

# **Codesys and FAULHABER V3.0 EtherCAT**

# Summary

This application note describes the necessary steps to control a FAULHABER MC V 3.0 ET version using a CODESYS based PLC. The MC is connected via its EtherCAT port to the PLC.

## Applies To

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

MCS ET

## Licensing

EtherCAT is a registered trademark and patented technology, licensed by Beckhoff Automation GmbH, Germany.

## **Related FAULHABER Documents**

Document	Description
Motion Manager 6	Instruction Manual for FAULHABER Motion Manager PC software
Quick start description	Description of the first steps for commissioning and operation of FAULHABER Motion Controllers
Drive functions	Description the operating modes and functions of the drive
Com Manual EtherCAT	Description of the EtherCAT services implemented in a FAULHABER MotionController

# Description

The example shows a MC 5005 S ET operated in profile position mode controlled by a CODESYS PLC. The PLC application will start or stop the drive and does alternating position steps between two fixed absolute references while modifying the profile parameters in each step. It is created in structured text (ST).

In this example a Raspberry PI (RasPi) is used as a PLC runtime environment. As of now the runtime for the RaspPi can be downloaded free of charge out of the 3S Web shop (registration required) and can be used for training and testing purpose. Restrictions apply however. The maximum on time of the runtime is 2h for the free version. There is no limit for the number of starts however. And of course the RasPi is non hard real time environment, so a jitter in the timing is to be expected. Nevertheless this is a convenient low cost way to test the capabilities of such a combination which has been successfully tested with distributed clock synchronization down to 2ms communication cycle time.



The example has been implemented and tested for either a RasPi 2B or a 3B version.



## Figure 1 Test setup: Controller MC 5005 S ET + Motor 2250 BX4 + PLC RasPi 3B

The single Ethernet port of the RasPi has to be used for the EtherCAT interface. The access from the engineering environment installed at a PC to the runtime at the RasPi requires a second network interface. Therefore the RasPi has to be connected to the office network via either its onboard WLAN (RasPi 3B) or a USB WLAN dongle (RasPi 2B).

Following the installation of the RasPi, the CODESYS engineering environment and the CODESYS runtime are explained. These steps usually only have to be taken once at the very beginning. In a second step the creation of an application is explained and an example application is listed.



# **Software Installation**

Before stepping through the example here, some installations are required.

Configuration and first tests of the FAULHABER MotionController are done using the FAULHA-BER MotionManager. The most recent version of the MotionManager can be downloaded directly from the FAULHABER web page.

The CODESYS environment has to be downloaded from the 3S Webshop. All the mentioned components are available free of charge. A registration for the web shop is required though.

Package	Source	Size
Setup_CODESYSV35SP10Patch2.exe (CODESYS Development System)	3S Web shop	1 GB
CODESYS Control for Raspberry PI 3.5.10.20.package	3S Web shop	10 MB
OSCATBasic.package	3S Web shop	8 MB
raspberrypi_codesysv3_firststeps_xx.pdf	3S Web shop	1 MB
NOOBS	Raspberry.org	1.5 GB

Table 1 List of software components required for a CODESYS environment

## Install and configure the RasPi

Start with the installation of the NOOBS at the RasPi and configure it to at least have SSH enabled. VNC can be used to remotely connect to the RasPi sparing the extra monitor and keyboard for the PLC device.

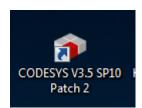


V2 raspberrypi - VNC Viewer			_	
🗕 🖉 🔁 🖉 🔇	👹 Raspberry Pi Configu		Va	🕈 🛜 📣 🛛 🔹 01:15 🔺
Wastebasket				
States of States				
and the second se				
	Rasp	berry Pi Configuration –		
	System	ces Performance Localisation	and the second second	
	Camera	Enabled     O     Disabler	And the second	
	SSH:	Enabled Obisable	-	
	VNC:	Inabled O Disabled	and pression	
	SPI:	Enabled • Disabled		
	12C:	O Enabled • Disabled		
A STATES	Serial:	⊖ Enabled ● Disabled	Contraction of the Party of the	The second second
	1-Wire:	Enabled • Disabled		
	Remote GPIO:	Enabled • Disabled	the loss	and the second second
		Cancel OK		
		Cancer	- Start	
	/			ALL BE ALL
				A Start Start
				and the second
to the second second				

## Figure 2 Raspberry Startup screen showing the system preferences. Additional connection e.g. SPI might be enabled, if additional hardware is connected to the RasPi

SSH is used by the CODESYS engineering environment to install its runtime.

## Install the CODESYS Environment



In a second step install the CODESYS Engineering Environment to your PC. Here we used the V3.5 SP10. This is the main tool for creation of PLC programs and visualizations. Installation will take some time and be prepared to have additional components downloaded and installed to your computer in this step (see Figure 3).



and the set of				
-	Microsoft Visual C++ 2015 Update 3 Redistributable Package (x86)			
ending N	Microsoft Visual C++ 2008 SP1 Redistributable Package (x86)			
Pending Microsoft Visual C++ 2008 SP1 Redistributable MFC Security Update KB2538243(x86)				
ending N	Microsoft Visual C++ 2010 SP1 Redistributable Package (x64)			
ending N	Microsoft Visual C++ 2010 SP1 Redistributable Package (x86)			
ending N	Microsoft Visual C++ 2005 SP1 Redistributable MFC Security Update KB2538242(x86)			
ending N	Microsoft Visual C++ 2005 SP1 Redistributable MFC Security Update KB2538242(x64)			
ending N	Microsoft Visual C++ 2013 Redistributable Package (x86)			

## Figure 3 Additional components installed by the CODESYS engineering environment

After the installation of the engineering environment, we can add the specific components for the example here.

First of all we need to download the RasPi Solution from the 3S web shop. This solution is distributed as a package that needs to be added to the engineering environment via the package manager in the Tools menu (see Figure 4).



Refresh			Sort by: Name	•	Install
Name	Version	Installation date	Update info	Lic	Uninstall
CODESYS Control for Raspberry PI CODESYS SoftMotion COSCAT BASIC	3.5.10.20 4.1.1.0 3.3.3.0	19.04.2017 19.04.2017 19.04.2017	Free version 3.5.11.10 available! Free version 4.2.2.0 available!	No No	Details Updates Search updates Download CODESYS Store Rating CODESYS Store
< ☐ ☐ Display versions	III			4	Close

# Figure 4 CODESYS Package manager allows to add additional components such as the RasPi environment

You might also download and add the OSCAT Basic lib, a library with general purpose function blocks for the CODESYS environment. We do use one of them later in the listing.

The lib can then be installed using the Tools/LibraryRepository.

	a Wash I thinks	- Inde			-a-8-
Ele Edit Vew Broject Ubravies Build Online Debu			The second se		
1996日(日本市 B × (A な)日			[19] 行 19 [9] 第1第1		
	🚺 Library Re				
Devices	Device Re	pository	MC5005 EtherCAT_Master II Device Ibrary	Manager X	
= 🗿 АСФРАНЦА	🛃 yisualizat	ion Styles Repository	a library / 🐨 Properties III Details / 📰 Placeholders 🏼 🎁 Library repos	itory	
Device (CODESYS Control for Rapberry Pi SL)	License B	epository		Namespace	Effective version
= DI PLC Logic	License M	anaor	mee, 2.5.10.0 (25 - Smart Software Solutions GmbH)		3.5.10.0
O Application     Drary Nanager	Scripting		<ul> <li>BCAT)</li> </ul>	USCAT BASIC	3.3.3.0
PLC_PRG (PRG)			Steadpoint Logging Functions, 3.5.5.0 (25 - Smart Software Solutions GebH)	SPLog	1550
= 28 Tesk Configuration	Qustonia		100rvEtherCA7, 3.5.10.0 (25 - Snart Software Solutions Griter)	10ntherota	3.5.10.0
= S MainTask	Qotions		10 (35 - Smart Software Solutions GobH)	1s0n/6P10	1.0.1.10
B EtherCAT_Master.EtherCAT_Task	Lipdate R	aspberryPI	endand, 3.5. 10.0 (System)	3oStandard	3.5.10.0
B) PLC PRG		🕴 📲 943,8esc = 5	H1_Basic, 4.1.1.0 (35 - Smart Software Solutions GobH)	SHO_Basic	4.1.1.0
			43_CNC, 4.1.1.0 (35 - Smart Software Solutions GmbH)	SHIT_CHC	4.1.1.0
			= 943_Robotics, 4.1.1.0 (35 - Smart Software Solutions Gribh)	SH3_Robotics	4.1.1.0
			Visu = SH3_Robotics_Visu, 4.1.0.0 (35 - Smart Software Solutions GmbH)	SM3_Robotics_Veu	
			ratori = SH3_Transformatori, 4.1.1.0 (35 - Snart Software Solutoris Gribh)	TRAPO	4110
		- Standard = St	andard, 3.5.9.0 (System)	Standard	3.5.9.0
<b>3</b> Set					
GPIOs_A_B (GPIOs A/B)     Onewire		[			
Comera device					
Called Service					
Cevices 2 POUs					
Chanas () 100.8 ■ Detacasjan - Tricki (stravic), 5 merszaget)) ■ Merszaget - Tricki (stravic), 5 merszaget))					Garnet user (nobioda)

Figure 5 Installation of additional libraries using the LibraryRepository



Only after this step, we are now able to install the CODESYS runtime component at the RasPi. This installation is done directly out of the CODESYS runtime environment. There is an extra menu entry in the Tools menu Update Raspberry). This will open up a window where you have to identify your RasPi (IPAddress) and add the login information.

Ele Edit View Project Build Online Debug 10 D≩ ⊒ I ⊕ I co co X Po Ba × I AN	Tools Window Help Package Manager Ubrary Repository	= <(1)= <2
Devices v 0	Perice Repository     Yousil Element Repository     Yousilation Styles Repository     License Bepository     License Bepository     License Benger     Spripting     Qustamize	910 Patch 2
	Qptions Update RaspberryPI	PLC

Package	
Version	V3.5.10.20
Package directory	C:\Users\wage\CODESYS Control for Rat Change
Network	
IPAddress	192.168.0.43
	Scan
Login	
Username	pi
Password	•••••

Press Scan to identify the RasPis is your network environment. Alternatively add the IPAddress manually.

Add the login information or the RasPis SSH. Default is:

Username: pi

Password: raspberry

## Figure 6 Update Raspberry

Finally, in order to use the FAULHABER MCs in CODESYS projects, the device description files distributed with the MotionManager have to be installed to the CODESYS Device Repository – once again the Tool menu. The description files can be found in the installation directory of the MotionManager/ESI, e.g. something like C:\Program Files (x86)\Faulhaber\Motion Manager 6\ESI.



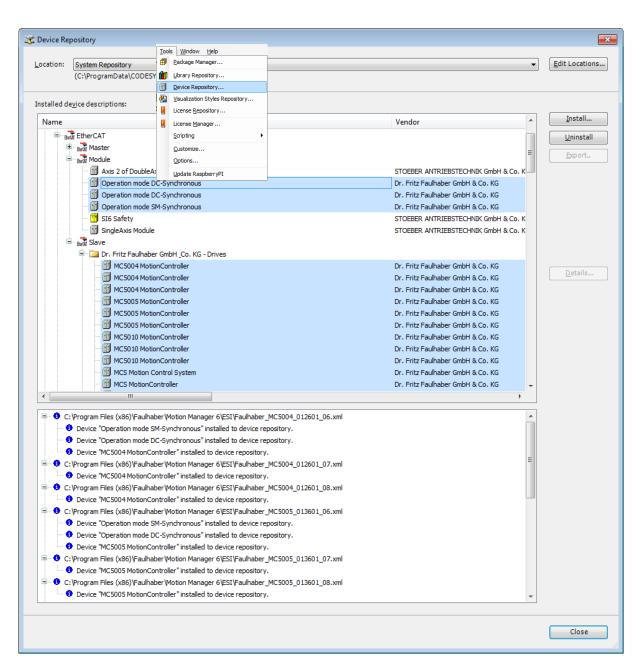


Figure 7 Adding FAULHABER MotionController to the CODESYS Device Repository

After this final step all the prerequisites are finished. We can now start with the application example.



# Create a new PLC application

Creation of a PLC application requires a minimum of 4 steps.

- 1. Open a new PLC project
- 2. Create a system description
- 3. Add the functionality
- 4. Test the system behavior

## **Open a new project**

A new CODESYS project can be created using either the direct link in the Start Page or via menu File/New Project.

Here we use a Standard project. Select whatever path is convenient and enter a project name.

Bie Edit View Project Build Online Debug Tools Window Help 管部副画いへよ物能XIAL Gilleli 100 - Cilleli 106 000 → ■ ペロロロロ 106 000 → ■ ペロロロロロロロロロロロロロロロロロロロロロロロロロロロロロロロロロロ	•
Devices • 4 X	•
Start Fage A	•
CODESYS V3.5 SP10 Patch 2	
New Project	
Basic Opera	
Ne □ Drojects □ Droj	Â
Empty project HMI project	
Recent Pro	
Standard project with Application Composer	
	Sn Sn
a aa uudi projeet kiiu 1 de iiee apalicaation he automation of	R
Par A project containing one device, one application, and an empty implementation for PLC_PRG	Wähl dem
Name: DefaultDemoET_ST	Schal
Location: C:\Users\wage\CODESYS Control for Raspberry P1\Projekte3.5.10.20	
35 SP11	
OK Cancel	
New 64 bit version of the CODESYS Development System (parallel to 32 Bit version) // v	ve
Institucation center for important technical information // improvements of LD and CFC en	dit)
"memory tools" replaces memory view // data breakpoints // performance  Close page after project load	
	- Ψ F
Messages - Total 0 error(s), 0 warning(s), 0 message(s)	_
Copying new runtime	

## Figure 8 Open an new CODESYS project

In a second step the target type and programming type for the single program organization unit (POU) PLC\_PRG are selected (see Figure 9).PLC\_PRG is part of the new project and already configured to be executed by the main task.

Here we choose structured text (ST), a text based coding very similar to PASCAL.



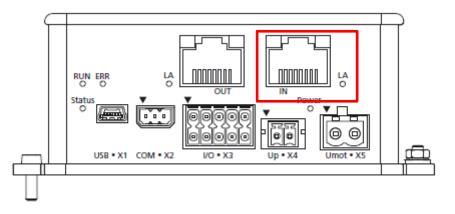
Standard Pr	oject					
	objects within - One program - A program F - A cyclic task	t to create a new standard project. This wizard will create the following in this project: mmable device as specified below PLC_PRG in the language specified below : which calls PLC_PRG to the newest version of the Standard library currently installed.				
	<u>D</u> evice:	CODESYS Control for Raspberry Pi SL (3S - Smart Software Solutions Gmbl				
	PLC_PRG in:	Structured Text (ST)				
		Continuous Function Chart (CFC) Continuous Function Chart (CFC) - page-oriented Function Block Diagram (FBD) Ladder Logic Diagram (LD) Sequential Function Chart (SFC) Structured Text (ST)				

## Figure 9 Selection of the target runtime and the programing style

Finally the new project is displayed in the project tree (see Figure 11). Listed under the Device node (which represents the RasPi PLC controller) there is one node for the PLC\_Logic. All other ones are empty.

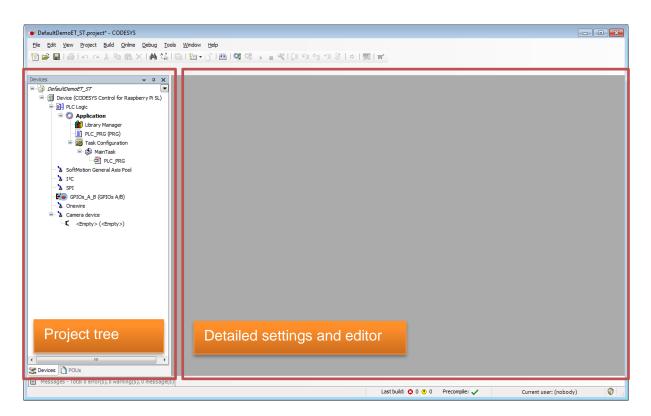
Within the 3S startup manual for the RasPi solution several simple examples in combination with different types of I/O connected to the RasPi are explained. Here we will focus on a simple system out of the RasPi and a single FAULHABER MC 50xx connected via EtherCAT (Figure 1).

In our example the Ethernet port of the RasPi is directly connected to the EtherCAT port labeled IN at the MotionController. No additional connection is necessary.



## Figure 10 Connections of a FAULHABER MotionController MC5005/10 S ET





## Figure 11 Project view in the CODESYS environment

## **Connect to the Device**

The CODESYS Development System can generate code for different targets. In order to tailor the configuration to the selected target your RasPi should be added to the project in a first step. To do this, a double-click on the device node in the project tree will open the device page (Figure 12) from where the network can be scanned for devices. Several CODESYS runtime systems might be identified. RasPi based targets cannot be identified by a blinking LED or something similar. One of the properties of the identified devices is the Device\_Name. So if you plan to use several of them, you should use different names here<sup>1</sup>.

The settings of the connected device are displayed on the device page after this step only (see Figure 13).

<sup>&</sup>lt;sup>1</sup> Targets can be renamed using the Device menu in the Device settings page.



DefaultDemoET_ST.project* - CODESYS			
Ele Edit View Project Build Online Debug Too	ols Window Help		
1 1 1 1 1 1 1 1 1 1 1 1 1 1			
Devices 👻 🕂 🗙	Device X		_
DefaultDemoET_ST			•
Device (CODESYS Control for Raspberry Pi SL)	Communication Settings Scan I	Network Gateway - Device -	
PLC Logic	Applications		
Application     Ibrary Manager	Applications		
PLC_PRG (PRG)	Backup and Restore		i
😑 💹 Task Configuration	Files		
🖻 🥩 MainTask	riies	Gateway	•
PLC_PRG     SoftMotion General Axis Pool	Log	Gateway-1	PC 1884
a solution deneral Axis Pool	PLC Settings	IP-Address:	PC1004
- SPI		localhost	
GPIOs_A_B (GPIOs A/B)	PLC Shell	Select Device	
- à Onewire ⊟- à Camera device	Users and Groups	Select Device Select the network path to the controller:	
K <empty> (<empty>)</empty></empty>	· · · · · · · · · · · · · · · · · · ·	Select the network path to the controller:	Device Name:  Scan network
	Task Deployment	RasPI_DFF_10 [002D]	RasPi_DFF_10
	Status		Device Address:
			Block driver:
	Information		UDP =
			Number of Echannels:
			4
			Target ID: 0000 0010
			Target Name: CODESYS Control
			for Raspberry Pi SL
			Target Type: 4102
			Target Vendor:
< +			harget venuor.
POUs			OK Cancel
Messages - Total 0 error(s), 0 warning(s), 0 message(s	0		
		Last build: 😋 0 🕐 0 Precom	pile: 🗸 Current user: (nobody) 🔇 🖉

# Figure 12 Identify the target controller (RasPi)

DefaultDemoET_ST.project* - CODESYS					- • •
Ele Edit View Project Build Online Debug Too	ols <u>W</u> indow <u>H</u> elp				
🎦 😅 🔚 🕘 🗠 🗠 👌 🗎 🗶 🖊 🍇	🖶   🏪 • 👔   🏙   🥞 🚿 🅠	📲 💐 (ji 🖻 🖕 ti 🎖   o   🛒 🛒			
Devices - 7 X	Device X				-
DefaultDemoET_ST	Commission Contract	Scan Network Gateway - Device -			
Ovice (CODESYS Control for Raspberry Pi SL)     Ovice (CODESYS Control for Raspberry Pi SL)	Communication Settings	outendy of Device of			
Application	Applications				
Library Manager	Backup and Restore				
🖃 🧱 Task Configuration 🖹 🥸 MainTask 🗍 PLC PRG	Files		Gateway	•••	
SoftMotion General Axis Pool	Log	Gateway-1		[002D] (active)	•
- 3 I <sup>2</sup> C - 3 SPI	PLC Settings	IP-Address: localhost	C	Device Name: RasPi_DFF_10	
GPIOs_A_B (GPIOs A/B)	PLC Shell	Port: 1217		Device Address: 102D	
Camera device Camera device Campty> ( <empty>)</empty>	Users and Groups		т	arget ID:	
	Task Deployment			000 0010 arget Type:	
	Status		4	102	
	Information		т 3	'arget Vendor: IS - Smart Software Solutions GmbH	
				arget Version: 1.5.10.20	
		1			
4					
2 Devices POUs					
Messages - Total 0 error(s), 0 warning(s), 0 message(s)					
		La	st build: 😋 0 🕐 0 🛛 Precompile: 🧹	<ul> <li>Current user: (nobody</li> </ul>	) 🔮 🔐

# Figure 13 Project view with identified target controller



## Create the system definition offline

Before we start programming the application, we should create a complete system definition. Here we need to add the EtherCAT subsystem. Additional examples of how to add drivers for SPI or I<sup>2</sup>C based subsystems can be found directly in the 3S startup description.

Additional components can be added to the project using the context menu in the project tree. Here the master is added to the node Device using Add Device. First we need to add the driver for the EtherCAT – the EtherCAT Master - using add Device out of the context menu of the project. Available devices can be selected from the Add Device window using filters and the tree view. As of now, there is only one EtherCAT master, so this is an easy one.

A double-click on the tree node of the EtherCAT master will open the settings of the component (Figure 15).

DefaultDemoET			ſ	Add I	Device							×
		ect Build Online Debug Tools → 🔏 Pas 📾 🗙 I 🚧 🖓 I		_								
		in a ratio of the date		Name:	EtherCAT	[_Master						
Devices	_	- 0 ×		Action	:							
E Device (C E Device (C	men	VF Candral for D condenses D D1		Ap	pend devi	ce 🔘 Insert d	levice 🥑	) <u>P</u> lug d	evice 🔘 🕽	Jpdate device		
= 0	8	Сору		Enter a	a string for	r a fulltext searc	h in all dev	vices	Vendor:	3S - Smart Sc	oftware Solutions G	mbH 🔻
	×			Nam	e			Vendo	r		Version	Description
	1	Browse		8-1	Fieldbus	ses						
		Properties		6	CAN CAN	lbus						
> Soft	钿			6	Bedi Ethe	erCAT						
2 SP1	•	Add Folder			Bell	Master						
E GPI		Add Device Update Device				EtherCAT M	laster	3S - Sm	art Softwar	e Solutions Gm	bH 3.5.10.0	EtherCAT Master.
		Edit Object				ernet Adapter						
κ.		Edit Object With			Ethe							
		Edit 10 mapping			- KIII Mod							
		Import mappings from CSV			- III Prof	înet IO						
		Export mappings to CSV										
	4	Online Config Mode										
		Reset origin device [Device]		•								
		Simulation		Gro	up by cat	edory 🔲 Disi	olav all ve	rsions(f	or experts o	only) 🥅 Disc	lay outdated vers	ions
											,	
				1		EtherCAT Maste 3S - Smart Sof					*	
					Catego	ries:	ware solu	luons Gm	DM			
					Version	: 3.5.10.0					=	
						lumber: tion: EtherCAT	Master					
					Descrip						-	~
				Apper		ed device as l	ast child	of				
				0	You can s	elect another t	arget nod	e in the i	navigator v	hile this wind	ow is open.)	
										(	Add Device	Close
										l	Autoconce	Close

## Figure 14 Add the EtherCAT Master to the project using Add Device

Next we need to configure the network interface to be used by the EtherCAT master at the target controller. A list of available network interfaces can be displayed by clicking the Browse ... button (Figure 15). Here we have to select the single Ethernet port of the target: eth0. The identification within the project can be either MAC based or name based. The latter allowing for easier portability of the project.

The cycle time of the EtherCAT system defaults to the cycle time of the main task. For the RasPi this is 4ms. We did test cycle times down to 2ms successfully with either a RasPi 2B or 3B target<sup>2</sup>.

<sup>&</sup>lt;sup>2</sup> If a different timing is required, first change the cyclic time of the main task, then adjust the Cycle Time of the EtherCAT and re-compile / re-load the system.



r			
DefaultDemoET_ST.project* - CODESYS			
Eile Edit View Project Build Online Debug Too	ols <u>W</u> indow <u>H</u> elp		
🎦 🚅 🖬 🕌 🗠 🗠 🌾 🛍 🛍 🗙 🖊 🎼	🛍   🖄 - 👔   🏙   🧐 🚿 🕞 🗉	💘   (⊒ 9⊒ 4⊒ 9   ⇔   🛒   ≓	
Devices - 7 X	Device EtherCAT_Mast	ter X	-
DefaultDemoET_ST			
Device (CODESYS Control for Raspberry Pi SL)     Device (CODESYS Control for Raspberry Pi SL)	General	Autoconfig Master/Slaves	ther CAT
Application	Sync Unit Assignment	EtherCAT NIC Setting	
Library Manager		-	Enable Redundancy
PLC_PRG (PRG)	EtherCAT I/O Mapping		
□-100 Task Configuration □-100 MainTask	Status	Source Address (MAC) 00-00-00-00-00 Browse	
Main Lask     Main Lask     EtherCAT_Master.EtherCAT		Network Name	
D PLC_PRG	Information	Select Network by MAC	
EtherCAT_Master (EtherCAT Master)		Distributed Clock	
- SoftMotion General Axis Pool			
- 3 SPI		Cycle Time 4000 💭 µs	
GPIOs_A_B (GPIOs A/B)		Sync Offset 20 📩 %	
a Onewire		Sync Window Monitoring	
🖻 - 🍐 Camera device		Sync Window 1 👘 µs	
<pre>L <empty> (<empty>)</empty></empty></pre>		Select Network Adapter	
		Select Network Adapter	
		MAC address Name Description	
		B827EB2C1E7D eth0	
		3E8B8B76A2AB tap0	
		A20973ETCC5E tapo	
		B827EB794B28 wlan0	
۰ III ا			
Cevices POUs			
Messages - Total 0 error(s), 0 warning(s), 0 message(s)	)		OK Abort
			i.

## Figure 15 EtherCAT Master settings and list of available network adapters

After the basic configuration of the EtherCAT subsystem we can add slaves to the system. This can either be done offline, by manually adding devices to the EtherCAT bus node or using the

online Config Mode see being available either from the context menu of the device in the project tree or out of the menu. Online Config will start the EtherCAT system configured in the last step and will scan the bus for known devices. Online Config avoids picking the wrong device or device revision.

In offline mode, you need to add the devices manually using the Add Device entry of the Ether-CAT master context menu. When using offline mode, please verify to have the correct device types and revision added to your project (Figure 16).



By default Vendor ID and product ID of the nodes are checked during startup of the EtherCAT subsystem. So even if a MC 5010 S ET and a MC 5005 S ET look very similar, they must not be mixed up here. Product revision is not checked by default, however in order to avoid communication problems the correct device description should be selected in offline mode.

Online mode even allows updating the contents of the EtherCAT slave EEPROM. This has to be done manually after a firmware update of the MC<sup>3</sup>.

<sup>&</sup>lt;sup>3</sup> A firmware update may add objects to the object dictionary of a servo drive. In order to correctly access all objects, the correct device description has to be used within the project. The device revision of an EtherCAT drive is identified via the device revision entry in the slave EEPROM.



inter a string for a full	ext search in all devices Vendor:	Dr. Fritz Faulhaber GmbH & Co. KG		•
Name		Vendor	Version	Description
🖃 🕤 Fieldbusses				
Brother CAT				
Brown Slave				
	. Fritz Faulhaber GmbH _Co. KG - Drives			
- 1	MC5004 MotionController	Dr. Fritz Faulhaber GmbH & Co. KG	Revision=16#0000006	EtherCAT Slave impor
- 1	MC5004 MotionController	Dr. Fritz Faulhaber GmbH & Co. KG	Revision=16#0000007	EtherCAT Slave impor
🖬	MC5004 MotionController	Dr. Fritz Faulhaber GmbH & Co. KG	Revision=16#0000008	EtherCAT Slave impor
🖬	MC5005 MotionController	Dr. Fritz Faulhaber GmbH & Co. KG	Revision=16#0000006	EtherCAT Slave impor
🖬	MC5005 MotionController	Dr. Fritz Faulhaber GmbH & Co. KG	Revision=16#0000007	EtherCAT Slave impor
1	MC5005 MotionController	Dr. Fritz Faulhaber GmbH & Co. KG	Revision=16#0000008	EtherCAT Slave impor
🖬	MC5010 MotionController	Dr. Fritz Faulhaber GmbH & Co. KG	Revision=16#0000006	EtherCAT Slave impor
- 1	MC5010 MotionController	Dr. Fritz Faulhaber GmbH & Co. KG	Revision=16#0000007	EtherCAT Slave impor
- 1	MC5010 MotionController	Dr. Fritz Faulhaber GmbH & Co. KG	Revision=16#0000008	EtherCAT Slave impor
- 1	MCS Motion Control System	Dr. Fritz Faulhaber GmbH & Co. KG	Revision=16#0000007	EtherCAT Slave impor
1	MCS MotionController	Dr. Fritz Faulhaber GmbH & Co. KG	Revision=16#0000008	EtherCAT Slave impor
· 🖬	MCS MotionControllSystem	Dr. Fritz Faulhaber GmbH & Co. KG	Revision=16#0000006	EtherCAT Slave impor
٠	m			4
Group by category	Display all versions (for experts on	ly) 🔲 Display outdated versions		
Vendor: Dr. Fr Categories: Version: Revis Order Numbe	EtherCAT Slave imported from Slave XML:	Faulhaber_MC5005_013601_08.xml Dev	vice: MC5005	-

## Figure 16 List of known devices



The configuration of the EtherCAT slave is done entirely using the slave settings of the MC node in the CODESYS Development System (see Figure 17). No configuration via FAULHABER MotionManager is necessary.

Review and modify the settings of the connected MC 5005 by double-click on the node MC5005 in the project tree an enable the expert settings. Expert settings are necessary to modify the PDO mappings of the device.



<ul> <li>♥ Unbenannt1.project* - CODESYS</li> <li>Ele Edit View Project Build Online Debug Tools Window E</li> <li>™ ♥ ■ ● ○ ○ × № № × ● ▲ ◇ □ □ ○ ○</li> </ul>		ī ¢ī +ī Ş   ¢   <b>∰   च</b>			×
Devices	General FMMU/Sync Expert Process Data Process Data Startup Parameters EtherCAT I/O Mapping Status Information	isster       IMCS005 ×         Address       Image: Startup checking         > Distributed Clock         > Startup checking         > DC cyclic unit control: assign to local µC         > Watchdog         Identification         Identification         © Disabled         C configured Station Alias (AD0 0x0012)         Explicit Device Identification (AD0 0x0134)         Data Word (2 Bytes)	Additional	1001 ¢	
B Messages - Total 0 error(s), 0 warning(s), 3 message(s)		Last build: 🔕 0 😗 0 🛛 Preci	ompile: 🗸	Current user: (nobody)	Ø.

# Figure 17 Basic EtherCAT Slave settings of the CODESYS system – expert settings enabled

By default Distributed Clock is used as synchronization method for the FAULHABER drives. Activate the configuration entries for MC 5005 and enable 'Expert Mode' as in Figure 17. To enable the Distributed Clock at the MC please check the entries at the General Tab according to Figure 18.

If Synchmanager (SM) synchronous operation is to be used, use the context menu of the MC 5005-OpMode in the project tree to change the plugged OpMode.

Of course, synch cycle times of the slave and of the application must be the same, here it is  $4ms = 4000 \mu s$ .



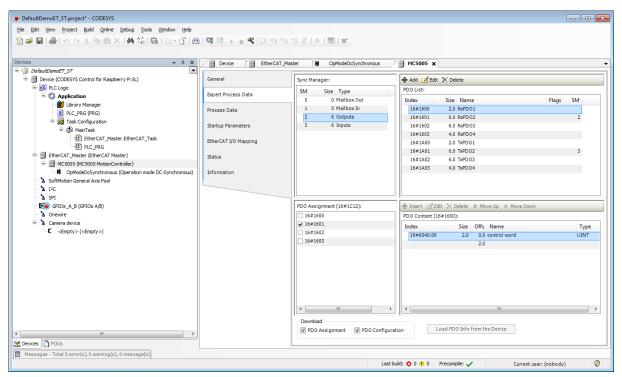
Unbenannt1.project* - CODESYS			- 🗆 X
Ele Edit View Project Build Online Debug Tools Window H	elp		
🎦 🚅 🛃 🗐 🗤 🗠 🌾 🛍 🗙 🖬 🏭 🛅 - 🖆	1 🕮 1 😋 🕬 🕟 🔐 📲 🔏 1 💷 🔊	I 4 +I 8   ↓   <b>#</b>   <del>#</del>	
Devices 👻 🖣 🗙	Device EtherCAT_M	aster MC5005 X	-
Unbenannt1     v     Green CODESYS Control for Raspberry Pi SL)	General	Address Additiona	
PLC Logic	FMMU/Svnc	AutoIncAddress 0	le Expert Settings
Application     Ibrary Manager	-	EtherCAT Address 1001 🖨 Optio	onal
PLC_PRG (PRG)	Expert Process Data	Distributed Clock	
i= 👹 Task Configuration i 🗟 🅸 MainTask	Process Data	Select DG DC-Synchronization	~
PLC_PRG	Startup Parameters	⊡ Enable 4000 vnc Unit Cycle (μs)	
- 🖽 EtherCAT_Master.EtherCAT_Task 	EtherCAT I/O Mapping	Sync0:	
MC5005 (MC5005 MotionController)	Status		Cycle Time (µs)
SoftMotion General Axis Pool     I <sup>2</sup> C	Status		
	Information	User Defined	Shift Time (µs)
GPIOS_A_B (GPIOS A/B)		Sync1:	
Onewire		Enable Sync 1	
🖃 🍐 Camera device		Sync Unit Cycle x 1 √ 4000 4000	Cycle Time (µs)
<pre>Leer&gt; (<empty>)</empty></pre>		O User Defined	Shift Time (µs)
			chine (ho)
		D Startup checking D Timeou	its
		DC cyclic unit control: assign to local µC	
		> Watchdog	
		Identification	
		Disabled	
		Configured Station Alias (ADO 0x0012) Value	1001
		O Explicit Device Identification (ADO 0x0134)	
		O Data Word (2 Bytes) ADO (hex)	16#0
Messages - Total 0 error(s), 0 warning(s), 3 message(s)			
		Last build: 😳 0 😗 0 Precompile: 🗸	Current user: (nobody) 🛛 🕅

Figure 18 Synch settings of the EtherCAT slave



# **Create / Modify PDO Mapping**

In order to tailor the data exchange between the PLC application and the MotionController to the application we need to review and configure the process data objects (PDO) of the drive using the Expert Process Data Tab of the drive settings.



## Figure 19 Expert view of the PDOs (defaults)

A maximum of 4 objects can be mapped to a single PDO.



To maintain the correct order of processing the controlword (0x6040.00), the OpMode (0x6060.00) and the target position (0x607A.00) have to be mapped to a single PDO. The controlword must not be mapped to more than 1 active PDO.

Within the default mapping the controlword is mapped to all RxPDOs, RxPDO 2 - 4 each covering a different simple access of controlword and the target value for either pos control (RxPDO2), speed control (RxPDO3) or torque control (RxPDO4). Same for the Tx direction. To avoid a mixed update of the controlword, only PDO2 is activated by default.



## Here we want to access:

Read		Write	
0x6041.00	statusword	0x6040.00	controlword
0x6061.00	Modes Of Op. Display	0x6060.00	Modes of Operation
0x6064.00	Position Actual Value	0x607A.00	Target Position
0x606C.00	Velocity Actual Value	0x6081.00	Profile Velocity
0x6077.00	Torque Actual Value	0x6083.00	Profile Acceleration
0x2311.01	Dig Input logic state	0x6084.00	Profile Deceleration

## Table 2 List of all parameters used in the interaction between PLC and servo drive

So the PDO Mapping here might be:

Read from the MC	Write to the MC
TxPDO1: not used, no change necessary	RxPDO1: not used, no change necessary
TxPDO2:	RxPDO2:
1. 0x6041.00 (statusword)	1. 0x6040.00 (controlword)
2. 0x6061.00 (Modes of Op Display)	2. 0x6060.00 (Modes of Operation)
3. 0x2311.01 (Dig Input logic state)	3. 0x607A.00 (Target Position)
4	4. –
TxPDO3:	RxPDO3:
1. 0x6064.00 (Position actual value)	1. 0x6081.00 (Profile Velocity)
2. 0x606C.00 (Velocity actual value)	2. 0x6083.00 (Profile Acceleration)
3. 0x6077.00 (Torque actual value)	3. 0x6084.00 (Profile Deceleration)
4	4
TxPDO4: not used, no change necessary	RxPDO4: not used, no change necessary

## Table 3 Proposed PDO mapping

To edit the mappings the 4 tables in the Expert Process Data tab can be used.



DefaultDemoET_ST.project* - CODESYS				
<u>File E</u> dit <u>V</u> iew <u>P</u> roject <u>B</u> uild <u>O</u> nline <u>D</u> ebug <u>T</u> ools <u>W</u> indow <u>H</u> elp				
🛅 🖆 🔚 🎒 Ior 🖓 🛍 🏙 🗙 🖬 🎼 🛅 📲 🛅 🖓 🗳 🛗	1 👒 💖 🕞 🗉 🔏 I 🗐 🕾 🐿	*= \$   \$   <b>#</b>   <b>#</b>		
Devices v 4 X	PLC PRG II Device	EtherCAT_Master MC5005 X	OpModeDcSynchronous	
Device (CODESYS Control for Raspberry Pi SL)	General	Sync Manager:	🕂 Add 📝 Edit 🗙 Delete	
PLC Logic		SM Size Type	PD0 List:	
= O Application	Expert Process Data	0 0 Mailbox Out	Index Size Name	Flags SM
Library Manager	Process Data	1 0 Mailbox In	16#1600 2.0 RxPD01	
PLC_PRG (PRG)     Section     Task Configuration		2 19 Outputs	16#1601 7.0 RxPD02	2
MainTask	Startup Parameters	3 14 Inputs	16#1602 12.0 RxPD03	2
EtherCAT_Master.EtherCAT_Task			16#1603 1.0 RxPD 04	
D PLC PRG	EtherCAT I/O Mapping		16#1A00 2.0 TxPDO1	
EtherCAT_Master (EtherCAT Master)	Status		16#1A01 4.0 TxPDO2	3
MC5005 (MC5005 MotionController)	510105		16#1A02 10.0 TxPDO3	3
OpModeDcSynchronous (Operation mode DC-Synchronous)	Information			
SoftMotion General Axis Pool	· · · · · · · · · · · · · · · · · · ·	(1)		
<b>≥</b> 1²C				
SPI				
GPIOs_A_B (GPIOs A/B)		PDO Assignment (16#1C13):	Insert ZEdit X Delete 🕆 Move Up 🕴 Move Down	
Dnewire     Camera device				
		☐ 16#1A00 ✔ 16#1A01	PDO Content (16#1601):	
Compty > (Compty >)		✓ 16#1A01 ✓ 16#1A02	Index Size Offs Name	Туре
		16#1A03	16#6040:00 2.0 0.0 control word	UINT
		. 10 100	16#607A:00 4.0 2.0 target position	DINT
			16#6060:00 1.0 6.0 Modes of operation 7.0	USINT
			7.0	
			3	
		<		
		Download		
< >		PDO Assignment VDO Configuration	Load PDO Info from the Device	
POUs				
Messages - Total 0 error(s), 0 warning(s), 5 message(s)				
<ul> <li>Pressages - rotar o error(s), o warning(s), 5 message(s)</li> </ul>			Last build: 😋 0 🕐 0 Precompile: 🗸 🛛 🔿	urrent user: (nobody)
			Last build: 👽 0 😗 0 Precomplie: 🗸 🛛 C	urrent user: (nobody)

## Figure 20 Expert view of the device PDOs, adjusted to Table 3

Edit the RxPDOs:

- 1. Select the Outputs in the Synch Manager table and enable RxPDO3 by checking it in the PDO Assignment
- 2. Select RxPDO2 in the PDO List
- 3. Add a new entry to RxPDO2 in the PDO Content table using the **+**Insert. Here we add 0x6060.00 Modes of Operation to the mapped entries.
- 4. If necessary the order of the objects can be modified using the Move up or Move down buttons
- 5. Select RxPDO3 in the PDO List
- 6. Delete the default entries (controlword, target velocity) using the **×**Delete and add 0x6081.00 Profile Velocity, 0x6083.00 Profile Acceleration and 0x6084.00 Profile Deceleration.
- 7. Enable additional PDO3 by checking them in the PDO Assignment table
- ⇒ The total size of the Outputs in the Synch Manager table should now be 19 bytes

### Edit the TxPDOs:

- 1. Select the Inputs in the Synch Manager table and enable TxPDO3 by checking it in the PDO Assignment
- 2. Select TxPDO2 in the PDO List
- 3. Add 0x6061.00 Modes of Operation Display to the mapped entries
- 4. Add 0x2311.01 Digital Input Logic States to the mapped entries
- 5. Delete 0x6064.00 Position actual value from TxPDO2



- 6. If necessary the order of the objects can be modified using the Move up or Move down buttons
- 7. Select TxPDO3 in the PDO List
- 8. Delete the default entries of the 0x6041.00 statusword and add 0x6064.00 Position actual value, 0x6077.00 Torque actual value.
- 9. Enable additional PDO3 by checking them in the PDO Assignment table
- ⇒ The total size of the Inputs in the Synch Manager table should now be 14 bytes



There is no need to explicitly configure the mappings beforehand in the MC using FAULHABER MotionManager because the PDO assignment and mappings are downloaded to the MC by the PLC during initialization.

Different from the CANopen environment we don't have to care about the transmission type of PDOs they will be updated in every communication cycle anyway.

## Additional configuration of the MC



To configure the correct motor or the I/Os of the MC please use FAULHABER MotionManager and the secondary interface of the MC which is either the USB interface for an external controller or the RS232 of an MCS. This configuration is mandatory before the motor is operated by the PLC.



# Create a PLC program

By default the program sequence of the project is stored in the PLC\_PRG component of the project tree. Here it is a structured text (ST) version. Structured text is similar to PASCAL. A doubleclick on the PLC\_PRG entry of the tree opens the editor tab.

The editor is divided into the variables section and the code section. All variables and instances of function blocks have to be defined in the variables section in between the key words.

Definition of global variables

## VAR

• • •

## END\_VAR

So if we want to interact with the MC we usually have to read the statusword and write the controlword. Both of them unsigned 16 bit.

The entries would look like:

## VAR

ControlWord: UINT:=0; StatusWord: UINT;

### END\_VAR

The variable ControlWord is set using a default value, StatusWord does not need a default value, it's updated out of the controller anyway.

Variables can also be created on the fly during coding. Each time a new variable name is used a window will pop up and gather the necessary attributes.

## Map PLC variables to the process image

Please note: the listing above does only generate the two variables for the PLC program. In order to access the MC the two variables have to be linked to the PDOs. This can be done using the EtherCAT I/O Mapping tab of the MC node in the project tree. A double-click into one of the variables (empty or already connected) will allow to select a new variable.

So here we should map the entry controlword in RxPDO2 to the PLC variable ControlWord and the statusword entry in TxPDO2 to the PLC variable StatusWord:



DefaultDemoET_ST.project* - CODESYS							
Ele Edit View Project Build Online Debug Tools Window Help							
🎦 📽 📕   🎒   🗢 🐃 🌾 陀 🖄 🖄   🎬   🎬 - 🗂   🎬	। 👒 ଔ 🕞 📲 🔏 🗔 १	∎ t⊒ *≣ β   φ   <b>∰</b>   <del>⊈</del>					
Devices 👻 4 🗙	EtherCAT_Master	Device Device OpModeDcSynchronou	us 📄 PL	C_PRG MC5005 X	Library Mana	aner	
DefaultDemoET ST					clorary mane	Jgci	
E Device (CODESYS Control for Raspberry Pi SL)	General	Find	Filter S	Show all	•		
PLC Logic		Variable	Mapping	g Channel	Address	Туре	Unit Description
Application	Expert Process Data	Application.PLC PRG.Control		control word	%QW0	UINT	control word
Library Manager	Process Data	+ Typication registered		Modes of operation	%Q82	USINT	Modes of ope
PLC_PRG (PRG)		÷- 5	nout Assistant				t positic
Image: Second secon	Startup Parameters	B- 🍫	Text search Categories				e velocit
EtherCAT_Master.EtherCAT_Task		B- %	Variables	A Name	Туре	Address	ore accele
	EtherCAT I/O Mapping	H - **		O Application     *      PLC_PRG	Application /ROGRAM		e decele
EtherCAT_Master (EtherCAT Master)	Status	+ * Application.PLC_PRG.Statu		Acc     Acc     AccDecSlope	DINT INT		s word
General MC5005 (MC5005 MotionController)				· ActPos	DBVT		s of ope
OpModeDcSynchronous (Operation mode DC-Synchronous)	Information			ControlWord     Pec	LBNT DBNT		l input k
🚡 SoftMotion General Axis Pool				PeltaAccDec     PigitalInputs	CONSTANT DBIT BYTE		ity actu
a I²C				OriveDilnput     PriveDilast	800L 800L		ie actua
SPI				Ø DriveEnable	BOOL		
CPLOS_A_B (GPLOS A/B)				PriveEnabled     PriveEnInput	8001 8001		
A Camera device				PriveEnLast	8001		
<pre>K <empty> (<empty>)</empty></empty></pre>			Structured view		D	Iter: None	•
			Dogumentation:		Insert with arguments	Copert with	gamespace prefix
			ControlWord: UINT :	:= 0;			<u> </u>
			(VAR)				
		(					= <b></b>
							-
		IEC Objects				ОК	Cancel
		Variable	opping type			_	
		IMC 5005	🍫 ETCSI	ave			
۰ III ا			a				
2 Devices POUs		🍾 = Create new variable	🍾 = Map to exi	isung variable			
Messages - Total 0 error(s), 0 warning(s), 0 message(s)							
			Last build: 👩	0 😗 0 Precompile: 🗸	Curre	nt user: (no	body) 🔇

# Figure 21 Mapping dialog - select the application variable to be updated by / to update the PDO

## Access to a single bit within the ControlWord/StatusWord

Sometimes we need to access a single bit within a variable and want to ignore the others. This can be done e.g. for the flags TargetReached (bit 10) and SetPointAcknowledge (bit 12) of the PP mode using the .Bit operator. The result is a Boolean information.

## //Flags of the PP mode

IsInPos := StatusWord.10; SetPointAck := StatusWord.12;

## Add additional libs to the project

If available it's recommended to rely on existing, proven code. This can be done using libraries. The OSCAT Basic lib installed earlier supplies some general purpose function blocks and should be added to the project. This is done using the Library Manager entry of the project tree. Once again a double-click will open the view where we can add a library. Filters are supported.



ekas • # X • @ DehulDemol7_57	EtherCAT_Master     Device     Device     OpModeDCSynchronous     Add library     X Delete library     Properties     Details     Placeholders	RCJRG BCSOS Brany Manager X
	Name 4 - 20 Stanson + 35 30 2 (31 - how 1 of these folders odd) 4 - 20 Stanson + 35 30 2 (31 - how 1 of these folders odd) 4 - 20 Stanson + 20 Stanson + 15 30 2 (31 - how 1 of these folders 4 - 20 Stanson + 20 - stanson + 10 - stanson (ded) 4 - 20 Stanson + 20 - stanson + 10 - stanson (ded) 4 - 20 Stanson + 20 - stanson + 10 - stanson (ded) 4 - 20 Stanson + 20 - stanson + 10 - s	
<ul> <li>compto (compto)</li> </ul>	* () (fiscalineous)	

## Figure 22 CODESYS LibraryManager

## Download and start the application

After having entered the code, the application can be compiled using either the Build menu, the build-icon or F11.

A successfully built application can be downloaded to the PLC using Online/Login (or the login icon). The application is downloaded but will remain stopped until we explicitly start it using Debug/Start or the start icon.

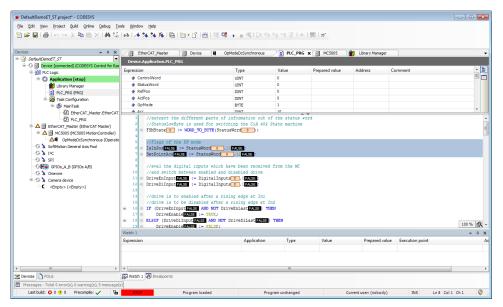


Figure 23 Debugging view of PLC\_PRG - Application still stopped



es 🗸 🖓 🕹		e 🛚 🗑 OpModeDcSynchronous	PLC_PR	5 × MC5005	Library Mana	iger		
	Device Application.PLC_PRG				-			
😔 🗊 Device [connected] (CODESYS Control for R	as Expression	Туре	Value	Prepared value	Address	Comment		
= ]] PLC Logic	ControlWord	LINT	0	Prepared value	Address	comment		E.
Application [run]	Controlword     StatusWord	UINT	64					
1 Ubrary Manager	RefPos	DINT	0					
PLC_PRG (PRG)     Task Configuration	ActPos	DINT	0					
	OpMode	BYTE	1					
EtherCAT_Master.EtherCA		DINT	10					
PLC PRG	· · · · · · · · · · · · · · · · · · ·	t parts of information out			1			
O a per       O Kee pertox_AB (GriCo AB)       O A conserte       O A conserte device       C A conserte device       C Accord a device	10 //and switch between a 11 DriveEnInput ALSE := 1 12 DriveDIInput ALSE := 1 13 14 //drive is to enabled	uts which have been receive mabled and disabled drive DigitalInpute[0].0FALS; JojitalInpute[0].1FALS; after a rising edge at Ini led after a rising edge at Ini led after a rising edge at Ini led After a rising edge at Ini Set Status; SE AND NOT DriveDilast[ALSE] T SE AND NOT DriveDilast[ALSE]	In2 Hen					100 %
	Watch 1							
	Expression		Application	Туре	Value	Prepared value	Execution point	

Figure 24 Debugging view of PLC\_PRG - Application running



Startup of the EtherCAT subsystem will take a few moments. If the EtherCAT does not start up after a download/start a reset of the controller (Online/Reset warm) might help. Otherwise please check, whether the device in the project really is the one connected to EtherCAT.

We can logout again using Online/Logout. The application can be explicitly stopped and restarted using the Debug menu.

Please note: The downloaded application will remain in the RasPi environment. It will be restarted automatically after the next reset of the RasPi.



# **Example Application**

## **Overview**

The purpose here is to start the drive in reaction to a digital input and cyclically move between two positions while also cyclically changing the profile parameters for acceleration and deceleration. The used OpMode is ProfilePosition Mode (0x6060.00 = 1).

This requires enabling the controller in a first step and then sending the target positions, waiting for being in position and updating the profile parameters. So there are several steps to be taken. A well suited solution pattern for such a problem is a step sequence. A step sequence is a pattern, where only a part of the PLC program is executed in each update cycle, depending of the step in which the program is. There is a special diagram to design these step sequences: sequential function chart (SFC). Here however we use ST and implement the step sequence using a

```
If (StepVariable = xxx) Then
...
ELSEIF (StepVariable = xxx) Then
...
END_IF
```

## Implement the steps.

Each step may have actions to be taken only at the entry into a step, code to be executed while within a step and actions to be taken when switching to the next step. Obviously there should also be a condition when to switch to the next step. And there can be different conditions to branch into different next steps.





Examples on how to create a step sequence in a CODESYS environment using sequential function chart (SFC) can be found at www.youtube.com.

The main sequence of this application is shown in the flowchart of Figure 26. After preprocessing the inputs (control word and digital inputs) a first step sequence deals with the finite state machine of the servo drive according to CiA 402.

The control word is used to step the MotionController from the initial disabled state to fully operational state. The commands in the controlword depend on the statusword received from the drive unit.

The main idea is: If we have been in disabled state (stored in the variable DriveEnabled) and the enable switch at the discrete I/Os (DigIn1) of the MotionController is pressed, we send a sequence of enabling commands to the MotionControllers controlword. Steps are taken depending on the state signaled by the statusword.

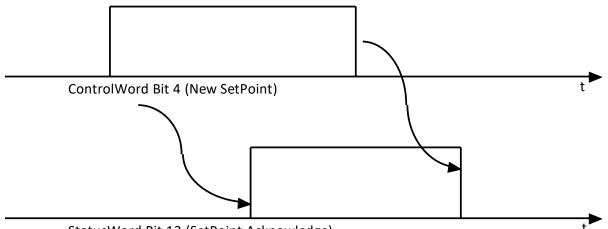


On the other hand, if we are already enabled and receive a disable command from Digln2, send a shutdown command to the controller to disable the operation once again.

Only after the state machine reached the operation enabled state, the DriveEnabled variable is set to true and the second step sequence for updating the position references is executed.

So finally – when the drive is enabled – the position references are sent to the drive. The first position reference of a sequence is always sent as an immediate value – which will be processed by the drive without waiting for any earlier commands to be finished. All subsequent references are sent only if the previous one has been reached.

Again there is a handshake between the PLC and the MotionController using bits in the controword and the statusword. The handshake between the PLC and the servo drive is shown in Figure 25. After a new reference position has been set in 0x607A.00 Target Position the New Setpoint bit in the controlword 0x6040.00 has to be set (in fact a rising edge is required to start the move). Only after this rising edge is received by the drive the new reference is acknowledged via the setpoint acknowledge bit in the statusword and the drive starts the move. It is Important to release the New Setpoint bit once again to create a next rising edge for the next position.



StatusWord Bit 12 (SetPoint Acknowledge)

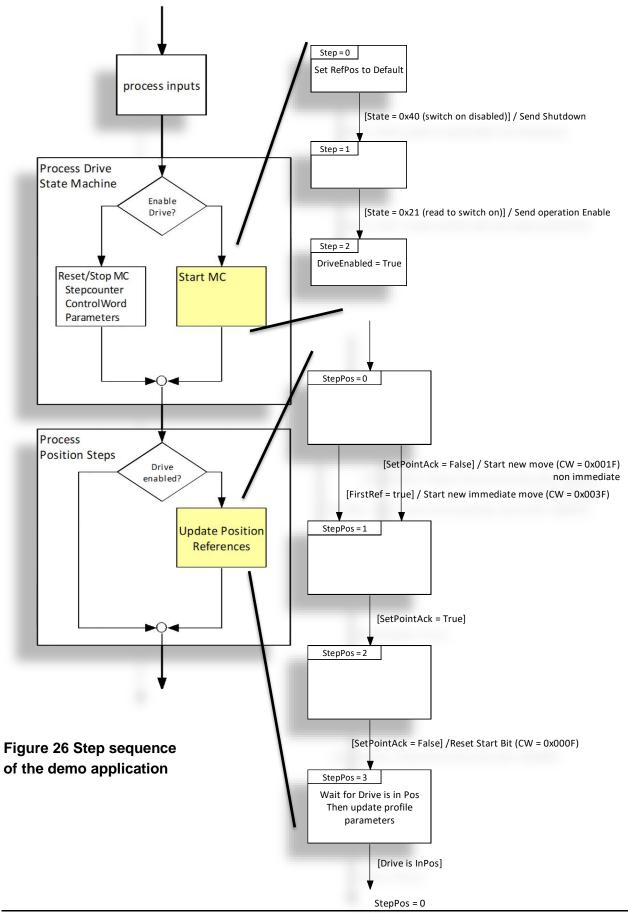
## Figure 25 Handshaking sequence between the PLC (ControlWord) and the servo drive (StatusWord)

If the new setpoint is flagged as an immediate one (bit 5 of the controlword), the new command is immediately executed even if a previous command is still ongoing. This feature is used here for the very first command.

All subsequent commands are sent, only if the drive has reached the previous target position and signaled this by setting the Target Reached bit in the statusword.

So once again a step sequence is well suited to organize the different steps of this interaction.







# Listing

## Variables

This part is displayed in the upper part of the editor window and holds the definition of all variable instances.

```
VAR
      ControlWord: UINT:=0;
      StatusWord: UINT;
      RefPos : DINT:= 0;
      ActPos : DINT;
      OpMode : BYTE := 1;
      Acc: DINT:=10;
      Dec: DINT:=10;
      ProfileSpeed: DINT:= 5000;
      DriveEnable : BOOL:= FALSE;
      FSMState: BYTE;
      IsInPos: BOOL;
      SetPointAck: BOOL;
      DriveEnabled: BOOL:=FALSE;
      DigitalInputs : BYTE;
      DriveEnInput : BOOL;
      DriveEnLast : BOOL := FALSE;
      DriveDiInput : BOOL;
      DriveDiLast : BOOL := FALSE;
      FirstRef: BOOL:=TRUE;
      StepEn: INT:=0;
      StepPos: INT:=0;
      AccDecSlope: INT:=0;
END_VAR
VAR CONSTANT
      MaxAccDec:DINT:=500;
      MinAccDec:DINT:=50;
      DeltaAccDec:DINT:=10;
```

#### END\_VAR



# Code

The code is displayed in the lower part of the editor window and contains the functional logic of the application.

```
//extract the different parts of information out of the status word
//StatusLowByte is used for switching the CiA 402 State machine
FSMState := WORD TO BYTE(StatusWord);
//Flags of the PP mode
IsInPos := StatusWord.10;
SetPointAck := StatusWord.12;
//eval the digital inputs which have been received from the MC
//and switch between enabled and disabled drive
DriveEnInput := DigitalInputs.0;
DriveDiInput := DigitalInputs.1;
//drive is to enabled after a rising edge at In1
//drive is to be disabled after a rising edge at In2
IF (DriveEnInput AND NOT DriveEnLast) THEN
     DriveEnable := TRUE;
ELSIF (DriveDiInput AND NOT DriveDiLast) THEN
     DriveEnable := FALSE;
END IF
//Store the flags to detect edges in the next step
DriveEnLast := DriveEnInput;
DriveDiLast := DriveDiInput;
//step through the CiA 402 drive state machine
//implemented as a step sequence station at StepEn = 0
IF (DriveEnable) THEN
     IF (StepEn = 0) THEN
            RefPos:=10000;
            IF (FSMState = 16#40) THEN
                  //if state is switch on disabled, send the switch on
                 ControlWord := 16#0006;
                 StepEn := 1;
            END IF
     ELSIF (StepEn = 1) THEN
            IF (FSMState = 16#21) THEN
                 //if state is switched on, send enable operation
                 ControlWord := 16#000F;
                 StepEn := 2;
            END IF
     ELSIF (StepEn = 2) THEN
            IF (FSMState = 16#27) THEN
                                                            denote the drive
                  //if state is operation enabled,
                 //being enabled
                 DriveEnabled := TRUE;
```



```
END IF
     END IF
ELSE
     //drive shall be switched of or stay switched of
     //reset all variables to default values
     ControlWord := 0;
     StepEn := 0;
     DriveEnabled := FALSE;
     StepPos := 0;
     FirstRef := TRUE;
END IF
//step sequence with the different position commands in PP mode
//first command is issued immediate
//the following commands are issed after having reached the previous target
//PP steps are sent only, if drive is enabled
IF (DriveEnabled) THEN
     //step 0: signal a new ref. This is either the default start value
     // set during enabling the drive
     //or the one calculated in StepPos step 3
     IF (StepPos = 0) THEN
           //if this is the first step after boot, this is sent flagged
           //as an immediate reference
           IF (FirstRef) THEN
                 ControlWord := 16\#003F;
                 StepPos := 1;
                 FirstRef := FALSE;
           ELSE
                 //otherwise (not the first step) this a non immediate ref
                 IF (SetPointAck) THEN
                       //new ref has been acknowleged, so reset
                                                                          the
                       //new setpoint flag until the acknowledge is reset
                       ControlWord := 16#000F;
                 ELSE
                       //signal the new ref and step to the next step
                       // only, if the Ack Bit has not been set
                       ControlWord := 16#001F;
                       StepPos := 1;
                 END IF
           END IF
     ELSIF (StepPos = 1) THEN
           //last action in step 0 was, to signal the new ref, so we have
           // to wait, until the ref has been acknowlegded
           //waiting for acknowledge of pos
           //only then we can switch to the next step
           IF (SetPointAck) THEN
                 StepPos := 2;
           END IF
     ELSIF (StepPos = 2) THEN
```



```
//setpoint has been acknowleded, new setpoint bit can be
           // reset now
           //we should wait until the ack bit has been reset too
           ControlWord := 16#000F;
           IF (NOT SetPointAck) THEN
                 StepPos := 3;
           END IF
     ELSIF (StepPos = 3) THEN
           //wait for InPos
           IF (IsInPos) THEN
                 //new pos has been reached
                 //now inivert the ref and change the profile parameters
                 // for the next step
                 RefPos :=(-RefPos);
                 //update profile Values
                 IF (AccDecSlope > 0) THEN
                       //we are still accelerating
                       IF (Acc < (MaxAccDec - DeltaAccDec)) THEN
                            Acc := Acc + DeltaAccDec;
                            Dec := Acc;
                       ELSE
                            AccDecSlope := 0;
                       END IF
                 ELSE
                       //we slow down the dynamic
                       IF (Acc > (MinAccDec + DeltaAccDec)) THEN
                            Acc := Acc - DeltaAccDec;
                            Dec := Acc;
                       ELSE
                            AccDecSlope := 1;
                       END IF
                 END IF
                 //new
                            profile
                                          values are calculated,
                 //we do now reset the sequence
                 StepPos := 0;
           END IF
     END IF
END IF
```



# Assignment of PLC variables to the process image

≝ ≝ - ≝ - ∞ - × = ≋ × + # \$;  ≞  ≊ - f   ⊞								
evices v 4 X	Device Ether	CAT_Master PLC_PRG 🕅 OpMode	DcSynchronoi	us 🍸 MC5005 🗙				
DefaultDemoET_ST     Device (CODESYS Control for Raspberry Pi SL)	General	Find	Filter Sho	w all				
	General				_			
	Expert Process Data	Variable	Mapping	Channel	Address	Туре	Unit	Description
Library Manager		Application.PLC_PRG.ControlWord	۵,	control word	<del>%QW0</del>	UINT		control word
PLC PRG (PRG)	Process Data	Application.PLC_PRG.RefPos	<b>*</b>	target position	%QD1	DINT		target position
Task Configuration		Application.PLC_PRG.OpMode	<b>~</b>	Modes of operation	%QB8	USINT		Modes of operation
B S MainTask	Startup Parameters	Application.PLC_PRG.ProfileSpeed	۵.	Profile velocity	%QD3	UDINT		Profile velocity
EtherCAT_Master.EtherCAT_Task		Application.PLC_PRG.Acc	۵	Profile acceleration	%QD4	UDINT		Profile acceleration
	EtherCAT I/O Mapping	Application.PLC_PRG.Dec	<b>~</b>	Profile deceleration	%QD5	UDINT		Profile deceleration
PLC_PRG		Application.PLC_PRG.StatusWord	۵.	status word	%IW0	UINT		status word
Green CAT Master (Ether CAT Master)	Status	🕸 - 🍫		Modes of operation display	%IB2	USINT		Modes of operation displ
OpModeDcSynchronous (Operation mode DC-Synchronous)	Information	Application.PLC_PRG.DigitalInputs	<b>~</b>	Digital input logical state	%IB3	USINT		Digital input logical state
SoftMotion General Axis Pool	Information	👜 - 🎭		Position actual value	%ID1	UDINT		Position actual value
<ul> <li>Sortiviouon General Axis Pool</li> <li>I2C</li> </ul>		🖮 - 🍫		velocity actual value (in rpm)	%ID2	DINT		velocity actual value (in i
SPI		ii. 🐌		Torque actual value	%IW6	UINT		Torque actual value
GPIOs_A_B (GPIOs A/B)     Onewire								
E GPIOs_A_B (GPIOs A/B)								
E GPIOs_A_B (GPIOs A/B) S Onewire S Camera device		<i>«</i>						
E FIOs_A_B (GPIOs A/B) - S Onewire ⊡ S Camera device		<	Reset	m mapping Always update va	iriables: Use p	arent device	setting	
E GFIOs_A_B (GFIOs A/B) → b Onewire ⇒ b Camera device		/	Reset		iriables: Use p	arent device	setting	
E GFIOs_A_B (GFIOs A/B) → b Onewire ⇒ b Camera device		IEC Objects			rriables: Use p	arent device	setting	
E GPIOs_A_B (GPIOs A/B) → b Onewire → b Camera device		IEC Objects Variable Mapping	g Type	Mapping Always update va	riables: Use p	arent device	setting	
E GFIOs_A_B (GFIOs A/B) → b Onewire ⇒ b Camera device		IEC Objects		Mapping Always update va	iriables: Use p	arent device	setting	
E GrIOs_A_B (GPIOs A/B) → Ъ Onewire ⇒ Ъ Camera device		IEC Objects Variable Mapping -   MC5005	g Type	Mapping Always update va	riables: Use p	arent device.	setting	

# Variants

## Add a homing sequence

The example above was about doing position control using PP mode. However there was no homing sequence used to reset the actual position to a defined position.

Different types of homings are available for the FAULHABER MotionController. Here we might use homing on a lower limit switch: 0x6098.00 homing method = 17. As the homing method usually does not change for a given application the method can be set via the MotionManager during system configuration and saved as a part of the systems application parameters.

However we need to add some additional steps in our sequence:

- 1. Switch OpMode to 0x6060.00 = 6: Homing
- Start the homing sequence by setting the start bit in the controlword 0x6040.00 0x6040.00 ← ControlWord = 0x00 1F
- 3. Wait for the homing being finished successfully by monitoring Bit 12 in the statusword 0x6041.00 if(0x6041.00 → StatusWord.12 = true) Then ...
- Save the successfully executed homing in a local flag, switch OpMode to 0x6060.00 = 1 (PP) and start the UpdatePosition sequence according to Figure 26

As we already covered the OpMode in the mappings of Table 3 we don't even need to edit the PDO mappings in this step.



## EEPROM Update

As mentioned the revision number stored in the EtherCAT slave (ESC) EEPROM should be identical to the revision of the firmware. Therefore after a FW update it is necessary to update the data stored in the ESC EEPROM too.

This can be done, when being connected to a running application or using the Online Config mode. In both cases the update is done using the Online tab of the MotionController settings. Simply press Write EEPROM XML will download the information out of the XML file used during system definition step.

DefaultDemoET_ST.project - CODESYS			
Eile Edit View Project Build Online Debug Tools Window	Help		
🎦 🚅 🛃 🗐 い つ 🎖 脂 🛍 🗙 🛤 🏭 🛅 🖷	🖞 🛗 I 🥨 💜 🕠 🖕 💘 I 💷	ia ta 8   o   <b>∭   ≓</b>	
Devices 👻 🕂 🗙	Device EtherCAT_Ma	er PLC_PRG 🙀 OpModeDcSynchronous 🥂 MC5005 🗙	<b>*</b>
DefaultDemoET_ST			
B Ovice [connected] (CODESYS Control for Raspberry Pi SL)	General	State Machine	
- I PLC Logic - Application [run]	Expert Process Data	Init Bootstrap Current State: Operational	
Library Manager		Pre-Op Safe-Op Requested State: Operational	
PLC_PRG (PRG)	Process Data	Op	
🖃 🙀 Task Configuration	Startup Parameters		
🖹 😻 MainTask	Startup Parametera	File access over EtherCAT	
	Online	Download Upload	
G      G	CoE Online	E2PROM Access Write E2PROM Read E2PROM Write E2PROM XML	
OpModeDcSynchronous (Operation mode DC-Sy     SoftMotion General Axis Pool	EtherCAT I/O Mapping		
	Status		
GPIOs_A_B (GPIOs A/B)	Information		
- 😏 🚡 Onewire			
🖹 😏 🏅 Camera device			
<pre>K <empty> (<empty>)</empty></empty></pre>			
	Watch 1		<b>▼</b> ∓ X
	Expression	Application Type Value Prepared value	Execution point Ac
۰ III ا	•	m	۶.
2 Devices POUs	Watch 1 D Breakpoints		
Messages - Total 0 error(s), 0 warning(s), 5 message(s)			
Last build: 🕴	0 😗 0 Precompile: 🧹 🔒	RUN Program loaded Program unchanged	Current user: (nobody)

## Adding a Visualization

CODESYS allows easily adding a target visualization. Examples can be found at youtube. Visualization is added via the context menu off the Application (Figure 27). The visualization is then added as a separate component to the application tree (Figure 28). A separate task is created also in this step.

The controls for numeric values, switches or lamps can then be added and edited via drag and drop out of the visualization toolbox. Of course, you will need to connect them to variables of the application. This is a convenient way to add a user interface even if no external switches are available in a demo environment. The visualization can both be displayed and used in the CODESYS engineering environment or directly using a web browser:

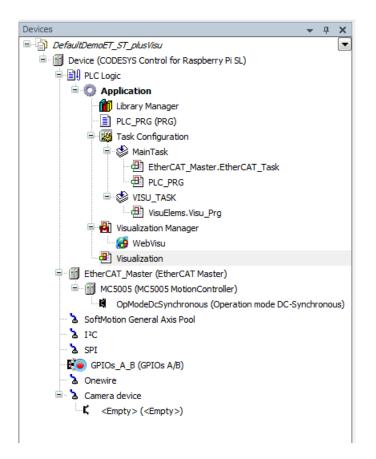
The target address is <IP number of the target controller>:8080/webvisu.htm e.g.:

### 192.168.0.45:8080/webvisu.htm



DefaultDemoET_ST_plutVisu.project - CODESVS			
file Edit Yew Broject Build Online Debug Tools Window Help			
9000	New PET COLORATION IN	CALLER & LIN THE LAW	
	The state of the s	1.1.2.0.10.10014	
Devices + 9 X	Device i Ether	CAT_Master 🖬 OpModeDcSynchronous 👔 PLC_PRG 🖉 MC5005 🗙 🌚 MainTask	
DefaultDemoET_ST_plusViau	General	Address Additional	
Device (CODESYS Control for Raspberry PI SL)     Device (CODESYS Control for Raspberry PI SL)     Device (CODESYS Control for Raspberry PI SL)	General	Fibercorry	
· Hit Pic Loge	Expert Process Data	(i) change of a change	
Appl X Cut	C. C	EtherCAT Address 1001	
al ala copy	Process Data		
🖬 📻 🏾 🔃 Paste	Contraction of the second	# Distributed Clock	
= 6 × Delete	Startup Parameters	Select DC DC Synchronous •	
Browse +	EtherCAT I/O Mapping	IV Enable 2000 Sync Unit Cycle (µs)	
EtherCAT Refactoring	20000	Sync0:	
	Statue	V Enable Sync 0	
B MCS R Properties	Selarmettan.	Sync Unit Cycle 1	
SoftHoto Add Object Marm confi	guration		
S IPC Add Faider O Application		🗇 User Defined 0 🔄 Shift Time (µs)	
- SPI 📑 Edit Object 🐼 Axis Group		Sync1:	
Edit Object With 6 Cam table.		Enable Sync 1	
3 Onesire		(i) Sync Unit Cycle 1 = 2000 + Cycle Time (µs)	
B & Camera d			
K <em application="" cnc="" delete="" device="" from="" setting<br="">Data Source</em>		🗇 User Defined 🛛 👘 Shift Time (µs)	
ar Dura Source	es manager	Startup checking	
External Fi		DC cyclic unit control: assign to local µC	
Global Varia		1 Watchdog	
Image Pool		Identification	
-O Interface	15	Disabled	
Setwork Ve	riable List (Receiver)	Configured Station Alias (ADO 0x0012) Value 1001	
Network Ve	risble List (Sender)		
T Persistent	(ariables		
d3] POU		<ul> <li>Explicit Device Identification (ADO 0x0134)</li> </ul>	
(B) POU far im	wit checks	Data Word (2 Bytes)     ADO (hex)	
Recipe Mar			
	v Configuration		
Symbol Cor			
Text List			
age Trace			
Trend reco	rding manager		
CI Unit conver	sion		
() Visualizatio	n		
👸 Visualizatio	n Manager		
Devices 🗋 POLIs			
Messages - Total 0 error(s), b warning(s), 5 message(s)			Last build: O 0 • 0 Precomple:  Current user: (nobody)
🚱 🤄 💌 🗭 🧕 🔾 ビ 🔳 🛥	Z 🔧 👿 🔹		15 11 🔹 🐨 🖛 전 🖸 💐 🖉 🕲 🕲 🖉 🎘 🕲 👘 1 41 🕼 19 - 16 8 🖉

Figure 27 Add a visualization to an application



## Figure 28 Application tree of a demo unit having target visualization



# **Additional Resources**

FAULHABER Application Notes		
App-Note 150	How to control a FAULHABER MC V3.0 ET using TwinCAT	
App Note 154	Updating the EtherCAT slave EEPROM using TwinCAT	



FAULHABER motion controllers: https://www.faulhaber.com/en/products/drive-electronics/motioncontrollers/



FAULHABER motion demos at Youtube: https://www.youtube.com/playlist?list=PLt9bbvNtjYyiKxuh\_XtoYGB5Zx2

<u>q04k3D</u>



#### **Rechtliche Hinweise**

**Urheberrechte**. Alle Rechte vorbehalten. Ohne vorherige ausdrückliche schriftliche Genehmigung der Dr. Fritz Faulhaber & Co. KG darf insbesondere kein Teil dieser Application Note vervielfältigt, reproduziert, in einem Informationssystem gespeichert oder be- oder verarbeitet werden.

**Gewerbliche Schutzrechte**. Mit der Veröffentlichung der Application Note werden weder ausdrücklich noch konkludent Rechte an gewerblichen Schutzrechten, die mittelbar oder unmittelbar den beschriebenen Anwendungen und Funktionen der Application Note zugrunde liegen, übertragen noch Nutzungsrechte daran eingeräumt.

Kein Vertragsbestandteil; Unverbindlichkeit der Application Note. Die Application Note ist nicht Vertragsbestandteil von Verträgen, die die Dr. Fritz Faulhaber GmbH & Co. KG abschließt, soweit sich aus solchen Verträgen nicht etwas anderes ergibt. Die Application Note beschreibt unverbindlich ein mögliches Anwendungsbeispiel. Die Dr. Fritz Faulhaber GmbH & Co. KG übernimmt insbesondere keine Garantie dafür und steht insbesondere nicht dafür ein, dass die in der Application Note illustrierten Abläufe und Funktionen stets wie beschrieben aus- und durchgeführt werden können und dass die in der Application Note beschriebenen Abläufe und Funktionen in anderen Zusammenhängen und Umgebungen ohne zusätzliche Tests oder Modifikationen mit demselben Ergebnis umgesetzt werden können.

**Keine Haftung**. Die Dr. Fritz Faulhaber GmbH & Co. KG weist darauf hin, dass aufgrund der Unverbindlichkeit der Application Note keine Haftung für Schäden übernommen wird, die auf die Application Note zurückgehen.

Änderungen der Application Note. Änderungen der Application Note sind vorbehalten. Die jeweils aktuelle Version dieser Application Note erhalten Sie von Dr. Fritz Faulhaber GmbH & Co. KG unter der Telefonnummer +49 7031 638 345 oder per Mail von mcsupport@faulhaber.de.

#### Legal notices

**Copyrights**. All rights reserved. No part of this Application Note may be copied, reproduced, saved in an information system, altered or processed in any way without the express prior written consent of Dr. Fritz Faulhaber & Co. KG.

**Industrial property rights**. In publishing the Application Note Dr. Fritz Faulhaber & Co. KG does not expressly or implicitly grant any rights in industrial property rights on which the applications and functions of the Application Note described are directly or indirectly based nor does it transfer rights of use in such industrial property rights.

**No part of contract; non-binding character of the Application Note**. Unless otherwise stated the Application Note is not a constituent part of contracts concluded by Dr. Fritz Faulhaber & Co. KG. The Application Note is a non-binding description of a possible application. In particular Dr. Fritz Faulhaber & Co. KG does not guarantee and makes no representation that the processes and functions illustrated in the Application Note can always be executed and implemented as described and that they can be used in other contexts and environments with the same result without additional tests or modifications.

**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 in connection with it.

**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.