CODESYS Control for Raspberry Pi (Demo)

1 General information

Order number: 000051	**Supplier information** 3S-Smart Software Solutions GmbH Memminger Straße 151 87439 Kempten Germany Support: Tel: +49 831 54031 66 support@codesys.com
Version: 1.1.0.0	
Short description CODESYS Control for Raspberry Pi (Demo)	

2 Requirements and restrictions

Programming system	CODESYS Development System Version 3.5.4.10 or higher
Target system	CODESYS Control Version 3.5.4.10 (this is part of this product)
Supported Platforms / Devices	Raspberry Pi (s. http://www.raspberrypi.org/)
Additional requirements	SD-card (minimum 4GB)
Restrictions	Runtime limitation (2 hours)

3 Price

This example is for free.

4 Required accessory, purchasable in the CODESYS Store.

-

5 Product description

This product contains a runtime limited CODESYS Control application for Raspberry Pi (see http://www.raspberrypi.org/) as well as driver support for the extension modules Raspberry PiFace Digital, Raspberry Pi Camera and several devices/breakouts with I²C communication interface.

This product is delivered as SD card image based on a Linux distribution (Raspbian). This image must be copied onto the memory card of the device. After starting Raspberry Pi, CODESYS Control runs for two hours without functional limitations and shuts down automatically.

The runtime system does not have realtime behavior. Its Jitter depends on many factors, especially on parallelly executed Linux applications, and is ideally about 50µs with maximum values of 400µs.

This product supports the following functionalities

- EtherCAT Master
- Modbus TCP Master and Slave
- Ethernet/IP Scanner
- Web Visualization
- SoftMotion CNC

This product consists of:

- SD card image for operating system and CODESYS Control
- CODESYS device description files for Raspberry Pi, Raspberry PiFace Digital, Raspberry Pi Camera, several devices/breakouts with I²C communication interface (SRF02, Adafruit PWM, MPU6050, MPU9150, AK8975)

This product offers amongst other things the possibility to plug and control additional devices via SPI or I²C.

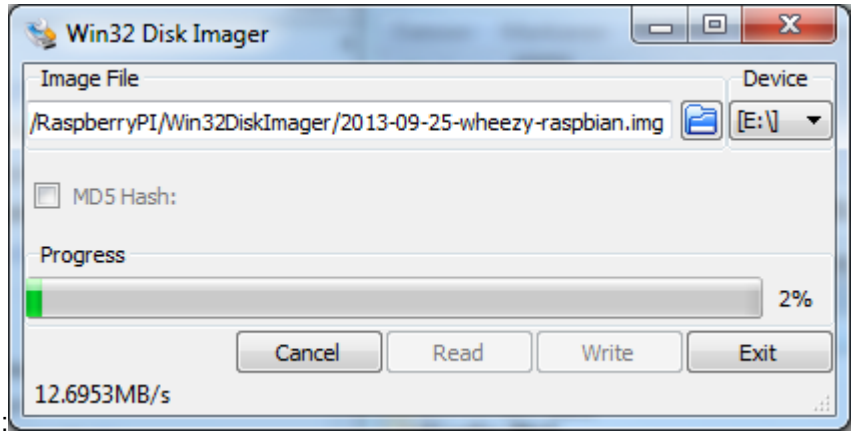
This product is provided for testing and training purpose and may not be used in the field.

6 Technical description

Preparations

1. Download and unzip the product zip file from CODESYS Store
2. Install the included package file inside CODESYS (Menu Tools -> Package Manager -> Install). Note down the installation path.
3. Generate a bootable SD card

Under Windows, the application *Win32diskimager* (download from http://sourceforge.net/projects/win32diskimager/) can be used to download the delivered image onto your SD card:



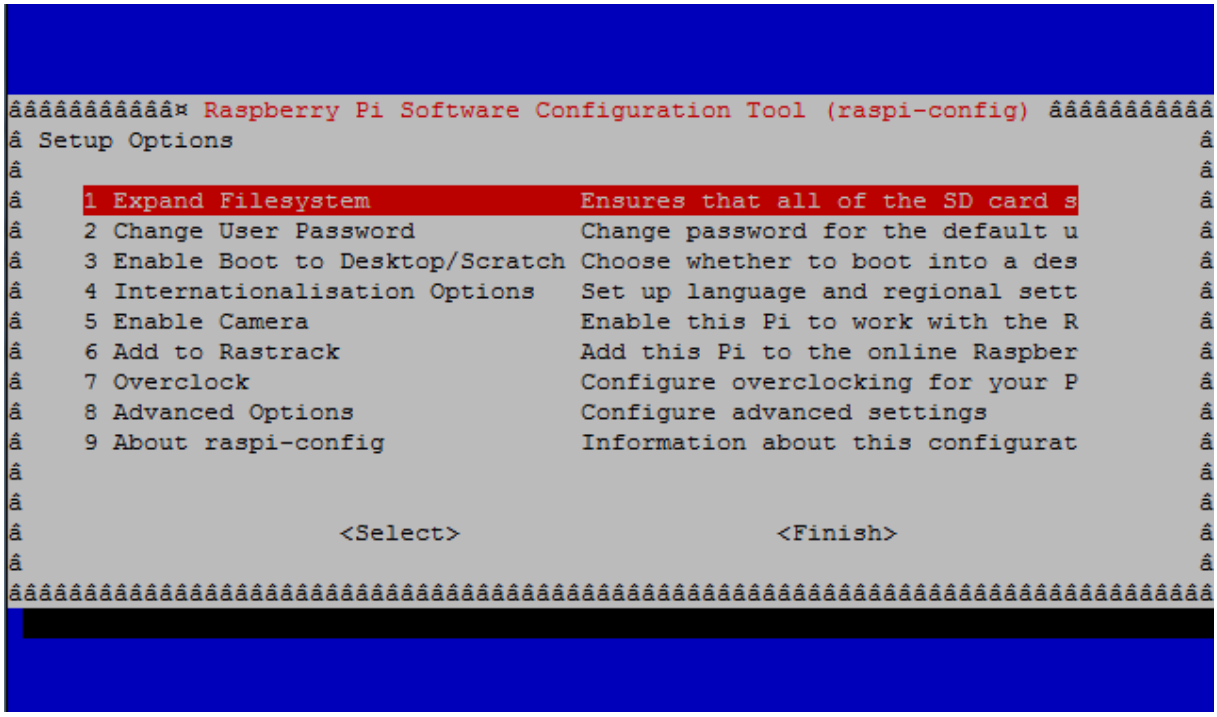
:

Reboot Raspberry Pi with this SD card afterwards. In order to obtain a valid IP address, the device should be connected to a network with a DHCP server. To find out this address, you may either plug a keyboard, mouse and a monitor to your device, log in (user: `pi`, password: `raspberry`), open a console and issue the command `ifconfig`.

Alternatively, you can start CODESYS and proceed like explained in the first example application (see below), without logging into the device and starting the application. Knowing the IP address you can also log in with Windows remote desktop.

4. Expand the file system according to the size of your SD card

From a console run `sudo raspi-config` and execute the following action:



5. After a reboot of your device it is ready for use.

Example applications

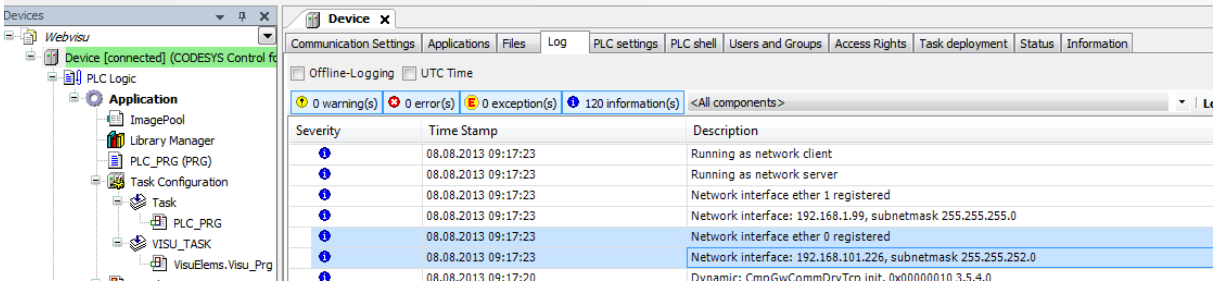
In the installation path (that you have noted down during installation of the package) you will find the following example projects:

1. **Webvisu.project**

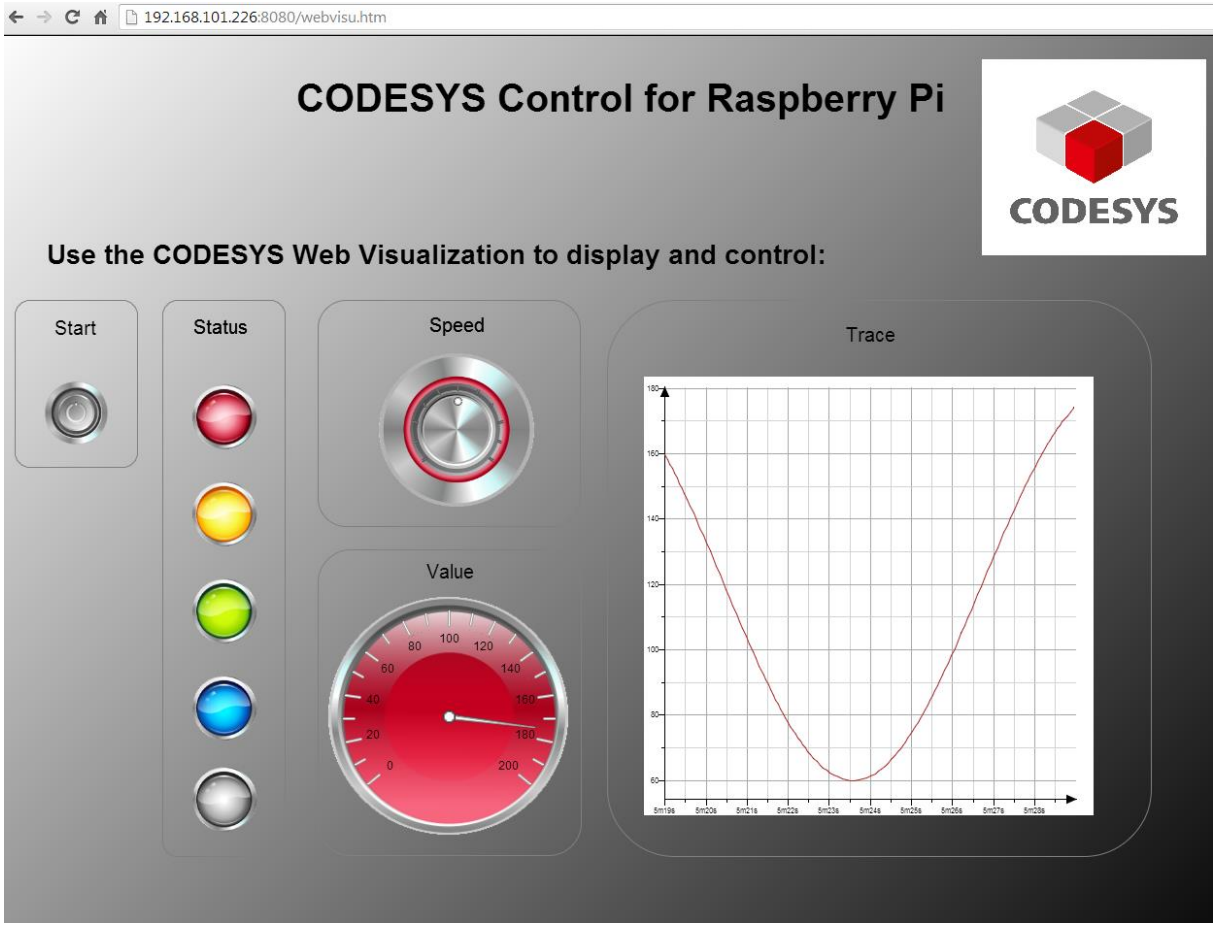
This example shows how CODESYS web visualization can be used. Open the project and download it onto your Raspberry Pi by doubleclicking on the node “Device” in the device tree (left hand side). Then press “Scan network” in the communication dialog tab and select your device that should now appear under the name “RaspberryPi”, if the device is in the same network with your programming PC. Select it and run “log in” from the menu “Online”. Then start the application with F5.

Start an internet browser (possibly also on your smartphone) and connect to <Netzwerk-Adresse>:8080/webvisu.htm.

You can find out the IP address of your device from the log outputs on your device:



In your browser you will see the visualization that has been designed in the project:



2. **Camera.project (precondition: Raspberry Pi Camera is connected)**

This project shows the usage of Raspberry Pi Camera (extension hardware) from your PLC:

Download the application to your controller, start the program and set the variable `xTakePicture` to `TRUE`. The camera will take a picture and store it in the local file system with the name „Picture.jpg“.

You can copy this file onto your programming PC with the help of the file dialog (doubleclick “Device”, tab “Files”, Button “Update” on the right side etc.).

3. GPIO.project

This project shows you can use free GPIOs. In the configuration of the GPIO device in the device tree the function of each GPIO can be defined:

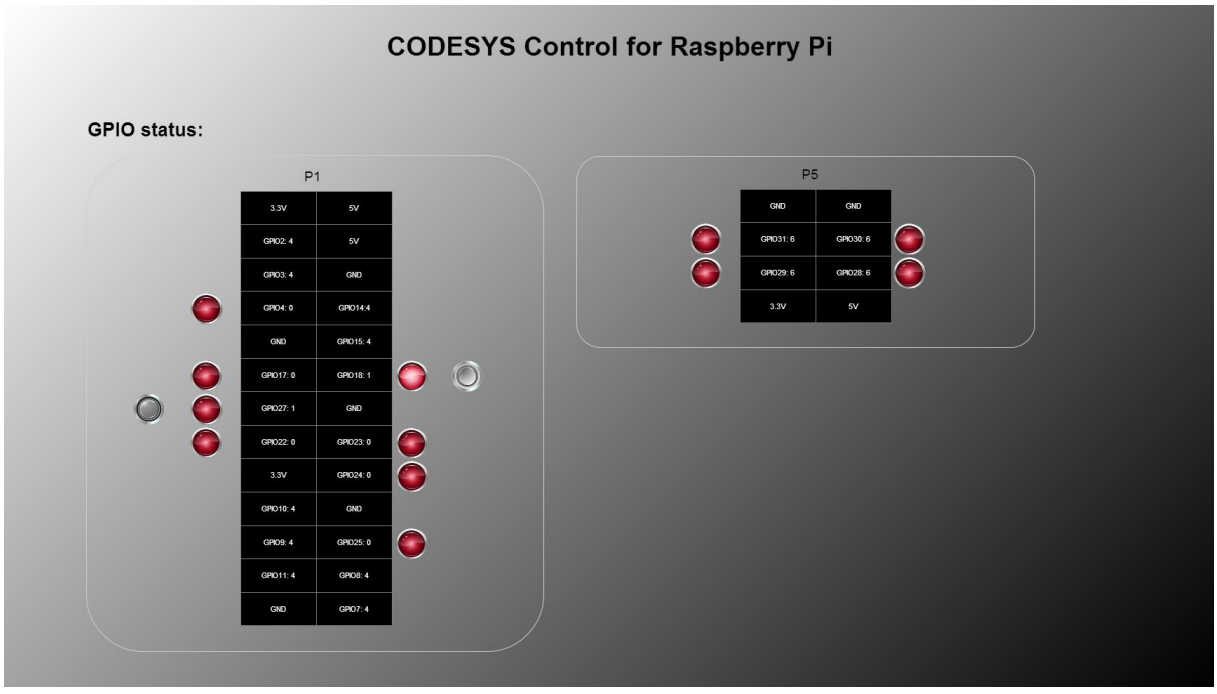
GPIOs

GPIOs Configuration | GPIOs I/O Mapping | Status | Information

Parameter	Type	Value	Default Value	Unit	Description
GPIO4	Enumeration of BYTE	not used	not used		configuration of GPIO4
GPIO17	Enumeration of BYTE	not used	not used		configuration of GPIO17
GPIO18	Enumeration of BYTE	Output	not used		configuration of GPIO18
GPIO22	Enumeration of BYTE	not used	not used		configuration of GPIO22
GPIO23	Enumeration of BYTE	not used	not used		configuration of GPIO23
GPIO24	Enumeration of BYTE	not used	not used		configuration of GPIO24
GPIO25	Enumeration of BYTE	not used	not used		configuration of GPIO25
GPIO27	Enumeration of BYTE	not used	not used		configuration of GPIO27
GPIO28	Enumeration of BYTE	not used	not used		configuration of GPIO28
GPIO29	Enumeration of BYTE	not used	not used		configuration of GPIO29
GPIO30	Enumeration of BYTE	not used	not used		configuration of GPIO30
GPIO31	Enumeration of BYTE	not used	not used		configuration of GPIO31

The inputs and outputs are available as DWORD in the tab „GPIOs I/O Mapping“. Bit <X> of the DWORD correlates with GPIO <X>.

In this example GPIO18 is used as output and blinks, controlled by a timer FB in PLC_PRG. A visualization screen displays the input values of the GPIOs and allows to set outputs.



4. PiFace.project (precondition: Raspberry PiFace Digital is connected)

This example shows the usage of Raspberry PiFace Digital (8 digital inputs and outputs).

Open the project, download it to the controller and start it. The simple application in `PLC_PRG` controls the relay output K0 depending on the key button S1 (on-switching of K0 is delayed for 1s) and the relay output K1 depending on button S2 (off-switching is delayed for 500ms).

Please note that this driver allows to connect more than one PiFace devices (hardware address is set with the jumpers JP1, JP2) by setting the corresponding parameter in the PiFace device in the device tree.

The library SPI_PiFace that operates as driver is provided as source code and can be seen as example how to communicate to other devices via SPI. The communication bases are on the library RaspberryPiPeripherals, for which a reference documentation is provided (see online help (F1) -> Libraries).

5. PiFaceIoDrv.project (precondition: Raspberry PiFace Digital is connected)

This example is similar to the one before. Instead of providing the I/O data as inputs of an automatically instanced function block, the example implements the data exchange as standard PLC IO driver via a process image, how it is typically done on PLCs.

The driver library IoDrvPiFace.library is provided as source code.

6. I2CExample.project (precondition: special hardware is connected via I²C)

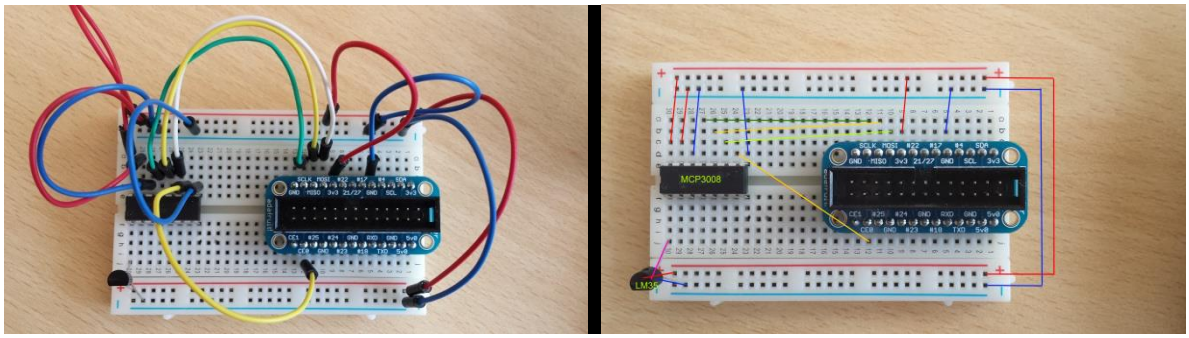
This example communicates with the following breakouts via I²C:

- Adafruit 16-channel/12-Bit PWM
- SRF02 (supersonic distance sensor)
- Drotek IMU 9DOF - MPU9150 (gyroskope, accelerometer, compass)

The libraries I2C_* that implement the data exchange are provided in source code and can be used as example for additional interface connections. The communication bases on the library RaspberryPiPeripherals, for which a reference documentation is provided (see online help (F1) -> Libraries).

7. MCP3008_Temperature.project (precondition: special hardware is connected via SPI)

This example shows how the value of an analog temperature sensor (LM35), that is connected to an A/D converter chip (MCP3008), can be read in CODESYS via SPI. MCP3008 can process 8 analogue channels. In this example we use only one of these. The following test installation is used:



The library SPI_MCP3008.library that implements the data exchange is provided in source code and can be used as example for additional interface connections. The communication bases on the library RaspberryPiPeripherals, for which a reference documentation is provided (see online help (F1) -> Libraries).

8. SoftMotion Servo Example (precondition: an Adafruit 16-channel/12Bit PWM circuit board is connected via I²C. On its first PWM channel a servo motor is connected)

This example shows, how CODESYS SoftMotion can be used with modelbuilding servo motors. An additional circuit board, Adafruit ID 815, acts as communication interface.

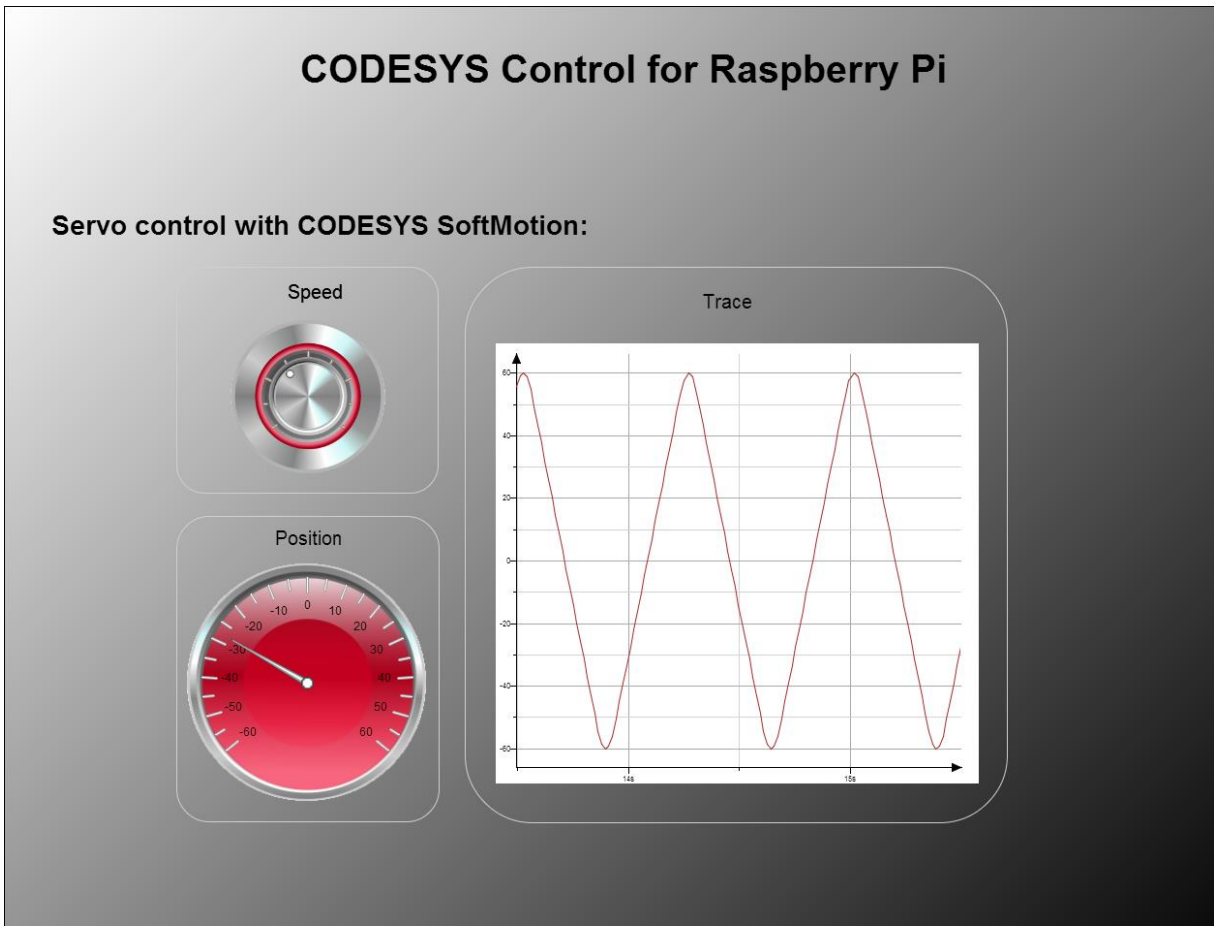
Open the project, download it to your Raspberry Pi. The motor starts continuously to turn. This has been programmed in SFC in PLC_PRG, which first enables the axis and then moves it between the positions -60 and +60, which had been configured as limit positions.

The set position command is transmitted to the servo via a PWM interface. In a fixed cyclic period (default: 50Hz, parameter of device Adafruit PWM Softmotion) a HIGH pulse is generated with a duration between 1ms to 2ms. 1ms represents the lower, 2ms the upper position limit. The movement range differs between servo types. To control the drive positions e.g. in degrees, one needs to measure the position range by moving the servo to its limits and enter the measured positions in the configuration screen of the axis device:

SoftMotion Drive: Basic | SM_Drive_Servo: Configuration | SM_Drive_Servo: I/O Mapping | Status | Information

Parameter	Type	Value	Default Value	Unit	Description
AXIS_REF: Standard					
wDriveID	WORD	1	1		ID of drive
bVirtual	BOOL	FALSE	FALSE		drive is simulated
dwRatioTechUnitsDenom	DWORD	1	1		conversion inc./tech.units denominator
iRatioTechUnitsNum	INT	1	1		conversion inc./tech.units numerator
iMovementType	INT	1	1		movement type: 0: rotary/modulo, 1: linear
fPositionPeriod	LREAL	360.0	360.0		modulo value for rotary drives
eRampType	INT	0	0		selects the velocity ramp type used by the FBs
fSWMaxVelocity	LREAL	1e3	1e3		maximum velocity value used for limit check at SMC_ControlAxisByPos
fSWMaxAcceleration	LREAL	1e5	1e5		maximum acceleration value used for limit check at SMC_ControlAxisByPos
fSWMaxDeceleration	LREAL	1e5	1e5		maximum deceleration value used for limit check at SMC_ControlAxisByPos
fRampJerk	LREAL	0	0		jerk used for bringing acceleration to 0 when sin² ramp is interrupted
fSWLimitPositive	LREAL	1000.0	60.0		software limit position in positive direction
fSWLimitNegative	LREAL	0.0	-60.0		software limit position in negative direction
fSWLimitDeceleration	LREAL	0	0		deceleration value used to stop when a software error has been detected
bSWLimitEnable	BOOL	FALSE	TRUE		activate/deactivate software limit
fSWErrorMaxDistance	LREAL	0	0		maximum distance that may be travelled for ramping down after a software error has been detect...
Servo: Configuration					
negative position [units]	LREAL	-60	-60.0		
positive position [units]	LREAL	60	60.0		
startup position [units]	LREAL	0.0	0.0		

Connect with a web browser to <network address>:8080/webvisu.htm and see the generated set positions and influence the velocity:



9. EtherCAT.project (precondition: The following devices are connected to the Pi's LAN adapter: Beckhoff EK1100 mit Beckhoff EL2008)

This example switches the eight available outputs on the connected hardware implementing an EtherCAT master.

Open the project, download and start it. The outputs of the clamp will change continuously.

Please note that the LAN port of your Raspberry Pi is then blocked for EtherCAT and hence cannot be used as communication and programming interface. It is recommended to use a USB WLAN adapter (e.g. Edimax N150) in this case.

10. Connect additional devices via I²C and SPI

To connect additional devices via I²C or SPI you can use the examples, that are included in this product and installed to the installation path, as base. To support a new device, you should generate a new device description and a new library. Execute the following steps:

A. Device descirption

- Generate a copy of an existing device description that is installed tot he installation path and rename it to <myDeviceName>.devdesc.xml.
- Modify the ID of this device. Set the high word to FFFF, the low word set locally unique:[1]

```
<Device hideInCatalogue="false">
  <DeviceIdentification>
    <Type>501</Type>
    <Id>FFFF 4711</Id>
    <Version>1.0.0.0</Version>
  </DeviceIdentification>
```

- Adapt the device information:

```
  <DeviceInfo>
    <Name name="local:ModelName">MCP3008</Name>
    <Description name="local:DeviceDescription">MCP3008</Description>
    <Vendor name="local:VendorName">3S - Smart Software Solutions GmbH</Vendor>
    <OrderNumber>-</OrderNumber>
```

- Enter the name, vendor and version of the library you are going to generate later:

```
<RequiredLib libname="Raspberry SPI MCP3008" vendor="3S - Smart Software Solutions GmbH" version="1.0.0.0" identifier="deviceLib">
```

- Set the name of the FB in the library, which will do the communication and represent the device:

```
<FBInstance basename="$(DeviceName)" fbname="MCP3008">
  <Initialize methodName="Initialize" />
  <CyclicCall methodname="AfterReadInputs" task="#buscycletask" whentocall="afterReadInputs" />
  <CyclicCall methodname="BeforeWriteOutputs" task="#buscycletask" whentocall="beforeWriteOutputs" />
</FBInstance>
```

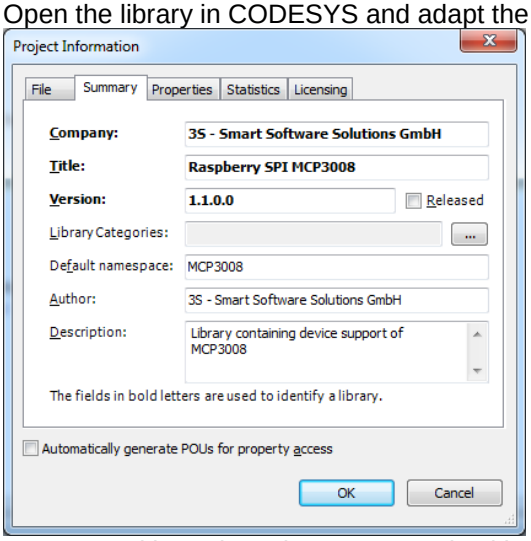
- Install the device description in your device repository in CODESYS. From now on you will be able to add your new device below „I²C devices“ or „SPI devices“.

B. library

- generate a copy of the example library. Rename it to <myDeviceName>.library.

[1] Please note, that if you want to distribute your new device support you need a registered customer ID at 3S - Smart Software Solutions GmbH

- Open the library in CODESYS and adapt the project information:



  Company, title and version must match with the device description.
- Rename the FB in the library. The new name must match with the one set in the device description.
- You can install a state machine in the body of the FB. iState=0 represents the init state, _iState=10 normal operation, _iState=1000 a mistake. Intermediate steps may be added if needed.
- In the method AfterReadInputs you should read the inputs from your device . For the communication use the standard methods of the base FBs i2c (read8, write8, etc.) or spi (transfer)
- In the method BeforeWriteOutputs you should write the outputs to your device
- Save the library and install it in your library repository.

C. Utilization

You can now generate a project and add the new device under the correct communication interface. This will automatically generate a FB instance that you can use in your application.

7 Screenshots