Parameter Reference

Basics

Basically there are three parameter types: Init Parameter, User Parameter and Program Parameter:

Init Parameters

… are default values which are stored in the positioning control at the factory. After delivery these values are active and can be changed by the user.

The Init-parameters can by re-set by means of a parameter reset in the menu **Controller → Reset** at any time. Deleting the memory in **Controller → Memory → Delete EPROM** also re-set the parameters to the init values.

User Parameters

… are the (Init) parameters changed by the user. These parameters are saved in the EPROM and remain valid even after the control has been turned off.

The user parameters are active whenever the controller is switched on, or will be activated before every program execution.

They are valid for all programs.

Program Parameters

The parameters which were changed within a program (with the SET command) are called program parameter. Thus it is possible to change the parameters at short notice without lasting consequences.

In contrast to the Init or user parameters these parameters are only valid while the program is being executed and in direct mode; thereafter they are erased and replaced by the user parameters.

For example, if you want to make the maximum trailing distance very large, then set this filter parameter with the SET command. After the program has been completed the user parameters are valid again.

Parameter Groups

For a better overview the parameters for the APOSS program are divided into groups; they are valid for all controls:

Global Parameters GL

All parameters independent of an axis such as the ***Activated Program Number*** PRGPAR and the I/O parameters such as ***Clear error*** I_ERRCLR are combined in the GL group and can be processed with GET and SET.

Axis parameters AX...

This group contains all axis-specific parameters which can be processed with the GET and SET commands and need axis information in the command:

Basic Settings (AXB, has been previous AXV)
Encoder Data (AXE)
Predefined I/Os (AXI) – input/output configuration
Homing (AXH) – parameters affecting home motion
Synchronization and Sync Marker (AXS)
Position Regulation (AXR) – everything concerning control behavior
Amplifier (AXA) – General amplifier data, PI velocity and current control

The column parameter group (AXE, AXR etc.) references to the dialogue fields **Controller → Parameters → Edit**, where you can define and change the parameters.

Using Parameters

There are three methods to access parameters:

- PC Software APOSS
- External PLC
- CAN bus

Parameter Changes and Storage

Parameters changed via the menu **Controller → Parameters → Axis** are saved in an EPROM and thus retained at power down.

Parameters changed from the APOSS application program with the command SET are only stored in RAM and thus lost at power down.

Parameters changed from the APOSS application program with the command SET are only active while the application program is running. These parameters can be saved in an EPROM and thus retained at power down by means of the command SAVEPROM.

!!! Please note that an EPROM has limited lifetime; but in can be reprogrammed approximately 10000 times.

General Information on the Parameter Values

Some limiting values are listed at 1 billion to make them more easily readable. However, the exact value is 1,073,741,823 (= MLONG)

Whether the input range listed is exceeded is not checked by the program, since due to the large domain there are no suitable test possibilities.

!!! Even within the areas indicated illogical inputs can result from the large differences in performance of the motors and the wide variety of possible applications. Therefore, it is the responsibility of the programmer and the user to observe the performance ranges of the drive and of the system.

!!! If the value of the parameter is out of the range defined by the minimum and maximum value, the command is not displayed and executed correct.

Parameter Overview

Parameter Code	Parameter Name	Internal Par. No.	Par. Group	Unit	Init Value
ACCMAXQC	Maximum acceleration	10	AXB	qc/st² * 1/65536	
AINFTIME	Defines the filter time for analog input	109	GL	–	0
AMPCOMMTYPE	Definition of the motor type in use for MACS control units with integrated power amplifiers.	95	AXA	–	0
AMPENCNO	Definition of the encoder input, which is in use as the feedback source of the velocity control loop.	126	AXA	–	-1
AMPENCRES	Resolution of the encoder in use by the velocity control loop.	127	AXA	qc/turn	-1
AMPENCRPM	Maximum velocity of the encoder in use by the velocity control loop.	128	AXA	rounds/min.	0
AMPMAXCUR	Maximum allowed motor current for MACS control units with integrated power amplifiers.	94	AXA	mA	2000
AMPMODE	Definition of the control loop structure in use for MACS control unit versions with integrated power amplifiers.	97	AXA	–	0
AMPPWMFREQ	Frequency of the PWM power amplifier of integrated MACS4 devices.	96	AXA	Hz	24000
BANDWIDTH	PID Bandwidth	35	AXR	%	1000
CANBAUD	CAN-Baud rate	101	GL	–	5
CANNR	CAN node ID	100	GL	–	0
CANSYNCTIMER	Defines the cycle time for sending SYNC telegrams on the CAN bus.	114	GL	ms	0
CURKILIM	Limit value for the integral sum of the PI current / torque control loop.	104	AXA	–	1000
CURKINT	Integral value of the PI current / torque control loop.	93	AXA	–	100
CURKPROP	Proportional value of the PI current / torque control loop	92	AXA	–	200
DFLTACC	Default acceleration	34	AXB	%	50
DFLTVEL	Default velocity	33	AXB	%	50
DRIVETYPE	Defines the mode and kind of reference output in use by the axis control.	0	AXR	–	0
ENCCONTROL	Configuration of position evaluation after a change of encoder source.	107	AXE		0
ENCODER	Encoder counts per turn	2	AXE	counts/ revolution	500
ENCODERCLOCK	Internal or external clock generation for SSI encoder	73	AXE	–	1
ENCODERFREQ	Specifies the frequency for SSI encoder.	74	AXE	kHz	262000
ENCODERTYPE	Defines the signal source of the feedback encoder for the axis control.	27	AXE	–	0

Parameter Code	Parameter Name	Internal Par. No.	Par. Group	Unit	Init Value
ENDSWMOD	Behavior at end limit switch	44	AXI	–	0
ERRCOND	Behavior after error	43	AXI	–	0
ESCCOND	Condition on program termination	70	AXI	–	0
FEEDDIST	User unit factor to get CANopen compatible user units.	112	AXE		1
FEEDREV	User unit factor to be CANopen compatible.	113	AXE		1
FFACC	Acceleration feed forward	37	AXR	%	0
FFVEL	Velocity Feed-forward	36	AXR	%	0
HOME_FORCE	Force HOME?	3	AXH	–	0
HOME_OFFSET	Zero point offset from Home Position	42	AXH	qc	0
HOME_RAMP	Ramp for home motion	41	AXH	%	10
HOME_VEL	Velocity of home motion	7	AXH	%	10
HOME_TYPE	Behavior during home motion	40	AXH	–	0
HOMEZEROVEL	Scaled velocity for moving into the encoder index as part of the HOME command or CANopen homing procedures.	116	AXH	typ. %	10
I_BREAK	Input for abort	105	GL	–	0
I_CONTINUE	Continue program	106	GL	–	0
I_ERRCLR	Clear error	107	GL	–	0
I_NEGLIMITSW	Negative limit switch	47	AXI	–	0
I_POSLIMITSW	Positive limit switch	46	AXI	–	0
I_PRGCHOICE	Input for beginning program choice	104	GL	–	0
I_PRGSTART	Input for program start	103	GL	–	0
I_REFSWITCH	Input for reference switch	45	AXI	–	0
I2TFTIME	Filter time in ms to build up the I^2 filtered value.	117	AXA	ms	0
I2TLIMIT	Limitation of the I2TVALUE	118	AXA	A^2 (1/1000)	0
JERKMIN	Minimum time required before reaching the maximum acceleration.	98	AXB	ms	0
KDER	Derivative factor for PID control	12	AXR	–	1
KILIM	Limit value for integral sum for PID control	21	AXR	–	0
KILIMTIME	Time (ms) which is used to increase or decrease the integral limit of the position control loop up to KILIM.	105	AXR	ms	0
KINT	Integral value for PID control	13	AXR	–	0
KPROP	Proportional factor for PID control	11	AXR	–	30
MENCCONTROL	Configuration of master position evaluation after a change of encoder source.	108	AXE		0
MENCODER	Resolution of the encoder 1 (master)	30	AXE	counts/ revolution	500
MENCODERCLOCK	Internal or external clock generation for SSI master encoder	77	AXE	–	1

Parameter Code	Parameter Name	Internal Par. No.	Par. Group	Unit	Init Value
MENCODERTYPE	Defines the signal source of the master encoder.	67	AXE	–	0
NEGLIMIT	Negative software end limit	4	AXI	qc	–500000
O_AXMOVE	Output for motion command active	64	AXI	–	0
O_BRAKE	Output for mechanical brake	48	AXI	–	0
O_ERROR	Output for error	108	GL	–	0
POSDRCT	Rotational direction	28	AXE	–	1
POSENCQC	User unit factor to be CANopen compatible.	114			1
POSENCREV	User unit factor to be CANopen compatible.	115			1
POSERR	Maximum tolerated position error	15	AXR	qc	20000
POSERRTIME	Time frame [ms] for triggering position error state.	111	AXR	ms	0
POSFACT_N	Denominator user factor	26	AXE	–	1000
POSFACT_Z	Numerator user factor	23	AXE	–	1000
POSLIMIT	Positive software end limit	5	AXI	qc	500000
PRGPAR	Activated program number	102	GL	–	–1
PROFTIME	Scan time for profile generator	29	AXR	ms	1000
RAMPMIN	Maximum acceleration	31	AXB	ms	1000
RAMPTYPE	Ramp type	32	AXB		0
REGWMAX	Size of the control window (activation)	38	AXR	qc	0
REGWMIN	Size of the control window (deactivation)	39	AXR	qc	0
REVERS	Reverse behavior of the controlled axis.	63	AXR	–	0
SWNEGLIMACT	Negative software end limit active	19	AXI	–	0
SWPOSLIMACT	Positive software end limit active	20	AXI	–	0
SYNCACCURACY	Size of the precision window for position synchronization	55	AXS	qc	1000
SYNCCFFVEL	Velocity feed forward [per mill of VCMD] for synchronization modes.	109	AXS		0
SYNCFACTM	Synchronization factor master (M:S)	49	AXS	qc	1
SYNCFACTS	Synchronization factor slave (M:S)	50	AXS	qc	1
SYNCFAULT	Marker number for fault	57	AXS	–	10
SYNCMARKM	Marker number for master	52	AXS	–	1
SYNCMARKS	Marker number for slave	53	AXS		
SYNCMFPAR	Marker filter configuration	17	AXS	–	0
SYNCMFTIME	Filter Time for Marker Filter	18	AXS	1 ms	0
SYNCMMAXCORR	Maximum Marker Correction	6	AXS	qc	0
SYNCMPULSM	Master Marker Distance	58	AXS	qc	500
SYNCMPULSS	Slave Marker Distance	59	AXS	qc	500
SYNCMSTART	Start behavior for marker synchronization	62	AXS	–	0
SYNCMTYPM	Master Marker Type	60	AXS	–	0

Parameter Code	Parameter Name	Internal Par. No.	Par. Group	Unit	Init Value
SYNCMTYPS	Slave Marker Type	61	AXS	–	0
SYNCMWINM	Master Marker Tolerance Window	68	AXS	qc	0
SYNCMWINS	Slave Marker Tolerance Window	69	AXS	qc	0
SYNCOFFTIME	Offset Filter Time	16	AXS	ms	0
SYNCPOSOFFS	Position offset for positioning synchronization	54	AXS	qc	0
SYNCREADY	Marker number for ready	56	AXS	–	0
SYNCSFTIME	Filter time [ms] for slave marker correction.	106	AXS	–	0
SYNCTYPE	Type of synchronization	51	AXS	–	0
SYNCVELREL	Relative Slave Velocity Limit	66	AXS	%	0
SYNCVFLIMIT	Sync error window [qc] for automatic deactivation of SYNCVFTIME.	110	AXS		0
SYNCVFTIME	Velocity filter	65	AXS	τ_filt (µs)	0
TESTTIM	Time in target window	24	AXI	ms	0
TESTVAL	Target window limit value	25	AXI	qc	1
TESTWIN	Size of target window	8	AXI	qc	0
TIMER	Sampling time for PID control and for sending output value to the drive.	14	AXR	ms	1
VELKILIM	Limit value for the integral sum of the PI velocity control loop.	103	AXA	–	1000
VELKINT	Integral value of the PI velocity control loop.	91	AXA	–	5
VELKPROP	Proportional value of the PI velocity control loop.	90	AXA	–	200
VELMAX	Maximum velocity	1	AXB	RPM	1000
VELMAXQC	Maximum velocity (read only)	9	AXB	qc/st * 1/65536	
VELRES	Velocity resolution	22	AXB		100
VMENCMTYP	Virtual master encoder marker type.	118	GL	–	0
VMENCRES	Virtual master encoder resolution.	117	GL	qc	500
VMENCTYP	Defines the source of the virtual encoder and the encoder type, which is in use if virtual master is disabled.	116	GL	–	0

Axis Parameters in Detail

DRIVETYPE - 0

Content Defines the mode and kind of reference output in use by the axis control.

Description DRIVETYPE defines the reference output to be used by the axis control. Following "output channels" can be used and defined as reference outputs:

- Local analog output of the control
- Analog output on a CANopen I/O module
- Process data word (PDO) to be used by a CANopen amplifier

The parameter value of DRIVETYPE is referred to as 'device type'. The number of the device type 'stands' for the used drive unit as aspect of the reference as well as of the actual value feedback source. The device type is used for the calculation of the parameter value of ENCODERTYPE (27) and MENCODERTYPE (67) if the encoder information is transferred via the bus.

DRIVETYPE = 0 The default device type is 0. This means for the parameter DRIVETYPE, that the setpoint setting is done via the local output of the control.

DRIVETYPE ≠ 0 = reference is transferred via the bus All other settings (value ≠ 0) mean that the reference information is transferred via the bus (e.g. CAN bus) to the target devices (e.g. analog CANopen module or CANopen amplifier). The bus device must be connected to the so-called slave bus, if a control with multiple separated CAN bus networks (e.g. MACS3) is in use. The bus node ID must be defined via the parameter REVERS (63).

Handling of the automatically generated CAN objects The necessary CAN objects (PDOs, GUARD-, SYNC-object) for the reference data exchange via the bus are defined automatically and the bus device is also initialized (NMT0). This object generation, which is done in the background, corresponds basically to the command CANINI. In contrast to the application object generation using CANINI, the automatic generated objects by DRIVETYPE remain untouched even when an updated CANINI definition is done by the application later on.

Even a CANDEL –1 command does not delete or stop the objects, which were automatically established by the DRIVETYPE setting. The automatic generated objects can just be deleted or reconfigured by a new differing SET DRIVETYPE command. Also in the case of program abort (by means of **Esc**) the objects remain existent up to the next program start. The start of a program deletes any formerly defined objects and initializes new objects according to the setting in the permanent parameters or at run-time according to any temporary parameter definition inside the program code.

Guarding When guarding is configured (i.e. DRIVETYPE > 0), the bus node is monitored and a missing feedback to the GUARD object (e.g. because a node malfunction) triggers the error 88. It is up to the user to take care that the parameter REVERS (63) is set properly (corresponding to the bus-ID), the bus node is configured (baud rate, bus-ID), the drive is connected, powered up, and ready for communication at the time when the parameter DRIVETYPE will be defined.

! ! ! If the bus node fails in a DRIVETYPE > 0 configuration, the guarding must be switched off by the application. This can be done within the error handling routine by setting of the negative DRIVETYPE value, until the failure of the device is fixed. Otherwise the identical error message comes up again and again after every deletion of the control error status 88 (by means of ERRCLR).

Adaptation of regulation cycle time

If the set point information is given to bus devices (CANopen I/O modules or amplifiers) which cannot receive and evaluate the PDO in a 1 ms sample rate, the regulation cycle time must be set higher with the command SET TIMER (e.g. Lenze to 5 ms, CANopen-I/O module 3 ms).

! ! !

Please note, that the rate of the SYNC object was also adapted to the parameter TIMER of the fastest axis until firmware version 6.5.16. For up-to-date firmware versions (6.5.16 or higher) the global parameter CANSYNCTIMER specifies the periodicity of sending the SYNC telegram now. This parameter has to be set to a proper value, which fits to the slowest CAN slave device in use.

PDO mapping for CANopen drives

In case of CANopen amplifiers, the so-called PDO-mapping must be done by the application before setting the parameter DRIVETYPE and ENCODERTYPE. PDO-mapping means the definition of which amplifier objects are linked into the transferred PDOs. The choice of most suitable objects depends on the CANopen amplifier. You can get sample source code suitable to the following mentioned device types from the zub machine control AG. Please tell us the used amplifier for that via e-mail (info@zub.ch).

Using analog output of CANopen I/O module

Instead of the analog output of the controller, also the analog outputs of a CANopen I/O module can be used to drive an amplifier. This is done by setting the DRIVETYPE to 91 – 94 depending on the number of analog output which is used on the module (1..4). Additionally, the REVERS must be set to ID * 100 to tell the controller which I/O module to use. With most of the I/O modules, you should set the TIMER to a value of 3, because sometimes they do not behave correctly if they get a PDO every ms. Be aware that baud rate is set accordingly and that other modules do not overload CAN BUS.

! ! !

This only works on slave bus.

This feature is especially used by a MACS3-2ax to control the second axis. In that case, normally the master encoder is used for feedback. (This is the default for a 2-axis system). If you have a 3-axis system, remember to disable axis 3 (MOTOR OFF x(3)) or to define a CAN-Encoder for axis 3. Otherwise, axis 3 will use the same encoder (master) by default and that will lead to a position error on axis 3 if you move axis 2.

The following lines give an example of how to use it.

```
canid = 3
MOTOR OFF x(2)
SET TIMER 1 / 3
SET DRIVETYPE x(2) –91 // set DRIVETYPE to CAN analogue output no. 1 without guarding
SET REVERS x(2) (canid * 100) //  use CAN I/O module with ID canid
MOTOR ON x(2)
```

DRIVETYPE for Danfoss Drives

With “DRIVETYPE 10” Danfoss Drives are supported. Here we use PDO-2 at the moment because mapping is not supported.

Every DRIVETYPE can be combined with ±100. (Danfoss i.e. 110 or –110). This means that the control word is not set any more by the controller. Neither MOTOR OFF nor other commands influence the control word anymore. The application is responsible for maintaining the control word by a command OUTDA.

See the following example:

```
#define FC_RUN      (0x047C)
#define FC_COAST    (0x0474)
#define FC_STOP     (0x063C) // ramp 2 active

SET ERRCOND 5   // don't do anything in case of error
PRINT "FC_RUN"
OUTDA FU1_PROCESS (FC_RUN << 16)   // set the FC in active mode

CSTART
CVEL 50
DELAY 600

PRINT "stop now "
OUTDA FU1_PROCESS (FC_STOP << 16)  // stop the FC with RAMP 2
DELAY 5        // wait until FC has seen new control word
MOTOR OFF  // now set reference to zero
DELAY 2000   // wait until ramp down ended
PRINT "coast now "
OUTDA FU1_PROCESS (FC_COAST << 16)  // now set FC to coasting mode
```

DRIVETYPE for Servostar

This DRIVETYPE uses PDO1 to communicate. Those PDOs must be configured as follows

Rx – Controlword (2 bytes), Reference (4 bytes)
Tx – Statusword (2 bytes), Position (4 bytes)

The following CANopen commands are issued

Error -> ShutDown (0x06)
Motor Off -> ShutDown (0x06)
Motor On -> Enable (0x0F)

In case of ERRCOND = 4 (constref), the error or motor off results in a QuickStop (0x02). This means that after such a quick stop, the ERRCOND must be set to 0 and then a MOTOR OFF must be executed to come back to the ReadyToSwitchOn state. Then you can reset the ERRCOND to 4 and use a MOTOR ON to change to OperationEnable state.

DRIVETYPE for Kollmorgen AKD

Starting with firmware 6.7.53 the new "DRIVETYPE 1"3 supports PDOs holding position and additional latching information, which is provided by the servo amplifier AKD of Kollmorgen:

RxPdo-1 contains StatusWord(2), ActualPosition(2), LatchPosition(2) ,LatchStatus(2)
TxPdo-1 contains ControlWord(2), CommandVelocity(4), LatchControl(2).

Velocity (= CommandVelocity) is transmitted in 1/1000 rpm. To calculate this velocity the VELMAX must be set correctly before the DRIVETYPE is set. If VELMAX is not the demanded scaling factor, it is also possible to use the new parameter AMPENCRPM to define the maximum velocity.

Portability

DRIVETYPE is available starting with control version 6.1.14 and compiler version 6.1.0;
DRIVETYPE 91 .. 94 (using the analog output of a CANopen I/O module) with version 6.4.41;
DRIVETYPE 10 for Danfoss drives with version 6.5.08;
DRIVETYPE 11 for Servostar with version 6.6.32.;
DRIVETYPE 12 for Lenze 8400 starting with version 6.6.81
DRIVETYPE 13 for Kollmorgen Servostar starting with version 6.7.53

Parameter Group

(not displayed in axis parameter fields)

Cross References

ENCODERTYPE, MENCODERTYPE, REVERS, TIMER, CANSYNCTIMER

Unit

–

Value Range –94 … 94
(= Device types at parameter value range of ENCODERTYPE (27) and MENCODERTYPE (67))

- 0 = Standard drive (analog output)
- 1 = CAN drive of type Lenze
- 2 = CANopen servo amplifier based on DS402
- 3 = CANopen servo amplifier maxon EPOS
  (customized expanded DS402 mode)
- 4 = CANopen servo amplifier zub DSA
  (customized expanded DS402 mode)
- 5 = CAN drive of type Lenze with a maximum reference value of 0x4000 (instead of 0x5000); this is conform to Profibus and can be configured via the parameter C011 in the Lenze drive.
- 10 = Danfoss drives (PDO 3 in use)
  ±100 (110, -110) control word is not set
- 11 = Servostar
- 12 = Lenze 8400
  (like Danfoss Drives, i.e. DRIVETYPE 10, but PDO 1 in use.)
- 13 = Kollmorgen AKD
- 91 = Analog output 1 to a CANopen I/O module
- 92 = Analog output 2 to a CANopen I/O module
- 93 = Analog output 3 to a CANopen I/O module
- 94 = Analog output 4 to a CANopen I/O module

! ! ! Negative values deactivate the guarding!

The bus device must be connected to the so-called slave bus if a control with multiple separated CAN-bus networks (e.g. MACS3) is in use. There is no need to use the bus offset identifier in the parameter value of DRIVETYPE due to this fixed bus network allocation.

Init Value 0

VELMAX - 1

Content	Rated speed of the drive.
Description	VELMAX defines the rated speed of the drive. This value is listed in RPM and is needed for the calculation of ramps and actual velocities.
! ! !	The nominal speed refers to the speed of the encoder.
Parameter Group	AXB Basic Settings
Unit	RPM
Value Range	1 … 65535
Init Value	1000

ENCODER - 2

Content Encoder counts per revolution

Description The parameter ***Encoder counts per revolution*** contains the position feedback transmitter of the actual encoder (incremental or absolute encoder) in relation to one motor revolution.

The number of quad-counts [qc] per revolution is calculated from this information. Quad-counts are the basic units for all path measurements. Quad-counts are generated by extracting of all edges of the A and B tracks. One encoder count corresponds to four quad-counts. In the case of absolute encoders, the absolute values are returned 1:1.

The number of quad-counts per revolution is needed for the index pulse search during the reference drive and for the conversion of velocity and acceleration to internal units.

The ***Encoder counts per revolution*** also supplies information whether during a HOME or INDEX movement the index signal has been missed. If more than a complete revolution is executed without registering an index signal then the corresponding error message will be made.

! ! ! Any change of the parameter ENCODER, means that the setting of the tuned position control loop parameters KPROP, KDER, KINT also have to be adapted.
If the value of ENCODER is doubled, then KPROP, KDER and KINT have to be halved.
If the value of ENCODER is halved, then KPROP, KDER and KINT have to be doubled to get the same position control behavior afterwards.

Parameter Group AXE Encoder

Cross Reference KPROP, KDER, KINT

Unit counts/revolution [PPR]

Value Range 1 … MLONG

! ! !	M1 and other new controller	No negative values are allowed. Whether index pulses are being used or not is entered in the parameter HOME_TYPE (40).
! ! !	MOCON, M1series, MODAK	All controllers up to version 5: The sign defines, if index pulses are being used or not: If index pulses are not used, the resolution have to be a negative value, in the other case a positive value.

Init Value 500

Limit values In order to guarantee correct position value capturing and control loop function, the maximum encoder frequency has to be taken in account. If the maximum encoder input frequency, specified in the MACS data sheet is exceeded, position information gets lost. The multiplication result of the parameters ENCODER and VELMAX has to be less than the specified maximum encoder frequency given in the MACS data sheet.

HOME_FORCE - 3

Content Forced movement to home position

Description If this parameter is set to yes = 1, then movement to the home position must be completed before any other positioning movement can be completed.

For a motion command that is not executed with a terminated home run the error 6 is triggered.

! ! ! For safety reasons and to avoid false positioning the parameter should always be set to 1 and thus forcing tracking of the home position. However, in this case it is necessary to consider that all programs must complete a HOME command before the first motion command in order to receive perfect functioning.

Parameter Group AXH Homing

Unit –

Value Range 0 … 1

0 = Home-run is not forced
After being turned on the current position is valid as the real zero point

1 = Home-run is forced
After turning on the control and after changing axis parameters a forced tracking of the home position must be made before a motion command is executed directly or by the program.

Internally the parameters can also contain the value 255, which indicates that a forced tracking of the HOME position is necessary and has already taken place.

Init Value 0

NEGLIMIT - 4

Content	Negative software limit switch
Description	NEGLIMIT indicates the ***negative position limit*** for all movements. If this value is exceeded then an error is triggered. NEGLIMIT is only active if SWNEGLIMACT (19) has been set. If a positioning command is entered which exceeds the limits set, then it is not executed.
! ! !	When using the command DEFORIGIN the path limitation is automatically adapted so that the original position of the positioning range is maintained.
! ! !	The path limitation is always given in quadcounts.
Parameter Group	AXI Predefined I/Os
Unit	qc
Value Range	–MLONG … MLONG
Init Value	–500000

POSLIMIT - 5

Content	Positive software limit switch
Description	POSLIMIT indicates the ***Positive position limit*** for all movements. If this value is exceeded then an error is triggered. POSLIMIT is only active if SWPOSLIMACT (20) is set. If a positioning command is entered which exceeds the limits set, then it is not executed.
! ! !	When using the command DEFORIGIN the path limitation is automatically adapted so that the original position of the positioning range is maintained.
! ! !	The path limitation is always given in quadcounts.
Parameter Group	AXI Predefined I/Os
Unit	qc
Value Range	–MLONG … MLONG
Init Value	500000

SYNCMMAXCORR - 6

Content Limits the maximum correction done by marker correction.

Description SYNCMMAXCORR is used to limit the maximum correction done by marker correction. This is working with SYNCM and SYNCC. The value is given in qc (slave) by the customer using the SET SYNCMMAXCORR command.

The value > 0 limits the marker correction by the given value. So if the correction would be bigger it is limited to the given value

The value < 0 sets the parameter so that no correction at all is done. The customer can switch off marker correction by this value

! ! ! If you have set SYNCMFTIME (18) or SYNCVFTIME (65) (negative), this correction will be spread over a certain time, depending on these factors.

Portability Support of value < 0 is starting with version 6.6.37

Parameter Group AXS Synchronization

Unit qc

Value Range -MLONG … MLONG

- 0 = no limitation
- > 0 = limits the marker correction by the given value
- < 0 = no correction at all

Init Value 0

HOME_VEL - 7

Content Velocity for movement to home position

Description HOME_VEL determines the ***HOME Velocity***, with which the movement to the reference switch is made. The velocity statement refers to the rated speed and depends on the VELRES (22) parameters. Generally this statement is made in % of the rated speed.

$$\text{Home Velocity [RPM]} = \text{HOME\_VEL (7)} * \frac{\text{VELMAX (1)}}{\text{VELRES (22)}}$$

! ! ! Since the program always searches for the reference switch in the same direction of rotation (depending on sign) this should be set at the limits of the motion area. Only in this manner is it possible to guarantee that the drive actually moves towards the reference switch when moving home and not away from it.

In order to maintain a good repeatability of the reference motion no more than 10% of the maximum speed should be used.

Parameter Group AXH Homing

Unit % (VELRES = 100)

Value Range – VELRES … VELRES

A negative sign means searching in the other direction.

Init Value 10

TESTWIN - 8

Content: Size of the target window.

Description: TESTWIN indicates the size of the target window. A position is only viewed as reached when the reference-run (trapeze) is executed, the actual position is within the window and the velocity is less than TESTVAL (Precondition: TESTWIN and TESTTIM are activated.)

In this content the velocity is given as TESTVAL in qc/TESTTIM.

The controller waits to execute the next command until the actual position is within the target window.

If TESTWIN is not active (0), then the target has been reached if the set position is the target position. However, this does not necessarily correspond with the actual position of the drive.

! ! ! If the target window surrounding the end position is selected to be too small, the drive could move in a very small area around the end position without reaching the target window. Thus the program would be 'stuck' after the corresponding positioning command.

A target window of 0 deactivates the monitoring of the actual position and only monitors the command position.

! ! ! Modified handling of TESTWIN to adapt to the needs of CANopen: If TESTTIM is set but TESTVAL is not set, then the CANopen case is assumed. In that case, it is checked if the time within the TESTWIN is longer than TESTTIM. If so, then the position has been reached. Otherwise, the position has not been reached.

Portability: TESTWIN handling is modified starting with version 6.6.53

Parameter Group: AXI Predefined I/Os

Unit: qc

Value Range: 0 … 10000

0 = Off

! ! ! TESTWIN must always be less than TESTVAL.

Init Value: 0

VELMAXQC - 9

Content Maximum velocity (Read only)

Description VELMAXQC determines the *Maximum Velocity*. All statements made with the assistance of VELRES (22) refer to this speed. The sample time [st] is 1 ms.

VELMAXQC is a read only parameter which cannot be changed. It is automatically calculated from VELMAX (1) which the user enters in RPM. These internal values are decisive for the permitted value range. However, in practice these limit values are of no importance since they exceed the encoder input frequency determined by the hardware by far.

Parameter Group AXB (not displayed in axis parameter fields)

Unit qc/st * 1/65536

Value Range 1 … MLONG (Read only)

ACCMAXQC - 10

Content Maximum acceleration (Read only)

Description ACCMAXQC determines the maximum acceleration. This is the amount of time the drive needs with a connected load to achieve the maximum rotation. All other statements made with the assistance of the scaling VELRES (22) refer to this acceleration.

This read only parameter is automatically calculated from RAMPMIN (31), which the user enters in ms.

! ! ! If a starting time is entered which is too short, which under the existing mechanical conditions causes non-achievable acceleration, usually a position error will occur.

Parameter Group AXB (not displayed in axis parameter fields)

Unit qc/st² * 1/65536

Value Range 1 … MLONG (Read only)

KPROP - 11

Content	Proportional value for PID position control loop.
Description	The ***Proportional factor*** KPROP of the ***position control*** loop indicates the linear correction factor with which the deviation between the current set and actual position is evaluated and a corresponding correction of the motor speed is made. Rule of Thumb: KPROP greater = Drive will become 'stiffer' KPROP too high = Tendency to overswing Typically the range of KPROP:KDER varies in between 4:1 ... 1: 4, if a power stage is use, which is configured for speed control.
! ! !	The value of this parameter has to be adjusted, if the parameter ENCODER is modified after KPROP was tuned. Examples: If the value of ENCODER is doubled, then KPROP has to be halved. If the value of ENCODER is halved, then KPROP has to be doubled. If a MACS control unit with integrated power amplifiers is in use and the control structure (= parameter AMPMODE) is modified, it might also be necessary to adapt the KPROP value.
Parameter Group	AXR Position Regulation
Unit	–
Cross Reference	KDER, KINT, KILIM, KILIMTIME, BANDWIDTH, ENCODER, AMPMODE
Value Range	1 … 65000
Init Value	30

KDER - 12

Content	Derivative Factor for PID position control loop.
Description	The ***Derivative factor*** KDER of the ***position control*** loop is the correction factor with which the changing speed of a motor position error is evaluated. The derivative factor works against the tendency to overswing due to a high P-share and 'dampens' the system. However, if the derivative factor selected is too large this will lead to a 'nervous' drive and a humming motor noise. Typically the range of KPROP:KDER varies in between 4:1 … 1: 4, if a power stage is use, which is configured for speed control.
! ! !	The value of this parameter has to be adjusted, if the parameters ENCODER or TIMER are modified after KDER was tuned. Examples: If the value of ENCODER is doubled, then KDER has to be halved. If the value of ENCODER is halved, then KDER has to be doubled. If the value of TIMER is doubled, then KDER has to be halved. If the value of TIMER is halved, then KDER has to be doubled. If a MACS control unit with integrated power amplifiers is in use and the control structure (= parameter AMPMODE) is modified, it might also be necessary to adapt the KDER value.
Parameter Group	AXR Position Regulation
Cross Reference	KPROP, KINT, KILIM, KILIMTIME, BANDWIDTH, ENCODER, TIMER, AMPMODE
Unit	–
Value Range	0 … 65000
Init Value	0

KINT - 13

Content	Integral value for PID position control loop.
Description	The ***Integral Factor*** KINT of the ***position control*** loop is the weighting factor, with which the sum of all motor position errors is evaluated. The integral factor of the PID filter causes a corresponding corrective motor torque which increases over time. Through the integral share a static position error is reduced to zero, even if a constant load is affecting the motor. However, an integral factor which is too large leads to a 'nervous' drive. The maximum value of the accumulated integral sum can be limited by the parameter KILIM. The activation of the integral factor can be influenced by the parameter KILIMTIME. It is possible to activate the integral factor all the time, or just to activate it during a movement or at standstill. This might be helpful for special positioning or synchronization requirements.
! ! !	The value of this parameter has to be adjusted, if the parameters ENCODER or TIMER are modified after KINT was tuned. Examples: If the value of ENCODER is doubled, then KINT has to be halved. If the value of ENCODER is halved, then KINT has to be doubled. If the value of TIMER is doubled, then KINT has doubled too. If the value of TIMER is doubled, then KINT has to be halved too. If a MACS control unit with integrated power amplifiers is in use and the control structure (= parameter AMPMODE) is modified, it might also be necessary to adapt the KINT value.
Parameter Group	AXR Position Regulation
Cross Reference	KPROP, KDER, KILIM, KILIMTIME, BANDWIDTH, AMPMODE
Unit	–
Value Range	0 … 65000
Init Value	0

TIMER - 14

Content Sampling time of the PID position regulation and for sending the returned value to the drive.

Description The TIMER parameter determines the sampling time of the position control algorithm.

The default value of 1ms has to be increased in case of …

- … very low pulse frequency, such as from 1 to 2 qc per sampling time. You need at least 10 to 20 qc per sampling time.
- … for very slow systems with a long dead time. If 1 ms is used here for control, large motors will vibrate.

The value should not be set higher than 1000 (= 1 s). This would be a very slow control.

This parameter influences also how often a new command value is send to the amplifier. For CAN bus drives this has to be adjusted in accordance to the drive features. Some drives do not accept new reference values every ms. In many cases the maximum update rate of frequency converters is just in between 2 ... 4 ms.

Up to firmware version 6.5.16 the parameter TIMER also determined the period of time in between SYNC telegrams requesting the status and encoder information of the drive. For multi-axis versions older 6.5.16, it has to be kept in mind, that the shortest definition of the TIMER defines how often a SYNC telegram is generated, on which all amplifiers on the bus have to react. This means, that even if there was even just one drive with low CAN bus performance connected, the TIMER of all CAN bus drives had to be set to the value of the low performance one.

Starting with firmware version 6.5.16 there is the new global parameter CANSYNCTIMER, which determines the period of time for sending cyclic SYNC telegrams. The value of the axis specific parameter TIMER has no meaning anymore for the periodicity of the SYNC telegram, but still for the rate of sending the set value via bus. The PDO set value telegram rate can be chosen different for each amplifier or frequency converter, which is controlled via CAN bus. The rate of the feedback information of the power stage device is defined by CANSYNCTIMER. If the encoder information is also provided through the bus, it has to be made sure, that new encoder information is present for each cycle of the position control loop. This means that the CANSYNCTIMER has to be set to a value corresponding to the TIMER setting of the axis and the performance of the bus-coupled power stage devices.

! ! ! Note that any modification of this parameter has a direct effect on the PID position regulation loop parameters KDER and KINT.
If the value of TIMER is doubled, then KDER has to be halved.
If the value of TIMER is halved, then KDER has to be doubled.
If the value of TIMER is doubled, then KINT has to be doubled too.
If the value of TIMER is halved, then KINT has to be halved too.

But: The current setting of KPROP must not be adjusted, if the setting of TIMER is modified. There is no dependency in between these two parameters.

Portability Starting with version 6.5.16 the setting or TIMER has no meaning anymore for the CAN SYNC!

Parameter Group AXR Position Regulation

Cross Reference KPROP, KDER, KINT, DRIVETYPE, CANSYNCTIMER

Unit ms

Value Range 1 … 1000

Practically the value should not be set higher than 1000 (= 1 s). This would already be a very slow control.

Init Value 1

POSERR - 15

Content	Maximum Tolerated Position Error
Description	The ***Maximum Tolerated Position Error*** POSERR defines the tolerance allowed between the current actual position and the calculated command position. If the value defined with POSERR is exceeded then the position control is turned off on all axes and a position error is turned off and a position error is triggered.
! ! !	The ***Maximum Tolerated Position Error*** does not affect the positioning accuracy, but merely determines how precisely the theoretically calculated path of motion must be followed, without an error being triggered.
! ! !	For safety reasons the ***Maximum Tolerated Position Error*** selected should not be too large since this could be dangerous for both the machine and its operator. On the other hand, if the values for the ***Position Error*** are too small this could result in frequent errors. As a guideline, it is wise to set the quadruple of Encoder counts per revolution. This corresponds to one encoder rotation.
Parameter Group	AXR Position Regulation
Unit	qc
Value Range	1 … MLONG
Init Value	1000

SYNCOFFTIME - 16

Content Compensation velocity of an offset (1. synchronize; 2. new offset)

Description The offset filter SYNCOFFTIME also influences the way, how a new SYNCPOSOFFS value is handled. The offset which has to be realized will be done step by step. The step which is realized every sample time (ms) is calculated as follows:

step size = SYNCMPULSM / SYNCOFFTIME (integer part).

So it will take SYNCOFFTIME to realize an offset of SYNCMPULSM. SYNCOFFTIME also has influence on the marker start correction and on the correction of marker error (see SYNCMFTIME).

SYNCOFFTIME defines the time which should be used to compensate one marker distance.

Starting with the firmware version 6.6.49 not only the offset over the given time is distributed, but also a trapezoidal movement is calculated which is superimposed onto the normal synchronizing movement.

This is used to calculate limiting accelerations and velocity with the assumption that normally 20 % for acceleration, 60 % with constant velocity, and another 20 % of the time to decelerate will be used. Based on that interpretation, it is calculated what acceleration and deceleration ramps should be used and how the maximum velocity needs to be limited to reach that.

This new distribution is not only used for offsets, but also for start corrections (first marker correction), normal marker corrections and accumulated errors caused by REVERS setting.

Portability Standard command is available starting with control version 6.2.35, VLT > 5.04; trapezoidal movement with firmware 6.6.49.

Parameter Group AXS Synchronization

Unit ms

Value Range 0 … MLONG

Init Value 0

SYNCMFPAR - 17

Content Marker filter configuration

Description This parameter SYNCMFPAR is used to influence the behavior of marker filtering, see SYNCMFTIME (18) and SYNCSFTIME (106).

If SYNCMFTIME is set, then the MMarkerDist is calculated a little differently. If the time set results in a number of markers which is smaller than 100, then at least 100 are taken to calculate the average marker distance (Filter). This only affects the calculation of the marker distance. The other filter calculations (deviation) are still done with the number of markers calculated by SYNCMFTIME and actual velocity of master.

See also Marker Correction illustration in chapter “Technical Reference”.

Portability Standard command is available starting with control version 6.2.35,
(in previous APOSS Versions this parameter could be set by ORGACC),
changed calculation of marker distance (if markers < 100) with version 6.7.16.

Parameter Group AXS Synchronization

Unit –

Value Range 0 or 1 or 2 or 4 or 16 or 64

The following values are bit valences and can be combined with each other:

- 0 = Normal filter function, see ***Filter Time for Marker Correction*** SYNCMFTIME
- 1 = Instead of the dynamic marker filter constant a constant value of SYNCMFTIME / 300 is used.
- 2 = Gear correction is not executed.
- 4 = The ***Filter Time for Marker Correction*** SYNCOFFTIME (16) is used instead of the ***Offset Filter Time*** to calculate the time constant for the correction value filter (G_Korrektur)
- 16 = Instead of calculation of the filtered marker distance and deviation, only the correction value is smoothed by a PT filter. Time constant for that filter on bit valence 4 basis (see above).
- 64 = Marker averaging and marker check are also executed when SYNCM is inactive.

For further descriptions of the selection see SYNCMFTIME (18).

Init Value 0

SYNCMFTIME - 18

Content: Defines the filter time for marker correction

Application Example: Newspaper manufacturing needed this sort of filtering to synchronizing a chain to the newspaper stream coming from a printing machine. Because the newspaper stream is not quite constant, the problem is that if synchronized without filter the movements of the chain are very hard and dynamic. With all other sort of filters the system starts swinging in sinusoidal waves.

When using this complex filter the synchronizing works very well and solves the problem.

Description: SYNCMFTIME is given in ms and is used as follows:

! ! ! Master velocity filtering SYNCVFTIME (65) is given in 1/1000 ms for a better resolution, but the marker filtering (SYNCMFTIME) is given in units of 1 ms.

Example:

```
SET SYNCVFTIME –50000
SET SYNCMFTIME 2000
```

This means, that the master velocity is filtered over a period of 50 ms. A marker error is corrected within a period of 2000 ms.

The actual filtered marker distance can be read out with SYSVAR 4238 indices if this filter is activated by setting SYNCMFTIME. To achieve the filter value, we internally calculate how much markers will pass by, if it runs at maximum allowed speed over a period of SYNCMFTIME.

The SYNCMFTIME and the parameters SYNCOFFTIME (16) and SYNCMFPAR (17) are used to influence the behavior of Marker Filtering (see below).

Filtering is handled as follows:

Calculation of Marker Filter only if SYNCMFTIME > 0

If SYNCMFPAR = 1

Every time when a real master marker is found, the Marker Filter (G_MarkerFilter) constant can be calculated as SYNCMFTIME/300.

if SYNCMFPAR = 0

Every time when a real master marker is found, the Marker Filter constant can be calculated as the:

Filtered Old Master Velocity * SYNCMFTIME / (SYNCMPULSM * 3)

which means, that the Marker Filter constant is used as time constant for filtering. Then the time which is needed to get an output corresponding to a steady input should be nearly SYNCMFTIME.

The full description of the formula and its mode of action is:

```
IF (SYNCMFPAR & 1) THEN
    MarkerFilter = max(SYNCMFTIME / 300, 1)
ELSE
    MarkerFilter = Oldmfiltvel * SYNCMFTIME/(SYNCMPULSM *3)
```

which means, that if MarkerFilter is used in a PT element as Tau, then the time which is needed to get an output corresponding to a steady input should be nearly SYNCMFTIME. The calculation is necessary, because the filter (PT) is executed every marker and not every ms.

ENDIF

This Marker Filter (MarkerFilter) is now used when filtering the marker distance (LastMMDist) with a PT element. The result is stored in the Filtered Marker Distance (MMarkerDist).

This result is then used to calculate the necessary gear correction (MmarkCorr) as follows:

MMarkCorr = (SYNCMPULSM – MMarkerDist) / MMarkerDist

in other words:

Gear correction = (SYNCMPULSM – Filtered marker distance) / Filtered marker distance

Filtering master velocity and gear correction

Every sample time when the filtered master_velocity (difference of actual and last master position) is calculated

IF (SYNCVFTIME < 0)

then Filtered Old Master Velocity (Oldmfiltvel) is calculated by executing a PT element (Tau = SYNCVFTIME / 1000)

ELSE

Oldmfiltvel = master velocity
Else the Filtered Old master velocity is set equal to the actual master velocity.

ENDIF

In case where SYNCMFTIME > 0 and SYNCMFPAR = 2 the gear correction is made by taking the current gear ratio and add the master velocity multiplied by the Gear correction:

```
Tmpdelta = Oldmfiltvel
IF (SYNCMFTIME > 0 and !(SYNCMFPAR & 2)) THEN
    gear correction is done as follows:
    Tmpdelta = Tmpdelta + master_velocity * MMarkCorr
ENDIF
```

Start correction only if SYNCMFTIME > 0

Start correction is the correction which must be realizes after start condition is fulfilled. That means either the first two markers (SYNCMSTART 1,6) were observed or the master velocity is reached and the first two markers (SYNCMSTART 2,3,4,5) has been observed. This start correction is split in such a way, that it will be eliminated after SYNCOFFTIME. (Actually it is divided by the amount of markers, which will be passed in SYNCOFFTIME at the actual master velocity and that value is added to the normal marker correction, see below, the result is stored in PFG_STARTKORRVAL)

If SYNCOFFTIME == 0, Start correction will be eliminated at once, which means that STARTKORRVAL will contain the total start correction or in other words the correction will be done between two markers.

Marker correction SYNCMFTIME > 0

Old version (6.2.20 – 6.2.33) (see online help)

Correction of marker error will be done as follows:

First the remaining start correction (PFG_STARTKORRREST) is subtracted from the marker error (PFG_KORRUNFILT). Then a PT filter for the remaining error (Tau = MarkerFilter) is used and the PFG_STARTKORRVAL is added to the result of PT. The result is stored in PFG_KORREKTUR as usual. Then this correction is spread over one marker distance. This is done by dividing the correction by the number of samples which will be necessary to pass one marker distance at actual master speed. This value is stored in PFG_KORRVAL and will be used every sample time to correct the calculated slave position.

Following SYNCMFPAR modification is included:

SYNCMFPAR & 4 → *Offset Filter is used instead of Marker Filter for correction PT*

SYNCMFPAR & 16 → *PT Filter is not used. Instead we use either MDEFDIST / (no of markers within SYNCOFFTIME) or KORREKTUR, whichever is smaller*

Since Version (6.2.35)

First the remaining start correction (PFG_STARTKORRREST) is subtracted from the marker error (PFG_KORRUNFILT). Then correction filtering time (PFG_KORRTAU) corresponding to the

SYNCOFFTIME (master velocity dependant see start correction number of markers) is set.

Now the sum of all marker distance errors (PFG_INTMMERROR) is used for a marker filter (PT (Tau = PFG_MARKERFILTER)) to calculate the filtered sum (PFG_MMARKERR). Then the filtered error sum is subtracted from the unfiltered one. This result is then used to correct the marker correction.

This corrected correction is then given into the correction filter PT (Tau = PFG_KORRTAU). The result of this correction (PT) filter is then stored (plus start correction part if necessary) into PFG_KORREKTUR.

Then we try to spread this correction over one marker distance. This is done by dividing the correction by the number of samples which will be necessary to pass one marker distance at actual master speed. This value is stored in PFG_KORRVAL and will be used every sample time to correct the calculated slave position.

Following SYNCMFPAR settings modify behavior:

SYNCMFPAR & 4 → PFG_KORRTAU is set to PFG_MARKERFILTER instead of using SYNCOFFTIME, which means correction time is used instead of SYNCOFFTIME.

SYNCMFPAR & 16 → No correction concerning the error of marker distances will be done.

Marker correction SYNCMFTIME == 0

In that case KORRUNFILT and KORREKTUR contain the same value, which also includes start correction in the first case (SYNCMFTIME > 0). This value is now handled as follows:

```
IF (SYNCVFTIME < 0) THEN
    KORRVAL = KORREKTUR / max((-SYNCVFTIME / 100) , 1)
ELSE
    KORRVAL = 0
ENDIF
```

In the first case where marker correction > 0, the correction is spread over a time of (-SYNCVFTIME / 100) ms.

In the second case the correction is added to the demand position at once.

The reaction of course is limited by the actual acceleration and deceleration in every case.

Marker distance calculation

If SYNCMFTIME is set, then the MMarkerDist is calculated a little differently. If the time set results in a number of markers which is smaller than 100, then at least 100 are taken to calculate the average marker distance (Filter). This only affects the calculation of the marker distance. The other filter calculations (deviation) are still done with the number of markers calculated by SYNCMFTIME and actual velocity of master.

See also Marker Correction illustration in chapter “Technical Reference”.

Portability Standard command with software version 6.2.20 onwards, from 6.2.35 onwards modification as describes above.

If SYNCMFTIME is 0, the behavior is same as up to control version 6.1.14, that means the filter time for marker correction depends on the parameter value of SYNCVFTIME (65).

Calculation of marker distance (if markers < 100) is changed starting with version 6.7.16.

Parameter Group AXS Synchronization

Unit ms

Value Range –MLONG … MLONG

0 = If SYNCVFTIME (65) is negative, the marker correction is spread by SYNCVFTIME /100

Init Value 0

SWNEGLIMACT - 19

Content Negative software end limit active

Description By setting this parameter to 1 the control is informed that the ***Negative Software End Limit*** should be monitored. Then it is checked whether the target position is located outside of the permissible movement range during every movement. In this case an error message is issued and the drive control is switched off.

In the positioning mode this means that the corresponding positioning process is not started and the error can be cleared with the ERRCLR command.

In synchronization and speed mode an error can only be recognized after the limit has been exceeded, thus when the error message is issued the drive is already outside of the permissible area of movement. It is possible to clear a software limit error and then drive in the opposite direction. If you try again to move in the wrong direction, then the error 98 F_LIMIT_VIOLATION occurs.

In this case, it is necessary to move the drive by hand back to the admissible area and to erase the error, or in the menu **Controller → Parameters → Edit** to temporarily turn off the corresponding ***Software End Limit*** and then delete the error.

Portability The behavior in case of limit switches is improved starting with firmware 6.6.60.

Parameter Group AXI Predefined I/Os

Unit –

Value Range 0 … 1

0 = inactive
1 = active

Init Value 0

SWPOSLIMACT - 20

Content Positive software end limit active

Description By setting this parameter (to 1) the control is informed that the ***Positive Software End Limit*** is to be monitored. In this case it is checked whether the target position is located outside of the permissible movement range during every movement. If necessary an error message is issued and the drive control is switched off.

In the positioning mode this means that the corresponding positioning process is not started and the error can be cleared with the ERRCLR command.

In the synchronization and speed mode an error can only be recognized after the limit has been exceeded, thus when the error message is issued the drive is already outside of the permissible area of movement.

It is possible to clear a software limit error and then drive in the opposite direction. If you try again to move in the wrong direction, then the error 98 F_LIMIT_VIOLATION occurs. In this case, it is necessary to move the drive by hand back to the admissible area and to erase the error, or in the menu **Controller → Parameters → Edit** to temporarily turn off the corresponding ***Software End Limit*** and then delete the error.

Portability Improved behavior in case of limit switches is available starting with firmware 6.6.60.

Parameter Group AXI Predefined I/Os

Unit –

Value Range 0 … 1

0 = inactive
1 = active

Init Value 0

KILIM - 21

Content	Limit value for the integral sum of the PID position control loop.
Description	This parameter limits the integral sum of the PID ***position control*** loop in order to avoid instability and PID wind-up in case of feedback error. A value of 0 means, that the integral part of the PID control is disabled. The integral part of the PID control can be automatically switched on and off depending on the motor state by the parameter KILIMTIME. This might be helpful for special positioning or synchronization requirements.
Parameter Group	AXR Position Regulation
Cross Reference	KPROP, KDER, KINT, KILIMTIME
Unit	–
Value Range	0 … 65000
Default	1000

VELRES - 22

Content	Velocity resolution
Description	The ***Velocity Resolution*** VELRES defines a relative size for the velocity values of the motion commands and parameters. The information concerning speed and acceleration can then be made in whole numbers in relation to this scaling. The value 100 means that the information in the commands are related to 100, thus in percent.
Parameter Group	AXB Basic Settings
Unit	–
Value Range	1 … 10000
Init Value	100

POSFACT_Z - 23

Content Numerator user factor
or in CAM mode conversion of the units qc into UU.

Description The ***User factor*** defines a conversion factor to the nearest thousandth of user units (UU) in quadcounts [qc]. All path information in motion commands are made in user units and are converted to quad-counts internally. By choosing these scaling units correspondingly it is possible to work with any technical measurement unit (for example mm).

This factor is a fraction which consists of a numerator and denominator.

POSFACT_Z / POSFACT_N = 1 User unit UU

Scaling determines how many quad-counts make up a user unit. For example, if it is 50375/1000, then one UU corresponds to exactly 50.375 qc.

In CAM-Mode, the parameter is used to fix the unit for the slave drive so that it is possible to work with meaningful units in the CAM-Editor. See example 2.

$$\frac{\text{Gearing Factor} * \text{Encoder Resolution} * 4}{\text{Scaling Factor}}\ \text{qc} = 1\ \text{UU}$$

provided that:

$$\text{Gearing Factor} = \frac{\text{Motor Revolutions}}{\text{Revolutions on Output}}$$

Encoder = Incremental encoder (in the case of absolute encoders, the multiplier 4 is omitted)

Scaling factor = Number of user units UU [qc] that correspond to one revolution at the drive

In addition, it is possible to compress or expand the curves with this factor without having to define new curves each time. The use of numerator and denominator for the gearing factor leads to a very precise result since transmission ratios can be represented as a fraction in virtually all cases.

Portability The factor is no longer limited to small values starting with version 6.4.43MCO 5.00, see also User Units in chapter "Programming with APOSS".

Parameter Group AXE Encoder

Unit –

Value Range 1 … MLONG/max. position (UU)

The limit depends on the maximum target position, whereby following context is valid:
max. position (UU) * POSFACT_Z < MLONG
where by POSFACT_Z = User factor * 1000

Example: POSA max. position (UU)

Init Value 1000

Example 1 Shaft or spindle

25 motor revolutions result in 1 spindle revolution; gearing factor = 25/1

Encoder resolution (incremental encoder) = 500

Spindle gradient = 1 revolution of the spindle = 5 mm

Scaling factor in case of working with 1/10 mm resolution = 5 * 10 = 50

$$\frac{25/1 * 500 * 4}{50}\ \text{qc} = \frac{25 * 10 * 4}{1}\ \text{qc} = \frac{1000}{1}\ \text{qc} = 1\ \text{UU}$$

Numerator user factor [23] = 1000
Denominator user factor [26] = 1

Example 2 Cylinder:

Gear factor = 5/1

Encoder resolution (incremental encoder) = 500

One revolution of the cylinder is 360 degrees. It is requested to work with a resolution of 1/10 degrees. This means that one revolution of the cylinder is divided into 3600 units.
Scaling factor = 3600

$$\frac{5/1 * 500 * 4}{3600}\ qc = \frac{5 * 500 * 4}{3600}\ qc = \frac{25}{9}\ qc = 1\ UU$$

$$1\ UU = \frac{POSFACT\_Z\ (23)}{POSFACT\_N\ (26)}$$

Numerator user factor [23] = 25
Denominator user factor [26] = 9

TESTTIM - 24

Content	Measuring time in target window
Description	Once the target window has been reached the position is read twice and compared with the parameter TESTVAL (25). If the result is less than TESTVAL, then the position has been reached, otherwise a new reading is taken. TESTTIM indicates the time interval between the measurements.
Parameter Group	AXI Predefined I/Os
Unit	ms
Value Range	0 … 10
! ! !	The reason for the limitation to 10 ms is to be seen, that the function 'diffval' waits effective and in this time the monitoring of the limit switch and the position error is not active. For this reason the value should be not too long.
Init Value	0

TESTVAL - 25

Content	Limit value for readings in target window
Description	Once the target window has been reached the position is read twice with an interval of TESTTIM (24) and the interval is compared with the ***Target Window Limit Value*** TESTVAL. The result determines whether the position is viewed as having been reached or not.
! ! !	For longer time intervals it must be taken into consideration that reaching this target position will be delayed by this amount of time in any case.
Parameter Group	AXI Predefined I/Os
Unit	qc
Value Range	1 … 10000
Init Value	1

POSFACT_N - 26

Content Denominator user factor
or in CAM mode conversion of the units qc in UU.

Description All path information in motion commands is made in user units and converted to quad-counts internally. By choosing these scaling units correspondingly it is possible to work with any technical measurement unit (for example mm).

This factor is a fraction which consists of a numerator and denominator.
1 User Unit = POSFACT_Z / POSFACT_N

Scaling determines how many quad-counts make up a user unit.

In CAM mode, the parameter is used to determine the units for the slave drive so that it is possible to work with meaningful units in the CAM-Editor. See prerequisites of the formula and example under POSFACT_Z (23)

$$\frac{\text{Gearing Factor} * \text{Encoder Resolution} * 4}{\text{Scaling Factor}}\ \text{qc} = 1\ \text{UU}$$

In addition, it is possible to compress or expand the curves with this factor without having to define new curves each time. The use of numerator and denominator for the gearing factor leads to a very precise result since transmission ratios can be represented as a fraction in virtually all cases.

Portability The factor is no longer limited to small values starting with version 6.4.43, see also Scaling in chapter “Programming with APOSS”.

Parameter Group AXE Encoder

Unit –

Value Range 1 … MLONG

Init Value 1000

Sample See POSFACT_Z (23)

ENCODERTYPE - 27

Content Defines the signal source of the feedback encoder for the axis control.

Description ENCODERTYPE defines the signal source of the feedback encoder (typically shaft encoder) for the axis control. The actual value information can be captured via the following "input channels" and has to be defined correspondingly:

- Local encoder input of the control
- CANopen encoder via process data word (PDO)
- CANopen amplifier via process data word (PDO)
- Encoder type via process data word (PDO) on the EtherCAT bus

Incremental and absolute encoder data can be used as encoder input information.

Peculiarities with absolute encoders Some types of linear absolute encoders tend to report sporadically wrong position information (typically a jump of the position value). These errors or encoder malfunction can be detected with the encoder types 3 and 4, if the leap is larger than half encoder resolution. The correction is made by means of an artificial position value which is calculated from the last velocity. If the error continues for more than 100 read-outs (> 100 ms), there will be no further correction which will lead then indeed to a tolerated position error exceeded. Total number of errors will be saved in an internal variable which can be read out via SYSVAR[16].

! ! ! The commands DEFORIGIN, HOME, INDEX, IPOS, and WAITNDX cannot be used with absolute encoders.

Device types If encoder information is captured via the bus, the calculation of the parameter value is done on the basis of the device types as defined in parameter DRIVETYPE with following formula:

SET ENCODERTYPE nn
where nn = guard * (busoffset + device type * 1000 + id)

The encoder types with device type 1 to 49 are handled like incremental encoders, where the relative change of the bus transferred position value (PDO) will be added to last position.

The encoder types with device type 50 are handled like absolute encoders, where overflow will happen by the encoder (see also CAN OPEN encoder definition).

Another encoder type gets its actual position and index position from a PDO on the EtherCAT bus. Example:

```
SET ENCODERTYPE 251000   // uses PDO[0] as EtherCAT encoder
// 200000  = EtherCAT bus
// 51(000) = EtherCAT encoder
// 100er   = PDO no (e.g. 200 = pdo 2)
// 1er  = offset in PDO in longs (0 = use 1. long in PDO)
```

The first long at that place will be used for the position, the second for the index position

e.g. SET ENCODERTYPE 251105

```
// Use PDO1 long with offset 5 (PCD[6]) for position,
// and long with offset 6 (PCD[7]) for index position
```

The implementation is similar to the CAN encoders (in can_opn.c).

ENCODERTYPE 9 In correlation with ENCODERTYPE 9 (slave simulation) the meaning of SYNCMTYPS which describes the type of marker input has been changed. In older firmware there was no simulation of markers or only with SYNCMTYPS 9. This is not true any more.

The definition is as follows.

SYNCMTYPS now supports the following values (if ENCODERTYPE == 9)

0 – virtual markers are produced with a distance of ENCODERFREQ.

n2 – At positive edge of input n the actual position is taken as marker position

n3 – At negative edge of input n the actual position is taken as marker position

The accuracy is of course limited to 1 ms which is the internal update rate.

Handling of the automatically generated CAN objects

The necessary CAN objects (PDOs, GUARD-, SYNC-object) for the information transfer of the actual position via the bus are defined automatically, and the bus node is also initialized (NMT0). This object generation, which is done in background, corresponds basically to the command CANINI. In contrast to the application object generation using CANINI, the automatic generated objects by ENCODERTYPE remain untouched even when an updated CANINI definition is done by the application later on. Even a CANDEL –1 command does not delete or stop the objects, which are automatically established by the ENCODERTYPE setting. The automatic generated objects can just be deleted or reconfigured by a new differing SET ENCODERTYPE. Also in the case of program abort (by means of **Esc**) the objects remain existent up to the next program start. The start of a program deletes any formerly defined objects and initializes new objects according to the setting of the permanent parameters or at run-time according to any temporary parameter definition inside the program code.

Guarding

When Guarding is configured (i.e. ENCODERTYPE > 0), the bus node is monitored and a missing feedback to the GUARD object (e.g. because a node malfunction) triggers the error 88. It is up to the user to take care that the parameter REVERS (63) is set properly (corresponding to the bus-ID), the bus node is configured (baud rate, bus-ID), the bus device is connected, powered up, and ready for communication at the time when the parameter ENCODERTYPE will be defined.

! ! !

If the bus device fails in an ENCODERTYPE > 0 configuration, the guarding must be switched off by the application. This can be done within the error handling routine by setting the negative ENCODERTYPE value, until the failure of the device is fixed. Otherwise the identical error message comes up again and again after every deletion of the control error status 88 (by means of ERRCLR).

PDO-mapping for CANopen drives

If the encoder information is provided by a CANopen amplifier, the so-called PDO-mapping must be done by the application before setting the parameter DRIVETYPE and ENCODERTYPE. PDO-mapping means the definition of which device objects are linked into the transferred PDOs. The choice of most suitable objects depends on the CANopen amplifier. You can get sample source code suitable to the following mentioned device types from the zub machine control AG. Please tell us the used amplifier for that via e-mail (info@zub.ch).

SYNC period

Please note, that the rate of the SYNC object was adapted to the parameter TIMER of the fastest axis until firmware version 6.5.16. For up-to-date firmware versions (6.5.16 or higher) the global parameter CANSYNCTIMER specifies the periodicity of sending the SYNC telegram now. This parameter has to be set to a proper value, which fits to the slowest CAN slave device in use.

Real-time acquisition of position values based on marker signals

It is recommended to use the local encoder inputs of the control for marker synchronizations or for positioning operations, which depend on real time acquisition of position values based on marker signals (see command IPOS, parameter SYNCMTYPS). Only in this case the (hardware) latching of the position values with the maximum accuracy of 1 qc is possible. In case of encoder information captured by the bus the accuracy of the marker acquisition decreases depending on the cycle time of the bus (parameter TIMER, CANSYNCTIMER) and the speed of the motor.

Portability M1 and other new controller.
Master simulation via CAN bus is available starting with control version 6.1.14;
encoder information via PDO on the EtherCAT with version 6.5.01;
absolute passive encoder (SSI) with version 6.7.05 (MACS4);
changed meaning of SYNCMTYPS concerning slave simulation with version 6.7.19.

Parameter Group AXE Encoder

Cross References MENCODERTYPE, DRIVETYPE, REVERS, IPOS, CANINI, TIMER, CANSYNCTIMER

Unit –

Value Range –MLONG … MLONG

0 = Incremental encoder

1 = Absolute encoder, standard ca. 262 kHz

2 = Absolute encoder, ca. 105 kHz

3 = Absolute encoder without overflow (linear) but with error correction, ca. 262 kHz

4 = Absolute encoder without overflow (linear) but with error correction, ca. 105 kHz

5 = Sine / cosine encoder

9 = Simulated encoder, i.e. reference position will be copied into the actual value (e.g. for program testing without connected motor)

10 = Incremental encoder at local encoder terminal 1

20 = Incremental encoder at local encoder terminal 2

30 = Incremental encoder at local encoder terminal 3

> 1000 = nn = Encoder information captured via CAN bus

nn = guard * (busoffset + device type * 1000 + id)

guard = –1, +1 (without / with Guarding)

busoffset = 100000 , 0 (Slave-Bus, Master-Bus)

device type (see also DRIVETYPE) =

0 = Standard drive (analog output)
1 = CAN drive of type Lenze (standard)
2 = CANopen servo amplifier based on DS402
3 = CANopen servo amplifier maxon EPOS
4 = CANopen servo amplifier zub DSA
5 = CAN drive of type Lenze (set point value limit)
10 = Danfoss Drives
50 = CANopen encoder
91 .. 94 = Analog outputs to a CANopen I/O module

id = 1…127 = Bus-ID, with device type 1…50 and data exchange via PDO 2

id = 257…511 = Bus ID 1…127 with device type 1, but data exchange via PDO 3

> 100000 = nn = Encoder information captured via EtherCAT

nn = busoffset + device type * 1000 + PDO no. + long offset

device type =

51 = Encoder information via PDO on the EtherCAT

Usage example:

```
SET ENCODERTYPE 251100   // uses pdo[1] as EtherCAT encoder
// 200000  = EtherCAT bus
// 51(000) = EtherCAT encoder
// 100er   = pdo nr (e.g. 200 = pdo 2)
//   1er   = offset in pdo in longs (0 = use 1. long in pdo)
```

The first long at that place will be used for the position, the second for the index position

e.g. SET ENCODERTYPE 251105

// Use PDO1 long with offset 5 (PDO[6]) for position, and long with offset 6 (PDO[7]) for index position

Init Value 0

POSDRCT - 28

Content Rotational direction

Description Normally a positive reference value brings about a positive change of the position. If this is not the case the reference value can be reversed internally by the parameter values –2 …2.

In some applications it is desirable to reverse the direction of rotation in user units. This is achieved by the negative sign in POSDRCT.

The direction of the synchronization (relation to master) can be turned by negative SYNCFACTM.

In the case of synchronization in CAM-Mode, you can determine a positive direction of rotation for the slave with POSDRCT. This is a prerequisite for the CAM functionality.

Internally, the encoder information is always used which comes in from the FPGA.

Starting with compiler 6.5.36 the feedback for the velocity control loop can be set separately from the feedback for the position control loop with the parameters AMPENCNO, AMPENCRES, and AMPENCRPM. Therefore the parameter POSDRCT can also configure the kind of the feedback evaluation (i.e. normal or inverted) separately for the position and the velocity control loop:

The decade (= 10er digit) configures the kind of feedback evaluation (normal or inverted) in use by the feedback source of the velocity control loop.

If single-digit configuration values (like for previous firmware versions) are in use, both signal sources will be evaluated in the same way.

All configuration values can be combined with a negative sign, which means that the sign of the user units will be inverted.

Portability Only M1 and other new controller;
starting with compiler 6.5.36: The kind of feedback evaluation (normal or inverted) can be configured separately for the velocity and position control loop.

Parameter Group AXE Encoder

Unit –

Value Range –22 … 22

1 = No change, i.e. positive reference values produce positive encoder values.:
Velocity and position feedback, as well as user units without any modification.

–1 = User units reserved
The sign of the user unit is reversed. Thus, positive reference values produce positive encoder values which are indicated as negative values, however. This applies to all outputs (APOS, CPOS, …) all user inputs (POSA, POSR, …) and all synchronization factors, as well as the velocities (CVEL, HOME_VEL).
Velocity and position feedback without any modification,
User units provided with an inverse sign.

2 = Reference reversed
The sign of the reference value is reversed internally (plus becomes minus and vice versa). This is equal to a reversal of the motor leads, or a transposition of the A and B tracks on the encoder.
Velocity and position feedback use an inverse sign,
User units without any modification.

–2 = UU and Reverence reversed
Same as 2, i.e. the sign of the reference value is reversed internally; in addition, the sign of the user unit is negated as in –1.
Velocity and position feedback use an inverse sign,
user units provided with an inverse sign.

12 = Position feedback uses an inverse sign,
Velocity feedback without any modification,
User units without any modification

21 = Position feedback without any modification,
Velocity feedback uses an inverse sign,
User units without any modification

22 = like POSDRCT 2, i.e.
Velocity and position feedback use an inverse sign,
User units without any modification

Each of the two digit POSDRCT configurations can also be in use with a negative sign to inverse the sign of the user units.

Init Value 1

PROFTIME - 29

Content Scan time for profile generator

Description The Parameter gives the possibility to set the sample time for the profile generator, which is independent of the sample time for the PID controller.

For demanding control tasks in the background (SYNCP, SYNCM, SYNCC), the execution time of the APOSS program may rise drastically. In such cases, the scan time of the profile generator can be increased to 2 ms in order to have more time available for the APOSS program. Values higher than 2 ms provide hardly any benefits.

! ! ! The VEL, ACC, and DEC have to be set after a SET PROFTIME command.

Parameter Group AXR Position Regulation

Unit ms

Value Range 1 … 5

1 = 1 ms
2 = 2 ms
3 = 3 ms
4 = 4 ms
5 = 5 ms

Init Value 1

MENCODER - 30

Content	Resolution of master encoder
Description	MENCODER indicates the number of counts of the master encoders in pulses counts per revolution. By setting MENCODER to a negative value, you can select that also master increments are scaled by SYNCFACTS and SYNCFACTM. That means 1 Master Unit [MU] = qc * SYNCFACTS / SYNCFACTM. This affects the following commands MAPOS, MIPOS, MAVEL, ON MAPOS, SETMORIGIN.
Portability	Only M1 and other new controller. Negative values are supported starting with version 6.4.53.
Parameter Group	AXE Encoder
Unit	Encoder counts per revolution
Value Range	- MLONG … MLONG
Init Value	500

RAMPMIN - 31

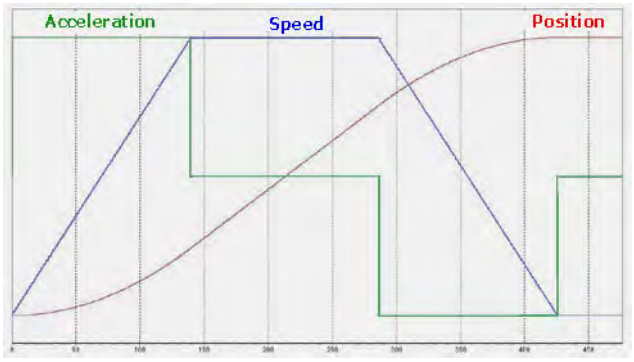
Content	Maximum acceleration
Description	The RAMPMIN parameter determines the shortest ramp (maximum acceleration). It indicates how long the acceleration phase lasts at the very least in order to achieve the rated velocity.
Portability	Only M1 and other new controller.
Parameter Group	AXB Basic Settings
Unit	ms
Value Range	1 … 360000
Init Value	1000

RAMPTYPE - 32

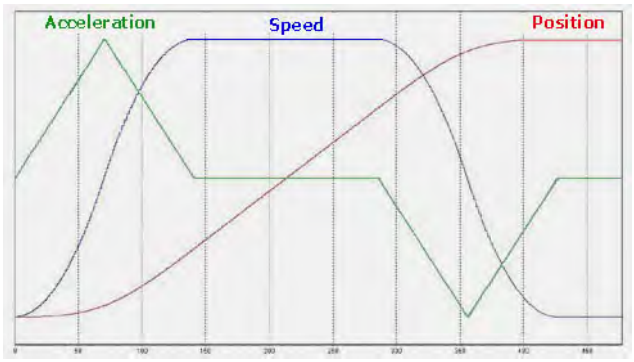
Content Ramp type

Description This parameter defines the ramp type: trapeze, sinusoidal, or limited jerk. These ramp type are relevant for all movements (POSA, POSR, CVEL, and MOTOR STOP), but not with SYNCx.

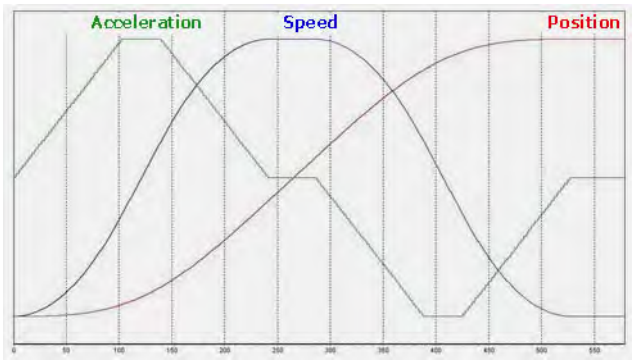
Ramp type 0 trapeze



Ramp type 1 S- ramp



Ramp type 2 limited jerk



Movements with limited jerk start with acceleration zero and increase acceleration by maximum Jerk until the maximum acceleration which is defined by RAMPMIN (31) is reached. Then the movement continues with maximum acceleration. At the end the acceleration will be decreased by maximum jerk until acceleration is zero again. The maximum jerk is calculated by JERKMIN.

There are four different parameters for the RAMPTYPE = 2 with limited jerk, see Limited Jerk in chapter “Motion Control Topics”.

Portability Only M1 and other new controller.
Value '2' is supported only with MACS3, starting with version 6.4.63,
APOSS development interface with version 6.7.40 and integrated compiler version 6.3.20.

Parameter Group AXB Basic Settings

Unit –

Value Range	0 … 2 0 = linear, trapeze 1 = S-ramp, sinusoidal 2 = movements with limited jerk
Init Value	0
Program Sample	JerkMinTest.m see **File** → Sample

DFLTVEL - 33

Content	Default velocity
Description	DFLTVEL indicates the default velocity which is always used when no velocity is defined in the process set. This value refers to the ***Velocity Resolution*** VELRES (22).
Portability	Only M1 and other new controller.
Parameter Group	AXB Basic Settings
Unit	1/VELRES
	Standard = 1/100 = %
Value Range	1 … VELRES
Init Value	50

DFLTACC - 34

Content	Default acceleration
Description	DFLTACC indicates the acceleration used when no explicit statements have been made. This statement is made in relation to RAMPMIN and refers to the VELRES (22) parameter, thus it corresponds to the Init settings in percent of maximum acceleration.
Portability	Only M1 and other new controller.
Parameter Group	AXB Basic Settings
Unit	1/VELRES
	Standard = 1/100 = %
Value Range	1 … VELRES
Init Value	50

BANDWIDTH - 35

Content Bandwidth within which the PID filter is active

Description The bandwidth in which the PID controller should function can be limited, for example to avoid the built-up of a vibration in case of operating a system which could be jeopardized by vibrations.

When bandwidth limitation (BANDWIDTH < 1000) is used, you also must enter considerably values for the parameters FFVEL (36) and FFACC (37) in order to achieve the corresponding control. A system adjusted in such a manner is not as dynamic as it could be, but is considerably more stable and tends to experience less uncontrolled vibrations.

Advantages of bandwidth limitation:

Bandwidth limitation reduces the risk of oscillation.

Disadvantages of bandwidth limitation:

Bandwidth limitation also limits the standstill torque of the control and the drive. This can be not usable for vertical applications, which need a permanent torque, even in standstill, to hold a load.

! ! ! Important: If bandwidth limitation is in use (i.e. BANDWIDTH < 1000), the feedforward control has to be activated. This means that the parameter FFVEL has to be set to a value of 1000 typically.

Portability Only M1 and other new controller.

Parameter Group AXR Position Regulation

Unit 1/10 %

The value 1000 means that the PID filter can output the full command value. For a BANDWIDTH of 500 only 50 % of the set value is output. Thus, values less than 1000 limit the P-share accordingly.

Value Range 0 … 1000

Init Value 1000

FFVEL - 36

Content Velocity Feedforward

Description When a control has a limited bandwidth then a base velocity must be set so that it can be ruled out that the control will entirely prevent the drive from running due to the limit set. FFVEL indicates the velocity with which the forward feed is completed.

The FFVEL default value 0 corresponds to a pure PID control. This means that the feedforward control is deactivated and the generated command value is just given by the PID control.

Feedforward control is mainly in use for …

- big drives (like asynchronous motors and frequency converters)
- sluggish drives
- systems which tend to oscillate and can just be used with bandwidth limitation.

Feedforward control is a must, if bandwidth limitation (i.e. Parameter BANDWIDTH < 1000) is in use.

The feedforward control generates a command value which corresponds to the command velocity. The part of command value generated by the PID regulation is added to the feed-forward command value. FFVEL has a scaling of 1000. This means, that commanding the defined maximum velocity (→ Parameter VELMAX) results in a command value of 10 V. If the subordinated servo amplifier or frequency converter is scaled to the same maximum velocity at 10 V, then FFVEL has to be set typically to 1000. Otherwise the following formula can be used to calculate FFVEL:

FFVEL = 1000 * VELMAX / (10 V speed of the power stage)

Portability Only M1 and other new controller.
With control version 6.1.14 onwards the parameter functions also, when the TIMER is unequal to 1.

Parameter Group AXR Position Regulation

Unit –

Value Range 0 … 100000

Init Value 0

Example

- The parameter VELMAX is set to 1400 rpm
- The servo amplifier or frequency converter is configured in a way, that a 10V command corresponds to 1600 rpm.

=> FFVEL = 1000 * 1400 / 1600 = 875

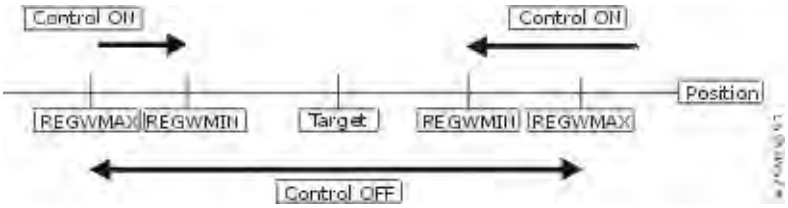
FFACC - 37

Content	Acceleration Feed forward
Description	Set the base acceleration whenever you have limited the bandwidth. Thus you will prevent the control from not accelerating at all due to the limit set. FFACC indicates the acceleration with forward feed is completed. For a normal PID algorithm this value is equal to 0.
Portability	Only M1 and other new controller.
Parameter Group	AXR Position Regulation
Unit	–
Value Range	0 … 100000
Init Value	0

REGWMAX - 38

Content Size of the control window (activation)

Description The parameters REGWMAX and REGWMIN (39) are used to turn the position control within defined areas (control windows) on and off: REGWMAX indicates the size of the window outside of which the control should begin again.



Portability Only M1 and other new controller.

Parameter Group AXR Position Regulation

Unit qc

Value Range 0 … MLONG

Init Value 0

REGWMIN - 39

Content Size of the control window (deactivation)

Description REGWMIN indicates the size of the window inside of which the control is to be deactivated until the REGWMAX (38) control window (activation) is reached again.

Portability Only M1 and other new controller.

Parameter Group AXR Position Regulation

Unit qc

Value Range 0 … MLONG

Init Value 0

HOME_TYPE - 40

Content	Behavior during movement to home.
Portability	Only M1 and other new controller.
Parameter Group	AXH Homing
Unit	–
Value Range	0 … 3

0 = Reverse and index
Moves to reference switch with HOME velocity and direction, then reverses and slowly leaves the switch, subsequently moves to the next index impulse.

1 = Reverse no index
like 0, but does not search for index impulse

2 = Forward and index
like 0 but without reversing, rather continues movement in the same direction out of the switch

3 = Forward no index
like 1 but without reversing

Init Value	0

HOME_RAMP - 41

Content Ramp for home motion.

Description Acceleration to be used during movement to home position. This statement refers to the minimum ramp, which is defined under the RAMPMIN (31) parameter. This unit results from the parameter VELRES (22) usually in % of the minimal ramp; 50% means half as fast, i.e. twice as long.

The following cohesion for HOME_RAMP results:

$$\text{HOME\_RAMP [ms]} \frac{\text{VELRES (22)}}{\text{HOME\_RAMP (41)}} * \text{RAMPMIN (31) [ms]}$$

! ! ! HOME_RAMP can never have a higher value than DFLTACC (34).

Portability Only M1 and other new controller.

Parameter Group AXH Homing

Unit VELRES

Value Range 1 … VELRES

Init Value 10

HOME_OFFSET - 42

Content Zero point offset for machine zero point or index

Description HOME_OFFSET is used to introduce an offset compared to the reference switch or index pulse. After homing, the drive is positioned to HOME_OFFSET. At this point the machine zero point or index is set.

Portability Only M1 and other new controller.

Parameter Group AXH Homing

Unit qc

Value Range –MLONG … MLONG

Init Value 0

ERRCOND - 43

Content: Behavior in event of an error

Description: In case of hard and software limit switches it is possible to clear a software limit error and then drive in the opposite direction. If you try again to move in the wrong direction, then a new error is generated.

Handling of hardware limit switches is the same as software limit switches. That means that you can clear the error and drive in the opposite direction. But if you try to move in the wrong direction, then the error 98 F_LIMIT_VIOLATION occurs.

Portability: Only M1 and other new controller.
Parameter value 4 is available starting with control version 6.1.14,
parameter value 5 with control version 6.1.3,
behavior in case of limit switches with firmware 6.6.60

Parameter Group: AXI Predefined I/Os

Unit: –

Value Range: 0 … 5

0 = Coast:
Standard, i.e. drive moves in COASTING, control loop is interrupted.

1 = Coast and brake:
Like 0, but brake output (if defined) is activated; see O_BRAKE (48)

2 = Controlled stop:
Motor stop with max. deceleration (stop ramp), subsequently standstill control

3 = Controlled stop and brake:
Like 2, brake output (if defined) is activated in addition, but only after MOTOR STOP.

4 = Only used for CAN-drives:
In case of error a special idle function is used, which does not set the reference value to zero or disable the frequency converter (FU), but set the bits to stop with FU ramp.
All other activities such as MOTOR OFF etc. must be set in the ON_ERROR routine.

5 = Jumps to the error routine, but the control will not switch off automatically. This can or must be initiated in the application program with a MOTOR OFF in the error routine. That way avoid in case of data transfer with CAN terminals that just a disturbed CAN communication stops the control. (E.g. when simple information data are transferred to a terminal and the correct display is not security relevant or the data will be updated cyclic anyway.)
Then you can check in the error routine at first, whether it is a CAN error (Err. 89) which can be deleted at once. In all other cases the control can be switched off in the error routine and the running process can be stopped corresponding the application requirements.

! ! ! A brake output has to be defined in parameter O_BRAKE (48).

Init Value: 0

ENDSWMOD - 44

Content	Behavior at limit switch
Description	This parameter defines the behavior when a positive or negative hardware end limit is activated. Error behavior see ERRCOND (43).
Portability	Only M1 and other new controller.
Parameter Group	AXI Predefined I/Os
Unit	–
Value Range	0 … 1 0 = Call error handler 1 = Controlled stop
Init Value	0

I_REFSWITCH - 45

Content	Input reference switch
Description	I_REFSWITCH defines which input should be used as home switch. It is possible to react to a positive or negative edge, using a positive or negative number. Behavior after reaching see HOME_TYPE (40).
Portability	Only M1 and other new controller.
Parameter Group	AXI Predefined I/Os
Unit	–
Value Range	1 … 8 = Reaction to a positive edge on input 1 … 8 0 = no function –1 … –8 = Reaction to a negative edge on input 1 … 8
Init Value	0

I_POSLIMITSW - 46

Content Positive limit switch

Description I_POSLIMITSW defines which input should be used as the positive limit switch. It is possible to react to a positive or negative edge, using a positive or negative number.

Limit switches and reference switches allow the usage of any input. That means not only 1..8 are supported but also larger numbers as well as virtual inputs or outputs.

! ! ! Using virtual inputs as limit switches is dangerous because it eliminates the direct connection between the limit switch and the controller. That may introduce a time delay (or other failure scenarios) which will limit the controller's ability to respond quickly to the switch.

Portability Only M1 and other new controller.
Any input and virtual I/Os is supported starting with firmware 6.6.60

Parameter Group AXI Predefined I/Os

Unit –

Value Range

0	=	no function
positive input no	=	Reaction to a positive edge on the corresponding input number
negative input no	=	Reaction to a negative edge on the corresponding input number

Init Value 0

I_NEGLIMITSW - 47

Content Negative limit switch

Description I_NEGLIMITSW defines which input should be used as the negative limit switch. It is possible to react to a positive or negative edge, using a positive or negative number.

Limit switches and reference switches allow the usage of any input. That means not only 1..8 are supported but also larger numbers as well as virtual inputs or outputs.

! ! ! Using virtual inputs as limit switches is dangerous because it eliminates the direct connection between the limit switch and the controller. That may introduce a time delay (or other failure scenarios) which will limit the controller's ability to respond quickly to the switch.

Portability Only M1 and other new controller.
Any input and virtual I/Os is supported starting with firmware 6.6.60.

Parameter Group AXI Predefined I/Os

Unit –

Value Range

0	=	no function
positive input no	=	Reaction to a positive edge on the corresponding input number
negative input no	=	Reaction to a negative edge on the corresponding input number

Init Value 0

O_BRAKE - 48

Content Output for brake

Description O_BRAKE indicates the output with which the brake can be activated.

The output is activated in the case of an error or abort if ERRCOND (43) is set to 1 or 3.

If an output is defined for the brake, this remains active even when the program is terminated with **Esc**.

A positive number means that the output is high (24 V) when the function is active. A negative number means that the output is low (0 V) when the function is active.

! ! ! The brake output must always be re-set by an OUT command in the program.

Portability Only M1 and other new controller.

Parameter Group AXI Predefined I/Os

Unit –

Value Range –8 … 8

Init Value 0

Example

```
ON ERROR GOSUB err_handle
SET O_BRAKE –1
SET ERRCOND 1
/**** Main program loop *******/
…
SUBPROG err_handle
    WAITI 1
        ERRCLR
    OUT 1 1
RETURN
```

SYNCFACTM - 49

Content Synchronization factor master (M:S)
or in CAM mode conversion qc in MU units

Description The synchronization is described with a ratio of qc (Master : Slave); SYNCFACTM determines the synchronization factor for the master.

SYNCFACTM (49) and SYNCFACTS (50) make the compensation of different drive factors possible or the adaptation of the slave speed in relation to the master speed set.

$$\text{Slave velocity} = \text{Master velocity} * \frac{\text{SYNCFACTS (50)}}{\text{SYNCFACTM (49)}}$$

In conjunction with curve synchronization the parameters SYNCFACTM and SYNCFACTS are used to transform qc into MU units. This allows the user to work with meaningful units in the CAM-Editor. See example 2 below.

$$\frac{\text{Gearing Factor} * \text{Encoder Resolution} * 4}{\text{Scaling Factor}}\ qc = 1\ MU$$

provided that:

$$\text{Gearing Factor} = \frac{\text{Motor Revolutions}}{\text{Revolutions on Output}}$$

Encoder = Incremental encoder (the multiplier 4 is omitted in the case of absolute encoders)

Scaling factor = Number of user units UU [qc] that correspond to one revolution at the drive.

Portability Only M1 and other new controller.
Starting with version 6.4.43 the factor is no longer limited to small values; see also Scaling in chapter “Programming with APOSS”.

Parameter Group AXS Synchronization

Unit qc

Value Range –MLONG … MLONG

–MLONG … –1 = turns the direction of synchronization (ratio to the master)

Init Value 1

Example 1 If the master is to run twice as fast as the slave, then the ratio is 2 : 1

SYNCFACTM = 2
SYNCFACTS = 1

Example 2 Conveyor belt:

The input should be possible in 1/10 mm resolution.

The drive is connected to the conveyor belt with a gearing of 25:11; this means that the motor makes 25 revolutions and the drive pulley 11.
Gear factor = 25/11

Incremental encoder directly on the master drive; encoder resolution = 4096

The drive pulley has 20 teeth/revolution, 2 teeth correspond to 10 mm; thus, 1 revolution = 100 mm conveyor belt feed.
Thus, the scaling factor is 1000

$$\frac{25/11 * 4096 * 4}{1000} qc = 1 \text{ MU}$$

$$\frac{25 * 4096 * 4}{1000 * 11} qc = \frac{2048}{55} qc = 1 \text{ MU} = \frac{\text{Syncfactor Master SYNCFACTM (49)}}{\text{Syncfactor Slave SYNCFACTS (50)}}$$

Set the following parameters in order to work with 1/10 degree division

SYNCFACTM = 2048

SYNCFACTS = 55

Example 3 Calculation of the scaling factor for a friction drive:

Assume that the output is equipped with a friction wheel (radius 60 mm); we want to work with a resolution of 1/10 mm:

One revolution on the output is thus calculated as follows:

Scaling factor = $2 \Pi r * 10 = 2 \Pi * 60 * 10 = 3969{,}91$

Scaling factor = 3970

Since an error will occur in any case due to the rounding, a marker adjustment must be performed after each full revolution.

SYNCFACTS - 50

Content Synchronization factor slave (M:S)
or in CAM mode conversion qc in MU units

Description The synchronization is described with a ratio of qc (Master : Slave); SYNCFACTS determines the synchronization factor for the slave.

SYNCFACTM (49) and SYNCFACTS (50) make the compensation of different drive factors possible or the adaptation of the slave speed in relation to the master speed set.

$$\text{Slave Velocity} = \text{Master Velocity} * \frac{\text{SYNCFACTS (50)}}{\text{SYNCFACTM (49)}}$$

In conjunction with CAM synchronization the parameters SYNCFACTM and SYNCFACTS are used to transform qc into MU units. This allows the user to work with meaningful units in the CAM-Editor. See example 2 below.

See prerequisites of the formula and example under SYNCFACTM (49).

Portability Only M1 and other new controller.
Starting with version 6.4.43 the factor is no longer limited to small values see also Scaling in chapter “Programming with APOSS”.

Parameter Group AXS Synchronization

Unit qc

Value Range 1 … 2 * MLONG/max. master velocity
whereby max. master velocity unit is in qc/PROFTIME(29)

Init Value 1

Sample See SYNCFACTM (49)

SYNCTYPE - 51

Content: Type of synchronization.

Description: If the default configuration (SYNCTYPE = 0) is in use, the position difference in between the master and slave is leveled out taking into account the programmed ACC, DEC and VEL limitations of the slave.

The configuration of the synchronization must be done by the bits of SYNCTYPE:

Bit 0: Not used.

Bit 1 (= +2): In case of CAM profiling (SYNCC, SYNCCMM, SYNCCMS) the bit 1 of SYNCTYPE flags, that the slave should follow the CAM profile without taking ACC, DEC and VEL limitations into account.

Bit 2 (= +4): The bit 2 of SYNCTYPE enables buffering of negative master movements. If the master movement is negative (sign of SYNCFACTM considered), then this movement is not forwarded to SYNCx but buffered. When the master than turns again and moves positive, then the same amount of quadcounts is consumed, before positive movement is continued. This allows suppressing small negative movements of the master without losing increments.

The amount of buffered increments can be read with the SYSVAR

PFG_SYNCMNEGBUFFER (4315) (0x2500 subindex 220)

! ! ! Note:
The previously selection “Look ahead” is not available anymore. The same functionality can be realized by the configuration of the parameter SYNCFFVEL.

Portability: Standard parameter with control version 6.1.14 onwards.

Change of the parameter values starting with firmware 6.7.65 (backward compatible); previous parameter value “1 = Look Ahead“ see SYNCFFVEL

Parameter Group: AXS Synchronization (It is not yet in the parameter dialog window.)

Unit: –

Value Range: 0 … 6

Bit 0 – not in use
(The previously available selection "Look ahead" is done by SYNCFFVEL.)

Bit 1

0 = standard
1 = CAM slave follows curve directly without using ACC, DEC, and VEL

Bit 2

0 = Standard
1 = Negative master movements are buffered and consumed later

Init Value: 0

SYNCMARKM - 52

Content	Marker number of the master
Description	SYNCMARKM and SYNCMARKS must be set according to the ratio between the number of marker signals from master and slave. A ration of 1:1 means that each slave marker will be aligned with each master marker. A ratio of 2:1 means that each slave marker will be aligned with each second master marker.
Portability	Only M1 and other new controller.
Parameter Group	AXS Synchronization
Unit	–
Value Range	0 … 10000
Init Value	1

SYNCMARKS - 53

Content	Marker number of the slave
Description	SYNCMARKM and SYNCMARKS must be set according to the ratio between the number of marker signals from master and slave. A ration of 1:1 means that each slave marker will be aligned with each master marker. A ratio of 2:1 means that each slave marker will be aligned with each second master marker.
Portability	Only M1 and other new controller
Parameter Group	AXS Synchronization
Unit	–
Value Range	0 … 10000
Init Value	1
Example	The master marker is an external signal which reports when a transport article arrives; the corresponding slave marker is the index impulse of the motor. If the motor always requires 3 rotations until the article arrives, then this means that 3 index impulses must elapse before a marker comes. Thus, this results in a ratio of 3 : 1; only every 3rd slave pulse is evaluated.

SYNCPOSOFFS - 54

Content	Position offset for position synchronization
Description	Defines the offset for position synchronization (SYNCP). The offset is also valid for position synchronization with marker correction (SYNCM). This position offset can be altered online at any time during the synchronization with a command.
! ! !	The offset will be executed immediately when the command SYNCP follows. When SYNCM is started, however, the system waits for the first evaluation of the market pulses. Only then is the offset applied. To avoid compatibility problems you should determine the start-up behavior of SYNCM with SYNCMSTART (62).
Portability	Only M1 and other new controls.
Parameter Group	AXS Synchronization
Unit	qc
Value Range	–MLONG/SYNCFACTS(50) … MLONG/SYNCFACTS(50)
Init Value	0

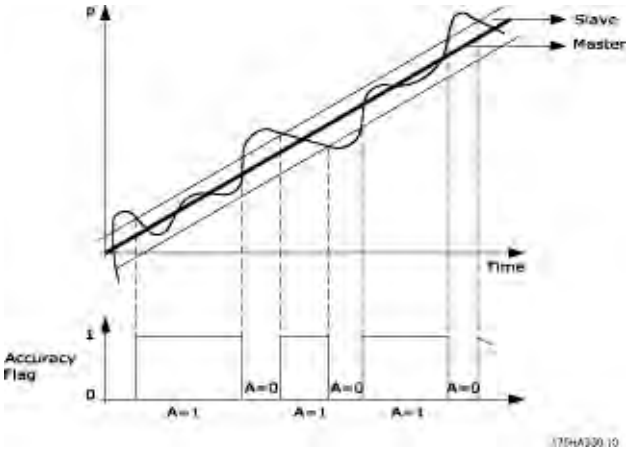
SYNCACCURACY - 55

Content Size of the accuracy window for position synchronization

Description Defines how large the difference between the actual master and slave position can be during a position synchronization (SYNCP and SYNCM), so that the required accuracy is still fulfilled. In contrast SYNCERR provides the actual synchronization error of the slave in user units.

In the program you can query whether SYNCACCURACY will be fulfilled using SYNCSTAT.

SYNCACCURACY is important for the marker synchronization in order to be able to report READY, since otherwise SYNCERR would have to be queried and compared beforehand.



Window set by SYNCACCURACY

The dark line is the position followed by master and slave follows it. SYNCACCURACY sets the window as 100. So when the slave within the window the accuracy flag sets.

! ! ! The new size of the sync windows will only be active after a new SYNC-command. This does not only concern the activation but also changes in size of the windows.

Portability Only M1 and other new controller

Parameter Group AXS Synchronization

Unit qc
or in CAM mode in UU

Value Range –MLONG … MLONG

0 … MLONG	A plus sign supplies the absolute value to SYNCERR.
–MLONG … –1	A minus sign supplies the synchronization error to SYNCERR with polarity sign. It is then possible to tell whether the synchronization is running ahead or behind.

Init Value 1000

SYNCREADY - 56

Content	Number of markers for READY
Description	SYNCREADY defines how often during marker synchronization (SYNCM and SYNCCMM) a synchronization evaluation with ACCURACY must be completed with accuracy so that ready is fulfilled. ACCURACY is checked during every correction. If ACCURACY is fulfilled then 1 is added until the set marker number has been achieved. Synchronization evaluation is always executed after m marker pulses by the master SYNCMARKM (52). ACCURAY and READY can be queried using SYNCSTAT.
Portability	Only M1 and other new controller
Parameter Group	AXS Synchronization
Unit	–
Value Range	0 … 10000
Init Value	1

SYNCFAULT - 57

Content	Marker number for fault
Description	Defines how often during marker synchronization (SYNCM and SYNCCMM) an inaccuracy may occur during a synchronization evaluation before a FAULT is registered. In the program this condition can be queried using SYNCSTAT.
Portability	Only M1 and other new controller
Parameter Group	AXS Synchronization
Unit	–
Value Range	0 … 10000
Init Value	10

SYNCMPULSM - 58

Content Master Marker Distance

Description SYNCMPULSM indicates how many qc (master) lie between two master markers or in CAM-Mode the distance between sensor and working position.

If the encoder index impulse is used as a marker signal, then the distance between two markers is the resolution [qc] of the encoder.

If external marker signals are used, then it is possible to measure the marker distance with the program 'MARKERCOUNT' (**File** → Sample), if it is unknown.

SYNCMPULSM is only valid for synchronization with marker correction (SYNCM and SYNCCMM).

In CAM synchronization, the distance of the sensor to the working position in MU will be indicated instead of the distance between two master markers. (The distance is brought about automatically by the master cycle length (Mt).)

If the parameter is larger than one master cycle length (Mt), a marker-FIFO register will be created automatically for the handling of the marker correction.

$$SYNCMPULSM * SYNCMARKM * SYNCFACTS * n < 2 * MLONG$$

whereby

n = number of the marker, which can be between master and slave, when the slave tries to catch up with the master when starting.

Portability only M1 and other new controller

Parameter Group AXS Synchronization

Unit qc
in CAM mode: MU

Value Range $0 \ldots \frac{2 * MLONG}{SYNCMARKM\ (52) * SYNCFACTS\ (50) * n}$

whereby

n = see above

Init Value 500

SYNCMPULSS - 59

Content Slave Marker Distance

Description SYNCMPULSS defines how many qc (slave) lie between two markers (slave) or in CAM-Mode the distance of the sensor to the working position in UU.

SYNCMPULSS is only valid for synchronization with slave marker correction (SYNCM and SYNCCMS).

$$SYNCMPULSS * 9 < 2 * MLONG$$

Portability Only M1 and other new controller

Parameter Group AXS Synchronization

Unit qc
in CAM mode: UU

Value Range 0 … 2 * MLONG / 9

Init Value 500

SYNCMTYPM - 60

Content Master Marker Type

Description Defines the signal type for the master marker: Index pulse of the encoder or external marker.

SYNCMTYPM is only valid for synchronizations with marker correction (SYNCM and SYNCCMM) or if the command MIPOS should be used in your program.

An extended marker type realizes a logical AND for marker input and a second input, e.g. to synchronize every second marker with a PLC control. This also influences the command MIPOS. MIPOS only delivers a new value, if the condition is true.

Digital Latch inputs (MACS4)

The tens place digits are used to select the digital input for latching.

```
SET SYNCMTYPM x(1) 42
    // use digital input 4 with rising edge as master latch input for axis 1
```

If no input is set, then the default assignment is straightforward. That means encoder 1 uses input 1, encoder 2 uses input 2 … encoder 6 uses input 6.

If the master encoder simulation (MENCODERTYPE = 6) is used on a MACS3, there is the possibility to set the SYNCMTYPM to external to register external markers. Of course, this is not so precise because only every ms is sampled if an input happened or not.

Portability Only M1 and other new controller.
Parameter value > 3 is available starting with control version 6.1.14;
digital latch inputs (MACS4) with version 6.5.18.

Parameter Group AXS Sync Marker

Cross Reference MIPOS, SYNCMTYPS (61)

Unit –

Value Range 0 … 3 ±100

0	=	encoder Z positive flank (index pulse)
1	=	encoder Z negative flank (index pulse)
2	=	external marker positive flank
3	=	external marker negative flank
tens place digits	=	nm n = digital input number (1 … 9) m = master marker type (0 … 3)
100 … 103	=	same as 0…3, but only valid if input 1 is high
200 … 203	=	same as 0…3, but only valid if input 2 is high
…		
–100…–103	=	same as 0…3, but only valid if input 1 is low
–200…–203	=	same as 0…3, but only valid if input 2 is low
…		

Init Value 0

SYNCMTYPS - 61

Content Slave Marker Type

Description Defines the signal type for the slave marker: index pulse of the encoder or external marker.

SYNCMTYPS is only valid for synchronizations with marker correction (SYNCM and SYNCCMS) or if the command IPOS should be used in the in your program.

Digital Latch inputs (MACS4):

The 10 place digits are used to select the digital input for latching.

If no input is set, then the default assignment is straightforward. That means encoder 1 uses input 1, encoder 2 uses input 2 … encoder 6 uses input 6.

SYNCMTYPS supports the following values if ENCODERTYPE == 9

0 – virtual markers are produced with a distance of ENCODERFREQ.
n2 – At positive edge of input n the actual position is taken as marker position
n3 – At negative edge of input n the actual position is taken as marker position

The accuracy is of course limited to 1 ms which is the internal update rate.

Portability Only M1 and other new controller.
Digital latch inputs (MACS4) are available starting with version 6.5.18.

Parameter Group AXS Sync Marker

Cross Reference IPOS, SYNCMTYPM (60)

Unit –

Value Range 0 … 3

0 = encoder Z positive flank (index pulse)
1 = encoder Z negative flank (index pulse)
2 = external marker positive flank
3 = external marker negative flank

tens place digits = nm
n = digital input number (1 … 9)
m = master marker type (0 … 3)

Init Value 0

SYNCMSTART - 62

Content Start behavior for synchronization with marker correction

Description SYNCMSTART indicates whether at start the synchronization should be made to the leading, subsequent or closest marker impulse of the master. SYNCMSTART is only valid for synchronization with marker correction (SYNCM and SYNCCMM).

SYNCMSTART tries to calculate a polynomial of 5th order to reach the master exactly in a marker position. The distance used to do that (number of markers) depends on the velocity difference as well as on the actual acceleration of the slave. Of course, the parameters RAMPMIN and JERKMIN are used to calculate the distance. It is tried to use a number of markers such that the actual acceleration given will not be exceeded.

Starting such a polynomial of 5th order, is only possible if the actual slave distance to the next marker corresponds to the actual master distance to the next master marker. Therefore it is easier to start if both marker positions are known when SYNCM starts. To decide that, we look at the index flags that are cleared with a SET ENCODERTYPE.

It is also easier to start if the slave velocity is zero. In such a case, it is just waited until the correlation between the marker and actual positions is acceptable. (Normally, this happens within 2 master markers).

If the master velocity is zero or near to zero, then a curve is not started at all but is leaved directly to SYNCMSTART 5 or 1005, and accordingly. The same is true if master velocity changes more than 10 % since the curve has been started (filtered velocity MasterVelSqSI – minimum 50 ms filter).

Portability Only M1 and other new controller.
Parameter 7 and 1007 are supported starting with firmware 6.7.14.

Parameter Group AXS Synchronization

Unit –

Value Range 0 … 7, 1000 … 1007, 2000

- 0 = Leading marker impulse: The slave marker following the first master marker (after SYNCM) is aligned with the first master marker.
- 1 = Subsequent marker impulse: The first slave marker (after SYNCM) is aligned with the following master marker.
- 2 = Closest marker impulse: After reaching the master velocity the next 2 markers will be aligned (correction can be forward or backward)
- 3 = After reaching the master velocity the next slave marker will be aligned with the master marker in front (correction is forward)
- 4 = After reaching the master velocity the next slave marker will be aligned with the marker behind (correction is backward)
- 5 = After reaching the master velocity the next slave marker will be aligned with the closest master marker (correction can be forward or backward, always the shortest distance).
- 6 = After the command SYNCM the first two markers are taken and the program synchronizes to these markers.
- 7 = Start with a polynomial 5th to reach the master exactly in a marker position.
- 1000 … 1007 = as above, an offset (SYNCPOSOFFS) is not active before the first marker correction is done.
- 2000 = in CAM mode: Counting of the master pulses in MU begins with the master marker.

! ! ! Only the parameter 2000 is effective in curve synchronizations.

Init Value 0

REVERS - 63

Content

Defines …
… the reverse behavior for the controlled axis,
… the use of bipolar or unipolar reference value and
… the ID of bus devices (DRIVETYPE unequal 0).

Description

The parameter REVERS determines three characteristics of controlled axes.

First of all the parameter REVERS characterizes the behavior while moving in reverse, i.e. moving in negative direction. The admissibility of the reverse drive can be defined as follows:

- Reversing is always allowed (= default)
- Reversing is only allowed, when the master is reversing
- Reversing is not allowed in general.

The corresponding restriction is always valid, i.e. for the drive synchronization (SYNCP, SYNCV, SYNCM,, SYNCC, etc.), for positioning commands (POSA, POSR), the speed command (CVEL), and also for a test run with the **Oscilloscope.** (Tip: When you want avoid the automatic reversing during a test run, set the parameter to 1 or 2.)

Secondly the parameter REVERS characterizes the way of analog reference output. Especially some type of frequency converters have just an unipolar reference input (e.g. 0…10 V instead ±10 V). By means of the parameter REVERS it can be defined for that case, that the analog reference output signal of the control is in the range of 0…10 V, and that the rotation direction (for reversing) is done via the digital output 1. A theoretical "negative" reference value (for reversing), will be converted in a positive analog signal (0…10 V) at the reference output; and the digital output 1 is set to 24 V as reversing information for the frequency converter. When forward driving and thus also theoretical positive reference value, then the analog value will be unchanged, and the digital output 1 is set to 0 V.

As a third characteristic of the controlled axis the parameter REVERS defines the node bus-ID in case of sending the reference value via the bus (i.e. parameter DRIVETYPE) ≠ 0). In this case the parameter value is calculated from the bus-ID (independent of the bus connection) multiplied with 100 plus possible restrictions of the reversing. When working with bus devices the parameter REVERS should be set before the parameter DRIVETYPE is set unequal 0.

The completing parameter value of REVERS is calculated by the addition of the single values of each characteristics.

REVERS negative numbers –10 .. –90:

To command FC drives need two separate enable signals for positive and negative direction. For that, we use two outputs starting with REVERS/10 and the next output in sequence (REVERS/10 + 1).

If the reference value is positive, output REVERS/10 is set to 1 and the other to 0. If the reference value is negative, the reference value is inverted and the outputs set vice versa. If motor is off, both outputs are reset.

Portability

Only 32-bit controller, like M1 and MACS.
Parameter values > 12 is available starting with control version 6.1.14,
negative numbers with firmware 6.5.16.

Parameter Group

AXR Position Regulation

Unit

–

Value Range 0 … 2, 10 … 12, 100 … 12702

The values of each characteristics must be added to define the parameter value of REVERS:

1. Characteristic: Behavior of Reversing

0 = reverse always allowed
1 = reverse only allowed when the master is reversed
2 = reverse not allowed

2. Characteristic: Reference value type

0 = bipolar reference signal (+/- 10V)
10 * number of the digital direction output = unipolar reference signal (0…10V) + digital direction signal at the defined output

3. Characteristic: Device bus-ID

0 = Use local reference output (DRIVETYPE = 0)
bus-ID * 100 = bus device (DRIVETYPE # 0)

The bus device must be connected to the so-called slave bus, if a control with multiple separated CAN bus networks (e.g. MACS3) is in use. There is no need to use the bus offset identifier in the parameter value of REVERS due to this fixed bus network allocation.

Samples Sample 1:
- Reversing always allowed
- Bipolar reference value (+/- 10 V)
- Local reference output in use
(i.e. parameter DRIVETYPE = 0)

=> REVERS = 0 + 0 + 0 = 0

Sample 2:
- Reversing is only allowed, when the master is reversing
- Unipolar reference signal (0…10V) + direction signal to output 1
- Local reference output in use
(i.e. parameter DRIVETYPE = 0)

=> REVERS = 1 + (10 * 1) + 0 = 11

Sample 3:
- Reversing always allowed
- Unipolar reference signal (0…10V) + direction signal to output 2
- Local reference output in use
(i.e. parameter DRIVETYPE = 0)

=> REVERS = 0 + (10 * 2) + 0 = 20

Sample 4:
- Reversing always allowed
- Bipolar reference value
- Reference output via bus to device with ID 1
(i.e. parameter DRIVETYPE # 0)

=> REVERS = 0 + 0 + (1 * 100) = 100

Sample 5:
- Reversing not allowed
- Bipolar reference value
- Reference output via bus to node with ID 96
(i.e. parameter DRIVETYPE # 0)

=> REVERS = 2 + 0 + (96 * 100) = 9602

Init Value 0

O_AXMOVE - 64

Content	Output for motion command active
Description	Set the output number which must be controlled by the AXMOVE function. The output is always activated as soon as a motion command is active, regardless in which mode (position, velocity or synchronization command). This function is not suitable for monitoring the motor, since the motor could be standing still although the control is in motion. It is possible to react to a positive or negative edge. A positive number means that the output is high (24 V) when the function is active. A negative number means that the output is low (0 V) when the function is active.
Portability	Only M1 and other new controller
Parameter Group	AXI Predefined I/Os
Unit	–
Value Range	–8 … 8
Init Value	0

SYNCVFTIME - 65

Content Velocity filter.

Syntax SET SYNCVFTIME x(n) value

n = axis number
value = filter time constants

Description This parameter configures the velocity filter which is used for the velocity synchronization. Since the velocity synchronization only uses the currently active master velocity and this can only accept very small values (e.g. 2 qc/st) and a small fluctuation in velocity can have dramatic effects. In order to even this out the following filter function is applied:

Cmdvel = Old_Cmdvel + (Actvel – Old_Cmdvel) * st/tau_filt

With the following:

Cmdvel	= set velocity
Old_Cmdvel	= last set velocity
Actvel	= actual velocity of the master
st	= sampling time (fixed 1ms)
tau_filt	= filter time constants

Generally the value for τ_filt is taken from a table, depending on the Encoder counts per revolution of the master. This value can be overwritten by the SYNCVFTIME parameter and is always used when SYNCVFTIME is not equal than zero.

If the speed filter is defined with a negative number, the corresponding value also applies for angle/position synchronization SYNCP and for marker correction SYNCM.

In this case filtering takes place as described above, but the errors made are summed up. This error sum is taken into the calculation with 1000/(τ*10) in each case, so that no position deviation can occur over prolonged periods.

The value returned by SYNCERR always contains the error made so that this is also used for the evaluation of the synchronicity. In the case of marker correction the correction value is balanced more slowly and with the same factor as the error sums.

If, for example, a filter factor of –100000 (100 ms) is used, a marker correction is balanced within 1 second (100 ms * 10). This allows a “taming” of the synchronization without restricting the acceleration.

Portability Only M1 and other new controller

Parameter Group AXS Synchronization

Cross Reference SYNCVFLIMIT

Unit τ_filt (µs)

Value Range –MLONG … MLONG

–999 … 999 = standard table

Encoder resolution	τ_filt (µs)
250	39500
256	38600
500	19500
512	19000
1000	9500
1024	9300
2000	4500
2048	4400
2500	3500
4096	1900
5000	1400

Init Value 0, i.e. the standard table is valid

SYNCVELREL - 66

Content Tolerated deviance of the slave drive from the master velocity in %.

Syntax SET SYNCVELREL x(n) value

n = axis number
value = percent value

Description This parameter indicates by how many percent the slave drive can deviate from the velocity of the master while attempting re-synchronization. For example, during changes in SYNCPOSOFFS (54) or at the start of synchronization or during the correction of deviation for marker evaluation. The following is valid:

If the slave need to catch up it runs with the maximum speed allowed; this is either the speed set with VEL or the master velocity calculated with

MAVEL + MAVEL * SYNCVELREL/100

depending which of the two is less. (MAVEL is the actual master velocity).

If the slave needs to slow down and wait for the master it will run with at least the following speed

MAVEL – MAVEL * SYNCVELREL/100.

That means, if SYNCVELREL is 50, for example, the slave will not run slower than

MAVEL/2.

Portability only M1 and other new controller

Parameter Group AXS Synchronization

Unit %

Value Range 0 … 100

0 = Off, i.e. no restriction

Init Value 0

MENCODERTYPE - 67

Content Defines the master encoder signal source.

Description MENCODERTYPE defines the signal source of the master encoder. The information can be captured via the following “input channels” and must be defined accordingly:
- Local encoder input of the control
- CANopen encoder via process data word (PDO)
- CANopen amplifier via process data word (PDO)
- Encoder type via process data word (PDO) on the EtherCAT bus

Incremental and absolute encoder data can be used as encoder input information.

Negative MENCODERTYPEs define that the master encoder is the command value of another axis in the system. This is especially useful if one axis is master and the second axis is slave. If then the command value is used instead of the actual value, then you get smoother movements.

The linked encoder does not have the identical values but it continues counting at the former position. This allows also DEFMORIGIN for the master or DEFORIGIN for the source axis to be used.

SYSVAR PFG_COMMANDPOS (4311) / PFG_OLDCOMMANDPOS (4312) had to be implemented to realize the above enhancement. They deliver the actual and the last commanded position (like 4096), but they deliver the exact values for every ms even if PROFTIME is bigger than 1000.

Peculiarities with absolute encoders When using an absolute encoders the commands DEFMORIGIN and MIPOS can not be used.

Master simulation MENCODERTYPE = 6 activates a master simulation, which can be used for example to read the master position via the bus. The velocity in quadcounts per sample time is set by a write access to the system variable SYSVAR[4105]. The simulated master position is read back by a read access to the system variable SYSVAR[4105].

If the master encoder simulation is used on a MACS3, there is the possibility to set the SYNCMTYPM to external to register external markers. Of course, this is not so precise because only every ms is sampled if an input happened or not.

Device types If encoder information is captured via the bus, the calculation of the parameter value is done on the basis of the device types as defined in parameter DRIVETYPE with following formula:

SET MENCODERTYPE nn
where nn = guard * (busoffset + device type * 1000 + id)

The encoder types with device type 1 to 49 are handled exactly like incremental encoders, where the relative change of the via bus transferred position value (PDO) will be added to last position.

The encoder types with device type 50 are handled like absolute encoders, where overflow will occur by the encoder (CANopen encoder definition).

Another encoder type gets its actual position and index position from a PDO on the EtherCAT bus:

SET MENCODERTYPE nn
where nn = busoffset + device type * 1000 + PDO no. + long offset

Example:

SET MENCODERTYPE 251000 // uses PDO[0] as EtherCAT encoder

Handling of the automatically generated CAN objects

The necessary CAN objects (PDOs, GUARD-, SYNC-object) for the information transfer of the actual position via the bus are generated automatically and the bus node is also initialized (NMT0). This object generation, which is done in background, corresponds basically to the command CANINI. In contrast to the application object generation using CANINI the automatic generated objects by MENCODERTYPE remain untouched even when an updated CANINI is done by the application later on. Even a CANDEL –1 command does not delete or stop the objects, which are automatically established by the MENCODERTYPE setting.

The automatic generated objects can just be deleted or reconfigured by a new differing SET MENCODERTYPE. Also in the case of program abort (by means of **Esc**) the objects remain existent up to the next program start. The start of a program deletes any formerly defined objects and initializes new objects according to the setting of the permanent parameters or at run-time according to any temporary parameter definition inside the program code.

Guarding

When Guarding is configured (i.e. MENCODERTYPE > 0), the bus node is monitored and a missing feedback to the GUARD object (e.g. because a node malfunction) triggers the error 88. It is up to the user to take care that the parameter REVERS (63) is set properly (corresponding to the bus-ID), the bus node is configured (baud rate, bus-ID), the drive is connected, powered up, and ready for communication at the time when the parameter MENCODERTYPE will be defined.

! ! !

If the bus device fails in a MENCODERTYPE > 0 configuration, the guarding must be switched off by the application. This can be done within the error handling routine by setting the negative MENCODERTYPE value, until the failure of the device is fixed. Otherwise the identical error message comes up again and again after every deletion of the control error status 88 (by means of ERRCLR).

PDO-mapping for CANopen drives

If the encoder information is provided by a CANopen amplifier, the so-called PDO-mapping must be done by the application before setting the parameter MENCODERTYPE. PDO-mapping means the definition of which device objects are linked into the transferred PDOs. The choice of most suitable objects depends on the CANopen amplifier. You can get sample source code suitable to the following mentioned device types from the zub machine control AG. Please tell us the used amplifier for that via e-mail (info@zub.ch).

SYNC period

Please note, that the rate of the SYNC object was adapted to the parameter TIMER of the fastest axis until firmware version 6.5.16. For up-to-date firmware versions (6.5.16 or higher) the global parameter CANSYNCTIMER specifies the periodicity of sending the SYNC telegram now. This parameter has to be set to a proper value, which fits to the slowest CAN slave device in use.

Real-time acquisition of position values based on marker signals

It is recommended to use the local encoder inputs of the control for marker synchronizations or for positioning operations, which depends on real time acquisition of marker signals (see command MIPOS, parameter SYNCMTYPM). The (hardware) latching of the position values with the maximum accuracy of 1 qc is only possible in this case.

! ! !

In case of encoder information captured by the bus, the accuracy of the marker acquisition decreases depending on the cycle time of the bus (parameter TIMER, CANSYNCTIMER) and the speed of the motor.

Portability

Only M1 and other new controller.
Master simulation via CAN-bus is available starting with control version 6.1.14;
absolute passive encoder (SSI) with version 6.7.05 (MACS4);
negative MENCODERTYPEs with version 6.7.03.

Parameter Group

AXE Encoder

Cross References

ENCODERTYPE, DRIVETYPE, REVERS, MIPOS, CANINI, TIMER, CANSYNCTIMER

Unit

–

Value Range –MLONG … MLONG

0 = Incremental encoder

1 = Absolute encoder, standard ca. 262 kHz

2 = Absolute encoder, ca. 105 kHz

5 = Sine / cosine encoder

6 = Master encoder simulation: The velocity is given with SYSVAR[4105] in quadcounts per sample time. Therefore in standard configuration with 1ms sample time the velocity has to be set in [qc/ms].

! ! ! Because SYSVAR[4105] is used also for the master position, reading and writing accesses to this SYSVAR deliver different results. In case of a writing access (SYSVAR[4105] = ...) the velocity for the master simulation is set. In case of a reading access (... = SYSVAR[4105]) the master position [qc] is returned.

10 = Incremental encoder at local encoder terminal 1

20 = Incremental encoder at local encoder terminal 2

30 = Incremental encoder at local encoder terminal 3

> 1000 = nn = encoder information captured via CAN bus

nn = guard * (busoffset + device type * 1000 + id)

guard = –1, +1 (without / with guarding)

device type (see also DRIVETYPE)
0 = Standard drive (analog output)
1 = CAN drive of type Lenze (standard)
2 = CANopen servo amplifier based on DS402
3 = CANopen servo amplifier maxon EPOS
4 = CANopen servo amplifier zub DSA
5 = CAN drive of type Lenze (reference value limit)
50 = CANopen encoder

busoffset = 100000 , 0 (slave-bus, master-bus)

id =
1…127 = bus-ID, with device type 1 … 50, and data exchange via PDO 2
257…511 = bus ID 1…127 device type 1, but data exchange via PDO 3

> 100000 = nn = Encoder information captured via EtherCAT

nn = busoffset + device type * 1000 + PDO no. + long offset

device type = 51 = Encoder information via PDO on the EtherCAT

Usage example:

```
SET MENCODERTYPE 251100   // uses pdo[1] as EtherCAT encoder
// 200000  = EtherCAT bus
// 51(000) = EtherCAT encoder
// 100s    = pdo no (e.g. 200 = pdo 2)
//   1s    = offset in pdo in longs (0 = use 1. long in pdo)
```

The first long at that place will be used for the position, the second for the index position

e.g. SET MENCODERTYPE 251105

```
// Use PDO1 long with offset 5 (PDO[6]) for position,
// and long with offset 6 (PDO[7]) for index position
```

Init Value 0

Program Sample SYNCC_MSIM.M in **File** → Sample

SYNCMWINM - 68

Content Tolerance window for marker monitoring

Description The ***Master Marker Tolerance Window*** SYNCMWINM shows how large the permitted tolerance for the occurrence of the markers is.

With the factory setting 0 the window is not monitored, which means that it is always synchronized to the next marker even if this has a considerably larger interval.

At every other setting only those markers are accepted which are within the window. If there is no marker within the tolerance window the corresponding flag (SYNCSTAT) is set and no marker correction takes place. The corresponding other marker is also ignored and only corrected the next time – i.e. no catching up to the next marker.

When SYNCM (or SYNCCSTART) is started the monitoring only begins when the first marker has been found.

! ! ! Changes of the parameter will become active immediately – not only after the next SYNCM command.

Parameter Group AXS Synchronization

Unit qc
or in CAM mode: MU

Value Range 0 … MLONG or … max. ***Marker interval*** SYNCMPULSM (58)

0	=	The window is not monitored.
1 … MLONG	=	Only one marker is accepted within the window. If no marker is within the tolerance window, the corresponding flag (SYNCSTAT) is set and no marker correction carried out. This flag can be reset with an interrupt (ON STATBIT).

Init Value 0

Sample Marker interval SYNCMPULSM = 30000
Tolerance window SYNCMWINM = 1000

Only one marker within an interval of 29000 to 31000 is accepted.

SYNCMWINS - 69

Content: Tolerance window for slave marker monitoring

Description: The ***Slave Marker Tolerance Window*** SYNCMWINS shows how large the permitted tolerance for the occurrence of the markers is.

With the factory setting 0 the window is not monitored, which means that it is always synchronized to the next marker even if this has a considerably larger interval.

At every other setting only those markers are accepted which are within the window. If there is no marker within the tolerance window the corresponding flag (SYNCSTAT) is set and no marker correction takes place. The corresponding other marker is also ignored and only corrected the next time – i.e. no catching up to the next marker.

When SYNCM (or SYNCCSTART) is started the monitoring only begins when the first marker has been found.

! ! ! Changes of the parameter will become active immediately – not only after the next SYNCM command.

Parameter Group: AXS Synchronization

Unit: qc
or in CAM mode: UU

Value Range: 0 … MLONG or. …max. ***Marker interval*** SYNCMPULSS (59)

0	=	The window is not monitored.
1 … MLONG	=	Only one marker is accepted within the window. If no marker is within the tolerance window, the corresponding flag (SYNCSTAT) is set and no marker correction carried out. This flag can be reset with an interrupt (ON STATBIT).

Init Value: 0

ESCCOND - 70

Content Condition on program termination

Description ESCCOND defines how the controller will react to a program termination using **Esc**.

Esc only terminates the program; all outputs remain in the current status (factory setting). The two new alternatives make it possible to specify conditions for the outputs. If, for example, a pump is activated, **Esc** would shut down the pump at 1, while at 2 it would continue to run.

Parameter Group AXI Predefined I/Os

Unit -

Value Range 0 … 2

0 = Controlled stop
The motor is stopped with maximum deceleration, the brake output is activated (if defined), the master simulation is stopped.
The outputs remain in the current status.

1 = Controlled stop + outputs = 0
As 0, but all are set at 0.
Exception: The brake output – if defined – is always activated.

2 = Controlled stop + outputs = 1
As 0, but all outputs are set at 1;
Exception: The brake output – if defined – is always activated.

Init Value 0

ENCODERCLOCK - 73

Content	Internal or external clock generation for SSI encoder
Description	It is possible to have an absolute passive encoder (SSI). ENCODERCLOCK setting to 0 means no clock is produced and passive mode is working.
Portability	Parameter is available starting with version 6.7.05 (MACS4).
Parameter Group	AXE Encoder Data
Unit	–
Value Range	0 = external clock generation for SSI encoder 1 = Internal clock generation for SSI encoder
Init Value	1

ENCODERFREQ - 74

Content	Specifies the frequency for SSI encoder.
Description	It is possible to have an absolute passive encoder (SSI). ENCODERFREQ sets the data rate of the SSI encoder.
Portability	Parameter is available starting with version 6.7.05 (MACS4).
Parameter Group	AXE Encoder Data
Unit	kHz
Value Range	39000 - 5000000
Init Value	262000

MENCODERCLOCK - 77

Content	Internal or external clock generation for SSI master encoder
Description	It is possible to have an absolute passive master encoder (SSI). MENCODERCLOCK setting to 0 means no clock is produced and passive mode is working.
Portability	Parameter is available starting with version 6.7.05 (MACS4).
Parameter Group	AXE
Unit	–
Value Range	0 = external clock generation for SSI master encoder 1 = Internal clock generation for SSI master encoder
Init Value	1

VELKPROP - 90

Content	Proportional value of the PI velocity control loop.
Description	The ***Proportional factor*** VELKPROP of the ***velocity control*** loop indicates the linear correction factor with which the deviation between the current set and actual velocity is evaluated and a corresponding correction of the motor speed is made.
! ! !	This parameter is just of importance, if a MACS control unit with integrated power amplifiers is in use and the parameter AMPMODE is set to a mode with an activated velocity control loop. Hint: For easy and quick setup without taking care of velocity control parameters, the AMPMODE can be set to a value without velocity control. For most applications and motor types this also offers good motor control quality and adequate accuracy of position and speed. Remark: The VELKPROP is independent of the parameter ENCODER. If stable velocity control parameters are found and the encoder in use is changed to one with another resolution later on, these settings can stay untouched and just the ENCODER resolution and the position control loop parameters KPROP, KDER, KINT have to be adapted.
Portability	Just available for MACS control units with integrated power amplifiers and firmware version 6.4.63 or higher.
Parameter Group	AXA Amplifier
Cross Reference	VELKINT, VELKILIM, AMPMODE
Unit	–
Value Range	-32767 … 32767
Init Value	200

VELKINT - 91

Content	Integral value of the PI velocity control loop.
Description	The ***Integral factor*** VELKPROP of the ***velocity control*** loop is the weighting factor, with which the sum of all velocity control errors is evaluated.
! ! !	This parameter is just of importance, if a MACS control unit with integrated power amplifiers is in use and the parameter AMPMODE is set to a mode with an activated velocity control loop. Hint: For easy and quick setup without taking care of velocity control parameters, the AMPMODE can be set to a value without velocity control. For most applications and motor types this also offers good motor control quality and adequate accuracy of position and speed. Remark: The VELKINT is independent of the parameter ENCODER. If stable velocity control parameters are found and the encoder in use is changed to one with another resolution later on, these settings can stay untouched and just the ENCODER resolution and the position control loop parameters KPROP, KDER, KINT have to be adapted.
Portability	Just available for MACS control units with integrated power amplifiers and firmware version 6.4.63 or higher.
Parameter Group	AXA Amplifier
Cross Reference	VELKPROP, VELKILIM, AMPMODE
Unit	–
Value Range	-32767 … 32767
Init Value	5

CURKPROP - 92

Content Proportional value of the PI current / torque control loop

Description The ***Proportional factor*** CURKPROP of the ***current / torque control*** loop indicates the linear correction factor with which the deviation between the commanded and actual current is evaluated and a corresponding correction of the motor current is made.

! ! ! This parameter is just of importance, if a MACS control unit with an integrated power amplifier is in use and the parameter AMPMODE is set to a mode with an activated current / torque control loop.

Hint:
The value of CURKPROP can be calculated!

Required input data of the formula

- Beta: Control dynamics factor
  0.1 ... 0.3 (0.1 = slow / 0.3 = fast)
- Ts: Sampling time [ms]
  0.000125 (for MACS4-DC3 and MACS4-DC6)
- U: Supply voltage of the power amplifier [V]
  12V ... 48V
- kc: Gain of the power amplifier
  6554 for MACS4-DC6
  3495 for MACS4-DC3
- L: Inductancy of the motor [H]
  see motor data sheet, but take care of the unit!
- R: Resistance of the motor winding [Ohm]
  see motor data sheet

Fixed internal factors

- ku = 2^-12 * 2^-15 = 0.000000007

Intermediate calculated values

- w = Beta * π / (2 * Ts)

Calculation formulas

- CURKPROP = ((2 * w) - (R / L)) * L / (U * kc * ku)
- CURKINT = ((w^2 * L) / (kc * ku * U)) * (Ts / 4)

Portability Just available for MACS control units with integrated power amplifiers and firmware version 6.4.63 or higher.

Parameter Group AXA Amplifier

Cross Reference CURKINT, CURKILIM, AMPMODE

Unit –

Range -32767 … 32767

Init Value 200

CURKINT - 93

Content Integral value of the PI current / torque control loop.

Description The ***Integral factor*** CURKPROP of the ***current / torque control*** loop is the weighting factor, with which the sum of all current control errors are evaluated.

! ! ! This parameter is just of importance, if a MACS control unit with integrated power amplifiers is in use and the parameter AMPMODE is set to a mode with an activated current / torque control loop.

Hint:
The value of CURKPROP can be calculated!

Required input data:

- Beta: Control dynamics factor
  0.1 ... 0.3 (0.1 = slow / 0.3 = fast)
- Ts: Sampling time [ms]
  0.000125 (for MACS4-DC3 and MACS4-DC6)
- U: Supply voltage of the power amplifier [V]:
  12V ... 48V
- kc: Gain of the power amplifier
  6554 for MACS4-DC6
  3495 for MACS4-DC3
- L: Inductance of the motor [H]:
  see motor data sheet, but take care of the unit!
- R: Resistance of the motor winding [Ohm]:
  see motor data sheet

Fixed internal factors:

- ku = 2^-12 * 2^-15 = 0.000000007

Intermediate calculated values:

- w = Beta * π / (2 * Ts)

Calculation formulas:

- CURKPROP = ((2 * w) - (R / L)) * L / (U * kc * ku)
- CURKINT = ((w^2 * L) / (kc * ku * U)) * (Ts / 4)

Portability Just available for MACS control units with integrated power amplifiers and firmware version 6.4.63 or higher.

Parameter Group AXR power amplifier control loop

Cross Reference CURKPROP, CURKILIM, AMPMODE

Unit –

Range -32767 … 32767

Init Value 100

AMPMAXCUR - 94

Content	Maximum allowed motor current for MACS control units with integrated power amplifiers.
Description	This parameter limits the maximum motor current. The motor current cannot exceed the defined value.
! ! !	In case of servo motors, it is common that the maximum current is set to a value higher than the nominal motor current. This allows applications with high dynamic acceleration and deceleration. But: It has to make sure, that the RMS motor current (depending on the profile and periodicity) is not higher than the nominal current otherwise the motor may get damaged! It is within the responsibility of the machine design engineer, that the motor is able to handle the required power. If the motor current is limited and the required torque for the application cannot be generated by the motor, this results in tracking errors. The motor current limitation is just active, if a MACS control unit with integrated power amplifiers is in use and the parameter AMPMODE is set to a mode with an activated current / torque control loop.
Portability	Just available for MACS control units with integrated power amplifiers and firmware version 6.4.63 or higher.
Parameter Group	AXA Amplifier
Cross Reference	AMPMODE
Unit	mA
Value Range	0 … 60000 (The maximum value depends on the MACS4 power amplifier in use.)
Default	2000

AMPCOMMTYPE - 95

Content Definition of the motor type in use for MACS control units with integrated power amplifiers.

Description This parameter defines the type and commutation of the motor connected to the MACS control unit power amplifier.

0 = No motor connected
1 = Brush motor connected
2 = Brushless motor using 120° hall sensors
3 = Brushless motor using 60° hall sensors
4 = Stepper motor, field oriented control
5 = Stepper motor, micro stepper control
6 = Coupled power amplifier control (2 output drivers for one brush motor)

The availability and application of the different AMPCOMMTYPEs depends strongly on the MACS control unit and OEM hardware in use.
The MACS4-DC3 can handle AMPCOMMTYPEs 0, 1, 2, 3
The MACS4-DC6 can handle AMPCOMMTYPEs 0, 1
The stepper motor control and coupled power amplifier control is limited to OEM versions of MACS hardware.

Portability Just available for MACS control units with integrated power amplifiers and firmware version 6.4.63 or higher.

Parameter Group AXA Amplifier

Cross Reference AMPMODE

Unit –

Value Range 0 … 6

Default 0

AMPPWMFREQ - 96

Content	Frequency of the PWM power amplifier of integrated MACS4 devices.
Description	This parameter defines the PWM frequency of the power amplifier. A higher frequency reduces audible noise and reduces heating up of low-inductance motors at standstill. The drawback is a little bit worse efficiency factor due to higher loss during each switching phase.
Portability	Just available for MACS control units with integrated power amplifiers and firmware version 6.4.63 or higher.
Parameter Group	AXA Amplifier
Cross Reference	AMPMODE
Unit	Hz
Value Range	10000 … 50000 The maximum PWM frequency and the adjustable values depend on the MACS4 hardware! Example: MACS4-DC3 and MACS4-DC6 Maximum PWM frequency is 48000 Hz. The given PWM frequency has to be a multiple of 8000 Hz, i.e. 16000, 24000, 32000, 40000 or 48000 kHz. If differing values are defined by the parameter, the quality of the regulation might be worse due to incorrect PWM adjustment.
Default	24000

AMPMODE - 97

Content Definition of the control loop structure in use for MACS control unit versions with integrated power amplifiers.

Description This parameter defines, what control loops are activated and how the control structure looks like for MACS control unit versions with integrated power amplifiers:

0 = Position control -> Velocity control -> Current control -> PWM
1 = Position control -> Velocity control -> PWM
2 = Position control -> Current control -> PWM
3 = Position control -> PWM

If just velocity and / or current / torque control is required by an application, then the parameter AMPMODE has to be configured to a mode with the requested control structure in it. The application program can use the commands MOTOR OFF and OUTAN to disable the position control loop and send the set value (defined by the command OUTAN) directly to the control loop underneath the (inactive) position control.

In fact MOTOR OFF disables all control loops of the given axis. The command OUTAN enables all control loops below the position control loop according to the AMPMODE setting.

If AMPMODE is set to 0 and MOTOR OFF and OUTAN is called up by the application,
… the velocity control loop and the current / torque control loop is enabled
… the OUTAN value is the set value for the velocity control loop

If AMPMODE is set to 1 and MOTOR OFF and OUTAN is called up by the application,
… just the velocity control loop is enabled
… the OUTAN value is the set value for the velocity control loop

If AMPMODE is set to 2 and MOTOR OFF and OUTAN is called up by the application,
… just the current / torque control loop is enabled
… the OUTAN value is the set value for the current / torque control loop

! ! ! Please notice, that there are no motor current limitations possible for the AMPMODEs 1 and 3. The current limitation (AMPMAXCUR) is just activated, if a control loop structure containing a current / torque control is selected.

A modification of the AMPMODE control structure might also influence the control quality and stability. An adapted tuning of formerly set control parameters might be necessary in such a case.

AMPMODE 3 is most adequate for easy and quick setup of a drive system. This mode just requires tuning of the parameters of the position control loop (KPROP, KDER, KINT).

Portability Just available for MACS control units with an integrated power amplifier and firmware version 6.4.63 or higher.

Parameter Group AXA Amplifier

Cross Reference KPROP, KDER, KINT, KILIM
VELKPROP, VELKINT, VELKILIM
CURKPROP, CURKINT, CURKILIM, AMPMAXCUR, AMPCOMMTYPE
MOTOR OFF, OUTAN

Unit –

Value Range 0 … 3

Default 0

JERKMIN - 98, 100, 101, 102

Content

JERKMIN	Defines the minimum time [ms] required … before reaching the maximum acceleration. or
JERKMIN2	to ramp the acceleration down from maximum acceleration to 0.
JERKMIN3	to ramp the deceleration up from 0 to maximum deceleration..
JERKMIN4	to ramp the deceleration down from maximum deceleration to 0.

Description

There are four different JERKMIN options, see "Limited Jerk" in chapter I.

The maximum Jerk used with RAMPTYPE (32) = 2 is calculated by JERKMIN using following formulas:

Maximum Acceleration = Maximum Velocity / RAMPMIN
Maximum Jerk = Maximum Acceleration / JERKMIN

Note, that RAMPMIN and JERKMIN are time values in milliseconds.

Calculation sample:
VELMAX = 3000 [RPM]
ENCODER = 500 counts/rev [PPR]
RAMPMIN = 500 ms
JERKMIN = 200 ms

This results in:
VELMAX = 3000 * 500 * 4/60 = 100,000 qc/s = 100 qc/ms
MaxAcc = 200,000 qc/s^2 = 0.2 qc/ms^2
MaxJerk = 1,000,000 qc/s^3 = 0.001 qc/ms^3

! ! ! A changed setting of JERKMIN is executed after the next movement command (POSA, POSR, CVEL, or MOTOR STOP) and used for the calculation of the acceleration. Thus it is possible in NOWAIT ON mode with a repeated POSA command to adapt during a movement online (i.e. without stop) the movement to new definition.

! ! ! If JERKMIN2, JERKMIN3 and/or JERKMIN4 is set to "0" this will default to the same value as JERKMIN.

Portability Only with MACS3 and firmware version 6.4.63 onwards, APOSS development interface version 6.7.40 with integrated compiler version 6.3.20.

Parameter Group AXB Basic Settings

Unit ms

Value Range 0 … MLONG

Init Value 0

VELKILIM - 103

Content	Limit value for the integral sum of the PI velocity control loop.
Description	This parameter limits the integral sum of the PI ***velocity control*** loop in order to avoid instability and PI wind-up.
! ! !	This parameter is just of importance, if a MACS control unit with an integrated power amplifier is in use and the parameter AMPMODE is set to a mode with an activated velocity control loop.
Portability	Just available for MACS control units with an integrated power amplifier and firmware version 6.6.18 or higher.
Parameter Group	AXA Amplifier
Cross Reference	VELKPROP, VELKINT, AMPMODE
Unit	–
Range	0 ... 32767
Init Value	1000

CURKILIM - 104

Content	Limit value for the integral sum of the PI current / torque control loop.
Description	This parameter limits the integral sum of the PI ***current / torque control*** loop in order to avoid instability and PI wind-up.
! ! !	This parameter is just of importance, if a MACS control unit with integrated power amplifiers is in use and the parameter AMPMODE is set to a mode with an activated current / torque control loop.
Portability	Just available for MACS control units with integrated power amplifiers and firmware version 6.6.18 or higher.
Parameter Group	AXA Amplifier
Cross Reference	CURKPROP, CURKINT, AMPMODE
Unit	–
Range	0 … 32767
Init Value	1000

KILIMTIME - 105

Content Time (ms) which is used to increase or decrease the integral limit of the position control loop up to KILIM.

Description The integral part of the PID ***position control loop*** can be active all the time or just during a movement or just at standstill. The value of this parameter determines this behavior.

The default value of 0 means, that the integral part of the PID position control loop is active all the time according to the parameter KINT and the limitation parameter KILIM.

A value bigger than 0 means, that the integral part of the PID position control loop is activate just during a motor movement. If the motor is in standstill, the integral part is reduced to zero. If the motor starts to move, the integration limit is increased from zero to the defined KILIM value within the period of time given by the KILIMTIME setting. If the motor stops again the integral part is reduced again by decreasing the limit to zero within the defined period. This handling of the integral part can be an advantage for synchronization applications, where low synchronization errors are requested, but no hard regulation at standstill is wanted.

A value less than 0 means, that the integral part of the PID position control loop is activate just at standstill. If the motor starts to move, the integral part is reduced to zero by decreasing the integration limit within the period of time given by the absolute value of the KILIMTIME setting. If the motor stops again, the integration limit increased from zero to the defined KILIM within the defined period of time. This handling of the integral part can be helpful to prevent nervous drive behavior during a movement, but still get a very accurate positioning result at standstill.

Portability MACS firmware 6.6.71 or higher

Parameter Group AXR Position Regulation

Cross Reference KPROP, KDER, KINT, KILIM

Unit ms

Range -32767 … 32767

Init Value 0

0 = Integral part of PID position regulation always active
< 0 = Integral part of PID position regulation just active at standstill
> 0 = Integral part of PID position regulation just active during movement

SYNCSFTIME - 106

Content	Filter time [ms] for slave marker correction.
Description	Filter time for slave encoder (like SYNCMFTIME for master). If SYNCSFTIME is set, then the MMarkerDist is calculated a little differently. If the time set results in a number of markers which is smaller than 100, then at least 100 are taken to calculate the average marker distance (Filter). This only affects the calculation of the marker distance. The other filter calculations (deviation) are still done with the number of markers calculated by SYNCSFTIME and actual velocity of master. See also Marker Correction illustration in chapter “Technical Reference”.
Portability	Parameter is available starting with firmware 6.6.18
Parameter Group	AXS
Cross Reference	SYNCMFTIME
Unit	ms
Range	0 … MLONG 0 = no filtering
Init Value	0

ENCCONTROL - 107

Content Configuration of position evaluation after a change of encoder source.

Description Soft encoder changing (Bit 0) means that when the encoder is changed, the position of the new encoder is ignored and only relative changes are added to actual position. This allows for changing between different encoders (i.e. different masters) without producing jumps. If this feature is deselected, the exact value of the new encoder will be taken when changing the encoders.

Soft zero setting (Bit 1) means, that a DEFORIGIN, SETORIGIN, DEFCORIGIN, or RSTORIGIN will not really change the internal encoder position but only the virtual encoder used for that axis. For example, this feature allows two axes to use the same encoder (one axis as slave, the other as master encoder). If this feature is enabled, the first axis can do a DEFORIGIN for the slave encoder without producing a jump for the master of axis 2. The real encoder position is unchanged.

Example:

```
SET ENCODERTYPE    10 / 20
    // axes one uses the first encoder as slave, axes 2 uses the second encoder
SET MENCODERTYPE  30 / 10       // axes 2 master is actually the encoder of axes 1
SET ENCCONTROL    2       // soft zero setting for axes 1 slave encoder
..
PRINT "apos x1 ",apos x(1), " mapos x(2) ",mapos x(2), "encoder(2) ",sysvar[42]
    // this print will produce a result like
    // " apos x(1)  25000 mapos x(2) 25000 encoder(2) 25000
..
DEFORIGIN x(1)
PRINT "apos x1 ",apos x(1), " mapos x(2) ",mapos x(2), "encoder(2) ",sysvar[42]
    // this print will produce a result like  " apos x(1)  0 mapos x(2) 25000 encoder(2) 25000
```

If MENCCONTROL was set of x(2) to 2, then also a DEFMORIGIN x(2) could be executed without causing any trouble.

Portability Parameter is available starting with firmware 6.6.18.

Parameter Group AXE Encoder

Unit –

Range

0	= Switch directly to position data of new source.
Bit 0 = 1	= soft change, i.e. position of old source + relative data of new
Bit 1 = 1	= soft zero setting (DEFORIGIN … does not reset encoder)

Init Value 0

MENCCONTROL - 108

Content Configuration of master position evaluation after a change of encoder source.

Description Soft encoder changing is useful if encoders should be switched while running. If this is done without using this parameter, then setting the new encoder will typically cause a position error, because the encoder values are not the same. If soft encoder changing is chosen, then it does not switch entirely to the value of the new encoder, but it keeps the old value and just adds the differences from the new encoder. This makes it possible to change encoders "on the run".

Soft zero setting is very useful, for example, if the encoder is used by another axis or for some other purpose and you don't want to really change the encoder value when you use a DEFMORIGIN. If soft zero setting is on, then you can use a DEFMORIGIN and the new reported actual position is zero afterwards. However, the encoder still has its old value (i.e. the value you can see if you read it by using the SYSVAR 0x0122022A or similar).

Portability Parameter is available starting with firmware 6.6.18.

Parameter Group AXE Encoder

Unit –

Range

0	= Switch directly to position data of new master source.
Bit 0 = 1	= soft change, i.e. position of old source + relative data of new
Bit 1 = 1	= soft zero setting

Init Value 0

SYNCFFVEL - 109

Content	Velocity feed forward [per mill of VCMD] for synchronization modes.
Description	SYNCFFVEL supports velocity dependent feed forward in synchronous modes (SYNCP / SYNCM / SYNCC). This parameter is either 0 (disabled), which is default, or a value which gives the feed forward in 1/1000 of command velocity. That means a value of 1000 adds a feed forward to MPCMD (sync command position) of VCMD (command velocity). To determine the value to use, the SYSVAR NORMTRACKERR can be used. If you find out that your normal NORMTRACKERR is 100 percent, then you should set the SYNCFFVEL to 1000. This allows to minimize SYNCERR. With such a feed forward the SYNCERR could be nearly eliminated without having the disadvantages of using an INT part in the PID. At the moment there is still the disadvantage that you have a bump if you are starting SYNCP while running. This occurs if you come up to velocity by a SYNCC or a CVEL and then start SYNCP. SYSVAR REG_NORMTRACKERR (4124): This SYSVAR gives back the track error in relation to command velocity in percent. For example, a value of 120 tells you that the track error is 1.2 times VCMD. This value is relatively constant if the conditions don't change (load, friction, …). This value typically is independent of the velocity.
Portability	Parameter and new SYSVAR 4124 are available starting with firmware 6.6.24
Parameter Group	AXS Synchronization
Cross Reference	SYNCP, SYNCM, SYNCC, see SYSVAR 4124 in SDO dictionary, axis process data
Unit	–
Range	0 … MLONG 0 = disabled
Init Value	0

SYNCVFLIMIT - 110

Content	Sync error window [qc] for automatic deactivation of SYNCVFTIME.
Description	SYNVFTIME is deactivated, when SYNCERR exceeds the value given by SYNCVFLIMIT. SYNVFTIME is activated again, when SYNCERR decreases below 1/5 of SYNCVFLIMIT. This parameter helps to avoid big synchronization errors when master is accelerating or decelerating and a large value for SYNCVFTIME is used. This parameter allows the deactivation of the SYNVFTIME filter if the error becomes bigger than SYNCVFLIMIT. If the filter error (PFG_MFILTERROR) exceeds the value of SYNCVFLIMIT [qc] then the filter is deactivated slowly. If the error becomes smaller than SYNCVFLIMIT/5, then the filter is activated again. In case of SYNCM, the internal filter error is reset every SYNCVFTIME to zero because it is only growing and never decreasing. In that case, after SYNCVFTIME it is again checked if it is small enough to reactivate it. The filter is not disabled or enabled immediately, but it will be increased or decreased slowly by 1 ms per ms. And it is not disabled totally, but it is leaved at least at 5 ms. In the case of SYNCP, the original filter value is still used to calculate the actual filter error (PFG_ MFILTERROR) to decide if the filter should be enabled again. In the case of SYNCM (no error correction), the filter error is reset to zero and at least SYNCVFTIME is waited to decide if the filter error stays below limit or not. If it stays below limit, then the filter is activated again. The filter error can be observed using the SYSVAR PFG_MFILTERROR.
Portability	Parameter is available starting with 6.6.56.
Parameter Group	AXS Synchronization
Cross Reference	SYNCVFTIME
Unit	qc
Range	0 … MLONG
Init Value	0

POSERRTIME - 111

Content Time frame [ms] for triggering position error state.

Description Too big tracking errors (cp. POSERR) will just trigger error state, if they exist longer than POSERRTIME..

This parameter has a default of 0. If this parameter is not 0, then a position error is only produced if the position error value (POSERR) is exceeded for a time longer than POSERRTIME. Internally the position error is checked every 20 ms for one axes. That means a detection of a position error can take up to NoOfAxes * 20 ms in the worst case. If you have for example 2 axes and the POSERRTIM is set to 300 ms, then we check if a position error has been detected more than (300 / 2 * 20) = 7 times. MCO 5.00 … more than (300 / 1 * 20) = 15 times.

If this is true then an error is generated.

Portability Parameter is available starting with firmware 6.6.32.

Parameter Group AXR Position Regulation

Cross Reference POSERR

Unit ms

Range 0 … 10000

0 = no time frame active, i.e. error state is triggered immediately.

Init Value 0

FEEDDIST - 112

Content	User unit factor to get CANopen compatible user units.
Description	Other factors: FEEDREV, POSENCQC, POSENCREV. Defaults are set in a way to be backwards compatible. User unit calculation formula: $UU * \frac{FEEDREV * POSFACT\_Z * POSENCQC}{FEEDDIST * POSFACT\_N * POSENCREV} = qc$ (The POSFACT_Z und POSFACT_N corresponds to the CANopen DS402 object 0x6091 “Gear ratio”.)
Portability	Parameter is available starting with firmware 6.6.51.
Parameter Group	AXE (not displayed in axis parameter fields)
Cross Reference	FEEDREV, POSENCQC, POSENCREV, POSFACT_Z, POSFACT_N
Unit	–
Range	1 … MLONG
Init Value	1

FEEDREV - 113

Content	User unit calculation [qc] to be CANopen compatible.
Description	User unit calculation formula: $UU * \frac{FEEDREV * POSFACT\_Z * POSENCQC}{FEEDDIST * POSFACT\_N * POSENCREV} = qc$ (The POSFACT_Z und POSFACT_N corresponds to the CANopen DS402 object 0x6091 “Gear ratio”.)
Portability	Parameter is available starting with firmware 6.6.51.
Parameter Group	AXE (not displayed in axis parameter fields)
Cross Reference	FEEDREV, POSENCQC, POSENCREV, POSFACT_Z, POSFACT_N
Unit	qc
Range	1 … MLONG
Init Value	1

POSENCQC - 114

Content	User unit factor to be CANopen compatible.
Description	User unit calculation formula: $UU * \frac{FEEDREV * POSFACT\_Z * POSENCQC}{FEEDDIST * POSFACT\_N * POSENCREV} = qc$ (The POSFACT_Z und POSFACT_N corresponds to the CANopen DS402 object 0x6091 “Gear ratio”.)
Portability	Parameter is available starting with firmware 6.6.51.
Parameter Group	AXE (not displayed in axis parameter fields)
Cross Reference	FEEDREV, POSENCQC, POSENCREV, POSFACT_Z, POSFACT_N
Unit	–
Range	1 … MLONG
Init Value	1

POSENCREV - 115

Content	User unit factor to be CANopen compatible.
Description	User unit calculation formula: $UU * \frac{FEEDREV * POSFACT\_Z * POSENCQC}{FEEDDIST * POSFACT\_N * POSENCREV} = qc$ (The POSFACT_Z und POSFACT_N corresponds to the CANopen DS402 object 0x6091 “Gear ratio”.)
Portability	Parameter is available starting with firmware 6.6.51.
Parameter Group	AXE (not displayed in axis parameter fields)
Cross Reference	FEEDREV, POSENCQC, POSENCREV, POSFACT_Z, POSFACT_N
Unit	–
Range	1 … MLONG
Init Value	1

HOMEZEROVEL - 116

Content	Scaled velocity for moving into the encoder index.
Description	Scaled velocity (cp. VELRES) for moving into the encoder index as part of the HOME command or CANopen homing procedures. The parameter HOMEZEROVEL is not used by the firmware presently. It is just needed for the application CANopenStateMachine, where the user can also define the velocity for the index pulse search (for the HOME command a fixed fraction of HOMEVEL is taken mostly. The user sets this value normally by SDO 0x6099 subindex 2 in user units per second. This value is converted automatically and stored in VELRES in parameter HOMEZEROVEL.
Portability	Parameter is available starting with firmware 6.6.51.
Parameter Group	AXH (not displayed in axis parameter fields)
Cross Reference	VELRES
Unit	typ. % (VELRES)
Range	1 … MLONG
Init Value	10

I2TFTIME - 117

Content Filter time in ms to build up the I² filtered value.

Description All controllers with embedded amplifiers also support a kind of I²T limitation. For that reason the following parameters are implemented.

I2TLIMIT	defines the limit. If this limit is exceeded an error 99 (F_I2TLIMIT) is released.
I2TFTIME	defines the filter time in ms which is used to build up the I² filtered value.
CURFTIME	defines the filter time for the measurement of the current. This does not influence the regulation of current, only the read out value which is used for the above I²T limiting and by the SYSVAR (see below).

AMP_I2TVALUE is a new SYSVAR [0x2700 / 27] which delivers the filtered value which is used for the limitation.

Internally every ms the current value in mA is read out and I² is built. This value is then scaled in 1/1000 A². The value is then put into a PT-filter and the result is used to compare with I2TLIMIT.

This value gives an idea of how many energy was fed into the motor and this of course is proportional to the energy heating up the motor.

! ! ! When this parameter is set to a value bigger than zero, it automatically enables surveillance of the limit; see I2LIMIT.

Portability Parameter is available starting with firmware 6.6.19.

Parameter Group AXA (not displayed in axis parameter fields)

Cross Reference I2LIMIT, CURFTIME

Unit ms

Range 0 … MLONG

Init Value 0

I2TLIMIT - 118

Content	Limitation of the parameter I2TVALUE (see 0x2700 Subindex 27).
Description	All controllers with embedded amplifiers also support a kind of I^2T limitation. For that reason the following parameters are implemented.

I2TLIMIT defines the limit. If this limit is exceeded an error 99 (F_I2TLIMIT) is released.

I2TFTIME defines the filter time in ms which is used to build up the I^2 filtered value.

CURFTIME defines the filter time for the measurement of the current. This does not influence the regulation of current, only the read out value which is used for the above I^2T limiting and by the SYSVAR (see below).

AMP_I2TVALUE is a new SYSVAR [0x2700 / 27] which delivers the filtered value which is used for the limitation.

Internally every ms the current value in mA is read out and I^2 is built. This value is then scaled in 1/1000 A^2. The value is then put into a PT-filter and the result is used to compare with I2TLIMIT.

This value gives an idea of how many energy was fed into the motor and this of course is proportional to the energy heating up the motor.

Example:

Assume the nominal current of a motor is 2.5 Amps. If the motor is now driving over a longer time period (about 3 times longer than I2TFTIME) with nominal load, then the readout value of AMP_I2TVALUE will be around 6250 which is calculated as 2.5 * 2.5 * 1000.

So this value gives an indicator to judge if the motor might be overloaded or not.

If you use this limitation to switch off your motor, this does of course not guarantee that your motor is really protected. The overheating of the motor depends on many other factors and can't be really calculated by a simple I^2T measurement

Portability	Parameter is available starting with firmware 6.6.19.
Parameter Group	AXA (not displayed in axis parameter fields)
Cross Reference	I2TVALUE see SDO object dictionary, I2TFTIME, Error_99
Unit	A^2 (1/1000)
Range	0 … MLONG
Init Value	0

CURFTIME - 119

Content	Filter time for the measurement of the current.
Description	Filter time for the measurement of the current. This does not influence the regulation of current, only the read out value which is used for the above I^2T limiting and by the SYSVAR AMP_GETCUR.
Portability	Parameter is available starting with firmware 6.6.19.
Parameter Group	(not displayed in axis parameter fields)
Cross Reference	SDO object dictionary
Unit	ms
Range	
Init Value	0

AMPENCNO - 126

Content Definition of the encoder input, which is in use as the feedback source of the velocity control loop.

Description The feedback source for the velocity control loop can be defined separately from the feedback source of the position control loop. There are 3 parameters required to configure the feedback system of the velocity control loop:

AMPENCNO	Definition of the encoder source
AMPENCRES	Encoder resolution [qc/rev.]
AMPENCRPM	Maximum velocity

By standard, AMPENCNO is set to -1. This will result in legacy behavior, which means the feedback encoder, which is in use for the position control loop will also be in use by velocity control loop (as it was for previous firmware versions).

If AMPENCNO is set to 1, encoder 1 will be used as feedback source, a value of 2 corresponds to encoder 2 and so on. Parameter AMPENCRES has to be used to define the resolution [qc/rev.] (!!) of the used encoder and AMPENCRPM must be set to the maximum velocity in RPM. The default value 0 of AMPENCRPM means that the maximum velocity given by VELMAX is also valid for the velocity control loop.

The parameter POSDRCT can be used to configure the kind of feedback evaluation (normal or inverted) separately for the velocity and position control loop.

Portability Parameter is available starting with compiler 6.5.36.

Parameter Group AXB (not displayed in axis parameter fields)

Cross Reference AMPENCRES, AMPENCRPM, POSDRCT

Unit –

Range

-1	like for previous versions, i.e. feedback source of the position control loop is also in use by the velocity control loop
1	Encoder 1 is in use as the feedback source of the velocity control loop
2	Encoder 2 is in use as the feedback source of the velocity control loop

and so on …

Init Value -1

AMPENCRES - 127

Content	Resolution [qc/turn] of the encoder in use by the velocity control loop.
Description	If the velocity control loop uses another feedback source than the position control loop (i.e. AMPENCNO > -1), the resolution of the corresponding encoder has to be configured by the parameter AMPENCRES too. The value of AMPENCRES must be defined as quadcounts/turn (which is different from the unit in use by the parameter ENCODER)!
Portability	Parameter is available starting with compiler 6.5.36.
Parameter Group	AXB (not displayed in axis parameter fields)
Cross References	AMPENCNO, AMPENCRPM, POSDRCT
Unit	qc/turn
Range	1 … MLONG
Init Value	-1

AMPENCRPM - 128

Content	Maximum velocity [rpm] of the encoder in use by the velocity control loop.
Description	If the feedback of the velocity and position control loop is defined separately (by AMPENCNO > -1), then parameter AMPENCRPM must be set to maximum velocity [RPM]. It must fit to the MAXVEL of the position encoder. If AMPENCRPM is configured with 0 as default, the VELMAX value will also be used for the velocity control loop. The maximum velocity of the velocity control loop corresponds to the maximum output value (= 100% duty cycle) of the power stage. This defines the actual maximum reachable velocity depending on the supply voltage of the power stage and the motor characteristics. For correct functionality the parameter value of AMPENCRPM (and VELMAX) must be adjust depending on these “hardware” values.
Portability	Parameter is available starting with compiler 6.5.36.
Parameter Group	AXB (not displayed in axis parameter fields)
Cross Reference	AMPENCNO, AMPENCRES, POSDRCT
Unit	RPM
Range	1 … 65535
Init Value	0

Global Parameters in Detail

CANNR - 100

Content	CAN node ID
Description	If a CAN component is used in the program this parameter, CANNR, determines the CAN node ID of the control unit. M1: The CAN number is evaluated from hex rotating switch. All other controller: The CAN number is defined while the interface settings. If the CANNR is set to 9999, no standard CAN objects are created. Standard CAN objects are necessary for the communication with the APOSS program when using the commands OUTMSG, INMSG, and INGLB.
Portability	Only M1 and other new controller.
Parameter Group	Global parameters GL
Unit	–
Value Range	0 … 2048
Init Value	0

CANBAUD - 101

Content CAN baud rate

Description CANBAUD defines the baud rate for CAN communication (MACS3 for master and slave BUS).

The baud rate can be set with Hex switch (only MACS3) or with APOSS command. Follow these instruction in case of using the Hex switch (or look up hardware manual):

- Hex switches to 00
- Power On
- Segment display shows "CAN"
- Within 30 seconds you have to choose baud rate for master by Hex switch (H) and for slave by Hex switch (L):

  1 = 10 kBaud
  2 = 20 kBaud
  3 = etc. see below

  If both do not change from zero, then 125 kBaud will be selected for both.
- After 30 s the baud rate is frozen and then display shows "Id". You have another 30 s time to select now your CAN-ID for the master bus. You can change this also later and power off and on again.

Of course baud rate can also be set by using the APOSS command SET and the parameter CANBAUD. This parameter has the following range:

n = MasterBaud * 10 + SlaveBaud
MasterBaud = 0 … 8 (Baud rate 5 kBaud .. 1 Mbaud)
SlaveBaud = 0 … 8 (Baud rate 5 kBaud .. 1 Mbaud)

Sample SET CANBAUD 57 // Set MasterCAN to 125 kBaud and SlaveCAN to 500 kBaud

Portability Only M1 and other new controller, Master and Slave only with MACS3.

Parameter Group Global parameters GL

Unit –

Value Range 0 … 8

0 = 5 kBaud
1 = 10 kBaud
2 = 20 kBaud
3 = 50 kBaud
4 = 100 kBaud
5 = 125 kBaud
6 = 250 kBaud
7 = 500 kBaud
8 = 1000 kBaud

Init Value 5 = Standard CAN-open modules or MACS2
55 = MACS3

PRGPAR - 102

Content Activated program number

Description With PRGPAR it is possible to set which program should be started after the conclusion of a program executed via **Autostart** (auto identification). This parameter can also be changed and stored with other programs or via the display.

If the program parameters are set within a program with the command SET, for example SET PRGPAR 5 then after the program is run program no. 5 will be started. Thus it is possible to start other programs from an APOSS program and link several programs together.

If no program number is activated and no input for the program start I_PRGSTART (103) is set, then the program with auto identification will be started.

! ! ! If no **Autostart** program is defined then it is not possible to start a program via PRGPAR (102); this always requires a terminated autostart program.

Portability Only M1 and other new controller;
parameter value +1000 with control version 6.1.14 onwards.

Parameter Group Global parameters GL

Unit –

Value Range –1 … 127 (optionally + 1000)

–1	=	program number is not activated, i.e. no program to start after autoexec
0…127	=	activated program number is started after power up (and autoexec)
0…127 +1000	=	activated program number is started, but at power on the motor is turned off (motor off)
–1 + 1000	=	(999) same as –1, but with motor disabled after power up

Init Value –1

Example

```
SET PRGPAR 5        // program number 5 is started immediately after
                    // power up and after execution of autoexec

SET PRGPAR 1005     // program number 5 is started,
                    // but at power on the motor is off
```

I_PRGSTART - 103

Content Defines input to be used to define type of program start.

Description If the input for I_PRGSTART ≠ 0, the autostart program is executed at first and the program waits until input I_PRGSTART will be active. This is then evaluated relevant to the program selection I_PRGCHOICE (104) in order to determine the number of the program to be run.

If no input for the program start I_PRGSTART is set, then the program with auto identification will be started.

Portability Only M1 and other new controller.

Parameter Group Global parameters GL

Unit –

Value Range 0 … 8

Init Value 0

I_PRGCHOICE - 104

Content Defines input to be used for a program selection.

Description If I_PRGCHOICE > 0 then this parameter indicates the input number starting at which the inputs for the ***Program Selection*** are used. This includes all numbers up to I_PRGSTART (103).

Example If I_PRGCHOICE = 3 and I_PRGSTART = 7, then upon activation of input 7 the inputs 3, 4, 5, and 6 will be evaluated binary and the result will be used as a program number.

Input	Level	Binary value
4	high	2
5	high	2^2
6	low	0
=> program to be started:		6

Thus it is possible to choose between a maximum of 90 programs, identified with the numerals 0 to 89.

Portability Only M1 and other new controller.

Parameter Group Global parameters GL

Unit –

Value Range 0 … 8

Init Value 0

I_BREAK - 105

Content	Input for abort
Description	I_BREAK defines digital input_n to be used to react and abort a program immediately, when activated. Such a program can be continued with CONTINUE. It is possible to react to a positive or negative edge, using a negative or positive number.
Portability	Only M1 and other new controller.
Parameter Group	Global parameters GL
Unit	–
Value Range	1 … 8 = Reaction to a positive edge on input 1 … 8 0 = no function –1 … -8 = Reaction to a negative edge on input 1 … 8
Init Value	0

I_CONTINUE - 106

Content	Continue program
Description	I_CONTINUE defines which input is used to continue aborted programs. It is possible to react to a positive or negative edge, using a negative or positive number.
Portability	Only M1 and other new controller.
Parameter Group	Global parameters GL
Unit	–
Value Range	1 … 8 = Reaction to a positive edge on input 1 … 8 0 = no function –1 … -8 = Reaction to a negative edge on input 1 … 8
Init Value	0

I_ERRCLR - 107

Content	Clear error
Description	I_ERRCLR defines which input is used to clear an error. It is possible to react to a positive or negative edge, using a negative or positive number.
Portability	Only M1 and other new controller.
Parameter Group	Global parameters GL
Unit	–
Value Range	1 … 8 = Reaction to a positive edge on input 1 … 8 0 = no function –1 … -8 = Reaction to a negative edge on input 1 … 8
Init Value	0

O_ERROR - 108

Content	Output for error
Description	The output defined by O_ERROR is set when an error has occurred. When the error is cleared this output is re-set. A positive number means that the output is high (24 V) when the function is active. A negative number means that the output is low (0 V) when the function is active.
! ! !	The setting of the O_ERROR parameter does not influence the use of the OUT and OUTB commands. With these commands it is also possible to change the outputs which have pre-defined functions.
Portability	Only M1 and other new controller.
Parameter Group	Global parameters GL
Unit	–
Value Range	–8 … 8
Init Value	0

AINFTIME - 109

Content Defines the filter time for analog input.

Description This global parameter is used to define the filter time for the analog input (MACS3). If this value is > 0, then the analogue input will be read every ms and a PT Filter is used to smoothen the input. Internally it is calculated as follows

filtval = (((analogue IN 1) * 1000) – filtval) / AINFTIME + filtval

When the command INAD 1 is used, we return filtval / 1000. (This is done for a better resolution).

Portability Parameter is available starting with firmware 6.2.35.

Parameter Group Global parameters GL

Unit –

Value Range 0 … MLONG

Init Value 0

CANSYNCTIMER - 114

Content Defines the cycle time for sending SYNC telegrams on the CAN bus.

Description The global parameter CANSYNCTIMER defines the period of time in between sending cyclic SYNC telegrams. SYNC telegrams are used as a trigger to request updated information (PDO) by CAN bus devices.

In case of CANopen servo amplifiers, frequency converters, or encoders (see parameters DRIVETYPE, ENCODERTYPE, MENCODERTYPE) data acquisition typically has to take place within each the position control cycle. The cycle time for the required SYNC telegram is defined by the parameter CANSYNCTIMER.

If the encoder information is provided through the bus, it must be made sure, that new encoder information is present for each cycle of the position control loop. This means that the CANSYNCTIMER must be set to a value corresponding to the TIMER setting of the axis and the performance of the bus-coupled power stage devices. The Timer should be like the following example (default is 1 ms):

```
SET TIMER 2 / 4
// setting the drives to appropriate transmit types (i.e. 1 and 2)
SET CANSYNCTIMER 2        // every 2 ms a SYNC is transmitted
// setting DRIVETYPE and REVERS
```

! ! ! Please note that the smallest axis-specific defined value of the parameter TIMER was in use to define the periodicity of the SYNC telegram until firmware version < 6.5.16.

Portability Parameter is available starting with firmware 6.5.16.

Parameter Group Global parameters GL

Cross Reverences DRIVETYPE, ENCODERTYPE, MENCODERTYPE, TIMER

Unit ms

Value Range 0 … MLONG

Init Value 1

VMENCTYP - 116

Content Defines the source of the virtual encoder and the encoder type, which is in use if virtual master is disabled

Description Defines the encoder type for the virtual encoder, when the virtual function is exit with a PULSACC 0 command. Furthermore the command velocity of another axis can be selected for the virtual master.

VMENCTYP = ax * 100 + enctype

where

ax = 0 means the value, which is given by PULSVEL commands is read out to the encoder output (see PULSVEL, PULSACC)
ax = 1.. n means the value is given by the command velocity of the corresponding axis.

enctype defines the ENCODERTYPE, which will be in use, if the virtual encoder is disabled, i.e. enctype corresponds to the values of the parameter ENCODERTYPE (0 = incremental, 1 = absolute, 5 = sinus cosinus).

If the command velocity of an axis is set by VMENCTYP, then PULSACC must also be set to the desired value to start the function (this value acts limitative too) as well as PULSVEL must be set. However the value of PULSVEL is irrelevant and can be set to 0. Only the command access is necessary to start the function.

Firmware 6.7.71 (and above) offers two additional possibilities to configure the virtual master:

If PULSACC is set to -1, the input of the virtual master is provided by a DDS defined by a SETSYS 15 command. In that case PULSACC and PULSVEL are also obsolete.

Furthermore there is an additional configuration possibility for users having their own or extended FPGA software installed: An offset of 100000 can be added to the parameter value of VMENCTYP. In that case the output of the virtual master is not transferred to the encoder output (e.g. MACS4 ENC4) but to an internal dead end node. This dead end node might be accessed by special FPGA software and the signal might be used for further processing. The input of the virtual master is still defined by the above mechanism. So it would be possible to use it together with a multiple of 100 (-> command velocity of a axis) or with PULSACC -1.

! ! ! Attention: Any change of VMENCTYP will only be processed with the next PULSVEL command.

Portability Parameter is available starting with firmware 6.6.18;
starting with firmware 6.7.71 new SYNCTYPE value to buffer negative master movements.

Parameter Group Global parameters GL

Cross Reference VMENCMTYP, VMENCRES, PULSVEL, PULSACC

Unit –

Range 0 … MLONG

0	=	the value given by the PULSVEL command, is read out to the encoder output
1 …	=	n * 100 + encodertype Usage of command velocity of axis n for virtual master and configuration of encoder type in case of disabled virtual master
+100000	=	Deactivation of the encoder output and usage of special FPGA-Code to process the virtual master signal internally.

Init Value 0

VMENCRES - 117

Content Virtual master encoder resolution.

Description Encoder resolution of virtual master used for index distances.

Sample: To simulate a 500 counts encoder, the parameter must be set as follows:

```
SET VMENCRES 2000      // Create every 2000 qc an index pulse
PULSACC 1000           // Accelerate within a second to velocity of 1000 qc/sec
PULSVEL 1000           // Run with velocity of 1000 qc/sec.
```

Portability Parameter is available starting with firmware 6.6.18.

Parameter Group Global parameters GL

Cross Reference VMENCTYP, VMENCMTYP

Unit qc

Range 0 … 65535

Init Value 500

VMENCMTYP - 118

Content Virtual master encoder marker type.

Description Defines the index type for the virtual master, analog to the settings for SYNCMTYP.

That means,

0 = Index pulse (encoder Z negative edge)
2 = external marker (positive edge)
3 = external marker (negative edge)

decades = nm
n = digital input number (1 … 9)
m = master marker type (0 … 3)

Portability Parameter is available starting with firmware 6.6.18.

Parameter Group Global parameters GL

Cross Reference VMENCTYP, VMENCRES

Unit –

Range 0 … MLONG

0 = zero pulse

Init Value 0