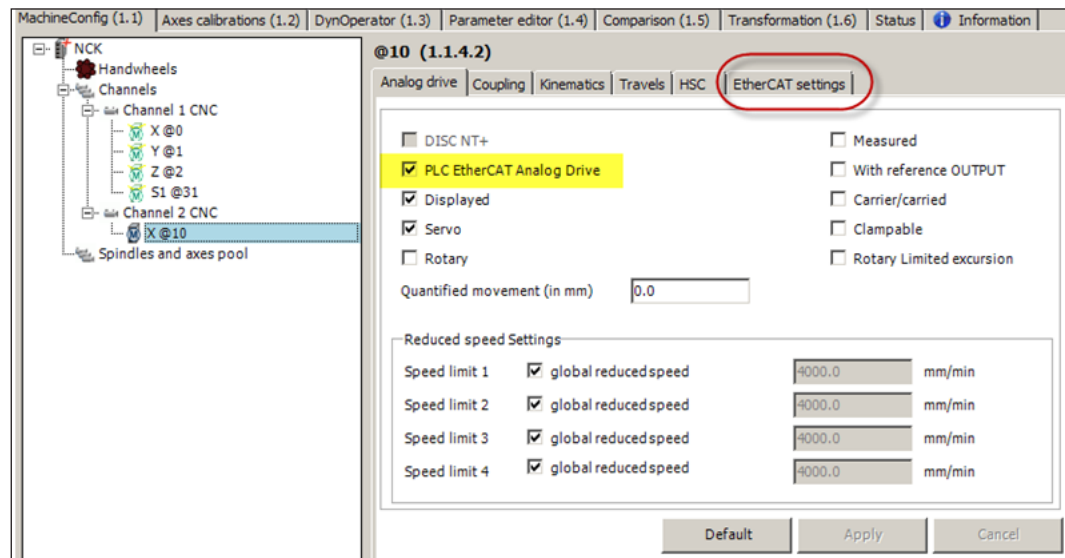
The ration *PosPerRev/LinePerRev* must be an integer value.

4.5.4 Parameterization

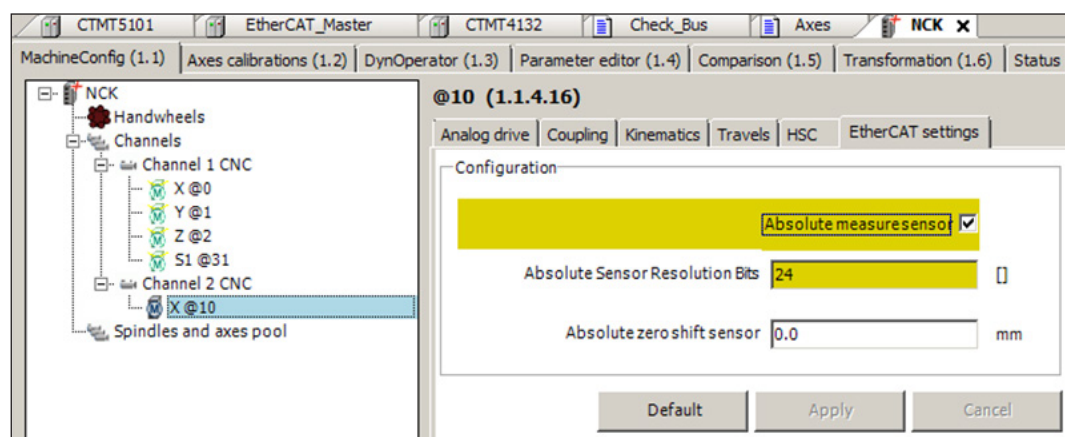
Pages not yet ready in the 1.00.00.10, the configuration is possible without wizard (P120, P74,P75).

EtherCAT controlled axes are declared like other axes except that the field PLC EtherCAT analog drive must be checked.

This creates an additional tab 'EtherCAT settings required for absolute measurements.



- The analog axis characteristic is reflected in parameter P120 N10.
- The measurement conversion in parameter P11.
- The servo system coefficient in P73 (calculated from the channel gain after entering the maximum speed).
- For SSI encoders the tab EtherCatsettings permits to define the Absolute characteristic as well as the number of bits of the data stream.



Page deliberately empty.

Summary

5	FlexiumPro functions	141
5.1	Automatic homing	141
5.1.1	Homing mode	142
5.1.1.1	Example of an homing program for multi channel	142
5.1.1.2	Automatic Homing Procedure in Homing Modes	143
5.1.2	Call by function G199: AUTO or MDI Mode	144
5.1.3	Case of axes synchronized by machine parameters	144
5.1.3.1	Case of several slave axes	144
5.2	Auxiliary Channels	145
5.2.1	Operating auxiliary axes	145
5.2.1.1	Reset	145
5.2.1.2	Auxiliary Axes	145
5.2.1.3	Programming	145
5.2.1.4	ISO Program	146
5.2.2	PLC exchange zone	146
5.3	Independent Channels	147
5.4	Duplicated and synchronized axes	148
5.4.1	Principle	148
5.4.2	Implementation	148
5.4.2.1	Defining Axes Couplings	149
5.4.2.2	Enabling axes duplication or synchronization	150
5.4.2.3	Synchronization	150
5.4.2.4	Enabling by Machine Parameters or E964xx	151
5.4.2.5	Enabling by External Parameters	152
5.4.3	Tuning	153
5.4.4	Restrictions	154
5.5	Tandem function	155
5.6	Spindle synchronization	156
5.6.1	Parameters	156
5.6.2	Activating the synchronization	157
5.6.3	Setting procedure	157
5.6.4	Example	158
5.7	Spindle commutation	159
5.7.1	Axis / Spindle commutation	159
5.7.1.1	General	159
5.7.1.2	PLC operation	160
5.7.1.3	Programming	160
5.7.1.4	Example	160
5.8	Delayed Power down in case of error	161
5.8.1	Example	162
5.8.1.1	CNC channel program	162
5.8.1.2	Auxiliary channel program	162
5.9	Acceleration profiles	163
5.9.1	Declaring the type of acceleration	166
5.9.2	Constraints related to trapezoidal or parabolic	166
5.9.2.1	Sequenced thread cutting (G38)	166
5.9.2.2	5-axes polynomial interpolation	166
5.10	Backtrack and resume	167

5.10.1	Auto recall.....	167
5.11	Smart Servo Balance	168
5.11.1	Adjustment.....	168
5.11.2	Settings.....	169
5.11.3	Verification: A correct setting can be verified with a Ball-Bar test.....	169
5.12	Electronic Gear box	170
5.12.1	General	170
5.12.2	Setup.....	171
5.12.2.1	Machine parameters	171
5.12.2.2	Leader axis.....	171
5.12.2.3	E parameters and PLC signals	172
5.12.2.4	Reset.....	172
5.12.2.5	Emergency	173
5.13	Five axes operation	174
5.13.1	General	174
5.13.2	Targeted kinematic chains	174
5.13.2.1	Five-axes kinematic chain.....	175
5.13.2.2	General settings	177
5.14	Kinematic measuring cycle248 with 3D touch probe	180
5.14.1	General	180

5 FlexiumPro functions

5.1 Automatic homing

For axes not fitted with absolute measurement systems, a specific homing macro is included in the CNC firmware. This macro recognises the type of axis (linear, modulo, ...) and include the processing for synchronized axes (one single Slave only).

For specific purposes (e.g. several slaves on one master axis) , it remains possible to retrieve this macro for editing (refer to FlexiumPro HMI manual M00076), however, it is however strongly suggested not to make any change for compatibility reasons.

Files

- **%9990.99** Homing macro main code
- **%9990** Defines the order of execution by calling %9990.9 with different parameters. Provides two unique features that make it reserved for an homing procedure:
 - It is the only program automatically executed if the user presses Cycle Start in homing mode
 - It is the only program where it is possible to move an axis that has not been previously referenced (homed).
- **%10199** Homing routine for call in MDI, Auto by means of G199

Two parameters must be passed before calling %9990.99:

- **L0** Axis number (0 .. 8 respectively X, Y, Z, U, V, W, A, B, C)
- **L1** Maximum retraction value. In case the axis is already on the home switch the retraction will not exceed the L1 value. This is to prevent an over travel in case the limit switch is defective. L1 should be positive in all cases.

Explicit programming of L1 is not mandatory, if its value is 0, then retraction is unlimited.

5.1.1 Homing mode

On machines with a single axis channel, homing is performed on each axis separately one after the other. %9990 should be edited to stipulate this order as well as the speed.

The standard file calls for homing Z then X and finally Y specify the order of execution and speed.

On machine with multiple channels, each channel should have a different program %9990.ii, 'ii' being replaced by the channel number. Such files do not exist as standard, they should be created as copy of %9990.

5.1.1.1 Example of an homing program for multi channel

Order and speed are specified in each %9990.ii .

As all channels can run simultaneously, it may be required to define synchronization points for preventing any collision.

There should be one such program for each channel.

If necessary synchronisation points must be defined to prevent any collision.

%9990.2	Example for Channel 2
IF [.RG80] = 999 THEN	
G77 H9990.9	Call by G199
ELSE	Call by Start in Home mode
L0=2 L1=20	Z axis retraction value limited to 20 mm
F2000 G77 H9990.9	Execution
G78 Q1 P1.1	Wait until channel 1 has executed the first operation
L0=0 L1=25	X axis, retraction limit 25 mm
F3000 G77 H9990.9	Execution
L0=1	Y axis
G77 H9990.9	Execution
ENDI	
M2	

The program for a single channel machine is quite similar – synchronizations points are not required. Obviously the synchronization as described is an example and should be adapted to the each machine.

5.1.1.2 Automatic Homing Procedure in Homing Modes

The user selects homing mode and presses the *Cycle* start button.
Programs %9990.i (or program %9990) are (is) started.

Homing is performed on each of the axes declared assuming they are declared in the channel and measured.

The operator is informed of the progress via messages:

- *X: MOVING OFF AXIS REFERENCE SWITCH*
- *X AXIS REFERENCE CYCLE IN PROGRESS*
- *X AXIS REFERENCING COMPLETE*

If homing was not possible, the cycle is stopped. The operator acknowledges the message below by pressing <Return>

- *X AXIS REFERENCING FAILED. QUIT:*

The following message is then displayed:

- *STOP AUTOMATIC REFERENCING CYCLE*

In case of error the following data should be controlled :

- P120, P9, P17, P16, P15, P1.

For more detailed information on these parameters please refer to FlexiumPro Parameters manual M00068.

1

2

3

4

5

6

7

A

5.1.2 Call by function G199: AUTO or MDI Mode

The call by G199 permits to execute homing on several axes in one single commands.

The list of axes is freely defined however the order of execution is fixed.

The syntax is as below:

- `[F...] [L1 = ...] G199 <axes list>` where `<axes list> = {X, Y, Z, U, V, W, A, B and/or C}`
- `F` : Feed rate for homing (optional)
- `L1` : Maximum retraction distance if the axis is already on the home switch (optional). If `L1 = 0` the retraction is unlimited.
- Homing is performed on all axes programmed in the list in the order XYZ UVW ABC.
- Homing is performed on the measured axes declared in the group.
- Macro %10199 is activated and calls programs %9990 (or 9990.i).
- Except in the case of a machine error, a return is always made to the calling program, **which is responsible for checking** whether homing was effective.
- Example: Homing of the C axis at 2000 deg/mm in MDI mode:
- `F2000 G199 C <Enter> and Cycle start`



CAUTION

Regardless of the order of the axes programmed after G199, the movements required for homing are carried out by the macro %10199 delivered in the order XYZ, then UVW and finally ABC.

5.1.3 Case of axes synchronized by machine parameters

Regardless whether it is called by function G199 or in homing mode, macro 9990.9 detects the presence of axes synchronized by machine parameters.

If only one axis is slaved to the programmed axis, homing is conducted jointly on the master and slave axes.

- Macro 9990.9 assumes that the master and slave axes have the same characteristics: type, homing direction, travel, switch presence, area covered by the switch, etc.
- Parameters P16 (reference switch position) may be slightly different for the two axes and therefore the encoder marker pulses may be reached "almost together" in the same movement.
- After homing the master axis, the macro continues movement over a length equal to the difference between P16 for the master and slave axes.

At the end of the cycle, the macro checks that the two axes are in the same state.

If homing is complete on both axes, the cycle ends. If not, the following message is displayed:

- `MASTER/SLAVE AXIS REFERENCE INCOMPLETE`

5.1.3.1 Case of several slave axes

If several axes are slaved to the programmed axis, the macro displays the following message at the beginning of the cycle:

- `MACRO 9990.9; ONLY CONTROLS ONE AXIS`

In such exceptional configuration, program 9990.9 will require to be adapted.

5.2 Auxiliary Channels

The function Auxiliary axes is intended to control digital axes independently of the machining. These axes are connected on the NCK but are not under direct control of a part program.

This functionality is additional to the CAN PLC controlled axes and provides an alternative solution for tool changer, palletization system or else.

The axes can be defined as PLCAxis by parameter P120 N8 Auxiliary axes. The PLCAxes cannot moved to CNC channels (chapter 5.2.1.2)

Being connected to the NCK, these axes share some features with the CNC interpolable axes:

- Axes are grouped per channel, they interpolate within a channel
- There is a maximum of 9 axes par channel
- The number of auxiliary channels is at most 32 minus the number of CNC channels considering that there is at least one CNC channel per NCK
- The maximum number of axes (auxiliary and CNC) connected to one NCK is 32
- The auxiliary axes are under control of an ISO program.



This function is intended for auxiliary axes (tool changer, pallet, etc.). Therefore it does not offer the same performances as regular axes in particular with respect to speed and acceleration.

5.2.1 Operating auxiliary axes

- Auxiliary axes can operate in all operating mode;
- All auxiliary channels are fully independent.

5.2.1.1 Reset

- Auxiliary channel can be reset via the channel specific reset request.
- An CNC reset does not reset the auxiliary channels.

5.2.1.2 Auxiliary Axes

The axes configured as Auxiliary axes cannot linked to CNC channels, they are free of charge but only in Auxiliary channels.

5.2.1.3 Programming

Auxiliary axes are controlled via an ISO program.

However the following functions are not accepted:

- M01: Optional stop
- M12: Programmed intervention
- G75: Emergency disengagement.

5.2.1.4 ISO Program

Auxiliary axes are under control of one single program “%9998.ii”, “ii” being the auxiliary channel number. This program must be stored in the OEM or USER zone of part programs.

The program is automatically selected when a Channel reset command is issued.

5.2.2 PLC exchange zone

The specific functions for auxiliary channels are:

Function	Description
...WCNC.Channel[x].ChannelEnable	Information required, should be set except to control the auxiliary axes in <i>Jog or Home</i> mode.
...WCNC.Channel[x].PlcAxisMode	Boolean information: - <i>False</i> = Automatic. - <i>True</i> = Block by block.
...WCNC.Channel[x].Start	The channel must be enabled for the <i>Start</i> to be active.
...WCNC.Channel[x].Reset	Should be sent to activate the auxiliary channel.

The signification of the other flags is identical as for the regular channels.

5.3 Independent Channels

The function '*Independent Channels*' gives the possibility to control each channel in a different mode, each channel with its own commands and its own part program.

The *PLC Flags* are at the channel level. They are described in the previous chapters of this document. For memory they are the following:

Read functions	Write functions																																										
<table><tr><th>Name</th></tr><tr><td>⌚ M01</td></tr><tr><td>⌚ BlockSkip</td></tr><tr><td>⌚ Intervention</td></tr><tr><td>⌚ ProgramActive</td></tr><tr><td>⌚ ProgramStop</td></tr><tr><td>⌚ CncError</td></tr><tr><td>⌚ AxisNotInPosition</td></tr><tr><td>⌚ EmergencyRetract</td></tr><tr><td>⌚ AxisRecall</td></tr><tr><td>⌚ Start</td></tr><tr><td>⌚ Stop</td></tr><tr><td>⌚ Reset</td></tr><tr><td>⌚ CycleNumber</td></tr><tr><td>⌚ TreadCutting</td></tr><tr><td>⌚ Rapid</td></tr><tr><td>⌚ MFunctionNoAck</td></tr><tr><td>⌚ Mode</td></tr><tr><td>⌚ MFunctionAck</td></tr><tr><td>⌚ DecodeM</td></tr><tr><td>⌚ ToolNumber</td></tr></table>	Name	⌚ M01	⌚ BlockSkip	⌚ Intervention	⌚ ProgramActive	⌚ ProgramStop	⌚ CncError	⌚ AxisNotInPosition	⌚ EmergencyRetract	⌚ AxisRecall	⌚ Start	⌚ Stop	⌚ Reset	⌚ CycleNumber	⌚ TreadCutting	⌚ Rapid	⌚ MFunctionNoAck	⌚ Mode	⌚ MFunctionAck	⌚ DecodeM	⌚ ToolNumber	<table><tr><th>Name</th></tr><tr><td>⌚ PlcAxisMode</td></tr><tr><td>⌚ Rapid</td></tr><tr><td>⌚ AckMFunction</td></tr><tr><td>⌚ PlcCallSubroutine</td></tr><tr><td>⌚ PlcInterrupt</td></tr><tr><td>⌚ ChannelEnable</td></tr><tr><td>⌚ EndExtMovement</td></tr><tr><td>⌚ FeedEnable</td></tr><tr><td>⌚ M01</td></tr><tr><td>⌚ BlockSkip</td></tr><tr><td>⌚ EmergencyRetract</td></tr><tr><td>⌚ AxisRecall</td></tr><tr><td>⌚ Start</td></tr><tr><td>⌚ Stop</td></tr><tr><td>⌚ Reset</td></tr><tr><td>⌚ FeedOverride</td></tr><tr><td>⌚ Mode</td></tr><tr><td>⌚ ProgramNumber</td></tr><tr><td>⌚ ReducedSpeed</td></tr><tr><td>⌚ ManualSpeed</td></tr></table>	Name	⌚ PlcAxisMode	⌚ Rapid	⌚ AckMFunction	⌚ PlcCallSubroutine	⌚ PlcInterrupt	⌚ ChannelEnable	⌚ EndExtMovement	⌚ FeedEnable	⌚ M01	⌚ BlockSkip	⌚ EmergencyRetract	⌚ AxisRecall	⌚ Start	⌚ Stop	⌚ Reset	⌚ FeedOverride	⌚ Mode	⌚ ProgramNumber	⌚ ReducedSpeed	⌚ ManualSpeed
Name																																											
⌚ M01																																											
⌚ BlockSkip																																											
⌚ Intervention																																											
⌚ ProgramActive																																											
⌚ ProgramStop																																											
⌚ CncError																																											
⌚ AxisNotInPosition																																											
⌚ EmergencyRetract																																											
⌚ AxisRecall																																											
⌚ Start																																											
⌚ Stop																																											
⌚ Reset																																											
⌚ CycleNumber																																											
⌚ TreadCutting																																											
⌚ Rapid																																											
⌚ MFunctionNoAck																																											
⌚ Mode																																											
⌚ MFunctionAck																																											
⌚ DecodeM																																											
⌚ ToolNumber																																											
Name																																											
⌚ PlcAxisMode																																											
⌚ Rapid																																											
⌚ AckMFunction																																											
⌚ PlcCallSubroutine																																											
⌚ PlcInterrupt																																											
⌚ ChannelEnable																																											
⌚ EndExtMovement																																											
⌚ FeedEnable																																											
⌚ M01																																											
⌚ BlockSkip																																											
⌚ EmergencyRetract																																											
⌚ AxisRecall																																											
⌚ Start																																											
⌚ Stop																																											
⌚ Reset																																											
⌚ FeedOverride																																											
⌚ Mode																																											
⌚ ProgramNumber																																											
⌚ ReducedSpeed																																											
⌚ ManualSpeed																																											

The mode Independent channels is selected via the flag ...WCNC.General.IndependentGroup which to be taken into account require a PLC request reset ...WCNC.General.Reset.

The status 'Independent Channels' is confirmed via the value 16 in ...RCNC.General.Mode.

It is also shown in the Flexium HMI status window by the word INDEP as opposed to COMM for common channel.

Current Program and Mode can be selected by the PLC at the Channel level.

e.g. ... WCNC.Channel[x].Mode := EnumMode.MDI

They can also be selected via the HMI, in such case the selection will be for the channel currently displayed.

Current program is selected in MDI mode by typing %<prog number> followed by <Enter>.



5.4 Duplicated and synchronized axes

5.4.1 Principle

The duplicated and synchronized axes function links a “*Slave*” axis with a “*Master*” axis.

The system calculates the master axis reference variation and adds it to the Slave axis reference.

Several Slave axes can be driven by the same Master axis,

The Slave axes make the same movements as the Master axis. A Slave axis may however move symmetrically with respect to the Master axis.

Duplication is done without feedback between Slave and Master (axes get the same command).

Synchronization adds a feedback to compensate for difference in reaction as well as a control of the error between Slave and Master. When synchronization is active the symmetrical move is not authorised anymore.

5.4.2 Implementation

Implementation consists of:

- Defining the couplings (i.e. assigning a Master axis to a Slave axis),
- Enabling the couplings (i.e. choosing the type of duplication).

The Slave axis must have the same characteristics as the Master axis, i.e.

- Speed
- Acceleration
- Position loop time constant.

5.4.2.1 Defining Axes Couplings

- Couplings are defined by giving for each axis its 'Master' address or -1 if there is no Master.
- They are defined by machine parameter P27 or external parameters E94100 to E94131.
- Couplings defined by machine parameters can not be changed by E Parameters.

Example

E94105 = 2

E94106 = 2

Axes 5 and 6 are both Slaves of Master axis 2

Coupling without synchronization (also called Duplication)

The activation in manual or Auto mode must be made via E parameters

Coupling with Symmetry

For such Duplicated axes, the Slave axis moves symmetrically with respect to its Master axis.

Coupling with symmetry needs to be defined by E parameters.

Coupling with Synchronization

For such axes, the Slave axis is synchronized with respect to its Master axis.

Synchronization consists of measuring the difference in movement of the Slave axis with respect to the Master axis and adding a fraction of it into the Slave reference.

Coupling cannot be both synchronized and symmetric.

Synchronization Error is the difference between the Master axis measurement and the Slave axis measurement.

Two thresholds are defined for checking this error:

- Threshold 1 (default 160microns): the maximum permissible error after synchronization.
- Above this value, the system generates a message but allows resynchronization of the Slave axis with its Master axis.
- Threshold 2 (default 600microns): the maximum error above which the system generates a message and inhibits synchronization.

The error generated by overrun of thresholds 1 and 2 clears the *CNCReady* signal and displays *ERROR 39*. These thresholds are defined by machine parameter P24 and cannot be modified by programming.

1

2

3

4

5

6

7

A

5.4.2.2 Enabling axes duplication or synchronization

Function	Description
Enabling in Automatic Mode E960xx	This is defined by E parameters E96000 to E96031 respectively. Coupling is immediately enabled by a non-zero value in the slave relative parameter.
Enabling Synchronization E962xx, P28	This is defined by machine parameter P28 (immediately at power on) or by E parameters E96200 to E96231. A nonzero value in the slave relative parameter validates the synchronization.
Enabling in HOME, JOG modes or Cycle hold E961xx	This is defined by E parameters E96100 to E96131. Coupling is immediately enabled by a nonzero value for the slave relative parameter.
Enabling Symmetry E963xx	This is defined by E parameters E96300 to E96331. A nonzero value for the slave relative parameter indicates that symmetry is active on this coupling.
Enabling synchronization with offset = 0 E964xx	This is equivalent to synchronization by P28 except that can be activated during machining

For more detailed information on E parameters please refer to FlexiumPro Programming manual M00073.

5.4.2.3 Synchronization

Synchronization differs according to whether it is enabled by machine parameter P28 or external parameter.

Enabled by machine parameter	Master and slaves are aligned on the same measure position : typically case of a Gantry structure; master and slave are inked by the mechanic.
Enabled by E962xx	The offset present at the moment the synchronization is activated will be maintained as long as synchronization is active. This is typically the case of axes linked by the part after it is clamped.
Enabled by E964xx	Identical to the synchronization by machine parameter. For the synchronization to be effective, E962xx must also be written AFTER E964xx. Obviously the axes must be in close position before activating the synchronization.

5.4.2.4 Enabling by Machine Parameters or E964xx

Synchronization by machine parameter is enabled immediately at power on and cannot be cancelled by programming.

Function	Description
In Manual	Action on the <i>JOG control</i> of the Master axis moves the Slave axis synchronously with the Master axis. Action on the <i>JOG control</i> of the Slave axis is inoperative. An error message will be issued.
In Homing	A homing simulation from the keyboard or a homing request for a Slave axis are inoperative. An error message will be issued in case of trying to home the slave by the jog buttons. A programmed (G199) or manually controlled homing request for the Master axis is also applied to all its Slave axes, the synchronization error detection threshold 1 is inhibited. When all the axes have reached their home position and if the synchronization error is below threshold 2 the system will realign the coupled axes by gradually setting the Slave axis reference equal to the Master axis reference, then it initializes the control of the synchronization error. This requires the PLC flag <i>Flexium_Pro.WCNC.General.SynchAxisPowerOn</i> to be set. This flag should reflect the fact that the drives are energized and able to control the motors.  It is possible, although not recommended, to give a different name to the slave axis with respect to the master axis; this will obviously be the case in case of synchronization by E964xx. In this case if the homing routine tries to move the slave a synchronization error will occur.
In MDI and AUTO Modes	The coupling cannot be modified, programming the Master axis moves the Slave axis synchronously. Programming a Slave axis is prohibited.
Synchronization Error	An overrun of synchronization threshold will clear the <i>CNCReady</i> signal and generate ERROR 39. If the error is below threshold 2, an CNC reset will force the system to resynchronizes the coupled axes by gradually setting the Slave axis reference equal to the Master axis reference, assuming that the signal <i>Flexium_Pro.WCNC.General.SynchAxisPowerOn</i> is ON. If the error is above threshold 2, the error cannot be cancelled. It means the machine is excessively skewed and the axes must be repositioned manually then the system must be reset.

5.4.2.5 Enabling by External Parameters

The Slave axis and Master axis references can be different. The system performs synchronization by holding: *Slave axis reference - Master axis reference = constant difference*.

Function	Description
At Power On	The axes are independent. When an external parameter E962xx = will be set to 1 is programmed, synchronization of the corresponding coupling will become active.
In Manual	If synchronization is enabled, the movements of the Slave axis are synchronous with the Master axis. Otherwise, they are independent.
In Homing	Before enabling synchronization, homing is carried out independently on the axes. Homing simulation from the keyboard is authorized. Once the synchronization is enabled, if the axes were not homed, only homing of the Master axis is required (homing from the keyboard is refused). Since a difference is allowed between the references, there is no tentative of resynchronization.  In such a case, the software limits of the Slave axis are ignored.
In MDI and AUTO Modes	It is possible to modify the coupling by programming, e.g.: disable, re-enable, change of Master axis.
Synchronization Error	An overrun of synchronization threshold will clear the <i>CNCReady</i> signal and generate <i>ERROR 39</i>. Cancelling the Synchronization Error if the error is below threshold 2 after an CNC reset by the operator, as soon as signal <i>Flexium_Pro.WCNC.General.SynchAxisPowerOn</i> will be is set, the system resynchronizes the coupled axes by gradually setting the Slave axis reference equal to the Master axis reference. If the error is above threshold 2, it cannot be cancelled. The axes must be repositioned manually then the system must be reset.

5.4.3 Tuning

Function	Description
Parameter P24	Parameter P24 is used to tune the function. Please also refer to FlexiumPro Parameters manual M00068.
Words 00 to 31	Threshold 1 for each axis (default 160). A fault is generated if this threshold is exceeded but resynchronization is possible.
Words 32 to 63	Fraction of the sync error to be reinjected in the slave reference (default 250/1000).
Word 64	Maximum synchronization error (default 600µm). No possibility of resynchronization if exceeded.

5.4.4 Restrictions

The following conditions and restrictions apply to parameter setting and programming with axis couplings:

- **A CNC reset does not cancel the couplings programmed**
- A coupling can only be redefined if it is disabled (no on-the-fly change of Master axis).
- An axis cannot be its own Master.
- A coupling cannot be enabled for duplication, synchronization or symmetry unless it is already defined.
- A coupling defined by machine parameter P27 cannot be modified but can be enabled or disabled by programming.
- A coupling enabled by machine parameter P28 cannot be modified and must have been defined by parameter P27.
- The characteristics of a coupling cannot be modified unless the coupling is disabled.
- Synchronization can only be enabled if the Slave axis does not belong to any channel: either the axis has been freed by programming E7x005 = -1 (x = programming the address of the axis), or the corresponding word of parameter P9 contains FF.
- It is recommended to program function G09 (accurate stopping at end of block before going to next block) on the movements preceding and following a programmed synchronization.
- A programming error generates ERROR 92.

For more detailed information on machine parameters please refer to FlexiumPro Parameters manual M00068.

5.5 Tandem function

Three possibilities can be found under the name *Tandem function*:

- Anti Backlash
- Torque duplication
- Winding duplication.

These are NUMDrive X features. The tuning is done in the drive parameters.

The NCK parameters are required to activate the function, they can be set by the *Tab TandemFunction* available when an axis is linked to a digital drive and assuming the checkbox 'Tandem' has been activated in the option settings of the NCK advanced editor.

For detailed information, please refer to the NUM DrivePRO Parameters manual M00066.

1

2

3

4

5

6

7

A

5.6 Spindle synchronization

The system offers the possibility to synchronize one or several Slave spindles on a Master spindle. The functionality is achieved for a speed controlled Master spindle.

Synchronization requires the signals:

- ...WCNC.General.SynchAxisPowerOn to be set and the related spindles not to be locked
- ...WCNC.Spindle[x].Lock, PLC controlled
- ...WCNC.Spindle[x].PlcSpeedControl or limited in speed ...WCNC.Spindle[x].ReduceSpeed.

Synchronization is based on E parameters, some of them could be preset in the machine parameters.

5.6.1 Parameters



Refer to the function 'Synchro' in the FlexiumPro Programming manual M00073.

Spindle synchronization parameters		
E parameter	Function	Description
E90800 to E90831	Synchro	Enable synchronization on master spindle
E90832 to E90863	Synchro	Spindle speed for maximum torque
E90864 to E90895	Synchro	Spindle speed for maximum power
E90900 to E90931	Synchro	Position gap for master slave spindle synchronization
E90932 to E90963	Synchro	Spindle Integration constant
E94132 to E94163	Synchro	Master spindle for spindle synchronization

5.6.2 Activating the synchronization

Synchronization is activated by loading the physical address of the Master spindle into parameter E94132 to E94163 respectively for physical address 0 to 31 of the slave spindles of the Slave spindle.

Examples spindle @20 = master, @30 and @31 slaves

E94162 = 20

E94163 = 20

Synchronization is refused if the spindles are not homed (error 94).

The spindle must be in rotation before requesting synchronization. This will define the synchronization direction, forward if they rotate in the same direction, reverse if not.

5.6.3 Setting procedure

- Check that the speed control is accurate for all spindles.
- Reduce as much as possible the spindle acceleration ramp generated by the drives.
- If possible choose for the slave spindle the one with the highest dynamic.
- Adjust parameters E90832 to E90895 according to the motor characteristics.
- Accurately adjust the spindle acceleration (E90700 to E90731).
If the master spindle has a higher dynamic than the slave set them both at the Slave value.
- Adjust proportional gain on the Slave.
 - E90664 .. E90695
 - E90932 .. E90963 = 500 to start with
 - Cancel Integration coefficient
 - E90932 to E90963 = 0
 - Adjust Phase lead compensation
 - E90800 to E90831 = 0 to start with.
- To activate the spindle synchronization the slave spindle MUST be released (M61 - see P7 N2 to authorize the release of a running spindle) and the information *FlexiumPro.WCNC.General.SynchAxisPowerOn* set to TRUE.
- Activate the synchronization by E941xx(xx= 32 to 63) of the slave spindle = @ of the master spindle.
- Check the *sync error* by E909xx (xx = 00 to 31). If the synchronisation needs to be improved:
 - Adjust the Integration coefficient (E90900 to E90932) making sure there is no overshoot (value around 500)
 - Make sure there is no oscillation at high speeds by increasing the value of the phase lead compensation.

5.6.4 Example

%100.0 Example for spindle synchronization

- **Cancel previous synchronization of spindle @11**

E94143 = -1

M62:1

M62#1:2

- **Parameters setting**

E90842 = 3000 E90843 = 3000 (speed for maximum torque)

E90874 = 6000 E90875 = 6000 (speed for maximum power)

- **If necessary adjust the indexing acceleration and gain**

E90710 = 20000 E90711 = 20000 (acceleration)

E90674 = 1000 E90675 = 1000 (gain)

- **Phase lead correction**

E90810 = 1000 E90811 = 1000

- **Integral coefficient**

E90942 = 0 E90943 = 0

M4 M40 S100

(Start Master)

M3:2 S0:2

(Direction for slave)

M61:2

(Release slave MANDATORY)

EC180

(Sync offset)

E94143=10

(Spindle 11 is slave of spindle 10)

- **\$ Spindles Synchronized**

N5 G4 F.2

G79 E93543 =0 N5

(Wait for sync OK)

E90911 = 500

(Set Integral)

G4F2

M0

(For testing and adjustments in MDI)

E94143 = - 1

(Release sync)

M2

5.7 Spindle commutation

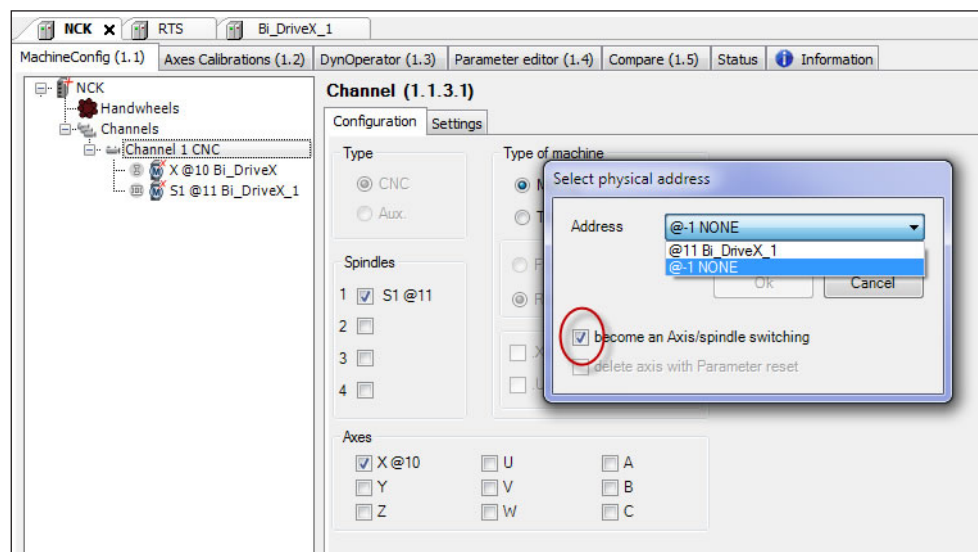
5.7.1 Axis / Spindle commutation

5.7.1.1 General

FlexiumPro offers the possibility to use the same kinematic either as an axis or a spindle (Switch between spindle and C axis). This feature is available for each rotary axis.

Operation with FlexiumPro Tools:

- Declare a spindle
- Declare a rotary axis and stipulate it can be a switchable unit.
 - Set this axis at the same address of the spindle to switch with:



In the NUMDrive X drive parameters set MULTI/DIVI (V306/V307) for the axis resolution (generally 1/10000°).

In this configuration, **P40** (measurement conversion in the spindle settings) **is not related to the number of counts anymore** but must be set to the resolution required for the spindle (generally 3600 or 36000).

The parameters E90464 to E90495 return the ratio between the Axis resolution and the spindle resolution.

A -1 value of this parameter means that a setting is not correct.

Obviously, the higher spindle resolution the better but keep in mind that the spindle count increase per scan should not exceed 32000.

e.g. Max speed = 8000 rpm. Resolution at 1/100° <-> P40 = 36000, Scan = 2ms



The internal resolution of the spindle is always the power of two immediately higher than the required resolution: in this case it will be 1/65536 turn. In the above example, at max speed the spindle makes 17476 inc. /scan, therefore the resolution of 1/100° is acceptable but a resolution of 1/1000° will not permit to rotate faster than 1870rpm.

5.7.1.2 PLC operation

Switching from spindle to axis (via E942xx) changes the drive from Speed control to Position Control. This information is available via the flag *ControlMode of ServoExtendedStatus*.

Take it into account for Torque and Speed validation.

It is possible to change the set of speed loop parameters of the drive (via E353xx) but the set of position parameters is generally not necessary as the kinematic is the same changing.

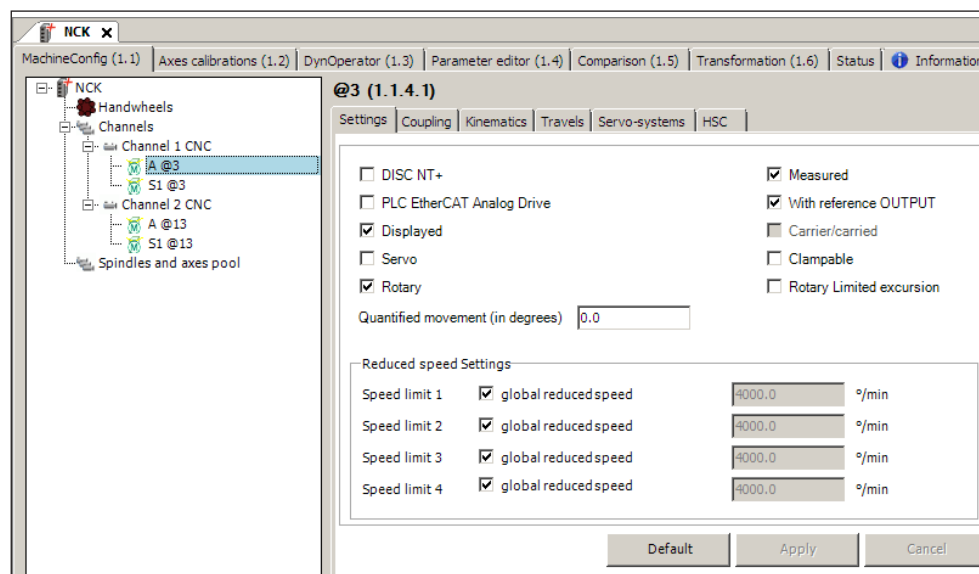
Changing it will result in the necessity to home the axis again. E35300 to E35331 define the set of parameters in axis mode whereas E35332 to E35363 defines it for spindle mode.

5.7.1.3 Programming

Only two instructions are required:

- E942xx = xx (for axis mode) or xx+32 (for spindle mode)
- E910xx (servo status of axe xx) should be set to 1 when axis mode and set to 0 otherwise.

5.7.1.4 Example



Considering the above configuration the switching spindle <-> axis is performed with the following macros.

Channel 1 set A as an axis

E94203 = 3 Set @3 coupler as an axis

E91003 = 1 Set servo status of coupler @ 3

Channel 2 set A as a spindle

E94213 = 45 Set @13 coupler as spindle

E91013 = 0 Open servo status of axis @13

5.8 Delayed Power down in case of error

This function is used when a machine shut down due to a servo error may have dangerous consequences (e.g. Grinder, Gear Hobber...).

It is intended to mask the error long enough to permit disengagement and return to a safe position.

The functionality is base on the following E parameters:

E93700 to E93731	Axis experiencing measurement error or excess following error. Boolean parameters related to axes 0 to 31. A parameter is set to indicate a measurement or excess following error on the corresponding axis. • These parameters are read-only and are cleared only on a reset. • These parameters can be read by dynamic operator. In this case, they consist in a list of 32 bits, where bit 0 addresses axis 0, bit 1 axis 1, etc. In this case, the entire list is addressed by E93700.
E92100 to E92131	Boolean parameters related to physical axes 0 to 31. • These parameters are read/ write by program and dynamic operators. • When accessed by dynamic operators, they consist in a list of 32 bits, where bit 0 addresses axis 0, bit 1 axis 1, etc. In this case, the entire list is addressed by E92100. • When a measurement or excess following error is detected - This error is immediately stored in E937xx - If the corresponding parameter E921xx is set, then the error 40+x or 210+x is not issued and the <i>...RCNC.General.CncReady</i> flag remains high. In case an excess following error occurs, the speed setting calculated during the previous sampling period is maintained. • After analysis of the possible errors causes, the bits set in list E93700 (axes in error) are reset in list E92100 (error masks). • The dynamic operators are executed • At the next scan, if an axis reports an error in the list 93700 and this error is not masked in the list 92100 then a machine error is generated and the <i>...RCNC.General.CncReady</i> flag is cleared.

Since the dynamic operators operate before the above mentioned step, they can delay the error generation by maintaining the faulty axis in list 92100. They can also monitor the following error.

Various actions are possible:

- Transmission to the PLC of the list of axes in error.
- Forcing the speed reference to zero for certain axes and braking at controlled speed on others using the channel feed rate override (E79002).
- Direct write on a PLC line for fast intervention of an external device, etc.

5.8.1 Example

Consider a CNC channel including two axes with addresses 0 and 1 and a auxiliary channel including two axes U and V with addresses 2 and 3.

If a excess following error occurs on an auxiliary axis, we want to force the speed setting for the auxiliary channel to zero and if the CNC axes are moving, request a stop of these axes by setting the feed rate override to zero, and then wait for the action to be performed before shutting down power.

5.8.1.1 CNC channel program

```
VAR
  [over]=79002
  [V_cn]=79004
  [lx_AP]=80000
  [lx_ERR]=93700
  [lx_CTRL]=92100
  [zero]=81000
ENDV

E[lx_AP]=12
E92102=1 E92103=1
E[zero]=0

01=9 E[lx_CTRL] / E[lx_AP] / / 5 /
02=5 E[ptar] / E[zero]
03=9 E[V_cn] / E[zero] / / 5 /
04=5 E[lx_CTRL] / E[lx_AP]
```

Addresses 2 and 3 = 1100 in binary = 12
First initialisation for monitoring axes 2 and 3

Test for fault on PLC axis – Jump to 5 if none
Set CNC override to zero
Wait If speed not zero
Reset mask lx_CTRL when speed Zero mask
reset ➡ CNCReady will be cleared

5.8.1.2 Auxiliary channel program

```
VAR
  [lx_ERR]=93700
  [zero]=81000
ENDV

E[zero]=0

05=9 E[zero] / E[lx_ERR] / / 8 /
06=2 E95002 / E90002 / E73000
07=2 E95003 / E90003 / E74000
```

Test for error on axes – Jump to 8 if none
Force auxiliary axes speed to 0
Idem for axis 3

5.9 Acceleration profiles

FlexiumPro provides for different types of accelerations:

- Constant
- Triangular
- Trapezoidal
- Parabolic.

It allows for a faster and smoother speed build-up and stopping especially at high speed..

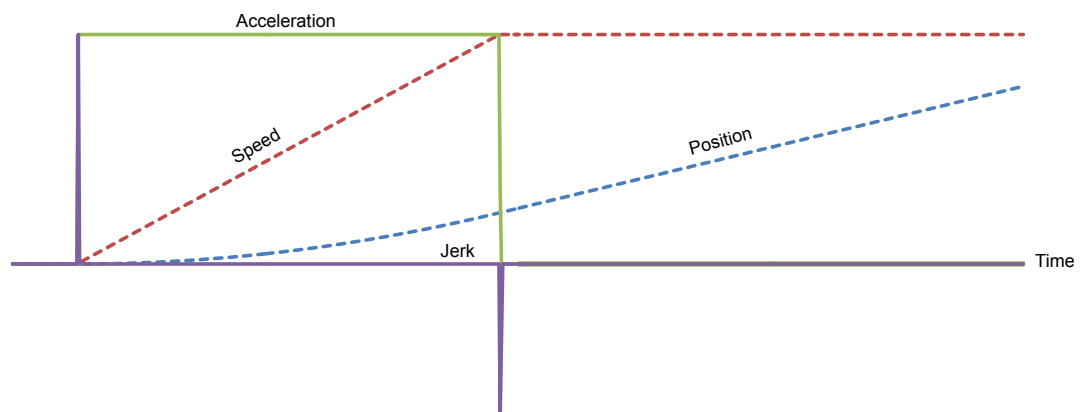
For more detailed information on E parameters please refer to FlexiumPro Programming manual M00073.

FlexiumPro provides four acceleration profiles. The selection is defined by machine parameters and can be modified by E11013 parameter during the process.

The different profiles are shown below. As they were made with identical parameter, they should not be compared for time performance because according to the selection the parameter setting can be different.

Constant acceleration profile

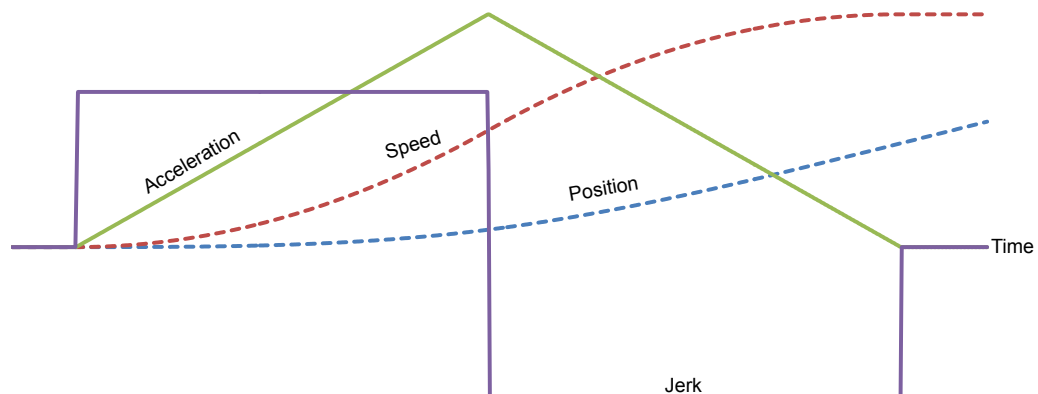
Basic profile – generally permits the fastest speed ramp up. However the very high jerk (acceleration variation) is the cause of shocks on the machine.



Triangular acceleration profile

The jerk is limited but the maximal acceleration is achieved only for a very short time.

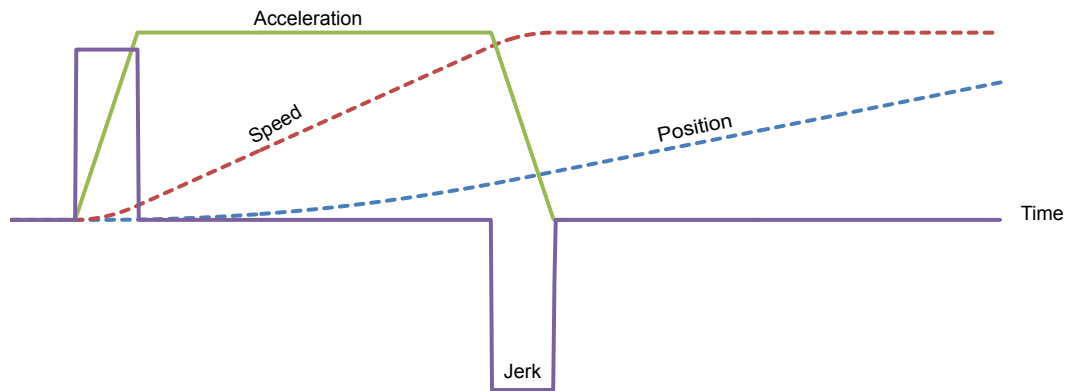
This profile is now seldom used.



Trapezoidal acceleration profile

This profile permits to reach the speed in a shorter and controlled time. This time is defined by parameter E32006.

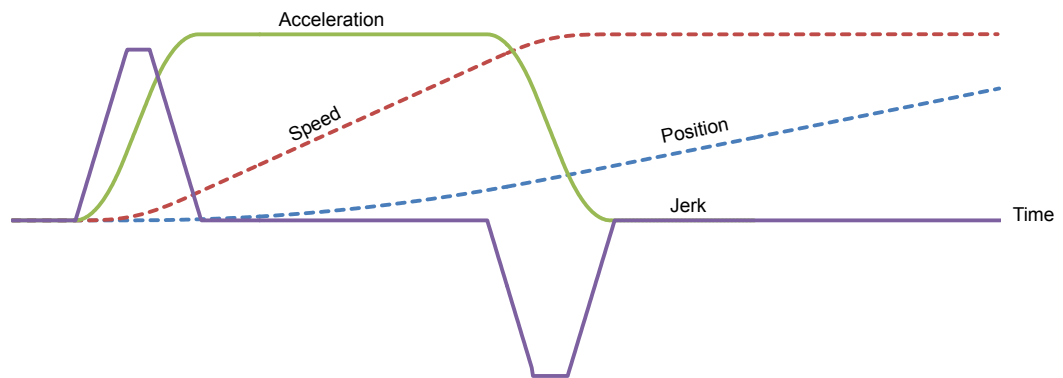
This permits higher performances, however they are still some jerk discontinuities.



Parabolic acceleration profile

Quite similar but the jerk variation is controlled (time to reach maximal acceleration in E32008). Therefore there are no more Jerk discontinuities and the shocks on the machine are minimal.

The time to achieve maximal jerk can be partly compensated by the possibility of a slightly higher acceleration without saturation.



E11013	Type of acceleration enabled by programming. The setting of this parameter is maintained after a reset. At power on, this parameter is set as defined by P53 N0: • P53 N0 = 0 E11013 = 0 Constant acceleration • P53 N0 = 1 E11013 = 1 Triangular acceleration • P53 N0 = 2 E11013 = -1 Trapezoidal or Parabolic acceleration.
E32006	This value in ms gives the time to achieve maximum acceleration (Trapezoidal or Parabolic). It is also the speed build-up time for very small speed levels, when speed variation $\leq 1 \text{ G} \cdot \text{sampling period}$. With trapezoidal acceleration, the setting of E32006 or P53 N1, N4, N7... is limited to 200 sampling periods.
E32008	Trapezoidal acceleration generates a constant jerk. A variation of this function is created to define a trapezoidal jerk. It is based on: E32008: Time (in ms) to reach a constant jerk The total acceleration time is defined by E32006 or P53 N1, N4, N7... Trapezoidal jerk is declared by: E11013= -1, E32006 = xxxx (acceleration rise time in ms) E32008 non zero At power on E32008 is set to zero if P53 N0, N3, N6,... is equal 0 or 1 and it is set to P53 N2, N5, N8 ... if P53 N0, N3, N6 = 2 (E11013=-1), it is programmable and its value is maintained after a reset. The setting of E32008 is limited to half the setting of E32006. E32008=0 (default value): Trapezoidal acceleration. E32008= P53 N1, N4, N7 /2: Trapezoidal jerk (limit value).

5.9.1 Declaring the type of acceleration

5.9.2 Constraints related to trapezoidal or parabolic

5.9.2.1 Sequenced thread cutting (G38)

These types of acceleration cannot be used with sequenced thread cutting.

During analysis of function G38 or G38+, if E11013 = -1 it will be forced to 0.

It is not re-enabled automatically and must be reprogrammed by E11013= -1.

If trapezoidal acceleration is present during a G33 thread cutting cycle, it is cancelled during the cycle, and automatically re-enabled at the end.

5.9.2.2 5-axes polynomial interpolation

Speed is managed differently in the 5-axis smooth polynomial interpolator than in the linear and circular interpolators.

In a sequence of blocks including linear axes and rotary axes with large movements on the rotary axes compared with the linear axes, when the classical interpolated blocks are followed by polynomial interpolated blocks, then a speed of zero is forced at the end of the block preceding the first polynomial block.

Similarly, when a polynomial block is followed by a line or a circle, a speed of zero is forced at the end of the last polynomial block.

5.10 Backtrack and resume

This is an optional function that can be activated after a feed hold. It permits to backtrack on the trajectory for several blocks (parameter defined) and resume the movement.

This function is available on any channel.

Backtrack along the programmed trajectory will be executed by a maintained action on

`.WCNC.Channel[...].BacktrackBackward`.

Resume by a maintained action on `.WCNC.Channel[...].BacktrackForward`.

The flag `.RCNC.Channel[...].BackForwardOnPath` indicates the status of the function.

In order to be able to backtrack, the system will memorize a certain number of trajectory blocks.

This number is defined in parameter P114.

Backtracking and return will be executed at the programmed speed. Resume can be executed in Auto, Block by block or Rapid and a change of mode is possible during this phase.

Please note that the T, S and M function are not memorized and will not be taken into account during the backtracking. Even though it is feasible backtracking should not include a tool change as this could lead to a collision.

Similarly, do not backtrack above a G10, G24, G26, G39, G39+.

It is possible to offset the programmed trajectory in Intervention mode, an axis recall will be necessary to cancel this offset. When no offset is active an action on the cycle start button is possible and normal execution will resume even before the interruption point.

For more detailed information on G codes please refer to FlexiumPro Programming manual M00073.

5.10.1 Auto recall

A similar function is auto recall. After axis disengagement in Intervention mode and in addition to the free return, it is possible to return along the disengagement path.

This return is activated by a permanent action on the flag `...WCNC.General.BacktrackAuto`.

The number of trajectory elements for this phase is named '*Number of stored points*' in FlexiumPro Tools and declared in word 2 of P114.

It is limited to 10. If more than 10 trajectory elements were executed the system will interpolate between the current position and the 10th.

The return speed is the jog speed. However the last part of the resume path could be executed at the programmed feedrate. The length of this last part (Recall distance) is declared in word 2 of P114.

5.11 Smart Servo Balance

Interpolated axes should have the same gain to achieve a correct machining. This is unfortunately not always possible without sharply decreasing the machine performances when an axis has really different characteristics.

However we may need to interpolate such axis with the others for a particular operation.

Smart Servo Balance will help to achieve a good interpolation of axes with different gain by adjusting the axis lag thanks to a phase lead and phase lag compensation.

This function relies on the following external parameters:

E98600 to E98631	Anticipation time constant in 1/100 ms
E98632 to E98663	Delay time constant in 1/100 ms
E92300 to E92331	Validation



This filter is exclusive of High Speed Machining and Partial feed forward. It must be inhibited during the Homing process. When the filter is active the time constant indicated above has a minimum value of 1 scan.

5.11.1 Adjustment

For a good adaptation all axes involved in interpolation must have the function enabled.

The adjustment will be made according to the highest time constant (lowest gain) in the channel.

The axis with the lowest performances will get the minimum values in its E986xx and E986xx+32 parameters.

The setting for the other axes is:

- Time constant of the axis in E986xx
- Time constant of the worst axis in E986xx + 32.

Example considering a scan period of 2 ms:

Axis	Address	Gain	Time constant (ms)
X	@0	1	60
Y	@1	2	30
Z	@2	3	20

5.11.2 Settings

X Axis

E92300 = 1	Activation
E98600 = 6000	This axis time constant
E98632 = 6000	Worst axis time constant 2ms as well Worst axis time constant 2ms as well

Y Axis

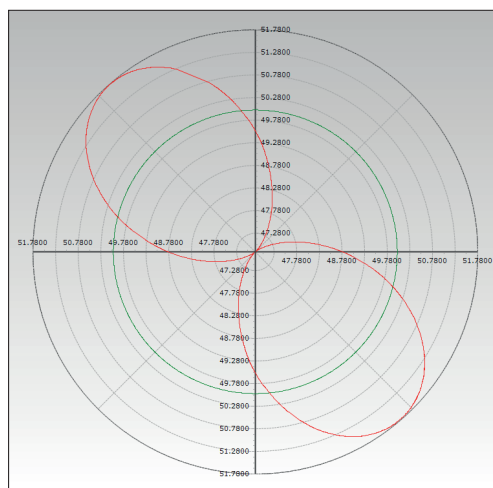
E92301 = 1	Activation
E98601 = 3000	Anticipation time constant = this axis time constant
E98633 = 6000	Worst axis time constant

Z Axis

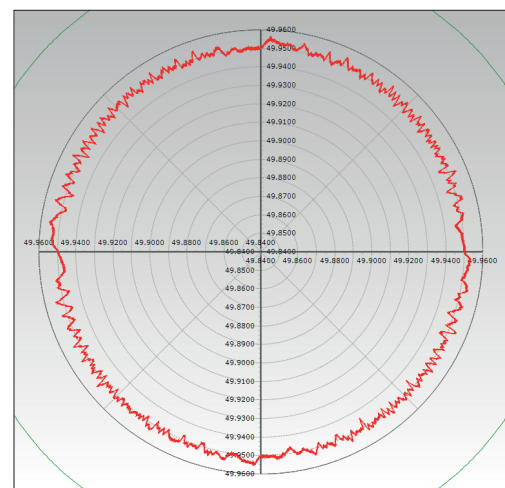
E92302 = 1	Activation
E98602 = 2000	Anticipation time constant = this axis time constant
E98634 = 6000	Delay time constant = Worst of the channel

5.11.3 Verification: A correct setting can be verified with a Ball-Bar test

Example: Two axes X with a gain of 2 and Y with a gain of 1



Without compensation



With compensation

Please note. This graph does not represent the trajectory but the error which is largely expanded around the nominal value; additionally the scales are different on both figures.

5.12 Electronic Gear box

5.12.1 General

FlexiumPro offers Electronic Gear box (EGB) as an option. This permits the synchronization between a leader and a follower axis.

This synchronization can be incremental. This means a follower can have several leaders and a follower can become the leader of synchronization (up to 5 cascaded leaders).

Synchronization is based by default on the reference of the axes (not the measure) and involves the use of the N/M function for continuous rotation of the master

The rule of synchronism can be a ratio or a table with linear or cubic spline interpolation between data points. Programming information is given in the FlexiumPro Extended Programming manual M00075.

The main functions are listed below:

G770	Start an axis in rotation.
G771	EGB synchro definition/destruction.
G775	EGB activation / deactivation.
G776	Waiting for EGB synchronization completion before proceeding.
G779	Synchro-homing (...or "gear alignment").

Example of program

G771+ C A P37 Q41

G01 A45 C135

G779 C

G770 A M3 S5000

G01 X... Y...

G775+ C

G01 X... Y...

G776 C F4

G01 X...Y...

Defines C-A synchronization with a ratio 37/41.

Synchronization not effective moving independently C and A axis to their synchro-homing.

Freeze synchro-homing position.

Start rotating A axis @ 5000 rpm counter clock-wise.

Prepare any other movement, note that C is still at standstill.

Activates the synchro "on the fly" of C to A, via C-A

EGB: ...C starts to accelerate, to reach A...

Additional optional movements, while C is synchronizing its reference to A.

Wait here for 4 seconds maximum, for C EGB synchro completion, before to proceed with PP execution.

Continue with PP...

5.12.2 Setup

5.12.2.1 Machine parameters

The axes are defined as usual with the following complements:

- The parameter P120 N11 contains the bitmap of the EGB leader axes which use the measured position to lead the follower (instead of using the standard theoretical positions).
- The parameter P116 N0-N31 contains the synchro tolerance. If the synchro error exceeds this parameter the machine goes in emergency.
- The parameter P116 N32-N63 contains the cut frequency for the low pass measure filter. If is set to 0 the filter is disabled.

5.12.2.2 Leader axis

The first step is to be able to run the top leader axis continuously.

This is similar to a spindle except that this axis is controlled in position and its speed override is controlled in Channel 8, in particular the rotation of this axis is not stopped by a operator stop command.

This axis is controlled via the N/M function that should be activated:

- Operation in Common channels mode only.
- E912xx = 1 (xx leader axis physical address).
- WCNC.General.NMAuto. must be set to TRUE.

Once this is done, it is possible to run this axis via the G770 command:

- G770 C M3 S2500 (forever run C axis counter-clockwise @ 2500 rpm.
- G770 C M5 (Stop C axis rotation).

Synchro definition and activation are handled by part program.

(see FlexiumPro Extended programming manual M00075)

5.12.2.3 E parameters and PLC signals

“x” stands for axis name in the channel (0-X , 1-Y, .. , 8-C).

E7x200 – 7x205	The term EGB synchro defines an offset to align the different axes before synchronization. These parameters contain the EGB synchro homing programmed with G779. E7x200 to 7x204 contains the first to fifth leader homing value, 7x205 contains the follower homing value.
E7x206	Contains the EGB synchro tolerance (image of P116 N0 to N31).
E7x207	Contains the current synchro error.
E7x208	E7x209 contains the minimum and maximum value of the EGB synchro error (E7x207).

The following flags found in the structure Axis[i] are intended to inform the PLC about the EGB status:

EgbActFollower	Indicates the axis is an active follower.
EgbEmeFollower	Indicates the axis is a follower in synchronization emergency.
EgbFwerSynczed	Indicates the axis a follower and is synchronized with their leaders.
Rotating	Indicates the axis is rotating continuously (like a spindle).
RotatingSpeedReached	Indicates the axis is rotating at the requested speed.

5.12.2.4 Reset

The Reset function doesn't open the EGB link. This is done to allow the operator to move the axis in manual or MDI after an emergency, maintaining if it is possible the synchronism.

The RAZ open the EGB link only in case of synchronization error.

When an axis is turning in NM-AUTO with G770, the RAZ procedure stops the NM-AUTO axis, waits that the axis actually stops and then terminates the procedure.

5.12.2.5 Emergency

In order to prevent damages due to a synchronization error, an emergency retract procedure should be prepared (G75).

This emergency retract procedure will be automatically triggered by a excessive synchronization error. If there is no emergency retract procedure programmed an error 162 ext. 2 will be issued.

If axes were synchronized at the time of this emergency, the NCK try to maintain all the follower axes synchronized.

When the emergency retract procedure is executed due to EGB synchronization emergency in E11022 parameter is written the value 3.

If at the moment of the emergency the follower axis is synchronized, during the retract phase the NCK try to maintain all the follower axes synchronized.

In E11023 parameter is written the physical address of the follower axis which went in emergency and the *PLC Axis signal EgbEmeFollower* is set.

1

2

3

4

5

6

7

A

5.13 Five axes operation

5.13.1 General

The five axes settings regroup Inclined Plane machining and RTCP (Rotation around Tool Center Point). To correctly execute the trajectory, it is necessary for the NCK to know the kinematic structure of the machine in order to execute the required compensation. Defining this structure is accomplished via the tab 'Transformation' in the NCK editor.

5.13.2 Targeted kinematic chains

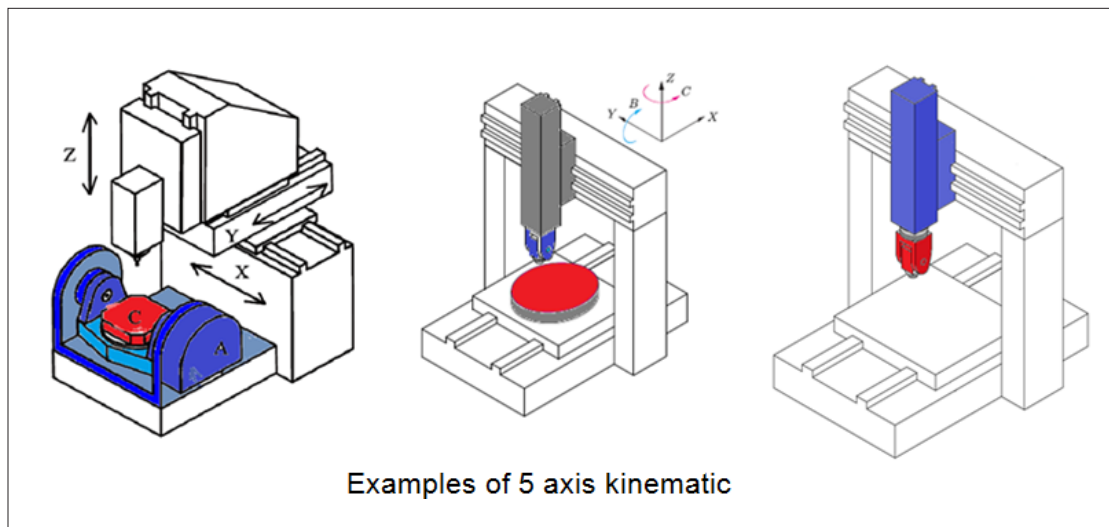
A 5-axis kinematics is mainly characterized by a swivel axis of the tool holder mechanical part.

This transformation takes in charge of the Five-axes machines with the following milling unit:

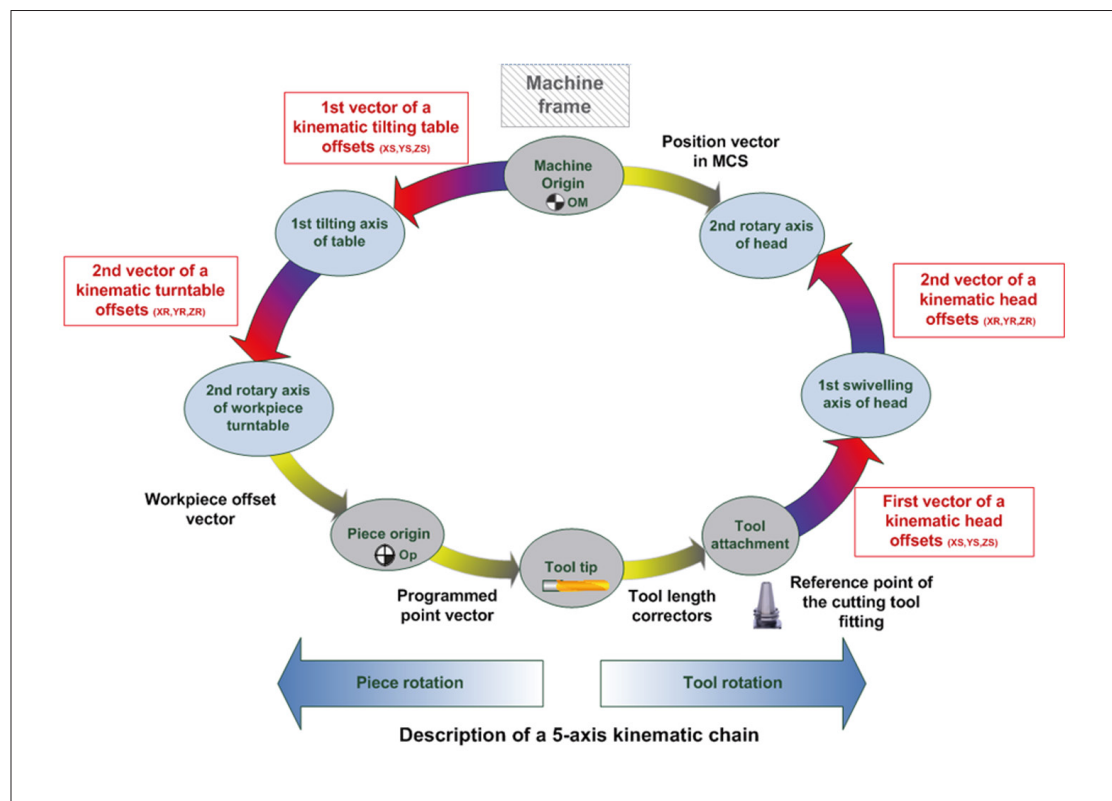
- A tilting turntable
- A swivel head and a rotary table combined
- A dual twist head but also a head with nutating¹ spindle.

It is well adapted to conventional 4-axes machine, equipped with either a single turntable or with a single swivelling axis head.

1) The word of "nutating" meaning an oscillatory motion or a slow spinning about an axis of a rotating body, a nutating head has a rotating pivot axis in an inclined plane.



5.13.2.1 Five-axes kinematic chain



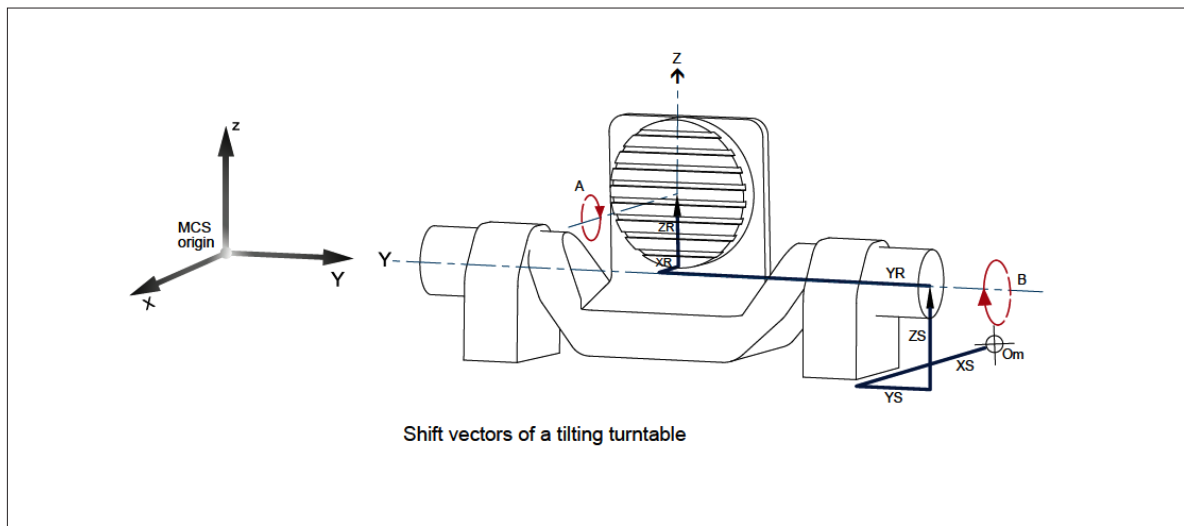
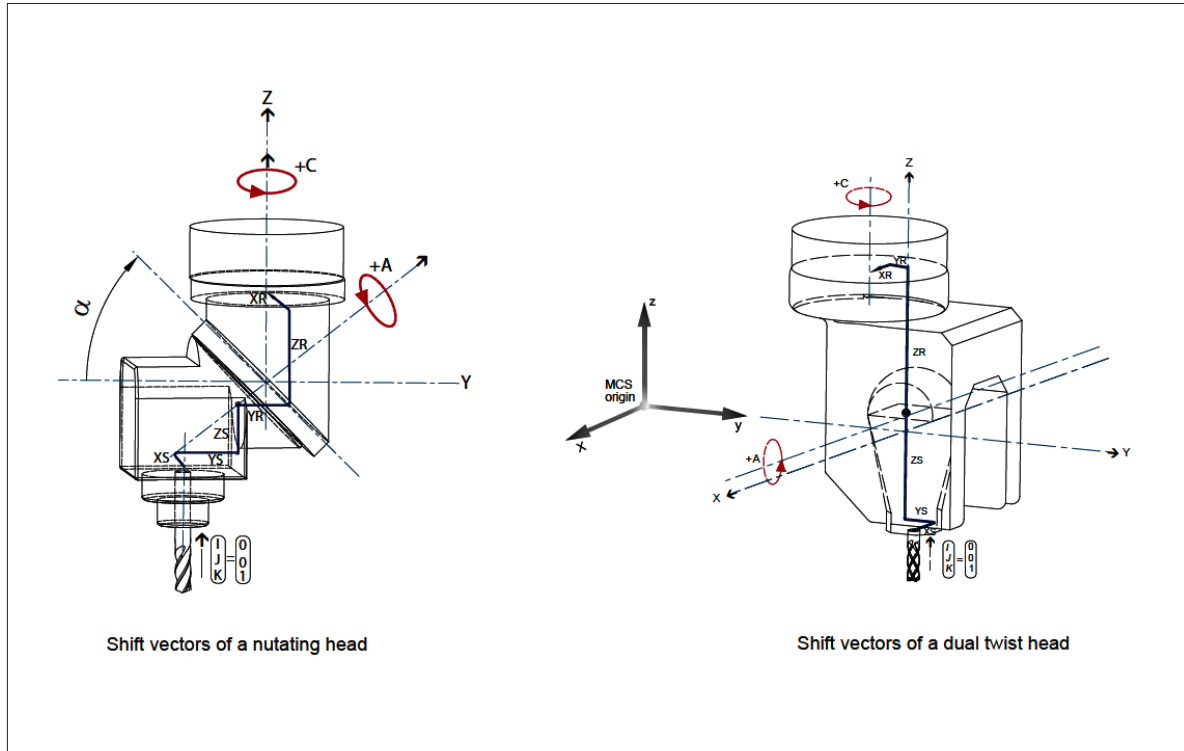
A Five-axes kinematics is mainly characterized by a swivel axis of the tool holder mechanical part or by a tilting axis of a turntable. "S" axis indicates the swivel/tilt axis.

The other, named "R" axis denotes a rotary axis of the tool kinematic part or of a kinematic part carrying the workpiece.

The first vector of kinematic offsets features either a swivelling axis, from the tool holder to the pivot axis or either a tilting table from the machine frame to the pivot axis. (XS, YS and ZS are its spatial components along xyz direction).

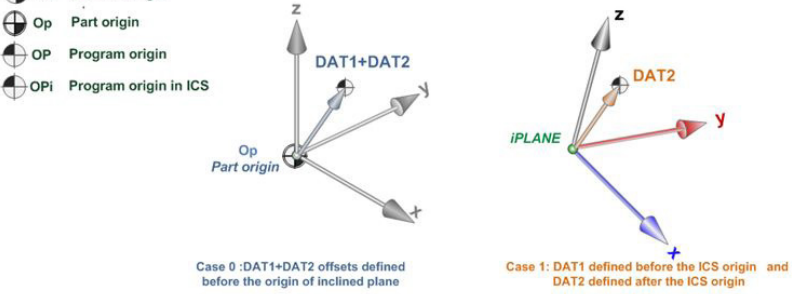
The second vector describes the kinematic offsets between the two pivot axes (XR, YR and ZR are its spatial components along xyz direction).

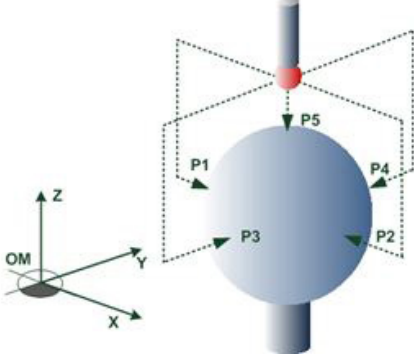
These two vectors are expressed with respect to machine frame O-xyz.



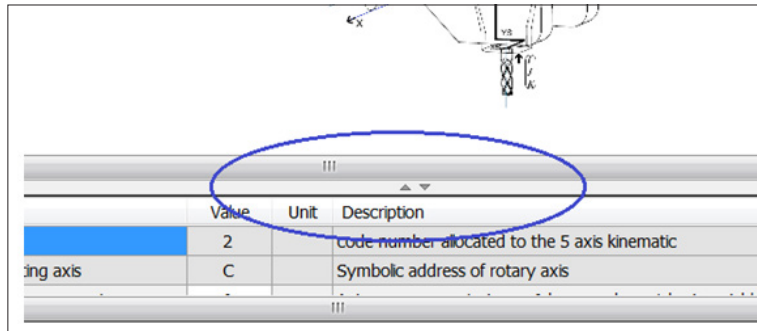
A list of different configurations managed by the RTCP function is described in annex A.

5.13.2.2 General settings

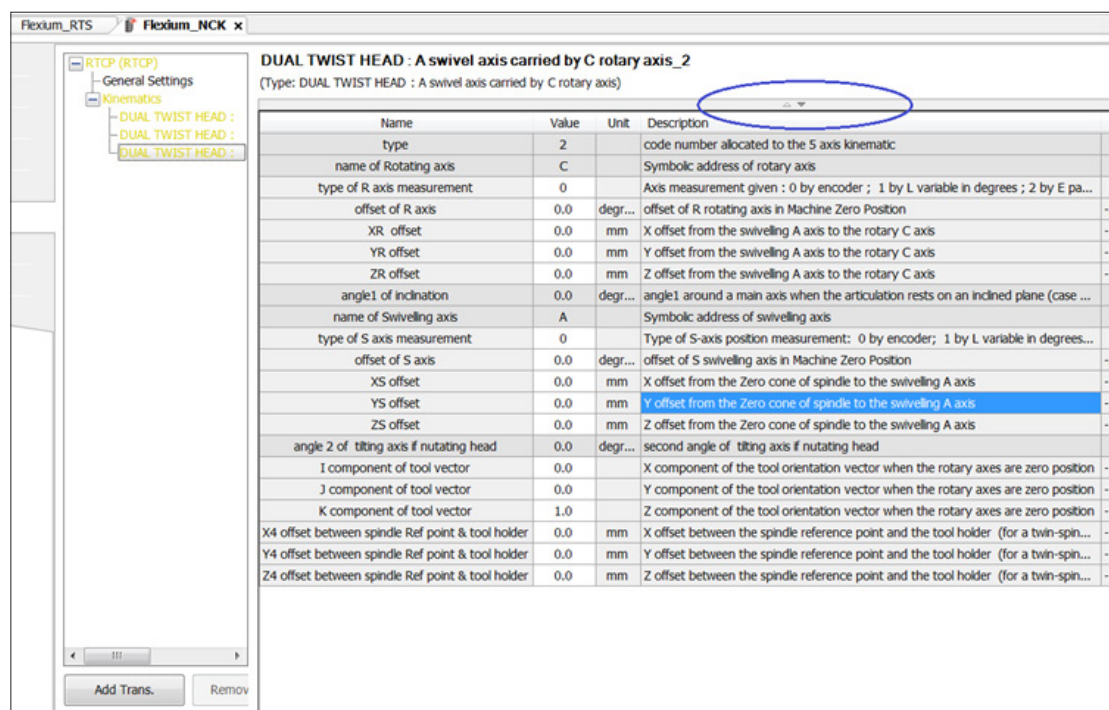
Variable	Explanations
E address for saving context data	Starting address for saving the current context of the RTCP function (25 data). Default value: 80170 meaning the backup of the 25 variables of context begin at the external address E80170. Caution The starting address of E parameter should allow for saving the 25 variables, otherwise an error "Unrecognised E parameter" would be generated if the end address does not exist. If a data segment is specified then only one external parameter is used, the recommended value is 80194.
Mode of datum programming	This variable specifies how DAT2 is applied when an inclined plane is associated with the RTCP function  OM Machine origin  Op Part origin  OP Program origin  OPi Program origin in ICS  Case 0 : DAT1+DAT2 offsets defined before the origin of inclined plane Case 1 : DAT1 defined before the ICS origin and DAT2 defined after the ICS origin 0: DAT2 is applied before the declaration of iPLANE. 1: DAT2 is applied after the declaration of iPLANE. Default value: 0
M code for RTCP enable status	Number of M function sent to the PLC by RTCP function when RTCP is enable Default value: 252.
M code for RTCP disable status	Number of M function sent to the PLC by RTCP function when RTCP is disable Default value: 253.
Customer program	Number of a user program called by RTCP function just before enabling. A null value stands no file. Default value: 0 Caution If this macro file doesn't exist, error 25 is generated in the inclined plane definition block.

Variable	Explanations
File of calibrating data	Optional calibration of 5 axis machine:  The RTCP function is able to automatically manage the output results of a kinematic optimization cycle (Kinematics measuring Cycle248 with 3D touch probe). This variable specifies the Cycle248 Data File number containing the 5-axis calibration results (the number of file: %10248.9) Default value: 0 (no file). Caution If this file doesn't exist, error 25 is generated in the subroutine call block.

- Two lift up or down arrows ▲ ▼ can be used respectively to extend or to reduce the editing area.
- Acting on the bottom arrow ▼ shows the drawing of the kinematics at the expense of the editing area.



- The up arrow ▲ completely restores the edition area of the selected kinematics parameters.



5.14 Kinematic measuring cycle248 with 3D touch probe

5.14.1 General

In Five-axes machining, the FlexiumPro uses kinematic transformations to give the positioning accuracy and the desired orientation of the tool center along the tool path.

The kinematic model, involved in this transformation, is described by two ideal pivot-axes (two revolute joints) and two offset vectors to locate it.

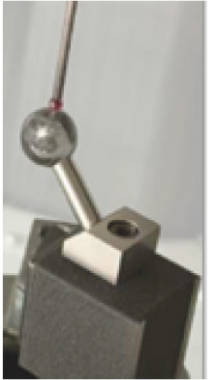
It is therefore necessary to identify precisely the location of pivot points of these rotary axes.

The kinematics measuring cycle248 enables the measurements⁽¹⁾ of spatial deviations for indexing positions of rotary axes involved in the 5 axis transformation and optimizes the component values of geometric offset vectors. The corrections of pivot point parameters are automatically saved in the Flexium controller.

The benefits are:

- An improvement rotary axis performance by eliminating the positioning and orientation errors coming from the 5 axis transformation.
- A cost effective application for checking the precision of the rotary axes.

This cycle requires a calibrated 3D touch probe and a single calibration sphere mounted on the work-piece table.

	
A calibration sphere, clamped on the work-piece table, is used as a reference feature for measurements.	A calibrated 3D touch probe mounted in spindle.

The cycle measures the characteristics of the sphere by probing its surface for various angular positions of the rotary axes.

After the measuring step, the cycle computes the corrections of kinematic offset components.

- (1) The cycle248 is an indirect calibration method using artefacts to estimate the geometric errors and assuming these errors are mainly due to an inaccurate modeling of TCP control and assuming the main axes of the machine are previously calibrated. The error components are not directly measured but computed through inverse kinematics model.

Page deliberately empty.

Page deliberately empty.

Summary

6	FlexiumPro Tools - Instruments	185
6.1	General	185
6.2	Drive Test points	186
6.3	Oscilloscope	187
6.3.1	Stimulus	187

1

2

3

4

5

6

7

A

Page deliberately empty.

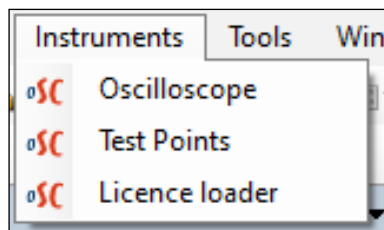
6 FlexiumPro Tools - Instruments

6.1 General

FlexiumPro Tools provides a set of instruments for adjusting the machine.

The different instruments are:

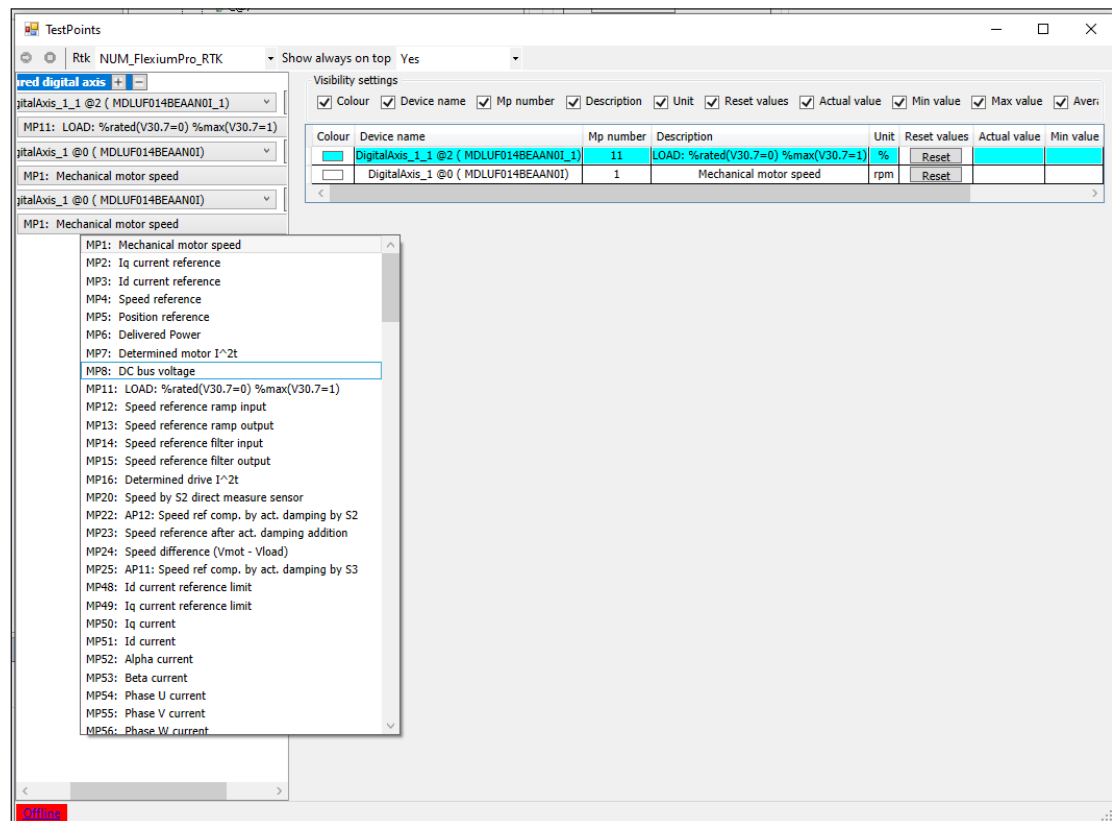
- *Test points*: Display of drive test points
- *Oscilloscope*: Digital oscilloscope for axes
- *License loader*: Storing of the license file (refer to chapter 3.2)



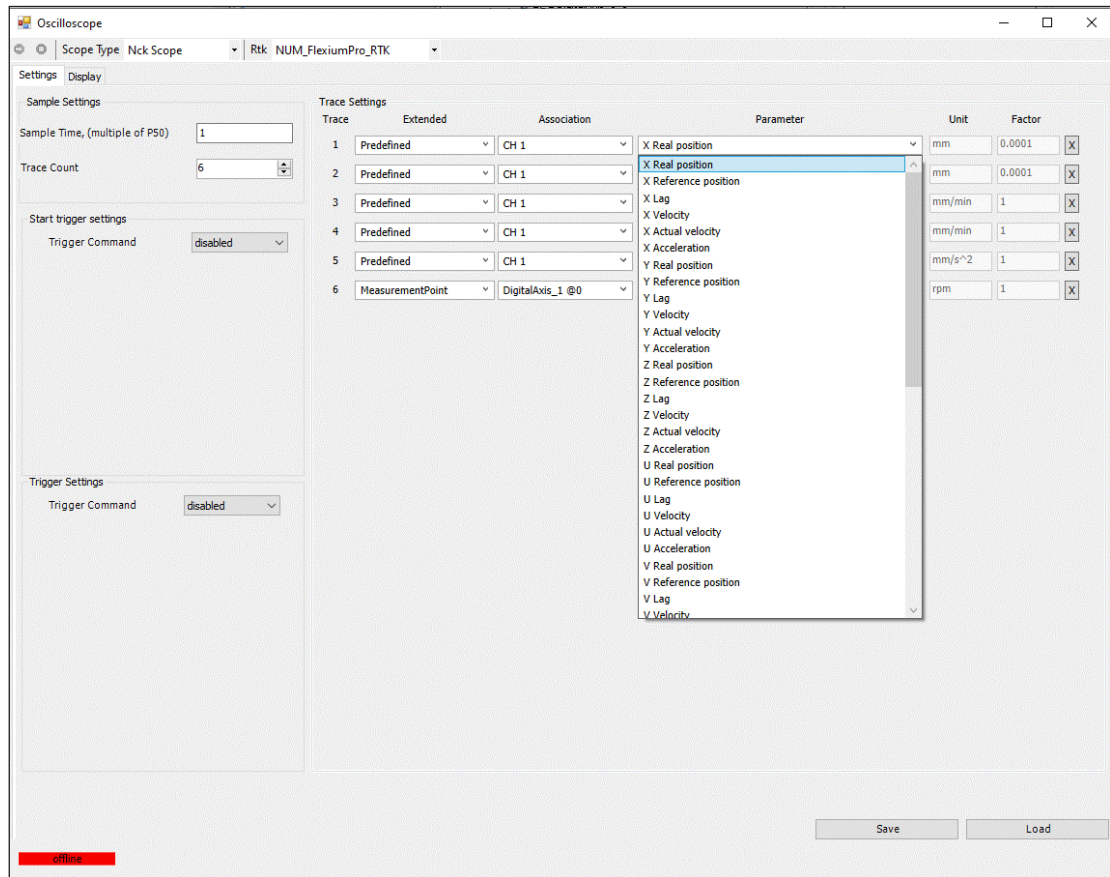
6.2 Drive Test points

This instrument opens a window in which it is possible to select several test points (the performance of the tools downgrade depending by test points displayed) tests points.

According to the answer the system will start with the previous values or the new values defined.

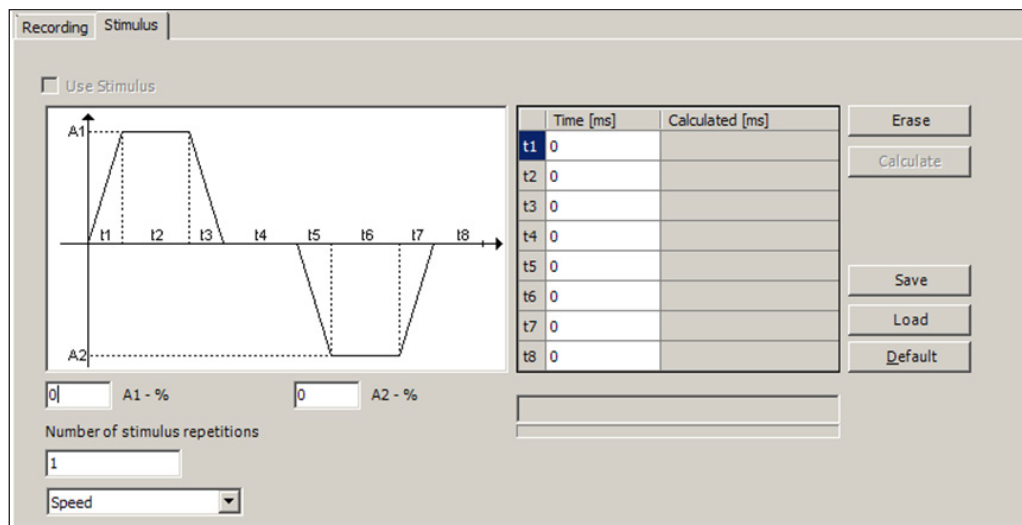


6.3 Oscilloscope



6.3.1 Stimulus

The stimulus shape can be defined by the following screen. The amplitude is defined as a percentage of the max speed. As the other settings it can be saved or restored.



Page deliberately empty.

Summary

7	Machine parameters setup	191
7.1	Introduction	191
7.2	General configuration	191
7.2.1	Data tables.....	191
7.2.2	Related parameters	192
7.3	Axes and Spindle declaration	193
7.3.1	Data tables.....	193
7.3.1.1	Channels definition.....	193
7.3.1.2	Axis and Spindle per channel	193
7.3.1.3	Axes and Spindles definition in each channel	194
7.3.1.4	Handwheels definition	194
7.3.2	Related parameters	195
7.4	Axis and Spindle characteristics	196
7.4.1	Data tables.....	196
7.4.1.1	Axis data tables.....	196
7.4.1.2	Spindle data tables.....	197
7.4.2	Related parameters	198
7.5	Axis and Spindle dynamic	199
7.5.1	Related parameters	199
7.6	NCK controlled analog axes	200
7.6.1	Related parameters	200

Page deliberately empty.

7 Machine parameters setup

7.1 Introduction

As seen from chapter 3.6 on the machine parameters can be set via Advanced Graphic Editors.

However FlexiumPro Tools also provides a text editor for all machine parameters that can be used in conjunction of the graphic editor.

The following chapter groups the parameters by family in order to gather the information required for the initial settings.

At the beginning of each sub-chapter, a table is displayed, for entering the values describing the machine.

These tables facilitate the information gathering process. Detailed information about machine parameter is given in the FlexiumPro Parameters manual M00068.

7.2 General configuration

7.2.1 Data tables

Channel No.	1	2	3	4	5	6	7	8
CNC PLC Settings								
Number of channels								
Sampling period								
Block execution time								
Memory zones								
Memory segments								
Stack size								
Look ahead								
Miscellaneous functions								
Subroutine call by M codes								
Max M codes with Ack								
Overrides values								
Backup – Recall								
Misc settings								

7.2.2 Related parameters

Parameter	Definition
P7	Miscellaneous functions
P35	Subroutine call by M function
P37	Maximum number of M functions with acknowledge
P50	Sampling period
P51	Minimum block execution time
P52	Radial error tolerated on circles - Path checks in RTCP - Circles programming tolerance
P54	Number of blocks in look ahead buffer
P58	Number of E81000/E82000 parameters and program stack reservation
P59	Memory segments of the symbolic variables
P67	Potentiometer min/max range
P95	Memory zones allocation
P97	Number of channels
P98	Number of origins DAT2
P99	CNC - PLC communication setting
P114	Backup on path, Auto recall after INTERV
P115	Final distance at work rate for recall

7.3 Axes and Spindle declaration

7.3.1 Data tables

7.3.1.1 Channels definition

Channel No.	1	2	3	4	5	6	7	8
CNC Channel								
Auxiliary channel								
Milling channel								
Lathe channel								
Main front turret								
Main rear turret								
Secondary turret								

7.3.1.2 Axis and Spindle per channel

	Main axes			Secondary axes			Rotary axes			Spindles			
Name	X	Y	Z	U	V	W	A	B	C	Main	Aux1	Aux2	Aux3
Channel 1													
Channel 2													
Channel 3													
Channel 4													
Channel 5													
Channel 6													
Channel 7													
Channel 8													



The total Axes + Spindles must not exceed 32.
Switchable units (axis<-> Spindles counts only as 1.

7.3.1.3 Axes and Spindles definition in each channel

	Main linear			Secondary linear			Rotary			Spindles			
Name	X	Y	Z	U	V	W	A	B	C	M	X1	X2	X3
Physical address (@ 0-31)													
Logical address													
Measured													
Displayed													
Interpolated													
Indexed by CNC													
Diameter programmed													
With clamps													
Modulo													
Axis / Spindle Switchable													
Rotary Limited travel													
With Quantified Movement													
Carrier / Carried													
Coupled (slave)													
Analog via PLC													
Analog NCK connected													



Remarks for Spindle.

M : Main spindle

X1, X2, X3 : Auxiliary spindle 1, 2 or 3.

7.3.1.4 Handwheels definition

	Handwheel 1			Handwheel 2			Handwheel 3			Handwheel 4			
Presence													
NCK Connected													
PLC Connected													



The total of NCK connected axes, spindles and handwheels cannot exceed 4.

7.3.2 Related parameters

Parameter	Definition
P0	Displayed axes
P1	Modulo or limited excursion rotary axes
P4	Axes programmable by diameter - Internal measurement - Dimension format
P5	Channel configuration
P6	Spindle definition and assignment to channels
P8	Axes with clamps
P9	Axis assignment to a channel
P14	Handwheel declaration
P27	Coupling assignment for duplicated and synchronized axes
P28	Enable slave axis synchronization
P64	Carrier or carried axes
P65	Axes with quantified movements
P71	Axis address associated to the analog port connected spindle
P85	Drives coupling assignment for anti-backlash function
P86	Drives coupling assignment for torque duplication function
P87	Drives coupling assignment for winding duplication function
P116	Multi Level Electronic Gear Box function (MLEGB)
P120	Machine configuration
P121	Network Topology configuration

7.4 Axis and Spindle characteristics

7.4.1 Data tables

7.4.1.1 Axis data tables

	Main axes			Secondary axes			Rotary axes		
Name	X	Y	Z	U	V	W	A	B	C
Address (@ 0-31)									
Acceleration profile									
Maximum speed									
Jog speed slow									
Jog speed normal									
Jog speed high									
Reduced speed 1									
Reduced speed 2									
Reduced speed 3									
Reduced speed 4									
Acceleration work									
Acceleration rapid									
Approach speed									
Override ranges									
Homing direction									
Home position									
Max over-travel									
Min over-travel									
Backlash									
In-position window									
Max following error									

7.4.1.2 Spindle data tables

	Spindles			
Name	M	X1	X2	X3
Physical address (@ 0-31)				
Resolution				
Minimum speed M40				
Minimum speed M41				
Minimum speed M42				
Minimum speed M43				
Minimum speed M44				
Minimum speed M45				
Maximum speed M40				
Maximum speed M41				
Maximum speed M42				
Maximum speed M43				
Maximum speed M44				
Maximum speed M45				
Speed at max RPM M40				
Speed at max RPM M41				
Speed at max RPM M42				
Speed at max RPM M43				
Speed at max RPM M44				
Speed at max RPM M45				

7.4.2 Related parameters

Parameter	Definition
P13	Handwheel measurement conversion coefficient
P15	Homing direction
P16	Machine home position
P17	Axis travel limits
P18	Backlash error compensation
P22	In-position window
P23	Maximum following error
P30	Maximum axis traverse rates
P31	JOG and reduced speeds
P32	Maximum permissible acceleration
P33	Approach speed
P34	Reduced speed
P40	Number of sensor spindle pulses by revolution
P42	Spindle origins
P43	Spindle indexing speed and speed limit by PLC
P44	Spindle indexing In-position window
P46	Speed range for spindles @0 ... @7
P47	Speed range for spindles @8 ... @15
P48	Speed range for spindles @16 ... @23
P49	Speed range for spindles @24 ... @31
P53	Acceleration profile

7.5 Axis and Spindle dynamic

7.5.1 Related parameters

Parameter	Definition
P19	High Speed Cutting: feed-forward acceleration time constant - dry friction
P20	Dry friction correction amplitude
P24	Synchronized axes correction coefficients
P25	Synchronized axes thresholds
P45	Spindle indexing gain
P55	Speed anticipation coefficient - High Speed Cutting: number of filter term
P56	Channel specific time constant
P57	Dynamic lag control
P62	Rigid tapping setting
P63	Servo adjustment for rigid tapping
P66	Time constant for axis

7.6 NCK controlled analog axes

7.6.1 Related parameters

Parameter	Definition
P11	Measurement conversion coefficient for analog axes
P70	Analog ports configuration
P71	Axis address associated to the analog port connected spindle
P72	Direction of axis speed reference for analog axes
P73	Servo System loop coefficient for analog axes

Page deliberately empty.

Page deliberately empty.

Summary

A	Appendix	205
A.1	Axis calibration	205
A.1.1	General	205
A.1.2	Minimum and maximum travel points not corrected	206
A.1.3	Minimum and maximum travel points corrected	206
A.2	Inter-axis calibration	207
A.2.1	Description.....	207
A.2.1.1	General	207
A.2.1.2	Special case of correction in a single point	207
A.2.2	Examples of Inter-axis Compensation Calibration	208
A.2.2.1	Correction of a perpendicular alignment fault	208
A.2.2.2	Correction of spindle flexion	208
A.2.2.3	Correction of sag of the cross piece of a gantry machine	209
A.2.2.4	Correction of an axis expansion	209
A.2.3	Tools for Inter-axis Compensation Calibration	209
A.2.4	Records of the Corrections on the Axes	210
A.3	Dynamic Inter-axis Calibration	211
A.3.1	Addresses of the Correction Parameters	211
A.3.2	Correction Tables	211
A.3.3	Writing and Enabling the Correction Tables	213
A.3.3.1	Conditions for Writing Parameters E81xxx and E82xxx	213
A.3.3.2	Conditions for Writing Parameters E940xx	213
A.3.3.3	Procedure for Writing and Enabling the Correction Tables	213
A.4	Example of MP04 programming	214
A.5	Example of MP06 / MP07 programming	216
A.6	Kinematic structures managed by the RTCP and iPLANE functions.....	220
A.7	Summary of the reference postures of five axis kinematics.....	221
A.8	E80x parameters used by RTCP and by iPLANE functions.....	229
A.9	Time charts.....	230
A.9.1	General	230
A.9.2	Order of execution of S, T and M functions	231
A.9.3	Details on M functions	232
A.9.4	Details on E parameters	235
A.9.5	Axes clamp	236
A.9.5.1	Management by .WCNC.Channel[x].EndExtMovement.....	236
A.9.5.2	Management by Axis.Stop	237
A.9.6	Spindle indexing	238
A.9.6.1	Indexing at constant deceleration	238
A.9.6.2	Indexing at constant deceleration not chosen	239

1

2

3

4

5

6

7

A

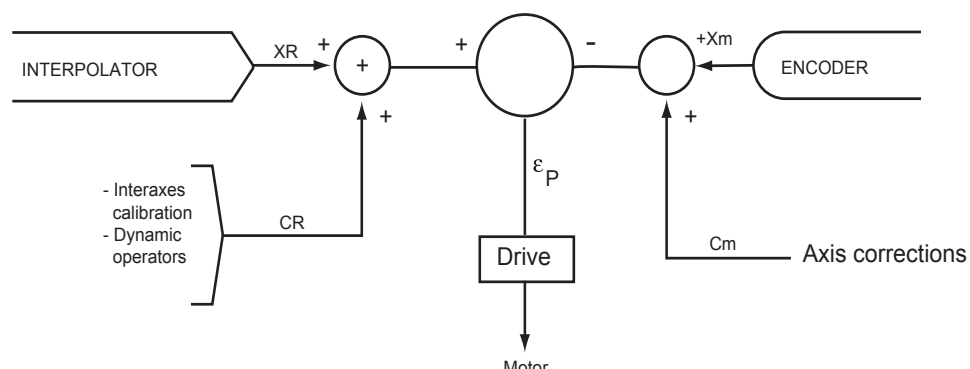
Page deliberately empty.

A Appendix

A.1 Axis calibration

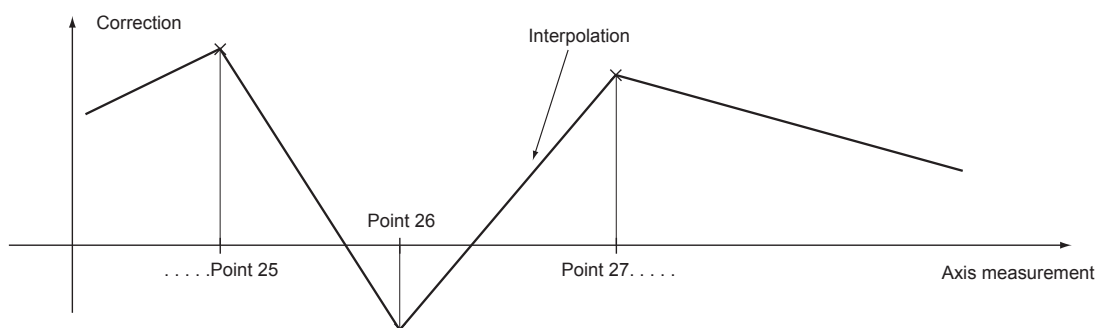
A.1.1 General

Axis calibration allows the system to add a correction to the measured axis position based on the real axis position.



The axis calibration is carried out on both linear and rotary axes. The corrections are entered for a limited number of points per axis.

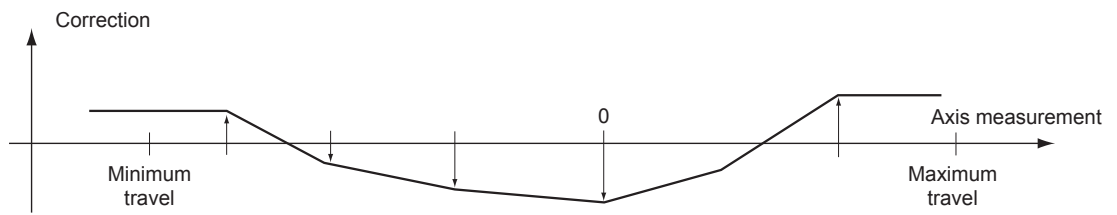
The system computes the corrections between two points by linear interpolation.



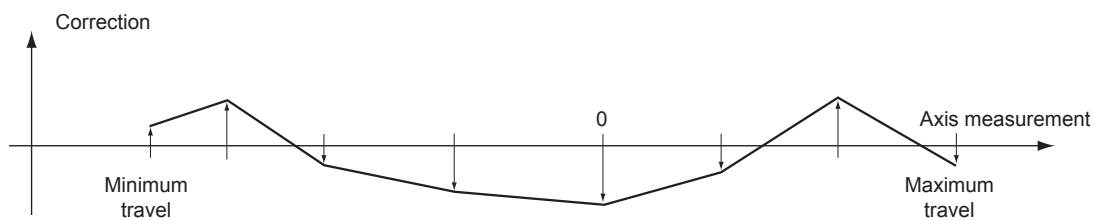
It is recommended to correct the measurements of the minimum and maximum travel points (defined by machine parameter P17). Otherwise, the value of the last correction is applied to these points.

For more detailed information on P17 please refer to FlexiumPro Parameters manual M00068.

A.1.2 Minimum and maximum travel points not corrected



A.1.3 Minimum and maximum travel points corrected



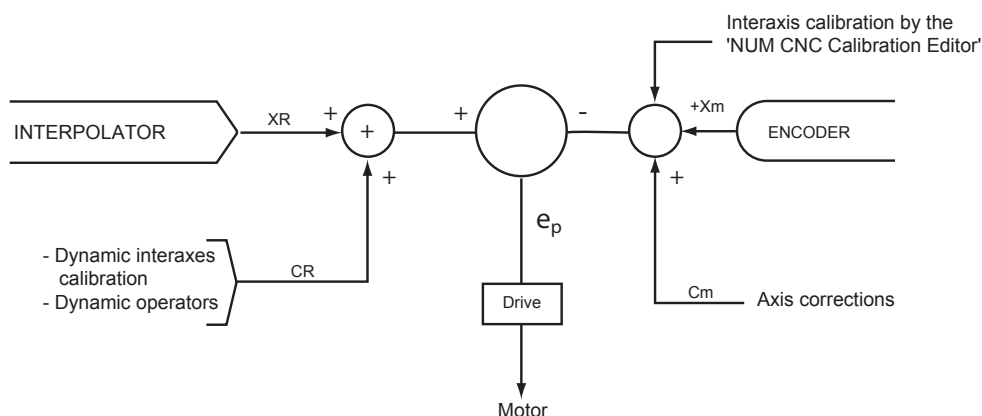
There are no restrictions on how the points are distributed between axes.

A.2 Inter-axis calibration

A.2.1 Description

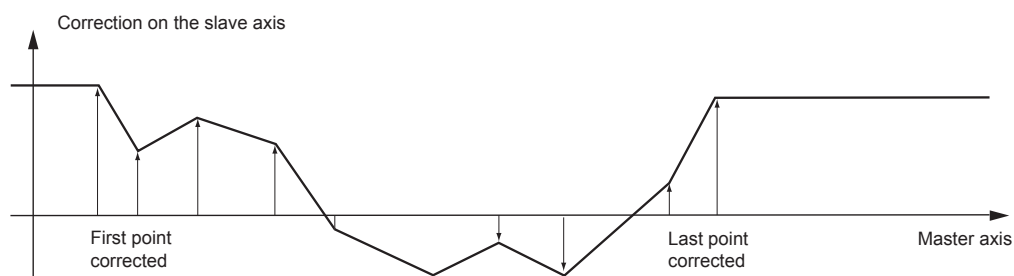
A.2.1.1 General

Inter-axis calibration allows the system to add a correction to the reference of a slave axis (reference created by the interpolators) from the position reference of a master axis.

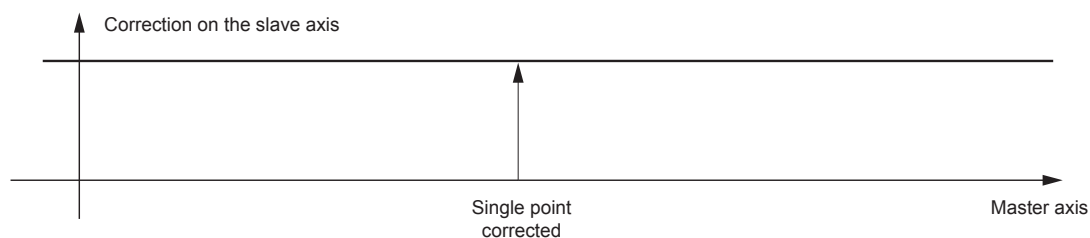


The corrections concern both linear and rotary axes.

They are entered for a limited number of points per axis. The system computes the corrections between two points by linear interpolation. Beyond the boundary points, the corrections are constant.



A.2.1.2 Special case of correction in a single point

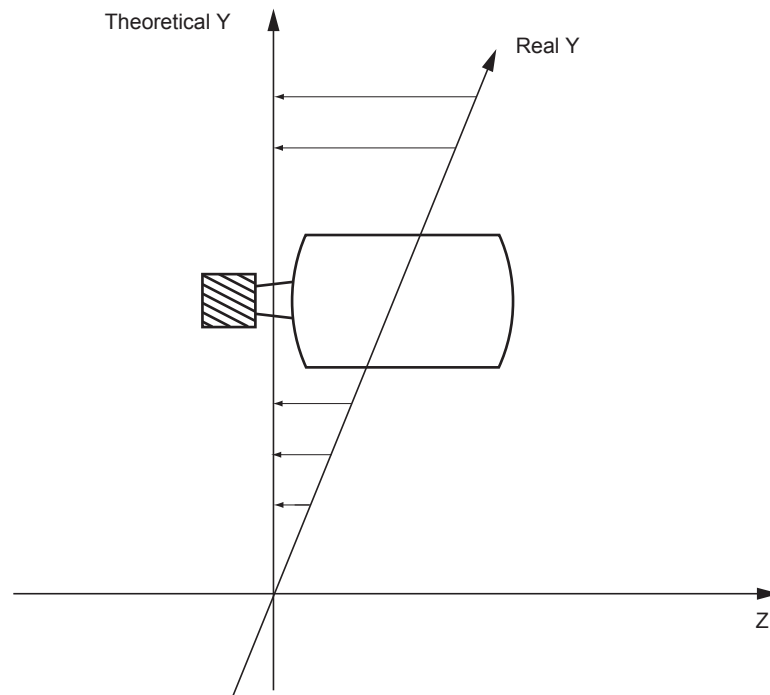


A.2.2 Examples of Inter-axis Compensation Calibration



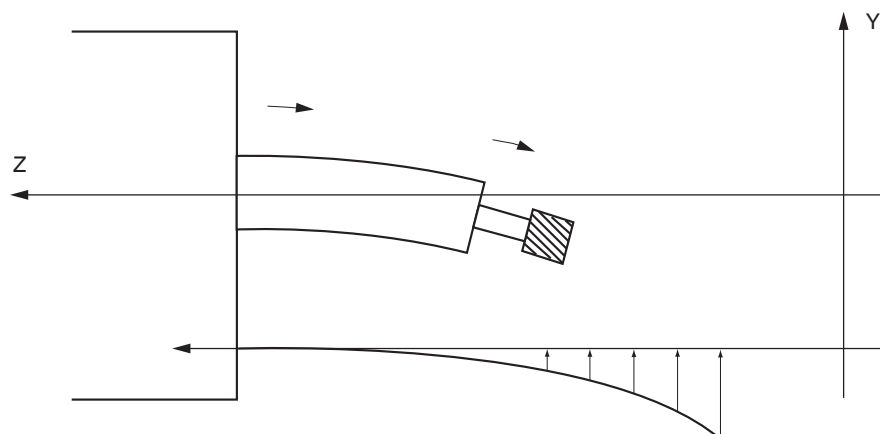
The deformations are exaggerated for clarity in the examples below.

A.2.2.1 Correction of a perpendicular alignment fault



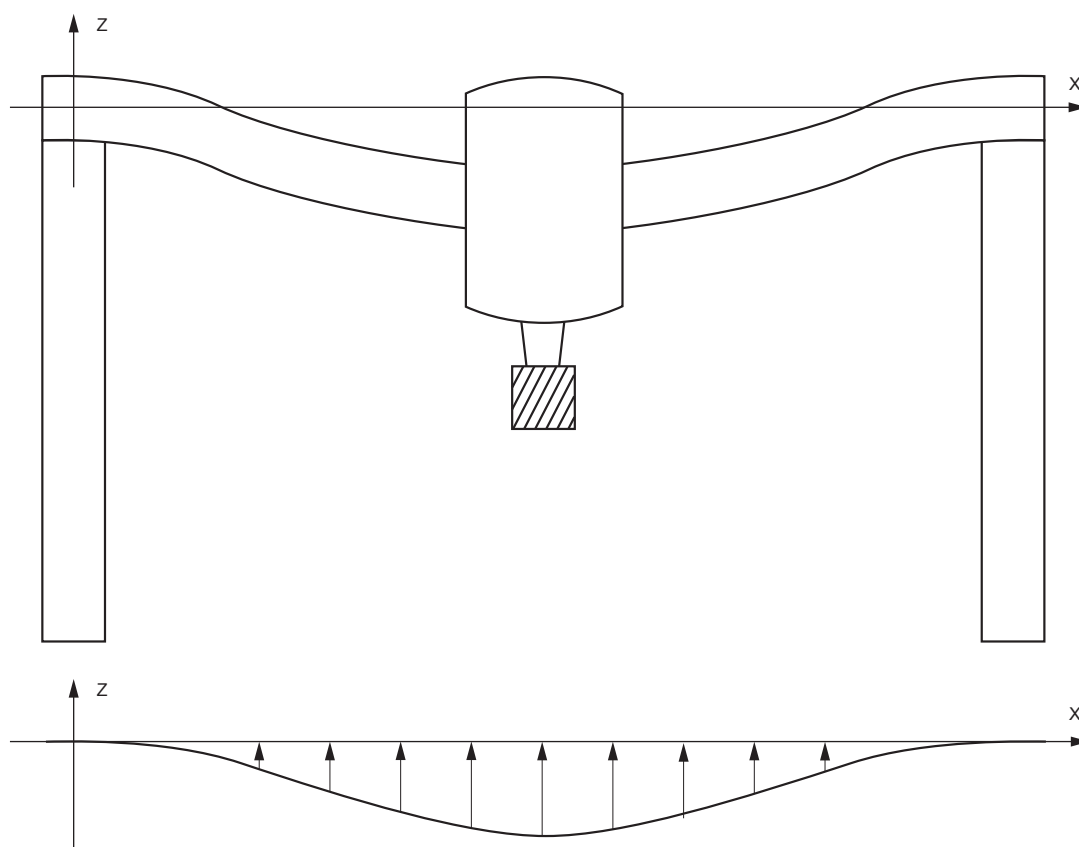
Z is corrected according to the position of the Y axis.

A.2.2.2 Correction of spindle flexion



Y is corrected according to the extension of the spindle

A.2.2.3 Correction of sag of the cross piece of a gantry machine



Sag in Z is corrected according to the position on the X axis.

A.2.2.4 Correction of an axis expansion

An axis can be self-corrected (exclusively for the dynamic calibration: master axis same as slave axis) to take thermal expansions into account.

The correction tables can be written by the PLC program according to the measured temperatures then used by the system. The application of these corrections may prove to be difficult because of the thermal inertia of machines.

A.2.3 Tools for Inter-axis Compensation Calibration

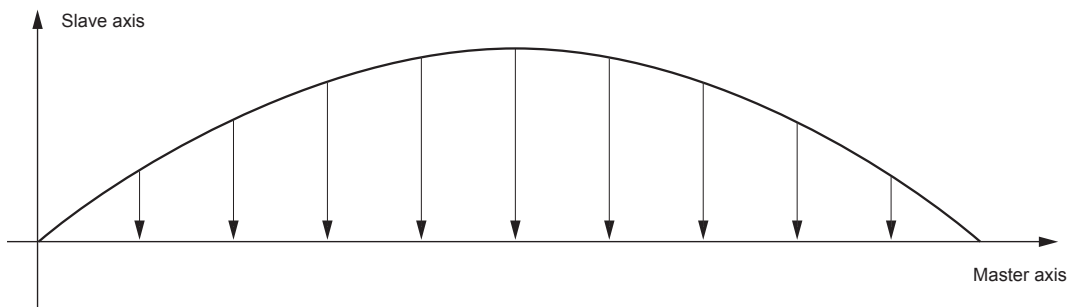
Two tools can be used for inter-axis calibration:

- Calibration by the 'NUM CNC Calibration Editor' (see chapter 3.6.2) to make permanent corrections. These corrections take into consideration the deformations of the machine,
- Dynamic calibration using parameters E81xxx and E82xxx (see annex A.3). They can be modified at any time by the part program. These corrections are suited to variables that change over time such as thermal expansion.

The effects of inter-axis calibration by the 'NUM CNC Calibration Editor' and of the dynamic calibration are added together, which is why it is not recommended to use both methods simultaneously.

A.2.4 Records of the Corrections on the Axes

The corrections to be made to the slave axis are measured for a series of reference positions on the master axis.



The corrections are made in the system internal units (IU).

Master axis No.: Slave axis No.:

Master axis position:							
Slave axis correction:							

The measured values should be transferred to the correction tables.
(see chapter [3.6.2.2](#))



For inter-axis calibration by the '*NUM CNC Calibration Editor*', the maximum corrections are ± 9999 units.
For dynamic inter-axis calibration, the maximum difference between two consecutive corrections is ± 65.000 mm.

A.3 Dynamic Inter-axis Calibration

The possibility of writing correction tables from a part program depends on the setting of machine parameter P7 (Please refer to FlexiumPro Parameters manual M00068):

- *P7 word 0 bit 5 = 0*: write by part program enabled,
- *P7 word 0 bit 5 = 1*: write by part program inhibited.

A.3.1 Addresses of the Correction Parameters

The corrections are made using parameters E81xxx, E82xxx and E940xx:

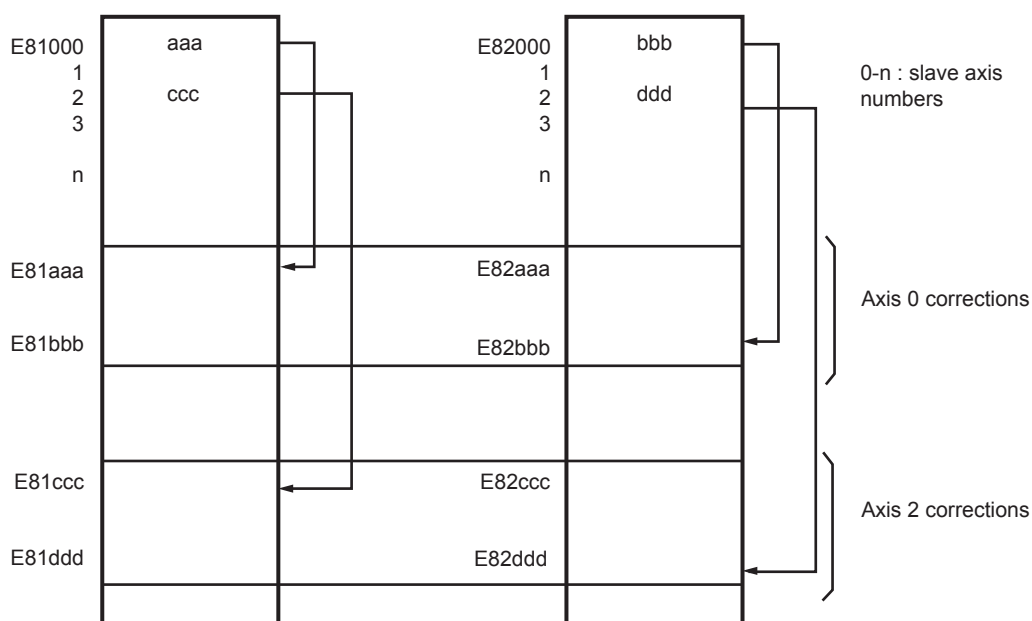
- *parameters E81xxx* address the reference positions of the master axes,
- *parameters E82xxx* address the corresponding corrections on the slave axes,
- *parameters E940xx* assign a master axis to a slave axis.

The current slave axis correction is accessible in read-only parameters E950xx.

A.3.2 Correction Tables

The size of the correction tables (number of E81xxx parameters and E82xxx parameters) is defined by machine parameter P58, word 0 (Please refer to FlexiumPro Parameters manual M00068).

The maximum size of the correction tables is 1000 E81xxx E82xxx parameters. The correction tables can be schematically represented as follows:



The first 32 parameters E81xxx and E82xxx are each assigned to the axis with the equivalent physical address. For instance, E81003 and E82003 are assigned to axis 3. They define the limits of the correction table assigned to that axis.

The subsequent parameters define:

- The reference position of the master axis (parameters E81xxx),
- The corresponding correction to be applied to the slave axis (parameters E82xxx).

The reference positions and slave axis corrections are in system internal units.

For more detailed information on Exxxx please refer to FlexiumPro Programming manual M00073.

A master axis is assigned to a slave axis by $E940xx = yy$ where:

- xx is the slave axis number,
- yy is the corresponding master axis number.

$E940xx = -1$ means that no master axis is assigned to slave axis xx .

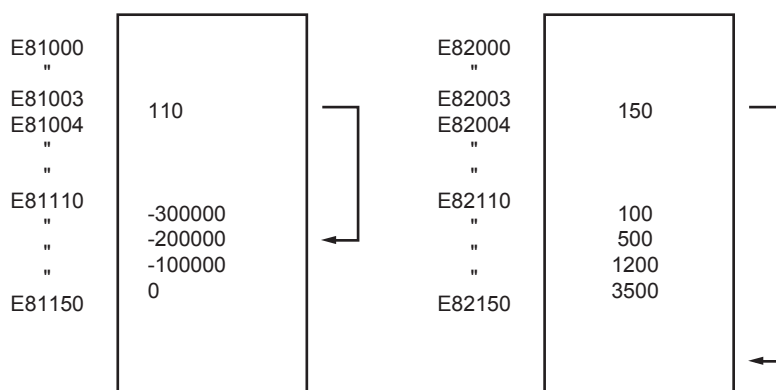


For a corrected axis, the reference positions on the master axis must be defined in increasing order.

Any unused location in the correction table can be used as extra E80xxx parameters (local data written and read by the CNC).

Example

$E94003 = 1$



- $E94003 = 1$ means that slave axis 3 is assigned to master axis 1.
- $E81003 = 110$ and $E82003 = 150$ mean that the parameters defining the corrections of axis 3 are between E81110 and E81150 for the reference positions on master axis 1 and between E82110 and E82150 for the corresponding corrections to be applied to slave axis 3.
- $E81110 = -300000$ and $E82110 = 100$ mean that the first reference position on master axis 1 is located at -300000 IU (i.e. -300 mm if the internal system unit is micrometer) and that the corresponding correction on slave axis 3 is 100 IU. (i.e. 100µm if the internal system unit is micrometer)

A.3.3 Writing and Enabling the Correction Tables

Parameters E81xxx, E82xxx and E940xx can be written by the part program.

A.3.3.1 Conditions for Writing Parameters E81xxx and E82xxx

All E940xx parameters are equal to -1

This means that no correction table is enabled.

All E81xxx and E82xxx parameters can be modified without restriction.

At least one E940xx parameter is different from -1

This means that there is at least one master axis.

All the master axes must be reset (by setting parameters E940xx to -1).



As long as there is a master axis, you must not modify:

- the parameters defining the limits of the correction tables
- the parameters defining the reference positions and the corrections.

A.3.3.2 Conditions for Writing Parameters E940xx

To change the master axis, it is necessary to disable the correction linkage (parameter = -1).

For instance, the following steps are necessary to change from master axis 3 to master axis 1 for slave axis 2:

- E94002 = -1,
- E94002 = 1.

An axis can be slaved to its own reference, e.g. E94002 = 2.

A consistency test is made to the correction table when writing an E940xx parameter (table limits, master axis reference points listed by increasing order). When an inconsistency is detected, the parameter setting is rejected and error 95 is generated.

A.3.3.3 Procedure for Writing and Enabling the Correction Tables

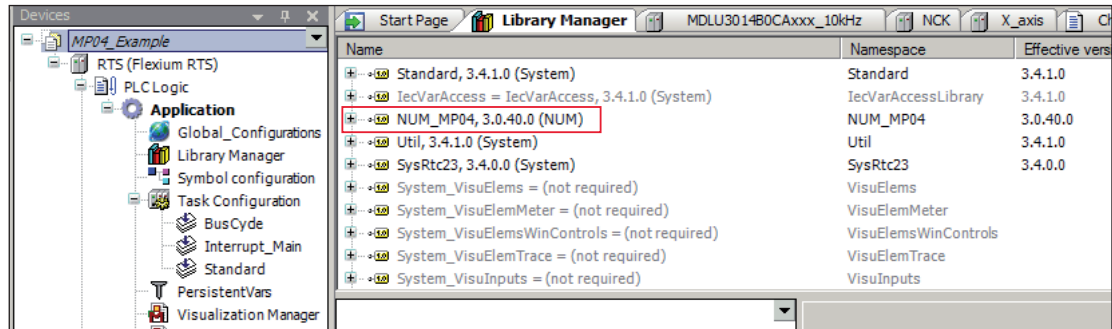
The logical order for writing parameters E81xxx, E82xxx and E940xx is as follows:

- Cancel enabling of all the correction tables: *E940xx = -1*,
- Write the correction tables by modifying parameters *E81xxx and E82xxx*,
- Assign master axes to the slave axes: *E940xx=yy*.

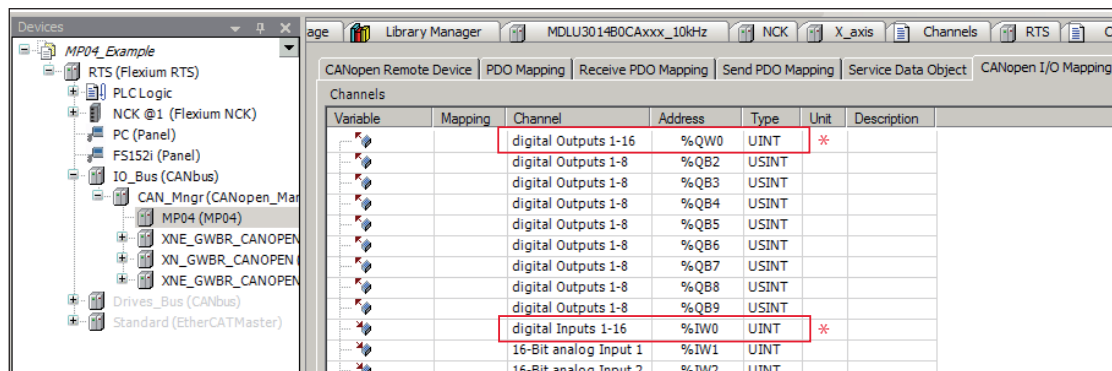
A.4 Example of MP04 programming

- 1 - Add the library NUM_MP04 in your project
- 2 - Add all MP04 machine panel of the machine in the device tree
- 3 - Create an instance of NUM_MP04 for each physical machine panel
- 4 - Set the hardware address of each instance
- 5 - Program using the predefined IOs.

1 Include the Library NUM_MP04 in the Library manager



2 Include an MP04 on the CAN bus in the device tree and check the I/O mapping (first %QW and first %IW)



Make a note of the two values (*) above.

3 Create the Following Global variable file : (named GVL in this example)

```
VAR_GLOBAL  
FX_MP04 : NUM_MP04; // Will bet the MP04 Name  
END_VAR
```

FX_MP04 is an example of name of the first machine panel.

If they are several machine panels on the machine, each one should have a different name.

4 Create the following Global variable file in the device window

VAR_CONFIG		
GVL.FX_MP04.inpcan	AT %IB0	: MP04inpcan;
GVL.FX_MP04.outcan	AT %QB0	: MP04outcan;
END_VAR		



The syntax should be exactly respected with the following remark:

- The prefix GVL. In the second file should be replaced by the name of the first file.
- The value behind %IB and %QB is the number found in the I/O mapping of the MP04.
- When creating a global variable file a line VAR_GLOBAL is automatically inserted, replace GLOBAL by CONFIG in the second file.

5 Examples of programming (Structured text)

NCK.WCNC.General.Start	:= FX_MP04.xBtnStart	// Processing the START button
FX_MP04.xLampStart	:= NCK.RCNC.General.Start	// Activating the start lamp
NCK.WCNC.Spindle[1].Override	:= FX_MP04.bSpeed	// Spindle 1 override

A.5 Example of MP06 / MP07 programming

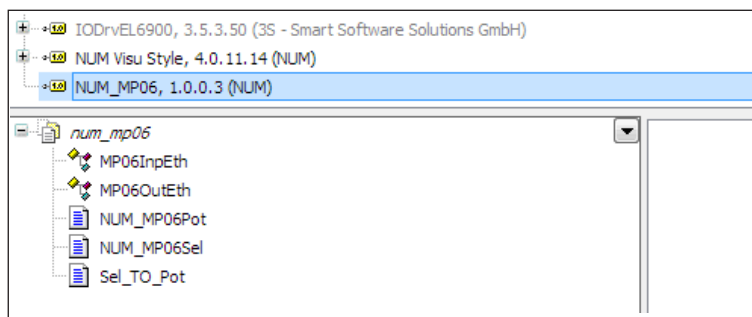
MP06 is a machine panel similar in aspect to the MP04 and connected on the EtherCAT bus. It is available in two versions:

- With Potentiometers for Spindle and Feed override
- With selectors for Spindle and Feed override.

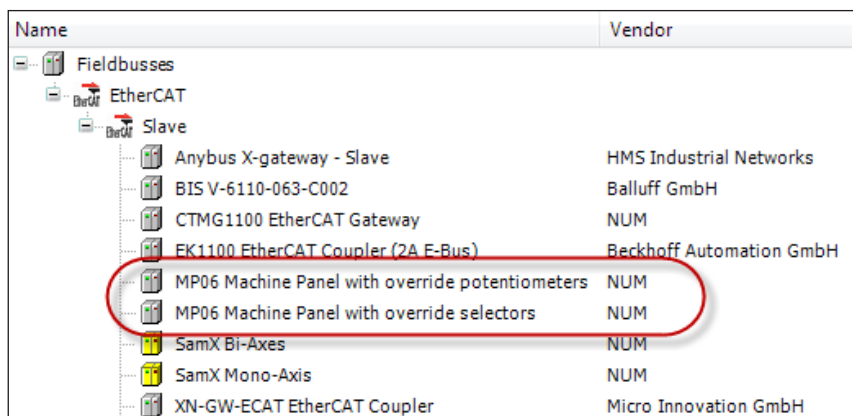
A dedicated library is delivered with three functions:

- *NUM_MP06Pot* to manage an MP06 equipped with standard potentiometers for Speed and Feed override.
- *NUM_MP06Sel* to manage an MP06 equipped with selector switches for Speed and Feed override.
- *Sel_To_Pot* to convert a wished override value on a certain switch position into the value manageable by CNC.

1 Add the NUM_MP06 library to the library manager



2 Add the MP06 device into the project



3 Define the instance of the right function to be used depending from the MP06 hardware in use

```

1  VAR_GLOBAL
2      MY_MP06 :          NUM_MP06Pot;
3      // MY_MP06 :      NUM_MP06Sel;
4      // Select the type of hardware
5      // with Potentiometers or Selectors
6  END_VAR

```

4 Check the I/O addresses and assign it in a configuration file

- The configuration file must be a separate GVL object.
- Please note the `VAR_CONFIG` in the GVL configuration file.
- The GVL module must be defined in the *Devices* view (not in the POU).

Mngt_MP06 MP06 X						
Slave Process Data Startup parameters EtherCAT Configuration EtherCAT I/O Mapping Status Information						
Channels						
Variable	Mapping	Channel	Address	Type	Unit	Description
		Led_A0	%QX500.0	BIT		Led_A0
		Led_A1	%QX500.1	BIT		Led_A1
		Led_A2	%QX500.2	BIT		Led_A2

Mngt_MP06 MP06 X						
Slave Process Data Startup parameters EtherCAT Configuration EtherCAT I/O Mapping Status Information						
Channels						
Variable	Mapping	Channel	Address	Type	Unit	Description
		Btn_A0	%IX500.0	BIT		Btn_A0
		Btn_A1	%IX500.1	BIT		Btn_A1
		Btn_A2	%IX500.2	BIT		Btn_A2

Mngt_MP06 Mngt_MP06_Config X MP06						
{attribute 'qualified_only'}						
VAR_CONFIG						
Mngt_MP06.My_MP06.InpEth	AT	%IB500.0	:	NUM_MP06.MP06InpEth;		
Mngt_MP06.My_MP06.OutEth	AT	%QB500.0	:	NUM_MP06.MP06OutEth;		
END_VAR						

5 RUN the object MY_MP06

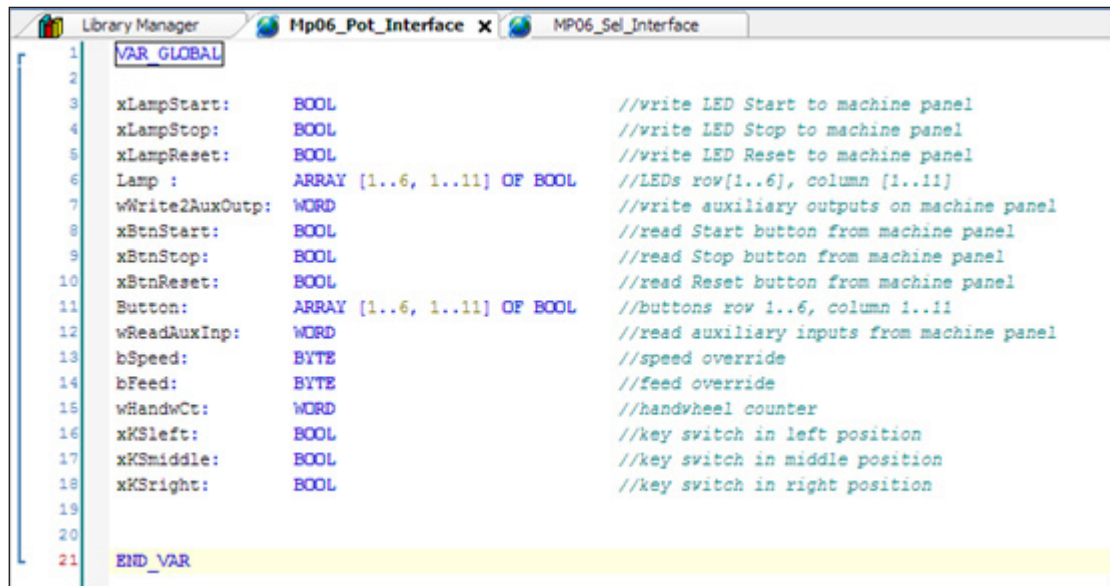
```

66
67
68  // Call an instance of the MP06 object
69  MY_MP06();
70
71
72

```

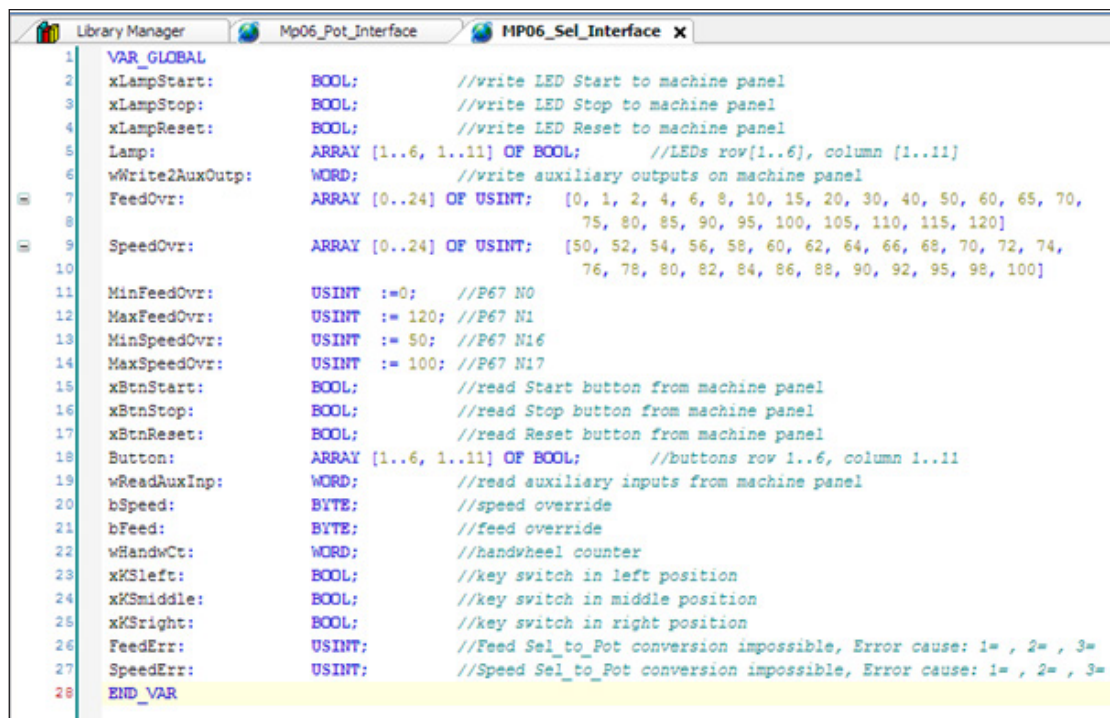
6 Interface

MP06_Pot_interface (version with potentiometers)



They are 12 auxiliary input as well as 12 auxiliary outputs available on the back of the MP06.

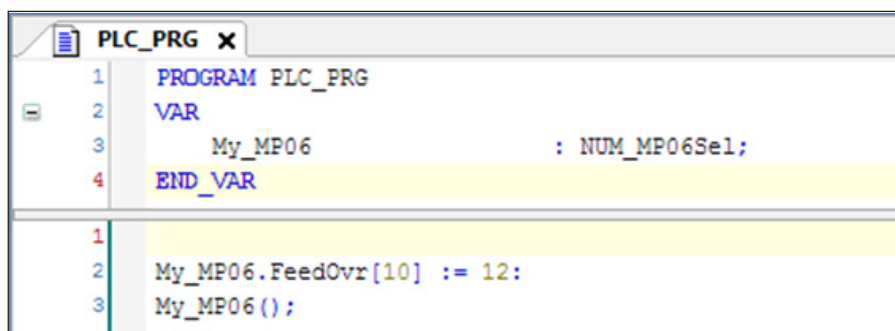
MP06_Sel_interface (version with selector)



This object permits to assign to each selector position an override percentage value by filling the desired data into the dedicated arrays *FeedOvr* or *SpeedOvr*.

Default values are assigned but can be changed by the PLC program.

Example: For position 7 of the FeedOverride we want 12% instead of the default 10



```

1  PROGRAM PLC_PRG
2  VAR
3      My_MP06                : NUM_MP06Sel1;
4  END_VAR

1
2  My_MP06.FeedOvr[10] := 12;
3  My_MP06();

```

To convert the selector values to values acceptable by the NCK interface, it is necessary to fill in the variables *MinFeedOvr*, *MaxFeedOvr*, *MinSpeedOvr*, *MaxSpeedOvr*.

This is performed by the implicit function *Sel_To_Pot*.

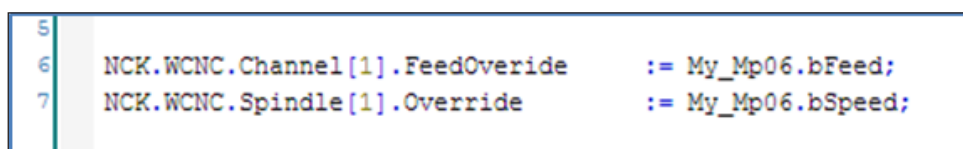
This function doesn't check if the above variables are equal to the CNC parameter value, but some congruency test are done to avoid critical errors.

The test performed are:

- If $\text{Min_Ovr} > \text{Max_Ovr}$ → $_Err = 1$ and $b_ = 0$
- If $_Ovr[x] < \text{Min_Ovr}$ → $_Err = 2$ and $b_ = 0$
- If $_Ovr[x] > \text{Max_Ovr}$ → $_Err = 3$ and $b_ = 0$

"_" must be substituted with Feed or Speed in the above formulas.

The values are then sent to the NCK as usual.



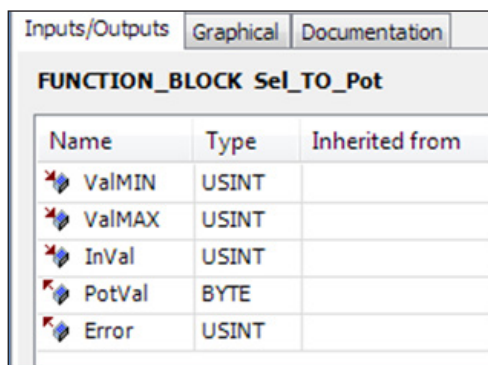
```

5
6  NCK.WCNC.Channel[1].FeedOverride := My_Mp06.bFeed;
7  NCK.WCNC.Spindle[1].Override    := My_Mp06.bSpeed;

```

For the OEM who want to use their own selector it is possible to use the function *Sel_To_Pot* as described above to convert the selector data to a value compatible with the NCK interface

The function *Sel_To_Pot* is available for this:



Name	Type	Inherited from
ValMIN	USINT	
ValMAX	USINT	
InVal	USINT	
PotVal	BYTE	
Error	USINT	

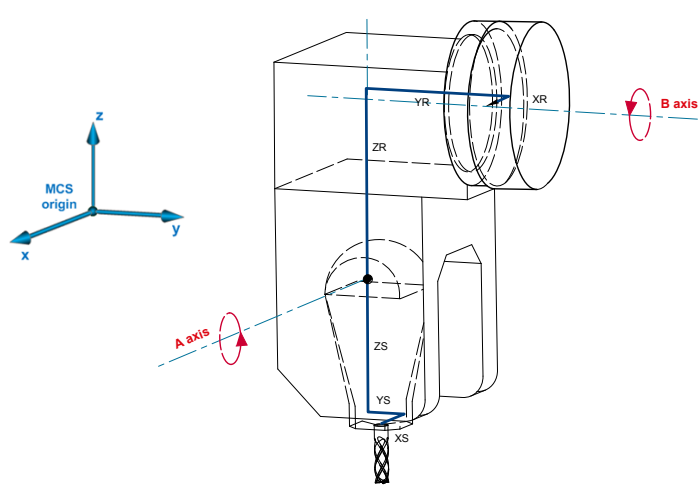
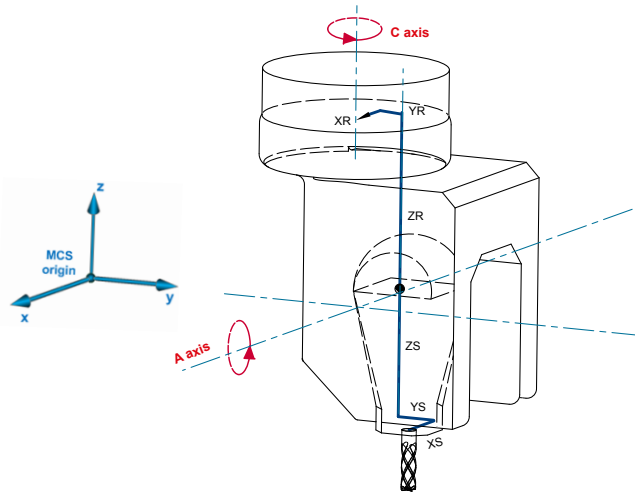
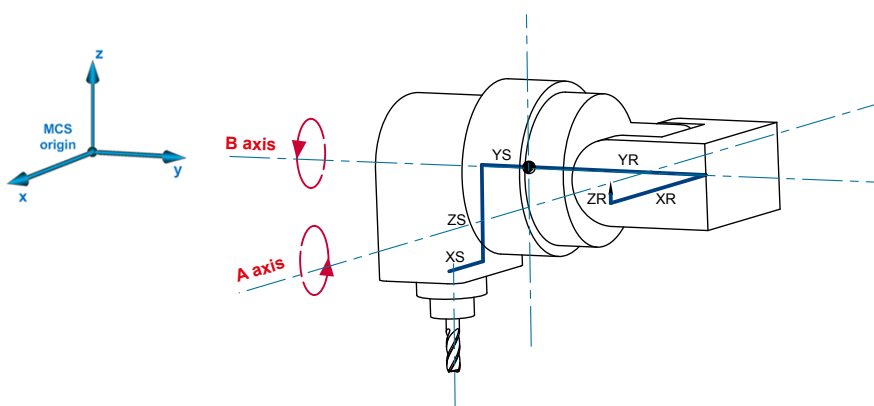
A.6 Kinematic structures managed by the RTCP and iPLANE functions

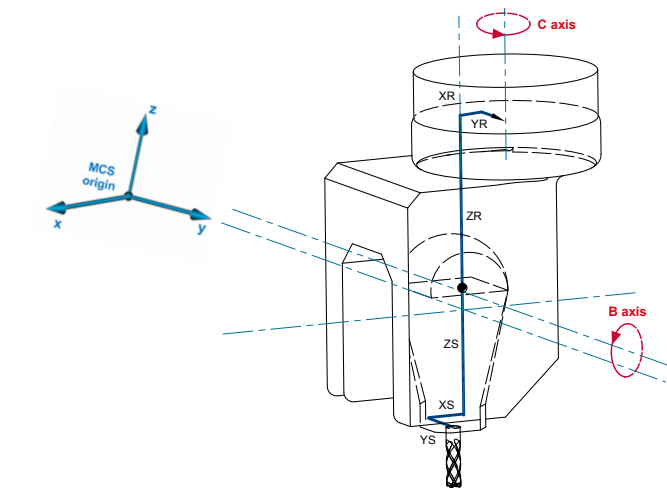
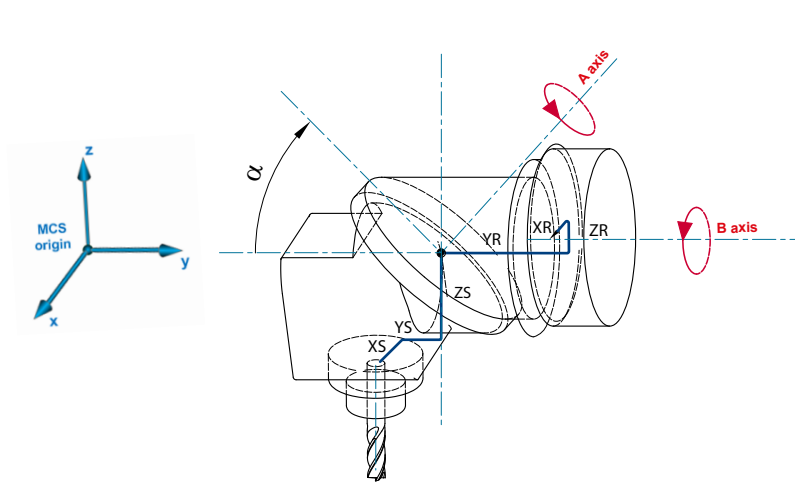
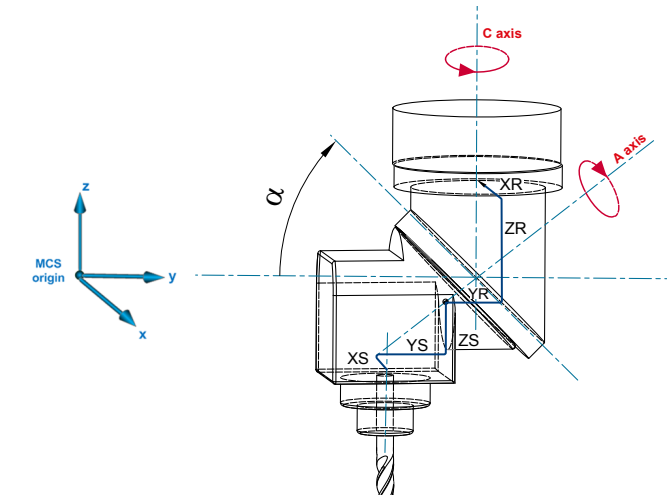
The kinematic configuration and functionality are fully compatible with Fleixum⁺

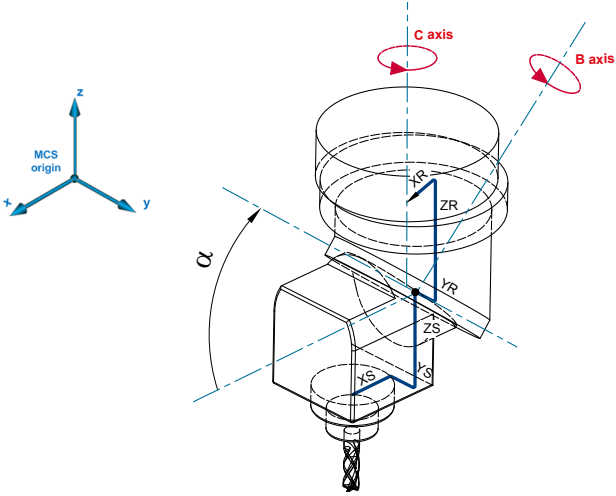
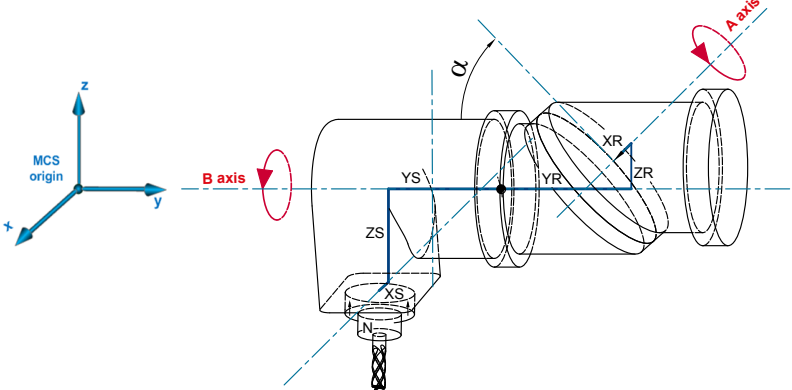
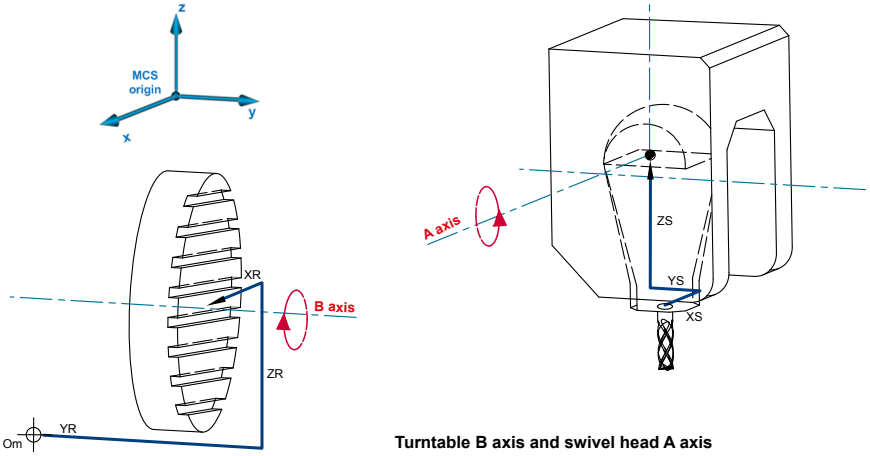
This list is not exhaustive and may evolve with the version of RTCP and iPLANE functions.

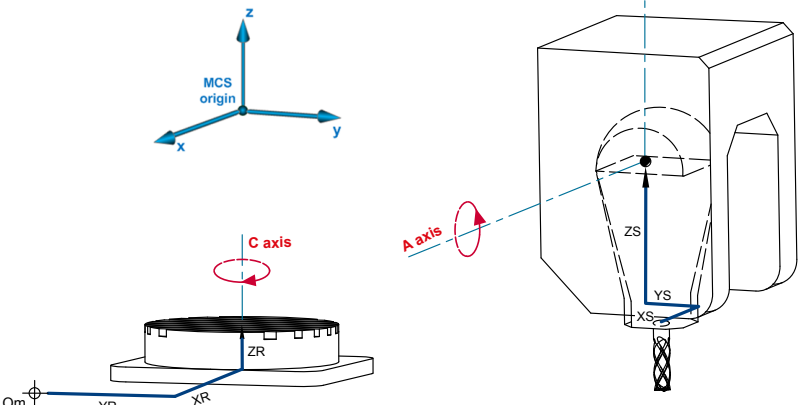
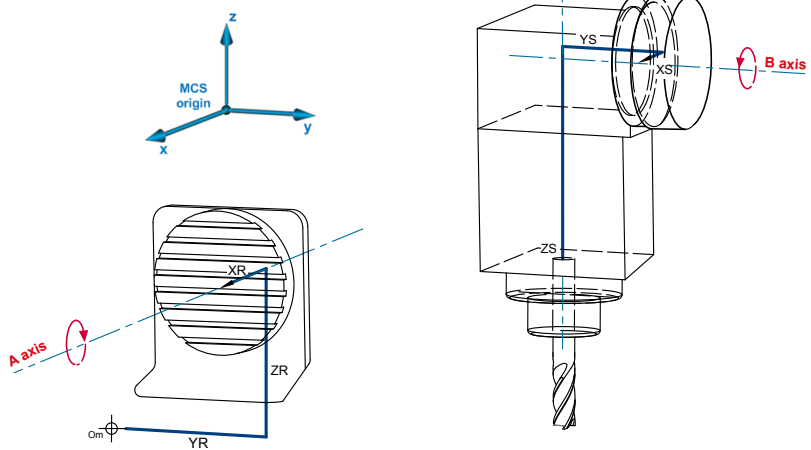
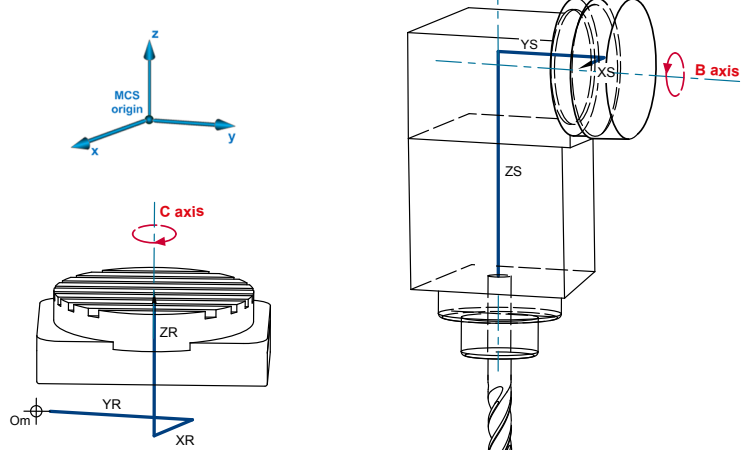
01	DUAL TWIST HEAD A ON B	R = B axis	S = A axis
02	DUAL TWIST HEAD A ON C	R = C axis	S = A axis
03	DUAL TWIST HEAD B ON A	R = A axis	S = B axis
04	DUAL TWIST HEAD B ON C	R = C axis	S = B axis
05	NUTATING ROTARY HEAD TILTED A ON B	R = B axis	S = A axis
06	NUTATING ROTARY HEAD TILTED A ON C	R = C axis	S = A axis
07	NUTATING ROTARY HEAD TILTED B ON C	R = C axis	S = B axis
08	NUTATING ROTARY HEAD B ON TILTED A	R = A axis	S = B axis
09	B ROTARY TABLE & A AXIS HEAD	R = B axis	S = A axis
10	C ROTARY TABLE & A AXIS HEAD	R = C axis	S = A axis
11	A ROTARY TABLE & B AXIS HEAD	R = A axis	S = B axis
12	C ROTARY TABLE & B AXIS HEAD	R = C axis	S = B axis
13	TILTING ROTARY TABLES A ON B	R = A axis	S = B axis
14	TILTING ROTARY TABLES A ON C	R = A axis	S = C axis
15	TILTING ROTARY TABLES B ON A	R = B axis	S = A axis
16	TILTING ROTARY TABLES B ON C	R = B axis	S = C axis
17	TILTING ROTARY TABLES C ON A	R = C axis	S = A axis
18	TILTING ROTARY TABLES C ON B	R = C axis	S = B axis
19	SWIVEL A AXIS HEAD		S = A axis
20	SWIVEL B AXIS HEAD		S = B axis
21	A ROTARY TABLE	R = A axis	
22	B ROTARY TABLE	R = B axis	
23	C ROTARY TABLE	R = C axis	

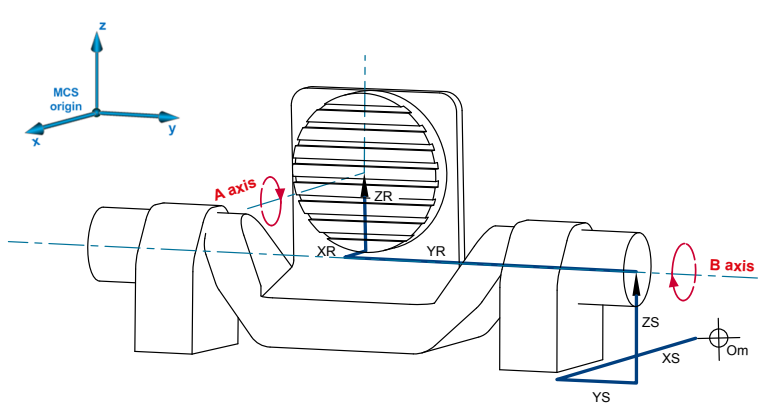
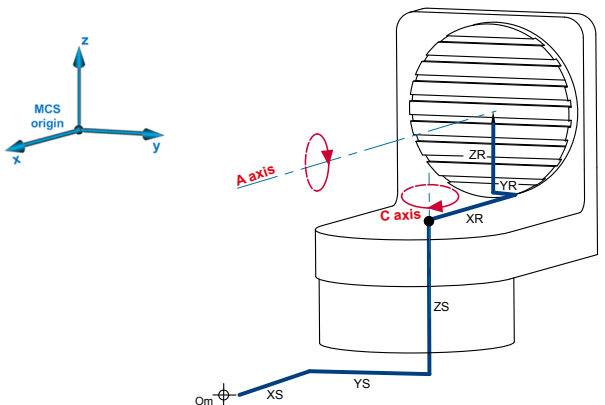
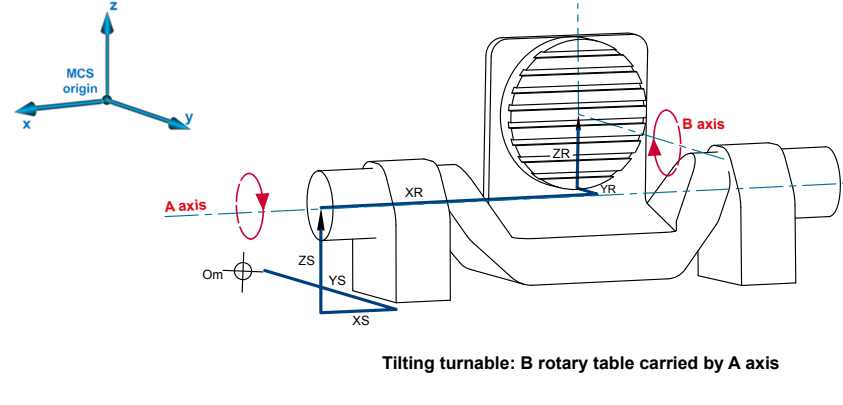
A.7 Summary of the reference postures of five axis kinematics

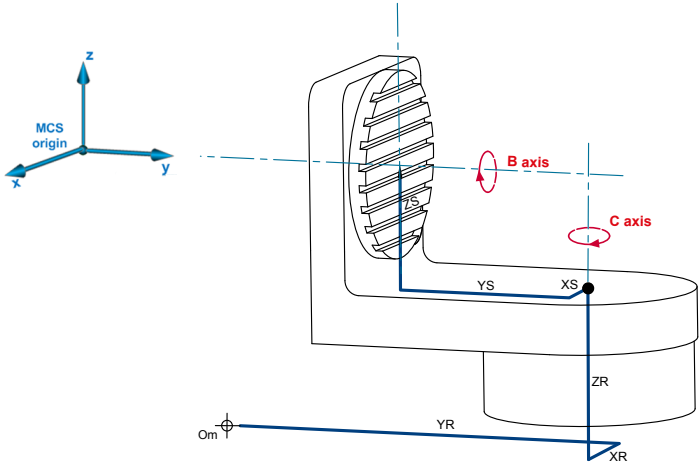
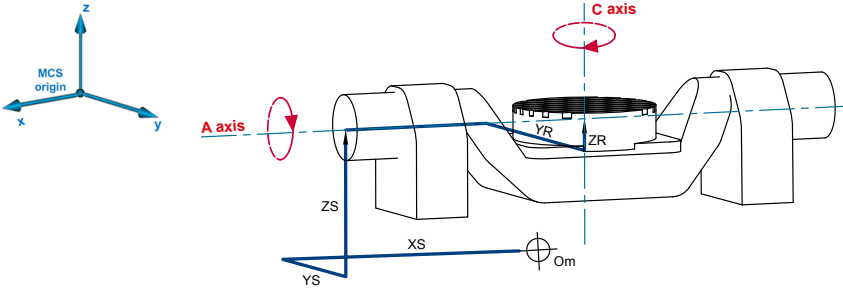
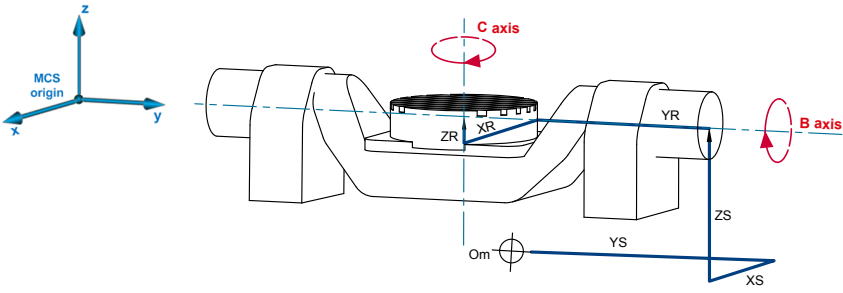
Type of kinematics	Representation of kinematic
Type 1	 Dual twist head: A axis on B axis
Type 2	 Dual twist head: A axis on C axis
Type 3	 Dual twist head: B axis on A axis

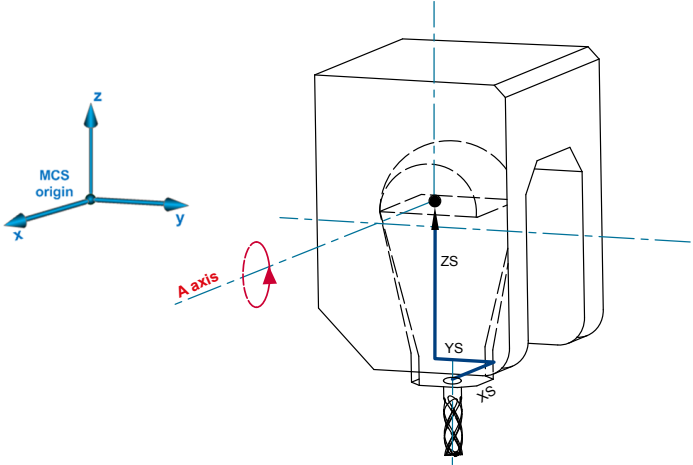
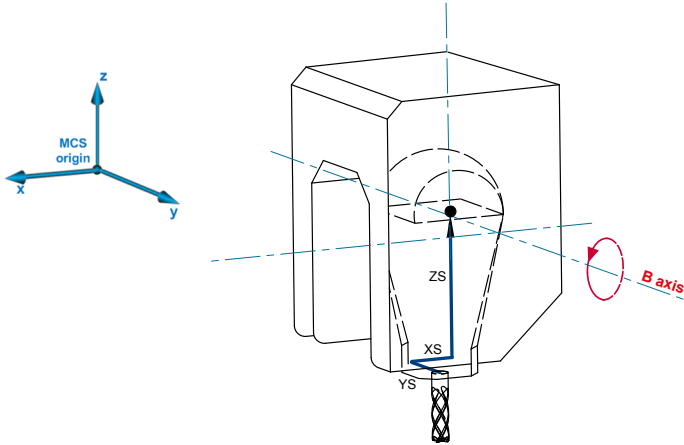
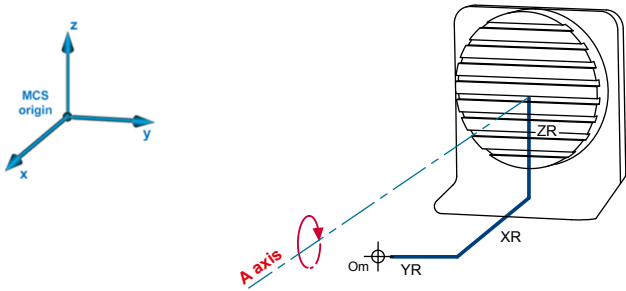
Type of kinematics	Representation of kinematic
Type 4	 Dual twist head: B axis on C axis
Type 5	 Dual twist head: A inclined on B axis
Type 6	 Dual twist head: A inclined axis on C axis

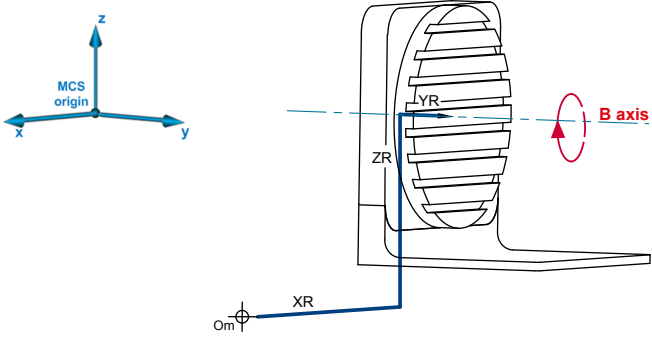
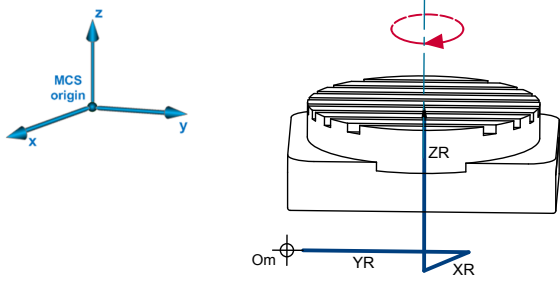
Type of kinematics	Representation of kinematic
Type 7	 Dual twist head: B inclined axis on C axis
Type 8	 Dual twist head: B axis on A inclined axis
Type 9	 Turntable B axis and swivel head A axis

Type of kinematics	Representation of kinematic
Type 10	 Turntable C axis and swivel head A axis
Type 11	 Turntable A axis and swivel head B axis
Type 12	 Turntable C axis and swivel head B axis

Type of kinematics	Representation of kinematic
Type 13	 Tilting turntable: A rotary table carried by B axis
Type 14	 Tilting turntable: A rotary table carried by C axis
Type 15	 Tilting turntable: B rotary table carried by A axis

Type of kinematics	Representation of kinematic
Type 16	 Tilting turntable: B rotary table carried by C axis
Type 17	 Tilting turntable: C rotary table carried on A axis
Type 18	 Tilting turntable: C rotary table carried by B axis

Type of kinematics	Representation of kinematic
Type 19	 The diagram shows a 3D coordinate system with axes x, y, and z. The origin is labeled 'MCS origin'. To the right, a 3D model of a machine head is shown. A vertical dashed line represents the ZS axis. A horizontal dashed line represents the XS axis. A red curved arrow indicates rotation around the A axis, which is parallel to the XS axis. The head is mounted on a vertical column. Simple twist head: A axis
Type 20	 The diagram shows a 3D coordinate system with axes x, y, and z. The origin is labeled 'MCS origin'. To the right, a 3D model of a machine head is shown. A vertical dashed line represents the ZS axis. A horizontal dashed line represents the XS axis. A red curved arrow indicates rotation around the B axis, which is parallel to the ZS axis. The head is mounted on a vertical column. Simple twist head: B axis
Type 21	 The diagram shows a 3D coordinate system with axes x, y, and z. The origin is labeled 'MCS origin'. To the right, a 3D model of a turntable is shown. A horizontal dashed line represents the XR axis. A vertical dashed line represents the ZR axis. A red curved arrow indicates rotation around the A axis, which is parallel to the XR axis. The turntable is mounted on a base with a center point labeled 'Om' and a vertical axis labeled 'YR'. Turntable A axis

Type of kinematics	Representation of kinematic
Type 22	 Turntable B axis
Type 23	 Turntable C axis

A.8 E80x parameters used by RTCP and by iPLANE functions

The RTCP function uses a series of 25 E Parameters for saving the current context of RTCP internal calculated variables.

If no memory segment is allocated, then the backup address begins at E80170 but can be customized by the machine manufacturer.

E parameter	Function	Description
E80170 to E80194 (1)	RTCP	Reserved by the RTCP function in case of an usual customization (see the customization program %10151.9).
E80194 (1)	RTCP	Reserved by the RTCP function for saving the kinematic configuration Tn.
(1) If no memory segment is allocated.		

The iPLANE function uses 4 E parameters. The default addresses are:

E parameter	Function	Description
E80196	iPLANE	Used for saving the kinematic configuration Tn.
E80197	iPLANE	Used to define the coordinate system of the inclined plane function.
E80198	iPLANE	According to the type of the R rotary axis measurement, may contain the position measurement of the rotary axis..
E80199	iPLANE	According to the type of the S swivel axis measurement, may contain the position measurement of the swivel axis.

A.9 Time charts

A.9.1 General

Except otherwise indicated, the following charts have been recorded with the base parameters of a FlexiumPro Tools application:

- Bus_cycle: 10 ms
- P50: 2ms
- Exchange NC <-> PLC: every 3 x P50.

In the following traces the movement refers to the theoretical reference and does not consider the lag.

Speed, movements and other values that are not directly available in the PLC are fetched by dynamic operators and transmitted to the PLC via E37000 parameters.

Due to asynchronism between NCK tasks and PLC task, it's possible that some signal durations may slightly vary. This will be explicitly described when it occurs.

The program used to display the traces will be shown in each case. These programs contain an M80 function which is required to trigger the recording.

A.9.2 Order of execution of S, T and M functions

The trace is recorded with the following part program:

```
%10.0( )
G0 X0 M40 M3 S0 M5 M11 T0
M80
N05 G1 F800 X10 M3 S500
N10 M8 S1000 T5 X20
N15 S1500 X30 M9 M101
N20 S1000 X40
N25 S2000 X50
N30 X60 M300
N35 X70
N40 T2 X80 S100
M2
```

Comments

- T code is emitted first at the beginning of the block, followed by S code and then Forward M functions (1).
- The axes are stopped when those functions are emitted.
- Execution continues before the S code has reached its final value (increasing or decreasing). Management by the PLC is required if the machine requires that the spindle speed is reached before continuation.
- Coded M functions are emitted as a pulse at end of block after all other functions (2).
- On the fly M functions are emitted at the beginning of the block without axis stop (3).



A.9.3 Details on M functions

The trace is recorded with the following part program:

```
%10.0( )  
G0 X0 M40 M9 M3 S0 M5 M11 T0  
M80  
N05 G1 X20 M50  
N10 X30 M8  
N15 X40 M100  
N20 X60 M150  
N25 X80 M50  
N30 M150  
N35 M150  
N40 X100  
M2
```

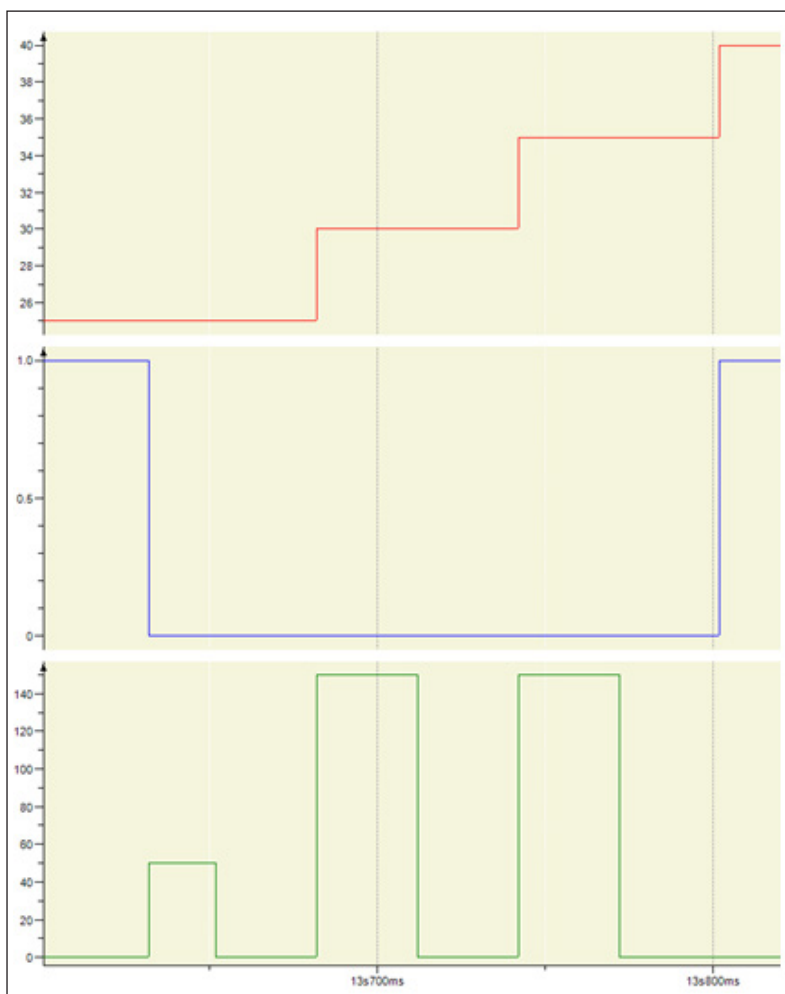
Comments

- Coded M functions are emitted as a pulse, The value is maintained as long as *AckMFunction* is low.
- The code will be present in the PLC for at least two PLC cycles due to the asynchronism of tasks the charts can show 20 or 30ms with the base values.
- Several consecutive M functions (2) are separated by a 0 value during a certain time (here 30ms).
- Decoded M functions (1) are followed by a short dwell (here 30 ms) to permit the management of the acknowledge.

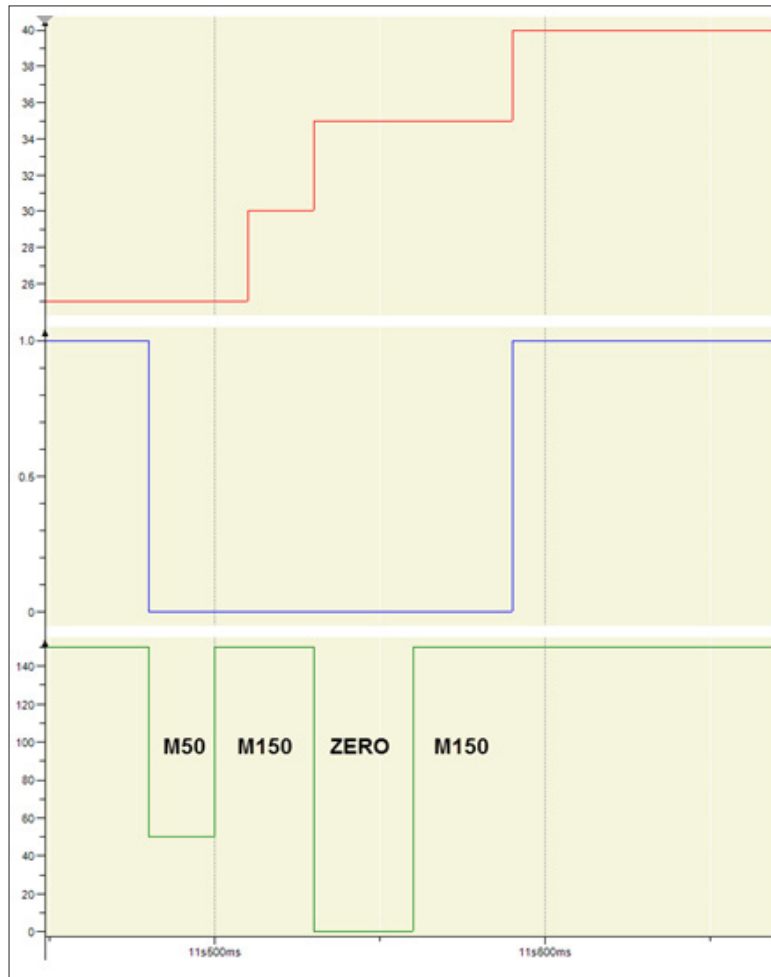


Details

- From top to bottom:
 - Bloc number
 - Axis moving
 - M Code
- In standard mode M coded are always separated by a zero value.
- The duration of a pulse is at least two PLC cycles. It is possible to act on the *AckMFunction* while the value is different of zero.



- When Fast coded M functions are activated (P7 N2.5) the value remains as long as another coded M is not programmed.
- Two consecutive M codes are NOT separated by a ZERO value except if they are identical.
- You will notice that the time where the axes are stopped is greatly reduced.



A.9.4 Details on E parameters

The trace is recorded with the following part program:

```
%10.0( )
E10000=0 E10002 = 0 E30000 = 0
M80
N100 G4 F.2
N00 M1
N05 E10000 = 0 E10002 = 0
N10 E10000 = 1 E10002 = 0
N15 E10000 = 0 E10002 = 1
N20 E10000 = 1 E10002 = 1
N25 E10000 = 0 E10002 = 0
N30 E10000 = 1 E10002 = 0 E30000 = 1200
N35 E10000 = 0 E10002 = 1 E30000 = 1400
N40 E10000 = 1 E10002 = 1 E30000 = 1600
N45 E10000 = 0 E10002 = 0 E30000 = 1800
N50 E10000 = 1 E10002 = 0 E30000 = 1000
N55 E10000 = 0 E10002 = 1 M1
N60 E10000 = 1 E10002 = 1 M1
N65 E10000 = 0 E10002 = 0 M1
N70 E10000 = 1 E10002 = 0 M1
M2
```

Comments

- E 10xxx parameters stop bloc preparation but do not require an acknowledgement therefore they are transmitted as fast as possible in the PLC and some values can overlap each other if no precaution is taken (Left of the first cursor)
- An M1 (for example) makes the bloc executable. Therefore all values are visible in the PLC (right of the second cursor – Blocs N55 and up)
- E30xxx parameters are transmitted by protocol therefore we can see all successive values in the PLC.



A.9.5 Axes clamp

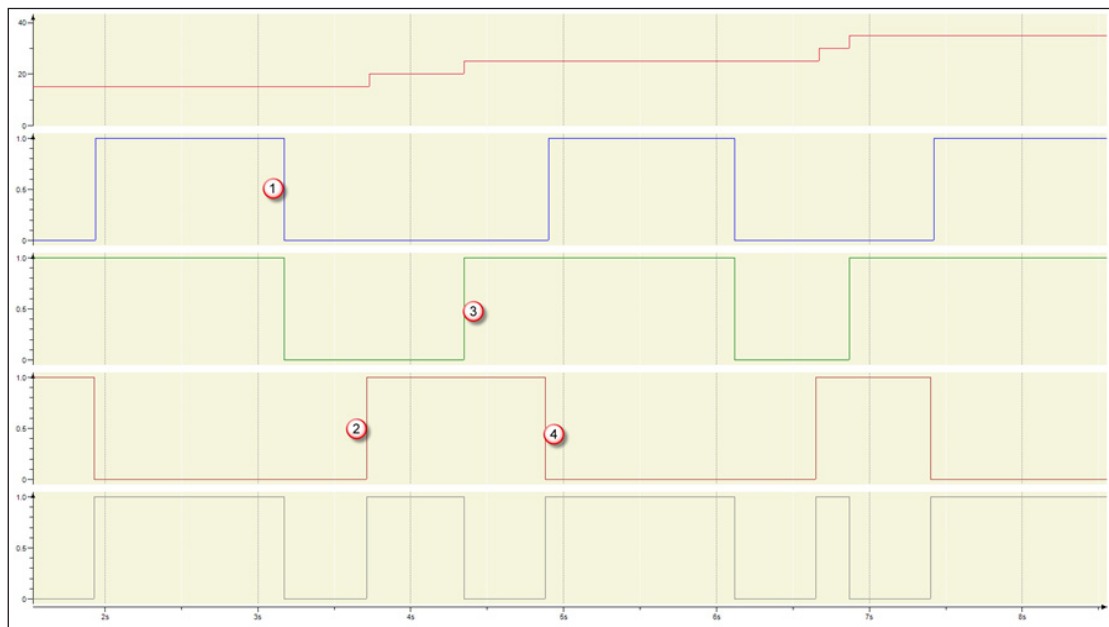
The trace is recorded with the following part program:

```
%10 () ()  
M11 G0 X0  
M80  
M10  
G1 F500 X5  
Y5  
X10  
G4 F.2  
X15  
M11  
G0 X0  
M2
```

A.9.5.1 Management by .WCNC.Channel[x].EndExtMovement

From top to bottom

- Bloc number
- Movement on X
- Request for unclamping
- Clamping (when set)
- .WCNC.Channel[x].EndExtMovement.



Comments

Recorded in M 10 mode – In M11 Clamping are ignored.

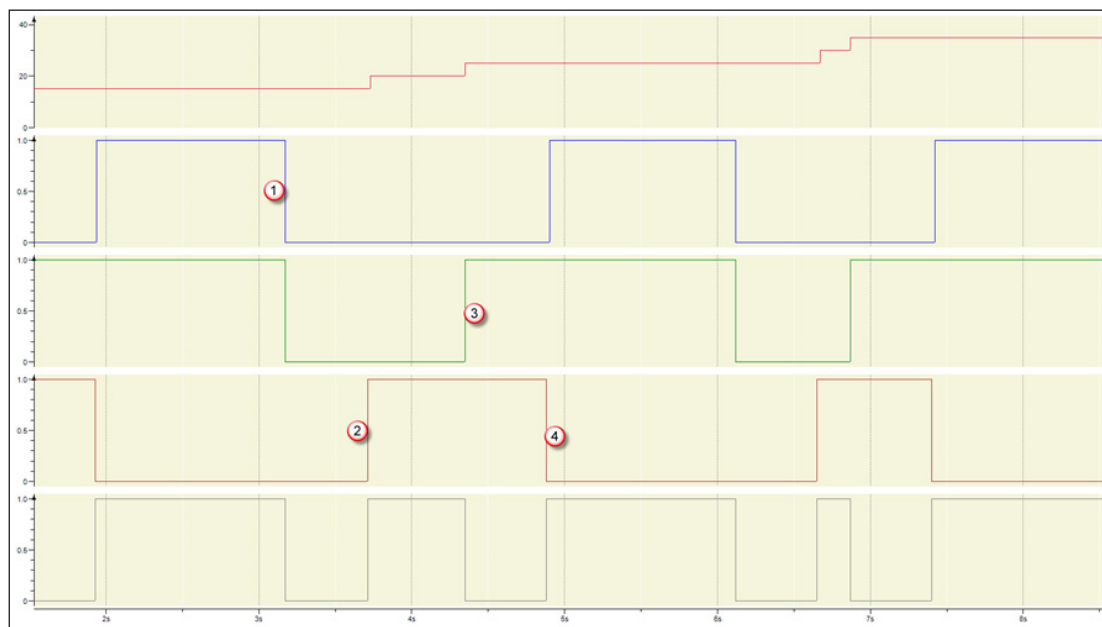
M10 is set by default if some axes with clamps are declared.

1. At the end of the movement the NCK requests clamping.
2. Axis clamped => next block can be executed.
3. Before movement starts there is a request for unclamping.
4. Axis unclamped => movement can start.

A.9.5.2 Management by Axis.Stop

From top to bottom

- Bloc number
- Movement on X
- Request for unclamping
- Clamping (when set)
- .WCNC.Channel[x].EndExtMovement.



Comments

Recorded in M 10 mode – In M11 Clamping are ignored.
M10 is set by default if some axes with clamps are declared.
At the end of the movement the NCK requests clamping
As the clamping feedback is linked to the axis stop and not `.WCNC.Channel[x].EndExtMovement`,
the next block (without
X movement) can be executed immediately
Before the movement starts there is a request for unclamping.
As long as the unclamp feedback is not obtained the movement is stopped.

Conclusion

Axes clamping management can be done via `.WCNC.Channel[x].EndExtMovement` and `WCNC.Axis[x].Stop`.
The difference will be when a movement stops.
Management by Axis[i].Stop permits an immediate program continuation without having received the clamping done feedback.
This can be chosen if such functionality is acceptable.

A.9.6 Spindle indexing

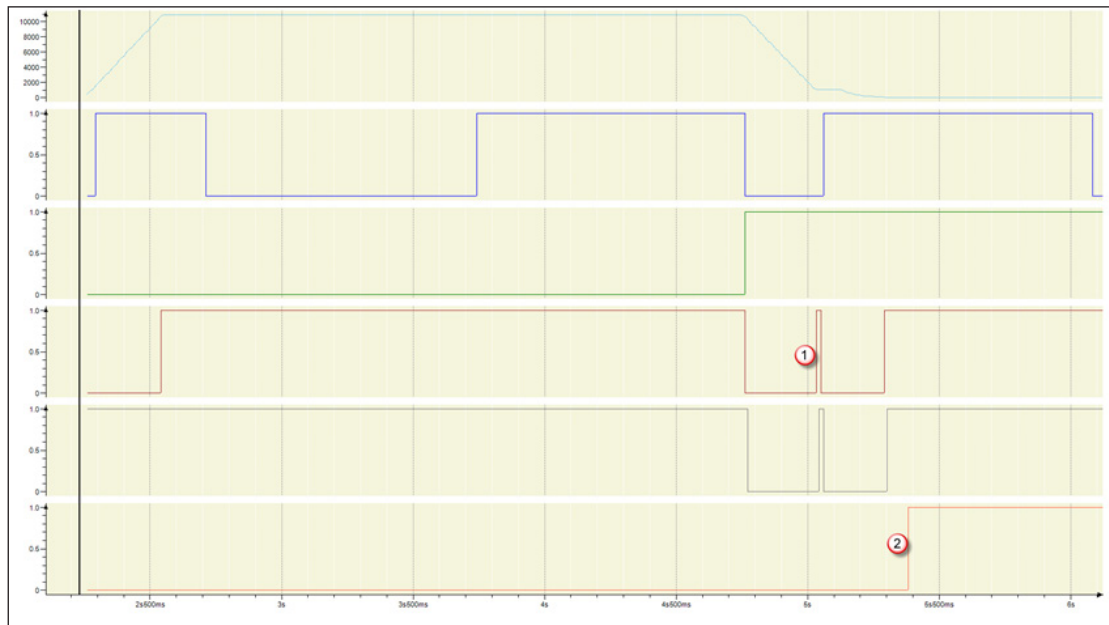
The trace is recorded with the following part program:

```
%10 () ()  
M3 S2000 M49 M11  
G0 X0  
G4 F1  
M80  
G1 X10 F600 M19  
X20  
M2
```

A.9.6.1 Indexing at constant deceleration

From top to bottom

- Spindle speed
- Axes movements
- M19
- AckMFunction
- Speed reached
- Spindle in position.



Comments

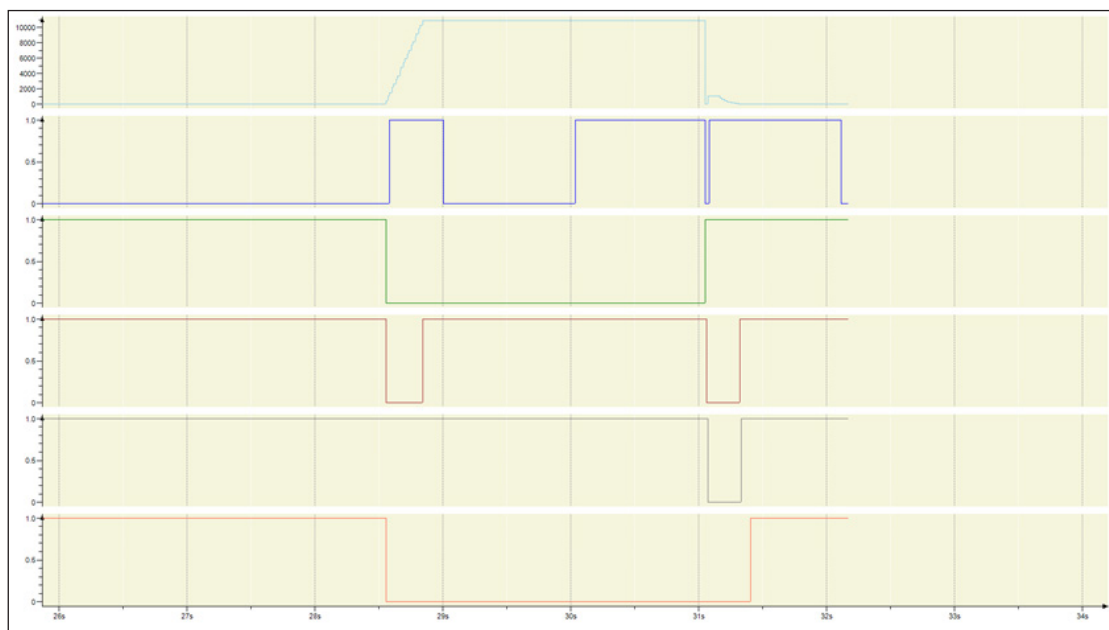
- M19 is issued at the end of the block with a short stop.
- AckMFunction should be cleared as soon as M19 is triggered. The spindle will decelerate down to the Spindle indexing speed value. If AckMFunction is not managed, the indexing will continue over the next block.
- When the speed indexing value is reached then AckMFunction can be released and execution can continue (1).
- Spindle indexing will continue by itself over the next block.
- It is possible to prevent execution of the next block by managing NoStopBlockEnd until Spindle in position is reached (2).

A.9.6.2 Indexing at constant deceleration not chosen

The functionality is identical except that the spindle speed will be immediately set to the indexing speed. The deceleration must be managed by the drive.

From top to bottom

- Spindle speed
- Axes movements
- M19
- AckMFunction
- Speed reached
- Spindle in position.



Page deliberately empty.

Published by NUM Spa

Via F. Somma 62
20012 Cuggiono (MI) (Italy)
Tel. : +39 02 979 691
website: www.num.com

Printed 03.2026