Operating a MC V3.0 EtherCAT driver as a CODESYS SoftMotion drive

Summary

This Product Application Note describes how to use a FAULHABER MC V3.0 EtherCAT driver connected to a CODESYS SoftMotion axis. A SoftMotion axis can be used to generate the complete motion trajectory directly within the master, different from a drive operated in Profile Position Mode where the built-in profile-generator does this job. Here the general setup of the CODESYS configuration is explained for a single axis system as well as the expected settings on the drive side.

Applies To

Any FAULHABER MC V3.0 EtherCAT drive

MCS32xx BX4 ET, MCS3274 BP4 ET

MC 5004 P ET, MC 5005 S ET, MC 5010 S ET

Description

Before a drive can be configured to be used as a SoftMotion drive, a CODESYS project as to be created and at least one CiA402 ET or CANopen[1] drive is to be connected to the system. In this example again a RaspberryPi is used as the CODESYS master whereas a FAULHABER MC 5005 S ET drive is used as the servo to be commanded.

General information about how to install the CODESYS environment on a RaspberryPi and to create a first project driving a FAULHABER MC V3.0 ET drive can be found in our Product Application Note 164 at our support page www.faulhaber.com/en/support/drive-electronics.

Prerequisites

Before starting to set-up your CODESYS environment please use the MotionManager to configure your drive:

- Select the motor and feedback-system using “Select motor” tool
- Configure the load inertia using the “Configure controller” tools
- Tune the control-loops
- Verify the dynamic parameters like acceleration and deceleration which can be safely used in your application by operating the drive using the MotionManager MotionCockpit
- Configure whatever additional I/Os might be used e.g. for a drive-based homing sequence
- Save your settings in the drive and store a copy of it at your PC

CODESYS project configuration

The general setup is the same as detailed in AppNote 164:

- Create a new project

[1] Use of a CiA402 CANopen servo-drive has not been tested. Please note SoftMotion will require high update-rates of the intermediate demand-values anyway, so ET might be the better choice.

- Select the correct target environment – here it is a RaspberryPi
- Connect to the target device
- Add an EtherCAT Master to your project and configure the network interface to be used on your target device
- Add the FALUHABER MC as a subsystem to the EtherCAT master and check the communication settings
- Start thinking about the process-image, which is the collection of parameters to be exchanged cyclically when the system is running.

After these steps we end up with the PDO mappings page of the drive configuration. But before configuring the PDOs we add the SoftMotion axis.

Add a SoftMotion Axis to the drive

The SoftMotion CiA 402 Axis is added to the MC 5005 using the context menu of the driver in the project tree, Figure 1.

Whenever the drive is identified by CODESYS to be a CiA 402 servo-drive this option is visible in the context menu.

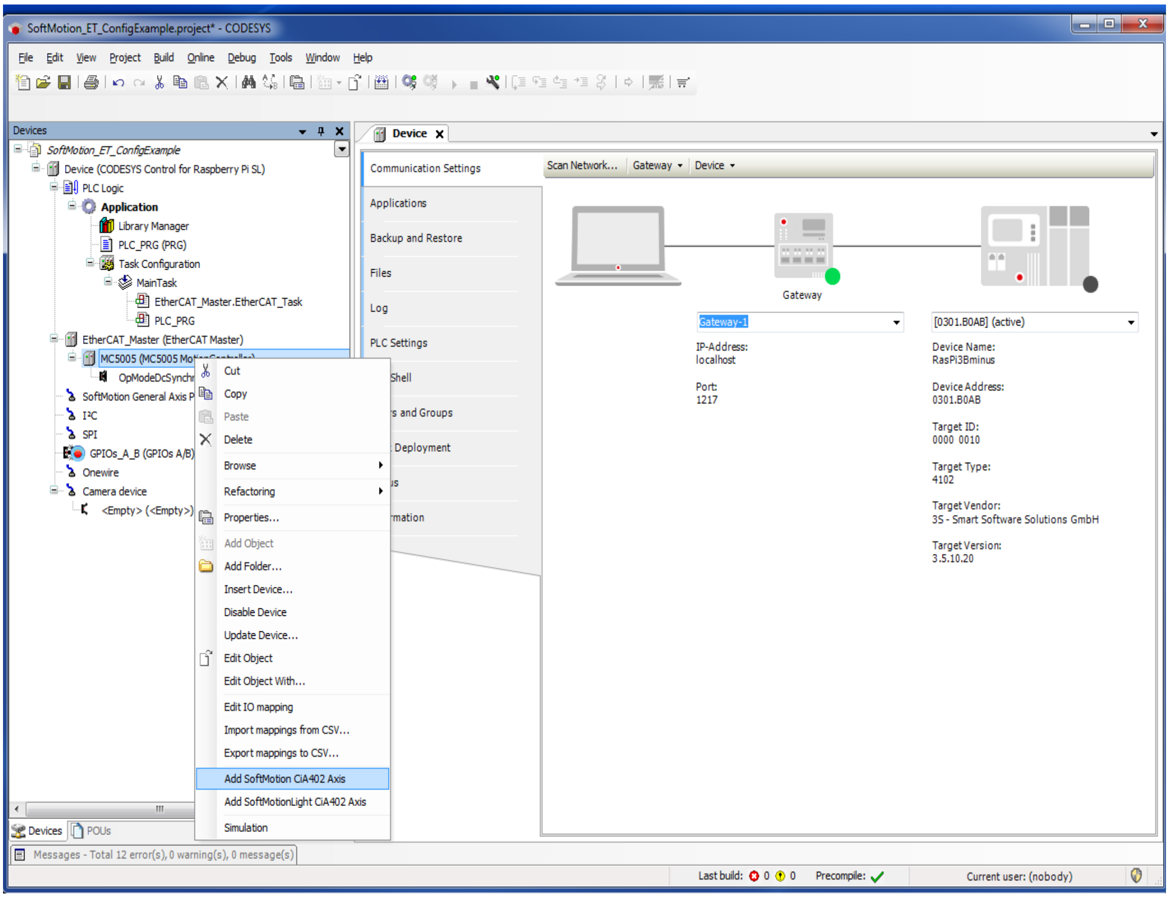


Figure 1 Add a CiA402 SoftMotion Axis to the MC 5005 S ET

FAUHABER drivers have not been tested at CODESYS so afterwards a warning will pop up – which can be ignored.

Adjust the settings of the SoftMotion drive to your axis

In the project tree the SoftMotion axis is organized as a sub-system of the MC 5005. There are two configuration tabs of the SoftMotion axis which are to be considered.

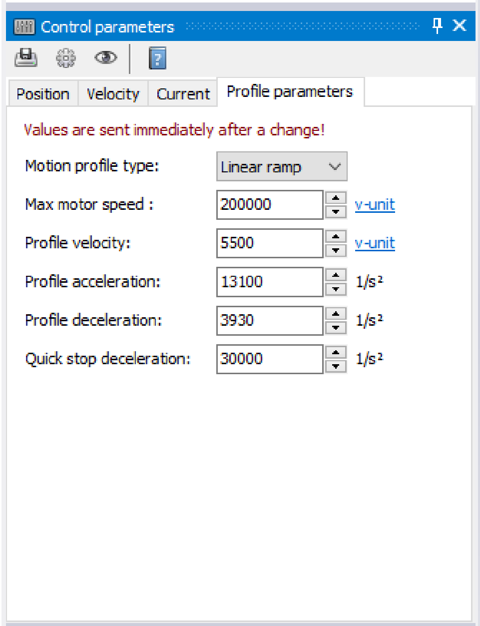
Scaling of the motion

As for the scaling there are two tabs where the necessary information must be entered. Most important is the correct scaling of the drive position or any transmission or gearhead being used. This information is entered on top of the Scaling/Mapping tab in Figure 3.

What is required	How to be set
Increments per motor turn **What is the encoder resolution in increments per motor turn?**	This is the resolution that has been set in the drive. Default setting would be the native resolution of the motor feedback. Changing the settings of the Factor Group inside the MC is not recommended, the parameters of the transmission can be entered at the SoftMotion axis. The value can be entered either hex (16#10000) or decimal. For a typical 12-bit feedback as e.g. our linear hall sensors we would enter 4096 increments per turn.
Gear reduction ratio **How many motors turns do we have per turn of the output shaft of the gearhead?**	If the gearhead or transmission has an integer reduction ratio it can be entered directly. When there is a non-integer reduction ratio like 63/17 = 3.705… for a 4:1 22F GH enter the numerator – the bigger number – at the motor side and the denominator – the smaller number – on the gear output side.
Units to be used **How do you want to count the position within your application? Default is 1.0 per one output turn.**	If you got a linear system you might want to scale the position to be used by the pitch of a screw. So e.g. for a 1mm pitch a setting of 1 will have you positions in mm because at one turn of the shaft the nut will move by 1mm. If you got a rotating system a scaling in ° might also be a good option. So, use 360 units in your application per turn to have all your positions being in °.

Next would be the dynamic limits of the drive (Figure 2), which is the maximum velocity [u/s], acceleration [u/s²], deceleration [u/s²] and jerk [u/s³].

These will be used to limit the values used in the application. The values can easily be taken from the drive configuration within the MotionManager, after these have been tested.

	Velocity [u/s] Here this is the maximum speed in turns per s. Can be calculated out of the maximum profile velocity in the control parameters window of the MotionManager. $\text{Velocity [u/s]} = \frac{Profile\ velocity}{60}$
	Acceleration [u/s²] Enter the maximum acceleration which still resulted in a satisfying behavior of your axis. Maximum would be the Profile acceleration as in the control parameters window of the MotionManger.
	Deceleration [u/s²] Enter the maximum deceleration which still resulted in a satisfying behavior of your axis. Maximum would be the Profile acceleration as in the control parameters window of the MotionManger.
	Jerk [u/s³] The FAULHABER drivers don't limit the jerk so there is no value that can be transferred 1:1. Depends on the stiffness of the application and on the selected motion profile. Numbers are usually higher than the ones for acceleration as the jerk is the derivative of the acceleration.

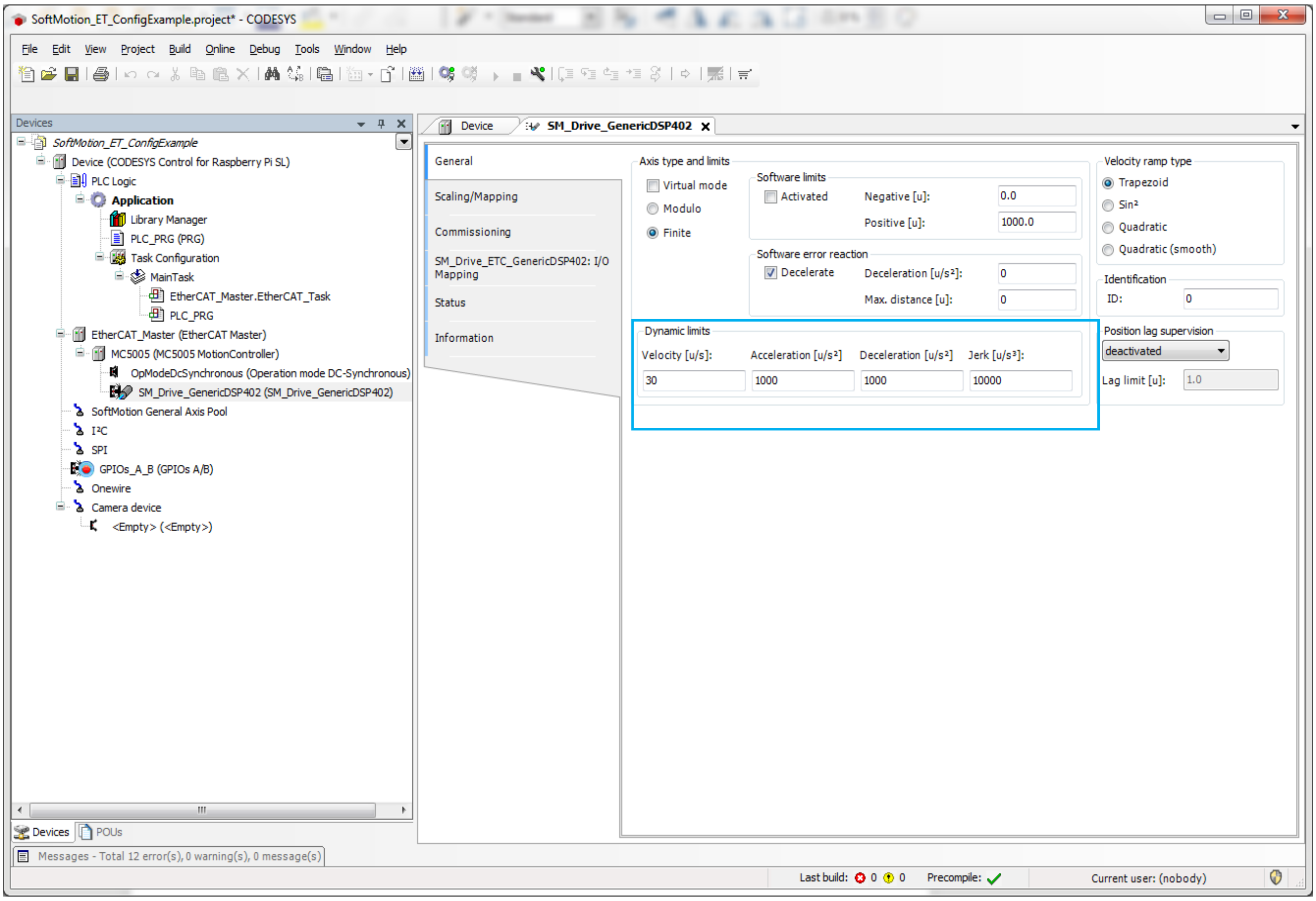


Figure 2 General setting tab for the SoftMotion axis

PDO Mapping

After the Scaling, it is the Mapping of the process-image that must be considered. The Scaling/Mapping tab of the SoftMotion drive (Figure 3) gives an overview which parameters of a CiA 402 CANopen servo-drive would be used, if available. Plus, there is a checkbox to facilitate automated mapping. This mapping here applies for the mapping between the process image created by the PDOs of the drive and the connected SoftMotion axis only. It makes use of whatever is available in the process image created by the PDOs configured but it will not generate the PDOs configuration. That is still to be done and selected by hand depending on the application.

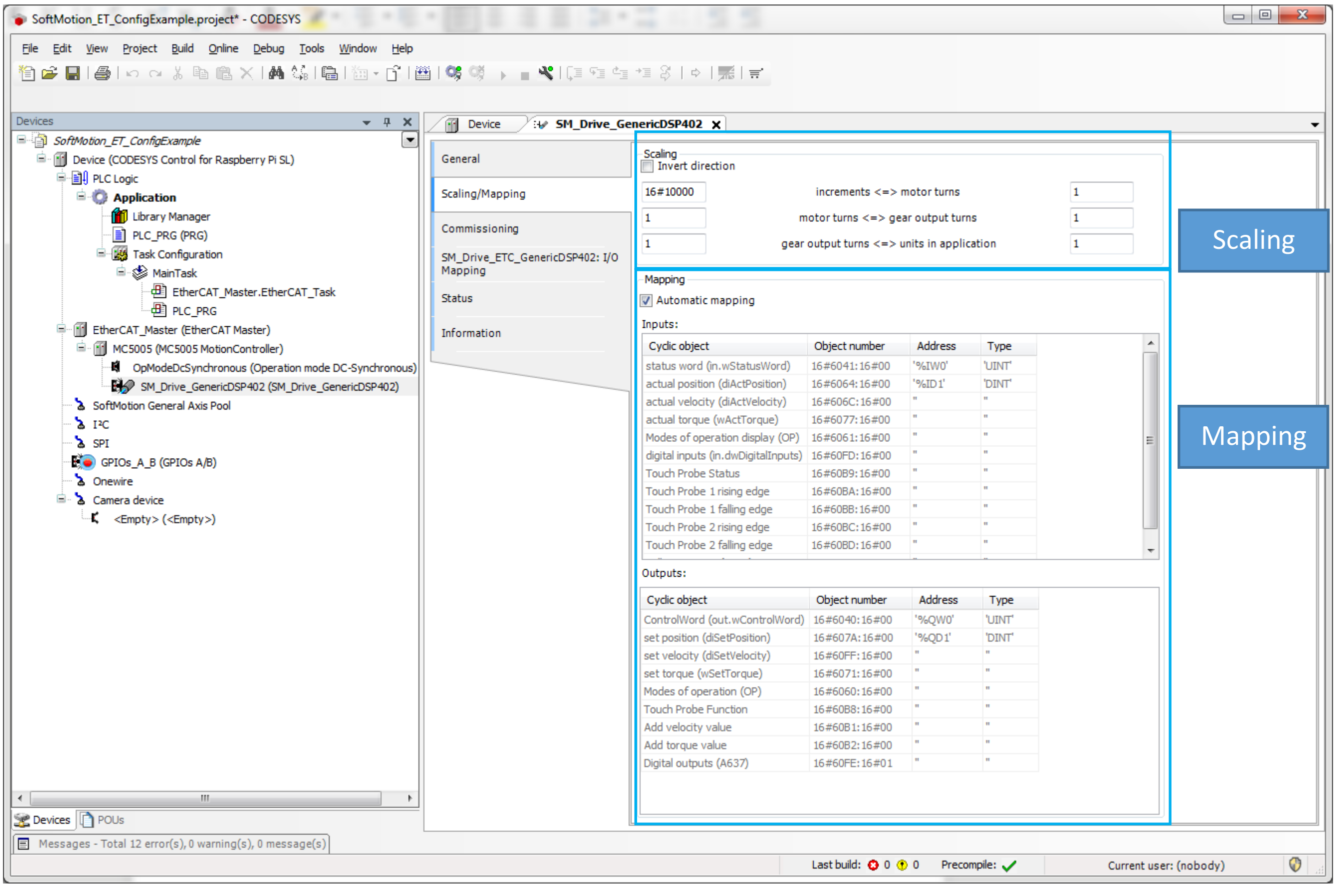


Figure 3 Scalings and mappings of the axis

So, what we must do is, try to make the necessary parameters available.
We do this, by creating an appropriate PDO-mapping for the MC 5005 used here and then un-check and re-check the automatic mapping checkbox.

The procedure for the PDO mapping itself is the very same as in Product Application Note 164.

First enable the expert settings for the MC on the General Settings Tab (Figure 4).

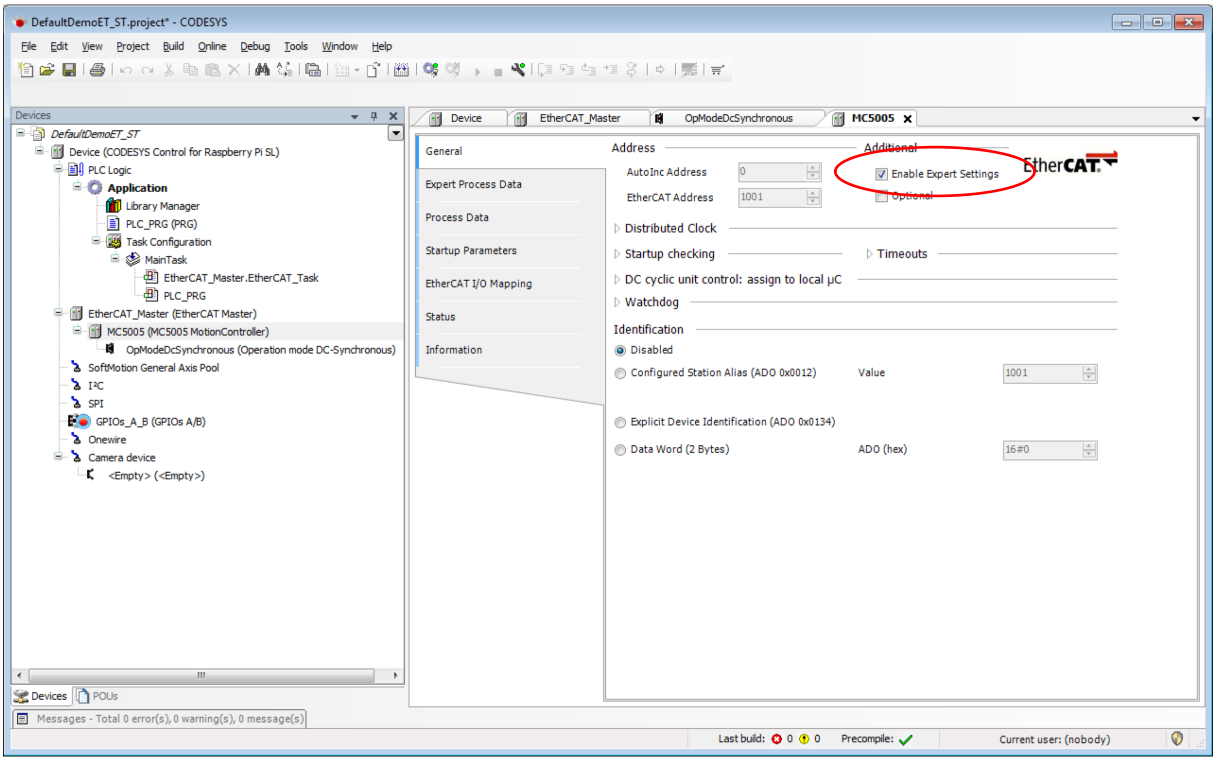


Figure 4 General settings of the MC 5005 S ET

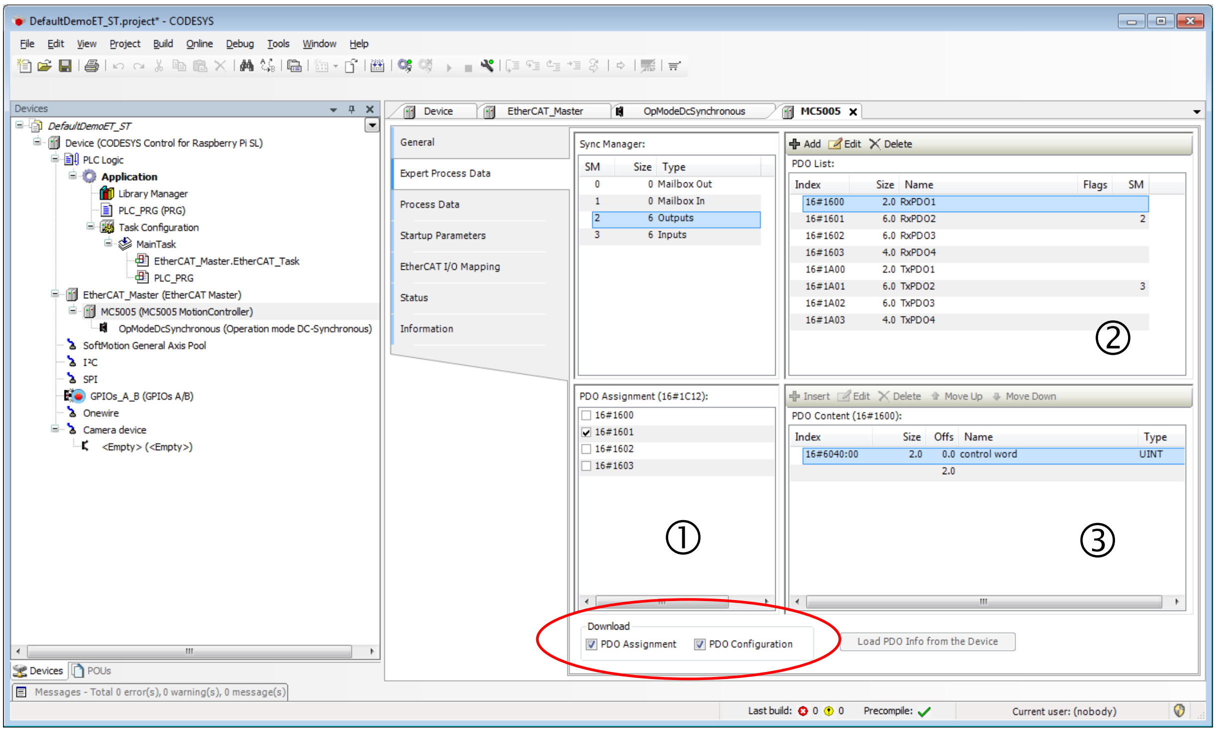


Figure 5 Expert settings for the PDO mapping

Next modify and add the PDOs on the Expert Process Data Tab (Figure 5):

①: activate the number of PDOs required. A maximum of 4 parameters can be mapped into a single PDO. In an ET system there is no limitation in data length, different from a CANopen system.

②: select the PDO to be edited

③: edit the contents of the selected PDO by removing unwanted objects and adding whatever is desired.

Finally make sure to have the checkboxes to download the PDO Assignment and the PDO configuration activated.

Steps how to configure the PDO mapping can again be found in the Product Application Note 164.

To have CODESYS re-check whether there are new useful parameters for a SoftMotion axis, un-check and re-check the automatic mapping checkbox in Figure 3.

Most of the parameters used by the SoftMotion axis are available at the FAULHABER drive. Whether to map them or not depends on the application. If no torque mode or no speed mode is used, then mapping these target values is not necessary of course. The respective offsets however are used for the cyclic modes like CSP, nevertheless.
A mapping of the drive could therefor look like:

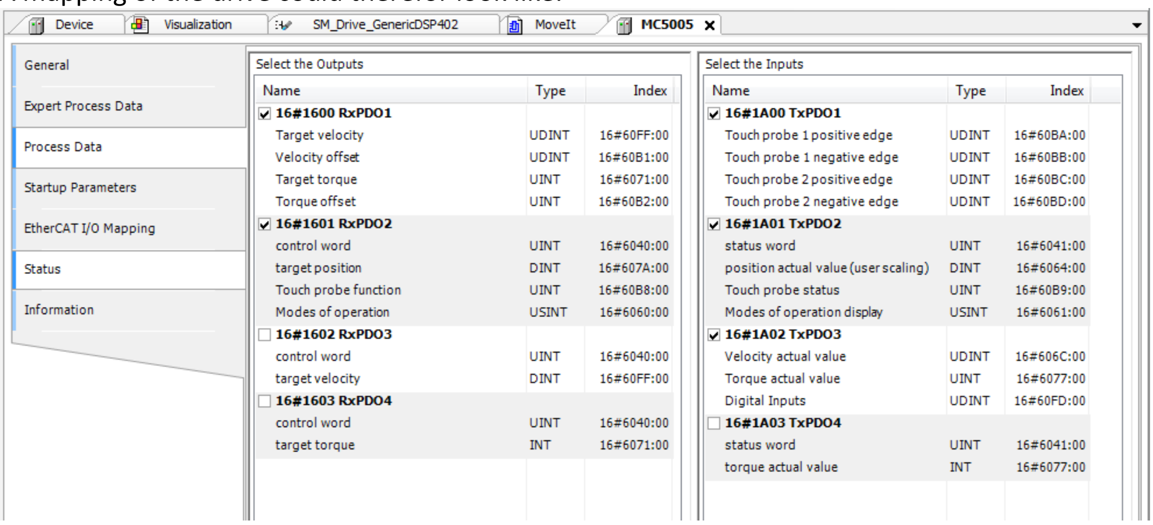


Figure 6 PDO-Mapping for a servo-drive linked to a SoftMotion axis

The SoftMotion axis will not only care for the target-values but also set the operating mode, use the feed-forward offsets of the cyclic modes and seems to be able to use the touch-probe function built into these drives.
Here two RxPDOs and three TxPDOs are used. There are no requirements in how to assign the objects to the PDOs as the CSP mode doesn't require any special sequence.

From Master to drive (RxPDO)		From drive to Master (TxPDO)	
0x60FF.00	Target Velocity	**0x60BA.00**	Touch probe 1 positive edge
0x60B1.00	Velocity Offset	**0x60BB.00**	Touch probe 1 negative edge
0x6071.00	Target Torque (if used)	**0x60BC.00**	Touch probe 2 positive edge
0x60B2.00	Torque offset	**0x60BD.00**	Touch probe 2 negative edge
0x6040.00	Control word	**0x6041.00**	Status word
0x607A.00	Target position	**0x6064.00**	Position actual value
0x60B8.00	Touch probe function	**0x60B9.00**	Touch probe status
0x6060.00	Modes of operation	**0x6061.00**	Modes of operation display
		0x606C.00	Velocity actual value
		0x6077.00	Torque actual value
		0x60FD.00	Digital Inputs

Special settings in the MotionController

Scaling of the drive velocity

The scalings in Figure 2 and Figure 3 used the drive position in increments and rescaled them to whatever application specific scaling shall be used. When using the speed of the drive the scalings in Figure 2 and Figure 3 use turns/s but that's not how speed-values are expected to be scaled in the process image.

The profile generator built into the SoftMotion axis can generate not only whatever intermediate target position is to be used (position demand) but will also calculate the target speed required for this motion. This speed can be used as a feed-forward value for the control-loop in the FAULHABER MotionContoller – the velocity offset and the torque offset values are used here. The scaling of the velocity used between the drive and CODESYS has then to be in increments/s.

The scaling of the speed and the position can always be done using the factor group. When it is set to the default values, any actual- or target-position would be given in increments per turn of the motor, same scaling as received from the encoder. The speed is internally scaled to min^{-1}.

Default settings are

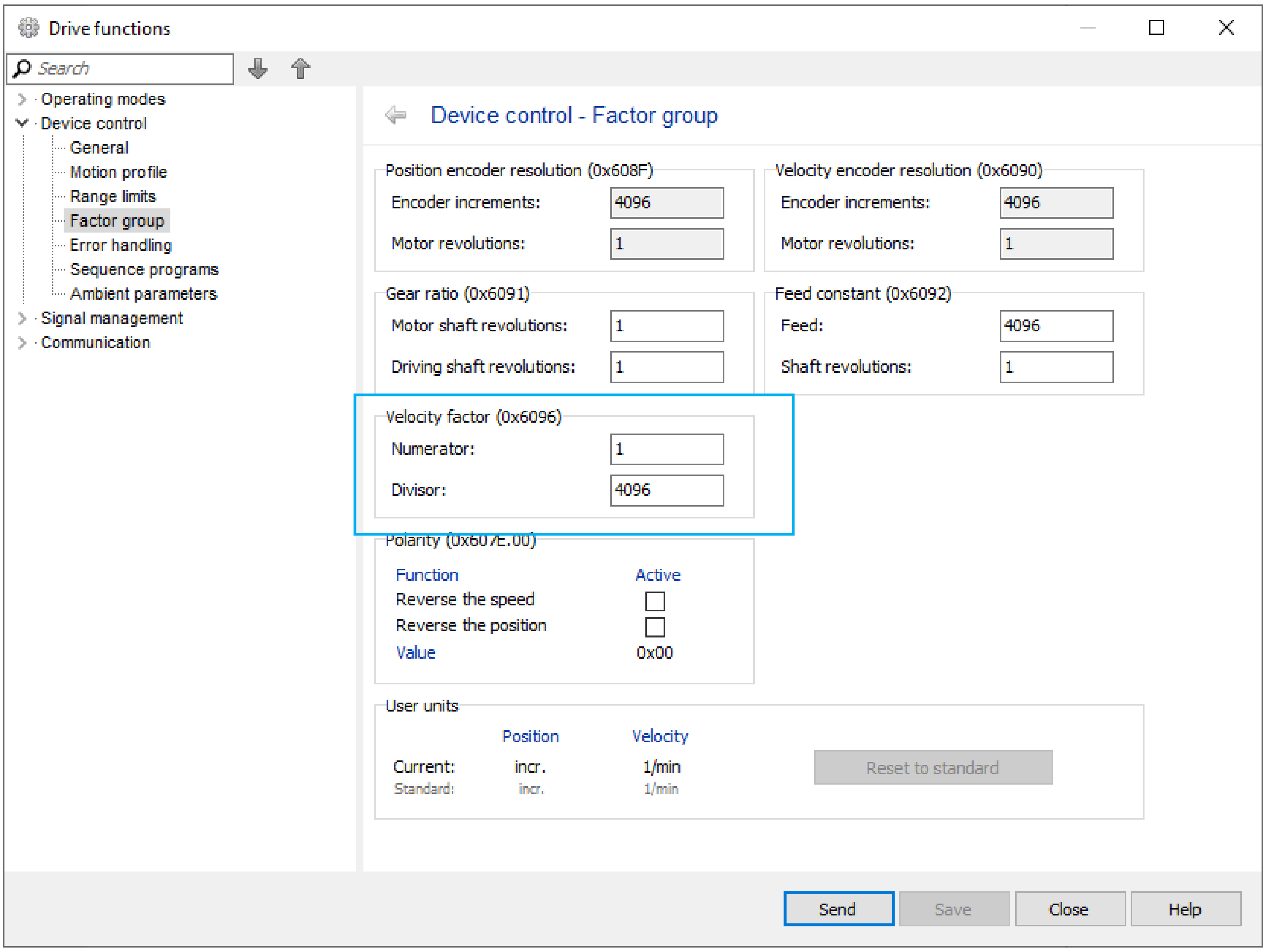


Figure 7 Default Factor Group settings of the motor 2232 BX4 used here (linear hall sensors)

We don't need to rescale the position as the scaling here has already been entered into the SoftMotion axis in Figure 3. But we need to rescale the speed. The Velocity factor (Figure 7, Figure 8) is used to do this.

The default settings calculate the speed out of an internal one by dividing it by the encoder resolution. The result is a speed in min^{-1}. So, to have it in increments/s we remove the division by the encoder resolution, but we must divide the min^{-1} value by 60 to have a 1/s value. So, the new setting for the velocity factor looks like in Figure 8.

Re-scaling of the drive speed is done on the drive side only. There is no change of configuration on the CODESYS side as it expects the feedback to be in this scaling anyway.

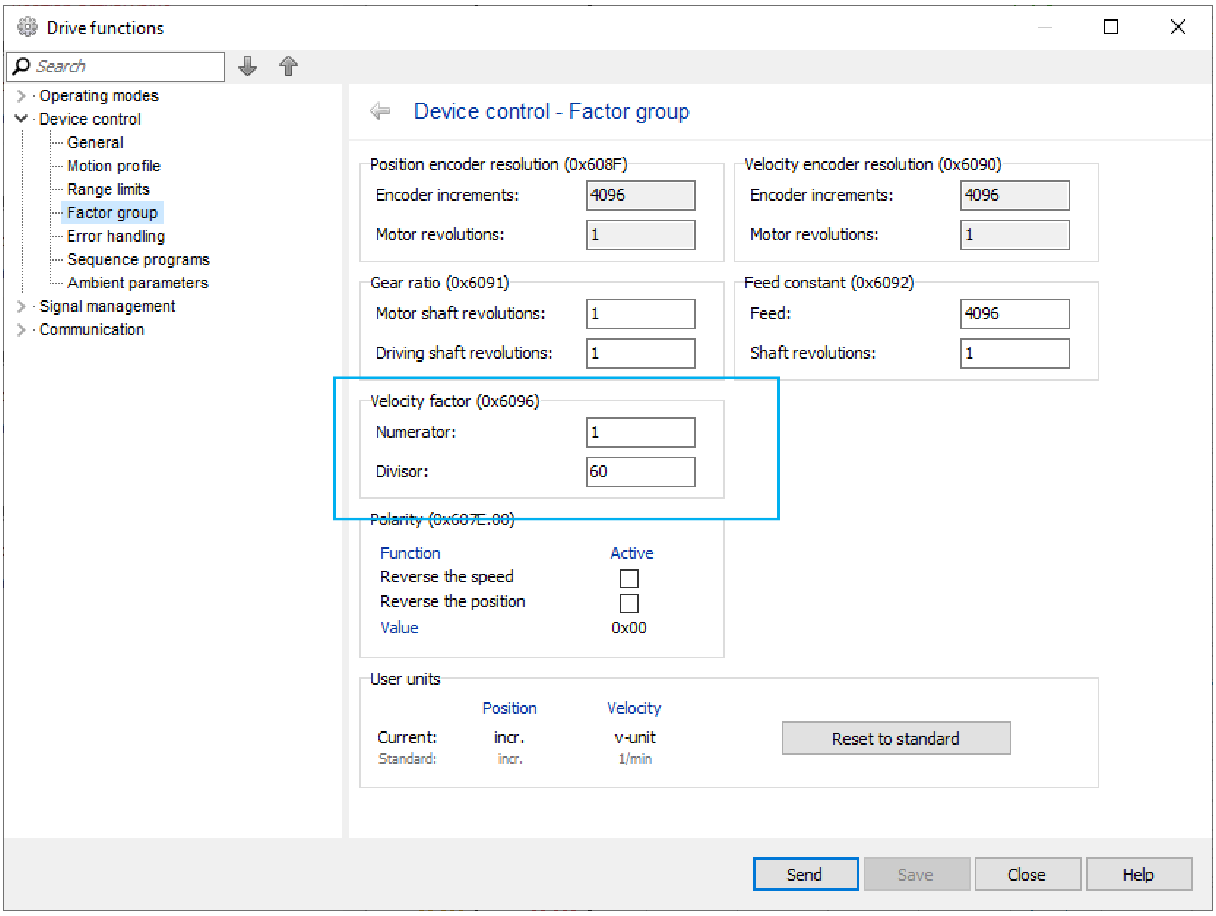


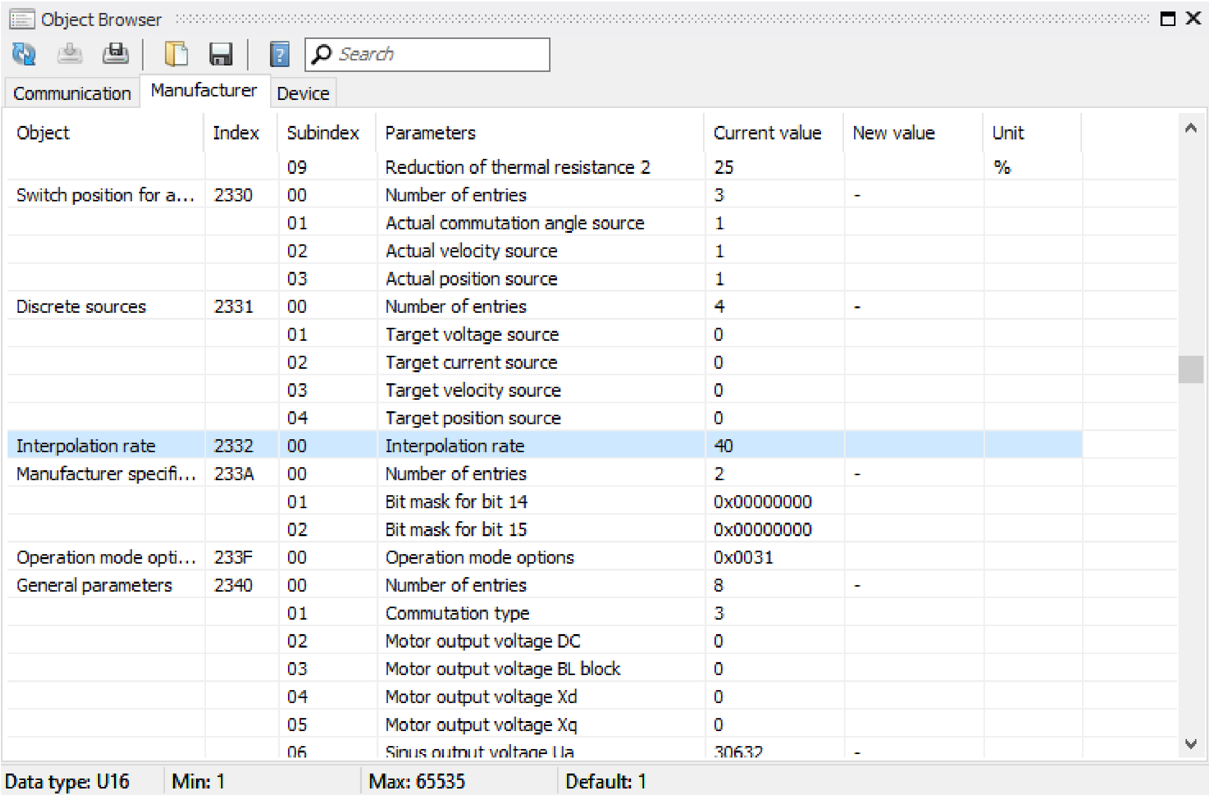
Figure 8 Modified Factor Group settings to have the speed in increments/s

Using interpolation of the cyclic target values

Even when using EtherCAT from point of view of the control loop new target values are applied step wise. Reason is the different update rate. The control loop gets updated every 100µs whereas typical EtherCAT communication cycles could be in a range of 1ms ... 2ms. Here the default of the RaspberyPi CODESYS runtime of 4ms was used. Compared to the control-loop this is an updated target position only every 40th cycle which results in steps in the target position. The behavior can be improved, if the received target values are not applied immediately but are interpolated over one communication cycle.

Starting from firmware revision L3 of the FAULHABER MotionControllers and MotionControl Systems interpolation of the received target values is available for all cyclic modes – CSP, CSV and CST.

To use it, simply enter the communication cycle in multiples of 100µs into object 0x2332.00 – interpolation rate (Figure 9).



Object Browser

Communication | Manufacturer | Device

Object	Index	Subindex	Parameters	Current value	New value	Unit
		09	Reduction of thermal resistance 2	25		%
Switch position for a…	2330	00	Number of entries	3	-	
		01	Actual commutation angle source	1		
		02	Actual velocity source	1		
		03	Actual position source	1		
Discrete sources	2331	00	Number of entries	4	-	
		01	Target voltage source	0		
		02	Target current source	0		
		03	Target velocity source	0		
		04	Target position source	0		
Interpolation rate	2332	00	Interpolation rate	40		
Manufacturer specifi…	233A	00	Number of entries	2	-	
		01	Bit mask for bit 14	0x00000000		
		02	Bit mask for bit 15	0x00000000		
Operation mode opti…	233F	00	Operation mode options	0x0031		
General parameters	2340	00	Number of entries	8	-	
		01	Commutation type	3		
		02	Motor output voltage DC	0		
		03	Motor output voltage BL block	0		
		04	Motor output voltage Xd	0		
		05	Motor output voltage Xq	0		
		06	Sinus output voltage Ua	30632	-	

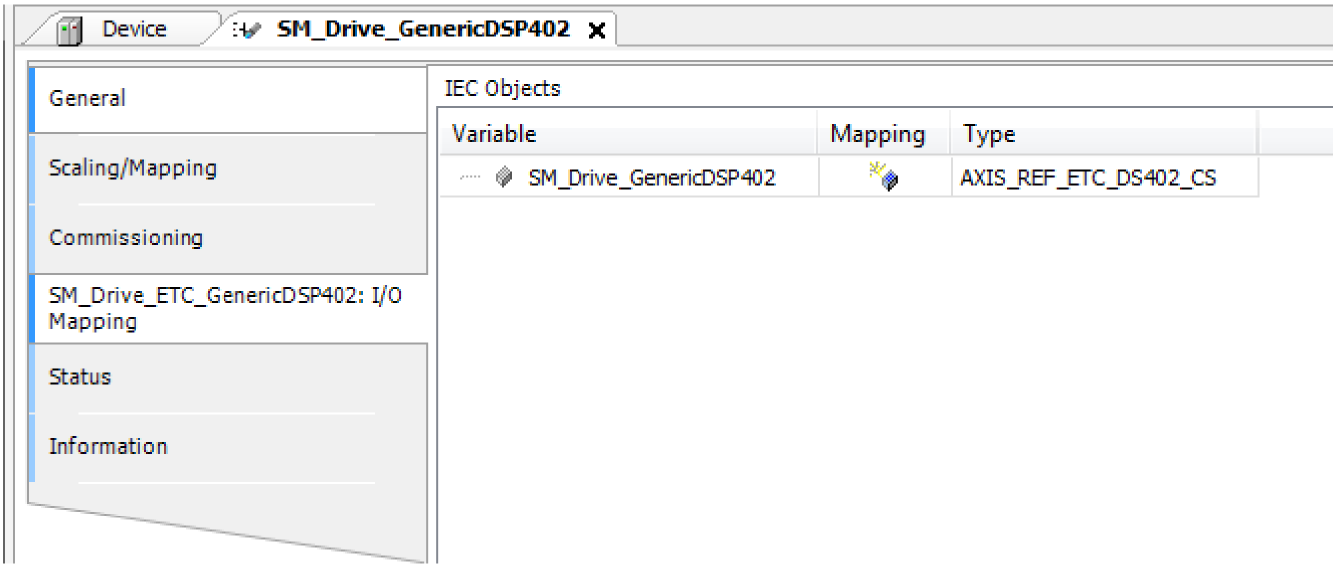
Data type: U16 | Min: 1 | Max: 65535 | Default: 1

Figure 9 set the interpolation rate for cyclic operating modes

Create your application

In Product Application Note 164 the complete application – the interaction with the drive – was implemented in structured text. When using a SoftMotion axis the standardized SoftMotion function blocks can be used to interact with the drive.

The SoftMotion FBs all use the drive data structure AXIS_REF_ETC_DS402_CS. Each SoftMotion axis creates one instance of it. This structure holds all the necessary data which are the ones exchanged with the drive plus maybe some internal ones too which we don't care about.



Here a short example using only a few of these FBs was implemented (Figure 10). There is no real automatic action implemented. The inputs are simply mapped to a visualization and are then switched manually. The used FBs were:

MC_Power

The MC_Power block is interacting with the control word and the status word to enable or disable the power-stage and control-loop. It requires to be connected to the correct instance of the axis structure – the very same which has been instanced by the SoftMotion axis. Additionally, there is a Boolean input which enables the FB to be evaluated and two Boolean inputs to start the drive and to enable the control -loop. As these FAULHABER low voltage drives don't need to be activated – not connected to the AC-grid by switching a contactor, we can use the same input variable for both.

There are some outputs for the MC_Power block which would have to be used, if this was an automated machine, but that's not required in this simple example.

MC_MoveAbsolute

The MC_MoveAbsolute block is used to move a single axis in a Point to Point manner (PTP). To do so, it takes an absolute target position – in whatever scaling we configured for the SoftMotion axis – here turns. And it takes some profile parameters as we would use them if we would use PP mode and the built-in profile generator. Here however CSP is used and automatically activated by the SoftMotion axis[2].

[2] At leat it has been activated when the Modes of Operation parameter was mapped to the SoftMotion axis. Whether it can force the OpMode using a SDO transfer has not been tested.

The block of course again needs to access the instance of the axis structure. Then there are the parameters for the motion itself and the binary “Execute” input which will start the motion at a rising edge of the input value. Here a RS-FF was additionally used to reset this execute signal once the target position has been reached.

MC_ReadActualPosition

The MC_ReadActualPosition block is used to read the actual drive position in an application specific scaling out of then instance of the axis structure. It again needs access to the axis structure and there is again an enable input to have it executed.

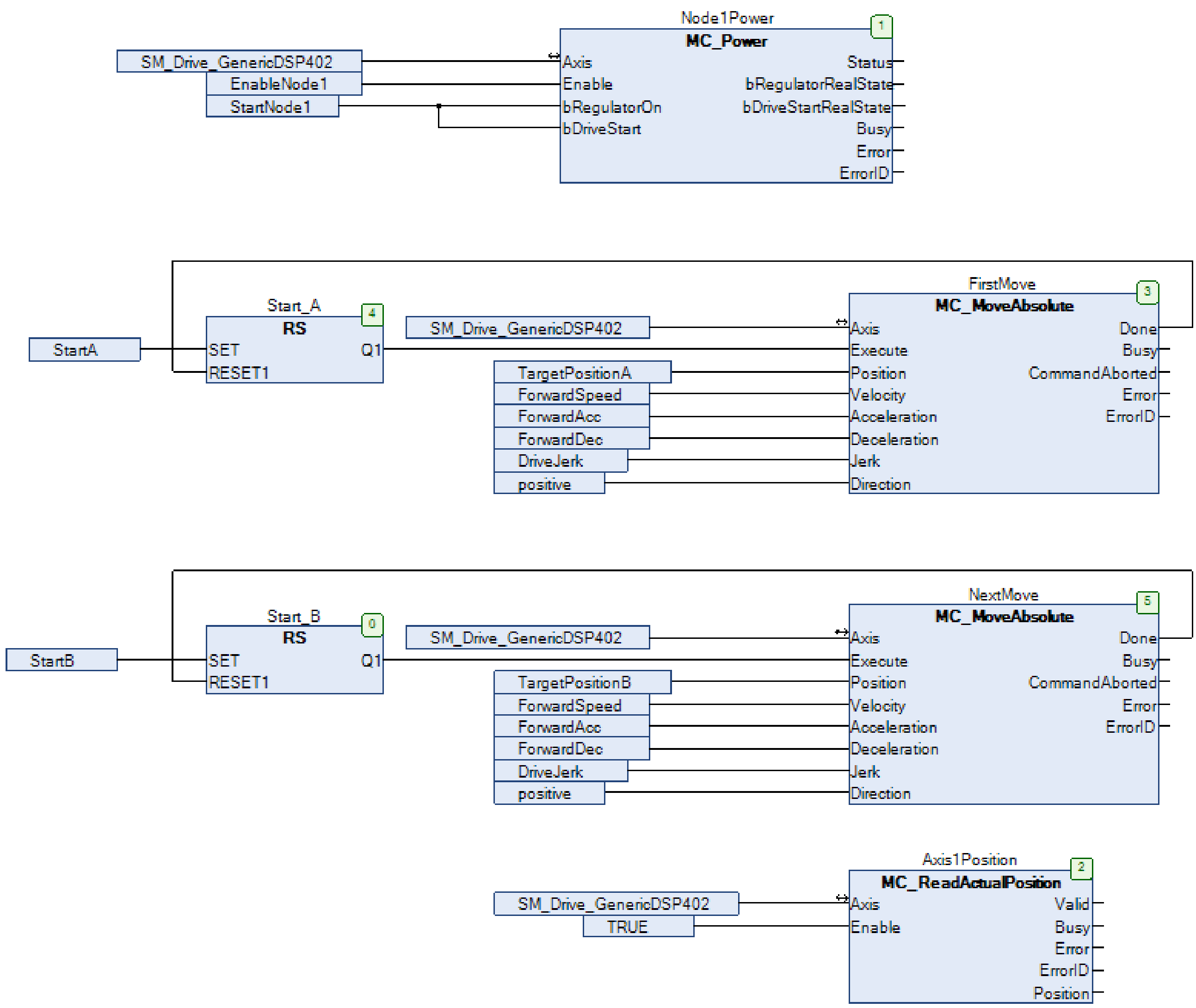


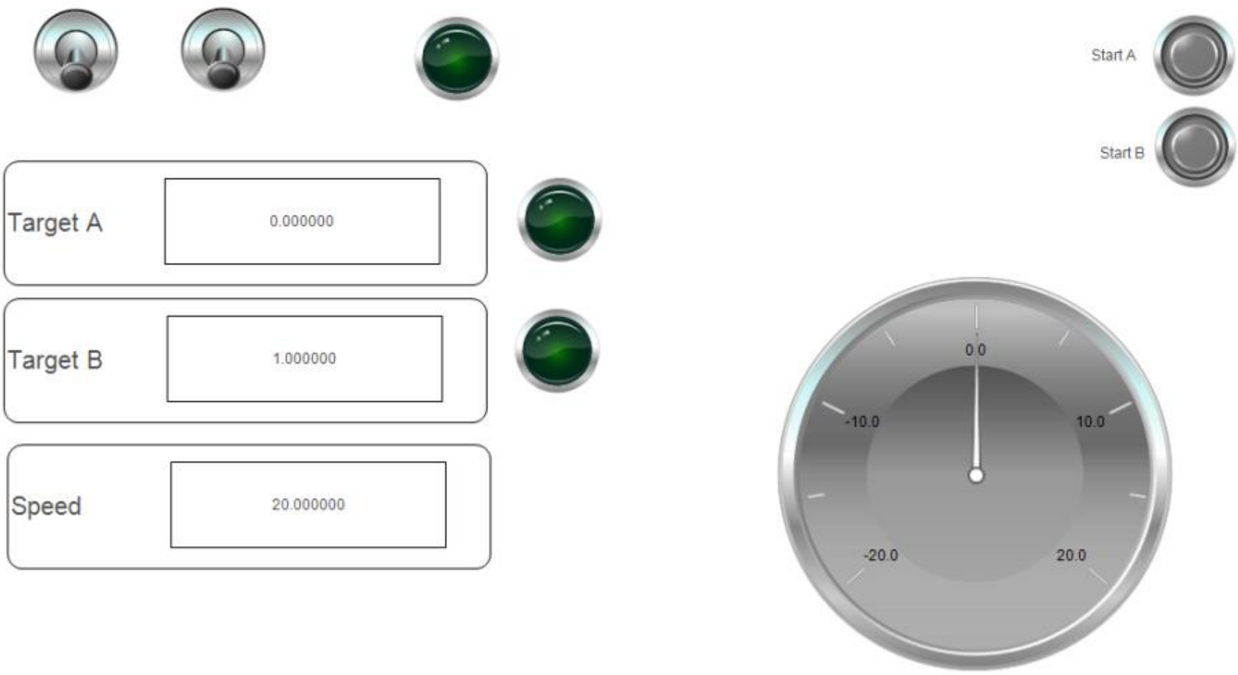
Figure 10 Example application moving between two discrete positions

Visualization

The Inputs in Figure 10 were connected to controls in a CODESYS visualization. Some of the drive outputs were displayed too. The first inputs are the two Boolean variables which input the MC_Power block. Its output is connected to a lamp.

The two absolute target positions were connected to a text-input each as well as the profile speed used by both moves. The StartA and StartB inputs finally were connected to two push-buttons on the top right of the visualization.

Please note, the positions have been scaled in turns and can be set here as floating values. The speed of the SoftMotion axis is calculated in turns/s. This example used fixed but different values for the actual acceleration and deceleration.



To interact with the drive, there is no need to be logged into it using the CODESYS engineering environment. Alternatively, you could use any web browser and log into the device using its name or its IP address:

<IP number of the target controller>:8080/webvisu.htm e.g.:

192.168.0.45:8080/webvisu.htm

Run the drive

Review the drive behavior using the MotionManager

Even when the drive is connected to a master via EtherCAT the MotionManager can be used in parallel to check the drive behavior. Any parameter which is part of the process image configured for the drive (Figure 6) will however be instantly overwritten by the PLC. Therefore, enabling or disabling the drive as well as changing the target position would be meaningless. We can however watch the drive move.

The Graphical analysis tool of the MotionManager is used to do this. The signals selected here were:

- Position actual value – which at the drive is in increments – here 4096 increments/turn
- Position demand value – the interpolated target position – same scaling as for the actual one
- Following error actual value – the following error calculated out of the upper two – same scaling
- Velocity actual value – which we had to rescale to increments/s, results in big numbers

We can see what looks like an almost perfect following to the two steps, the red-line of the actual value can't be seen it's hidden by the orange one, the demand. The detailed analysis based on the following error shows some systematic lag which is proportional to the speed and some disturbances which might be caused by whatever disturbance the linear position feedback might have contained.

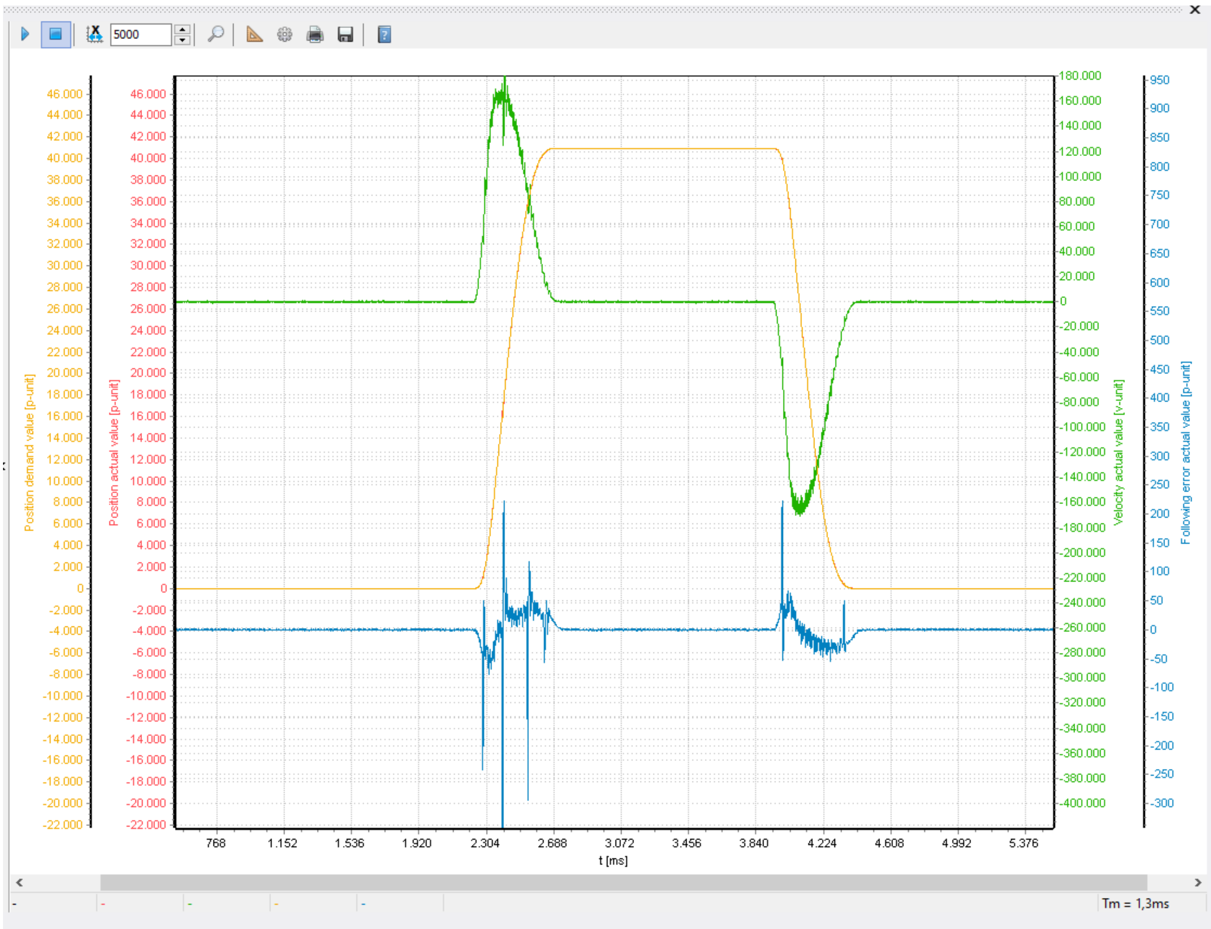
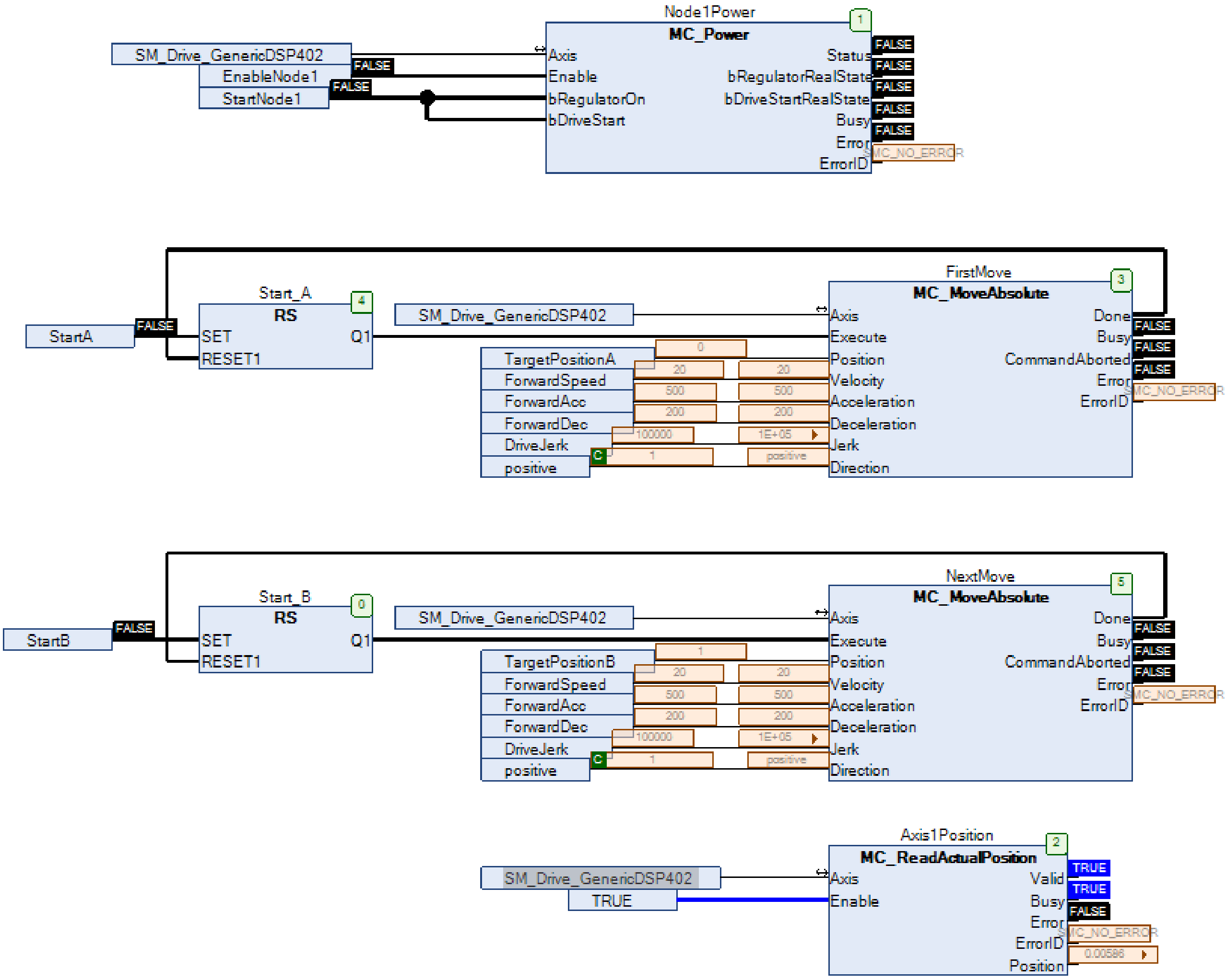


Figure 11 tracing the drive behavior using the MotionManager

Aside of the Graphical analysis tool the MotionManager can also be used in parallel to check the list of recorded errors and receive EMCY message sent by the drive.

Real-time view of the application

The second perspective on the system behavior would be the real-time view of the application within the CODESYS engineering environment. This view is available whenever you are logged into the target. Here you can see the states of the used FBs as well as the actual values of the numeric parameters. This view can be necessary when the drive behavior is somehow strange, and the interactions of whatever controlling flags shall be reviewed.



Additional Resources

FAULHABER Application Notes	
App-Note 164	How to control a FAULHABER MC V3.0 ET out of a CODESYS environment

FAULHABER manuals at https://www.faulhaber.com/en/

FAULHABER demo systems at YouTube

No liability. Owing to the non-binding character of the Application Note Dr. Fritz Faulhaber & Co. KG will not accept any liability for losses arising from its application by customers and other users. In particular, this Application Note and its use cannot give rise to any claims based on infringements of industrial property rights of third parties, due to defects or other problems as against Dr. Fritz Faulhaber GmbH & Co. KG.

Amendments to the Application Note. Dr. Fritz Faulhaber & Co. KG reserves the right to amend Application Notes. The current version of this Application Note may be obtained from Dr. Fritz Faulhaber & Co. KG by calling +49 7031 638 345 or sending an e-mail to mcsupport@faulhaber.de.