AMKmotion
Software description
AmkExMotionBase

Version: 2023/28
Part no.: 207236
Translation of the "Original Dokumentation"

AMKmotion
MEMBER OF THE ARBURG FAMILY

Imprint

Name: PDK_207236_AmkExMotionBase

Version:

Version	Change	Letter symbol
2023/28	AMKmotion Design	LeS

Previous Version: 2019/39

Product Version:

Product	Software version (Part no.)
Software	SW A7 CDS MotionBase V1.00 2019/33 (207564)

Purpose: Description of the AMK library 'AmkExMotionBase'

Reservation: We reserve the right to modify the content of the documentation as well as the delivery options for the product.

Publisher: AMKmotion GmbH + Co KG
Gaußstraße 37-39
73230 Kirchheim unter Teck
Germany

Phone +49 7021 50 05-0
Fax +49 7021 50 05-176

E-mail: info@amk-motion.com

Registration court: AG Stuttgart, HRA 230681, Kirchheim unter Teck,
Tax Id no.: DE 145 912 804

Complementary: AMKmotion Verwaltungsgesellschaft mbH, HRB 774646

Service: Phone +49 7021 50 05-190, Fax -193

For fast and reliable troubleshooting, you can help us by informing our Customer Service about the following:

- Type plate data for each unit
- Software version
- Device configuration and application
- Type of fault/problem and suspected cause
- Diagnostic messages (error messages)

E-mail service@amk-motion.com

Internet address: www.amk-motion.com

Content

Imprint **2**
1 About this documentation **8**
1.1 Structure of this document 8
1.2 Keeping this document 8
1.3 Target group 8
1.4 Purpose 8
1.5 Display conventions 8
1.6 Appendant documents 8
2 Library AmkExMotionBase **9**
2.1 POUS 9
2.1.1 Basic 9
2.1.1.1 General 9
2.1.1.1.1 MIN_MAX (FB) 9
2.1.2 BasicFunctions 9
2.1.2.1 StateMachineLicenseCheck (ENUM) 10
2.1.2.2 fboGetAttribut (F) 10
2.1.2.3 fdiCeil (F) 10
2.1.3 BasicSupport 10
2.1.3.1 ADD_LIMIT (FB) 10
2.1.3.2 DI_TO_COUNT (FB) 11
2.1.3.3 fboResetWarningError (F) 12
2.1.3.4 fboSetWarningError (F) 12
2.1.3.5 FdiGetSysTime (F) 12
2.1.3.6 PID_TO_KPKIKD (FB) 12
2.1.3.7 SignPositiv (F) 14
2.1.4 Convert (conversion blocks) 14
2.1.4.1 Counter 15
2.1.4.1.1 COUNT_TO_DI (FB) 15
2.1.4.2 Polynominal 15
2.1.4.2.1 CAMXYVA_TO_PROF (FB) 15
2.1.4.2.2 MC_CAM_REF (FB) 16
2.1.4.2.3 XYVA_TO_PROF (FB) 17
2.1.4.3 Visu 19
2.1.4.3.1 PROF_TO_VISU (FB) 19
2.1.5 DeviceAccessAsync (asynchronous device access blocks) 22
2.1.5.1 Command 22
2.1.5.1.1 Control 22
2.1.5.1.1.1 SET_CTRL_ERR_RESET_x_FL (FB) 22
2.1.5.1.2 Status 23
2.1.5.1.2.1 GET_STAT_ERR_RESET_ACK_x_QFL (FB) 23
2.1.5.1.3 HANDLE_FL_QFL (FB) 24
2.1.6 FastFunctions 25
2.1.6.1 POS_J 25
2.1.6.1.1 Functions 25
2.1.6.1.1.1 8Bit 25
2.1.6.1.1.2 16Bit 26
2.1.6.1.1.3 32Bit 26
2.1.6.1.1.4 64Bit 27
2.1.6.1.1.5 bitCount (F) 27
2.1.6.1.1.6 ceil_ulint (F) 27
2.1.6.1.1.7 floor (F) 27

2.1.6.1.1.8 fmod (F) 27
2.1.6.1.1.9 fract (F) 27
2.1.6.1.1.10 random (F) 27
2.1.6.1.1.11 slope1CalcAdjustAmax2_Decelerate (F) 27
2.1.6.1.1.12 slope1CalcAdjustVE_Accelerate (F) 28
2.1.6.1.1.13 slope1CalcAdjustVE_Decelerate (F) 28
2.1.6.1.1.14 slope1CalcAdjustVN (F) 28
2.1.6.1.1.15 slope1CalcEstimateVN (F) 28
2.1.6.1.1.16 slope1CalcExe (F) 28
2.1.6.1.1.17 slope1CalcInit (F) 28
2.1.6.1.1.18 slope1CalcInitP1P4 (F) 28
2.1.6.1.1.19 slope1CalcInitP6P9 (F) 28
2.1.6.1.1.20 slope1CalcPhase1 (F) 28
2.1.6.1.1.21 slope1CalcPhase2 (F) 28
2.1.6.1.1.22 slope1CalcPhase3 (F) 28
2.1.6.1.1.23 slope1CalcPhase4 (F) 28
2.1.6.1.1.24 slope1CalcPhase5 (F) 28
2.1.6.1.1.25 slope1CalcPhase6 (F) 28
2.1.6.1.1.26 slope1CalcPhase7 (F) 28
2.1.6.1.1.27 slope1CalcPhase8 (F) 29
2.1.6.1.1.28 slope1CalcPhase9 (F) 29
2.1.6.1.1.29 slope1CalcStandardProfile (F) 29
2.1.6.1.1.30 truncDint (F) 29
2.1.6.1.2 POS_J (FB) 29
2.1.6.2 CAM_CONT_1 (FB) 31
2.1.6.3 CAM_PROF_1 (FB) 35
2.1.6.3.1 Mathematical consideration of functional relationships 40
2.1.6.3.2 Technical realization of derivations 41
2.1.6.3.2.1 Y- / XY-tables 41
2.1.6.3.2.2 'XYVA-tables' 42
2.1.6.3.3 Table types 42
2.1.6.3.3.1 Y table 42
2.1.6.3.3.2 XY table 43
2.1.6.3.3.3 XYVA table 44
2.1.6.3.4 Number of table interpolation points 46
2.1.6.3.5 Operating modes 47
2.1.6.3.5.1 Phasing in / phasing out 47
2.1.6.3.5.2 Start with auto-stop 48
2.1.6.3.5.3 Immediate start with auto-stop 48
2.1.6.3.6 Online table switchover 48
2.1.6.3.6.1 Synchronous table changeover 49
2.1.6.3.6.2 Non-synchronous table changeover 49
2.1.6.4 PID_CTRL (FB) 50
2.1.6.5 PM_CORRECT (FB) 52
2.1.6.6 PM_DETECT (FB) 54
2.1.6.7 RATIO_ABS (FB) 58
2.1.6.8 RATIO_INC (FB) 59
2.1.6.9 RATIO_INC_1 (FB) 60
2.1.6.10 VGEN (FB) 62
2.1.6.11 VGEN_A (FB) 64
2.1.6.12 VGEN_AJ (FB) 67
2.1.7 FifoHandling (FIFO functions) 72
2.1.7.1 FIFO_HANDLER (FB) 72
2.1.8 ID_Access (ID access functions) 74

2.1.8.1 AllElementsOfOneID 75
2.1.8.1.1 READ_ID_ALL (FB) 75
2.1.8.2 ElementaryAccess 76
2.1.8.2.1 READ_ID_DINT (FB) 76
2.1.8.2.2 READ_ID_DINT_TMP (FB) 77
2.1.8.2.3 READ_ID_LIST (FB) 77
2.1.8.2.4 READ_LIST_512 (FB) 78
2.1.8.2.5 READ_SDO (FB) 79
2.1.8.2.6 WRITE_ID_DINT (FB) 80
2.1.8.2.7 WRITE_ID_DINT_TMP (FB) 80
2.1.8.2.8 WRITE_ID_LIST (FB) 81
2.1.8.2.9 WRITE_LIST_512 (FB) 82
2.1.8.2.10 WRITE_SDO (FB) 83
2.1.8.3 HigherAccess 84
2.1.8.3.1 READ_ID_DINT_ONCE (FB) 84
2.1.8.3.2 WRITE_ID_DINT_ONCE (FB) 85
2.1.8.4 MoreIds 86
2.1.8.4.1 READ_N_IDS_DINT (FB) 86
2.1.8.4.2 WRITE_N_IDS_DINT (FB) 87
2.1.9 Support functions 88
2.1.9.1 ForInternalUse 88
2.1.9.1.1 CLEAR_DINT (FB) 88
2.1.9.1.2 LOCK_EXEC (FB) 88
2.1.9.2 Sercos 88
2.1.9.2.1 CMD_BY_ID (FB) 88
2.1.9.2.2 CMD_START_STOP_BY_ID (FB) 89
2.1.9.2.3 STATE_BY_ID (FB) 90
2.1.10 TabCalc - AMK table calculation blocks 91
2.1.10.1 Operating tables 91
2.1.10.1.1 CALC_OP (FB) 91
2.1.10.2 Phasing in tables 93
2.1.10.2.1 CALC_IN_ALLDEF (FB) 93
2.1.10.2.2 CALC_IN_INDEF (FB) 96
2.1.10.2.3 CALC_IN_SYNCDEF (FB) 99
2.1.10.3 Phasing out tables 102
2.1.10.3.1 CALC_OUT_ALLDEF (FB) 102
2.1.10.3.2 CALC_OUT_OUTDEF (FB) 105
2.1.10.3.3 CALC_OUT_SYNCDEF (FB) 108
2.1.10.4 Positioning profiles 111
2.1.10.4.1 CALC_POS_SPEEDDEF (FB) 111
2.1.10.4.2 CALC_POS_TIMEDEF (FB) 115
2.1.10.5 Support (support blocks) 118
2.1.10.5.1 CALC_CHECK (FB) 118
2.1.10.6 TAB_CALC (FB) 120
2.2 Types 122
2.2.1 Data types 122
2.2.1.1 Basic 123
2.2.1.1.1 General 123
2.2.1.1.1.1 ST_XY_DI (ST) 123
2.2.1.2 Convert 123
2.2.1.2.1 Polynominal 123
2.2.1.2.1.1 EN_XYVA_CONVERT (EN) 123
2.2.1.2.1.2 EN_XYVA_CONV_ERR (EN) 123
2.2.1.2.1.3 ST_POLY_PARA_1 (ST) 123

2.2.1.3 FifoHandling 123
2.2.1.3.1 EN_FIFO_HANDLER_MODE (EN) 123
2.2.1.3.2 EN_FIFO_STATE (EN) 124
2.2.1.3.3 ST_FIFO_HEADER (ST) 124
2.2.1.4 HigherLevel 125
2.2.1.4.1 Visu 125
2.2.1.4.1.1 ST_VISU_TAB (ST) 125
2.2.1.5 POS_J 125
2.2.1.5.1 EN_POS_J_MODE (EN) 125
2.2.1.5.2 EN_SlopeErrorCode (EN) 126
2.2.1.5.3 ST_SlopeCalcData (ST) 126
2.2.1.5.4 ST_SlopeDataIn (ST) 126
2.2.1.5.5 ST_SlopeDataOut (ST) 126
2.2.1.6 Sequencial Positioning 126
2.2.1.6.1 ST_POS_ELE (ST) 126
2.2.2 Enumerations 127
2.2.2.1 Basic 127
2.2.2.1.1 EN_AFP_CODE (EN) 127
2.2.2.1.2 EN_AFP_MODE (EN) 128
2.2.2.1.3 EN_CAM_CONT_MODE (EN) 129
2.2.2.1.4 EN_CAM_PROF_MODE (EN) 129
2.2.2.1.5 EN_CORRECT_MODE (EN) 129
2.2.2.1.6 EN_DETECT_MODE (EN) 130
2.2.2.1.7 EN_POS_AJ_MODE (EN) 130
2.2.2.1.8 EN_POS_MODE (EN) 130
2.2.2.1.9 EN_PROF_TAB_TYPE (EN) 130
2.2.2.1.10 EN_TAB_CALC_KIND (EN) 130
2.2.2.1.11 EN_TAB_CALC_PAR (EN) 131
2.2.2.1.12 EN_TARGET_INFO (EN) 131
2.2.2.1.13 EN_VGEN_MODE (EN) 132
2.2.2.2 Cmd_Afp 132
2.2.2.2.1 EN_HOME_MODE (EN) 132
2.2.2.2.2 EN_SPOS_MODE (EN) 133
2.2.2.2.3 EN_SYNC_MODE (EN) 133
2.2.2.2.4 EN_SYNC_MODE_AL (EN) 133
2.2.2.2.5 EN_SYNC_MODE_CAM (EN) 133
2.2.2.3 MoreIds 134
2.2.2.3.1 EN_ACCESS_N_IDS (EN) 134
2.2.2.3.2 EN_READ_ALL_IDS_STATE (EN) 134
2.2.2.3.3 EN_READ_ID_LIST_ALL_STATE (EN) 134
2.2.2.4 System 134
2.2.2.4.1 EN_AMK_LIB (EN) 134
2.2.3 Structures 134
2.2.3.1 Basic 134
2.2.3.1.1 ST_LOCAL_TIME_INFO (ST) 134
2.2.3.2 CamContactor 135
2.2.3.2.1 ST_CONT (ST) 135
2.2.3.2.2 ST_CONT_TAB (ST) 135
2.2.3.3 CamProfile 136
2.2.3.3.1 SMC_CAMTappet (ST) 136
2.2.3.3.2 SMC_CAMTAPPETACTION (ENUM) 136
2.2.3.3.3 SMC_CAMTAPPETTYPE (ENUM) 136
2.2.3.3.4 SMC_CAMXYVA (ST) 136
2.2.3.3.5 ST_POLY_PARA (ST) 136

2.2.3.3.6 ST_POLY_PARA_INTERNAL (ST) 137
2.2.3.3.7 ST_PROF_TAB (ST) 137
2.2.3.3.8 ST_PROF_TAB_INFO (ST) 137
2.2.3.3.9 ST_PROF_XY (ST) 138
2.2.3.3.10 ST_PROF_XYTAB (ST) 138
2.2.3.3.11 ST_PROF_XYVATAB (ST) 139
2.2.3.3.12 ST_PROF_YTAB (ST) 142
2.2.3.4 PmControl 143
2.2.3.4.1 ST_CORR_FIFO (ST) 143
2.2.3.5 System 144
2.2.3.5.1 ST_PLC_VAR_POINTERS (ST) 144
2.2.4 Structures_1 144
2.2.4.1 AllElementsOfOneID 144
2.2.4.1.1 ST_ID_ALL (ST) 144
2.2.4.1.2 ST_ID_NAME (ST) 145
2.2.4.1.3 ST_ID_UNIT (ST) 145
2.2.4.2 ElementaryAccess 146
2.2.4.2.1 ST_LIST_512 / ST_LIST_1024 / ST_LIST_2048 / ST_LIST_4096 (ST) 146
2.2.4.2.2 ST_LIST_VAR_LEN (ST) 147
2.2.4.3 MoreIds 147
2.2.4.3.1 ST_ID_VALUE (ST) 147
2.2.4.3.2 ST_N_ID_VALUES (ST) 148
2.3 Projektinformationen 148
2.3.1 GetBooleanProperty (F) 148
2.3.2 GetCompany (F) 148
2.3.3 GetNumberProperty (F) 148
2.3.4 GetTextProperty (F) 148
2.3.5 GetTextProperty2 (F) 148
2.3.6 GetTitle (F) 149
2.3.7 GetVersion (F) 149
2.3.8 GetVersionProperty (F) 149
3 Appendix **150**
3.1 Error bit information 150
3.2 Table 1: Global AmkFile function block error codes 152
Glossary **153**
Your opinion is important! **154**

1 About this documentation

1.1 Structure of this document

Topic	Chapter	Chapter number
Validity, use and the propose of the documentation	Imprint	-
	About this documentation	1
Description of the 'AmkExMotionBase' library	AmkExMotionBase Library	2
Abbreviations and explanation of terms	Glossary	-

1.2 Keeping this document

This document must permanently be available and readable at the place where the product is in use. If the product is used at another place or changed the owner, the document must be passed on.

1.3 Target group

Any person that is qualified and intends to work with this product must read, understand and follow this document:

- PLC programming

1.4 Purpose

This document is addressed to any person who handles the product. It gives information about the following topics:

- PLC programming

1.5 Display conventions

Display	Meaning
	This symbol points to parts of the text to which particular attention should be paid!
0x	0x followed by a hexadecimal number, e. g. 0x500A
'Names'	Names are represented with apostrophes e. g. parameters, variables, etc.
See 'chapter name' on page x	Executable cross-reference in electronic output media
Blue text	Executable link in electronic output media

1.6 Appendant documents

Software description

AMK part-no.	Title
207438	Controllers A7 includes the following PLC libraries: • AmkEtC_M • AmkExBase
207568	AmkExAfl library

2 Library AmkExMotionBase

The 'AmkExMotionBase.lib' library created in IEC61131-3 replaces the functions and function blocks of the AMK libraries' AmkBase.lib, AmkSupport.lib and AmkSystem.lib.

2.1 POUS

Basic	Basic (Basic functions)
BasicSupport	BasicSupport (Supported basic functions)
Convert	Convert (Conversion blocks)
DeviceAccessAsync	DeviceAccessAsync (asynchronous device access blocks)
FastFunctions	FastFunctions (Fast functions)
FifoHandling	FifoHandling (FIFO functions)
ID_Access	ID_Access (ID access functions)
Support	Support (Supported functions)
TabCalc	TabCalc (Table calculation blocks)

2.1.1 Basic

MIN_MAX	Extreme value determination with reset

2.1.1.1 General

2.1.1.1.1 MIN_MAX (FB)

The 'MIN_MAX' function block provides the extreme values (minimum, maximum) of the input variables in the two output variables.

User interface

Input variables

Name	Type	Description
boReset	BOOL	Reset signal FALSE: Extreme value generation in progress TRUE: 'diMinVal' := 'diMaxVal' := 'diActVal'
diActVal	DINT	Input value, actual value Minimum and maximum since the last reset are provided as output values

Output variables

Name	Type	Description
diMinVal	DINT	Minimum output value since last reset
diMaxVal	DINT	Maximum output value since last reset

2.1.2 BasicFunctions

StateMachineLicenseCheck	Reserved for AMK internal use!
fboGetAttribut	Reserved for AMK internal use!
fdiCeil	Reserved for AMK internal use!

2.1.2.1 StateMachineLicenseCheck (ENUM)

Reserved for AMK internal use!

2.1.2.2 fboGetAttribut (F)

Reserved for AMK internal use!

2.1.2.3 fdiCeil (F)

Reserved for AMK internal use!

2.1.3 BasicSupport

The user has direct access to BasicSupport functions and function blocks.

ADD_LIMIT	Addition with limitation
DI_TO_COUNT	AMK 32-bit counter output value converter
fboResetWarningError	Reset warning / error - Reserved for AMK internal use!
fboSetWarningError	Set warning / error - Reserved for AMK internal use!
FdiGetSysTime	Get time differences
PID_TO_KPKIKD	AMK PID converter
SignPositiv	Check if sign positive - Reserved for AMK internal use!

2.1.3.1 ADD_LIMIT (FB)

The 'ADD_LIMIT' function block adds two DINT type variables. The result is also DINT.
The result can be limited to a minimum or a maximum value.

Thus:

diValAB :=:= diValA + diValB where: diMin ≤ diValAB ≤ diMax

And:

boMin = TRUE if (diValA + diValB) < diMin
boMax = TRUE if (diValA + diValB) > diMax

Application note:
Combined with the 'PID_CTRL' function block, 'ADD_LIMIT' supports additive feed-forward control with limitation of the command variable, for example

User interface

Input variables

Name	Type	Description
diValA	DINT	Input value A, added to input value B
diValB	DINT	Input value B, added to input value A
diMax	DINT	Maximum permissible value
diMin	DINT	Minimum permissible value

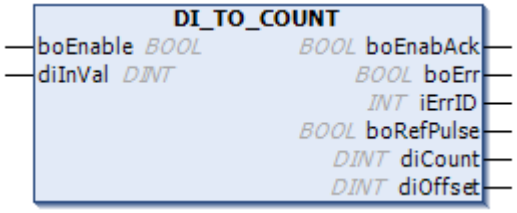
Output variables

Name	Type	Description
diValAB	DINT	Output value AB Results from the addition of input value A and input value B taking the limit into account
boMax	BOOL	boMax = TRUE: The output value AB has been limited to the maximum permissible output value
boMin	BOOL	boMin = TRUE: The output value AB has been limited to the minimum permissible output value

2.1.3.2 DI_TO_COUNT (FB)

The 'DI_TO_COUNT' function block converts the 32-bit AMK pulse encoder information into various values that are relevant to the process.

User interface



Input variables

Name	Type	Description
boEnable	BOOL	Enable signal: With a positive edge, the initialisation of the block starts. As long as 'boEnable' = TRUE, the block remains enabled and is processed by the PLC. In the state 'boEnable' = FALSE the block is no longer enabled and is thus no longer processed.
diInVal	DINT	Input value Pulse encoder information 32-bit AMK data format: $diInVal_{LW}$ = low word: latched counter reading for zero pulse $diInVal_{HW}$ = high word: current 16-bit counter reading

Output variables

Name	Type	Description
boEnabAck	BOOL	Acknowledgement: Function block is initialised and enabled
boErr	BOOL	The function block is in an error state FALSE: No error (permitted commanding or warning) TRUE: Error
iErrID	INT	Error identity number: Diagnostic number is output iErrID = 0: No error iErrID ≠ 0, boErr = TRUE: Error iErrID ≠ 0, boErr = FALSE: Warning
boRefPulse	BOOL	Homing pulse: The output adopts the value 'boRefPulse' = TRUE for one cycle Display of zero pulse detected through the input logic of the square-wave pulse encoder

Name	Type	Description
diCount	DINT	32-bit Counter value, generated for each cycle from the changes in the value of the current 16-bit counter reading The value is reset on a positive edge at 'boEnable' ('diCount' = 0)
diOffset	DINT	Offset of the counter value to the homing pulse Unit: Incr $diOffset = diInVal_{LW} - diInVal_{HW}$

2.1.3.3 fboResetWarningError (F)

Reserved for AMK internal use!

2.1.3.4 fboSetWarningError (F)

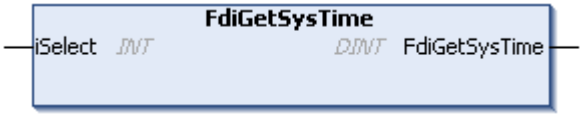
Reserved for AMK internal use!

2.1.3.5 FdiGetSysTime (F)

The 'FdiGetSysTime' queries time differences based on an internal system time base.

The time base is updated independently of the PS cycle time but dependent upon the target system. The maximum time difference that can be measured is also determined by the target system.

User interface



Input variables

Name	Type	Description
iSelect	INT	Time selection used to select the time determination mode Value: 0 — Meaning: Time difference; time between two calls

Output variables

Name	Type	Description
FdiGetSysTime	DINT	System time Unit: µs

2.1.3.6 PID_TO_KPKIKD (FB)

The 'PID_TO_KPKIKD' function block calculates the proportional time, the reset time, the derivative time, and the sampling time for the controller parameters Kp, Ki, and Kd from the proportional component. The 'PID_CTRL' function block requires these values as input values.

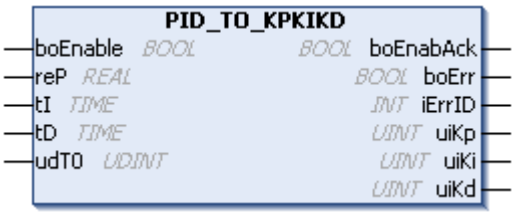
The function block is called in the asynchronous program level PLC_PRG.

Thus:

'PID_CTRL' combines with 'PID_TO_KPKIKD' to form a PID controller whose standard control parameters are independent of the sampling time.
This separation makes it possible for the two modules to work with different time frequencies in applications with limited processing time.

The time-consuming calculation of Kp, Ki, Kd with 'PID_TO_KPKIKD' can be performed at asynchronous program level PLC_PRG while the runtime-optimized control algorithm is being processed at synchronous program level FPLC_PRG.

User interface



Input variables

Name	Type	Description
boEnable	BOOL	Enable signal: With a positive edge, the initialisation of the block starts. As long as 'boEnable' = TRUE, the block remains enabled and is processed by the PLC. In the state 'boEnable' = FALSE the block is no longer enabled and is thus no longer processed.
reP	REAL	Proportional component Internal normalization 1/256: reP = 1 -> Kp = 256 -> P gain of PID_CTRL = 1
tI	TIME	Integration time constant: reset time Tn Unit: ms Default: 4294967295
tD	TIME	Differentiation time constant: derivative time Tv Unit: ms
udT0	UDINT	Sampling time for processing the PID algorithm Unit: 0.001 ms Default: 1000

Output variables

Name	Type	Description
boEnabAck	BOOL	Acknowledgement: Function block is initialised and enabled
boErr	BOOL	The function block is in an error state FALSE: No error (permitted commanding or warning) TRUE: Error

<table>
<tr><th>Name</th><th>Type</th><th>Description</th></tr>
<tr><td>iErrID</td><td>INT</td><td>Error identity number: Diagnostic number is output
<table>
<tr><td colspan="2">iErrID = 0</td><td>No error</td></tr>
<tr><td>iErrID ≠ 0</td><td>boErr = TRUE</td><td>Error</td></tr>
<tr><td>iErrID ≠ 0</td><td>boErr = FALSE</td><td>Warning</td></tr>
</table>
<table>
<tr><th>Value</th><th>Meaning</th></tr>
<tr><td>1</td><td>Kp rounded to 0</td></tr>
<tr><td>2</td><td>Kp rounded to 1</td></tr>
<tr><td>3</td><td>Kp limited to 32767</td></tr>
<tr><td>4</td><td>Ki rounded to 0</td></tr>
<tr><td>5</td><td>Ki rounded to 1</td></tr>
<tr><td>6</td><td>Ki limited to 32767</td></tr>
<tr><td>7</td><td>Kd rounded to 0</td></tr>
<tr><td>8</td><td>Kd rounded to 1</td></tr>
<tr><td>9</td><td>Kd limited to 32767</td></tr>
</table></td></tr>
<tr><td>uiKp</td><td>UINT</td><td>Proportional gain (P) of the PID controller
<table>
<tr><td>Unit</td><td>1/256</td></tr>
</table></td></tr>
<tr><td>uiKi</td><td>UINT</td><td>Integration gain (I) of the PID controller.
<table>
<tr><td>Unit</td><td>1/256</td></tr>
</table></td></tr>
<tr><td>uiKd</td><td>UINT</td><td>Differential gain (D) of the PID controller
<table>
<tr><td>Unit</td><td>1/256</td></tr>
</table></td></tr>
</table>

2.1.3.7 SignPositiv (F)

Reserved for AMK internal use!

2.1.4 Convert (conversion blocks)

Counter

COUNT_TO_DI	Generate DINT pulse encoder information

Polynomial

CAMXYVA_TO_PROF	Convert 3S structures into AMK structure
MC_CAM_REF	Table header information specific to 3S
XYVA_TO_PROF	Conversion of table interpolation points

Visu

PROF_TO_VISU	Calculate visualization tables for graphical representation of X / XY table characteristics

2.1.4.1 Counter

2.1.4.1.1 COUNT_TO_DI (FB)

The 'COUNT_TO_DI' function block converts the counter values 'boRefPulse', 'diCount', 'diOffset' into an AMK counter value / pulse encoder information.

'COUNT_TO_DI' is the inverse of the block 'DI_TO_COUNT'. (See document Software description AmkBase Bibliothek, Part no. 204986)

User interface

Input variables

Name	Type	Description
boRefPulse	BOOL	Homing pulse Displays a detected zero pulse How long is the signal pending for?
diCount	DINT	32-bit Counter value generated from the value changes in the current 16-bit counter status read during each cycle
diOffset	DINT	Offset of the counter value to the homing pulse Unit: Incr

Output variables

Name	Type	Description
diOutVal	DINT	Pulse encoder information AMK 32-bit data format Low word = $diOutVal_{LW}$ generated on the homing pulse from the counter value corrected by the offset High word = $diOutVal_{HW}$ current 16-bit counter reading

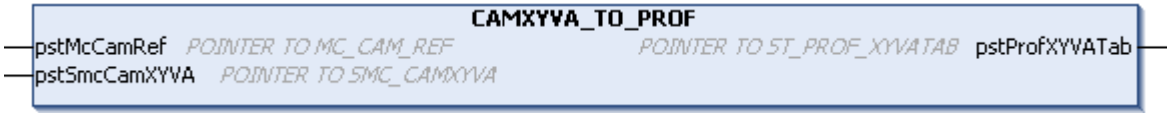
2.1.4.2 Polynominal

2.1.4.2.1 CAMXYVA_TO_PROF (FB)

The 'CAMXYVA_TO_PROF' function block converts the CODESYS structures 'MC_CAM_REF' and 'ARRAY[0...N] OF SMC_CAMXYVA' into the AMK structure 'ST_PROF_XYVATAB' and generates a pointer to this structure for the 'CAM_PROF_1'.

The definition of these structures is based on 3S libraries which are only integrated with the full Softmotion license.

User interface



Input variables

Name	Type	Description
pstMcCamRef	POINTER	POINTER TO MC_CAM_REF Pointer to the 3S header structure generated by the CAM editor 'MC_CAM_REF'
pstSmcCamXYVA	POINTER	POINTER TO SMC_CAMXYVA Pointer to the 3S interpolation point array generated by the CAM editor ARRAY [0..N] OF SMC_CAMXYVA

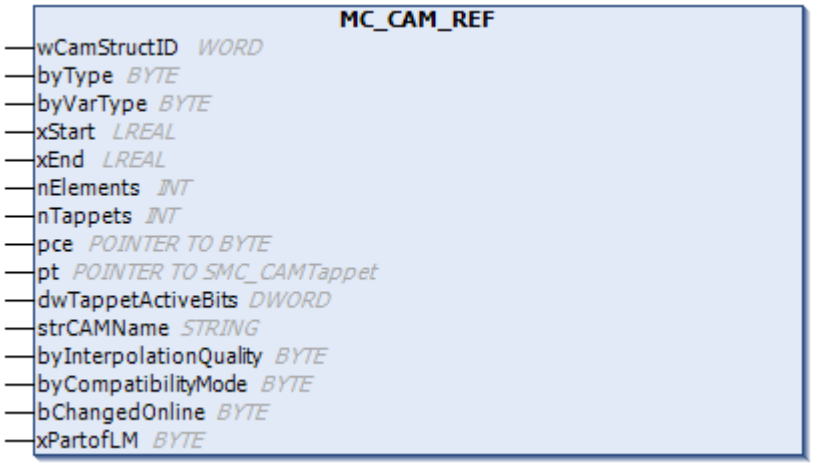
Output variables

Name	Type	Description
pstProfXYVATab	POINTER	POINTER TO ST_PROF_XYVATAB Pointer to the AMK-specific XYVA table structure ST_PROF_XYVATAB expected by the 'CAM_PROF_1' AMK function block

2.1.4.2.2 MC_CAM_REF (FB)

The 'MC_CAM_REF' function block contains the type definitions specific to 3S which are needed by the CODESYS cam disk editor in order to write polynomial tables.

User interface



Input variables

Name	Type	Description
byType	BYTE	Table type (Not used in the context of the specific function for AMK)
xStart	LREAL	Start of master position
xEnd	LREAL	End of master position
nElements	INT	Number of table elements is used, for example, by the 'CAMXYVA_TO_PROF' block to calculate 'uiNoElement' 'uiNoElement' = 'nElements' - 1
nTappets	INT	Number of cams (Not used in the context of the specific function for AMK)
strCAMName	STRING	CAM-Name (Not used in the context of the specific function for AMK)

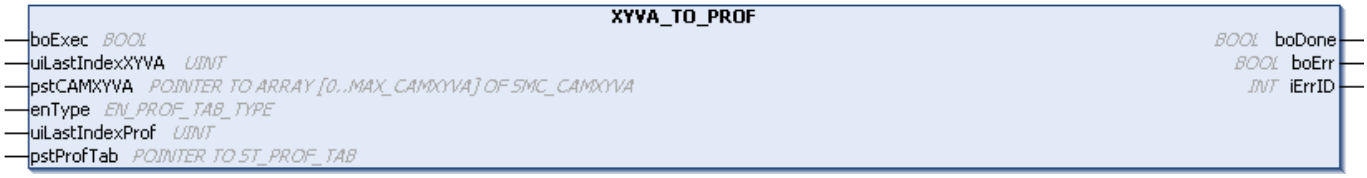
2.1.4.2.3 XYVA_TO_PROF (FB)

The 'XYVA_TO_PROF' function block converts a 3S polynomial XYVA interpolation point table 'ARRAY[0..MAX_CAMXYVA] OF SMC_CAMXYVA' into an AMK Y or XY table (See document Software description AmkBase Bibliothek, Part no. 204986).

However, unlike the 'CAMXYVA_TO_PROF' block, the description of the curve is converted in full. A table based on the Y or XY format is calculated from a table in XYVA format based on the 'SMC_CAMXYVA' format.

The 'XYVA_TO_PROF' block thus supports offline conversion of the XYVA format and the use of the result with a 'CAM_PROF_1' which does not support the XYVA format, or supports the display of a table in XYVA format in a graph through the 'PROF_TO_VISU' block.

User interface



Input variables

Name	Type	Description
boExec	BOOL	Function execution: With a positive edge, the execution of the block starts. As long as 'boExec' = TRUE, the block is processed by the PLC. In the state 'boExec' = FALSE execution of the block is ended.
uiLastIndexXYVA	UINT	Last array index of the XYVA input table 'SMC_CAMXYVA' Range: 1.. MAX_CAMXYVA where MAX_CAMXYVA:UINT := 31
pstCAMXYVA	POINTER	POINTER TO ARRAY [0..MAX_CAMXYVA] OF SMC_CAMXYVA Pointer to the XYVA input table specific to 3S
enType	ENUM	EN_PROF_TAB_TYPE Table type Selection of the phasing out table type See document Software description AmkBase Bibliothek, Part no. 204986 Default: PROF_YTAB Range: PROF_YTAB = Y table; PROF_XYTAB = XY table
uiLastIndexProf	UINT	Last array index of the Y / XY output table 'CAM_PROF_1' Range: 1 .. MAX_PROF_Y_IND if 'enType' := PROF_YTAB; 1 .. MAX_PROF_XY_IND if 'enType' := PROF_XYTAB
pstProfTab	POINTER	POINTER TO ST_PROF_TAB Pointer to the Y / XY output table

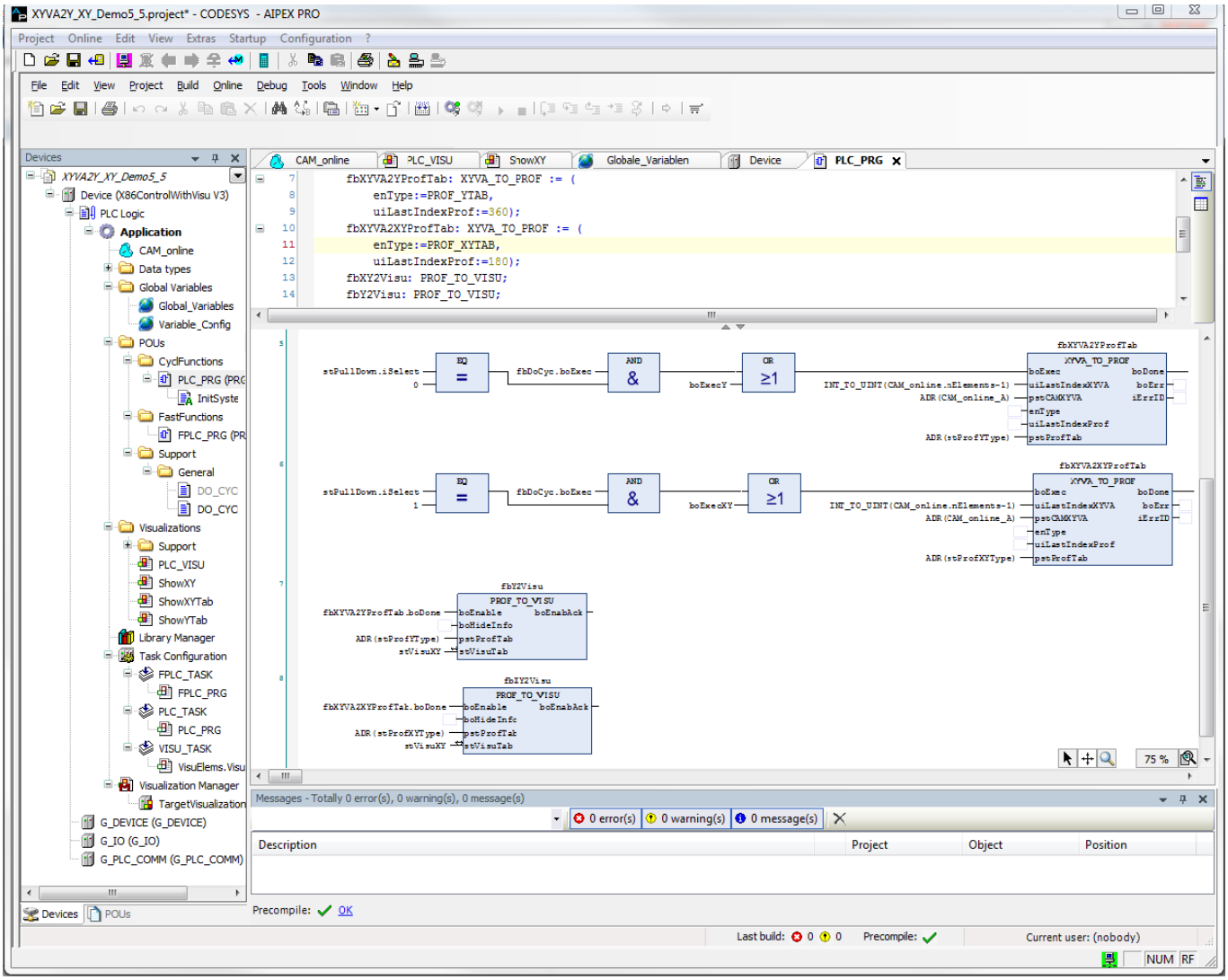
Output variables

Name	Type	Description
boDone	BOOL	Response that the function block has been completely executed.
boErr	BOOL	The function block is in an error state FALSE: No error (permitted commanding or warning) TRUE: Error

<table>
<tr><th>Name</th><th>Type</th><th>Description</th></tr>
<tr><td>iErrID</td><td>INT</td><td>Error identity number: Diagnostic number is output
<table>
<tr><td colspan="2">iErrID = 0</td><td>No error</td></tr>
<tr><td>iErrID ≠ 0</td><td>boErr = TRUE</td><td>Error</td></tr>
<tr><td>iErrID ≠ 0</td><td>boErr = FALSE</td><td>Warning</td></tr>
</table>
<table>
<tr><td>Range</td><td>Meaning</td></tr>
<tr><td>Type</td><td>EN_XYVA_CONV_ERR</td></tr>
<tr><td>1</td><td>XYVA_CONV_ILL_TYPE
Invalid table type</td></tr>
<tr><td>2</td><td>XYVA_CONV_ILL_IND_PROF
Invalid index of output table</td></tr>
<tr><td>3</td><td>XYVA_CONV_ILL_INDXYVA
Invalid index of input table</td></tr>
<tr><td>4</td><td>XYVA_CONV_ILL_START
Starting point of XYVA table is not {0,0}</td></tr>
<tr><td>5</td><td>XYVA_CONV_ILL_END_IND
Invalid index of input or output table at end of conversion, possibly due to the value for 'uiLastIndexProf' being too low</td></tr>
</table></td></tr>
</table>

Description

Abbildung 1: XYVA_TO_PROF: Conversion for online visualization



2.1.4.3 Visu

2.1.4.3.1 PROF_TO_VISU (FB)

The 'PROF_TO_VISU' function block converts a Y or XY table into a structure that is suitable for displaying the curve characteristic in a graph ('ST_VISU_TAB').

User interface

Input variables

Name	Type	Description
boEnable	BOOL	Enable signal: With a positive edge, the initialisation of the block starts. As long as 'boEnable' = TRUE, the block remains enabled and is processed by the PLC. In the state 'boEnable' = FALSE the block is no longer enabled and is thus no longer processed.
boHideInfo	BOOL	Suppress display of X axis information in ViXY
pstProfTab	POINTER	POINTER TO ST_PROF_TAB Pointer to the Y / XY output table

Output variables

Name	Type	Description
boEnabAck	BOOL	Acknowledgement: Function block is initialised and enabled

Input and output variables

Name	Type	Description
stVisuTab	STRUCT	ST_VISU_TAB Structure of the visualization table Basis for the ViXY and ViCursor visualization blocks

Description

ViXY visualization supports online display of the curve characteristic in a graph (an XY diagram).
The curve characteristic is approximated with MAX_VISU_XY := 80 linear partial segments.
ViCursor visualization allows the cursor of the XY diagram to be manipulated.

ViXY and ViCursor are linked by referencing visualizations with the 'ST_VISU_TAB' structure.

Abbildung 2: PROF_TO_VISU: Online visualization

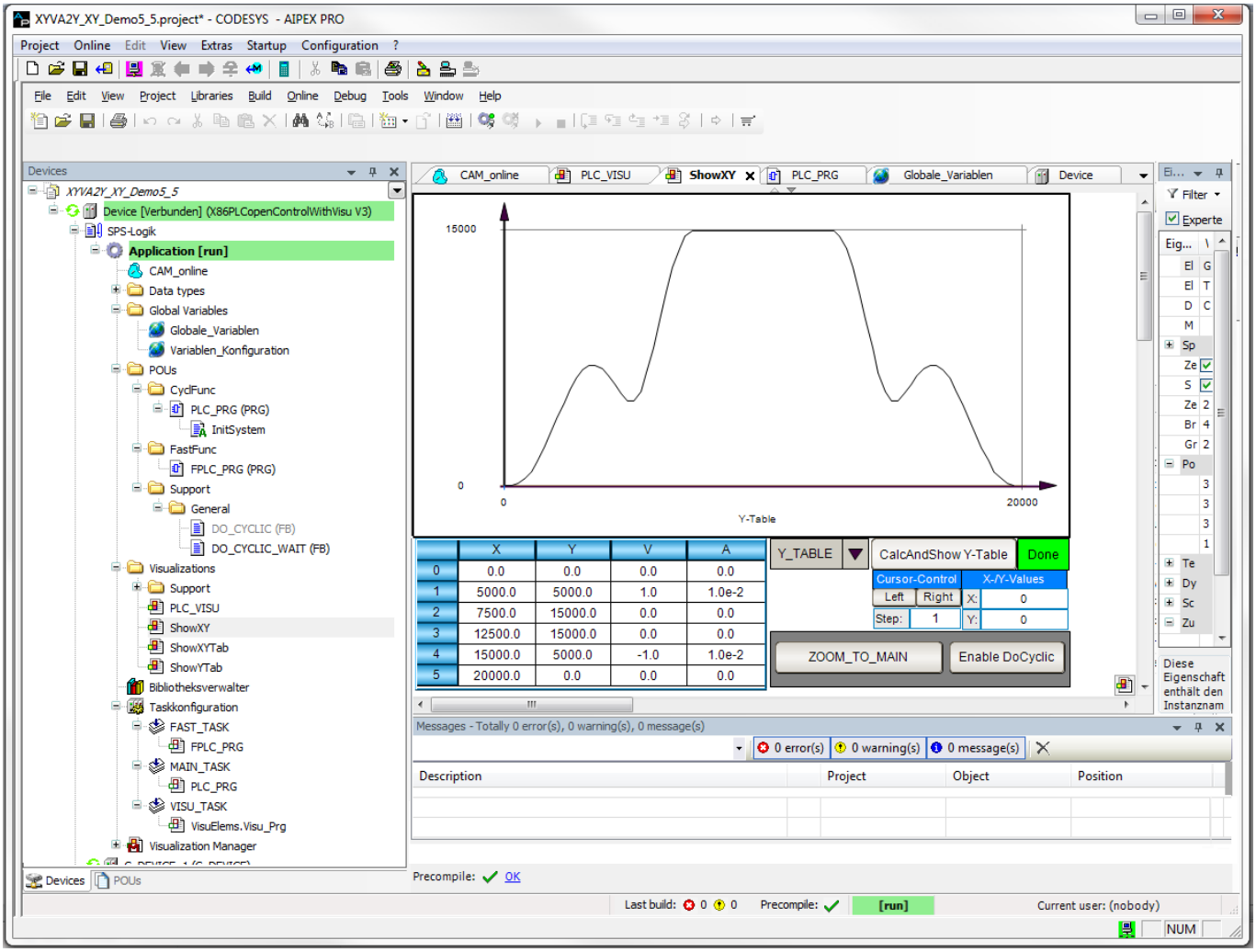
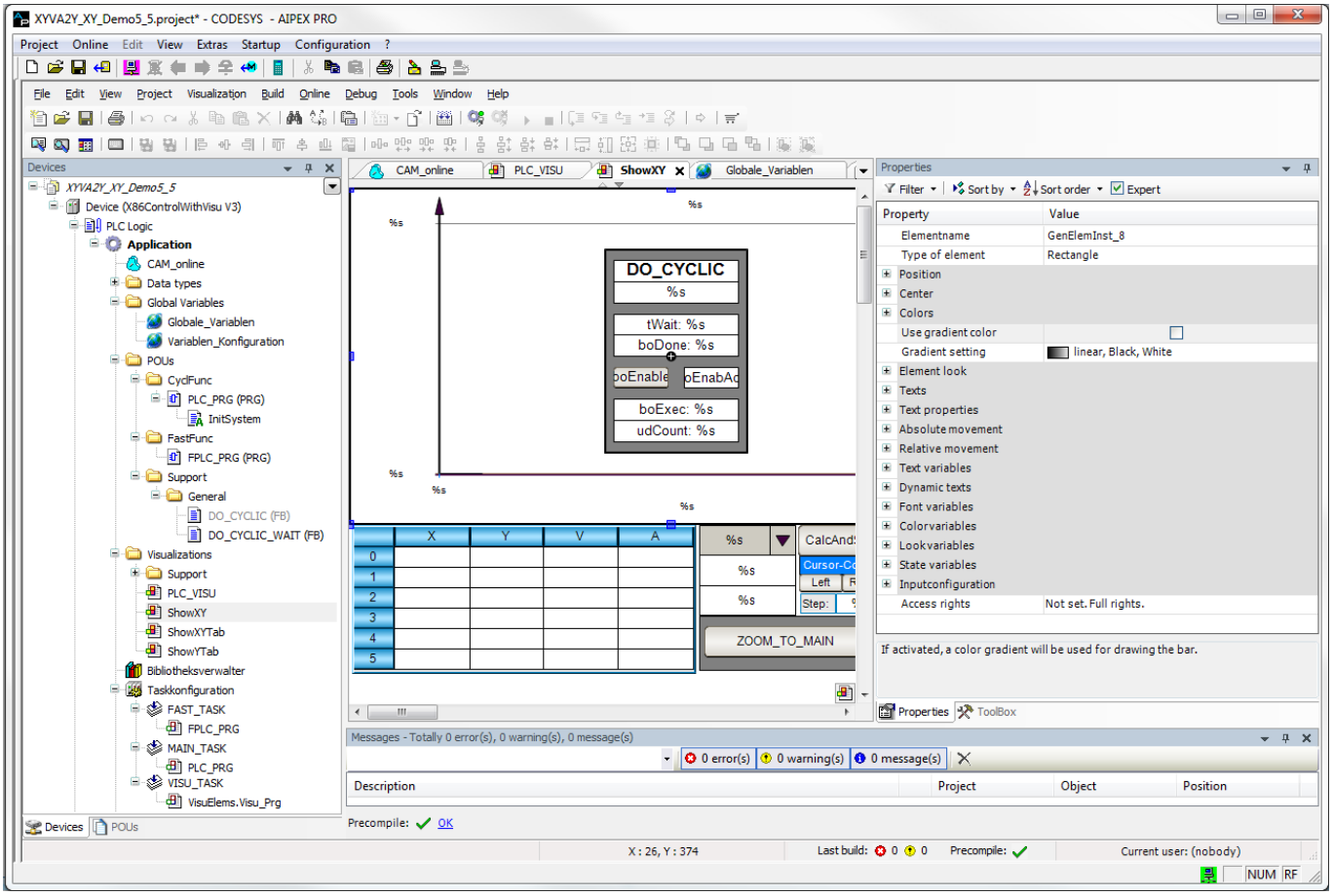
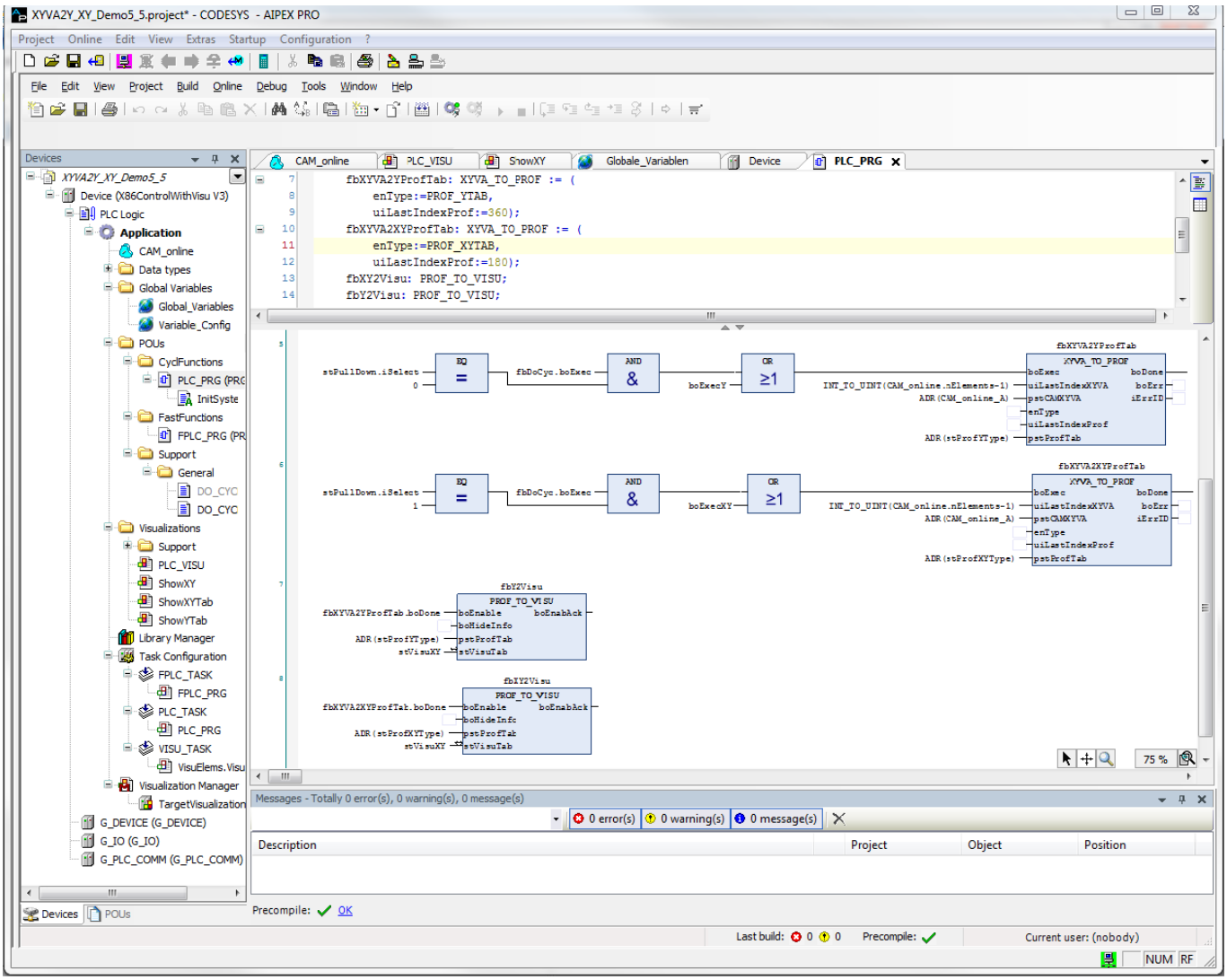


Abbildung 3: PROF_TO_VISU: Referenced visualization



To display an XYVA table in a graph, the table can be converted to a Y or XY table first with the 'XYVA_TO_PROF' block.

Abbildung 4: PROF_TO_VISU: Conversion for online visualization



2.1.5 DeviceAccessAsync (asynchronous device access blocks)

Command

Control

SET_CTRL_ERR_RESET_x_FL	Set "error reset" (FL)

Status

GET_STAT_ERR_RESET_ACK_x_QFL	Get "error reset acknowledge" (QFL)

Support block

HANDLE_FL_QFL	Block is used to organize error resets (FL)

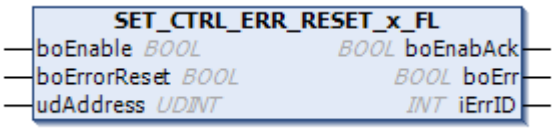
2.1.5.1 Command

2.1.5.1.1 Control

2.1.5.1.1.1 SET_CTRL_ERR_RESET_x_FL (FB)

This block sets "error reset" (FL) through the 'boErrorReset' variable.

User interface



Input variables

Name	Type	Description
boEnable	BOOL	Enable signal: With a positive edge, the initialisation of the block starts. As long as 'boEnable' = TRUE, the block remains enabled and is processed by the PLC. In the state 'boEnable' = FALSE the block is no longer enabled and is thus no longer processed.
boErrorReset	BOOL	Error Reset (FL = clear error)
udAddress	UDINT	EtherCAT address

Output variables

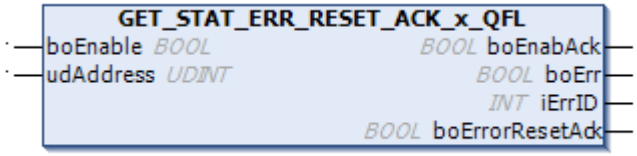
Name	Type	Description
boEnabAck	BOOL	Acknowledgement: Function block is initialised and enabled
boErr	BOOL	The function block is in an error state FALSE: No error (permitted commanding or warning) TRUE: Error
iErrID	INT	Error identity number: Diagnostic number is output iErrID = 0: No error iErrID ≠ 0, boErr = TRUE: Error iErrID ≠ 0, boErr = FALSE: Warning Error: 1: Not configured device Information 2: Unassigned input / output variable 3: Invalid device instance (e.g. symbolic device identifier wrong assigned)

2.1.5.1.2 Status

2.1.5.1.2.1 GET_STAT_ERR_RESET_ACK_x_QFL (FB)

This block queries "error reset acknowledge" (QFL) through the 'boDcBusEnabAck' variable.

User interface



Input variables

Name	Type	Description
boEnable	BOOL	Enable signal: With a positive edge, the initialisation of the block starts. As long as 'boEnable' = TRUE, the block remains enabled and is processed by the PLC. In the state 'boEnable' = FALSE the block is no longer enabled and is thus no longer processed.
udAddress	UDINT	EtherCAT address

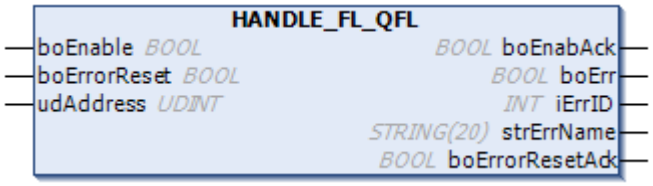
Output variables

Name	Type	Description
boEnabAck	BOOL	Acknowledgement: Function block is initialised and enabled
boErr	BOOL	The function block is in an error state FALSE: No error (permitted commanding or warning) TRUE: Error
iErrID	INT	Error identity number: Diagnostic number is output iErrID = 0: No error iErrID ≠ 0, boErr = TRUE: Error iErrID ≠ 0, boErr = FALSE: Warning Error: 1: Not configured device Information 2: Unassigned input / output variable 3: Invalid device instance (e.g. symbolic device identifier wrong assigned)
boErrorResetAck	BOOL	Error Reset Acknowledge (QFL = acknowledgement clear error)

2.1.5.1.3 HANDLE_FL_QFL (FB)

The 'Handle_FL_QFL' block is used to organize error resets (FL) with generation of acknowledgement information (QFL). In EtherCAT-based devices, ID99 'Diagnosis reset status class 1' is only read; as long as 'boErrorReset' is activated.

User interface



Input variables

Name	Type	Description
boEnable	BOOL	Enable signal: With a positive edge, the initialisation of the block starts. As long as 'boEnable' = TRUE, the block remains enabled and is processed by the PLC. In the state 'boEnable' = FALSE the block is no longer enabled and is thus no longer processed.
boErrorReset	BOOL	Error Reset (FL = clear error)
udAddress	UDINT	EtherCAT address

Output variables

Name	Type	Description
boEnabAck	BOOL	Acknowledgement: Function block is initialised and enabled
boErr	BOOL	The function block is in an error state FALSE: No error (permitted commanding or warning) TRUE: Error
iErrID	INT	Error identity number: Diagnostic number is output iErrID = 0: No error iErrID ≠ 0, boErr = TRUE: Error iErrID ≠ 0, boErr = FALSE: Warning
strErrName	STRING (20)	Block name of the module generating the error 'DEV_ER__FL': SET_CTRL_ERR_RESET_x_FL 'DEV_ERA__QFL': GET_STAT_ERR_RESET_ACK_x_QFL
boErrorResetAck	BOOL	Error Reset Acknowledge (QFL = acknowledgement clear error)

2.1.6 FastFunctions

The function blocks in the 'FastFunctions' group must be called in the 'FPLC_PRG' program block which is called in the PLC as a real-time task with equidistant sampling. These function blocks generate output values whose increment by sampling time is used as the cyclic setpoint for drive control.

POS_J	Fast positioning module with variable specification of the position, velocity, acceleration and jerk
CAM_CONT_1	Camshaft control with higher accuracy of position assignment
CAM_PROF_1	Table-based function interpolator with additional output of the 1st and 2nd derivation of the table function
PID_CTRL	PID controller
PM_CORRECT	Printing mark correction
PM_DETECT	Printing mark detection and correction value generation
RATIO_ABS	Multiplication and division
RATIO_INC	Multiplication and division based on exact increments
RATIO_INC_1	Multiplication and division based on exact increments
VGEN	Velocity setpoint
VGEN_A	Velocity setpoint with limited acceleration
VGEN_AJ	Velocity setpoint with limited acceleration and jerk

2.1.6.1 POS_J

2.1.6.1.1 Functions

2.1.6.1.1.1 8Bit

LREAL_TO_BYTE_CHECK (F)

Reserved for AMK internal use!

LREAL_TO_BYTE_CHECK_1 (F)

Reserved for AMK internal use!

LREAL_TO_SINT_CHECK (F)
Reserved for AMK internal use!

LREAL_TO_SINT_CHECK_1 (F)
Reserved for AMK internal use!

LREAL_TO_USINT_CHECK (F)
Reserved for AMK internal use!

LREAL_TO_USINT_CHECK_1 (F)
Reserved for AMK internal use!

2.1.6.1.1.2 16Bit

LREAL_TO_INT_CHECK (F)
Reserved for AMK internal use!

LREAL_TO_INT_CHECK_1 (F)
Reserved for AMK internal use!

LREAL_TO_UINT_CHECK (F)
Reserved for AMK internal use!

LREAL_TO_UINT_CHECK_1 (F)
Reserved for AMK internal use!

LREAL_TO_WORD_CHECK (F)
Reserved for AMK internal use!

LREAL_TO_WORD_CHECK_1 (F)
Reserved for AMK internal use!

2.1.6.1.1.3 32Bit

LREAL_TO_DINT_CHECK (F)
Reserved for AMK internal use!

LREAL_TO_DINT_CHECK_1 (F)
Reserved for AMK internal use!

LREAL_TO_DWORD_CHECK (F)
Reserved for AMK internal use!

LREAL_TO_DWORD_CHECK_1 (F)
Reserved for AMK internal use!

LREAL_TO_UDINT_CHECK (F)
Reserved for AMK internal use!

LREAL_TO_UDINT_CHECK_1 (F)
Reserved for AMK internal use!

2.1.6.1.1.4 64Bit

LREAL_TO_LINT_CHECK (F)
Reserved for AMK internal use!

LREAL_TO_LINT_CHECK_1 (F)
Reserved for AMK internal use!

LREAL_TO_LWORD_CHECK (F)
Reserved for AMK internal use!

LREAL_TO_LWORD_CHECK_1 (F)
Reserved for AMK internal use!

LREAL_TO_ULINT_CHECK (F)
Reserved for AMK internal use!

LREAL_TO_ULINT_CHECK_1 (F)
Reserved for AMK internal use!

2.1.6.1.1.5 bitCount (F)
Reserved for AMK internal use!

2.1.6.1.1.6 ceil_ulint (F)
Reserved for AMK internal use!

2.1.6.1.1.7 floor (F)
Reserved for AMK internal use!

2.1.6.1.1.8 fmod (F)
Reserved for AMK internal use!

2.1.6.1.1.9 fract (F)
Reserved for AMK internal use!

2.1.6.1.1.10 random (F)
Reserved for AMK internal use!

2.1.6.1.1.11 slope1CalcAdjustAmax2_Decelerate (F)
Reserved for AMK internal use!

2.1.6.1.1.12 slope1CalcAdjustVE_Accelerate (F)

Reserved for AMK internal use!

2.1.6.1.1.13 slope1CalcAdjustVE_Decelerate (F)

Reserved for AMK internal use!

2.1.6.1.1.14 slope1CalcAdjustVN (F)

Reserved for AMK internal use!

2.1.6.1.1.15 slope1CalcEstimateVN (F)

Reserved for AMK internal use!

2.1.6.1.1.16 slope1CalcExe (F)

Reserved for AMK internal use!

2.1.6.1.1.17 slope1CalcInit (F)

Reserved for AMK internal use!

2.1.6.1.1.18 slope1CalcInitP1P4 (F)

Reserved for AMK internal use!

2.1.6.1.1.19 slope1CalcInitP6P9 (F)

Reserved for AMK internal use!

2.1.6.1.1.20 slope1CalcPhase1 (F)

Reserved for AMK internal use!

2.1.6.1.1.21 slope1CalcPhase2 (F)

Reserved for AMK internal use!

2.1.6.1.1.22 slope1CalcPhase3 (F)

Reserved for AMK internal use!

2.1.6.1.1.23 slope1CalcPhase4 (F)

Reserved for AMK internal use!

2.1.6.1.1.24 slope1CalcPhase5 (F)

Reserved for AMK internal use!

2.1.6.1.1.25 slope1CalcPhase6 (F)

Reserved for AMK internal use!

2.1.6.1.1.26 slope1CalcPhase7 (F)

Reserved for AMK internal use!

2.1.6.1.1.27 slope1CalcPhase8 (F)

Reserved for AMK internal use!

2.1.6.1.1.28 slope1CalcPhase9 (F)

Reserved for AMK internal use!

2.1.6.1.1.29 slope1CalcStandardProfile (F)

Reserved for AMK internal use!

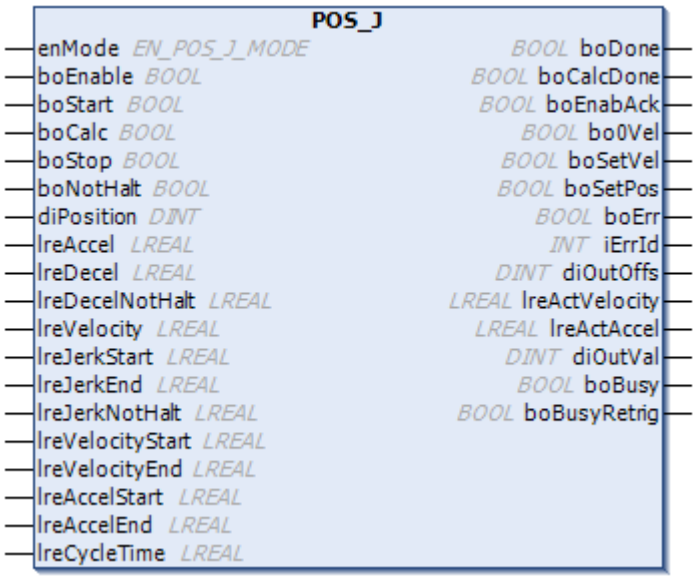
2.1.6.1.1.30 truncDint (F)

Reserved for AMK internal use!

2.1.6.1.2 POS_J (FB)

The 'POS_J' function block supports fast positioning controlled via binary inputs.
The movement sequence is defined with the position ('diPosition'), velocity ('lreVelocity'), acceleration ('lreAccel') and deceleration ('lreDecel'), jerk limitation ('lreJerkStart' / 'lreJerkEnd'), along with the selected operating mode ('enMode').
All parameters can be changed during the positioning process.

User interface



Input variables

Name	Type	Description
enMode	ENUM	EN_POS_J_MODE Selection mode of the selected movement sequence Default: POS_REL Range / Meaning: POS_J_REL: Relative positioning POS_J_REL_RETRIG_EXT: Relative positioning, retriggerable of a new position POS_J_REL_RETRIG: Relative positioning with extended fuctionality , retriggerable

Name	Type	Description
boEnable	BOOL	Enable signal: With a positive edge, the initialisation of the block starts. As long as 'boEnable' = TRUE, the block remains enabled and is processed by the PLC. In the state 'boEnable' = FALSE the block is no longer enabled and is thus no longer processed.
boStart	BOOL	With a positive edge, the execution of the block starts.
boCalc	BOOL	Calculate positioning
boStop	BOOL	With a positive edge, the execution of the block is aborted or completed.
boNotHalt	BOOL	EMERGENCY STOP: The setpoint of the velocity is decreased to zero along the emergency-stop ramp. Once initiated, an emergency stop cannot be aborted.
diPosition	DINT	Setpoint position Definition of the final position Unit: incr Default: 600000
lreAccel	LREAL	Acceleration for positioning and jog mode Range: $1{,}43\ 10^{-13} < \lvert\text{'lreAccel'}\rvert < 1{,}43\ 10^{+16}$ Unit: $incr/s^2$ Default: 1000000
lreDecel	LREAL	Deceleration for positioning and jog mode Range: $1{,}43\ 10^{-13} < \lvert\text{'lreDecel'}\rvert < 1{,}43\ 10^{+16}$ Unit: $incr/s^2$ Default: 1000000
lreDecelNotHalt	LREAL	Emergency stop ramp; deceleration in case of emergency stop. The value entered is valid for deceleration from actual speed to speed 0 Unit: $incr/s^2$
lreVelocity	LREAL	Setpoint velocity with which the final velocity is set Range: $1.43\ 10^{-13} < \lvert\text{'lreVelocity'}\rvert < 1.43\ 10^{+13}$ Unit: incr/s Default: 200000
lreJerkStart	LREAL	Jerk limitation (start) for positioning and jog mode Unit: $incr/s^3$ Default: 10000000
lreJerkEnd	LREAL	Jerk limitation (end) for positioning and jog mode Unit: $incr/s^3$ Default: 10000000
lreJerkNotHalt	LREAL	Emergency stop: Jerk limitation Unit: $incr/s^3$
lreVelocityStart	LREAL	Velocity (start) Unit: incr/s
lreVelocityEnd	LREAL	Velocity (end) Unit: incr/s

Name	Type	Description
lreAccelStart	LREAL	Acceleration (start) Unit: incr/s^2
lreAccelEnd	LREAL	Acceleration (end) Unit: incr/s^2
lreCycleTime	LREAL	Cycle time [s]

Output variables

Name	Type	Description
boDone	BOOL	Response that the function block has been completely executed.
boCalcDone	BOOL	Calculate positioning is done
boEnabAck	BOOL	Acknowledgement: Function block is initialised and enabled
bo0Vel	BOOL	When 'bo0Vel' is active, no setpoint is output.
boSetVel	BOOL	When 'boSetVel' is active, the target velocity has been reached.
boSetPos	BOOL	In retrigger mode the signal for a cycle becomes active when the retrigger position has been reached.
boErr	BOOL	The function block is in an error state FALSE: No error (permitted commanding or warning) TRUE: Error
iErrId	INT	Error identity number: Diagnostic number is output iErrID = 0: No error iErrID ≠ 0, boErr = TRUE: Error iErrID ≠ 0, boErr = FALSE: Warning
diOutOffs	DINT	Offset value before the retrigger is started; only in retrigger mode Unit: incr
lreActVelocity	LREAL	Output value velocity Unit: incr/s
lreActAccel	LREAL	Output value acceleration Unit: incr/s^2
diOutVal	DINT	Output value position Unit: incr
boBusy	BOOL	Execution message: This bit remains set as long as the block is being processed.
boBusyRetrig	BOOL	Execution message: This bit is set as long as the retrigger positioning is being processed.

2.1.6.2 CAM_CONT_1 (FB)

The 'CAM_CONT_1' function block is a cam switch with higher accuracy.

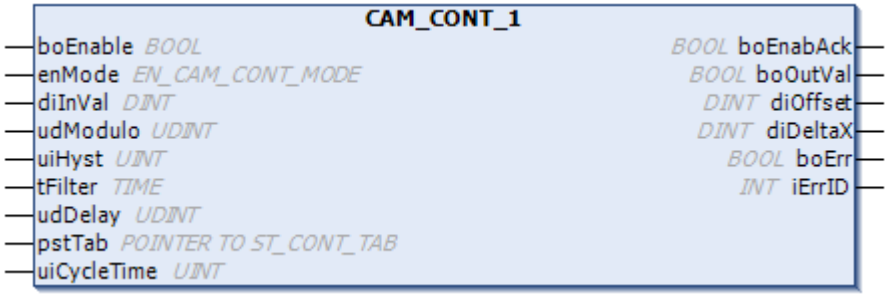
It controls a binary output variable (cam) as a function of the 'diInVal' input variable. The input variable can be a position value or a temporal value, for example.
The signal states of the binary output are defined with a cam table.
The switching points are defined based on the setting of the cam on / off position ('diOn' / 'diOff') in the cam table.
Up to 16 cams can be distributed on the track at will.
The switching points can be changed in the cam table "online", i.e. while the function block is activated ('boEnable' = TRUE).
Each block instance constitutes a cam track.

Extensions between CAM_CONT_1 and CAM_CONT:

- Additional output information:
  diOffset: DINT (offset of boOutVal in relation to the current scanning position, at time "k * T0")
  diDeltaX: DINT (Actual approximation of the speed Δx / T0 = "diInVal (k) - diInVal (k-1)")
  with: T0 = sampling time; K = actual time index
- Higher resolution of the delay time (udDelay)
  udDelay : UDINT [0.001 ms]

These extensions allow a more accurate time output together with, for example, the 'Timestamp' outputs of the I/O extension of the Ax PLC modules, the I/ option of the iSA module or the external EL2252 EtherCat modules of the cam switch (see AmkDevAccess Library: FB CAM_CONT_TS).

User interface



Input variables

Name	Type	Description
boEnable	BOOL	Enable signal: With a positive edge, the initialisation of the block starts. As long as 'boEnable' = TRUE, the block remains enabled and is processed by the PLC. In the state 'boEnable' = FALSE the block is no longer enabled and is thus no longer processed.
enMode	ENUM	EN_CAM_CONT_MODE Selection mode between incremental and absolute input evaluation Default: CAM_CONT_INC Range / Meaning: CAM_CONT_INC: Incremental input value evaluation CAM_CONT_ABS: Absolute input value evaluation
diInVal	DINT	Input value of the camshaft control (position)
udModulo	UDINT	Modulo value In mode 'enMode' = CAM_CONT_INC, this is the value at which cam table evaluation restarts at "0" Range: 0 … $+2^{31}$-1 Default: 20000

Name	Type	Description
uiHyst	UINT	Hysteresis value (H), applied to the on and off edges (X_{on}, X_{off}) of a cam signal Default: 0 (hysteresis not active) In conjunction with dead-time compensation, the hysteresis must be set higher than the dead-time compensation path Xdead Thus: $X_{dead} = T_{dead} * n * G / 60000$ where X_{dead} Dead-time compensation path [incr] T_{dead} Dead time [ms] n Speed [rpm] G Encoder resolution [incr/rev] In incremental input value evaluation mode ('enMode' = CAM_CONT_INC), the following must be true: $H < udModulo -(X_{off} - X_{on})$ for $X_{off} > X_{on}$ $H < X_{on} - X_{off})$ for $X_{off} < X_{on}$
tFilter	TIME	Filter time constant Attenuates the impact of changes in velocity in the context of dead-time compensation Default: t#1 ms
udDelay	UDINT	Dead-time constant To calculate the offset of the binary information depending on the current velocity in the context of dead-time compensation Resolution: t#0.001 ms **Default**: 0 (dead-time compensation not active)
pstTab	POINTER	POINTER TO ST_CONT_TAB Pointer to the cam table
uiCycleTime	UINT	Cycle time (usually corresponding to ID2 'SERCOS cycle time') Unit: 0.001 ms

Output variables

Name	Type	Description
boEnabAck	BOOL	Acknowledgement: Function block is initialised and enabled
boOutVal	BOOL	Cam Output signal
diOffset	DINT	Offset of 'boOutVal' to the actual sampling positio, at the time 'k*T_0' T_0 Sampling time [ms] k Acutal time index
diDeltaX	DINT	Actual approximation for the velocity $\Delta X/T_0$ = 'diInVal(k) - diInVal(k-1)' T_0 Sampling time [ms] k Acutal time index

<table>
<tr><th>Name</th><th>Type</th><th>Description</th></tr>
<tr><td>boErr</td><td>BOOL</td><td>The function block is in an error state
<table>
<tr><td>FALSE</td><td>No error (permitted commanding or warning)</td></tr>
<tr><td>TRUE</td><td>Error</td></tr>
</table></td></tr>
<tr><td>iErrID</td><td>INT</td><td>Error identity number: Diagnostic number is output
<table>
<tr><td colspan="2">iErrID = 0</td><td>No error</td></tr>
<tr><td>iErrID ≠ 0</td><td>boErr = TRUE</td><td>Error</td></tr>
<tr><td>iErrID ≠ 0</td><td>boErr = FALSE</td><td>Warning</td></tr>
</table>
<table>
<tr><th>Value</th><th>Meaning</th></tr>
<tr><td>1</td><td>Modulo value limited to maximum</td></tr>
<tr><td>2</td><td>Filter time constant set to 1</td></tr>
<tr><td>3</td><td>Filter time constant limited to maximum</td></tr>
<tr><td>4</td><td>Dead-time constant set to 0</td></tr>
<tr><td>5</td><td>Dead-time constant set to 1</td></tr>
<tr><td>6</td><td>Dead-time constant limited to maximum</td></tr>
<tr><td>7</td><td>Modulo value = 0 when mode CAM_CONT_INC</td></tr>
</table></td></tr>
</table>

Description

The camshaft control has the following properties:

- Incremental or absolute mode
- Filter in the context of dead-time compensation
- Dead-time compensation
- Hysteresis

Mode

- **Set incremental input value** ('enMode' = CAM_CONT_INC):
  The 'diInVal' input variable is processed as a 32-bit signed fixed-point number (32-bit integer value).
  In response to every call, the block generates the input value differences from two consecutive items of input information and adds these up to a positive 32-bit value. The internal counter works modulo; in other words, it counts up to a configurable final value 'udModulo' and then starts again at zero.
- **Set absolute input value** ('enMode' = CAM_CONT_ABS):
  The 'diInVal' input variable is processed as a 32-bit signed fixed-point number (32-bit integer value). Overshoot at the end of the travel range is limited.

Filter

To attenuate the impact of changes in velocity for dead-time compensation, multiple speed values are averaged. The 'tFilter' filter time constant determines the number of velocity values for which averaging is performed (number = 'tFilter' [ms]/uiCycleTime [ms]).

Dead-time compensation

For dead-time compensation the binary information is offset leading based on the current velocity. The 'tDelay' dead-time constant accounts for the time taken to calculate the offset.

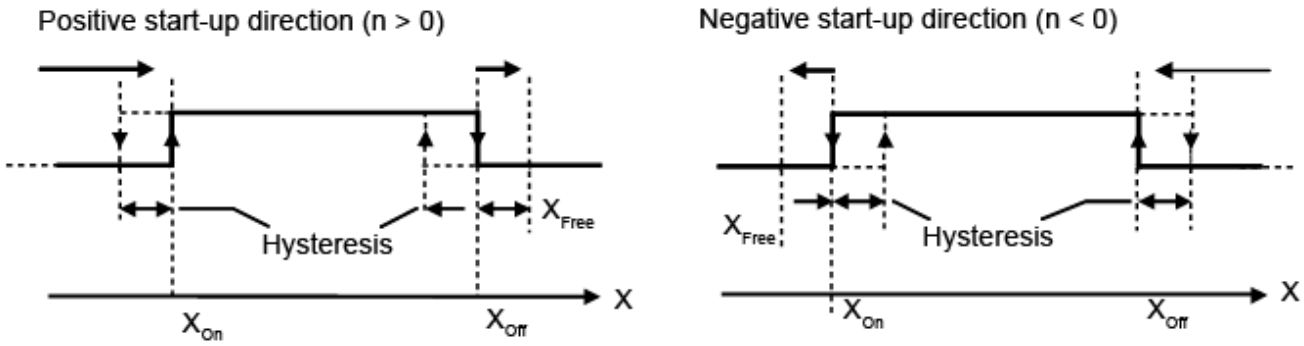
Hysteresis

The hysteresis ensures that the binary output always adopts a stable state, even if the input value of the block is moving around a rising or falling cam edge at the time.

The generation of the hysteresis (X_{on}, X_{off}) is illustrated in the figure below:

- Positive approach direction (n > 0; X increasing)
- Negative approach direction (n < 0; X decreasing)

Abbildung 5: CAM_CONT: Hysteresis generation



A "positive approach direction" results in the following behavior at the binary output (cam):

- Cam information "0" is output starting from a position $X < X_{on}$.
- Cam information "1" is output as of position $X \geq X_{on}$.
- Cam information "1" is retained during reverse rotation to position $X \geq X_{on}–H$.
  Cam information "0" is output during further reverse rotation to position $X < X_{on}–H$.
- Cam information "0" is output during forward rotation starting from position $X \geq X_{off}$.
  - Cam information "0" is retained in the event of reverse rotation to position $X \geq X_{off}–H$ before position $X = X_{free} = X_{off}+H$ is reached.
    Cam information "1" is output during further reverse rotation.
  - In the event of reverse rotation after position $X \geq X_{free}$ has been reached, the cam signal is generated according to the "negative approach direction".

A "negative approach direction" results in the following behavior:

- Cam information "0" is output starting from a position $X \geq X_{off}$.
- Cam information "1" is output as of position $X < X_{off}$.
- Cam information "1" is retained during forward rotation to position $X < X_{off}+H$.
  Cam information "0" is output during further forward rotation.
- Cam information "0" is output during reverse rotation starting from position $X < X_{on}$.
  - Cam information "0" is retained in the event of reverse rotation to position $X < X_{on}+H$ prior to overshooting position $X = X_{free} = X_{on}–H$.
    Cam information "1" is output during further forward rotation.
  - In the event of reverse rotation after position $X < X_{free}$ has been reached, the cam signal is generated according to the "positive approach direction".

Switchover between hysteresis generation of positive (negative) approach direction takes place once a cam has completed its rotation and position X_{free} has been reached or overshot.

2.1.6.3 CAM_PROF_1 (FB)

The 'CAM_PROF_1' function block provides a table-based function interpolator.

CAM_PROF_1 further outputs the following output value:

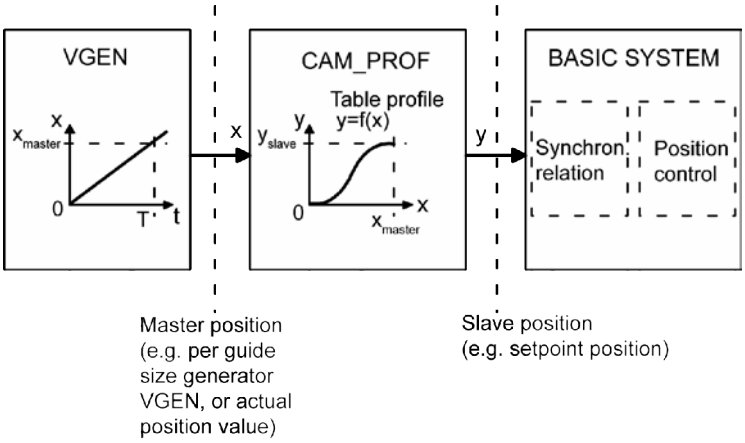
- 1. derivation, y '= f'(x), of the table-based function
- 2. derivation, y ' '= f "(x), of the table-based function
- Actual table input (X) position (master position) in the current cycle
- Actual table output (Y) position (slave position) in the current cycle

The function interpolator assigns an output value 'diOutVal' to an input value 'diInVal' based on tables.

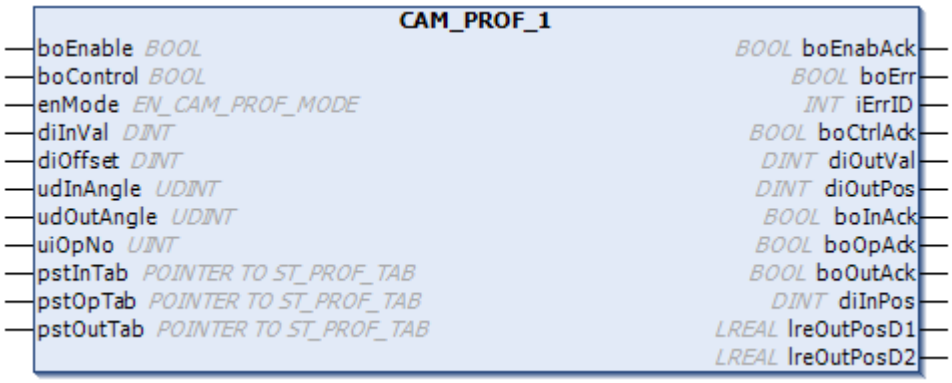
- For Y and XY tables, interpolation points are used to describe the assignment y = f(x).
  The interpolation between the points is linear.
- In the context of the XYVA format, the assignment y = f(x) is described section by section with 5th order polynomials.

Input value can be any internal or external value, e.g. the actual position of a master or a defined incremental number for each sampling time. The output value corresponds to the position setpoint of a slave drive, for example.

Abbildung 6: CAM_PROF_1: Principle of the table-based function interpolator



User interface



Input variables

Name	Type	Description
boEnable	BOOL	Enable signal: With a positive edge, the initialisation of the block starts. As long as 'boEnable' = TRUE, the block remains enabled and is processed by the PLC. In the state 'boEnable' = FALSE the block is no longer enabled and is thus no longer processed.
boControl	BOOL	Start / Stop Table control based on table interpolator 'enMode'

<table>
<tr><th>Name</th><th>Type</th><th>Description</th></tr>
<tr><td>enMode</td><td>ENUM</td><td>EN_CAM_PROF_MODE
Selection mode of the required movement sequence
<table>
<tr><td>Default</td><td>CAM_PROF_CONT</td></tr>
</table>
<table>
<tr><th>Range</th><th>Meaning</th></tr>
<tr><td>CAM_PROF_
CONT</td><td>Continuous movement
without taking 'boControl' into account</td></tr>
<tr><td>CAM_PROF_
PIN_POUT</td><td>Input / output mode
with control of table transition by 'boControl'</td></tr>
<tr><td>CAM_PROF_
START_STOP</td><td>Start / stop mode
with control of table transition by 'boControl' and automatic stop at end of table</td></tr>
<tr><td>CAM_PROF_IM_
START_STOP</td><td>Immediate start / stop mode
with control of table transition by 'boControl' and automatic stop at end of table</td></tr>
<tr><td>CAM_PROF_
CONT_R</td><td>Continuous movement
without taking 'boControl' into account
the movement sequence starts relative to the current position X_{act}/Y_{act}</td></tr>
<tr><td>CAM_PROF_
PIN_POUT_R</td><td>Input / output mode
with control of table transition by 'boControl'
the movement sequence starts relative to the current position X_{act}/Y_{act}</td></tr>
<tr><td>CAM_PROF_
START_STOP_R</td><td>Start / stop mode
with control of table transition by 'boControl' and automatic stop at end of table
the movement sequence starts relative to the current position X_{act}/Y_{act}</td></tr>
<tr><td>CAM_PROF_IM_
START_STOP_R</td><td>Immediate start / stop mode
with control of table transition by 'boControl' and automatic stop at end of table
the movement sequence starts relative to the current position X_{act}/Y_{act}</td></tr>
</table></td></tr>
<tr><td>diInVal</td><td>DINT</td><td>x Input value (table x axis)
<table>
<tr><td>Unit</td><td>incr</td></tr>
</table></td></tr>
<tr><td>diOffset</td><td>DINT</td><td>Offset of the counter value to the homing pulse
<table>
<tr><td>Unit</td><td>Incr</td></tr>
</table></td></tr>
<tr><td>udInAngle</td><td>UDINT</td><td>Phasing in angle
Maximum permissible position on the x axis up to which phasing in is permitted
(relevant for 'enMode' = CAM_PROF_PIN_POUT / CAM_PROF_START_STOP / CAM_PROF_PIN_POUT_R / CAM_PROF_START_STOP_R)
<table>
<tr><td>Unit</td><td>incr</td></tr>
<tr><td>Default</td><td>1000</td></tr>
</table></td></tr>
<tr><td>udOutAngle</td><td>UDINT</td><td>Phasing out angle
Maximum permissible position on the x axis up to which phasing out is permitted
(relevant for 'enMode' = CAM_PROF_PIN_POUT / CAM_PROF_PIN_POUT_R)
<table>
<tr><td>Unit</td><td>incr</td></tr>
<tr><td>Default</td><td>1000</td></tr>
</table></td></tr>
</table>

Name	Type	Description
uiOpNo	UINT	Number of operating table cycles to be processed [1] (relevant for 'enMode' = CAM_PROF_START_STOP / CAM_PROF_IM_START_STOP / CAM_PROF_START_STOP_R / CAM_PROF_IM_START_STOP_R) Default: 3
pstInTab	POINTER	POINTER TO ST_PROF_TAB Pointer to phasing-in table. (for all 'enMode' except CAM_PROF_CONT / CAM_PROF_CONT_R) Range: pointer to 0 — Meaning: Table not necessary Range: pointer to ST_PROF_YTAB — Meaning: Y table Range: pointer to ST_PROF_XYTAB — Meaning: XY table Range: pointer to ST_PROF_XYVATAB — Meaning: XYVA table Range: pointer to ST_PROF_TAB — Meaning: either Y, XY, or XYVA table
pstOpTab	POINTER	POINTER TO ST_PROF_TAB Reference to operating table (table-supported cam) Range: pointer to ST_PROF_YTAB — Meaning: Y table Range: pointer to ST_PROF_XYTAB — Meaning: XY table Range: pointer to ST_PROF_XYVATAB — Meaning: XYVA table Range: pointer to ST_PROF_TAB — Meaning: either Y, XY, or XYVA table
pstOutTab	POINTER	POINTER TO ST_PROF_TAB Pointer to phasing-out table. (for all 'enMode' except CAM_PROF_CONT / CAM_PROF_CONT_R) Range: pointer to 0 — Meaning: Table not necessary Range: pointer to ST_PROF_YTAB — Meaning: Y table Range: pointer to ST_PROF_XYTAB — Meaning: XY table Range: pointer to ST_PROF_XYVATAB — Meaning: XYVA table Range: pointer to ST_PROF_TAB — Meaning: either Y, XY, or XYVA table

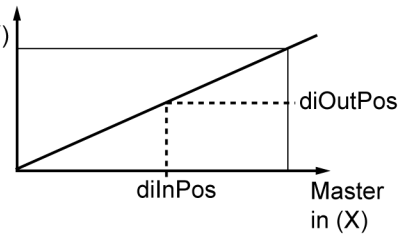
Output variables

Name	Type	Description
boEnabAck	BOOL	Acknowledgement: Function block is initialised and enabled

<table>
<tr><th>Name</th><th>Type</th><th>Description</th></tr>
<tr><td>boErr</td><td>BOOL</td><td>The function block is in an error state
<table><tr><td>FALSE</td><td>No error (permitted commanding or warning)</td></tr><tr><td>TRUE</td><td>Error</td></tr></table></td></tr>
<tr><td>iErrID</td><td>INT</td><td>Error identity number: Diagnostic number is output
<table><tr><td colspan="2">iErrID = 0</td><td>No error</td></tr><tr><td>iErrID ≠ 0</td><td>boErr = TRUE</td><td>Error</td></tr><tr><td>iErrID ≠ 0</td><td>boErr = FALSE</td><td>Warning</td></tr></table>Warning:<table><tr><th>Value</th><th>Meaning</th></tr><tr><td>1</td><td>Offset too high</td></tr><tr><td>2</td><td>Input angle too high</td></tr><tr><td>3</td><td>Output angle too high</td></tr></table>Error:<table><tr><th>Value</th><th>Meaning</th></tr><tr><td>1</td><td>Illegal mode</td></tr><tr><td>2</td><td>Phasing in table required</td></tr><tr><td>3</td><td>Operating table required</td></tr><tr><td>4</td><td>Phasing out table required</td></tr><tr><td>5</td><td>Illegal element number in phasing in table</td></tr><tr><td>6</td><td>Illegal element number in operating table</td></tr><tr><td>7</td><td>Illegal element number in phasing out table</td></tr><tr><td>8</td><td>Illegal number of master increments in phasing in table</td></tr><tr><td>9</td><td>Illegal number of master increments in operating table</td></tr><tr><td>10</td><td>Illegal number of master increments in phasing out table</td></tr><tr><td>11</td><td>Illegal number of operating tables</td></tr><tr><td>12</td><td>Illegal x-value sequence in phasing in table</td></tr><tr><td>13</td><td>Illegal x-value sequence in operating table</td></tr><tr><td>14</td><td>Illegal x-value sequence in phasing out table</td></tr><tr><td>15</td><td>Illegal phasing in table type</td></tr><tr><td>16</td><td>Illegal operating table type</td></tr><tr><td>17</td><td>Illegal phasing out table type</td></tr><tr><td>18</td><td>Illegal starting value for phasing in table (≠ 0)</td></tr><tr><td>19</td><td>Illegal starting value for operating table (≠ 0)</td></tr><tr><td>20</td><td>Illegal starting value for phasing out table (≠ 0)</td></tr></table></td></tr>
<tr><td>boCtrlAck</td><td>BOOL</td><td>Acknowledgement of table control
<table><tr><td>FALSE</td><td>Output of output values inactive</td></tr><tr><td>TRUE</td><td>Output of output values active, 'boControl' applies</td></tr></table></td></tr>
<tr><td>diOutVal</td><td>DINT</td><td>y Output value as sum of output increments</td></tr>
<tr><td>diOutPos [1)]</td><td>DINT</td><td>y Output position
<table><tr><td>Unit</td><td>Incr</td></tr></table>Y-table position to display the table ordinate: $0 \le y \le y_{max}$ (y_{max} = table Y-end position); with y = f(x).</td></tr>
<tr><td>boInAck</td><td>BOOL</td><td>Acknowledge end of phasing in table
Pulse at end of phasing in table; 'boInAck' = TRUE for 2 sampling time points</td></tr>
<tr><td>boOpAck</td><td>BOOL</td><td>Acknowledge end of operating table
Pulse at end of operating table; 'boOpAck' = TRUE for 2 sampling time points</td></tr>
</table>

Name	Type	Description
boOutAck	BOOL	Acknowledge end of phasing out table Pulse at end of phasing out table; 'boOutAck' = TRUE for 2 sampling time points
diInPos 1)	DINT	x Input position Unit: Incr X-table position, to display the table abscissa: $0 \le x \le x_{max}$ (x_{max} = table X-end position); with y = f(x).
lreOutPosD1	LREAL	1. derivation, y '= f'(x), of the table-based function
lreOutPosD2	DINT	2. derivation, y ' '= f ''(x), of the table-based function

1) Slave out (Y)



2.1.6.3.1 Mathematical consideration of functional relationships

For a mathematical consideration of the functional relationships, the following variables are important:

Variable (input / output)	Mathematical identifier	Meaning	Dimension
diInPos (I)		Abscissa (x-axis)	[x]
diOutPos (O)		Ordinate (y-axis)	[y]
lreOutPosD1 (O)		1. derivative (after x)	[y] / [x]
lreOutPosD2 (O)		2. derivative (after x)	[y] / $[x]^2$

The functional relationship 'y = f (x)' is stored in the form of a table:

- As a point sequence (Y- / XY- table)
- As a spline parameter sequence (XYVA- table)

If, for example, a path-to-path relationship is given with 'y = f (x)' (e.g. if the position 'y' of the slave drive is described as a function of the position 'x' of the master drive), the following time dependencies apply:

Input variables (x-axis):

Position:

Velocity:

Acceleration:

Output variables (y-axis):

Position:

Velocity:

Acceleration:

For a constant velocity the following is valid:

Input variables (x-axis):

Output variables (y-axis):

That means:

- v_y is proportional to the first derivative f '(x)
- a_y is proportional to the second derivative f "(x)

For the specific case of a constant velocity in the x-axis (e.g. the master drive moves at a constant velocity v_0) the values of the output variables are:

- lreOutPosD1 is proportional to the velocity of the y-axis (slave axis) and
- lreOutPosD2 is proportional to the acceleration of the y-axis (slave axis)

2.1.6.3.2 Technical realization of derivations

With the CAM_PROF_1 the derivations for **Y- / XY-tables** can be calculated in sections and approximately (e. g. by numerical differentiation)

A requirement for the numerical differentiation of the function sequences (Y- / XY-tables) described as a point sequence is:

- The point sequences are continuously differentiable at least twice
- The resolution of the point sequences is sufficiently high

With the CAM_PROF_1 the derivations for **XYVA-table**s, with the knowledge of the corresponding function 'y = f (x)', but in the form of a mathematical derivation of the derived functions

The advantage of the closed calculation of the derivatives of the polynomial described functional sequences (XYVA-tables) lies in the exact calculation of this (for any position of the function sequences).

2.1.6.3.2.1 Y- / XY-tables

For 'Y / XY tables', with linear interpolation, the following two different cases can be distinguished:

For a 'Y-table' yTab (of the type ST_PROF_YTAB), for each segment 'k' (with k = 1 to N):

1. Derivation (lreOutPosD1)

2. Derivation (lreOutPosD2)

with

For an 'XY-table' xyTab (of type ST_PROF_XYTAB), for each segment 'k' (with k = 1 to N):

1. Derivation (lreOutPosD1)

In the last point of segment N (theoretical):

2. Derivation (lreOutPosD2)

with:

and

2.1.6.3.2.2 'XYVA-tables'

For 'XYVA tables', with section-defined polynomials of the fifth order, the following definitions are given.

For an 'XYVA-table' xyvaTab (of type ST_PROF_XYVATAB) for each segment 'k' (with k = 1 to xyvaTab.uiNoElement):

Basic function (diOutPos)

with

and

1. Derivation (lreOutPosD1)

2. Derivation (lreOutPosD2)

2.1.6.3.3 Table types

The 'CAM_PROF_1' function interpolator can process three types of table

- Y tables – with equidistant X axis resolution
- XY tables – with freely definable X axis resolution
- XYVA tables – for section-by-section definition of the curve with 5th order polynomials.

The reference to the table structure is made with the pointer variables 'pstInTab', 'pstOpTab', and 'pstOutTab'.
The transition between the tables is controlled with the binary input signal 'boControl'.
The assignment of the table can be changed by exchanging the address (this corresponds to table switchover).

2.1.6.3.3.1 Y table

Y tables are based on the 'ST_PROF_YTAB' structure. They support the definition of a function y=f(x) with equidistant x axis resolution.

In the table structure, only the y values of the function y=f(x) are described.

The corresponding x values are generated in the 'CAM_PROF_1' block with 'uiNoElement'+1 equidistant points.

Thus:

$$A = \frac{\text{udMasterInc}}{\text{uiNoElement}}$$

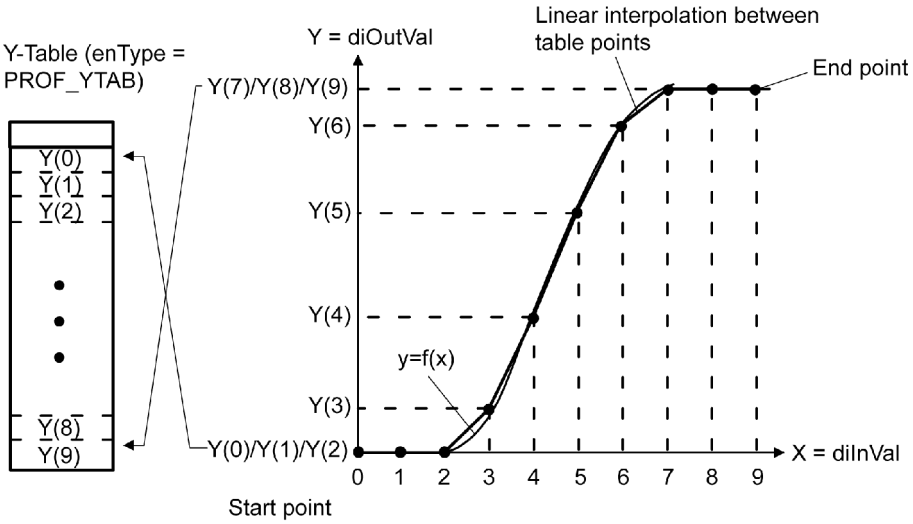
where A: equidistant spacing

Tabelle 1: CAM_PROF_1: Table structure of the Y table

Header information:	**enType = PROF_YTAB**	**uiNoElement**
	udMasterInc	
Interpolation points:	diY[0] = 0	
	diY[1]	
	...	
	diY[360]	

Limiting to 'uiNoElement' = 360 is negated by mode 'enType' = PROF_YTAB_NL

Abbildung 7: CAM_PROF_1: Principle of the function interpolator with Y table



Advantages of the Y table

- High information density, as only the y values are saved
- Transparent presentation

Disadvantages of the Y table

- Points also have to be defined in sections with linear function characteristic

2.1.6.3.3.2 XY table

XY tables are based on the 'ST_PROF_XYTAB' structure. They support the definition of a function y=f(x) with any x axis resolution.

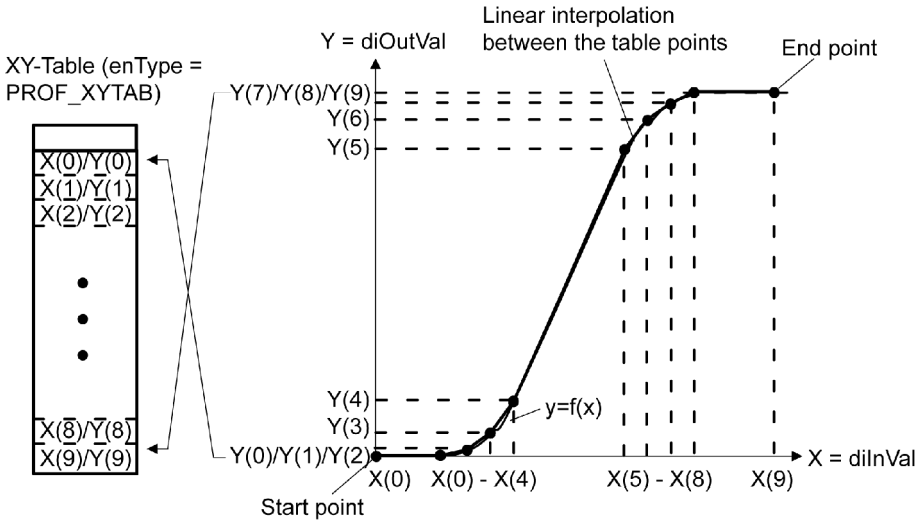
In the table structure, the x and y values of the function y=f(x) are described by 'uiNoElement'+1 pair of values.

Tabelle 2: CAM_PROF_1: Table structure of the XY table

Header information:	enType = PROF_XYTAB	uiNoElement
	udMasterInc (not used)	
Interpolation points:	stElement[0].diX = 0	
	stElement[0].diY = 0	
	stElement[1].diX	
	stElement[1].diY	
	...	
	stElement[180].diX	
	stElement[180].diY	

Limiting to 'uiNoElement' = 180 is negated by mode 'enType' = PROF_XYTAB_NL

Abbildung 8: CAM_PROF_1: Principle of the function interpolator with XY table



Advantages of the XY table

- Since the point spacing can be freely defined, the point density can be adapted to the curvature of the curve
- Only the start and end points need to be specified to describe linear sections

Disadvantages of the XY table

- Fewer points, since both x and y values must be specified
- Lack of transparency in presentation based on pairs of points

2.1.6.3.3.3 XYVA table

XY tables are based on the 'ST_PROF_XYVATAB' structure. They support the definition of a function y=f(x) based on 'uiNoElement' 5th order polynomials.

A pointer of the table structure references the dX, dY, dV, and dA values which describe the function y=f(x) section by section.

dX and dY describe the interpolation point of the function, dV describes the velocity (value of first derivation), and dA describes the acceleration (value of second derivation) in this point.

The table interpolation point structure 'stCam_A': ARRAY [0...3] OF SMC_CAMXYVA and the structure 'stCam': MC_CAM_REF are generated automatically based on the cam disk editor under CODESYS.

The 'CAMXYVA_TO_PROF' function block converts the CODESYS structures into the AMK structure 'ST_PROF_XYVATAB'. (

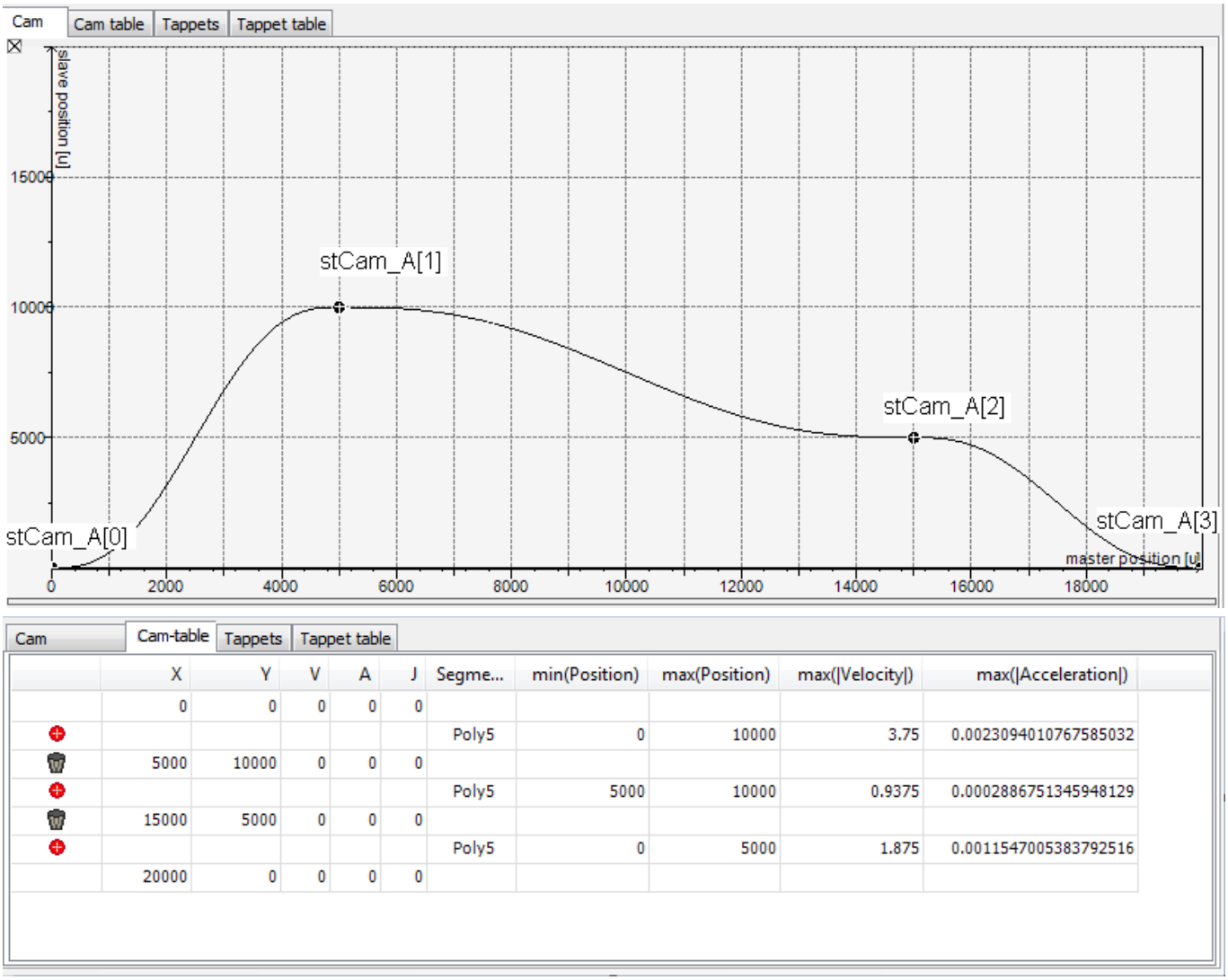
Tabelle 3: CAM_PROF_1: Table structure of the XYVA table

Header information:	**enType = PROF_XYVATAB**	**uiNoElement**
	udMasterInc (not used)	
Pointer to interpolation point table:	pstCamXYVA	

Table structure of the interpolation point table

XYVA interpolation point table:
stCam[0].dX = 0
stCam[0].dY = 0
stCam[0].dV
stCam[0].dA
...
stCam[N].dX
stCam[N].dY
stCam[N].dV
stCam[N].dA

Abbildung 9: CAM_PROF_1: Principle of the function interpolator with XYVA tables



Cam | Cam-table | Tappets | Tappet table

	X	Y	V	A	J	Segme...	min(Position)	max(Position)	max(|Velocity|)	max(|Acceleration|)
	0	0	0	0	0					
						Poly5	0	10000	3.75	0.0023094010767585032
	5000	10000	0	0	0					
						Poly5	5000	10000	0.9375	0.0002886751345948129
	15000	5000	0	0	0					
						Poly5	0	5000	1.875	0.0011547005383792516
	20000	0	0	0	0					

The figure shows the functional principle of the XYVA interpolator based on 5th order polynomials defined section by section. The example includes N = 3 polynomials with 4 sampling points stCam_A[0] ... stCam_A[3].

CODESYS structures 'stCAM' and 'stCAM_A'

```
stCAM: MC_CAM_REF   :=   (nElements = 4,
                          byType = 3,
                          xStart = 0.000000,
                          xEnd = 20000.000000,
                          nTappets = 0,
                          strCamName = "stCAM");

stCAM_A: ARRAY[0…3] OF SMC_CAMXYVA  :=   (dX = 0.000000, dY = 0.000000, dV = 0.000000, dA = 0.000000),
                                          (dX = 5000.000000, dY = 10000.000000, dV = 0.000000, dA = 0.000000),
                                          (dX = 15000.000000, dY = 5000.000000, dV = 0.000000, dA = 0.000000),
                                          (dX = 20000.000000, dY = 0.000000, dV = 0.000000, dA = 0.000000),
```

Advantages of the XYVA table

- Freely definable points with specification of 1st and 2nd derivation at start and end of section

Disadvantages of the XYVA table

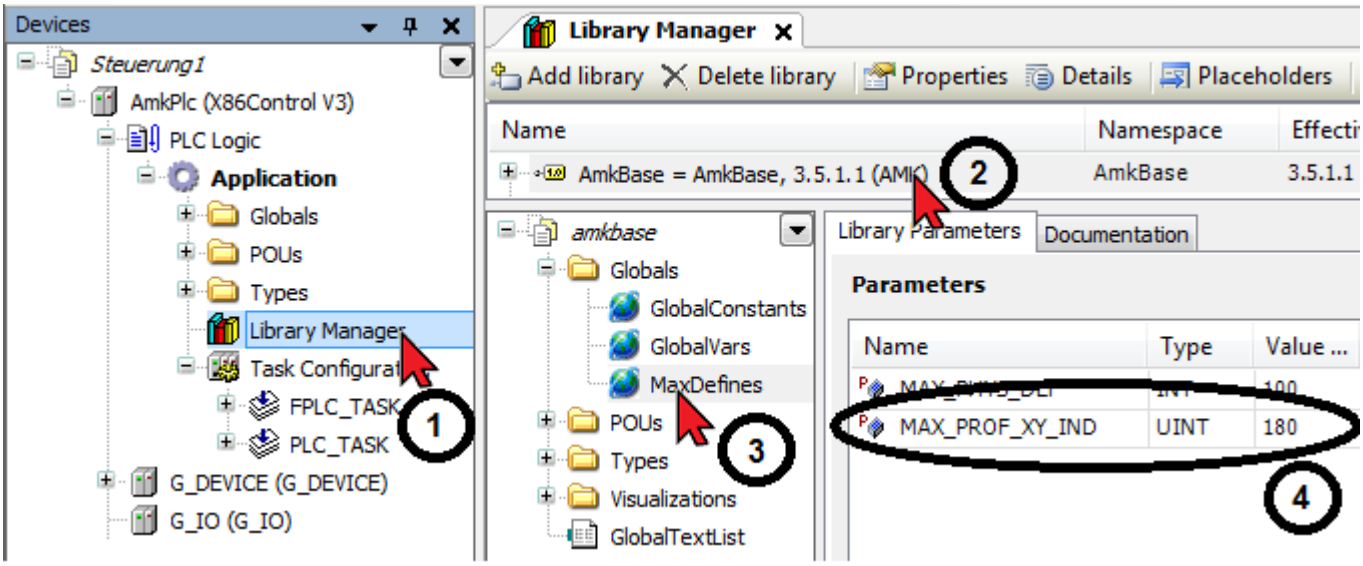
- Increased processing overhead for online determination of 5th order polynomials

2.1.6.3.4 Number of table interpolation points

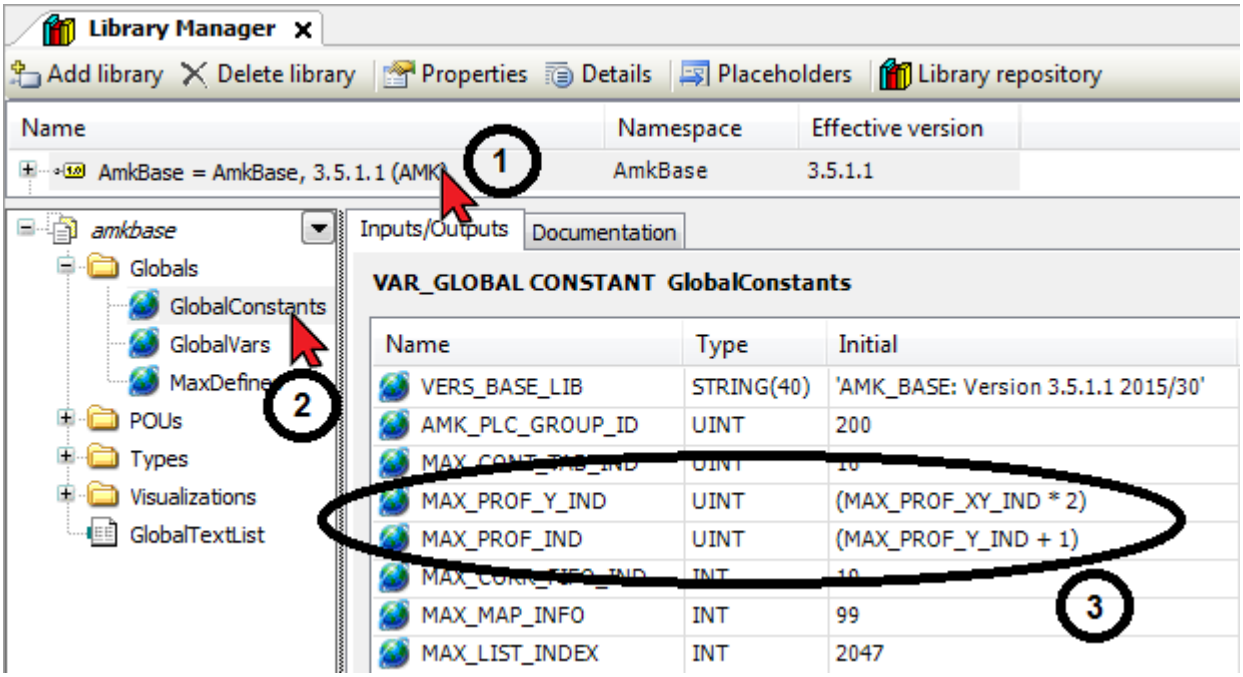
The number of table interpolation points are increased by the parameter MAX_PROF_XY_IND. (Library AmkBase → folder Globals → MaxDefines)

Default values:
181 XY-table sections (0 ... 180)
361 Y-table sections (0 ... 360)

Change the number of table interpolation points with CODESYS V3



The sizes MAX_PROF_Y_IND and MAX_PROF_IND are determined automatically.



If the maximum number of table interpolation points are increased,
using the parameter 'MAX_PROF_XY_IND', further the table types with the extension NL must be used.
_NL stands for 'not limited'

Default values: **181 XY table sections** **361 Y table sections**	**Advanced:** **> 181 XY table sections** **> 361 Y table sections**
'PROF_YTAB' 'PROF_XYTAB'	'PROF_YTAB**_NL**' 'PROF_XYTAB**_NL**'

2.1.6.3.5 Operating modes

The following operating modes are defined according to the mode selected in 'enMode':

- CAM_PROF_CONT, CAM_PROF_CONT_R
  Table-based continuous movement of a drive
- CAM_PROF_PIN_POUT, CAM_PROF_PIN_POUT_R
  Table-based continuous movement of a drive, which the option of input / output via special input / phasing out tables
- CAM_PROF_START_STOP, CAM_PROF_START_STOP_R
  Table-based movement of a drive after starting at the zero point of the table and with automatic stop at the end of the table
- CAM_PROF_IM_START_STOP, CAM_PROF_IM_START_STOP_R
  Immediate table-based movement of a drive after starting, with automatic stop at the end of the table

The relative movement modes ending '_R' are equivalent to the corresponding absolute modes but support the inclusion of a movement sequence in a table. This means that the interpolation in the table starts relative to the current master position (x = diInVal) and slave position (y = diOutVal).

The 'diOffset' variable sets the current master position x.
Once 'boEnable' has been activated, the assigned slave position y to which the slave axis is being moved can be read at 'diOutPos'.
The axes adopt the start position (x,y). The movement coupled via the table is resumed at this point.

Continuous movement

In 'CAM_PROF_CONT' mode, table interpolation is activated on a positive edge at 'boEnable'.
From this point on, incremental changes at the input variable 'diInVal' (x) generate incremental changes at the output variable 'diOutVal' (y).

The assignment y = f(x) is made according to the table definitions.
Only the 'pstOpTab' operating table is required for this mode.
The control signal 'boControl' is not evaluated.

2.1.6.3.5.1 Phasing in / phasing out

In 'CAM_PROF_PIN_POUT' mode, a positive edge at 'boEnable' activates input increment acquisition.
From this point in time, incremental changes at the input variable 'dInVal' generate changes in the master reference point (x).

The input, working, and phasing out tables assigned with 'pstInTab', 'pstOpTab', and 'pstOutTab' are evaluated. The operating table is a minimum requirement.
The control signal 'boControl' is evaluated to control input and output behavior.
The assignment y = f(x) is made according to the corresponding table definition.

A positive edge at 'boControl' and 'x ≤ udInAngle' activates the table interpolator.
Incremental changes according to the phasing in table are output at the output variable 'diOutVal' (y).
The transition to the operating table occurs at the end of the phasing in table. The operating table is processed until a negative edge occurs on 'boControl' and 'x ≤ udOutAngle'.
After this, there is a transition to the phasing out table with automatic stop of interpolation at the end of the phasing out table.

Incremental changes at the input variable 'dInVal' continue to generate changes in the master reference point (x). In other words, the reference to the master axis is retained.

If there is no phasing in table or no phasing out table ('pstInTab' = 0 or 'pstOutTab' = 0), the operating table is used instead.

2.1.6.3.5.2 Start with auto-stop

In 'CAM_PROF_START_STOP' mode, a positive edge at 'boEnable' activates input increment acquisition.
From this point in time, incremental changes at the input variable 'dInVal' generate changes in the master reference point (x).

The input, working, and phasing out tables assigned with 'pstInTab', 'pstOpTab', and 'pstOutTab' are evaluated.
The operating table is a minimum requirement.
The control signal 'boControl' is evaluated to control startup behavior.
The assignment y = f(x) is made according to the corresponding table definition.

A positive edge at 'boControl' and 'x ≤ udInAngle' activates the table interpolator.
Incremental changes according to the phasing in table are output at the output variable 'diOutVal' (y).
The transition to the operating table occurs at the end of the phasing in table.
The operating table is processed n times (with n = 'uiOpNo').
After this, there is a transition to the phasing out table with automatic stop of interpolation at the end of the phasing out table.

Incremental changes at the input variable 'dInVal' continue to generate changes in the master reference point (x). The reference to the master axis is retained.

If there is no phasing in table ('pstInTab' = 0), the process starts directly with the operating table.
If there is no phasing out table ('pstOutTab' = 0), the process stops at the end of the last pass of the operating table!

2.1.6.3.5.3 Immediate start with auto-stop

In 'CAM_PROF_IM_ START_STOP' mode, table interpolation is activated on a positive edge at 'boEnable'.
From this point on, incremental changes at the input variable 'diInVal' (x) generate incremental changes at the output variable 'diOutVal' (y).

The input, working, and phasing out tables assigned with 'pstInTab', 'pstOpTab', and 'pstOutTab' are evaluated.
The operating table is a minimum requirement.
The control signal 'boEnable' is evaluated to control startup behavior.
The assignment y = f(x) is made according to the corresponding table definition.

There is no direct reference to the master axis. Instead, a reference is generated at the time of the positive edge at 'boControl'.
After this, travel continues with reference to the master increment inputs.

Incremental changes according to the phasing in table are output at the output variable 'diOutVal' (y).
The transition to the operating table occurs at the end of the phasing in table.
The operating table is processed n times (with n = 'uiOpNo').
After this, there is a transition to the phasing out table with automatic stop of interpolation at the end of the phasing out table.

Incremental changes at the input variable 'dInVal' continue to generate changes in the master reference point (x). The reference to the master axis is retained.

If there is no phasing in table ('pstInTab' = 0), the process starts directly with the operating table.
If there is no phasing out table ('pstOutTab' = 0), the process stops at the end of the last pass of the operating table!

2.1.6.3.6 Online table switchover

The 'CAM_PROF_1' function block supports switchover between tables by changing the value of the pointer variables 'pstInTab', 'pstOpTab', and 'pstOutTab'. This can be done online while the table interpolator is active:

- Synchronous table changeover, switchover at table zero point
- Non-synchronous table changeover, switchover away from table zero point.

It is possible to switch between entirely different tables:

- The interpolation points and their number ('uiNoElement') can differ.
- The table type ('enType') can differ.
- The x axis resolution ('udMasterInc' or 'stElement[uiNoElement].diX') can differ.

If the tables have different x axis resolutions, the current master position must be converted to the assigned position of the new table. Thus:

$$X_{new} = X_{curr} \times \frac{X_{max\_new}}{X_{max\_curr}}$$

X_{curr} x master position in the current table

X_{max_curr} x axis resolution of the current table

X_{new} x master position in the new table

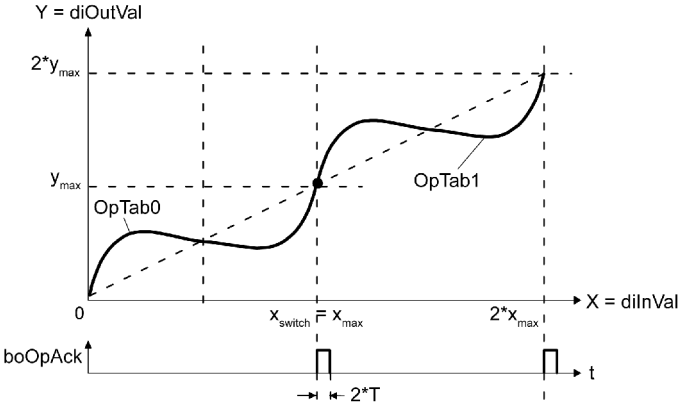
X_{max_new} x axis resolution of the new table

Since the 'CAM_PROF_1' block works with table pointers ('pstInTab', 'pstOpTab', 'pstOutTab'), these pointers must always reference a correct table structure.
Block processing is aborted if the corresponding pointer variable is found to have a value of 0 (no table) when attempting to access a table!

2.1.6.3.6.1 Synchronous table changeover

The binary output signals 'boInAck', 'boOpAck', and 'boOutAck' are provided for synchronous table changeover.
These signals generate a TRUE signal at the end of the corresponding table; this signal remains pending for two sampling cycles.
Table changeover is synchronous if it takes place during this time, i.e. at the zero point of the table.
As shown in the figure below, the process is similar to that when changing between phasing in and operating tables (or between operating and phasing out tables).
If the maximum x and y final values of the tables are identical, the reference to the master is retained.

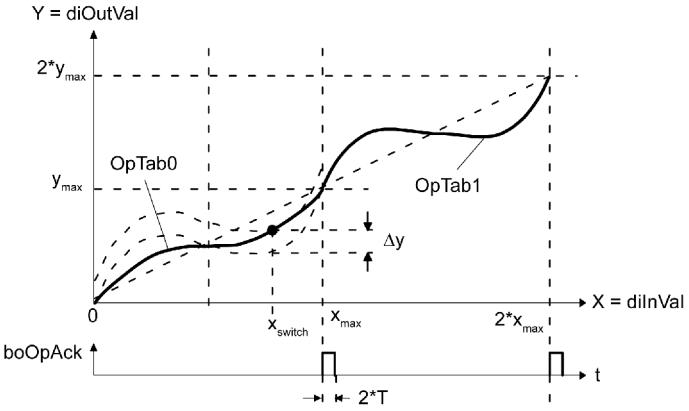
Abbildung 10: CAM_PROF_1: Synchronous table switchover



2.1.6.3.6.2 Non-synchronous table changeover

The binary output signals 'boInAck', 'boOpAck', and 'boOutAck' do not have to be evaluated for non-synchronous table changeover.
Table changeover is non-synchronous if it takes place at a point in time at which the assigned binary signal is set to FALSE, i.e. not at the zero point of the table.
As shown in the figure below, the y point on the table being phased in is shifted to the current y point of the previous table (by Δy).
A defined reference to the master cannot be retained even if the maximum x and y final values of the tables are identical.

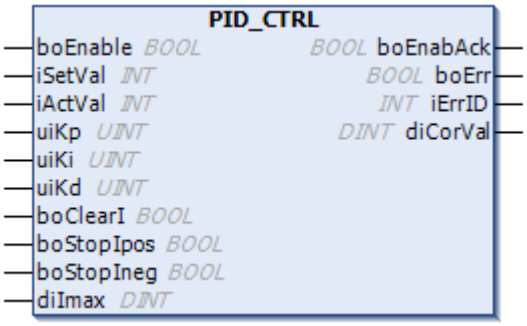
Abbildung 11: CAM_PROF_1: Non-synchronous table changeover



2.1.6.4 PID_CTRL (FB)

The 'PID_CTRL' (PID controller) function block supports configurable drive control of internal and external variables.

User interface



Input variables

Name	Type	Description
boEnable	BOOL	Enable signal: With a positive edge, the initialisation of the block starts. As long as 'boEnable' = TRUE, the block remains enabled and is processed by the PLC. In the state 'boEnable' = FALSE the block is no longer enabled and is thus no longer processed.
iSetVal	INT	Setpoint
iActVal	INT	Actual value
uiKp	UINT	Proportional gain (P) of the PID controller Unit: 1/256
uiKi	UINT	Integration gain (I) of the PID controller. Unit: 1/256
uiKd	UINT	Differential gain (D) of the PID controller Unit: 1/256
boClearI	BOOL	Delete accumulated integral action
boStopIpos	BOOL	Integrator stop on positive increase in integral action
boStopIneg	BOOL	Integrator stop on negative increase in integral action

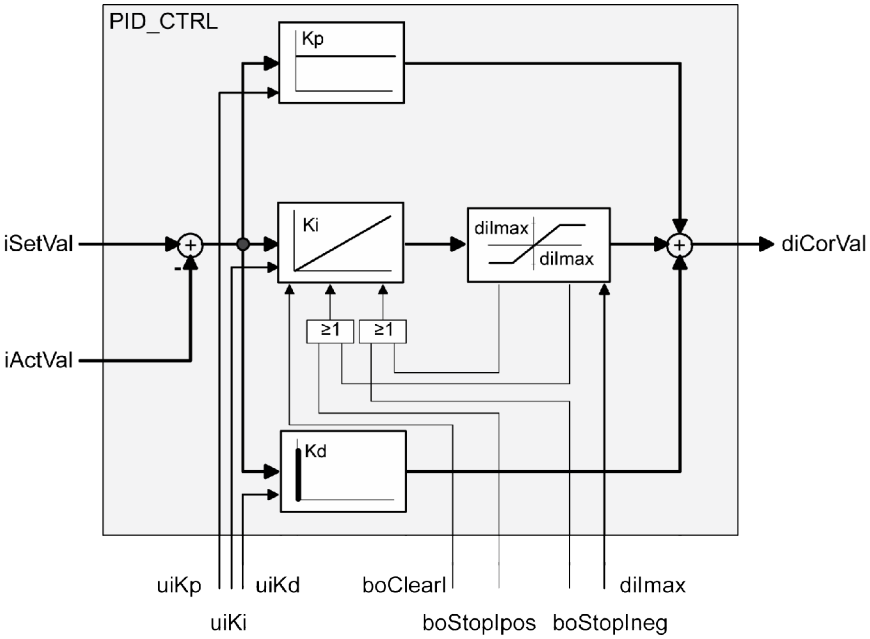
Name	Type	Description
diImax	DINT	Maximum permissible integral action Range: 0 ... 2^{31}-1 Unit: 1/256 Default: 256000

Output variables

Name	Type	Description
boEnabAck	BOOL	Acknowledgement: Function block is initialised and enabled
boErr	BOOL	The function block is in an error state FALSE: No error (permitted commanding or warning) TRUE: Error
iErrID	INT	Error identity number: Diagnostic number is output iErrID = 0: No error iErrID ≠ 0, boErr = TRUE: Error iErrID ≠ 0, boErr = FALSE: Warning
diCorVal	DINT	Command variable

Description

Abbildung 12: PID_CTRL: Principle of the PID controller



To optimize processing time, no floating-point operations are used for the PID algorithm. Also, Kp, Ki, and Kd are normalized to 1/256.

Thus:

where

y(k) = diCorVal
Δx(k) = iSetVal - iActVal

'diImax' limits the integral action. The integrator stops automatically when this limit is reached.
The integrator can be controlled externally with the 'boStopIpos' and 'boStopIneg' input variables, e.g. by applying a limit with 'ADD_LIMIT'.
The 'boClearI' input variable provides a means of externally deleting the accumulated integral action.

The amplification factors Kp, Ki, and Kd are calculated from the proportional component P, the reset time Tn, the derivative time Tv, and the sampling time T0 with the 'PID_TO_KPKIKD' function block.

'ADD_LIMIT' queries an additive command variable (feed-forward control) with limiting of the result.

2.1.6.5 PM_CORRECT (FB)

The 'PM_CORRECT' function block supports controlled output of correction values.
Controlled output means linear interpolation of the correction value across a definable 'diInVal' range.

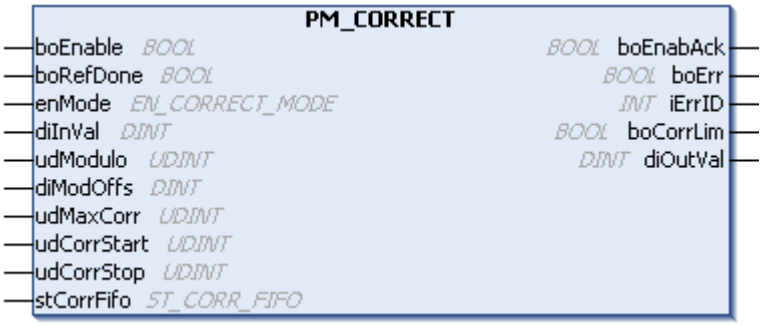
The function block takes the correction values from the FIFO structure 'stCorrFifo', the content of which is written by the 'PM_DETECT' block, for example. 'stCorrFifo' supports spacing of the printing mark sensors greater than one format.

Further properties:

- Definition of interpolation behavior
  Parameter to determine the interpolation range and the maximum permissible correction per format
- Indication whether the correction value is being limited

Combined with the 'PM_DETECT' function block, 'PM_CORRECT' facilitates efficient printing mark control.

User interface



Input variables

Name	Type	Description
boEnable	BOOL	Enable signal: With a positive edge, the initialisation of the block starts. As long as 'boEnable' = TRUE, the block remains enabled and is processed by the PLC. In the state 'boEnable' = FALSE the block is no longer enabled and is thus no longer processed.
boRefDone	BOOL	Homing cycle completed Acknowledgement signal to indicate that a homing cycle has been completed. • In mode 'enMode' = DETECT_AUTO, 'boRefDone' = FALSE when the block is activated or on a positive edge change at 'boRefStart'. Once the first mark has been detected, 'boRefDone' = TRUE is set. • In mode 'enMode' = DETECT_MANUAL, this variable is of no significance. 'boRefDone' = TRUE always applies.
enMode	ENUM	EN_CORRECT_MODE Selection mode of operating mode; 'enMode' can be changed online

Default	CORRECT_SET2OUT

Range	Meaning
CORRECT_SET2OUT	Only the correction value is output
CORRECT_ADD2OUT	The interpolation of the correction value is output added to the input value 'diInVal'
CORRECT_SET2OUT_NB	Only the correction value is output Avoidance of reversal of direction
CORRECT_ADD2OUT_NB	The interpolation of the correction value is output added to the input value 'diInVal' Avoidance of reversal of direction

Name	Type	Description
diInVal	DINT	Input value referenced by the subsequent input variables
udModulo	UDINT	Modulo format Describes the setpoint distance between two consecutive printing marks. The value is saved when the block is activated (positive edge at 'boEnable'). A subsequent change does not affect the active block. Range: 0 ... 1000000000 Default: 2000
diModOffs	DINT	Modulo offset Set PM position within modulo format. This information is required in order to set the offset following an automatic homing cycle. The 'diModOffs' variable is normally generated from the 'diModVal' output of the 'PM_DETECT' block. If 'diModOffs' is greater than 'udModulo', the modulo residual ('diModOffs' % 'udModulo') applies.
udMaxCorr	UDINT	Maximum permissible correction value to which the correction value output per modulo format is limited. The value can be changed when the block is active Range: 0 ... 999999999 Default: 2000
udCorrStart	UDINT	Correction starting value at which the output of correction values commences. The value can be changed when the block is active Range: 0 ... 999999999 Default: 15000
udCorrStop	UDINT	Correction stop value at which the output of correction values ends. The value can be changed when the block is active Range: 0 ... 999999999 Default: 19999

Output variables

Name	Type	Description
boEnabAck	BOOL	Acknowledgement: Ramp function is active
boErr	BOOL	The function block is in an error state FALSE: No error (permitted commanding or warning) TRUE: Error

<table>
<tr><th>Name</th><th>Type</th><th>Description</th></tr>
<tr><td>iErrID</td><td>INT</td><td>Error identity number: Diagnostic number is output
<table>
<tr><td colspan="2">iErrID = 0</td><td>No error</td></tr>
<tr><td>iErrID ≠ 0</td><td>boErr = TRUE</td><td>Error</td></tr>
<tr><td>iErrID ≠ 0</td><td>boErr = FALSE</td><td>Warning</td></tr>
</table>
Warning
<table>
<tr><th>Range</th><th>Meaning</th></tr>
<tr><td>1</td><td>Excess velocity of input values (Δ diInVal) so that the permissible range can be detected correctly, for example</td></tr>
<tr><td>2</td><td>Read from correction FIFO not possible because, for example, mark detection has not yet commenced</td></tr>
<tr><td>3</td><td>Impermissible direction of rotation prevented
only for 'enMode' = CORRECT_SET2OUT_NB, CORRECT_ADD2OUT_NB</td></tr>
</table>
Error
<table>
<tr><th>Range</th><th>Meaning</th></tr>
<tr><td>1</td><td>Illegal FIFO index</td></tr>
<tr><td>2</td><td>Illegal mode</td></tr>
<tr><td>3</td><td>Illegal modulo format; 'udModulo' > 1000000000</td></tr>
<tr><td>4</td><td>Illegal "maximum valid correction value":
'udMaxCorr' ≥ 'udModulo'</td></tr>
<tr><td>5</td><td>Illegal correction starting value:
'udCorrStart' ≥ 'udModulo'</td></tr>
<tr><td>6</td><td>Illegal correction stopping value:
'udCorrStop' ≥ 'udModulo'</td></tr>
<tr><td>7</td><td>Illegal correction starting / stopping value combination</td></tr>
</table></td></tr>
<tr><td>boCorrLim</td><td>BOOL</td><td>Correction limiting
Display a limit of the correction value according to 'udMaxCorr'.
The variable is set to true for one cycle after 'udCorrStart' and before 'udCorrStop'</td></tr>
<tr><td>diOutVal</td><td>DINT</td><td>Output value
Output of the correction value in the form of a linear interpolation covering the range 'udCorrStart' through 'udCorrStop' ('enMode' = 'CORRECT_SET2OUT' or 'CORRECT_SET2OUT_NB').
Outputs the correction value in the form of a linear interpolation covering the range 'udCorrStart' through 'udCorrStop' additively linked to the input value 'diInVal'
('enMode' = 'CORRECT_ADD2OUT' or 'CORRECT_ADD2OUT_NB').</td></tr>
</table>

Input and output variables

Name	Type	Description
stCorrFifo	STRUCT	ST_CORR_FIFO Correction value FIFO Transfer of detected correction values with 'PM_DETECT' function block The correction values are output at the 'diOutVal' output in the form of a linear interpolation, distributed across the range from 'udCorrStart' to 'udCorrStop'

2.1.6.6 PM_DETECT (FB)

The 'PM_DETECT' function block is used to detect a printing mark and calculate correction values.

The correction values are calculated from the difference between the setpoint position and the position at the time at which a printing mark is detected.
The correction values are saved in a FIFO structure.

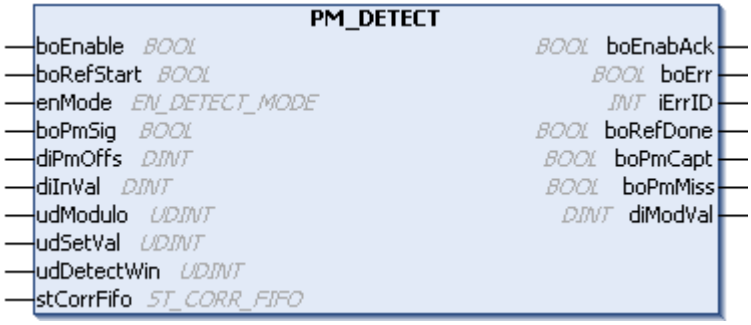
Thus the spacing between the printing mark sensors can be greater than one format.

Further properties:

- Automatic homing to the first printing mark
  Definition of the coordinate reference
- Allowance of a validity range for the printing mark
- Signaling of printing mark detection

Combined with the 'PM_CORRECT' function block, 'PM_DETECT' facilitates efficient printing mark control (PMC).

User interface



Input variables

Name	Type	Description
boEnable	BOOL	Enable signal: With a positive edge, the initialisation of the block starts. As long as 'boEnable' = TRUE, the block remains enabled and is processed by the PLC. In the state 'boEnable' = FALSE the block is no longer enabled and is thus no longer processed.
boRefStart	BOOL	Start of a new homing cycle; alignment with next printing mark Only applies in conjunction with 'enMode' = DETECT_AUTO. When the block is activated in 'DETECT_AUTO' mode, a homing cycle is started without 'boRefStart' being evaluated. A positive edge change at 'boRefStart' triggers repeat homing without the block having to be reactivated (positive edge at 'boEnable').
enMode	ENUM	EN_DETECT_MODE Selection mode of the operating mode Default: DETECT_AUTO Range: DETECT_AUTO — Automatic homing with reference to the first printing mark Range: DETECT_MANUAL — Manual homing The printing mark must be aligned manually with the sensor prior to enabling with 'boEnable'.
boPmSig	BOOL	Printing mark signal (PM signal) Signal indicating a printing mark inside the modulo format (The signal must remain pending for at least 1 sampling time)
diPmOffs	DINT	Printing mark offset (PM offset) Describes the deviation between the time-discrete input value 'diInVal' (kT0) and the actual input value 'diInVal'(TboPmSig) at the time of the edge change on the printing mark signal The following applies:
diInVal	DINT	Input value referenced by the subsequent input variables

Name	Type	Description
udModulo	UDINT	Modulo format Describes the setpoint distance between two consecutive printing marks. The value is saved when the block is activated (positive edge at 'boEnable'). A subsequent change does not affect the active block. Range: 0 ... 1000000000 Default: 2000
udSetVal	UDINT	PM setpoint Describes the expected distance between printing mark and printing mark sensor. The value is saved when the block is activated (positive edge at 'boEnable'). A subsequent change does not affect the active block. If udSetVal ≥ udModulo, n correction values "0" are entered in 'stCorrFifo'. This corresponds to a slip in the correction value of n formats. In other words, the mark sensor is positioned n formats upstream of the tool position. Default: 1000
udDetectWin	UDINT	Permissible range A 'boPmSig' flag signal is permitted within this range. The value can be changed online when the block is active Range: 0 ... 999999999 Default: 5000

Output variables

Name	Type	Description
boEnabAck	BOOL	Acknowledgement: Function block is initialised and enabled
boErr	BOOL	The function block is in an error state FALSE: No error (permitted commanding or warning) TRUE: Error

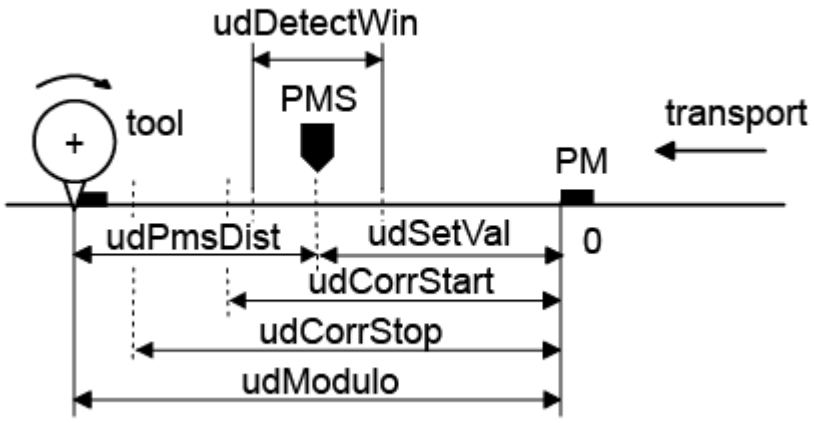
<table>
<tr><th>Name</th><th>Type</th><th>Description</th></tr>
<tr><td>iErrID</td><td>INT</td><td>Error identity number: Diagnostic number is output
<table>
<tr><td colspan="2">iErrID = 0</td><td>No error</td></tr>
<tr><td>iErrID ≠ 0</td><td>boErr = TRUE</td><td>Error</td></tr>
<tr><td>iErrID ≠ 0</td><td>boErr = FALSE</td><td>Warning</td></tr>
</table>
Warning
<table>
<tr><th>Value</th><th>Meaning</th></tr>
<tr><td>1</td><td>Excess velocity of input values (Δ 'diInVal') so that the permissible range can be detected correctly, for example</td></tr>
<tr><td>2</td><td>Write to correction FIFO not possible, e.g. because correction is not being carried out</td></tr>
<tr><td>3</td><td>Illegal direction of rotation, e.g. if the direction of rotation changes compared with the reaching of the first mark; a correction value is not entered in the FIFO</td></tr>
</table>
Error
<table>
<tr><th>Value</th><th>Meaning</th></tr>
<tr><td>1</td><td>Illegal FIFO index</td></tr>
<tr><td>2</td><td>Illegal mode</td></tr>
<tr><td>3</td><td>Illegal modulo format
'udModulo' > 1000000000</td></tr>
<tr><td>4</td><td>Illegal PM setpoint
'udSetVal'/'udModulo' > MAX_CORR_FIFO_IND_1</td></tr>
<tr><td>5</td><td>Illegal validity range
'udDetecWin' ≥ 'udModulo'</td></tr>
</table></td></tr>
<tr><td>boRefDone</td><td>BOOL</td><td>Homing cycle completed
Acknowledgement signal to indicate that a homing cycle has been completed.
In mode 'enMode' = DETECT_AUTO, 'boRefDone' = FALSE when the block is activated or on a positive edge change at 'boRefStart'.
Once the first mark has been detected, 'boRefDone' = TRUE is set.
In mode 'enMode' = DETECT_MANUAL, this variable is of no significance. 'boRefDone' = TRUE always applies.</td></tr>
<tr><td>boPmCapt</td><td>BOOL</td><td>Printing mark detected
Signal indicating that a printing mark has been detected inside the permissible range.
'boPmCapt' is TRUE for one cycle only.
At the same time, the calculated correction value is entered in the FIFO correction value.</td></tr>
<tr><td>boPmMiss</td><td>BOOL</td><td>Printing mark missing
Signal indicating that a mark has not been detected inside the permissible range.
'boPmMiss' is TRUE for one cycle only.
At the same time, the value "0" is entered in the FIFO correction value.</td></tr>
<tr><td>diModVal</td><td>DINT</td><td>Modulo value
Displays the current modulo position (0 ≤ diModVal < udModulo).
The sign depends on the direction of rotation.</td></tr>
</table>

Input and output variables

Name	Type	Description
stCorrFifo	STRUCT	ST_CORR_FIFO Correction value FIFO Transfer of detected correction values with 'PM_CORRECT' function block 'diCorrVal'[...] = 'diModVal' - 'udSetVal'

Description

Abbildung 13: PM_DETECT: Input variables



The figure illustrates the relationship of some of the input variables used by the 'PM_DETECT' and 'PM_CORRECT' blocks.

Where: (udSetVal % udModulo) < udCorrStart < udCorrStop < udModulo

If udPmsDist > udModulo, then: udSetVal = [udModulo • (n+1)] -X
where

X = udPmsDist % udModulo

$$n = \frac{udPmsDist - 1}{udModulo}$$

2.1.6.7 RATIO_ABS (FB)

The 'RATIO_ABS' function block is used to multiply and divide 32-bit values.

However, unlike 'RATIO_INC_1', for example, the input value 'dInVal' is treated as an absolute value (no input differences are generated).
Although the product 'diInVal' x 'diMultiplier' is generated as a 64-bit value initially, it must be possible for the result following division by 'udDivider' to be displayed as a 32-bit value.

Thus:

$$diOutVal = diInVal \times \frac{diMultiplier}{udDivider}$$

User interface

Input variables

Name	Type	Description
boEnable	BOOL	Enable signal: With a positive edge, the initialisation of the block starts. As long as 'boEnable' = TRUE, the block remains enabled and is processed by the PLC. In the state 'boEnable' = FALSE the block is no longer enabled and is thus no longer processed.
diInVal	DINT	Absolute Input value Input value differences are not generated

Name	Type	Description
diMultiplier	DINT	Multiplier by which the input value differences are multiplied Default: 10000
udDivider	UDINT	Divisor by which the input value differences are divided Default: 10000

Output variables

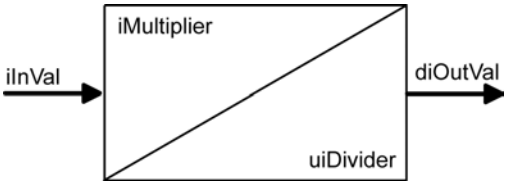
Name	Type	Description
boEnabAck	BOOL	Acknowledgement: Function block is initialised and enabled
boErr	BOOL	The function block is in an error state FALSE: No error (permitted commanding or warning) TRUE: Error
iErrID	INT	Error identity number: Diagnostic number is output iErrID = 0: No error iErrID ≠ 0, boErr = TRUE: Error iErrID ≠ 0, boErr = FALSE: Warning Warning 2: Divisor = 0; set to 1 5: Output value cannot be displayed as DINT
diOutVal	DINT	Output value

2.1.6.8 RATIO_INC (FB)

The 'RATIO_INC' function block performs multiplication and division to specific increments; the ratio of input increments to output increments is defined.

The calculation algorithm ensures that any remainder is not lost. Moreover, the overrun of the 16-bit input value is managed by working with incremental difference:

Abbildung 14: RATIO_INC: Principle



$$\frac{\Delta diOutVal}{\Delta iInVal} = \frac{iMultiplier}{uiDivider}$$

$$diOutVal(k) = diOutVal(k-1) + \frac{iMultiplier}{uiDivider} \times (iInVal(k) - iInVal(k-1))$$

k Sampling point (in time)

iInVal(k-1) := iInVal(k) for k = 0
(positive edge of 'boEnable')

User interface

Input variables

Name	Type	Description
boEnable	BOOL	Enable signal: With a positive edge, the initialisation of the block starts. As long as 'boEnable' = TRUE, the block remains enabled and is processed by the PLC. In the state 'boEnable' = FALSE the block is no longer enabled and is thus no longer processed.
iInVal	INT	Input value e.g. the low-order word of the 32-bit actual position
iMultiplier	INT	Multiplier by which the input value differences are multiplied Range: -32766 ... +32766 Unit: incr Default: 10000
uiDivider	UINT	Divisor by which the input value differences are divided Range: 1 ... +32767 Unit: incr Default: 10000

Output variables

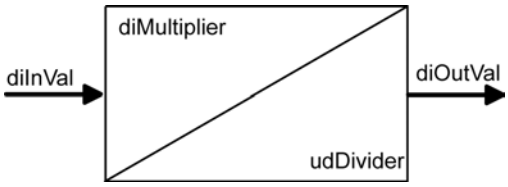
Name	Type	Description
boEnabAck	BOOL	Acknowledgement: Function block is initialised and enabled
boErr	BOOL	The function block is in an error state FALSE: No error (permitted commanding or warning) TRUE: Error
iErrID	INT	Error identity number: Diagnostic number is output iErrID = 0: No error iErrID ≠ 0, boErr = TRUE: Error iErrID ≠ 0, boErr = FALSE: Warning Warning 1: Multiplier too high; limited to minimum or maximum value 2: Divisor = 0; set to 1 3: Divisor > 32767; remainder not taken into account
diOutVal	DINT	Output value Sum of incoming increments, weighted with the 'iMultiplier'/'uiDivider' ratio

2.1.6.9 RATIO_INC_1 (FB)

The 'RATIO_INC_1' function block performs multiplication and division to specific increments; the ratio of input increments to output increments is defined.

'RATIO_INC_1' is equivalent to the 'RATIO_INC' function block, except that the 'diInVal', 'diMultiplier', and 'udDivider' input variables are 32-bit values.

Abbildung 15: RATIO_INC_1: Principle



k		Sampling point (in time)
diInVal(k-1)	:= diInVal(k)	for k = 0 (positive edge of 'boEnable')

User interface

Input variables

Name	Type	Description
boEnable	BOOL	Enable signal: With a positive edge, the initialisation of the block starts. As long as 'boEnable' = TRUE, the block remains enabled and is processed by the PLC. In the state 'boEnable' = FALSE the block is no longer enabled and is thus no longer processed.
diInVal	DINT	Input value
diMultiplier	DINT	Multiplier by which the input value differences are multiplied Default: 10000
udDivider	UDINT	Divisor by which the input value differences are divided Default: 10000

Output variables

Name	Type	Description
boEnabAck	BOOL	Acknowledgement: Function block is initialised and enabled
boErr	BOOL	The function block is in an error state FALSE: No error (permitted commanding or warning) TRUE: Error
iErrID	INT	Error identity number: Diagnostic number is output iErrID = 0: No error iErrID ≠ 0, boErr = TRUE: Error iErrID ≠ 0, boErr = FALSE: Warning Warning 2: Divisor = 0; set to 1 4: Output difference cannot be displayed as DINT; the difference is limited to the maximum DINT value
diOutVal	DINT	Output value

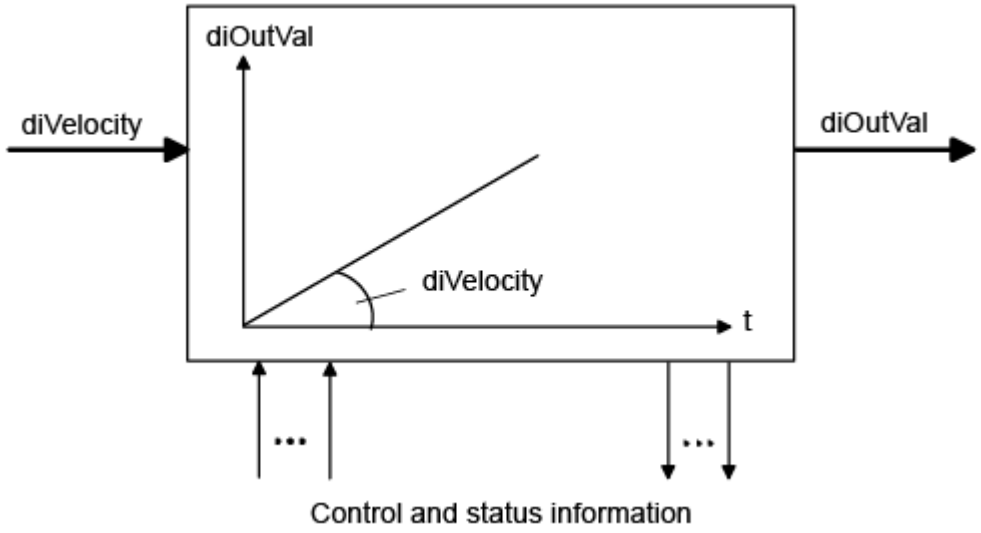
2.1.6.10 VGEN (FB)

The 'VGEN' function block is a velocity generator.
The output value is a position setpoint which changes in proportion with the velocity. It is also possible to output a defined number of increments.

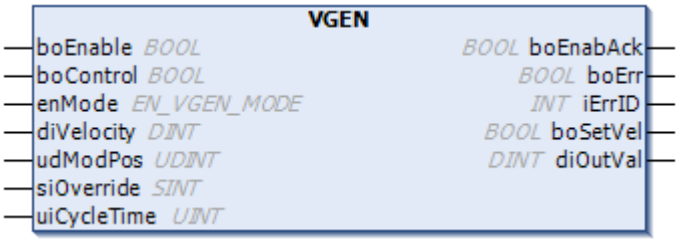
The following functions are supported:

- Generation of an increment increase in accordance with a definable velocity.
- Online changes to input parameters
- Modes for continuous and cyclic increment generation

Abbildung 16: VGEN: Block diagram



User interface



Input variables

Name	Type	Description
boEnable	BOOL	Enable signal: With a positive edge, the initialisation of the block starts. As long as 'boEnable' = TRUE, the block remains enabled and is processed by the PLC. In the state 'boEnable' = FALSE the block is no longer enabled and is thus no longer processed.
boControl	BOOL	Start / Stop

Range		Meaning
enMode = VGEN_CONT	FALSE	Velocity = 0
	TRUE	Velocity = 'diVelocity'
enMode = VGEN_CYCLE	FALSE -> TRUE	Velocity = 'diVelocity' for a series of 'udModPos' increments

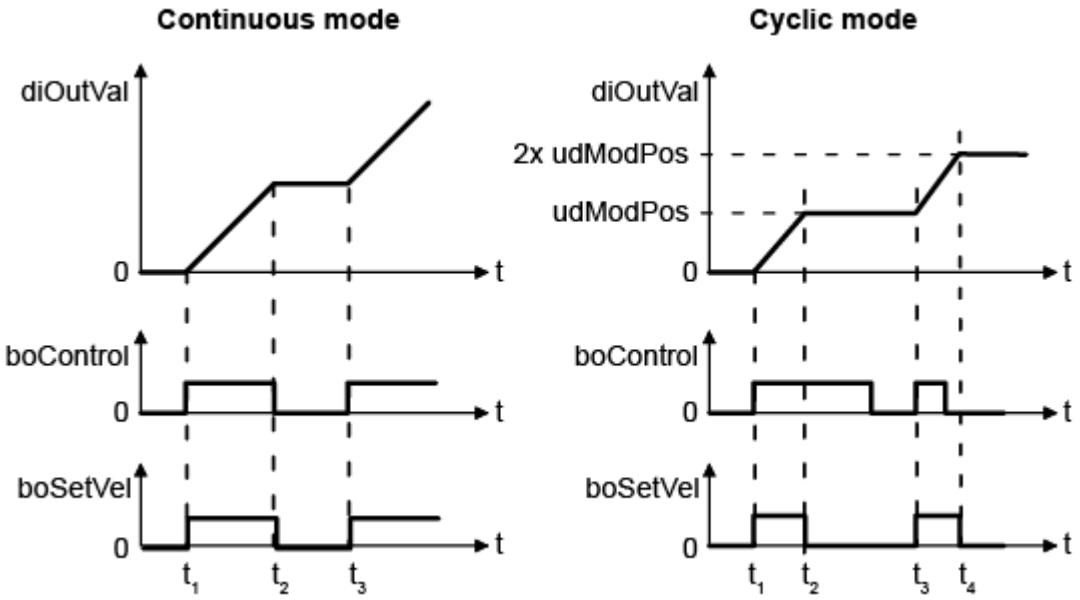
Name	Type	Description
enMode	ENUM	EN_VGEN_MODE Selection mode of the operating mode Default: VGEN_CONT VGEN_CONT: Continuous increment increase, as long as 'boControl' = TRUE VGEN_CYCLE: Increment increase by the defined value 'udModPos' on a positive edge at 'boControl'
diVelocity	DINT	Setpoint velocity Definition of the final velocity Range: -300000000 ... 300000000 Unit: incr/s Default: 1000
udModPos	UDINT	Modulo position Number of increments to be output in mode 'enMode' = VGEN_CYCLE Unit: incr Default: 2000
siOverride	SINT	Velocity output factor Range: -100 ... +100 Unit: % Default: 100
uiCycleTime	UINT	Cycle time (usually corresponding to ID2 'SERCOS cycle time') Unit: 0.001 ms

Output variables

Name	Type	Description
boEnabAck	BOOL	Acknowledgement: Function block is initialised and enabled
boErr	BOOL	The function block is in an error state FALSE: No error (permitted commanding or warning) TRUE: Error
iErrID	INT	Error identity number: Diagnostic number is output iErrID = 0: No error iErrID ≠ 0, boErr = TRUE: Error iErrID ≠ 0, boErr = FALSE: Warning Warning Range: 1 — Meaning: Illegal mode Error Range: 1 — Meaning: Illegal setpoint velocity (limited to minimum or maximum value) Range: 2 — Meaning: Illegal override (limited to minimum or maximum value)
boSetVel	BOOL	When 'boSetVel' is active, the target velocity has been reached.
diOutVal	DINT	Output value

Description

Abbildung 17: VGEN: Principle of operation



'enMode' is used to differentiate between continuous and cyclic operation.

- Continuous operation: 'enMode' = VGEN_CONT
  Generation of a continuous increment increase, corresponding to a predefined setpoint velocity ('diVelocity' [incr/s]). The increment increase is controlled with the control signal ('boControl').

  'boControl' = TRUE t1 ≤ t ≤ t2; t3 < t

  'boControl' = FALSE t < t1; t2 ≤ t ≤ t3
- Cyclic operation: 'enMode' = VGEN_CYCLE
  Generation of a defined increment increase ('udModPos' [incr]) corresponding to a predefined setpoint velocity ('diVelocity' [incr/s]). The increment increase is controlled with the positive edge of the control signal ('boControl').

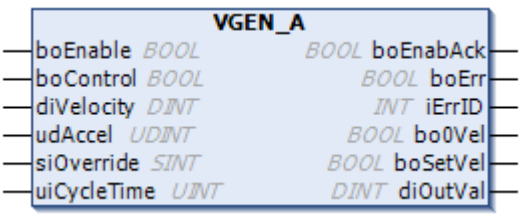
  'boControl' =FALSE -> TRUE (t = t1; t = t3): Increment increase udModPos t1 ≤ t ≤ t2; t3 ≤ t ≤ t4

2.1.6.11 VGEN_A (FB)

The function block 'VGEN_A' is a velocity generator with definable acceleration.
The output value is a position with an increment difference proportional to the velocity 'diVelocity' and a change in increment difference proportional to the acceleration 'udAccel'.
The block can be used for direct control of a drive. It can also be used as an input value generator for other blocks.
The following functions are supported:

- Generation of an increment increase in accordance with a definable velocity
- Specification of a defined acceleration / deceleration
- Online changes to input parameters.

User interface



Input variables

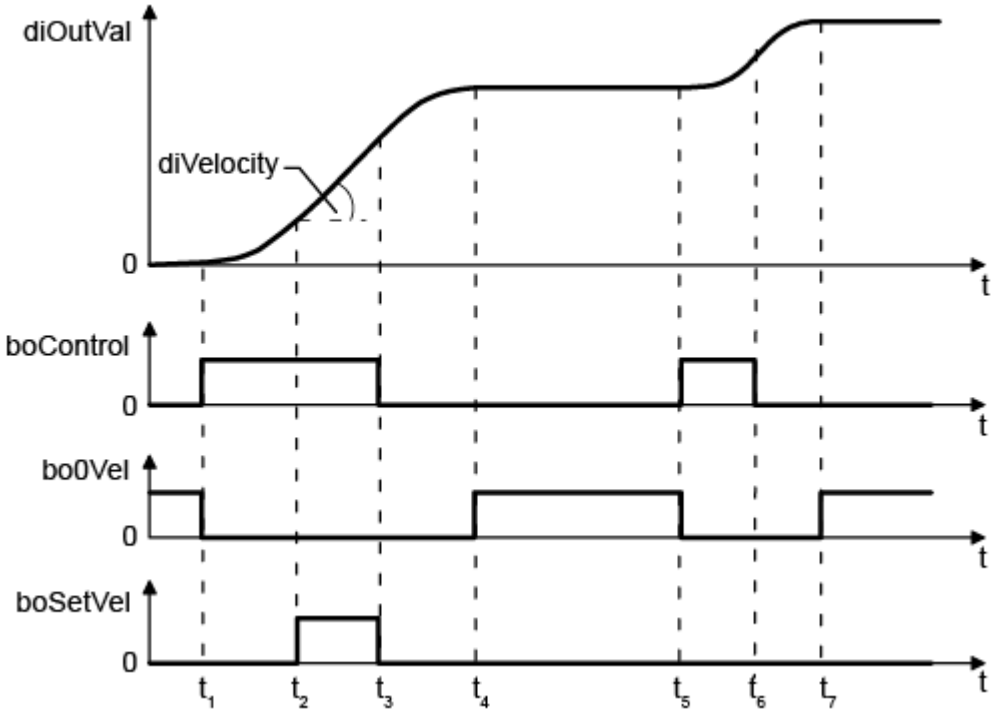
Name	Type	Description
boEnable	BOOL	Enable signal: With a positive edge, the initialisation of the block starts. As long as 'boEnable' = TRUE, the block remains enabled and is processed by the PLC. In the state 'boEnable' = FALSE the block is no longer enabled and is thus no longer processed.
boControl	BOOL	Start / Stop Control of increment output FALSE: Reduction in velocity proportional to 'udAccel' until velocity 0 is reached TRUE: Increase in velocity proportional to 'udAccel' until velocity 'diVelocity' is reached
diVelocity	DINT	Setpoint velocity Definition of the final velocity Range: -300000000 ... 300000000 Unit: incr/s Default: 500000
udAccel	UDINT	Acceleration with which the target velocity is run Range: -4000000000 ... 4000000000 Unit: incr/s^2 Default: 100000
siOverride	SINT	Velocity output factor Range: -100 ... +100 Unit: % Default: 100
uiCycleTime	UINT	Cycle time (usually corresponding to ID2 'SERCOS cycle time') Unit: 0.001 ms

Output variables

Name	Type	Description
boEnabAck	BOOL	Acknowledgement: Function block is initialised and enabled
boErr	BOOL	The function block is in an error state FALSE: No error (permitted commanding or warning) TRUE: Error
iErrID	INT	Error identity number: Diagnostic number is output iErrID = 0: No error iErrID ≠ 0, boErr = TRUE: Error iErrID ≠ 0, boErr = FALSE: Warning Error 1: Illegal setpoint velocity (limited to minimum or maximum value) **2**: Illegal acceleration (limited to minimum or maximum value) **3**: Illegal override (limited to minimum or maximum value)

Name	Type	Description
bo0Vel	BOOL	When 'bo0Vel' is active, no setpoint is output.
boSetVel	BOOL	When 'boSetVel' is active, the target velocity has been reached.
diOutVal	DINT	Output value

Abbildung 18: VGEN_A: Principle of operation



t = t1	boControl = TRUE	The output of position setpoints (increments) commences once the control signal has been activated
t1 ≤ t ≤ t2		Acceleration phase In this range, the velocity is increased proportional to the acceleration 'udAccel' until the predefined setpoint velocity 'diVelocity' is reached.
t = t2		Setpoint velocity reached; 'boSetVel' = TRUE
t2 ≤ t ≤ t3		In this range, the velocity remains constant (constant increment difference).
t = t3	boControl = FALSE	Once the control signal has been deactivated, the difference of the output increments is reduced.
t3 ≤ t ≤ t4		Deceleration phase In this range, the velocity is reduced proportional to the acceleration 'udAccel' until velocity 0 is reached.
t = t4		Standstill reached; 'bo0Vel' = TRUE
t4 ≤ t ≤ t5		Standstill phase In this range, the output value does not change (velocity = 0)
t = t5	boControl = TRUE	Renewed output of position setpoints
t5 ≤ t ≤ t6		Renewed acceleration
t = t6	boControl = FALSE	Acceleration phase is interrupted before the setpoint velocity is reached
t6 ≤ t≤ t7		Deceleration phase starts immediately without setpoint velocity being reached
t = t7		Standstill reached; 'bo0Vel' = TRUE

2.1.6.12 VGEN_AJ (FB)

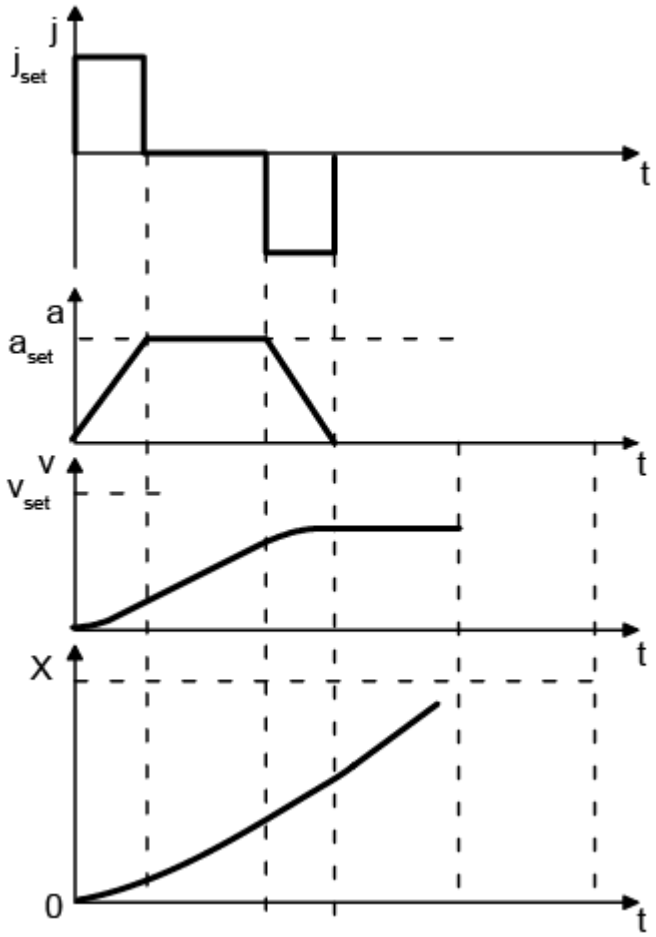
The block 'VGEN_AJ' is a velocity generator with definable values for acceleration and jerk.
The output value is a position with an increment difference proportional to the velocity 'diVelocity' a change in increment difference proportional to the acceleration 'udAccel', and a change in increment difference change proportional to the jerk 'udAccJerk' / 'udDecJerk' / 'udQDecelJerk'.

The block can be used for direct control of a drive. It can also be used as an input value generator for other blocks.

The following functions are supported:

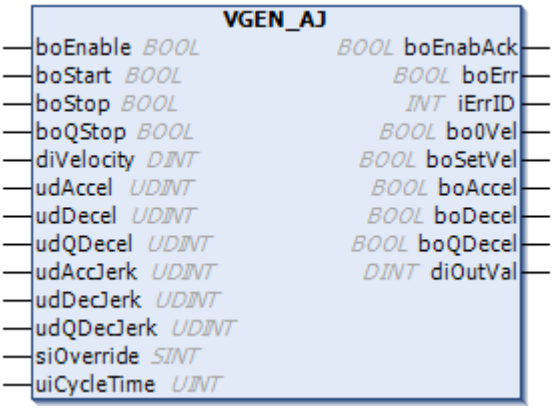
- Generation of an increment increase in accordance with a definable velocity.
- Specification of a defined acceleration (deceleration).
- Special quick stop mode with assigned deceleration values.
- Jerk default values for the various ramps.
- Velocity override
- Online changes to input parameters

Abbildung 19: VGEN_AJ: Principle of operation



The figure illustrates the graphical relationship between jerk j_{set} , acceleration, a_{set}, velocity v_{set}, and the resulting position characteristic X.

User interface



Input variables

Name	Type	Description
boEnable	BOOL	Enable signal: With a positive edge, the initialisation of the block starts. As long as 'boEnable' = TRUE, the block remains enabled and is processed by the PLC. In the state 'boEnable' = FALSE the block is no longer enabled and is thus no longer processed.
boStart	BOOL	Acceleration ramp is generated with the 'udAccel' and 'udAccJerk' parameters until velocity 'diVelocity'
boStop	BOOL	Braking ramp is generated with the 'udDecel' and 'udDecJerk' parameters until velocity 0
boQStop	BOOL	Fast braking ramp is generated with the 'udQDecel' and 'udQDecJerk' parameters until velocity 0. The fast braking ramp cannot be interrupted.
diVelocity	DINT	Setpoint velocity Definition of the final velocity Range: -300000000 … 300000000 Unit: incr/s Default: 500000
udAccel	UDINT	Acceleration with which the target velocity is run Range: -4000000000 ... 4000000000 Unit: incr/s^2 Default: 100000
udDecel	UDINT	Deceleration with which a lower target velocity is achieved Range: -4000000000 ... 4000000000 Unit: incr/s^2 Default: 100000
udQDecel	UDINT	Fast deceleration with which a lower target velocity is achieved Range: -4000000000 ... 4000000000 Unit: incr/s^2 Default: 500000

Name	Type	Description
udAccJerk	UDINT	Jerk during acceleration Range: -4000000000 ... 4000000000 Unit: incr/s^2 Default: 100000
udDecJerk	UDINT	Jerk during deceleration Range: -4000000000 ... 4000000000 Unit: incr/s^2 Default: 100000
udQDecJerk	UDINT	Jerk during fast deceleration Range: -4000000000 ... 4000000000 Unit: incr/s^2 Default: 100000
siOverride	SINT	Velocity output factor Range: -100 ... +100 Unit: % Default: 100
uiCycleTime	UINT	Cycle time (usually corresponding to ID2 'SERCOS cycle time') Unit: 0.001 ms

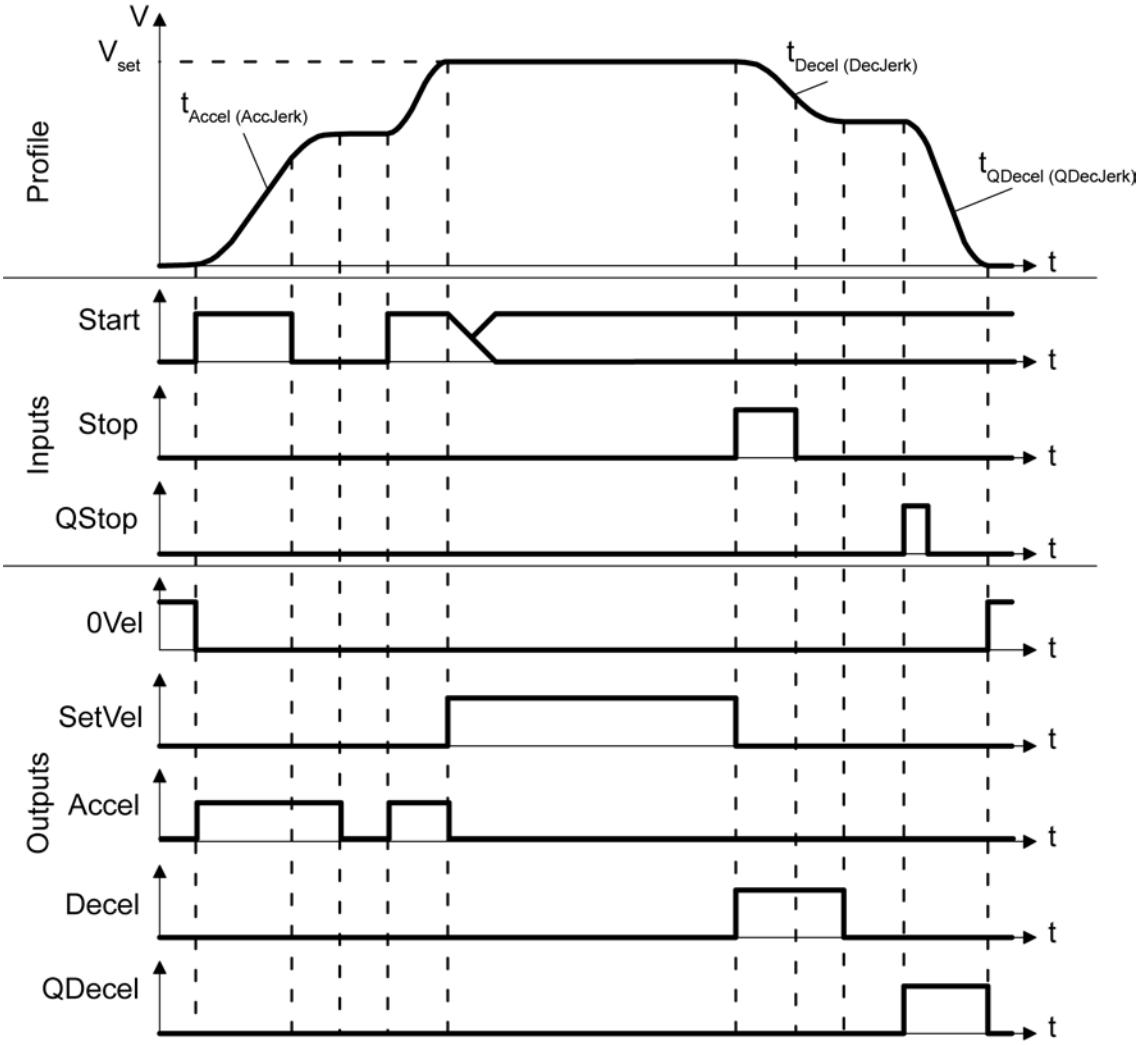
Output variables

Name	Type	Description
boEnabAck	BOOL	Acknowledgement: Function block is initialised and enabled
boErr	BOOL	The function block is in an error state FALSE: No error (permitted commanding or warning) TRUE: Error
iErrID	INT	Error identity number: Diagnostic number is output iErrID = 0: No error iErrID ≠ 0, boErr = TRUE: Error iErrID ≠ 0, boErr = FALSE: Warning Error 1: Velocity too high 2: Acceleration set to 0 / deceleration set to 0 3: Acceleration too high / deceleration too high 4: Jerk set to 0 5: Jerk too high 6: Override too high 7: Jerk corrected (more than 20% of the setpoint)
bo0Vel	BOOL	When 'bo0Vel' is active, no setpoint is output.
boSetVel	BOOL	When 'boSetVel' is active, the target velocity has been reached.
boAccel	BOOL	Acceleration phase active
boDecel	BOOL	Deceleration phase active
boQDecel	BOOL	Fast deceleration phase active

Name	Type	Description
diOutVal	DINT	Output value

Description

Abbildung 20: VGEN_AJ: Behavior of the velocity generator

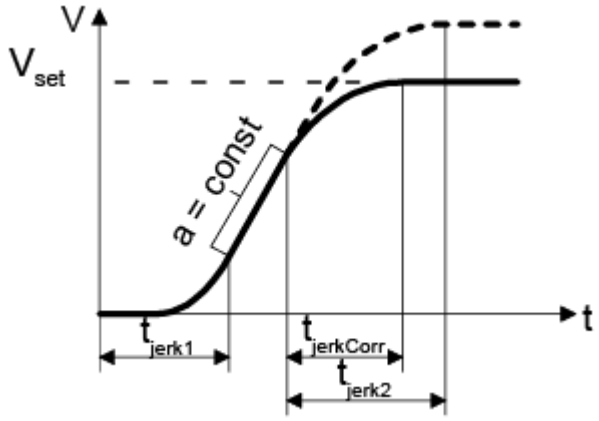


The figure shows how the velocity generator uses different values for the acceleration, deceleration, and fast deceleration ramps. Accordingly, the generator uses three different jerk values, one for each ramp.

All parameters can be changed during block runtime.
However, please note:

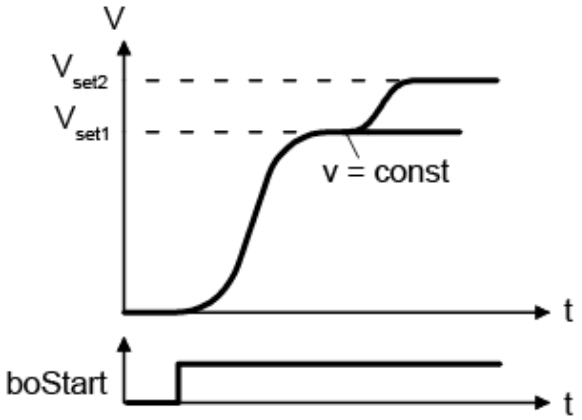
- If the jerk setpoint changes during the constant acceleration phase such that the velocity setpoint (or velocity 0) would be exceeded (or undershot), the jerk phase commences immediately. The minimum possible jerk is calculated for this.

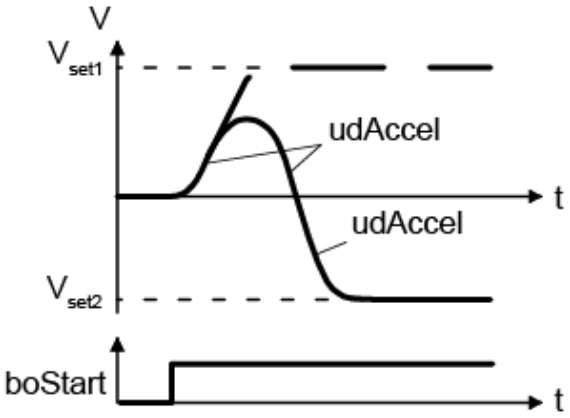
  Abbildung 21: VGEN_AJ: Response to change in jerk



- If the jerk changes after the second jerk phase following constant acceleration / deceleration has commenced, this change is not taken into account.
- If the setpoint velocity or the velocity override changes during the acceleration ramp, the changed value is applied if: 'boStart' = TRUE, 'boStop' = FALSE, 'boQStop' = FALSE.
  If the changed velocity requires negative acceleration, this is done with a ramp defined by 'udAccel' and 'udAccJerk'. Although it looks like a deceleration ramp, in this case the values of the acceleration ramp apply.

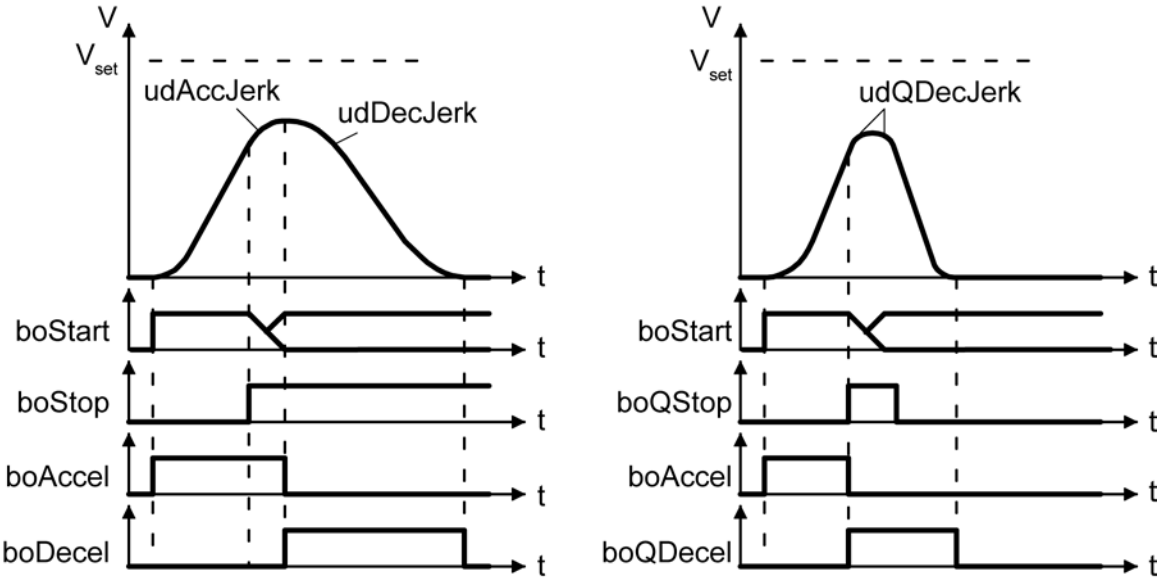
  Abbildung 22: VGEN_AJ: Response to change in override velocity





- If the 'boStop' input becomes active during the acceleration ramp ('boStop' = TRUE), 'VGEN_AJ' responds initially by generating a jerk phase with 'udAccJerk' until the current acceleration becomes 0. After this, the block starts a deceleration ramp as defined by 'udDecel' and 'udDecJerk'.
  The procedure differs for a fast stop: 'VGEN_AJ' reduces the current acceleration immediately with 'udQDecJerk'. If the current acceleration is 0, the process continues with a deceleration ramp with 'udQDecel'.

  Abbildung 23: VGEN_AJ: Different response to normal stop and fast stop



2.1.7 FifoHandling (FIFO functions)

FIFO_HANDLER	FIFO block

2.1.7.1 FIFO_HANDLER (FB)

The 'FIFO_HANDLER' function block serves as a FIFO memory (FIFO stands for first in first out).
The block is characterized as follows:

- The information managed in the FIFO (a FIFO element) can be structured at will
- The FIFO is organized so that it can also be used for communication between two processes (thread save)
- The size of the FIFO can be specified as variable

The 'FifoInit()' action must be executed before the rest of the FIFO function can be used.
The values for 'uiEleSize', 'uiFifoSize', 'pbyFifo', and 'stFifoHeader' predefined in the context of 'FifoInit()' must not be changed again subsequently
'FifoReset()' is only possible if there has not yet been a FIFO overrun

User interface

Input variables

Name	Type	Description
boExec	BOOL	Function execution: With a positive edge, the execution of the block starts. As long as 'boExec' = TRUE, the block is processed by the PLC. In the state 'boExec' = FALSE execution of the block is ended.

<table>
<tr><th>Name</th><th>Type</th><th>Description</th></tr>
<tr><td>enMode</td><td>ENUM</td><td>EN_FIFO_HANDLER_MODE
Selection mode of the function
The function is executed in the context of a [Fifoxxxxx] action
<table>
<tr><td>Default</td><td>FIFO_INIT</td></tr>
</table>
<table>
<tr><th>Range</th><th>Meaning</th></tr>
<tr><td>FIFO_INIT
[FifoInit]</td><td>Initialize FIFO
The relevant input variables are added to the 'stFifoHeader' variable and the memory made available for the FIFO is cleared (FifoClear).</td></tr>
<tr><td>FIFO_CLEAR
[FifoClear]</td><td>Clear FIFO
'uiEleNmb' and 'udEleInd' are cleared (:= 0).
The internal state 'stFifoHeader.enFifoState' = FIFO_STATE_READY is also set.</td></tr>
<tr><td>FIFO_RESET
[FifoReset]</td><td>Reset output index
The action sets 'udEleInd' := 0 and 'uiEleNmb' to the number of elements already written to the FIFO.
This enables the elements already written to be read again.
 
The action can only be used if there has not yet been a FIFO overrun: 'stFifoHeader.enFifoState' = FIFO_STATE_READY</td></tr>
<tr><td>FIFO_READ
[FifoRead]</td><td>Read out FIFO element
Read the first FIFO element written and not yet read out. This current element is then written to the address referenced in 'pbyEle' with 'uiEleSize' bytes.
'uiEleNmb' is decremented, 'udEleInd' is incremented.</td></tr>
<tr><td>FIFO_WRITE
[FifoWrite]</td><td>Write FIFO element
The element referenced by 'pbyEle' is written. 'uiEleSize' bytes are written.
'uiEleNmb' is incremented.</td></tr>
</table></td></tr>
<tr><td>uiEleSize</td><td>UINT</td><td>Size (in bytes) of the element to be written / read</td></tr>
<tr><td>pbyEle</td><td>POINTER</td><td>POINTER TO BYTE
enMode = FIFO_READ:
Pointer to the address starting from which the element read out is saved
enMode = FIFO_WRITE:
Pointer to the address starting from which data is transferred to the FIFO</td></tr>
<tr><td>uiFifoSize</td><td>UINT</td><td>Size (in bytes) of the memory made available for the FIFO organization</td></tr>
<tr><td>pbyFifo</td><td>POINTER</td><td>POINTER TO BYTE
Pointer to the address starting from which memory capacity is made available for the FIFO organization</td></tr>
</table>

Output variables

<table>
<tr><th>Name</th><th>Type</th><th>Description</th></tr>
<tr><td>boDone</td><td>BOOL</td><td>Response that the function block has been completely executed.</td></tr>
<tr><td>boErr</td><td>BOOL</td><td>The function block is in an error state
<table>
<tr><td>FALSE</td><td>No error (permitted commanding or warning)</td></tr>
<tr><td>TRUE</td><td>Error</td></tr>
</table></td></tr>
</table>

Name	Type	Description
iErrID	INT	Error identity number: Diagnostic number is output iErrID = 0 — No error iErrID ≠ 0 — boErr = TRUE — Error iErrID ≠ 0 — boErr = FALSE — Warning 1 Illegal mode 2 Invalid element size 3 Element pointer not initialized 4 Invalid FIFO range size 5 FIFO range pointer not initialized 6 Illegal FIFO header information 7 FIFO is not initialized 8 Reset function illegal 9 FIFO full 10 FIFO empty
uiEleNmb	UINT	Number of elements written to the FIFO and not yet read back.
udEleInd	UDINT	FIFO position from which data is currently being read with 'FifoRead()'.

Input and output variables

Name	Type	Description
stFifoHeader	STRUCT	ST_FIFO_HEADER FIFO header information Organization of the FIFO

2.1.8 ID_Access (ID access functions)

AllElementsOfOneID
Read all elements of SERCOS-based parameters

READ_ID_ALL	Read all parameter elements

ElementaryAccess
Element parameter access

READ_ID_DINT	Read parameter value
READ_ID_DINT_TMP	Read parameter value
READ_ID_LIST	Read parameter values from a list
READ_LIST_512	Read parameter values from a 512-byte list
READ_SDO	SDO read access
WRITE_ID_DINT	Write parameter value
WRITE_ID_DINT_TMP	Write parameter value
WRITE_ID_LIST	Write parameter values in a list
WRITE_LIST_512	Write parameter values in a 512-byte list
WRITE_SDO	SDO write access

HigherAccess
Simplified parameter access

READ_ID_DINT_ONCE	Read parameter value
WRITE_ID_DINT_ONCE	Write parameter value

MoreIds
Multiple parameter access

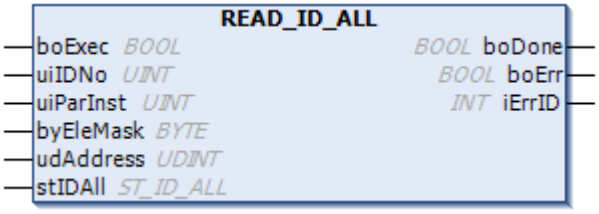
READ_N_IDS_DINT	Read N parameter values
WRITE_N_IDS_DINT	Write N parameter values

2.1.8.1 AllElementsOfOneID

2.1.8.1.1 READ_ID_ALL (FB)

The 'READ_ID_ALL' function block reads elements of a parameter stored in the AMK subsystem.

User interface



Input variables

Name	Type	Description
boExec	BOOL	Function execution: With a positive edge, the execution of the block starts. As long as 'boExec' = TRUE, the block is processed by the PLC. In the state 'boExec' = FALSE execution of the block is ended.
uiIDNo	UINT	ID number to be read out
uiParInst	UINT	Instance or Parameter set number or instance number
byEleMask	BYTE	Element mask Selection of the parameter element to be read. Range: 0 = Not used; 1 = Not used; 2 = Name; 3 = Attribute; 4 = Unit; 5 = Minimum; 6 = Maximum; 7 = Value
udAddress	UDINT	EtherCAT address

Output variables

Name	Type	Description
boDone	BOOL	Response that the function block has been completely executed.
boErr	BOOL	The function block is in an error state FALSE: No error (permitted commanding or warning) TRUE: Error

Name	Type	Description
iErrID	INT	Error identity number: Diagnostic number is output iErrID = 0 — No error iErrID ≠ 0 — boErr = TRUE — Error iErrID ≠ 0 — boErr = FALSE — Warning Range: Siehe 'Error bit information' auf Seite 150.

Input and output variables

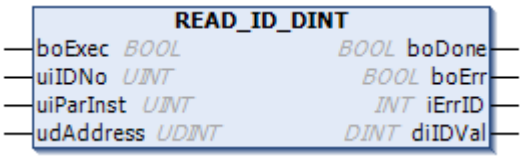
Name	Type	Description
stIDAll	STRUCT	ST_ID_ALL Parameter information Accommodates the element information

2.1.8.2 ElementaryAccess

2.1.8.2.1 READ_ID_DINT (FB)

The 'READ_ID_DINT' function block reads the value of a parameter stored in the AMK subsystem.

User interface



Input variables

Name	Type	Description
boExec	BOOL	Function execution: With a positive edge, the execution of the block starts. As long as 'boExec' = TRUE, the block is processed by the PLC. In the state 'boExec' = FALSE execution of the block is ended.
uiIDNo	UINT	ID number to be read out
uiParInst	UINT	Instance or Parameter set number or instance number
udAddress	UDINT	EtherCAT address

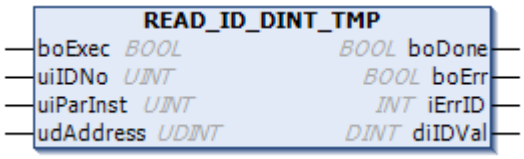
Output variables

Name	Type	Description
boDone	BOOL	Response that the function block has been completely executed.
boErr	BOOL	The function block is in an error state FALSE — No error (permitted commanding or warning) TRUE — Error
iErrID	INT	Error identity number: Diagnostic number is output iErrID = 0 — No error iErrID ≠ 0 — boErr = TRUE — Error iErrID ≠ 0 — boErr = FALSE — Warning Range: Siehe 'Error bit information' auf Seite 150.
diIDVal	DINT	Parameter value read from database

2.1.8.2.2 READ_ID_DINT_TMP (FB)

The 'READ_ID_DINT_TMP' function block reads the value of a parameter organized temporarily in the AMK subsystem.

User interface



Input variables

Name	Type	Description
boExec	BOOL	Function execution: With a positive edge, the execution of the block starts. As long as 'boExec' = TRUE, the block is processed by the PLC. In the state 'boExec' = FALSE execution of the block is ended.
uiIDNo	UINT	ID number to be read out
uiParInst	UINT	Instance or Parameter set number or instance number
udAddress	UDINT	EtherCAT address

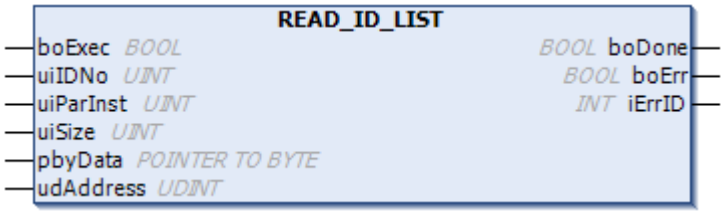
Output variables

Name	Type	Description
boDone	BOOL	Response that the function block has been completely executed.
boErr	BOOL	The function block is in an error state FALSE: No error (permitted commanding or warning) TRUE: Error
iErrID	INT	Error identity number: Diagnostic number is output iErrID = 0: No error iErrID ≠ 0, boErr = TRUE: Error iErrID ≠ 0, boErr = FALSE: Warning Range: Siehe 'Error bit information' auf Seite 150.
diIDVal	DINT	Parameter value read from database

2.1.8.2.3 READ_ID_LIST (FB)

The 'READ_ID_LIST' function block reads in values of a list parameter from the database of an AMK subsystem.

User interface



Input variables

Name	Type	Description
boExec	BOOL	Function execution: With a positive edge, the execution of the block starts. As long as 'boExec' = TRUE, the block is processed by the PLC. In the state 'boExec' = FALSE execution of the block is ended.

Name	Type	Description
uiIDNo	UINT	ID number to be read out
uiParInst	UINT	Instance or Parameter set number or instance number
uiSize	UINT	Maximum data length available to accommodate the information to be read. uiSize ≤ SIZEOF(variable) referenced by 'pbyData'!
pbyData	POINTER	POINTER TO READ DATA Pointer referencing the structure / variable which is receiving the information read.
udAddress	UDINT	EtherCAT address

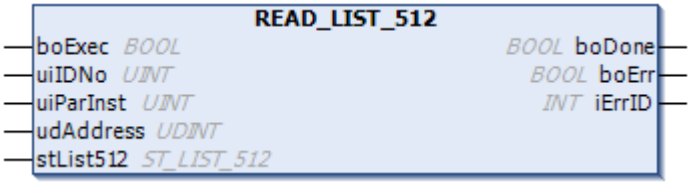
Output variables

Name	Type	Description
boDone	BOOL	Response that the function block has been completely executed.
boErr	BOOL	The function block is in an error state FALSE: No error (permitted commanding or warning) TRUE: Error
iErrID	INT	Error identity number: Diagnostic number is output iErrID = 0: No error iErrID ≠ 0 and boErr = TRUE: Error iErrID ≠ 0 and boErr = FALSE: Warning Range: Siehe 'Error bit information' auf Seite 150.

2.1.8.2.4 READ_LIST_512 (FB)

The 'READ_LIST_512' function block reads in values of a list parameter that is up to 512 bytes in size from the database of an AMK subsystem.

User interface



Input variables

Name	Type	Description
boExec	BOOL	Function execution: With a positive edge, the execution of the block starts. As long as 'boExec' = TRUE, the block is processed by the PLC. In the state 'boExec' = FALSE execution of the block is ended.
uiIDNo	UINT	ID number to be read out
uiParInst	UINT	Instance or Parameter set number or instance number
udAddress	UDINT	EtherCAT address

Output variables

Name	Type	Description
boDone	BOOL	Response that the function block has been completely executed.

Name	Type	Description
boErr	BOOL	The function block is in an error state FALSE: No error (permitted commanding or warning) TRUE: Error
iErrID	INT	Error identity number: Diagnostic number is output iErrID = 0: No error iErrID ≠ 0, boErr = TRUE: Error iErrID ≠ 0, boErr = FALSE: Warning Range: Siehe 'Error bit information' auf Seite 150.

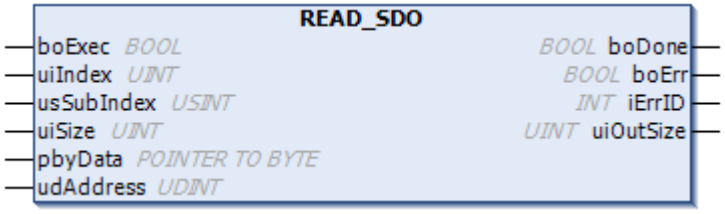
Input and output variables

Name	Type	Description
stList512	STRUCT	ST_LIST_512 List 512 Accommodates list information

2.1.8.2.5 READ_SDO (FB)

The function block 'READ_SDO' reads a value from a CAN object.

User interface



Input variables

Name	Type	Description
boExec	BOOL	Function execution: With a positive edge, the execution of the block starts. As long as 'boExec' = TRUE, the block is processed by the PLC. In the state 'boExec' = FALSE execution of the block is ended.
uiIndex	UINT	Index of the SDO whose value is being read
usSubIndex	USINT	Subindex of the SDO whose value is being read
uiSize	UINT	Maximum data length available to accommodate the information to be read. uiSize ≤ SIZEOF(variable) referenced by 'pbyData'!
pbyData	POINTER	POINTER TO READ DATA Pointer referencing the structure / variable which is receiving the information read.
udAddress	UDINT	EtherCAT address

Output variables

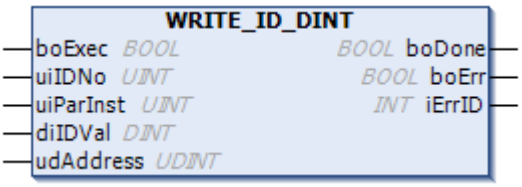
Name	Type	Description
boDone	BOOL	Response that the function block has been completely executed.

Name	Type	Description
boErr	BOOL	The function block is in an error state FALSE: No error (permitted commanding or warning) TRUE: Error
iErrID	INT	Error identity number: Diagnostic number is output iErrID = 0: No error iErrID ≠ 0 and boErr = TRUE: Error iErrID ≠ 0 and boErr = FALSE: Warning Range: Siehe 'Error bit information' auf Seite 150.
uiOutSize	UINT	Current data length entered (read) in the structure referenced by the 'pbyData' pointer.

2.1.8.2.6 WRITE_ID_DINT (FB)

The 'WRITE_ID_DINT' function block writes the value of a parameter stored in the AMK subsystem.

User interface



Input variables

Name	Type	Description
boExec	BOOL	Function execution: With a positive edge, the execution of the block starts. As long as 'boExec' = TRUE, the block is processed by the PLC. In the state 'boExec' = FALSE execution of the block is ended.
uiIDNo	UINT	ID number to be read out
uiParInst	UINT	Instance or Parameter set number or instance number
diIDVal	DINT	Parameter value written to database
udAddress	UDINT	EtherCAT address

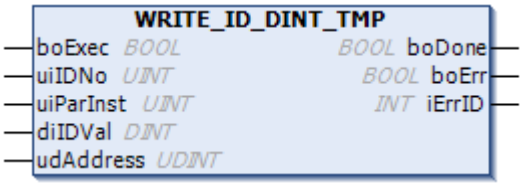
Output variables

Name	Type	Description
boDone	BOOL	Response that the function block has been completely executed.
boErr	BOOL	The function block is in an error state FALSE: No error (permitted commanding or warning) TRUE: Error
iErrID	INT	Error identity number: Diagnostic number is output iErrID = 0: No error iErrID ≠ 0 and boErr = TRUE: Error iErrID ≠ 0 and boErr = FALSE: Warning Range: Siehe 'Error bit information' auf Seite 150.

2.1.8.2.7 WRITE_ID_DINT_TMP (FB)

The 'WRITE_ID_DINT_TMP' function block writes the value of a parameter organized temporarily in the AMK subsystem.

User interface



Input variables

Name	Type	Description
boExec	BOOL	Function execution: With a positive edge, the execution of the block starts. As long as 'boExec' = TRUE, the block is processed by the PLC. In the state 'boExec' = FALSE execution of the block is ended.
uiIDNo	UINT	ID number to be read out
uiParInst	UINT	Instance or Parameter set number or instance number
diIDVal	DINT	Parameter value written to database
udAddress	UDINT	EtherCAT address

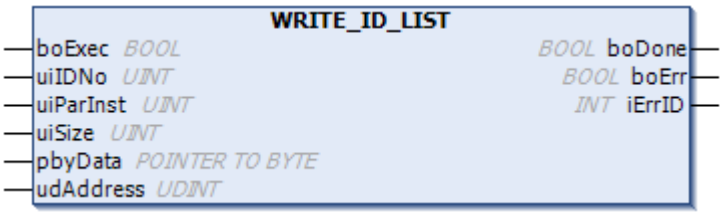
Output variables

Name	Type	Description
boDone	BOOL	Response that the function block has been completely executed.
boErr	BOOL	The function block is in an error state FALSE: No error (permitted commanding or warning) TRUE: Error
iErrID	INT	Error identity number: Diagnostic number is output iErrID = 0: No error iErrID ≠ 0, boErr = TRUE: Error iErrID ≠ 0, boErr = FALSE: Warning Range: Siehe 'Error bit information' auf Seite 150.

2.1.8.2.8 WRITE_ID_LIST (FB)

The 'WRITE_ID_LIST' function block writes values of a list parameter to the database of an AMK subsystem.

User interface



Input variables

Name	Type	Description
boExec	BOOL	Function execution: With a positive edge, the execution of the block starts. As long as 'boExec' = TRUE, the block is processed by the PLC. In the state 'boExec' = FALSE execution of the block is ended.
uiIDNo	UINT	ID number to be written
uiParInst	UINT	Instance or Parameter set number or instance number

Name	Type	Description
uiSize	UINT	Maximum data length of the information to be written. uiSize ≤ SIZEOF(variable) referenced by 'pbyData'!
pbyData	POINTER	POINTER TO WRITE DATA Pointer referencing the structure / variable which contains the information to be written.
udAddress	UDINT	EtherCAT address

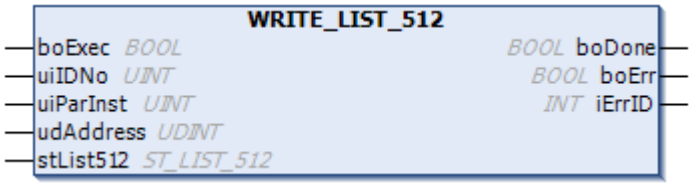
Output variables

Name	Type	Description
boDone	BOOL	Response that the function block has been completely executed.
boErr	BOOL	The function block is in an error state FALSE: No error (permitted commanding or warning) TRUE: Error
iErrID	INT	Error identity number: Diagnostic number is output iErrID = 0: No error iErrID ≠ 0, boErr = TRUE: Error iErrID ≠ 0, boErr = FALSE: Warning Range: Siehe 'Error bit information' auf Seite 150.

2.1.8.2.9 WRITE_LIST_512 (FB)

The 'WRITE_LIST_512' function block writes values of a list parameter that is up to 512 bytes in size to the database of an AMK subsystem.

User interface



Input variables

Name	Type	Description
boExec	BOOL	Function execution: With a positive edge, the execution of the block starts. As long as 'boExec' = TRUE, the block is processed by the PLC. In the state 'boExec' = FALSE execution of the block is ended.
uiIDNo	UINT	ID number to be written
uiParInst	UINT	Instance or Parameter set number or instance number
udAddress	UDINT	EtherCAT address

Output variables

Name	Type	Description
boDone	BOOL	Response that the function block has been completely executed.

Name	Type	Description
boErr	BOOL	The function block is in an error state FALSE: No error (permitted commanding or warning) TRUE: Error
iErrID	INT	Error identity number: Diagnostic number is output iErrID = 0: No error iErrID ≠ 0, boErr = TRUE: Error iErrID ≠ 0, boErr = FALSE: Warning Range: Siehe 'Error bit information' auf Seite 150.

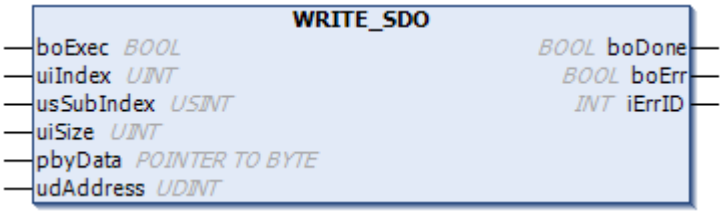
Input and output variables

Name	Type	Description
stList512	STRUCT	ST_LIST_512 List 512 Accommodates list information

2.1.8.2.10 WRITE_SDO (FB)

The function block 'WRITE_SDO' is used to write a value to a CAN object.

User interface



Input variables

Name	Type	Description
boExec	BOOL	Function execution: With a positive edge, the execution of the block starts. As long as 'boExec' = TRUE, the block is processed by the PLC. In the state 'boExec' = FALSE execution of the block is ended.
uiIndex	UINT	Index of the SDO whose value is being written
usSubIndex	USINT	Subindex of the SDO whose value is being written
uiSize	UINT	Maximum data length available to accommodate the information to be read. uiSize ≤ SIZEOF(variable) referenced by 'pbyData'!
pbyData	POINTER	POINTER TO WRITE DATA Pointer referencing the structure / variable which contains the information to be written.
udAddress	UDINT	EtherCAT address

Output variables

Name	Type	Description
boDone	BOOL	Response that the function block has been completely executed.

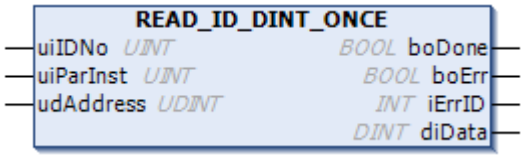
Name	Type	Description
boErr	BOOL	The function block is in an error state FALSE: No error (permitted commanding or warning) TRUE: Error
iErrID	INT	Error identity number: Diagnostic number is output iErrID = 0: No error iErrID ≠ 0, boErr = TRUE: Error iErrID ≠ 0, boErr = FALSE: Warning Range: Siehe 'Error bit information' auf Seite 150.

2.1.8.3 HigherAccess

2.1.8.3.1 READ_ID_DINT_ONCE (FB)

The 'READ_ID_DINT_ONCE' function block reads in the value of a parameter from the database of an AMK subsystem. The handshake 'boExec' / 'boDone' does not have to be organized.

User interface



Input variables

Name	Type	Description
uiIDNo	UINT	ID number to be read out
uiParInst	UINT	Instance or Parameter set number or instance number
udAddress	UDINT	EtherCAT address

Output variables

Name	Type	Description
boDone	BOOL	Response that the function block has been completely executed.
boErr	BOOL	The function block is in an error state FALSE: No error (permitted commanding or warning) TRUE: Error
iErrID	INT	Error identity number: Diagnostic number is output iErrID = 0: No error iErrID ≠ 0, boErr = TRUE: Error iErrID ≠ 0, boErr = FALSE: Warning Range: Siehe 'Error bit information' auf Seite 150.
diData	DINT	Parameter value read from database

Actions

Name	Description
Start	Read: The process is started with the start action and acknowledged with 'boDone' = TRUE • The acknowledgement not revoked until the next start action is underway • The input parameters must be specified before the start action is triggered

2.1.8.3.2 WRITE_ID_DINT_ONCE (FB)

The 'WRITE_ID_DINT_ONCE' function block writes the value of a parameter to the database of an AMK subsystem. The handshake 'boExec' / 'boDone' does not have to be organized.

User interface

Input variables

Name	Type	Description
uiIDNo	UINT	ID number to be written
uiParInst	UINT	Instance or Parameter set number or instance number
diData	DINT	Parameter value written to database
udAddress	UDINT	EtherCAT address

Output variables

Name	Type	Description
boDone	BOOL	Response that the function block has been completely executed.
boErr	BOOL	The function block is in an error state FALSE: No error (permitted commanding or warning) TRUE: Error
iErrID	INT	Error identity number: Diagnostic number is output iErrID = 0: No error iErrID ≠ 0, boErr = TRUE: Error iErrID ≠ 0, boErr = FALSE: Warning Range: Siehe 'Error bit information' auf Seite 150.

Actions

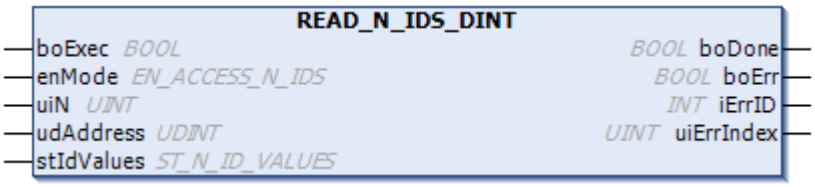
Name	Description
Start	Write: The process is started with the start action and acknowledged with 'boDone' = TRUE • The acknowledgement not revoked until the next start action is underway • The input parameters must be specified before the start action is triggered

2.1.8.4 MoreIds

2.1.8.4.1 READ_N_IDS_DINT (FB)

The 'READ_N_IDS_DINT' function block reads a defined number of parameter values from the database or temporary values of an AMK subsystem.

User interface



Input variables

Name	Type	Description
boExec	BOOL	Function execution: With a positive edge, the execution of the block starts. As long as 'boExec' = TRUE, the block is processed by the PLC. In the state 'boExec' = FALSE execution of the block is ended.
enMode	ENUM	EN_ACCESS_N_IDS Selection mode Definition of remanent or temporary parameters Default: ACCESS_N_IDS_DINT_TMP Range: ACCESS_N_IDS_DINT_TMP – Read temporary parameter value Range: ACCESS_N_IDS_DINT_REM – Read remanent parameter value
uiN	UINT	Number of parameters to be read Range: 1 ... MAX_INDEX_FOR_ID_VALUES
udAddress	UDINT	EtherCAT address

Output variables

Name	Type	Description
boDone	BOOL	Response that the function block has been completely executed.
boErr	BOOL	The function block is in an error state FALSE: No error (permitted commanding or warning) TRUE: Error
iErrID	INT	Error identity number: Diagnostic number is output iErrID = 0: No error iErrID ≠ 0, boErr = TRUE: Error iErrID ≠ 0, boErr = FALSE: Warning Range: Siehe 'Error bit information' auf Seite 150.

Name	Type	Description
uiErrIndex	UINT	An error occurred when reading in the value cited Range: 0 — No error Range: 1 ... MAX_INDEX_FOR_ID_VALUES — Index no. of the parameter affected by an error

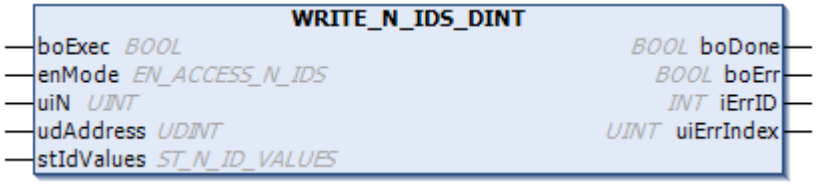
Input and output variables

Name	Type	Description
stIdValues	STRUCT	ST_N_ID_VALUES List of parameters that can be read / written.

2.1.8.4.2 WRITE_N_IDS_DINT (FB)

The 'WRITE_N_IDS_DINT' function block writes a defined number of parameter values to the database or temporary values of an AMK subsystem.

User interface



Input variables

Name	Type	Description
boExec	BOOL	Function execution: With a positive edge, the execution of the block starts. As long as 'boExec' = TRUE, the block is processed by the PLC. In the state 'boExec' = FALSE execution of the block is ended.
enMode	ENUM	EN_ACCESS_N_IDS Selection mode Definition of remanent or temporary parameters Default: ACCESS_N_IDS_DINT_TMP Range: ACCESS_N_IDS_DINT_TMP — Write temporary parameter value Range: ACCESS_N_IDS_DINT_REM — Write remanent parameter value
uiN	UINT	Number of parameters to be written Range: 1 ... MAX_INDEX_FOR_ID_VALUES
udAddress	UDINT	EtherCAT address

Output variables

Name	Type	Description
boDone	BOOL	Response that the function block has been completely executed.
boErr	BOOL	The function block is in an error state FALSE: No error (permitted commanding or warning) TRUE: Error

<table>
<tr><th>Name</th><th>Type</th><th>Description</th></tr>
<tr><td>iErrID</td><td>INT</td><td>Error identity number: Diagnostic number is output
<table>
<tr><td colspan="2">iErrID = 0</td><td>No error</td></tr>
<tr><td>iErrID ≠ 0</td><td>boErr = TRUE</td><td>Error</td></tr>
<tr><td>iErrID ≠ 0</td><td>boErr = FALSE</td><td>Warning</td></tr>
</table>
Range: Siehe 'Error bit information' auf Seite 150.</td></tr>
<tr><td>uiErrIndex</td><td>UINT</td><td>An error occurred when writing the value cited
<table>
<tr><th>Range</th><th>Meaning</th></tr>
<tr><td>0</td><td>No error</td></tr>
<tr><td>1 ... MAX_INDEX_FOR_ID_VALUES</td><td>Index no. of the parameter affected by an error</td></tr>
</table></td></tr>
</table>

Input and output variables

Name	Type	Description
stIdValues	STRUCT	ST_N_ID_VALUES List of parameters that can be read / written.

2.1.9 Support functions

The blocks are essentially intended for internal system development. They are used, for example, in superior quality blocks and made available to users in a format customized to meet the requirements of their applications.

ForInternalUse
Internal library functions

CLEAR_DINT LOCK_EXEC	The 'ForInternalUse' groups blocks that are required in the context of the internal function. These blocks are not relevant for the application in the context of application programming. Therefore, they are not described in more detail here.

Sercos

CMD_BY_ID	Executes ID-based commanding
CMD_START_STOP_BY_ID	Executes ID-based start / stop commanding
STATE_BY_ID	Checks the status of ID-based commanding

2.1.9.1 ForInternalUse

2.1.9.1.1 CLEAR_DINT (FB)

Reserved for AMK internal use!

2.1.9.1.2 LOCK_EXEC (FB)

Reserved for AMK internal use!

2.1.9.2 Sercos

2.1.9.2.1 CMD_BY_ID (FB)

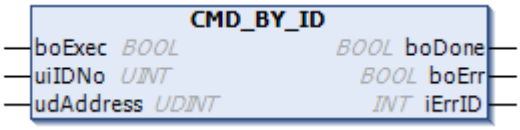
This block executes ID-based commanding via the EtherCAT bus. In accordance with to the SERCOS standard, a complete commanding cycle is executed referencing the ID number specified in the 'uiIDNo' input variable.

For a commandable ID (e.g. ID148 'Drive homing cycle command'):

- a value of 3 is written,
- a check is made to ascertain if 3 can be read back (no error, otherwise error code 15),

- a value of 0 is written,
- a check is made to ascertain if 0 can be read back (end)

User interface



Input variables

Name	Type	Description
boExec	BOOL	Function execution: With a positive edge, the execution of the block starts. As long as 'boExec' = TRUE, the block is processed by the PLC. In the state 'boExec' = FALSE execution of the block is ended.
uiIDNo	UINT	Parameter number (ID)
udAddress	UDINT	EtherCAT address

Output variables

Name	Type	Description
boDone	BOOL	Response that the function block has been completely executed.
boErr	BOOL	The function block is in an error state FALSE: No error (permitted commanding or warning) TRUE: Error
iErrID	INT	Error identity number: Diagnostic number is output iErrID = 0: No error iErrID ≠ 0, boErr = TRUE: Error iErrID ≠ 0, boErr = FALSE: Warning Error 15: Commanding error

2.1.9.2.2 CMD_START_STOP_BY_ID (FB)

This block executes ID-based start / stop commanding via the EtherCAT bus.

User interface

Input variables

Name	Type	Description
boExec	BOOL	Function execution: With a positive edge, the execution of the block starts. As long as 'boExec' = TRUE, the block is processed by the PLC. In the state 'boExec' = FALSE execution of the block is ended.
boStartNotStop	BOOL	Commanding ID number FALSE: Stop (code=0) TRUE: Start (code=3)

Name	Type	Description
uiIDNo	UINT	Parameter number (ID)
udAddress	UDINT	EtherCAT address

Output variables

Name	Type	Description
boDone	BOOL	Acknowledgement: Function block is initialised and enabled
boErr	BOOL	The function block is in an error state FALSE: No error (permitted commanding or warning) TRUE: Error
iErrID	INT	Error identity number: Diagnostic number is output iErrID = 0: No error iErrID ≠ 0, boErr = TRUE: Error iErrID ≠ 0, boErr = FALSE: Warning Error Value 15: Commanding error

2.1.9.2.3 STATE_BY_ID (FB)

This block checks the status of ID-based commanding via the EtherCAT bus.

User interface

Input variables

Name	Type	Description
boEnable	BOOL	Enable signal: With a positive edge, the initialisation of the block starts. As long as 'boEnable' = TRUE, the block remains enabled and is processed by the PLC. In the state 'boEnable' = FALSE the block is no longer enabled and is thus no longer processed. If 'boEnable' = TRUE, the ID is read continuously
uiIDNo	UINT	Parameter number (ID)
udAddress	UDINT	EtherCAT address

Output variables

Name	Type	Description
boEnabAck	BOOL	Response that the function block has been completely executed.
boErr	BOOL	The function block is in an error state FALSE: No error (permitted commanding or warning) TRUE: Error

Name	Type	Description
iErrID	INT	Error identity number: Diagnostic number is output iErrID = 0 — No error iErrID ≠ 0 — boErr = TRUE — Error iErrID ≠ 0 — boErr = FALSE — Warning
diData	DINT	Parameter value 0 — Inactive 3 — Active 7 — Idle 15 — Error

2.1.10 TabCalc - AMK table calculation blocks

TabCalc containing blocks for calculating special table profiles. The basis is provided by the 'TAB_CALC' block.
TabCalc is divided into the following table types:

OperatingTables	Operating tables
PhasingInTables	Phasing in tables
PhasingOutTables	Phasing out tables
PositioningProfiles	Positioning profiles
Support	Support functions

2.1.10.1 Operating tables

CALC_OP	Calculation of the operating table

2.1.10.1.1 CALC_OP (FB)

The 'CALC_OP' function block calculates the operating table based on a synchronous straight line to the start of a $\sin^2$ smoothing function with tangential merging.

User interface

Input variables

Name	Type	Description
boExec	BOOL	Function execution: With a positive edge, the execution of the block starts. As long as 'boExec' = TRUE, the block is processed by the PLC. In the state 'boExec' = FALSE execution of the block is ended.
udMasterInc	UDINT	Increments of the master drive which produce a table cycle Max. table X value Range: 0 ... 5000000 Unit: incr

Name	Type	Description
diOutInterv	DINT	Output interface defining the output increments per table cycle Max. table Y value
uiNoElement	UINT	Element number of the last table element calculated, number of table interpolation points
uiXSin	UINT	Sine start x position [°] at which the change from synchronous straight line to sin^2 smoothing function takes place
uiSync	UINT	Synchronous factor Ratio of output increments to input increments Incline of synchronous straight line; uiSync := 100 corresponds to an incline of 1 Range: 100 ... 32767 Unit: %

Output variables

Name	Type	Description
boDone	BOOL	Response that the function block has been completely executed.
boErr	BOOL	The function block is in an error state FALSE: No error (permitted commanding or warning) TRUE: Error
iErrID	INT	Error identity number: Diagnostic number is output iErrID = 0: No error iErrID ≠ 0, boErr = TRUE: Error iErrID ≠ 0, boErr = FALSE: Warning Error 1: Incorrect number of elements the maximum number is dependent on the table type 'enTabType' 2: Incorrect parameter set variant dependent on the table type 'enTabKind' 3: 'udMasterInc' value too high 4: 'diOutInterv' value too high / too low 5: 'diPar1' illegal value dependent on 'enTabType' and 'enTabKind' 6: 'diPar2' illegal value dependent on 'enTabType' and 'enTabKind' 7: 'diPar3' illegal value dependent on 'enTabType' and 'enTabKind' 8: 'diPar4' illegal value dependent on 'enTabType' and 'enTabKind' 9: Illegal synchronous point 10: Illegal phasing in point 11: Illegal phasing out point 12: Illegal sine starting point 13: Velocity too low 14: Acceleration too low

Input and output variables

Name	Type	Description
stDestTab	STRUCT	ST_PROF_TAB Profile table structure

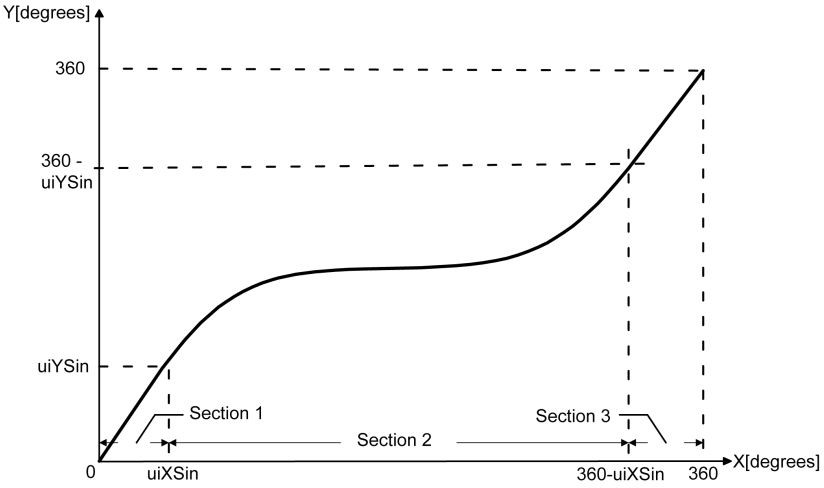
Description

The position setpoints for the periodic movement of an axis are stored in the operating table.

The output values of the operating table (y coordinates) are assigned to the incoming increments of a master axis (x coordinates). This assignment is made with the 'CAM_PROF_1' block. The table is calculated in the Y table format.

The output values for the operating table are calculated so that the process will start with linear movement with a variable synchronous factor 'uiSync'. Harmonic smoothing movement is calculated with a $\sin^2$ function.

Abbildung 24: CALC_OP: operating function



Section 1	0 ≤ x < uiXSin
Section 2	uiXSin ≤ x < 360-uiXSin
Section 3	360-uiXSin ≤ x < 360

2.1.10.2 Phasing in tables

A phasing in table phases a stationary axis into a movement sequence which is controlled by the 'CAM_PROF_1' block, for example.

CALC_IN_ALLDEF	Calculation of the phasing in table based on phasing in point and synchronous point
CALC_IN_INDEF	Calculation of the phasing in table based on phasing in point and synchronous ratio
CALC_IN_SYNCDEF	Calculation of the phasing in table based on phasing in y position and synchronous point

2.1.10.2.1 CALC_IN_ALLDEF (FB)

The 'CALC_IN_ALLDEF' function block calculates the phasing in table based on the phasing in point and the synchronous point. The phasing in table calculated is based on two parabolic partial sections with tangential transition and tangential merging into the synchronous straight line.

User interface

Input variables

Name	Type	Description
boExec	BOOL	Function execution: With a positive edge, the execution of the block starts. As long as 'boExec' = TRUE, the block is processed by the PLC. In the state 'boExec' = FALSE execution of the block is ended.
udMasterInc	UDINT	Increments of the master drive which produce a table cycle Max. table X value Range: 0 ... 5000000 Unit: incr
diOutInterv	DINT	Output interface defining the output increments per table cycle Max. table Y value Unit: incr
uiNoElement	UINT	Element number of the last table element calculated, number of table interpolation points Range: 5 ... ((SIZEOF(ST_PROF_TAB)-8)/4)-1
uiXIn	UINT	Phasing in point x coordinate Phasing in starts from this position, transition to parabola Range: 0 ... 360 Unit: °
iYIn	INT	Phasing in point y coordinate Phasing in starts from this position, transition to parabola Range: -360 ... 360 Unit: °
uiXSync	UINT	Synchronous point x coordinate The change to the synchronous straight line starts from this position Range: 0 ... 360 Unit: °
uiSync	UINT	Synchronous factor Ratio of output increments to input increments Incline of synchronous straight line; uiSync := 100 corresponds to an incline of 1 Range: 100 ... 32767 Unit: %

Output variables

Name	Type	Description
boDone	BOOL	Response that the function block has been completely executed.

<table>
<tr><th>Name</th><th>Type</th><th>Description</th></tr>
<tr><td>boErr</td><td>BOOL</td><td>The function block is in an error state
<table>
<tr><td>FALSE</td><td>No error (permitted commanding or warning)</td></tr>
<tr><td>TRUE</td><td>Error</td></tr>
</table></td></tr>
<tr><td>iErrID</td><td>INT</td><td>Error identity number: Diagnostic number is output
<table>
<tr><td colspan="2">iErrID = 0</td><td>No error</td></tr>
<tr><td>iErrID ≠ 0</td><td>boErr = TRUE</td><td>Error</td></tr>
<tr><td>iErrID ≠ 0</td><td>boErr = FALSE</td><td>Warning</td></tr>
</table>
Error
<table>
<tr><th>Value</th><th>Meaning</th></tr>
<tr><td>1</td><td>Incorrect number of elements
the maximum number is dependent on the table type 'enTabType'</td></tr>
<tr><td>2</td><td>Incorrect parameter set variant
dependent on the table type 'enTabKind'</td></tr>
<tr><td>3</td><td>'udMasterInc' value too high</td></tr>
<tr><td>4</td><td>'diOutInterv' value too high / too low</td></tr>
<tr><td>5</td><td>'diPar1' illegal value
dependent on 'enTabType' and 'enTabKind'</td></tr>
<tr><td>6</td><td>'diPar2' illegal value
dependent on 'enTabType' and 'enTabKind'</td></tr>
<tr><td>7</td><td>'diPar3' illegal value
dependent on 'enTabType' and 'enTabKind'</td></tr>
<tr><td>8</td><td>'diPar4' illegal value
dependent on 'enTabType' and 'enTabKind'</td></tr>
<tr><td>9</td><td>Illegal synchronous point</td></tr>
<tr><td>10</td><td>Illegal phasing in point</td></tr>
<tr><td>11</td><td>Illegal phasing out point</td></tr>
<tr><td>12</td><td>Illegal sine starting point</td></tr>
<tr><td>13</td><td>Velocity too low</td></tr>
<tr><td>14</td><td>Acceleration too low</td></tr>
</table></td></tr>
</table>

Input and output variables

Name	Type	Description
stDestTab	STRUCT	ST_PROF_TAB Profile table structure

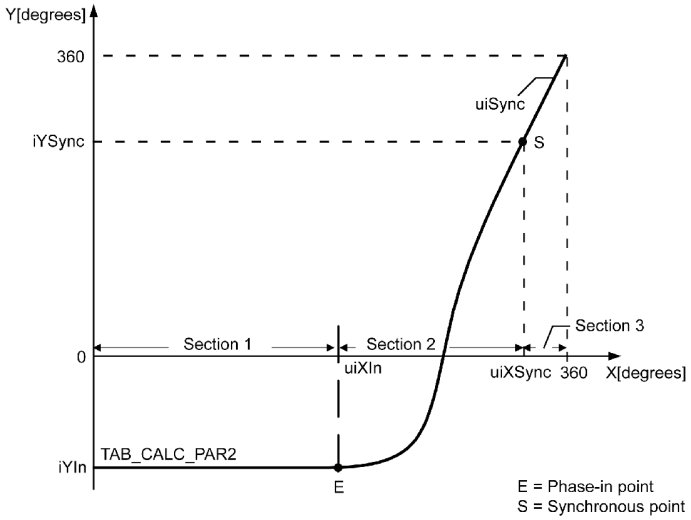
Description

The values for the phasing in table are calculated using two parabolas. This results in an extended definition range for the parameters of the phasing in process. Accordingly, the start value for the phasing in process can be less than 0 degrees.

The parabolas are calculated so that the acceleration value remains constant throughout the phasing in process.

The input parameters are the x and y coordinates of the phasing in point, the x coordinate of the synchronous point, and the synchronous factor.
The table is calculated in the Y table format.

Abbildung 25: CALC_IN_ALLDEF: Phasing in function with defined phasing in and synchronous points



Section 1	0 ≤ x < uiXIn	
Section 2	uiXIn ≤ x < uiXSync	Phasing in process: Two symmetrical parabolas $y = ax^2$ First parabola: vertex at 'uiXIn' Second parabola: vertex at 'uiXVer'
Section 3	uiXSync ≤ x < 360	

Calculation of the parabolas

The factor a of the second parabola $y = -a(x-uiXVer)^2$ is calculated from the equation y' = uiSync/100 = -2a(x-uiXVer).
In the synchronous point S:

uiXVer: x coordinate of the vertex of the second parabola

The following dependencies and conditions apply:

-
-
-

2.1.10.2.2 CALC_IN_INDEF (FB)

The 'CALC_IN_INDEF' function block calculates the phasing in table based on the phasing in point and the synchronous ratio. The calculated phasing in table is based on a partial section of a parabola with tangential merging into the synchronous straight line.

User interface

Input variables

Name	Type	Description
boExec	BOOL	Function execution: With a positive edge, the execution of the block starts. As long as 'boExec' = TRUE, the block is processed by the PLC. In the state 'boExec' = FALSE execution of the block is ended.

Name	Type	Description
udMasterInc	UDINT	Increments of the master drive which produce a table cycle Max. table X value Range: 0 ... 5000000 Unit: incr
diOutInterv	DINT	Output interface defining the output increments per table cycle Max. table Y value Unit: incr
uiNoElement	UINT	Element number of the last table element calculated, number of table interpolation points Range: 5 ... ((SIZEOF(ST_PROF_TAB)-8)/4)-1
uiXIn	UINT	Phasing in point x coordinate Phasing in starts from this position, transition to parabola Range: 0 ... 360 Unit: °
iYIn	INT	Phasing in point y coordinate Phasing in starts from this position, transition to parabola Range: -360 ... 360 Unit: °
uiSync	UINT	Synchronous factor Ratio of output increments to input increments Incline of synchronous straight line; uiSync := 100 corresponds to an incline of 1 Range: 100 ... 32767 Unit: %

Output variables

Name	Type	Description
boDone	BOOL	Response that the function block has been completely executed.
boErr	BOOL	The function block is in an error state FALSE: No error (permitted commanding or warning) TRUE: Error

<table>
<tr><th>Name</th><th>Type</th><th>Description</th></tr>
<tr><td>iErrID</td><td>INT</td><td>Error identity number: Diagnostic number is output
<table>
<tr><td colspan="2">iErrID = 0</td><td>No error</td></tr>
<tr><td>iErrID ≠ 0</td><td>boErr = TRUE</td><td>Error</td></tr>
<tr><td>iErrID ≠ 0</td><td>boErr = FALSE</td><td>Warning</td></tr>
</table>
Error
<table>
<tr><th>Value</th><th>Meaning</th></tr>
<tr><td>1</td><td>Incorrect number of elements
the maximum number is dependent on the table type 'enTabType'</td></tr>
<tr><td>2</td><td>Incorrect parameter set variant
dependent on the table type 'enTabKind'</td></tr>
<tr><td>3</td><td>'udMasterInc' value too high</td></tr>
<tr><td>4</td><td>'diOutInterv' value too high / too low</td></tr>
<tr><td>5</td><td>'diPar1' illegal value
dependent on 'enTabType' and 'enTabKind'</td></tr>
<tr><td>6</td><td>'diPar2' illegal value
dependent on 'enTabType' and 'enTabKind'</td></tr>
<tr><td>7</td><td>'diPar3' illegal value
dependent on 'enTabType' and 'enTabKind'</td></tr>
<tr><td>8</td><td>'diPar4' illegal value
dependent on 'enTabType' and 'enTabKind'</td></tr>
<tr><td>9</td><td>Illegal synchronous point</td></tr>
<tr><td>10</td><td>Illegal phasing in point</td></tr>
<tr><td>11</td><td>Illegal phasing out point</td></tr>
<tr><td>12</td><td>Illegal sine starting point</td></tr>
<tr><td>13</td><td>Velocity too low</td></tr>
<tr><td>14</td><td>Acceleration too low</td></tr>
</table></td></tr>
<tr><td>uiXSync</td><td>UINT</td><td>Synchronous point x coordinate
The change to the synchronous straight line starts from this position
<table>
<tr><td>Unit</td><td>°</td></tr>
</table></td></tr>
</table>

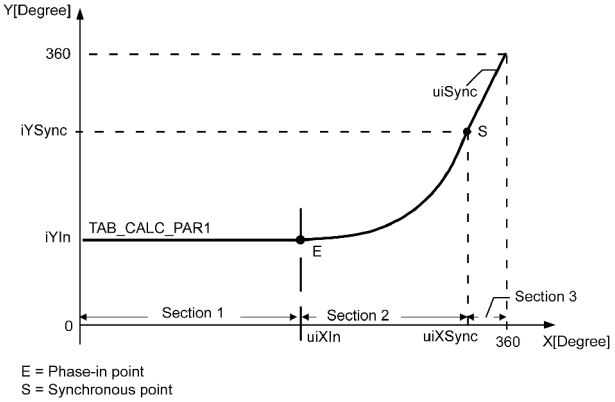
Input and output variables

Name	Type	Description
stDestTab	STRUCT	ST_PROF_TAB Profile table structure

Description

For the 'CALC_IN_INDEF' block, the phasing in curve is calculated with a parabola. Here, the input parameters are the x and y coordinates of the phasing in point, the x coordinate of the synchronous point, and the synchronous factor. The table is calculated in the Y table format.

Abbildung 26: CALC_IN_INDEF: Phasing in function with defined phasing in point



Section 1	0 ≤ x < uiXIn	
Section 2	uiXIn ≤ x < uiXSync	Phasing in process: parabola $y = ax^2$
Section 3	uiXSync ≤ x < 360	

Calculation of the parabola

The factor a of the parabola $y = -ax^2$ is calculated from the equation y' = uiSync/100 = 2ax .
In the synchronous point S:

Therefore, it follows that for a:

where

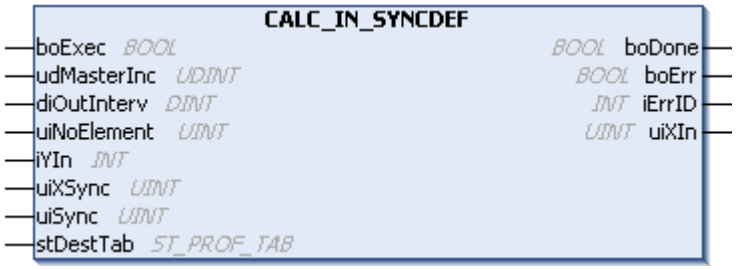
The following dependencies and conditions apply:

-
-

2.1.10.2.3 CALC_IN_SYNCDEF (FB)

The 'CALC_IN_SYNCDEF' function block calculates the phasing in table based on the y coordinates of the phasing in position and the synchronous point.
The phasing in table calculated is based on a parabolic partial section with tangential merging into the synchronous straight line.

User interface



Input variables

Name	Type	Description
boExec	BOOL	Function execution: With a positive edge, the execution of the block starts. As long as 'boExec' = TRUE, the block is processed by the PLC. In the state 'boExec' = FALSE execution of the block is ended.
udMasterInc	UDINT	Increments of the master drive which produce a table cycle Max. table X value Range: 0 ... 5000000 Unit: incr

Name	Type	Description
diOutInterv	DINT	Output interface defining the output increments per table cycle Max. table Y value Unit: incr
uiNoElement	UINT	Element number of the last table element calculated, number of table interpolation points Range: 5 ... ((SIZEOF(ST_PROF_TAB)-8)/4)-1
iYIn	INT	Phasing in point y coordinate Phasing in starts from this position, transition to parabola Range: -360 ... 360 Unit: °
uiXSync	UINT	Synchronous point x coordinate The change to the synchronous straight line starts from this position Range: 0 ... 360 Unit: °
uiSync	UINT	Synchronous factor Ratio of output increments to input increments Incline of synchronous straight line; uiSync := 100 corresponds to an incline of 1 Range: 100 ... 32767 Unit: %

Output variables

Name	Type	Description
boDone	BOOL	Response that the function block has been completely executed.
boErr	BOOL	The function block is in an error state FALSE: No error (permitted commanding or warning) TRUE: Error

<table>
<tr><th>Name</th><th>Type</th><th>Description</th></tr>
<tr><td>iErrID</td><td>INT</td><td>Error identity number: Diagnostic number is output
<table>
<tr><td colspan="2">iErrID = 0</td><td>No error</td></tr>
<tr><td>iErrID ≠ 0</td><td>boErr = TRUE</td><td>Error</td></tr>
<tr><td>iErrID ≠ 0</td><td>boErr = FALSE</td><td>Warning</td></tr>
</table>
Error
<table>
<tr><th>Value</th><th>Meaning</th></tr>
<tr><td>1</td><td>Incorrect number of elements
the maximum number is dependent on the table type 'enTabType'</td></tr>
<tr><td>2</td><td>Incorrect parameter set variant
dependent on the table type 'enTabKind'</td></tr>
<tr><td>3</td><td>'udMasterInc' value too high</td></tr>
<tr><td>4</td><td>'diOutInterv' value too high / too low</td></tr>
<tr><td>5</td><td>'diPar1' illegal value
dependent on 'enTabType' and 'enTabKind'</td></tr>
<tr><td>6</td><td>'diPar2' illegal value
dependent on 'enTabType' and 'enTabKind'</td></tr>
<tr><td>7</td><td>'diPar3' illegal value
dependent on 'enTabType' and 'enTabKind'</td></tr>
<tr><td>8</td><td>'diPar4' illegal value
dependent on 'enTabType' and 'enTabKind'</td></tr>
<tr><td>9</td><td>Illegal synchronous point</td></tr>
<tr><td>10</td><td>Illegal phasing in point</td></tr>
<tr><td>11</td><td>Illegal phasing out point</td></tr>
<tr><td>12</td><td>Illegal sine starting point</td></tr>
<tr><td>13</td><td>Velocity too low</td></tr>
<tr><td>14</td><td>Acceleration too low</td></tr>
</table></td></tr>
<tr><td>uiXIn</td><td>UINT</td><td>Phasing in point x coordinate
Phasing in starts from this position, transition to parabola
<table>
<tr><td>Unit</td><td>°</td></tr>
</table></td></tr>
</table>

Input and output variables

Name	Type	Description
stDestTab	STRUCT	ST_PROF_TAB Profile table structure

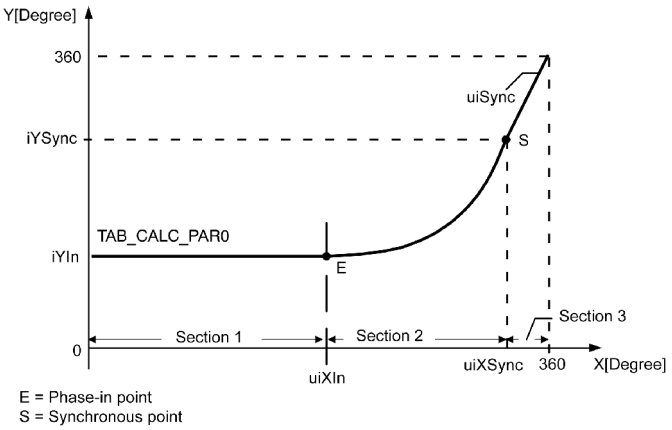
Description

The phasing in table is calculated with a parabola.

The input parameters are the y coordinates of the phasing in point, the x coordinate of the synchronous point, and the synchronous factor.

The table is calculated in the Y table format.

Abbildung 27: CALC_IN_SYNCDEF: Phasing in function with defined phasing in and synchronous points



Section 1	0 ≤ x < uiXIn	
Section 2	uiXIn ≤ x < uiXSync	Phasing in process: parabola $y = ax^2$
Section 3	uiXSync ≤ x < 360	

Calculation of the parabola

The factor a of the parabola $y = -ax^2$ is calculated from the equation y' = uiSync/100 = 2ax .
In the synchronous point S:

Therefore, it follows that for a:

The following dependencies and conditions apply:

-
-

2.1.10.3 Phasing out tables

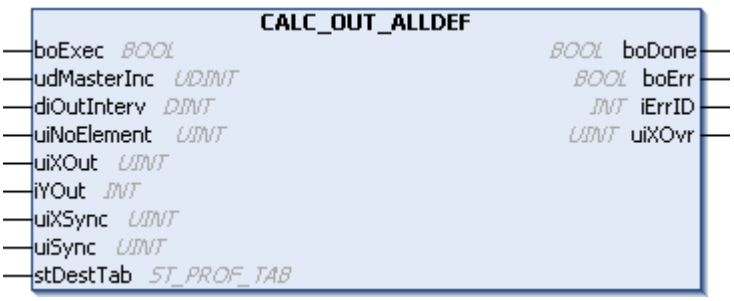
If an axis is moved in an operating table with the help of the 'CAM_PROF_1' block, for example, the phasing out table serves the purpose of phasing out from this sequence and stopping in a defined angular position.

CALC_OUT_ALLDEF	Calculation of the phasing out table based on phasing out point and synchronous point
CALC_OUT_OUTDEF	Calculation of the phasing out table based on phasing out point and synchronous ratio
CALC_OUT_SYNCDEF	Calculation of the phasing out table based on the y value of the phasing out position and synchronous point

2.1.10.3.1 CALC_OUT_ALLDEF (FB)

The 'CALC_OUT_ALLDEF' function block calculates the phasing out table based on the phasing out point and the synchronous point.
The phasing out table calculated is based on two parabolic partial sections with tangential transition and tangential exit out of the synchronous straight line.

User interface



Input variables

Name	Type	Description
boExec	BOOL	Function execution: With a positive edge, the execution of the block starts. As long as 'boExec' = TRUE, the block is processed by the PLC. In the state 'boExec' = FALSE execution of the block is ended.
udMasterInc	UDINT	Increments of the master drive which produce a table cycle Max. table X value Range: 0 ... 5000000 Unit: incr
diOutInterv	DINT	Output interface defining the output increments per table cycle Max. table Y value Unit: incr
uiNoElement	UINT	Element number of the last table element calculated, number of table interpolation points Range: 5 ... ((SIZEOF(ST_PROF_TAB)-8)/4)-1
uiXOut	UINT	Phasing out point x coordinate Phasing out ends with this position, transition from parabola to standstill Range: 0 ... 360 Unit: °
iYOut	INT	Phasing out point y coordinate Phasing out ends with this position, transition from parabola to standstill Range: -360 ... 360 Unit: °
uiXSync	UINT	Synchronous point x coordinate The synchronous straight line ends from this position Range: 0 ... 360 Unit: °
uiSync	UINT	Synchronous factor Ratio of output increments to input increments Incline of synchronous straight line; uiSync := 100 corresponds to an incline of 1 Range: 100 ... 32767 Unit: %

Output variables

Name	Type	Description
boDone	BOOL	Response that the function block has been completely executed.
boErr	BOOL	The function block is in an error state FALSE: No error (permitted commanding or warning) TRUE: Error

<table>
<tr><th>Name</th><th>Type</th><th>Description</th></tr>
<tr><td>iErrID</td><td>INT</td><td>Error identity number: Diagnostic number is output
<table>
<tr><td colspan="2">iErrID = 0</td><td>No error</td></tr>
<tr><td>iErrID ≠ 0</td><td>boErr = TRUE</td><td>Error</td></tr>
<tr><td>iErrID ≠ 0</td><td>boErr = FALSE</td><td>Warning</td></tr>
</table>
Error
<table>
<tr><th>Value</th><th>Meaning</th></tr>
<tr><td>1</td><td>Incorrect number of elements
the maximum number is dependent on the table type 'enTabType'</td></tr>
<tr><td>2</td><td>Incorrect parameter set variant
dependent on the table type 'enTabKind'</td></tr>
<tr><td>3</td><td>'udMasterInc' value too high</td></tr>
<tr><td>4</td><td>'diOutInterv' value too high / too low</td></tr>
<tr><td>5</td><td>'diPar1' illegal value
dependent on 'enTabType' and 'enTabKind'</td></tr>
<tr><td>6</td><td>'diPar2' illegal value
dependent on 'enTabType' and 'enTabKind'</td></tr>
<tr><td>7</td><td>'diPar3' illegal value
dependent on 'enTabType' and 'enTabKind'</td></tr>
<tr><td>8</td><td>'diPar4' illegal value
dependent on 'enTabType' and 'enTabKind'</td></tr>
<tr><td>9</td><td>Illegal synchronous point</td></tr>
<tr><td>10</td><td>Illegal phasing in point</td></tr>
<tr><td>11</td><td>Illegal phasing out point</td></tr>
<tr><td>12</td><td>Illegal sine starting point</td></tr>
<tr><td>13</td><td>Velocity too low</td></tr>
<tr><td>14</td><td>Acceleration too low</td></tr>
</table></td></tr>
<tr><td>uiXOvr</td><td>UINT</td><td>Maximum overshoot
Corresponds to the y position of the vertex of the first parabola in position 'uiXVer'</td></tr>
</table>

Input and output variables

Name	Type	Description
stDestTab	STRUCT	ST_PROF_TAB Profile table structure

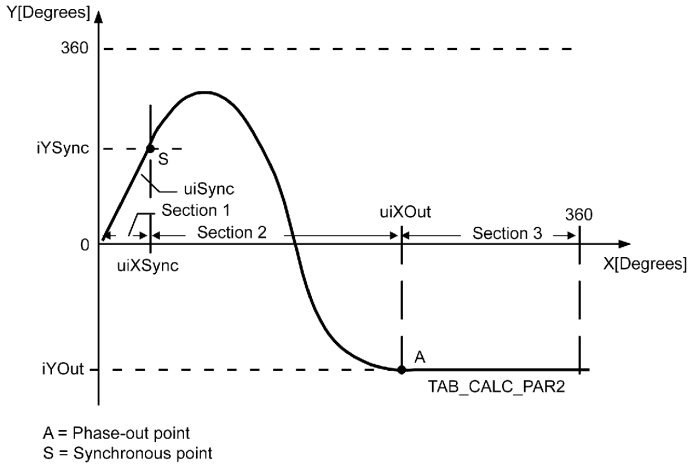
Description

The values of the phasing out table are calculated using two parabolas. This results in an extended definition range for the parameters of the phasing out process. Accordingly, the final value for the phasing out process can be less than 0 degrees.

The vertex of the first parabola is calculated so that the acceleration value remains constant throughout the phasing out process.

The input parameters are the x and y coordinates of the phasing out point, the x coordinate of the synchronous point, and the synchronous factor.
The table is calculated in the Y table format.

Abbildung 28: CALC_OUT_ALLDEF: Phasing out function with defined phasing out and synchronous points



Section 1	0 ≤ x < uiXSync	
Section 2	uiXSync ≤ x < uiXOut	Phasing out process: Two symmetrical parabolas $y = ax^2$ First parabola: vertex at 'uiXIn' Second parabola: vertex at 'uiXVer'
Section 3	uiXOut ≤ x < 360	y=iYOut

Calculation of the parabolas

The factor a of the first parabola $y = -a(x-uiXVer)^2$ is calculated from the equation $y' = -2a(x-uiXVer)$.
In the synchronous point S:

uiXVer: x coordinate of the vertex of the first parabola

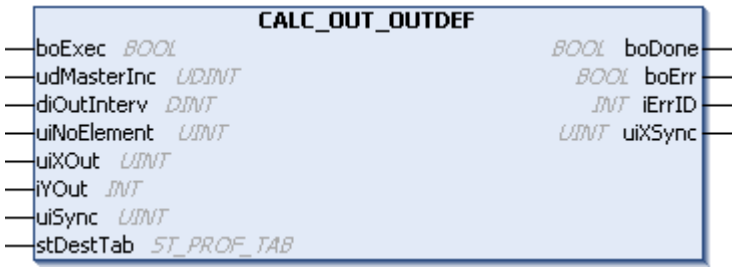
The following dependencies and conditions apply:

-
-

2.1.10.3.2 CALC_OUT_OUTDEF (FB)

The 'CALC_IN_OUTDEF' function block calculates the phasing out table based on the phasing out point and the synchronous point.
The calculated phasing out table is based on a partial section of a parabola with tangential exit out of the synchronous straight line.

User interface



Input variables

Name	Type	Description
boExec	BOOL	Function execution: With a positive edge, the execution of the block starts. As long as 'boExec' = TRUE, the block is processed by the PLC. In the state 'boExec' = FALSE execution of the block is ended.

Name	Type	Description
udMasterInc	UDINT	Increments of the master drive which produce a table cycle Max. table X value Range: 0 ... 5000000 Unit: incr
diOutInterv	DINT	Output interface defining the output increments per table cycle Max. table Y value Unit: incr
uiNoElement	UINT	Element number of the last table element calculated, number of table interpolation points Range: 5 ... ((SIZEOF(ST_PROF_TAB)-8)/4)-1
uiXOut	UINT	Phasing out point x coordinate Phasing out ends with this position, transition from parabola to standstill Range: 0 ... 360 Unit: °
iYOut	INT	Phasing out point y coordinate Phasing out ends with this position, transition from parabola to standstill Range: -360 ... 360 Unit: °
uiSync	UINT	Synchronous factor Ratio of output increments to input increments Incline of synchronous straight line; uiSync := 100 corresponds to an incline of 1 Range: 100 ... 32767 Unit: %

Output variables

Name	Type	Description
boDone	BOOL	Response that the function block has been completely executed.
boErr	BOOL	The function block is in an error state FALSE: No error (permitted commanding or warning) TRUE: Error

<table>
<tr><th>Name</th><th>Type</th><th>Description</th></tr>
<tr><td>iErrID</td><td>INT</td><td>Error identity number: Diagnostic number is output
<table>
<tr><td colspan="2">iErrID = 0</td><td>No error</td></tr>
<tr><td>iErrID ≠ 0</td><td>boErr = TRUE</td><td>Error</td></tr>
<tr><td>iErrID ≠ 0</td><td>boErr = FALSE</td><td>Warning</td></tr>
</table>
Error
<table>
<tr><th>Value</th><th>Meaning</th></tr>
<tr><td>1</td><td>Incorrect number of elements
the maximum number is dependent on the table type 'enTabType'</td></tr>
<tr><td>2</td><td>Incorrect parameter set variant
dependent on the table type 'enTabKind'</td></tr>
<tr><td>3</td><td>'udMasterInc' value too high</td></tr>
<tr><td>4</td><td>'diOutInterv' value too high / too low</td></tr>
<tr><td>5</td><td>'diPar1' illegal value
dependent on 'enTabType' and 'enTabKind'</td></tr>
<tr><td>6</td><td>'diPar2' illegal value
dependent on 'enTabType' and 'enTabKind'</td></tr>
<tr><td>7</td><td>'diPar3' illegal value
dependent on 'enTabType' and 'enTabKind'</td></tr>
<tr><td>8</td><td>'diPar4' illegal value
dependent on 'enTabType' and 'enTabKind'</td></tr>
<tr><td>9</td><td>Illegal synchronous point</td></tr>
<tr><td>10</td><td>Illegal phasing in point</td></tr>
<tr><td>11</td><td>Illegal phasing out point</td></tr>
<tr><td>12</td><td>Illegal sine starting point</td></tr>
<tr><td>13</td><td>Velocity too low</td></tr>
<tr><td>14</td><td>Acceleration too low</td></tr>
</table></td></tr>
<tr><td>uiXSync</td><td>UINT</td><td>Synchronous point x coordinate
The synchronous straight line ends from this position
<table>
<tr><td>Unit</td><td>°</td></tr>
</table></td></tr>
</table>

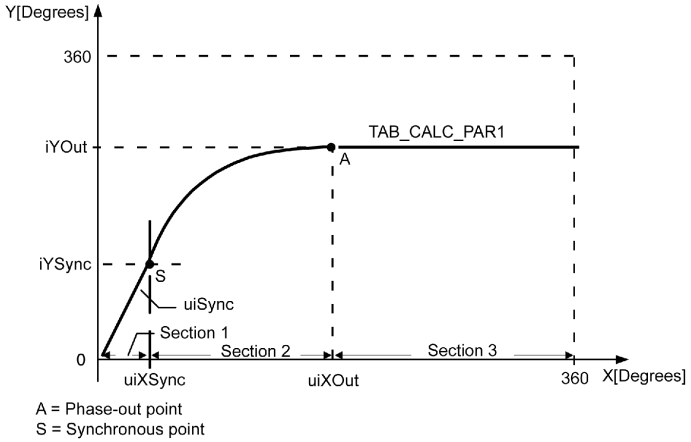
Input and output variables

Name	Type	Description
stDestTab	STRUCT	ST_PROF_TAB Profile table structure

Description

For the 'CALC_OUT_OUTDEF' block, the phasing out curve is calculated with a parabola. The input parameters are the x and y coordinates of the phasing out point and the synchronous factor. The table is calculated in the Y table format.

Abbildung 29: CALC_OUT_OUTDEF: Phasing out function with defined phasing out point



Section 1	0 ≤ x < uiXSync	
Section 2	uiXSync ≤ x < uiXOut	Phasing out process: Parabola $y = -ax^2$
Section 3	uiXOut ≤ x < 360	

Calculation of the parabola

The factor a of the parabola $y = -ax^2$ is calculated from the equation y' = uiSync/100 = -2ax .
In the synchronous point S:

Therefore, it follows that for a:

where

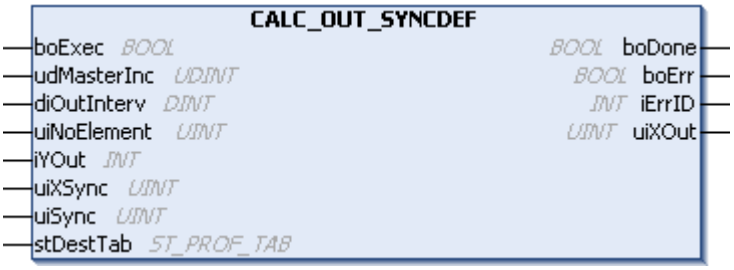
The following dependencies and conditions apply:

-
-

2.1.10.3.3 CALC_OUT_SYNCDEF (FB)

The 'CALC_OUT_SYNCDEF' function block calculates the phasing out table based on the y coordinate and the phasing out position.
The phasing out table calculated is based on a parabolic partial section with tangential exit out of the synchronous straight line.

User interface



Input variables

Name	Type	Description
boExec	BOOL	Function execution: With a positive edge, the execution of the block starts. As long as 'boExec' = TRUE, the block is processed by the PLC. In the state 'boExec' = FALSE execution of the block is ended.
udMasterInc	UDINT	Increments of the master drive which produce a table cycle Max. table X value Range: 0 ... 5000000 Unit: incr

Name	Type	Description
diOutInterv	DINT	Output interface defining the output increments per table cycle Max. table Y value Unit: incr
uiNoElement	UINT	Element number of the last table element calculated, number of table interpolation points Range: 5 ... ((SIZEOF(ST_PROF_TAB)-8)/4)-1
iYOut	INT	Phasing out point y coordinate Phasing out ends with this position, transition from parabola to standstill Range: -360 ... 360 Unit: °
uiXSync	UINT	Synchronous point x coordinate The synchronous straight line ends from this position Unit: °
uiSync	UINT	Synchronous factor Ratio of output increments to input increments Incline of synchronous straight line; uiSync := 100 corresponds to an incline of 1 Range: 100 ... 32767 Unit: %

Output variables

Name	Type	Description
boDone	BOOL	Response that the function block has been completely executed.
boErr	BOOL	The function block is in an error state FALSE: No error (permitted commanding or warning) TRUE: Error

<table>
<tr><th>Name</th><th>Type</th><th>Description</th></tr>
<tr><td>iErrID</td><td>INT</td><td>Error identity number: Diagnostic number is output
<table>
<tr><td colspan="2">iErrID = 0</td><td>No error</td></tr>
<tr><td>iErrID ≠ 0</td><td>boErr = TRUE</td><td>Error</td></tr>
<tr><td>iErrID ≠ 0</td><td>boErr = FALSE</td><td>Warning</td></tr>
</table>
Error
<table>
<tr><th>Value</th><th>Meaning</th></tr>
<tr><td>1</td><td>Incorrect number of elements
the maximum number is dependent on the table type 'enTabType'</td></tr>
<tr><td>2</td><td>Incorrect parameter set variant
dependent on the table type 'enTabKind'</td></tr>
<tr><td>3</td><td>'udMasterInc' value too high</td></tr>
<tr><td>4</td><td>'diOutInterv' value too high / too low</td></tr>
<tr><td>5</td><td>'diPar1' illegal value
dependent on 'enTabType' and 'enTabKind'</td></tr>
<tr><td>6</td><td>'diPar2' illegal value
dependent on 'enTabType' and 'enTabKind'</td></tr>
<tr><td>7</td><td>'diPar3' illegal value
dependent on 'enTabType' and 'enTabKind'</td></tr>
<tr><td>8</td><td>'diPar4' illegal value
dependent on 'enTabType' and 'enTabKind'</td></tr>
<tr><td>9</td><td>Illegal synchronous point</td></tr>
<tr><td>10</td><td>Illegal phasing in point</td></tr>
<tr><td>11</td><td>Illegal phasing out point</td></tr>
<tr><td>12</td><td>Illegal sine starting point</td></tr>
<tr><td>13</td><td>Velocity too low</td></tr>
<tr><td>14</td><td>Acceleration too low</td></tr>
</table></td></tr>
<tr><td>uiXOut</td><td>UINT</td><td>Phasing out point x coordinate
Phasing out ends with this position, transition from parabola to standstill
<table>
<tr><td>Range</td><td>0 ... 360</td></tr>
<tr><td>Unit</td><td>°</td></tr>
</table></td></tr>
</table>

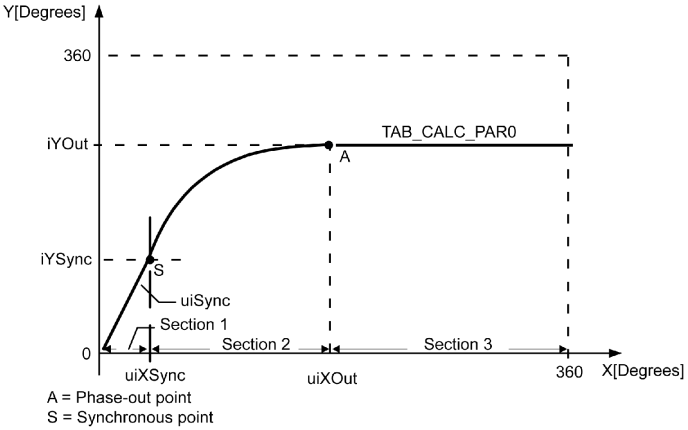
Input and output variables

Name	Type	Description
stDestTab	STRUCT	ST_PROF_TAB Profile table structure

Description

For the 'CALC_OUT_SYNCDEF' block, the phasing out curve is calculated with a parabola. The input parameters are the y coordinate of the phasing out point, the x coordinate of the synchronous point, and the synchronous factor. The table is calculated in the Y table format.

Abbildung 30: CALC_OUT_SYNCDEF: Phasing out function with defined synchronous points



Section 1	0 ≤ x < uiXSync	
Section 2	uiXSync ≤ x < uiXOut	Phasing out process: Parabola $y = -ax^2$
Section 3	uiXOut ≤ x < 360	

Calculation of the parabola

The factor a of the parabola $y = -ax^2$ is calculated from the equation $y' = uiSync/100 = -2ax$.
In the synchronous point S:

Therefore, it follows that for a:

where

The following dependencies and conditions apply:

-
-

2.1.10.4 Positioning profiles

Positioning tables are used in conjunction with the 'CAM_PROF_1' block for a positioning operation with a defined travel profile.

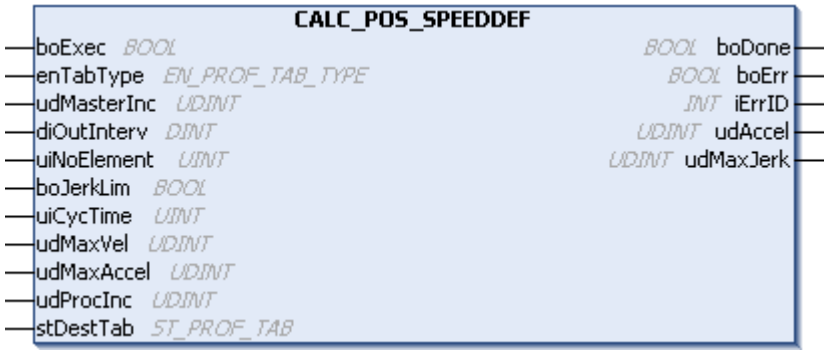
CALC_POS_SPEEDDEF	Calculation of the positioning table based on the maximum velocity
CALC_POS_TIMEDEF	Calculation of the positioning table based on percentage acceleration

2.1.10.4.1 CALC_POS_SPEEDDEF (FB)

The 'CALC_POS_SPEEDDEF' function block calculates the positioning table. Positioning increments, positioning time, maximum acceleration, and maximum velocity are predefined.

'CALC_POS_SPEEDDEF' is based on the 'TAB_CALC' function block with 'enParSet' := TAB_CALC_PAR1

User interface



Input variables

Name	Type	Description
boExec	BOOL	Function execution: With a positive edge, the execution of the block starts. As long as 'boExec' = TRUE, the block is processed by the PLC. In the state 'boExec' = FALSE execution of the block is ended.
enTabType	ENUM	EN_PROF_TAB_TYPE Table type, to differentiate between X and XY tables Default: PROF_YTAB Range / Meaning: PROF_YTAB: The y coordinate is saved as a table value, the x coordinate must be equidistant. PROF_YTAB_NL: ditto, the number of points is not limited PROF_XYTAB: x and y coordinates are saved as table values PROF_XYTAB_NL: ditto, the number of points is not limited
udMasterInc	UDINT	Increments of the master drive which produce a table cycle Max. table X value Range: 0 ... 5000000 Unit: incr
diOutInterv	DINT	Output interface defining the output increments per table cycle Max. table Y value Unit: incr
uiNoElement	UINT	Element number of the last table element calculated, number of table interpolation points Range: 'enTabType' = PROF_YTAB; PROF_YTAB_NL: 5 ... 'enTabType' = PROF_XYTAB; PROF_XYTAB_NL: 5 ...
boJerkLim	BOOL	Jerk limitation FALSE: No jerk limitation; constant acceleration TRUE: Constant jerk; linear increase in acceleration
uiCycTime	UINT	Cycle time Positioning takes place in this period of time, the master increments are input Unit: ms
udMaxVel	UDINT	Maximum velocity $\|v_{max}\|$ In the constant velocity range. Range: 1 … 65536000 Unit: 0.0001 rpm
udMaxAccel	UDINT	Maximum acceleration $\|a_{max}\|$ In the constant acceleration range Range: 1 … 65536000 Unit: 0.001 rev/s^2
udProcInc	UDINT	Process increments Encoder resolution on process side y Unit: incr/rev

Output variables

Name	Type	Description
boDone	BOOL	Response that the function block has been completely executed.
boErr	BOOL	The function block is in an error state FALSE: No error (permitted commanding or warning) TRUE: Error
iErrID	INT	Error identity number: Diagnostic number is output iErrID = 0: No error iErrID ≠ 0, boErr = TRUE: Error iErrID ≠ 0, boErr = FALSE: Warning Error 1: Incorrect number of elements the maximum number is dependent on the table type 'enTabType' 2: Incorrect parameter set variant dependent on the table type 'enTabKind' 3: 'udMasterInc' value too high 4: 'diOutInterv' value too high / too low 5: 'diPar1' illegal value dependent on 'enTabType' and 'enTabKind' 6: 'diPar2' illegal value dependent on 'enTabType' and 'enTabKind' 7: 'diPar3' illegal value dependent on 'enTabType' and 'enTabKind' 8: 'diPar4' illegal value dependent on 'enTabType' and 'enTabKind' 9: Illegal synchronous point 10: Illegal phasing in point 11: Illegal phasing out point 12: Illegal sine starting point 13: Velocity too low 14: Acceleration too low
udAccel	UDINT	Acceleration phase Acceleration as a proportion of the positioning cycle time Unit: %
udMaxJerk	UDINT	Maximum jerk during positioning Unit: rev/s^3

Input and output variables

Name	Type	Description
stDestTab	STRUCT	ST_PROF_TAB Profile table structure

Description

The 'CALC_POS_SPEEDDEF' function block supports positioning with a defined travel profile. The 'boJerkLim' can be set to select between positioning with jerk limitation and positioning without jerk limitation. A distinction can be made in the calculation between X and XY tables.

Abbildung 31: CALC_POS_SPEEDDEF: Positioning with jerk limitation

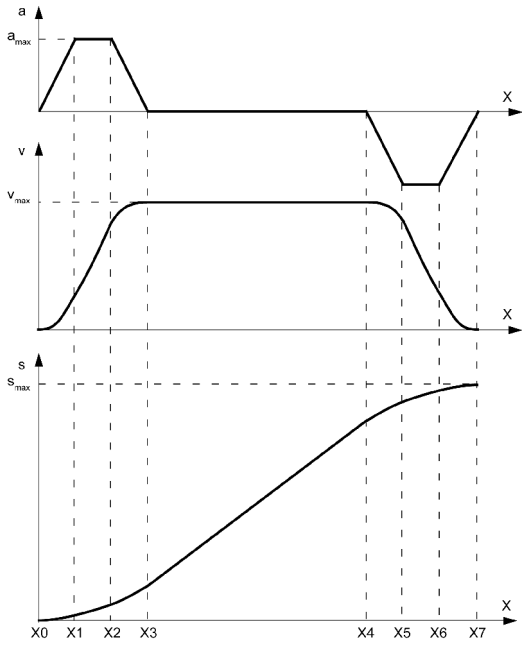
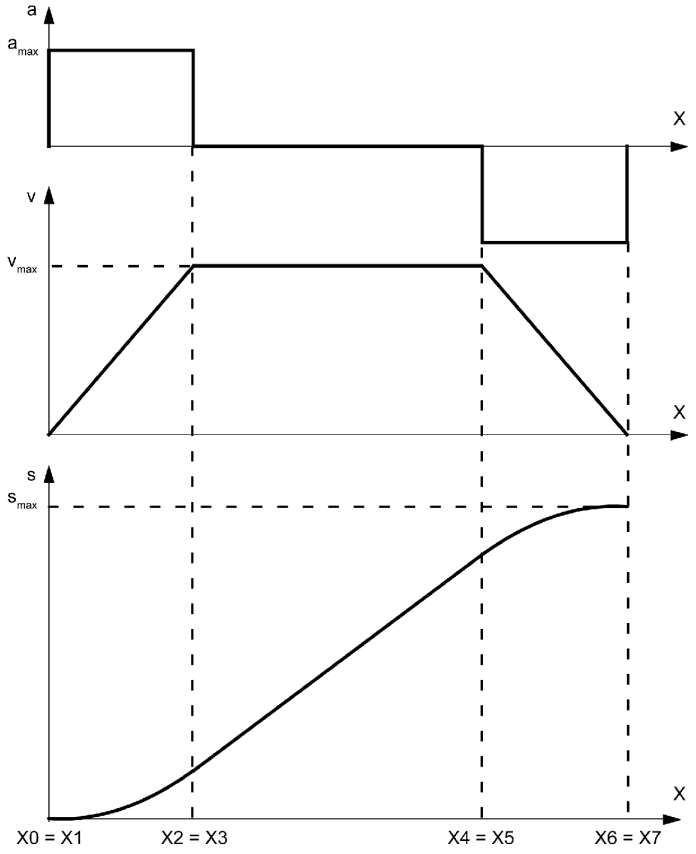


Abbildung 32: CALC_POS_SPEEDDEF: Positioning without jerk limitation



The following parameters must be specified prior to the table calculation:

- Encoder resolution of the axis to be traversed (udProcInd)
- Output interval s_{max} (diOutInterv)
- Cycle time T (uiCycTime)
- Maximum velocity v_{max} (udMaxVel)

So that the positioning operation can be completed in the specified cycle time, the following conditions apply for 'udMaxVel':

- Lower limit

  where udAccel = min = 1%
- Upper limit

  where udAccel = max = 50%

If 'udMaxVel' is lower than the permissible lower limit, the function block is terminated with an error message.
If the value is higher than the upper limit, 'udMaxVel' is set to the upper limit. In this case neither an error message nor a warning is output.

So that the acceleration can be completed in the specified cycle time, the following conditions apply for 'udMaxAccel':

- Lower limit
- Upper limit

Where the following applies for the acceleration time 'udAccel':

2.1.10.4.2 CALC_POS_TIMEDEF (FB)

The 'CALC_POS_TIMEDEF' function block calculates the positioning table based on the positioning increments, the positioning time, the max. acceleration, and the acceleration operation as a percentage of the positioning time.

User interface

Input variables

Name	Type	Description
boExec	BOOL	Function execution: With a positive edge, the execution of the block starts. As long as 'boExec' = TRUE, the block is processed by the PLC. In the state 'boExec' = FALSE execution of the block is ended.
enTabType	ENUM	EN_PROF_TAB_TYPE Table type, to differentiate between X and XY tables Default: PROF_YTAB Range / Meaning: PROF_YTAB: The y coordinate is saved as a table value, the x coordinate must be equidistant. PROF_YTAB_NL: ditto, the number of points is not limited PROF_XYTAB: x and y coordinates are saved as table values PROF_XYTAB_NL: ditto, the number of points is not limited
udMasterInc	UDINT	Increments of the master drive which produce a table cycle Max. table X value Range: 0 ... 5000000 Unit: incr

Name	Type	Description
diOutInterv	DINT	Output interface defining the output increments per table cycle Max. table Y value Unit: incr
uiNoElement	UINT	Element number of the last table element calculated, number of table interpolation points Range: 'enTabType' = PROF_YTAB; PROF_YTAB_NL: 5 ... 'enTabType' = PROF_XYTAB; PROF_XYTAB_NL: 5 ...
boJerkLim	BOOL	Jerk limitation FALSE: No jerk limitation; constant acceleration TRUE: Constant jerk; linear increase in acceleration
uiCycTime	UINT	Cycle time Positioning takes place in this period of time, the master increments are input Unit: ms
uiAccel	UINT	Acceleration phase Acceleration as a proportion of the positioning cycle time Unit: % Range: 1 … 50
udMaxAccel	UDINT	Maximum acceleration $\lvert a_{max} \rvert$ In the constant acceleration range Range: 1 … 65536000 Unit: 0.001 rev/s^2
udProcInc	UDINT	Process increments Encoder resolution on process side y Unit: incr/rev

Output variables

Name	Type	Description
boDone	BOOL	Response that the function block has been completely executed.
boErr	BOOL	The function block is in an error state FALSE: No error (permitted commanding or warning) TRUE: Error

<table>
<tr><th>Name</th><th>Type</th><th>Description</th></tr>
<tr><td>iErrID</td><td>INT</td><td>Error identity number: Diagnostic number is output
<table>
<tr><td colspan="2">iErrID = 0</td><td>No error</td></tr>
<tr><td>iErrID ≠ 0</td><td>boErr = TRUE</td><td>Error</td></tr>
<tr><td>iErrID ≠ 0</td><td>boErr = FALSE</td><td>Warning</td></tr>
</table>
Error
<table>
<tr><th>Value</th><th>Meaning</th></tr>
<tr><td>1</td><td>Incorrect number of elements
the maximum number is dependent on the table type 'enTabType'</td></tr>
<tr><td>2</td><td>Incorrect parameter set variant
dependent on the table type 'enTabKind'</td></tr>
<tr><td>3</td><td>'udMasterInc' value too high</td></tr>
<tr><td>4</td><td>'diOutInterv' value too high / too low</td></tr>
<tr><td>5</td><td>'diPar1' illegal value
dependent on 'enTabType' and 'enTabKind'</td></tr>
<tr><td>6</td><td>'diPar2' illegal value
dependent on 'enTabType' and 'enTabKind'</td></tr>
<tr><td>7</td><td>'diPar3' illegal value
dependent on 'enTabType' and 'enTabKind'</td></tr>
<tr><td>8</td><td>'diPar4' illegal value
dependent on 'enTabType' and 'enTabKind'</td></tr>
<tr><td>9</td><td>Illegal synchronous point</td></tr>
<tr><td>10</td><td>Illegal phasing in point</td></tr>
<tr><td>11</td><td>Illegal phasing out point</td></tr>
<tr><td>12</td><td>Illegal sine starting point</td></tr>
<tr><td>13</td><td>Velocity too low</td></tr>
<tr><td>14</td><td>Acceleration too low</td></tr>
</table></td></tr>
<tr><td>udMaxVel</td><td>UDINT</td><td>Maximum velocity $|v_{max}|$
In the constant velocity range.
<table>
<tr><td>Unit</td><td>0.0001 rpm</td></tr>
</table></td></tr>
<tr><td>udMaxJerk</td><td>UDINT</td><td>Maximum jerk during positioning
<table>
<tr><td>Unit</td><td>rev/s^3</td></tr>
</table></td></tr>
</table>

Input and output variables

Name	Type	Description
stDestTab	STRUCT	ST_PROF_TAB Profile table structure

Description

The 'CALC_POS_TIMEDEF' function block behaves in a similar way to the 'CALC_POS_SPEEDDEF' block.

The following parameters must be specified prior to the table calculation:

- Encoder resolution of the axis to be traversed (udProcInd)
- Output interval s_{max} (diOutInterv)
- Cycle time T (uiCycTime)
- Maximum acceleration (udMaxAccel)
- Acceleration phase (udAccel)

So that the acceleration can be completed in the specified cycle time, the following conditions apply for 'udMaxAccel':

- Lower limit
- Upper limit

Where the following applies for the acceleration time 'udMaxVel':

If 'udMaxAccel' is lower than the permissible lower limit, the function block is terminated with an error message.
If the value is higher than the upper limit, 'udMaxAccel' is set to the upper limit. In this case an error message is not output.

2.1.10.5 Support (support blocks)

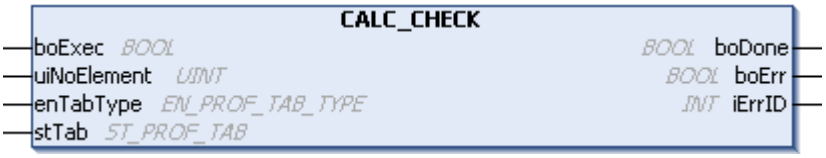
Support blocks are used by the higher-level blocks in the library. They cannot be used directly by the user.

CALC_CHECK	Support block

2.1.10.5.1 CALC_CHECK (FB)

The 'CALC_CHECK' function block monitors the maximum permissible value for 'uiNoElement' based on the table type 'enTabType' and the table structure 'stDestTab'.

User interface



Input variables

Name	Type	Description
boExec	BOOL	Function execution: With a positive edge, the execution of the block starts. As long as 'boExec' = TRUE, the block is processed by the PLC. In the state 'boExec' = FALSE execution of the block is ended.
uiNoElement	UINT	Element number of the last table element calculated, number of table interpolation points Range: 'enTabType' = PROF_YTAB; PROF_YTAB_NL: 5 ... 'enTabType' = PROF_XYTAB; PROF_XYTAB_NL: 5 ...
enTabType	ENUM	EN_PROF_TAB_TYPE Table type, to differentiate between X and XY tables Default: PROF_YTAB PROF_YTAB: Y table PROF_YTAB_NL: Unlimited Y table PROF_XYTAB: XY table PROF_XYTAB_NL: Unlimited XY table

Output variables

Name	Type	Description
boDone	BOOL	Response that the function block has been completely executed.

<table>
<tr><th>Name</th><th>Type</th><th>Description</th></tr>
<tr><td>boErr</td><td>BOOL</td><td>The function block is in an error state
<table>
<tr><td>FALSE</td><td>No error (permitted commanding or warning)</td></tr>
<tr><td>TRUE</td><td>Error</td></tr>
</table></td></tr>
<tr><td>iErrID</td><td>INT</td><td>Error identity number: Diagnostic number is output
<table>
<tr><td colspan="2">iErrID = 0</td><td>No error</td></tr>
<tr><td>iErrID ≠ 0</td><td>boErr = TRUE</td><td>Error</td></tr>
<tr><td>iErrID ≠ 0</td><td>boErr = FALSE</td><td>Warning</td></tr>
</table>
Error
<table>
<tr><th>Range</th><th>Meaning</th></tr>
<tr><td>1</td><td>Illegal value of uiNoElement</td></tr>
<tr><td>2</td><td>Illegal value of enTabType</td></tr>
</table></td></tr>
</table>

Input and output variables

Name	Type	Description
stTab	STRUCT	ST_PROF_TAB Profile table structure

Description

The function of the block is illustrated by the following IEC program:

```
FUNCTION_BLOCK CALC_CHECK
    VAR_INPUT
        boExec: BOOL;
        uiNoElement:UINT;
        enTabType:EN_PROF_TAB_TYPE;
    END_VAR
    VAR_OUTPUT
        boDone: BOOL;
        boErr: BOOL;
        iErrID: INT;
    END_VAR
    VAR_IN_OUT
        stTab: ST_PROF_TAB;
    END_VAR
    VAR
    END_VAR
```

```
IF boExec THEN
    CASE enTabType OF
        PROF_YTAB,PROF_YTAB_NL;
            IF uiNoElement>(SIZEOF(stTab)-8)/4-1 THEN
                boErr:=TRUE;
                iErrID:=1;
            ELSE
                boDone:=TRUE;
            END_IF
        PROF_XYTAB,PROF_XYTAB_NL:
            IF uiNoElement>(SIZEOF(stTab)-8)/8-1 THEN
                boErr:=TRUE;
                iErrID:=1;
            ELSE
                boDone:=TRUE;
            END_IF
        ELSE
            boErr:=TRUE;
            iErrID:=2; (* illegal enTabType; not supported by TAB_CALC *)
    END_CASE
ELSE
            boErr:=FALSE;
            boDone:=FALSE
            iErrID:=0;
END_IF
```

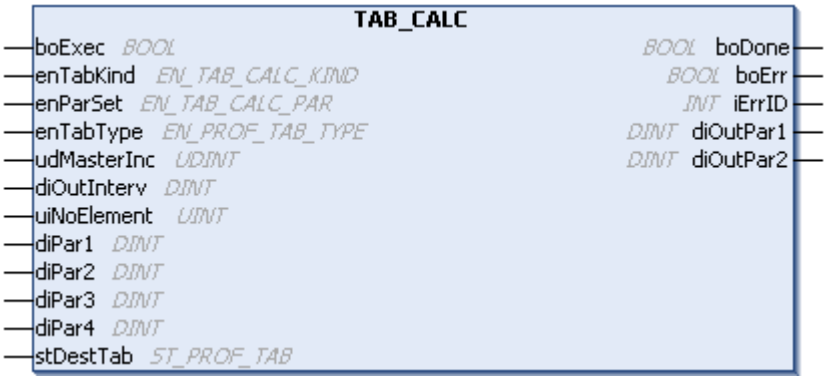
2.1.10.6 TAB_CALC (FB)

The 'TAB_CALC' function block is used to calculate table-based movement profiles.

'TAB_CALC' is a base block that is called by other function blocks in the AMK libraries. (see TabCalc documentation)

The calculation of the table interpolation points commences as soon as 'boExec' sees a transition from FALSE -> TRUE. For reasons associated with processing time, the calculation process is distributed across a number of PLC cycles.
The calculation ends with 'boDone' or, in the event of an error, with 'boErr'. After this, 'boExec' should be set to FALSE.

User interface



Input variables

Name	Type	Description
boExec	BOOL	Function execution: With a positive edge, the execution of the block starts. As long as 'boExec' = TRUE, the block is processed by the PLC. In the state 'boExec' = FALSE execution of the block is ended.

<table>
<tr><th>Name</th><th>Type</th><th>Description</th></tr>
<tr><td>enTabKind</td><td>ENUM</td><td>EN_TAB_CALC_KIND
Table type, for the specification of the fundamental table trend
<table>
<tr><td>Default</td><td>TAB_CALC_OP</td></tr>
</table>
<table>
<tr><th>Range</th><th>Meaning</th></tr>
<tr><td>TAB_CALC_IN</td><td>Phasing in table</td></tr>
<tr><td>TAB_CALC_OP</td><td>Operating table</td></tr>
<tr><td>TAB_CALC_OUT</td><td>Phasing out table</td></tr>
<tr><td>TAB_CALC_ POSJLI</td><td>Positioning profile with jerk limitation</td></tr>
<tr><td>TAB_CALC_POS</td><td>Positioning profile without jerk limitation</td></tr>
</table></td></tr>
<tr><td>enParSet</td><td>ENUM</td><td>EN_TAB_CALC_PAR
Parameter set variant, for the selection of the description parameters
<table>
<tr><td>Default</td><td>TAB_CALC_PAR0</td></tr>
</table>
<table>
<tr><th>Range</th><th>Meaning</th></tr>
<tr><td>TAB_CALC_ PAR0</td><td>First parameter set</td></tr>
<tr><td>TAB_CALC_ PAR1</td><td>Second parameter set</td></tr>
<tr><td>TAB_CALC_ PAR2</td><td>Third parameter set</td></tr>
</table></td></tr>
<tr><td>enTabType</td><td>ENUM</td><td>EN_PROF_TAB_TYPE
Table type, to differentiate between X and XY tables
<table>
<tr><td>Default</td><td>PROF_YTAB</td></tr>
</table>
<table>
<tr><th>Range</th><th>Meaning</th></tr>
<tr><td>PROF_YTAB</td><td>Equidistant X positions, Y positions defined by table value</td></tr>
<tr><td>PROF_XYTAB</td><td>X and Y position defined by table values</td></tr>
<tr><td>PROF_YTAB_NL</td><td>Equidistant X positions, Y positions defined by table value, not limited</td></tr>
<tr><td>PROF_XYTAB_ NL</td><td>X and Y position defined by table values, not limited</td></tr>
</table></td></tr>
<tr><td>udMasterInc</td><td>UDINT</td><td>Increments of the master drive which produce a table cycle
Max. table X value
<table>
<tr><td>Range</td><td>0 ... 5000000</td></tr>
<tr><td>Unit</td><td>incr</td></tr>
<tr><td>Default</td><td>20000</td></tr>
</table></td></tr>
<tr><td>diOutInterv</td><td>DINT</td><td>Output interface defining the output increments per table cycle
Max. table Y value
<table>
<tr><td>Unit</td><td>incr</td></tr>
<tr><td>Default</td><td>20000</td></tr>
</table></td></tr>
<tr><td>uiNoElement</td><td>UINT</td><td>Element number of the last table element calculated,
number of table interpolation points
<table>
<tr><td>Default</td><td>100</td></tr>
</table></td></tr>
<tr><td>diPar1</td><td>DINT</td><td rowspan="4">Parameter n, based on the selected table type and parameter set variant
<table>
<tr><td>Default</td><td>360</td></tr>
</table></td></tr>
<tr><td>diPar2</td><td>DINT</td></tr>
<tr><td>diPar3</td><td>DINT</td></tr>
<tr><td>diPar4</td><td>DINT</td></tr>
</table>

Output variables

<table>
<tr><th>Name</th><th>Type</th><th>Description</th></tr>
<tr><td>boDone</td><td>BOOL</td><td>Response that the function block has been completely executed.</td></tr>
<tr><td>boErr</td><td>BOOL</td><td>The function block is in an error state
<table>
<tr><td>FALSE</td><td>No error (permitted commanding or warning)</td></tr>
<tr><td>TRUE</td><td>Error</td></tr>
</table></td></tr>
<tr><td>iErrID</td><td>INT</td><td>Error identity number: Diagnostic number is output
<table>
<tr><td colspan="2">iErrID = 0</td><td>No error</td></tr>
<tr><td>iErrID ≠ 0</td><td>boErr = TRUE</td><td>Error</td></tr>
<tr><td>iErrID ≠ 0</td><td>boErr = FALSE</td><td>Warning</td></tr>
</table>
Warning
<table>
<tr><th>Value</th><th>Meaning</th></tr>
<tr><td>1</td><td>Difference from adjacent point too great (>32767)</td></tr>
</table>
Error
<table>
<tr><th>Value</th><th>Meaning</th></tr>
<tr><td>1</td><td>Incorrect number of elements
the maximum number is dependent on the table type 'enTabType'</td></tr>
<tr><td>2</td><td>Incorrect parameter set variant
dependent on the table type 'enTabKind'</td></tr>
<tr><td>3</td><td>'udMasterInc' value too high</td></tr>
<tr><td>4</td><td>'diOutInterv' value too high / too low</td></tr>
<tr><td>5</td><td>'diPar1' illegal value
dependent on 'enTabType' and 'enTabKind'</td></tr>
<tr><td>6</td><td>'diPar2' illegal value
dependent on 'enTabType' and 'enTabKind'</td></tr>
<tr><td>7</td><td>'diPar3' illegal value
dependent on 'enTabType' and 'enTabKind'</td></tr>
<tr><td>8</td><td>'diPar4' illegal value
dependent on 'enTabType' and 'enTabKind'</td></tr>
<tr><td>9</td><td>Illegal synchronous point</td></tr>
<tr><td>10</td><td>Illegal phasing in point</td></tr>
<tr><td>11</td><td>Illegal phasing out point</td></tr>
<tr><td>12</td><td>Illegal sine starting point</td></tr>
<tr><td>13</td><td>Velocity too low</td></tr>
<tr><td>14</td><td>Acceleration too low</td></tr>
</table></td></tr>
<tr><td>diOutPar1</td><td>DINT</td><td rowspan="2">Output parameter based on the selected table type and parameter set variant</td></tr>
<tr><td>diOutPar2</td><td>DINT</td></tr>
</table>

Input and output variables

Name	Type	Description
stDestTab	STRUCT	ST_PROF_TAB Profile table structure

2.2 Types

2.2.1 Data types

ST_FIFO_HEADER	The FIFO header information is used by the 'FIFO_HANDLER' function block to organize a FIFO

ST_POS_ELE	Specification of a positioning element comprising position and velocity value

2.2.1.1 Basic

2.2.1.1.1 General

2.2.1.1.1.1 ST_XY_DI (ST)

Elements

Name	Type	Description
diXPos	DINT	X-position
diYPos	DINT	Y-position

Structure definition

```
TYPE ST_XY_DI :
    STRUCT
        diXPos:DINT;
        diYPos:DINT;
    END_STRUCT
END_TYPE
```

2.2.1.2 Convert

2.2.1.2.1 Polynominal

2.2.1.2.1.1 EN_XYVA_CONVERT (EN)

Reserved for AMK internal use!

2.2.1.2.1.2 EN_XYVA_CONV_ERR (EN)

Reserved for AMK internal use!

2.2.1.2.1.3 ST_POLY_PARA_1 (ST)

Reserved for AMK internal use!

2.2.1.3 FifoHandling

2.2.1.3.1 EN_FIFO_HANDLER_MODE (EN)

Selection mode of the function
The function is executed in the context of a [Fifoxxxxx] action

Elements

Name	Type	Description
FIFO_INIT	INT	Initialize FIFO The relevant input variables are added to the 'stFifoHeader' variable and the memory made available for the FIFO is cleared (FifoClear).
FIFO_CLEAR	INT	Clear FIFO 'uiEleNmb' and 'udEleInd' are cleared (:= 0). The internal state 'stFifoHeader.enFifoState' = FIFO_STATE_READY is also set.

Name	Type	Description
FIFO_RESET	INT	Reset output index The action sets 'udEleInd' := 0 and 'uiEleNmb' to the number of elements already written to the FIFO. This enables the elements already written to be read again. The action can only be used if there has not yet been a FIFO overrun: 'stFifoHeader.enFifoState' = FIFO_STATE_READY
FIFO_READ	INT	Read out FIFO element Read the first FIFO element written and not yet read out. This current element is then written to the address referenced in 'pbyEle' with 'uiEleSize' bytes. 'uiEleNmb' is decremented, 'udEleInd' is incremented.
FIFO_WRITE	INT	Write FIFO element The element referenced by 'pbyEle' is written. 'uiEleSize' bytes are written. 'uiEleNmb' is incremented.

2.2.1.3.2 EN_FIFO_STATE (EN)

FIFO State

Elements

Name	Type	Description
FIFO_STATE_INIT	INT	FIFO not yet initialized
FIFO_STATE_READY	INT	FIFO ready for use
FIFO_STATE_READY_ TURNOVER	INT	FIFO overrun has occurred

2.2.1.3.3 ST_FIFO_HEADER (ST)

The FIFO header information is used by the 'FIFO_HANDLER' function block to organize a FIFO.

The FIFO header information must be created in the application. However, its content is used exclusively in the context of the internal organization of the FIFO; it does not have to be evaluated from the point of view of the application.

Structure elements

Name	Type	Description
uiInIndex	UINT	Write index: is incremented with FifoWrite().
uiOutIndex	UINT	Read index: is incremented with FifoRead().
uiMaxIndex	UINT	Maximum permissible index.

Name	Type	Description
enFifoState	ENUM	EN_FIFO_STATE FIFO state Default: FIFO_STATE_INIT Range / Meaning: FIFO_STATE_INIT: FIFO not yet initialized FIFO_STATE_READY: FIFO ready for use FIFO_STATE_READY_TURNOVER: FIFO overrun has occurred
uiEleSize	UINT	Size (in bytes) of the element to be written / read For FifoInit(), the value is taken from the corresponding input variable. After this, the input variable must not change again.
pbyFifo	POINTER	POINTER TO BYTE Pointer to the address starting from which memory capacity is made available for the FIFO organization

Structure definition

```
TYPE ST_FIFO_HEADER:
	STRUCT
		uiInIndex: UINT;
		uiOutIndex: UINT;
		uiMaxIndex: UINT;
		enFifoState: EN_FIFO_STATE;
		uiEleSize: UINT;
		pbyFifo: POINTER TO BYTE;
	END_STRUCT
END_TYPE
```

2.2.1.4 HigherLevel

2.2.1.4.1 Visu

2.2.1.4.1.1 ST_VISU_TAB (ST)

Reserved for AMK internal use!

2.2.1.5 POS_J

2.2.1.5.1 EN_POS_J_MODE (EN)

Selection mode of the selected movement sequence

Elements

Name	Type	Description
POS_J_REL	INT	Relative positioning
POS_J_REL_RETRIG	INT	Relative positioning, retriggerable of a new position
POS_J_REL_RETRIG_EXT	INT	Relative positioning with extended fuctionality , retriggerable

2.2.1.5.2 EN_SlopeErrorCode (EN)

Elements

Name	Type	Description
Slope1_Ok	INT	No error
Slope1_InvalidArguments	INT	Invalid Arguments: • diPosition ≤0 • lreAccelStart ≤0 • lreAccelEnd ≤0 • lreJerkStart <0 • lreJerkEnd <0
Slope1_VN_Adjusted	INT	The specified velocity is not reached (Example: Too short distance)
Slope1_VE_Adjusted	INT	The specified end velocity is not reached (Example: Too short distance)
Slope1_RInf	INT	Jerk is set to maximum value, exception profile (without jerk limitation) is moved
Slope1_Amax_Adjusted	INT	The default start acceleration is not reached. The setpoint was exceeded / dropped below
Slope1_Amax2_Adjusted	INT	The default deceleration is not reached. The setpoint was exceeded / dropped below
Slope1_AE_Adjusted	INT	The default end value of the acceleration is not reached. The setpoint was exceeded / dropped below
Slope1_AS_Adjusted	INT	The default start value of the acceleration is not reached. The setpoint was exceeded / dropped below
Slope1_Amax2_Neg	INT	An error occurred while adjusting the end acceleration, the value becomes negative. It is aborted
Slope1_Amax2_Diff_Error	INT	An error occurred while adapting the acceleration
Slope1_Amax2_Adjust_Error	INT	An error occurred while adapting the acceleration
Slope1_A_Adjusted	INT	Reserved for AMK internal use!
Slope1_CalcError	INT	Calculation error of the motion profile

2.2.1.5.3 ST_SlopeCalcData (ST)

Reserved for AMK internal use!

2.2.1.5.4 ST_SlopeDataIn (ST)

Reserved for AMK internal use!

2.2.1.5.5 ST_SlopeDataOut (ST)

Reserved for AMK internal use!

2.2.1.6 Sequencial Positioning

2.2.1.6.1 ST_POS_ELE (ST)

Specification of a positioning element comprising position and velocity value based on the 'ST_POS_ELE' structure.

Structure elements

Name	Type	Description
diPosition	DINT	Setpoint position Definition of the final position Overall increment increase of the output value Unit: incr **Default**: 600000
udVelocity	UDINT	Setpoint velocity Definition of the final velocity Increment difference of the output value by time Range: 0 ... 300000000 Unit: incr/s **Default**: 200000

Structure definition

```
TYPE ST_POS_ELE:
    STRUCT
        diPosition: DINT;
        udVelocity: UDINT;
    END_STRUCT
END_TYPE
```

2.2.2 Enumerations

2.2.2.1 Basic

2.2.2.1.1 EN_AFP_CODE (EN)

Select: Command code AFP

Elements

Name	Type	Description
AFP_ZERO	INT	The command influences control bits without triggering an active function
AFP_BREAK	INT	Breaks off a running AFP command (identified by Q_ CODE=1) and restores the basic status for new authorization of the system.
AFP_LOGIN	INT	With the LOGIN command the field bus acquires the unrestricted use of the system through the onedimensional commanding interface.
AFP_LOGOUT	INT	LOGOUT releases the commanding interface for all users again, BUS transfers the system control
AFP_CLRERR	INT	Delete errors
AFP_RDERR	INT	Read error number
AFP_RDDAT	INT	Read parameters from the system data management
AFP_WRDAT	INT	Write parameters into the system data management
AFP_RDELE	INT	Read list parameters from the data management
AFP_WRELE	INT	Write list parameters into the data management
AFP_WRTMP	INT	Tmp. (online) data change, possible for parameters

Name	Type	Description
AFP_ACT	INT	Allocation of feedback magnitudes (e.g. actual values)
AFP_RDSINCOS	INT	Read absolute encoder value
AFP_SETVAL_I	INT	The standard setpoint source 16bit
AFP_SETVAL_DI	INT	Standard setpoint source 32bit
AFP_BE	INT	UE (DC BUS enable) for units with main contactor / KE
AFP_XIADD	INT	Shift position feedback coordinates
AFP_RDVER	INT	Read operating system software version
AFP_RDIO	INT	Reading binary input / output ports
AFP_WRIO	INT	Writing binary input / output ports
AFP_RDTMP	INT	Read temporary parameter value
AFP_RDXSET	INT	Reading current position setpoint
AFP_SYSINIT	INT	System booting
AFP_MODE0	INT	Operating mode change to main op. mode
AFP_MODE1	INT	Operating mode change according to secondary operating mode 1
AFP_MODE2	INT	Operating mode change according to secondary operating mode 2
AFP_MODE3	INT	Operating mode change according to secondary operating mode 3
AFP_MODE4	INT	Operating mode change according to secondary operating mode 4
AFP_MODE5	INT	Operating mode change according to secondary operating mode 5
AFP_PARA	INT	Parameter set change
AFP_TORQUE	INT	Torque control
AFP_SPEED	INT	Speed control
AFP_HOME	INT	Homing cycle
AFP_SPOS	INT	Spindle positioning
AFP_POSA	INT	Absolute positioning
AFP_POSR	INT	Relative positioning
AFP_SYNC4	INT	Synchronous control in secondary operating mode 4
AFP_SYNC5	INT	Synchronous control in secondary operating mode 5
AFP_STOP	INT	Drive STOP Transition to speed control with digital setpoint “0”
AFP_MSTART	INT	Start of the measuring function
AFP_MSTOP	INT	Stop of the measuring function

2.2.2.1.2 EN_AFP_MODE (EN)

Selection mode AFP

Elements

Name	Type	Description
AFP_M_DI_I	INT	diSetVal and iSetVal accordino to block presetting
AFP_M_DI	INT	diSetVal accordino to block presetting (iSetVal accordino to ID presetting)
AFP_M_I	INT	iSetVal accordino to block presetting (diSetVal according to ID presetting)

Name	Type	Description
AFP_M_NONE	INT	diSetVal and iSetVal according to ID presetting

2.2.2.1.3 EN_CAM_CONT_MODE (EN)

Selection mode between incremental and absolute input evaluation

Elements

Name	Type	Description
CAM_CONT_INC	INT	Incremental input value evaluation
CAM_CONT_ABS	INT	Absolute input value evaluation

2.2.2.1.4 EN_CAM_PROF_MODE (EN)

Selection mode of the required movement sequence

Elements

Name	Type	Description
CAM_PROF_CONT	INT	Continuous movement without taking 'boControl' into account
CAM_PROF_PIN_POUT	INT	Input / output mode with control of table transition by 'boControl'
CAM_PROF_START_STOP	INT	Start / stop mode with control of table transition by 'boControl' and automatic stop at end of table
CAM_PROF_IM_START_STOP	INT	Immediate start / stop mode with control of table transition by 'boControl' and automatic stop at end of table
CAM_PROF_CONT_R	INT	Continuous movement without taking 'boControl' into account the movement sequence starts relative to the current position X_{act}/Y_{act}
CAM_PROF_PIN_POUT_R	INT	Input / output mode with control of table transition by 'boControl' the movement sequence starts relative to the current position X_{act}/Y_{act}
CAM_PROF_START_STOP_R	INT	Start / stop mode with control of table transition by 'boControl' and automatic stop at end of table the movement sequence starts relative to the current position X_{act}/Y_{act}
CAM_PROF_IM_START_STOP_R	INT	Immediate start / stop mode with control of table transition by 'boControl' and automatic stop at end of table the movement sequence starts relative to the current position X_{act}/Y_{act}

2.2.2.1.5 EN_CORRECT_MODE (EN)

Selection mode of operating mode; 'enMode' can be changed online

Elements

Name	Type	Description
CORRECT_SET2OUT	INT	Only the correction value is output
CORRECT_ADD2OUT	INT	The interpolation of the correction value is output added to the input value 'diInVal'
CORRECT_SET2OUT_NB	INT	Only the correction value is output Avoidance of reversal of direction
CORRECT_ADD2OUT_NB	INT	The interpolation of the correction value is output added to the input value 'diInVal' Avoidance of reversal of direction

2.2.2.1.6 EN_DETECT_MODE (EN)

Selection mode of the operating mode

Elements

Name	Type	Description
DETECT_AUTO	INT	Automatic homing with reference to the first printing mark
DETECT_MANUAL	INT	Manual homing The printing mark must be aligned manually with the sensor prior to enabling with 'boEnable'.

2.2.2.1.7 EN_POS_AJ_MODE (EN)

Selection mode of the selected movement sequence

Elements

Name	Type	Description
POS_AJ_REL	INT	Relative positioning
POS_AJ_REL_RETRIG	INT	Relative positioning, retriggerable

2.2.2.1.8 EN_POS_MODE (EN)

Selection mode of the selected movement sequence (operating mode)

Elements

Name	Type	Description
POS_REL	INT	Relative positioning
POS_REL_RETRIG	INT	Relative positioning, retriggerable
POS_MODULO	INT	Modulo positioning
POS_INTERPOSED	INT	Positioning with override function with retraction
POS_INTERPOSED_NB	INT	Positioning with override function without retraction
POS_REL_RETRIG_EXT	INT	Relative positioning, retriggerable, with extended functionality

2.2.2.1.9 EN_PROF_TAB_TYPE (EN)

Table type, to differentiate between X and XY tables

Elements

Name	Type	Description
PROF_YTAB	INT	Equidistant X positions, Y positions defined by table value
PROF_XYTAB	INT	X and Y positions defined by table values
PROF_YTAB_NL	INT	Equidistant X positions, Y positions defined by table value, not limited
PROF_XYTAB_NL	INT	X and Y positions defined by table values, not limited
PROF_XYVATAB	INT	Polynomial table: X and Y positions, velocity, acceleration defined by table values

2.2.2.1.10 EN_TAB_CALC_KIND (EN)

Table type, for the specification of the fundamental table trend

Elements

Name	Type	Description
TAB_CALC_IN	INT	Phasing in table
TAB_CALC_OP	INT	Operating table
TAB_CALC_OUT	INT	Phasing out table
TAB_CALC_POS_JLI	INT	Positioning profile with jerk limitation
TAB_CALC_POS	INT	Positioning profile without jerk limitation

2.2.2.1.11 EN_TAB_CALC_PAR (EN)

Parameter set variant, for the selection of the description parameters

Elements

Name	Type	Description
TAB_CALC_PAR0	INT	First parameter set
TAB_CALC_PAR1	INT	Second parameter set
TAB_CALC_PAR2	INT	Third parameter set

2.2.2.1.12 EN_TARGET_INFO (EN)

Selection of target system information

Elements

Name	Type	Description
TAR_VERSION	INT	Date of target system version
ADDR_SWITCH	INT	Not currently used
ACTIVE_BUS	INT	'Bus active' information
LIFE_CHECK	INT	Not currently used
MAGIC_LOW	INT	Not currently used
HW_VERSION	INT	Not currently used

Target system information

Value	Meaning
16#yyww	TAR_VERSION yy - year, ww = calendar week (e.g. 16#1345: calendar week 45 of year 2013)

Value	Meaning
2#xxxx.xxx0	ACTIVE BUS Bit 1 ... bit 7 are assigned to the corresponding instances 1 ... 7 according to ID2'SERCOS cycle time'. The corresponding bit is TRUE if the bus cycle is active.

Instance	Use
0	-
1	ACC bus master
2	Profibus slave (option A-SPB)
	EtherCAT slave (option A-SEC)
	CAN / ACC bus slave (option A-SCN)
	EtherNet/IP (option A-SIP)
	Profinet IO Device (option A-SPN)
3	I/O option
4	1st Ethernet X20
5	EtherCAT master (option A-MEC)
6	Reserved
7	2nd Ethernet X60

This information is of interest if the bus cycle time does not match the PGT cycle time

2.2.2.1.13 EN_VGEN_MODE (EN)

Selection mode of the operating mode

Elements

Name	Type	Description
VGEN_CONT	INT	Continuous increment increase, as long as 'boControl' = TRUE
VGEN_CYCLE	INT	Increment increase by the defined value 'udModPos' on a positive edge at 'boControl'

2.2.2.2 Cmd_Afp

2.2.2.2.1 EN_HOME_MODE (EN)

Homing cycle mode specific to AMK (operates according to ID32926 'AMK homing cycle parameter')

Elements

Name	Type	Description
CAM_OFF	INT	No cam evaluation
LIN_PULS_ZON	INT	Linear axis – pulse cam with zero pulse evaluation
LIN_PULS_ZOFF	INT	Linear axis – pulse cam without zero pulse evaluation
LIN_RANGE_ZON	INT	Linear axis – range cam with zero pulse evaluation
LIN_RANGE_ZOFF	INT	Linear axis – range cam without zero pulse evaluation
ROT_PULS_ZON	INT	Rotary axis – pulse cam with zero pulse evaluation
ROT_PULS_ZOFF	INT	Rotary axis – pulse cam without zero pulse evaluation
ROT_RANGE_ZON	INT	Rotary axis – range cam with zero pulse evaluation
ROT_RANGE_ZOFF	INT	Rotary axis – range cam without zero pulse evaluation

2.2.2.2.2 EN_SPOS_MODE (EN)

Mode, for the selection of the spindle positioning mode.

Elements

Name	Type	Description
UNKN_CAM_OFF	INT	if reference point unknown – no cam evaluation
UNKN_CPOS_ZON	INT	if reference point unknown – positive cam edge – with zero pulse evaluation
UNKN_CPOS_ZOFF	INT	if reference point unknown – positive cam edge – without zero pulse evaluation
UNKN_CNEG_ZON	INT	if reference point unknown – negative cam edge – with zero pulse evaluation
UNKN_CNEG_ZOFF	INT	if reference point unknown – negative cam edge – without zero pulse evaluation
ALW_CAM_OFF	INT	always – no cam evaluation
ALW_CPOS_ZON	INT	always – positive cam edge – with zero pulse evaluation
ALW_CPOS_ZOFF	INT	always – positive cam edge – without zero pulse evaluation
ALW_CNEG_ZON	INT	always – negative cam edge – with zero pulse evaluation
ALW_CNEG_ZOFF	INT	always – negative cam edge – without zero pulse evaluation

2.2.2.2.3 EN_SYNC_MODE (EN)

Synchronous mode

Elements

Name	Type	Description
SYNC_BY_ID	INT	synchronisation parameter per ID225 /ID32927
ANGLE_ALIGNM_OFF	INT	no angle alignment
UNKN_ABS	INT	if reference point unknown – absolute alignment
UNKN_REL	INT	if reference point unknown – relative alignment
ALW_ABS	INT	always – absolute alignment
ALW_REL	INT	always – relative alignment

2.2.2.2.4 EN_SYNC_MODE_AL (EN)

Align mode

Elements

Name	Type	Description
SYNC_ANY	INT	any alignment
DEF_OVER	INT	over-synchronous alignment
DEF_UND_ADMISS	INT	under-synchronous alignment – inversion of direction permissible
DEF_UND_INADM	INT	under-synchronous alignment – inversion of direction invalid

2.2.2.2.5 EN_SYNC_MODE_CAM (EN)

Cam mode

Elements

Name	Type	Description
SYNC_CAM_OFF	INT	no cam evaluation
CPOS_ZON	INT	positive cam edge – zero pulse evaluation
CPOS_ZOFF	INT	positive cam edge – no zero pulse evaluation
CNEG_ZON	INT	negative cam edge – zero pulse evaluation
CNEG_ZOFF	INT	negative cam edge – no zero pulse evaluation

2.2.2.3 MoreIds

2.2.2.3.1 EN_ACCESS_N_IDS (EN)

Selection mode

Definition of remanent or temporary parameters

Elements

Name	Type	Description
ACCESS_N_IDS_DINT_TMP	INT	Read / Write temporary parameter value
ACCESS_N_IDS_DINT_REM	INT	Read / Write remanent parameter value

2.2.2.3.2 EN_READ_ALL_IDS_STATE (EN)

Reserved for AMK internal use!

2.2.2.3.3 EN_READ_ID_LIST_ALL_STATE (EN)

Reserved for AMK internal use!

2.2.2.4 System

2.2.2.4.1 EN_AMK_LIB (EN)

Reserved for AMK internal use!

2.2.3 Structures

2.2.3.1 Basic

2.2.3.1.1 ST_LOCAL_TIME_INFO (ST)

Structure elements

Name	Type	Description
diLocToUtcDiff	DINT	Difference between local time and UTC (Coordinated Universal Time) Unit: ms
boIsDst	BOOL	Summertime flag

Structure definition

```
TYPE ST_LOCAL_TIME_INFO:
        STRUCT
                diLocToUtcDiff:DINT;
                boIsDst:BOOL;
        END_STRUCT
END_TYPE
```

2.2.3.2 CamContactor

2.2.3.2.1 ST_CONT (ST)

Structure elements

Name	Type	Description
diOn	DINT	Cam activation point 'diInVal' value at and above which 'boOutVal' = TRUE is set
diOff	DINT	Cam deactivation point 'diInVal' value at and above which 'boOutVal' = FALSE is set

Structure definition

```
TYPE ST_CONT:
    STRUCT
        diOn:DINT;
        diOff:DINT;
    END_STRUCT
END_TYPE
```

2.2.3.2.2 ST_CONT_TAB (ST)

The cam table is based on the 'ST_CONT_TAB structure', which permits the definition of up to 16 (MAX_CONT_TAB_IND) cam on and off points.

Structure elements

Name	Type	Description
uiActCams	UINT	The number of array elements (cam activation and deactivation points) currently taken into account in the 'stCam' substructure array Range: 1 ... uiMaxCams Default: 1
uiMaxCams	UINT	Maximum permissible number of array elements of the 'stCam' substructure array Default: 16 (constant)
diRes	DINT	(Not used in the context of the specific function for AMK)
stCam	ARRAY	ARRAY [1..MAX_CONT_TAB_IND] OF ST_CONT Array of cam activation and deactivation points

Structure definition

```
MAX_CONT_TAB_IND:UINT:=16;                    (* highest valid index for ST_CONT_TAB.stCam[1...] *)

TYPE ST_CONT_TAB:
    STRUCT
        uiActCams:UINT := 1;
        uiMaxCams:UINT := MAX_CONT_TAB_IND;
        diRes:DINT;
        stCAM:ARRAY[1..MAX_CONT_TAB_IND] OF ST_CONT;
    END_STRUCT
END_TYPE
```

2.2.3.3 CamProfile

2.2.3.3.1 SMC_CAMTappet (ST)

Reserved for AMK internal use!

2.2.3.3.2 SMC_CAMTAPPETACTION (ENUM)

Reserved for AMK internal use!

2.2.3.3.3 SMC_CAMTAPPETTYPE (ENUM)

Reserved for AMK internal use!

2.2.3.3.4 SMC_CAMXYVA (ST)

The 'SMC_CAMXYVA' structure contains the type definitions specific to 3S which are needed by the CODESYS cam disk editor in order to write polynomial tables.

Moreover, the XYVA table supported by 'CAM_PROF_1' is based on an 'ARRAY OF SMC_CAMXYVA' which is referenced by a pointer in the context of the 'ST_PROF_XYVATAB' structure.

Structure elements

Name	Type	Description
dX	LREAL	x position, master
dY	LREAL	y position, slave
dV	LREAL	; slave velocity for a constant master velocity of 1
dA	LREAL	; slave acceleration for a constant master velocity of 1

Structure definition

```
TYPE SMC_CAMXYVA:
    STRUCT
        dX:LREAL;
        dY:LREAL;
        dV:LREAL;
        dA:LREAL;
    END_STRUCT
END_TYPE
```

2.2.3.3.5 ST_POLY_PARA (ST)

Reserved for AMK internal use!

2.2.3.3.6 ST_POLY_PARA_INTERNAL (ST)

Reserved for AMK internal use!

2.2.3.3.7 ST_PROF_TAB (ST)

Up to 361 DINT type table elements can be defined with the 'ST_PROF_TAB' structure.

Modes 'enTabType' = PROF_YTAB_NL and 'enTabType' = PROF_XYTAB_NL lift this restriction on table elements.

Structure elements

Name	Type	Description
enType	ENUM	EN_PROF_TAB_TYPE Table type, to differentiate between X and XY tables Default: PROF_YTAB **PROF_YTAB**: Equidistant X positions, Y positions defined by table value **PROF_XYTAB**: X and Y positions defined by table values **PROF_YTAB_NL**: Equidistant X positions, Y positions defined by table value, not limited **PROF_XYTAB_NL**: X and Y positions defined by table values, not limited **PROFXYVATAB**: Polynomial table: X and Y positions, velocity, acceleration defined by table values
uiNoElement	UINT	Element number of the last table element calculated, number of table interpolation points
udMasterInc	UDINT	Increments of the master drive which produce a table cycle Max. table X value
diElement[0] .... diElement[360]	ARRAY	ARRAY [0..MAX_PROF_IND] OF DINT Table elements

Structure definition

```
MAX_PROF_Y_IND:UINT:=360;                    (* highest valid index for diElement[x] *)

TYPE ST_PROF_TAB:
    STRUCT
        enType:EN_PROF_TAB_TYPE ;
        uiNoElement:UINT;
        udMasterInc:UDINT;
        diElement:ARRAY[0…MAX_PROF_Y_IND] OF DINT;
    END_STRUCT
END_TYPE
```

2.2.3.3.8 ST_PROF_TAB_INFO (ST)

Reserved for AMK internal use!

2.2.3.3.9 ST_PROF_XY (ST)

Structure elements

Name	Type	Description
diX	DINT	X value of table interpolation point
diY	DINT	Y value of table interpolation point

Structure definition

```
TYPE ST_PROF_XY:
    STRUCT
        diX:DINT;
        diY:DINT;
    END_STRUCT
END_TYPE
```

2.2.3.3.10 ST_PROF_XYTAB (ST)

The 'ST_PROF_XYTAB' structure defines an XY table whose x axis can be split at will. The table structure contains the x and y values of the function y = f(x).

Structure elements

Name	Type	Description
enType	ENUM	EN_PROF_TAB_TYPE Table type, to differentiate between X and XY tables Default: PROF_XYTAB Range / Meaning: PROF_XYTAB: X and Y positions defined by table values PROF_XYTAB_NL: X and Y positions defined by table values, not limited
uiNoElement	UINT	Element number of the last table element calculated, number of table interpolation points Range: 1 ... 180 Default: 180
udMasterInc	UDINT	Increments of the master drive which produce a table cycle Max. table X value (not used for XY tables)

<table>
<tr><th>Name</th><th>Type</th><th>Description</th></tr>
<tr><td>stElement</td><td>ARRAY</td><td>ARRAY [0..MAX_PROF_XY_IND] OF ST_PROF_XY
Table elements, X and Y positions
<table>
<tr><th>Range</th><th>Meaning</th></tr>
<tr><td>stElement[0]</td><td>X / Y value at zero point of table</td></tr>
<tr><td>diX</td><td>X value at zero point of table, always 0</td></tr>
<tr><td>diY</td><td>Y value at zero point of table, always 0</td></tr>
<tr><td>stElement[1]</td><td>1st X / Y value of the table</td></tr>
<tr><td>diX</td><td>X value of the table</td></tr>
<tr><td>diY</td><td>Y value of the table</td></tr>
<tr><td>stElement[2]</td><td>2nd X / Y value of the table</td></tr>
<tr><td>diX</td><td>X value of the table</td></tr>
<tr><td>diY</td><td>Y value of the table</td></tr>
<tr><td>...</td><td>...</td></tr>
<tr><td>stElement[180]</td><td>180th X / Y value of the table</td></tr>
<tr><td>diX</td><td>X value of the table (where diX > stElement[179].diX)</td></tr>
<tr><td>diY</td><td>Y value of the table</td></tr>
</table></td></tr>
</table>

Structure definitions

Structure definition

```
MAX_PROF_XY_IND:UINT:=180;                (* highest valid index for 'stElement[0…]' *)

TYPE ST_PROF_YTAB:
    STRUCT
        enType:EN_PROF_TAB_TYPE:=PROF_XYTAB;
        uiNoElement:UINT:=MAX_PROF_Y_IND;
        udMasterInc:UDINT:=20000;
        stElement:ARRAY[0…MAX_PROF_XY_IND] OF ST_PROF_XY;
    END_STRUCT
END_TYPE
```

If the table type 'enType' = PROF_XYTAB_NL is selected, the value for MAX_PROF_XY_IND can be redefined at program level. This enables the original limit of up to 180 XY table sections to be increased.
(Siehe 'Number of table interpolation points' auf Seite 46.)

2.2.3.3.11 ST_PROF_XYVATAB (ST)

The 'ST_PROF_XYVATAB' contains the table information specific to AMK and then references the 'SMC_CAMXYVA' structure which is specific to 3S and contains the interpolation points.

Structure elements

<table>
<tr><th>Name</th><th>Type</th><th>Description</th></tr>
<tr><td>enType</td><td>ENUM</td><td>EN_PROF_TAB_TYPE
Table type
Designation for XYVA tables
<table>
<tr><td>Default</td><td>PROF_XYVATAB</td></tr>
</table>
<table>
<tr><th>Range</th><th>Meaning</th></tr>
<tr><td>PROF_
XYVATAB</td><td>X and Y positions, velocity, and acceleration defined by table values</td></tr>
</table></td></tr>
</table>

Name	Type	Description
uiNoElement	UINT	Element number of the last table element calculated, number of table interpolation points of the 'SMC_CAMXYVA' structure Range: 8... MC_CAM_REF.nElements-1 Default: 0
udMasterInc	UDINT	Increments of the master drive which produce a table cycle Max. table X value (not used for XYVA tables) Default: 20000
pstCamXYVA	POINTER	POINTER TO SMC_CAMXYVA Pointer to the 'SMC_CAMXYVA' structure containing the dX, dY, dV, and dA values. The sections of the 5th order polynomial are defined with these values.

Structure definition

```
TYPE ST_PROF_XYVATAB:
    STRUCT
        enType:EN_PROF_TAB_TYPE:=PROF_XYVATAB;
        uiNoElement:UINT:=0;
        udMasterInc:UDINT:=20000;
        pstCamXYVA: POINTER TO SMC_CAMXYVA;
    END_STRUCT
END_TYPE
```

Based on the CODESYS cam disk editor, the 'stCam_A':ARRAY [0...3] OF SMC_CAMXYVA structure, which is specific to 3S, is generated automatically with table interpolation points and 'stCam':MC_CAM_REF.

Abbildung 33: ST_PROF_XYVATAB: Cam disk editor

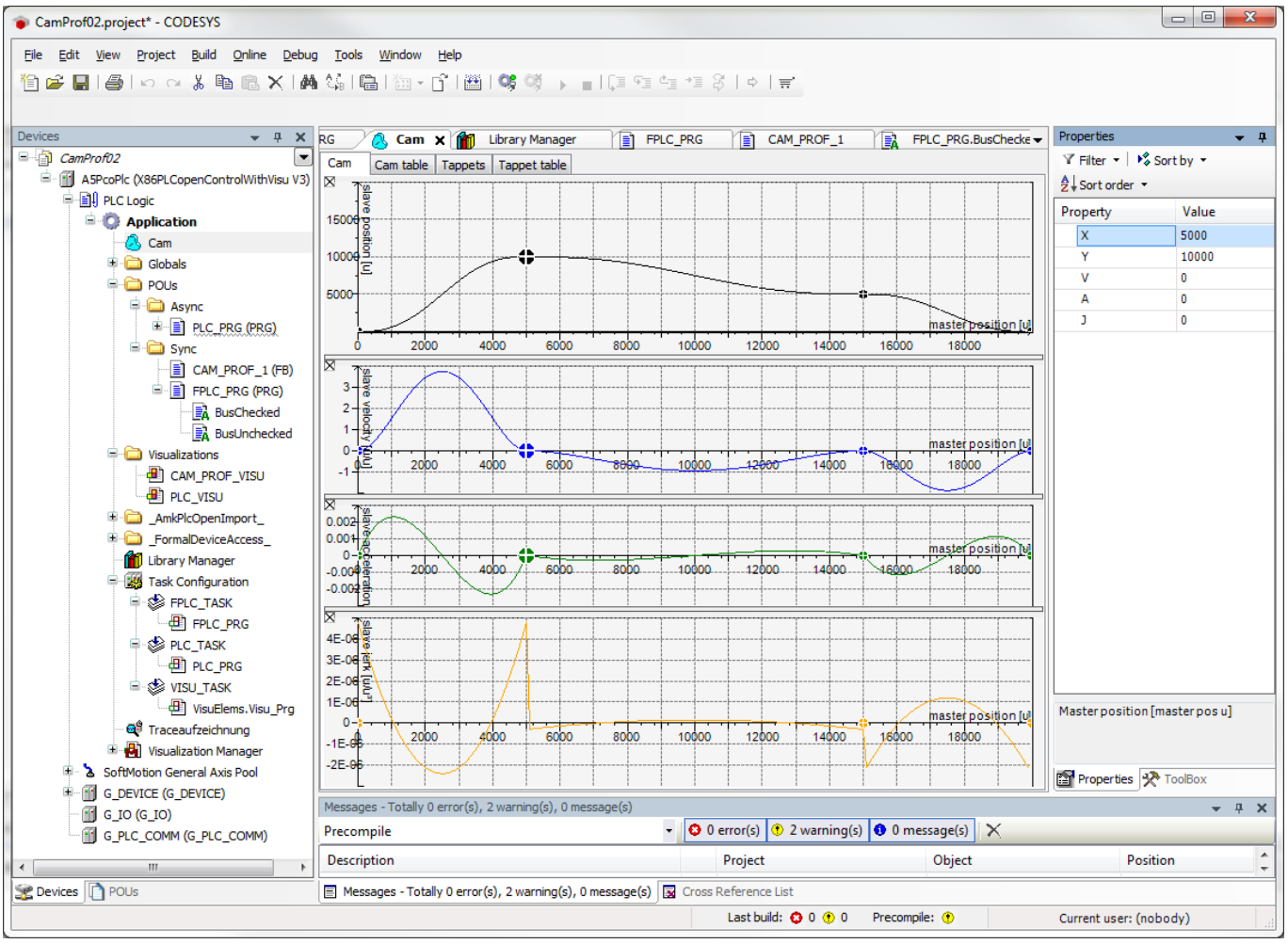
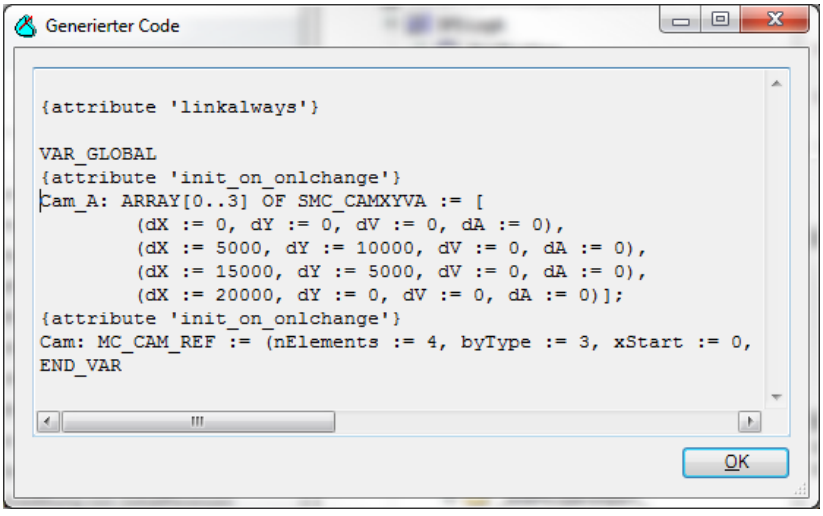


Abbildung 34: ST_PROF_XYVATAB: CAM structures specific to 3S



The definition of these structures is based on 3S libraries which are only integrated with the full Softmotion license.

2.2.3.3.12 ST_PROF_YTAB (ST)

The 'ST_PROF_YTAB' structure defines an XY table whose x axis is split equally. The table structure contains the y values of the function y = f(x).

The corresponding x values are generated in the 'CAM_PROF_1' block with 'uiNoElement'+1 equidistant points.
Thus:

$$A = \frac{udMasterInc}{uiNoElement}$$

where A: equidistant spacing

Structure elements

Name	Type	Description
enType	ENUM	EN_PROF_TAB_TYPE Table type, to differentiate between X and XY tables Default: PROF_YTAB Range / Meaning: **PROF_YTAB**: Equidistant X positions, Y positions defined by table value **PROF_YTAB_NL**: Equidistant X positions, Y positions defined by table value, not limited
uiNoElement	UINT	Element number of the last table element calculated, number of table interpolation points Range: 1 ... 360 Default: 360
udMasterInc	UDINT	Increments of the master drive which produce a table cycle Max. table X value Number of incoming increments at 'diInVal' necessary for the table to be run through once
diY	ARRAY	ARRAY [0..MAX_PROF_Y_IND] OF DINT Y value of table interpolation point diY[0]: 0: Y value at zero point of table diY[1]: 1st Y value of table diY[2]: 2nd Y value of table diY[360]: 360th Y value of table

Structure definition

```
MAX_PROF_Y_IND:UINT:=360;                    (* highest valid index for ST_PROF_TAB.diY[0…] *)

TYPE ST_PROF_YTAB:
    STRUCT
        enType:EN_PROF_TAB_TYPE:=PROF_YTAB ;
        uiNoElement:UINT:=MAX_PROF_Y_IND;
        udMasterInc:UDINT:=20000;
        diY:ARRAY[0…MAX_PROF_Y_IND] OF DINT;
    END_STRUCT
END_TYPE
```

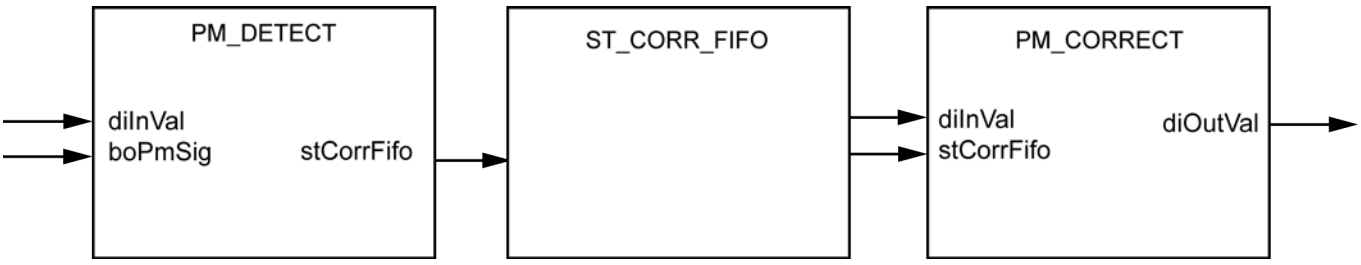
If the table type 'enType' = PROF_XYTAB_NL is selected, the value for 'MAX_PROF_Y_IND' can be redefined at program level. This enables the original limit of up to 360 Y table sections to be increased.
(Siehe 'Number of table interpolation points' auf Seite 46.)

2.2.3.4 PmControl

2.2.3.4.1 ST_CORR_FIFO (ST)

The 'ST_CORR_FIFO' is used to transfer correction values between the 'PM_DETECT' and 'PM_CORRECT' blocks.

Abbildung 35: ST_CORR_FIFO: Fundamental structure of printing mark control



The header information includes the following variables:

- 'uiInIndex'; to specify the write position (per PM_DETECT).
- 'uiOutIndex'; to specify the read position (with PM_CORRECT).

The array 'diCorrVal'[0 ... MAX_CORR_FIFO_IND] contains the correction values.

Header information	uiInIndex
	uiOutIndex
Correction values	diCorrVal[0]
	diCorrVal[1]
	...
	diCorrVal[MAX_CORR_FIFO_IND]

The corresponding index is incremented after every read or write operation. If the array limit 'MAX_CORR_FIFO_IND' is exceeded, the index is set to 0.

The following applies:

Before reading: uiInIndex = uiOutIndex -> FIFO is empty

Before writing: (uiInIndex + 1) MOD (MAX_CORR_FIFO_IND + 1) = uiOutIndex -> FIFO is full

Structure elements

Name	Type	Description
uiInIndex	UINT	Write index Range: 0 ... MAX_CORR_FIFO_IND
uiOutIndex	UINT	Read index Range: 0 ... MAX_CORR_FIFO_IND
diCorrVal	ARRAY	ARRAY [0..MAX_CORR_FIFO_IND] OF DINT Current correction value enter most recently in the FIFO structure 'stCorrFifo' Array for saving correction values

Structure definition

```
MAX_CORR_FIFO_IND: UNIT:=19;                    (* maximum valid index for diCorrVal *)

TYPE ST_CORR_FIFO:
        STRUCT
                uiInIndex:UNIT;
                uiOutIndex:UNIT;
                diCorrVal:ARRAY[0…MAX_CORR_FIFO_IND] OF DINT;
        END_STRUCT
END_TYPE
```

2.2.3.5 System

2.2.3.5.1 ST_PLC_VAR_POINTERS (ST)

Structure elements

Name	Type	Description
pbyInAsync	POINTER	POINTER TO BYTE Pointer to the asynchronous input variable range
pbyOutAsync	POINTER	POINTER TO BYTE Pointer to the asynchronous output variable range
pbyInSync	POINTER	POINTER TO BYTE Pointer to the synchronous input variable range
pbyOutSync	POINTER	POINTER TO BYTE Pointer to the synchronous output variable range

Structure definition

```
TYPE ST_PLC_VAR_POINTERS:
            STRUCT
                        pbyInAsync: POINTER TO BYTE;
                        pbyOutAsync: POINTER TO BYTE;
                        pbyInSync: POINTER TO BYTE;
                        pbyOutSync: POINTER TO BYTE;
            END_STRUCT
END_TYPE
```

2.2.4 Structures_1

2.2.4.1 AllElementsOfOneID

2.2.4.1.1 ST_ID_ALL (ST)

The 'ST_ID_ALL' groups all the elements of a parameter.

Structure elements

Name	Type	Description
diData	DINT	Parameter value
diMin	DINT	Minimum permissible value
diMax	DINT	Maximum permissible value

Name	Type	Description
udAttr	UDINT	Parameter attribute (according to SERCOS standard) Bit 0 ... 15 Scaling Bit 16 ... 18 Data length Bit 19 Function Bit 20 ... 23 Data type Bit 24 ... 27 Decimal places Bit 28 ... 30 Write-protected Bit 31 Not used
stUnit	STRUCT	ST_ID_UNIT Parameter unit displayed as a list with an ASCII string
stName	STRUCT	ST_ID_NAME Parameter name displayed as a list with an ASCII string

Structure definition

```
TYPE ST_ID_ALL:
    STRUCT
        diData:DINT;
        diMin:DINT;
        diMax:DINT;
        udAttr:UDINT;
        stUnit:ST_ID_UNIT;
        stName:ST_ID_NAME;
    END_STRUCT
END_TYPE
```

2.2.4.1.2 ST_ID_NAME (ST)

Parameter name, shown as list with ASCII string.

Structure elements

Name	Type	Description
uiActLen	UINT	Current list length
uiMaxLen	UINT	Maximum list length
strName	STRING	Parameter name

Structure definition

```
TYPE ST_ID_NAME:
    STRUCT
        uiActLen:UINT;
        uiMaxLen:UINT;
        strName: STRING(ID_NAME_SIZE);
    END_STRUCT
END_TYPE
```

2.2.4.1.3 ST_ID_UNIT (ST)

Parameter unit, shown as list with ASCII string.

Structure elements

Name	Type	Description
uiActLen	UINT	Current list length
uiMaxLen	UINT	Maximum list length
strUnit	STRING	Parameter unit

Structure definition

```
TYPE ST_ID_NAME:
    STRUCT
        uiActLen:UINT;
        uiMaxLen:UINT;
        strUnit: STRING(ID_UNIT_SIZE);
    END_STRUCT
END_TYPE
```

2.2.4.2 ElementaryAccess

2.2.4.2.1 ST_LIST_512 / ST_LIST_1024 / ST_LIST_2048 / ST_LIST_4096 (ST)

The 'ST_LIST_512' / '_1024' / '_2048' / '_4096' structures provide memory capacity for list parameters:

Structure	Memory capacity [bytes]	Header data [words]	User data [words]
ST_LIST_512	512	2	254
ST_LIST_1024	1024	2	510
ST_LIST_2048	2048	2	1022
ST_LIST_4096	4096	2	2046

Structure elements

Name	Type	Description
uiActLen	UINT	Current list length
uiMaxLen	UINT	Maximum list length
uiListEle	ARRAY	ST_LIST_512 : ARRAY [2..255] OF UINT ST_LIST_1024 : ARRAY [2..511] OF UINT ST_LIST_2048 : ARRAY [2..1023] OF UINT ST_LIST_4096 ARRAY [2..2047] OF UINT List elements user data

Structure definitions

```
TYPE ST_LIST_512:
    STRUCT
        uiActLen:UINT;
        uiMaxLen:UINT;
        uiListEle:ARRAY [2…255] OF UINT;
    END_STRUCT
END_TYPE
```

```
TYPE ST_LIST_1024:
    STRUCT
        uiActLen:UINT;
        uiMaxLen:UINT;
        uiListEle:ARRAY [2…511] OF UINT;
    END_STRUCT
END_TYPE

TYPE ST_LIST_2048:
    STRUCT
        uiActLen:UINT;
        uiMaxLen:UINT;
        uiListEle:ARRAY [2…1023] OF UINT;
    END_STRUCT
END_TYPE

TYPE ST_LIST_4096:
    STRUCT
        uiActLen:UINT;
        uiMaxLen:UINT;
        uiListEle:ARRAY [2…2047] OF UINT;
    END_STRUCT
END_TYPE
```

2.2.4.2.2 ST_LIST_VAR_LEN (ST)

The 'ST_LIST_VAR_LEN' structure provides memory capacity for list parameters.

Structure elements

Name	Type	Description
uiActLen	UINT	Current list length
uiMaxLen	UINT	Maximum list length
uiListEle	ARRAY	ARRAY [2..MAX_LIST_INDEX] OF UINT List elements user data

Structure definition

```
VAR_GLOBAL CONSTANT
   MAX_LIST_INDEX : INT := 2047;                (* maximum index of list elements, constants defined in AmkBase.lib *)
END_VAR

TYPE ST_LIST_VAR_LEN:
    STRUCT
        uiActLen:UINT;
        uiMaxLen:UINT;
        uiListEle:ARRAY [2…MAX_LIST_INDEX] OF UINT;
    END_STRUCT
END_TYPE
```

2.2.4.3 MoreIds

2.2.4.3.1 ST_ID_VALUE (ST)

The 'ST_ID_VALUE' structure contains the variable that describe a parameter.

Structure elements

Name	Type	Description
uiIDNo	UINT	Parameter number (ID)
uiParInst	UINT	Instance or Parameter set number or instance number
diIDVal	DINT	Parameter value

Structure definition

```
TYPE ST_ID_VALUE:
	STRUCT
		uiIDNo: UINT;
		uiParInst: UINT;
		diIDVal:DINT;
	END_STRUCT
END_TYPE
```

2.2.4.3.2 ST_N_ID_VALUES (ST)

The 'ST_N_ID_VALUES' structure groups all parameter values to be read and written.

Structure elements

Name	Type	Description
arr_stIdValue	ARRAY	ARRAY [1..MAX_INDEX_FOR_ID_VALUES] OF ST_ID_VALUE

Structure definition

```
VAR_GLOBAL CONSTANT
   MAX_INDEX_FOR_ID_VALUES : UINT := 10;     (* number of parameters*)
END_VAR

TYPE ST_N_ID_VALUES:
	STRUCT
		arr_stIdValue: ARRAY[1..MAX_INDEX_FOR_ID_VALUES] OF ST_ID_VALUE;
	END_STRUCT
END_TYPE
```

2.3 Projektinformationen

2.3.1 GetBooleanProperty (F)

Reserved for AMK internal use!

2.3.2 GetCompany (F)

Reserved for AMK internal use!

2.3.3 GetNumberProperty (F)

Reserved for AMK internal use!

2.3.4 GetTextProperty (F)

Reserved for AMK internal use!

2.3.5 GetTextProperty2 (F)

Reserved for AMK internal use!

2.3.6 GetTitle (F)

Reserved for AMK internal use!

2.3.7 GetVersion (F)

Reserved for AMK internal use!

2.3.8 GetVersionProperty (F)

Reserved for AMK internal use!

3 Appendix

3.1 Error bit information

Regardless of the type of access (READ_SDO / WRITE_SDO or READ_ID / WRITE_ID blocks), the following error codes describe the errors during data transport:

Error code	Error code from PLC (iErrID)	Description
0x00000002	0x0002	General error message
0x00000003	0x0003	Source module not available
0x00000004	0x0004	The addressed destination does not exist (routing address is incorrect)
0x00000005	0x0005	Memory errors
0x00000006	0x0006	Wrong module number
0x00000007	0x0007	Wrong element
0x00000008	0x0008	Resource error
0x00000009	0x0009	Protocol error (command)
0x0000000A	0x000A	Unused
0x0000000B	0x000B	Timeout
0x0000000C	0x000C	Internal error
0x0000000D	0x000D	Unknown command
0x0000000E	0x000E	Unused
0x0000000F	0x000F	Internal error
0x00000016	0x0016	No connection to target
0x00000017	0x0017	Error in 'Login', device already used

Valid for EtherCAT SOE

ID access (blocks READ_ID / WRITE_ID)

The error codes from the SOE slave device have the following meaning:

Error code	Error code from PLC (iErrID)	Description
0x00000000	0x0000	No error
0x00001001	0x1001	ID number not available
0x00001009	0x1009	Invalid access to element 1
0x00002001	0x2001	Name does not exist
0x00002002	0x2002	Name transmitted too short
0x00002003	0x2003	Name transmitted too long
0x00002004	0x2004	Name can not be changed
0x00002004	0x2004	Name is currently write protected
0x00003001	0x3001	Attribute does not exist
0x00003002	0x3002	Attribute transmitted too short
0x00003003	0x3003	Attribute transmitted too long
0x00003004	0x3004	Attribute can not be changed
0x00003005	0x3005	Attribute is currently write protected
0x00004001	0x4001	Unit not available
0x00004002	0x4002	Unit transmitted too short
0x00004003	0x4003	Unit transmitted too long
0x00004004	0x4004	Unit can not be changed
0x00004005	0x4005	Unit is currently write protected

Error code	Error code from PLC (iErrID)	Description
0x00005001	0x5001	Minimum input value not available
0x00005002	0x5002	Minimum input value transmitted too short
0x00005003	0x5003	Minimum input value transmitted too long
0x00005004	0x5004	Minimum input value can not be changed
0x00005005	0x5005	Minimum input value is currently write protected
0x00006001	0x6001	Maximum input value not available
0x00006002	0x6002	Maximum input value transmitted too short
0x00006003	0x6003	Maximum input value transmitted too long
0x00006004	0x6004	Maximum input value can not be changed
0x00006005	0x6005	Maximum input value is currently write protected
0x00007002	0x7002	Operating date transmitted too short
0x00007003	0x7003	Operating date transmitted too long
0x00007004	0x7004	Operating date can not be changed
0x00007005	0x7005	Operating date is currently write protected
0x00007006	0x7006	Operating date is less than the minimum input value
0x00007007	0x7007	Operating date is greater than the maximum input value
0x00007008	0x7008	Invalid operating date
0x00007009	0x7009	Operating date is write protected by password.
0x0000700A	0x700A	Operating date is write protected as a result of cyclic usage
0x0000700B	0x700B	Unauthorized indirect addressing
0x0000700C	0x700C	Operation date write protected as a result of other defaults (e.g., operating mode, ..)
0x0000700D	0x700D	Invalid floating number
0x0000700E	0x700E	Operating date write protected during 'parameterization level'
0x0000700F	0x700F	Operating date write protected during 'operating level'
0x00007010	0x7010	Procedure command already active
0x00007011	0x7011	Procedure command can not be interrupted
0x00007012	0x7012	Procedure command can not be executed at this time
0x00007013	0x7013	Procedure command can not be executed (invalid or incorrect parameters)
0x00008009	0x8009	General access error

Valid for EtherCAT COE and ACC

Index / sub-index access (blocks READ_SDO / WRITE_SDO)

The error codes from the COE / ACC slave device have the following meaning:

Error code	Error code from PLC (iErrID)	Description
0x05030000	0x5300	Toggle bit not changed
0x05040000	0x5400	SDO protocol timed out
0x05040001	0x5401	SDO Command Specifier invalid or unknown
0x05040002	0x5402	Invalid block size (Block Transfer mode only)
0x05040003	0x5403	Invalid sequence number (Block Transfer mode only)
0x05030004	0x5304	CRC error (Block Transfer mode only)
0x05030005	0x5305	Out of memory
0x06010000	0x6100	Access to this object is not supported
0x06010001	0x6101	Attempt, to write to a Write_Only parameter
0x06010002	0x6102	Attempt, to write to a Read_Only parameter
0x06020000	0x6200	Object is not present in the object directory
0x06040041	0x6441	Object can not be mapped to PDO

Error code	Error code from PLC (iErrID)	Description
0x06040042	0x6442	The number and / or the length of the mapped objects would exceed the PDO length
0x06040043	0x6443	General parameters Incompatibility
0x06040047	0x6447	General internal error in the device
0x06060000	0x6600	Access due to hardware failure aborted
0x06070010	0x6710	Data type or parameter length do not match or are unknown
0x06070012	0x6712	Data type does not match, parameter length too long
0x06070013	0x6713	Data type does not match, parameter length too short
0x06090011	0x6911	Sub-index not available
0x06090030	0x6930	General value range error
0x06090031	0x6931	Value range error: Parameter value too large
0x06090032	0x6932	Value range error: Parameter value too small
0x06090036	0x6936	Maximum value is less than minimum value
0x060A0023	0x6A23	Resource not available
0x08000000	0x8000	General error
0x08000020	0x8020	Data cannot be transferred or stored to the application
0x08000021	0x8021	Access not possible due to local application
0x08000022	0x8022	Can not access due to current device status
0x08000023	0x8023	Object Dictionary dynamic generation fails or no Object Dictionary is present (e.g. Object dictionary is generated from file and generation fails because of a file error)

3.2 Table 1: Global AmkFile function block error codes

Error code	Meaning
9	Invalid path or device
11	Global error - a more detailed description of this error cause does not exist
12	Global error - a more detailed description of this error cause does not exist
15	Too many files - there is no more memory capacity available in the file table
16	No more files found - the cannot find any files with this search criterion
19	File not found - the file name used could not be found in the system
26	Access denied - access is not possible at the current time, e.g. due to file access from another source
28	Directory is empty
29	Invalid directory
31	Data medium full - there is no more memory capacity available on the data medium
32	Disk full - there is no more memory capacity available
46	Directory already exists

Glossary

A

ACC
AMK CAN Communication (CAN bus interface with standard CANopen protocol DS301 and additional hardware synchronization signal)

ARRAY
List with equal format elements

ASCII
American Standard Code for Information Interchange

C

CoE
CAN application protocol over EtherCAT

D

Default
Factory setting

E

EtherCAT
Real-time Ethernet bus

F

FB
Function block

FIFO
First in - first out (Memory which outputs the saved values the same sequence they were saved)

FPLC_PRG
Real-time PLC task, synchronized to device cycle

I

ID
Parameter identification numbers acc. to SERCOS Standard

P

PDK_xxxxxx_abcdefgh
Product documentation; xxxxxx - AMK part no. , abcdefgh - name

PLC_PRG
Task which is not synchronized to the device cycle

POU
Program organization unit (PLC program elements; types program, function or function block)

S

SoE
Servodrive Profile (SERCOS) over EtherCAT (Acc. to IEC 61800-7-300)

Your opinion is important!

With our documentation we want to offer you the highest quality support in handling the AMKmotion products.
That is why we are now working on optimizing our documentation.
Your comments or suggestions are always of interest to us.
We would be grateful if you take a bit of time and answer our questions. Please return a copy of this page to us.

e-mail: Documentation@amk-motion.com
or
fax no.: +49 7021/50 05-199

Thank you for your assistance.
Your AMKmotion documentation team

1. How would you rate the layout of our AMKmotion documentation?

   (1) very good (2) good (3) satisfactory (4) less than satisfactory (5) poor

2. Is the content structured well?

   (1) very good (2) good (3) moderate (4) hardly (5) not at all

3. How easy is it to understand the documentation?

   (1) very easy (2) easy (3) moderately easy (4) difficult (5) extremely difficult

4. Did you miss any topics in the documentation?

   (1) no (2) if yes, which ones:

5. How would you rate the overall service at AMKmotion?

   (1) very good (2) good (3) satisfactory (4) less than satisfactory (5) poor

AMKmotion GmbH + Co KG
Phone : +49 7021/50 05-0, fax: +49 7021/50 05-199
E-Mail: info@amk-motion.com
Homepage: www.amk-motion.com