

XSG Electric Library

Version 20.1 – 06 2020

dSPACE

How to Contact dSPACE

Mail:	dSPACE GmbH Rathenaustraße 26 33102 Paderborn Germany
Tel.:	++49 5251 1638-0
Fax:	++49 5251 16198-0
E-mail:	info@dspace.de
Web:	http://www.dspace.com

How to Contact dSPACE Support

There are different ways to contact dSPACE Support:

- Visit our Web site at <http://www.dspace.com/goto?support>
- Send an e-mail or phone:
 - General Technical Support:
support@dspace.de
+49 5251 1638-941
 - SystemDesk Support:
support.systemdesk@dspace.de
+49 5251 1638-996
 - TargetLink Support:
support.tl@dspace.de
+49 5251 1638-700
- Use the dSPACE Installation Manager:
 - On your dSPACE DVD at *\Tools\InstallationManager.exe*
 - Via Start – Programs – dSPACE Installation Manager (after installation of the dSPACE software)
 - At <http://www.dspace.com/goto?im>

You can always find the latest version of the dSPACE Installation Manager here.
dSPACE recommends that you use the dSPACE Installation Manager to contact dSPACE Support.

Software Updates and Patches

dSPACE strongly recommends that you download and install the most recent patches for your current dSPACE installation. Visit <http://www.dspace.de/goto?support> for software updates and patches.

Important Notice

This document contains proprietary information that is protected by copyright. All rights are reserved. Neither the documentation nor software may be copied, photocopied, reproduced, translated, or reduced to any electronic medium or machine-readable form, in whole or in part, without the prior written consent of dSPACE GmbH.

© Copyright 2011 - 2014 by:
dSPACE GmbH
Rathenaustraße 26
33102 Paderborn
Germany

This publication and the contents hereof are subject to change without notice.
AutomationDesk, CalDesk, ConfigurationDesk, ControlDesk, SCALEXIO, SystemDesk and TargetLink are registered trademarks of dSPACE GmbH in the United States or other countries, or both. Other brand names or product names are trademarks or registered trademarks of their respective companies or organizations

Contents

About This Guide.....	6
<i>Document Symbols and Conventions</i>	6
<i>Accessing PDF File.....</i>	7
<i>Related Documents.....</i>	8
<i>Requirements.....</i>	9
Target Hardware	11
DS5203 FPGA Board.....	11
DS1514 FPGA Board + DS1552 I/O Module.....	12
DS1202 FPGA Board (MicroLabBox)	14
DS2655/DS6601/DS6602 FPGA Board + DS2655Mx I/O Module.....	15
Simulink Model Structure.....	17
General Description.....	17
XSG Electric Library.....	19
Library Contents.....	19
Description / Overview.....	21
Library Structure	21
Sensors: Incremental Encoder (TTL)	26
Processor Output.....	27
FPGA Main Component.....	29
Processor Input.....	31
Interface Examples.....	31
Sensors: Sine Encoder.....	32
Processor Output.....	33
FPGA Main Component.....	37
Processor Input.....	40
Interface Examples.....	40
Sensors: Hall Encoder.....	41
Processor Output.....	42
FPGA Main Component.....	47
Processor Input.....	49
Interface Examples.....	49
Sensors: Resolver.....	50
Processor Output.....	51
FPGA Main Component.....	55
Processor Input.....	57
Interface Examples.....	57
Sensors: SSI Encoder	59
Processor Output.....	59
FPGA Main Component.....	66
Processor Input.....	69
Interface Examples.....	69

Sensors: EnDat Encoder.....	71
Processor Output (Base Task).....	72
Processor Output (Memory Access Task)	76
FPGA Main Component.....	78
Processor Input (Base Task).....	80
Processor Input (Memory Access Task).....	80
Interface Examples.....	80
Sensors: HIPERFACE® Encoder	81
Processor Output.....	82
FPGA Main Component.....	88
Processor Input.....	91
Interface Examples.....	92
Sensors: BiSS Encoder.....	94
Processor Output (Base Task).....	95
Processor Output (Memory Access Task)	101
FPGA Main Component.....	103
Processor Input (Base Task).....	104
Processor Input (Memory Access Task).....	105
Interface Examples.....	105
Electric Machines: Brushless DC Motor (BLDC)	107
Processor Output.....	108
FPGA Main Component.....	116
Processor Input.....	120
Interface Examples.....	125
Electric Machines: Permanent Magnetized Synchronous Machine (PMSM)	127
Processor Output.....	128
FPGA Main Component.....	132
Processor Input.....	136
Interface Examples	137
Electric Machines: Table based Permanent Magnetized Synchronous Machine (PMSM Table based).....	140
Processor Output.....	142
FPGA Main Component.....	150
Processor Input.....	154
Interface Examples.....	158
Electric Machines: Squirrel Cage Induction Machine (SCIM)	160
Processor Output.....	162
FPGA Main Component.....	168
Processor Input.....	171
Interface Examples	175
Electric Machines: Squirrel Cage Induction Machine DQ (SCIM_MACHINE_DQ) ...	177
Processor Output.....	179
FPGA Main Component.....	188
Processor Input.....	191
Interface Examples.....	197
Electric Machines: Separately Exited DC machine	199
Processor Output.....	200
FPGA Main Component.....	206
Processor Input.....	208
Interface Examples.....	211
Power Electronics: DCM Inverter.....	212
Processor Output.....	212

FPGA Main Component.....	217
Processor Input.....	221
Interface Examples.....	225
Mechanic.....	227
Processor Output.....	228
FPGA Main Component.....	230
Processor Input.....	234
Interface Examples.....	235
Processor based model parts.....	237
Overview	237
Motor Inertia.....	238
Motor Position.....	240
Park Modulator	241
Clarke / Park transformation.....	242
Star / Delta conversion.....	245
Discrete PT1	249
Discrete Integrator.....	251
PI controller	253
PI Controller with external saturation	255
Space vector modulation.....	256
Three Level Space vector modulation.....	257
PMSM Controller Basic	259
PMSM Controller.....	262
Three Level PMSM Controller	266
Demo	268
Overview	268
Demo Installation	269
PMSM Closed Loop Demo	271
Functions	285
Overview	285
dSPACE Controller Adaption	286
Introduction.....	286
Plant Types	286
Tuning Methods.....	288
Controller Tuning GUI.....	295
Example: Cascade Controller for a Permanent Magnet Synchronous Machine	297

About This Guide

Document Symbols and Conventions

Symbols

The following symbols may be used in this document:

	Indicates a general hazard that may cause personal injury of any kind if you do not avoid it by following the instructions given.
	Indicates the danger of electric shock which may cause death or serious injury if you do not avoid it by following the instructions given.
	Indicates a hazard that may cause material damage if you do not avoid it by following the instructions given.
	Indicates important information that should be kept in mind, for example, to avoid malfunctions.
	Indicates tips containing useful information to make your work easier.

Naming Conventions

The following abbreviations and formats are used in this document:

%name% Names enclosed in percent signs refer to environment variables for file and path names, for example, %DSPACE_PYTHON25% specifies the location of your dSPACE installation in the file system.

< > Angle brackets contain wildcard characters or placeholders for variable file and path names, etc.

Precedes the document title in a link that refers to another document.

Indicates that a link refers to another document, which is available in dSPACE HelpDesk.

Accessing PDF File

Objective After you install your dSPACE software, the documentation for the installed products is available as Adobe® PDF file.

PDF files You can access the PDF files as follows:

Documentation root: <%INSTALLDIR%>\Doc\XSG_ELECTRIC.pdf

Related Documents

Below is a list of documents that you are recommended to read when working with the XSG Electric Library.

- RTI Reference
- RTI FPGA Programming Blockset Guide
- RTI FPGA Programming Blockset-Processor Interface Reference
- RTI FPGA Programming Blockset-FPGA Interface Reference
- Hardware Installation and Configuration Reference
- DS5203 RTLib Reference
- Real-Time Interface (RTI and RTI-MP) - Implementation Guide
- Modular Systems Based on DS100x - Installation and Configuration Guide

Requirements

The following tools have to be installed for using the DS5203 FPGA Board:

- MATLAB & Simulink (requires Fixed-Point Designer for bus-widths greater than 53 bits)
- Xilinx Vivado including XSG
- dSPACE Release

Below you can find the compatibility matrix for dSPACE Release, Xilinx and MATLAB for the XSG Electric Components and the XSG Electric Components Interface library:

RTI FPGA Programming Blockset	dSPACE Release	Operating System	MATLAB	Xilinx Design Tools
3.9	2020-A	Windows 7, Windows 10	R2018b, R2019a, R2019b	Vivado 2019.2

Figure 1: Compatibility matrix for XSG Electric Component 20.1

dSPACE Release	Operating System	MATLAB
2020-A	Windows 7, Windows 10	R2018b, R2019a, R2019b, R2020a

Figure 2: Compatibility matrix for XSG Electric Component Interface 20.1

	The XSG Utils Library is available with Xilinx Design Tools
	It is extremely recommended to secure that only one XSG Electric Components library version is used with one Matlab!
	Please note that the Library is optimized for timing, resource consumption and hardware interface for the following FPGAs (boards):

	- Kintex 7 (e.g. DS2655 / DS5203 Boards)
--	------------------------------------------

Target Hardware

DS5203 FPGA Board

Board description

With the DS5203 FPGA board (shown in the figure below), an integration of a FPGA application into a dSPACE modular PHS-Bus based system can be performed.

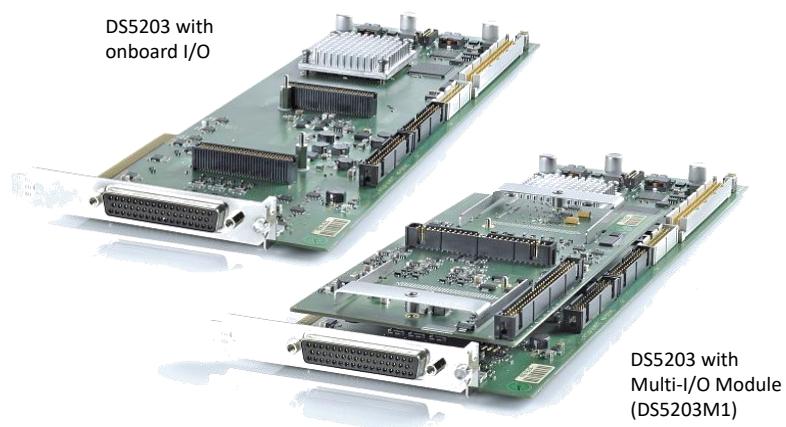


Figure 3: DS5203 Boards (PHS-based)

The board provides a Xilinx® Kintex-7 FPGA which can be programmed by the user by using the RTI FPGA Programming Blockset. The board features the following onboard I/O:

- 6 A/D channels with 10 MHz sample rate and adjustable voltage range
- 6 D/A channels with 10 MHz update rate and ± 10 V voltage range
- 16 digital single-ended input or output channels with configurable input voltage threshold and output voltage of 3.3 V or 5 V

An additional piggy back module (M1) to double the I/O can also be added, as well as a Resolver SC module (EV1099) to add an audio transformer to the DAC channels.

There are two FPGA types available:

- Kintex7 K410 (Xilinx Tool: Vivado + very huge amount of resources)
- Kintex7 K325 (Xilinx Tool: Vivado + huge amount of resources)

DS1514 FPGA Board + DS1552 I/O Module

Board description

Using the DS1514 FPGA base board for MicroAutoBox (DS1401) in combination with the DS1552 Multi-I/O piggy-back module / DS1554 Engine Control I/O Module, an integration of a FPGA application into a dSPACE MicroAutoBox can be performed.

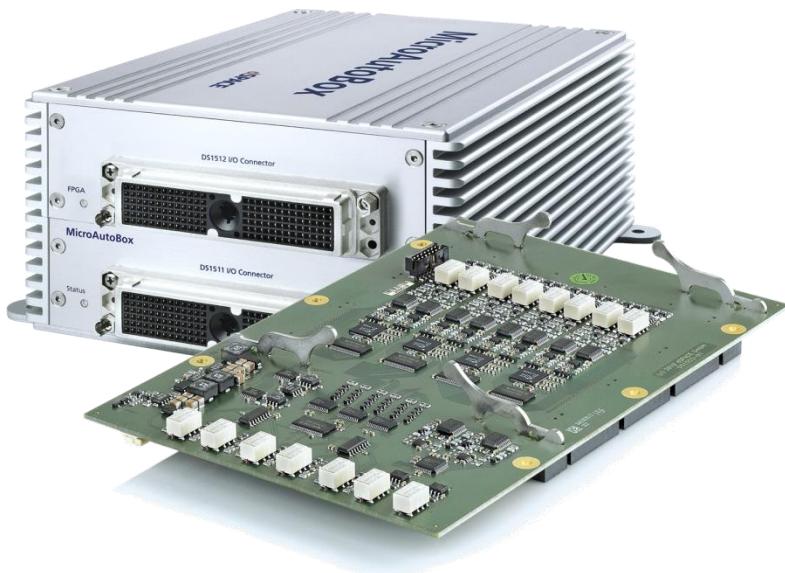


Figure 4: MicroAutoBox (DS1401) with DS1514 and DS1552

The board provides a Xilinx® Kintex-7 FPGA which can be programmed by the user, for example, by using the RTI FPGA Programming Blockset.

DS1552 I/O module:

- 8 A/D channels with a sample rate of 1 MHz and 0...5 V or ± 10 V voltage range
- 16 A/D channels with a sample rate of 200 kHz and ± 10 V voltage range
- 4 D/A channels with an update rate of 2.1 MHz and 0...5 V voltage range
- 16 digital single-ended 5 V input channels
- 16 digital single-ended output channels with configurable output voltage
- 8 digital single-ended input or output channels with configurable input voltage threshold and output voltage of 3.3 V or 5 V
- 3 crank/cam input channels with ± 40 V voltage range
- 1 Inductive zero voltage detector (for zero-crossing detection)
- 4 UART interfaces

DS1554 I/O module:

- 14 A/D channels with a sample rate of 1 MHz and ± 10 V voltage range

- 4 channels can be equipped with shunt resistors for current measurement
 - 40 digital out channels for general purpose actuator control
 - 8 bidirectional in- / output channels
- Input:
- Threshold adjustable for each channel from 1V to 7.5V
 - 4 channels can be equipped with shunt resistors for current sourcing sensors
- Output:
- Push-Pull drivers
 - One output voltage can be selected for all channels: 3.3V or 5V
- 2 Lambda channels, supported Lambdaprobes:
 - BOSCH LSU 4.9
 - BOSCH LSU ADV gasoline & diesel
 - BOSCH LSU 5.1
 - Knock channels:
 - 4 A/D channels with a sample rate of 1 MHz and ± 5 V voltage range
 - Crank / Cam channels
 - 5 digital in channels with configurable thresholds dedicated for crank/cam measurement
 - 1 Crank / Cam channel with zero crossing detection for crank / cam measurement

DS1202 FPGA Board (MicroLabBox)

Board description

Using the DS1202 FPGA board for MicroLabBox (shown in the figure below) with on-board I/O, an integration of a FPGA application into a dSPACE MicroLabBox can be performed. Three different variants are available as shown in the figure below.

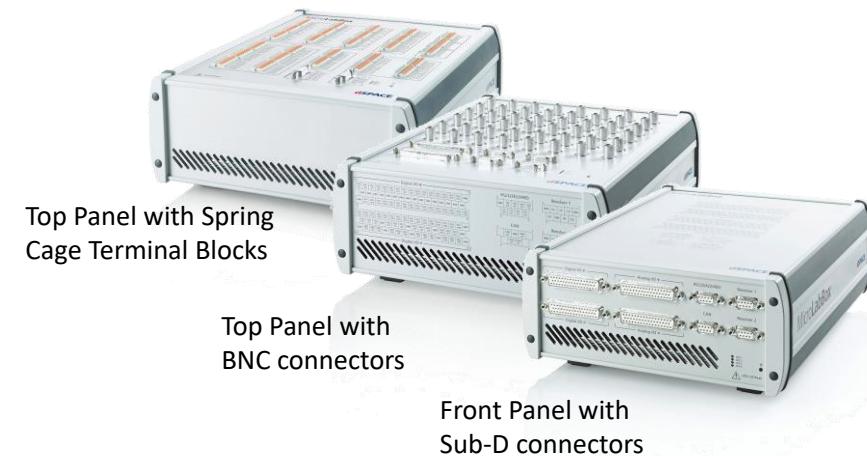


Figure 5: MicroLabBox with DS1202

The DS1202 FPGA board provides a Xilinx® Kintex-7 FPGA which can be programmed by the user by using the RTI FPGA Programming Blockset. It features the following I/O:

- 8 A/D channels with 10 MHz sample rate and ± 10 V voltage range
- 24 A/D channels 1 MHz sample rate and ± 10 V voltage range
- 16 D/A channels with 1 MHz update rate and ± 10 V voltage range
- 48 digital single-ended input or output channels with configurable input voltage threshold and output voltage of 2.5V, 3.3 V or 5 V
- 12 digital differential RS485/RS422 channels

DS2655/DS6601/DS6602 FPGA Board + DS2655Mx I/O Module

Board description

With the DS2655/DS6601/DS6602 FPGA base module and the DS2655M1 I/O module / DS2655M2 I/O module (shown in the figure below), an integration of a FPGA application into a dSPACE modular IOCNET based system (SCALEXIO) can be performed.

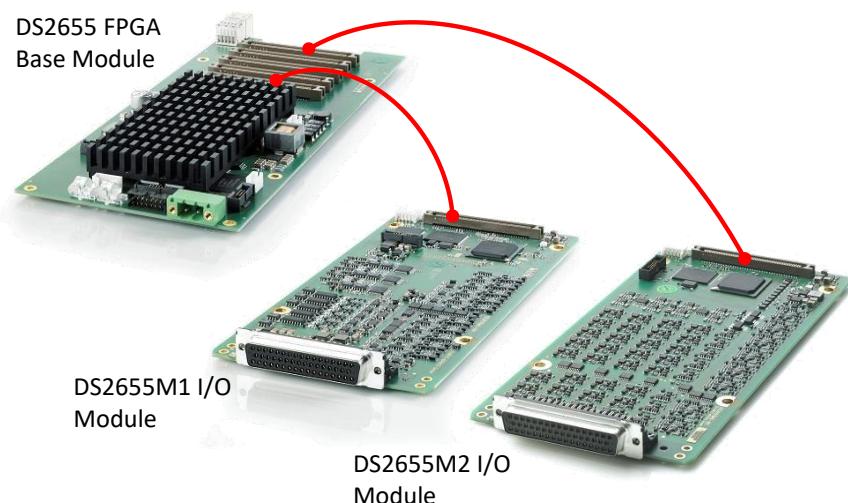


Figure 6: DS2655 FPGA base module and the DS2655M1 I/O module / DS2655M2 I/O module (SCALEXIO)

The DS2655 FPGA base module provides a Xilinx® Kintex-7 FPGA which can be programmed by the user by using the RTI FPGA Programming Blockset. Up to 5 I/O modules can be connected to the DS2655/DS6601/DS6602. There are two FPGA type available:

- DS6602 Kintex Ultrascale KU15P
(Xilinx Tool: Vivado + very huge amount of recourses)
- DS2655 Kintex7 K410; DS6601 Kintex Ultrascale KU035
(Xilinx Tool: Vivado + huge amount of recourses)
- DS2655 Kintex7 K160
(Xilinx Tool: Vivado + standard amount of recourses)

DS2655M1:

- 5 A/D channels with 2 MHz sample rate and adjustable voltage range
- 5 D/A channels with 7.8125 MHz update rate and ± 10 V voltage range
- 10 digital single-ended input or output channels with configurable input voltage threshold and output voltage of 3.3 V or 5 V

The I/O of the DS2655M1 enables most of the features provided by the XSG AC Motor Control library. Processing of SSI sensors, as it required RS485 or RS422

digital I/O drivers, please use the DS2655M2 I/O module. The DS2655M2 I/O module features 32 digital I/O channels with the following features:

DS2655M2:

- Push-Pull drivers
- One output voltage can be selected for all channels: 3.3V or 5V
- 16 of the channels are extended with RS485/RS232 transceivers
- RS232: max. 250 kBaud
- RS485: max. 40 MBaud
- Max. of following functions are possible:
 - 32 x digital input
 - 32 x digital output (push/pull or push or pull)
 - 16 x digital output (push/pull/tristate)
 - 8 x RS232 RX channels are free) (24 digital I/O or 8 x RS232 TX
 - 8 x RS232 TX channels are free) (24 digital I/O or 8 x RS232 RX
 - 8 x RS485 RX (16 digital I/O channels are free)
 - 8 x RS485 TX (16 digital I/O channels are free)
 - 8 x RS485 RX/TX (8 digital I/O channels are free)

Simulink Model Structure

General Description

Description

This chapter describes the user interface for accessing the DS5203 FPGA Board from Simulink. In general, there are two different ways to access the DS5203 board, either via PHS bus for communication between processor and FPGA or via digital or analog in channels on the board. The FPGA output is similar to its input: either information can be written from the FPGA to the processor via PHS-bus register, or information can be sent out via hardware channels.

Two interfaces are available for handling the communication.

- **Processor interface** (read and write on the processor side)
- **FPGA interface** (read and write on the FPGA side)

The processor interface comes with the regular RTI license, the FPGA interface comes with an extra RTI FPGA license. Here you can see the RTIFPGA library with both parts, the processor-based and the FPGA-based elements.

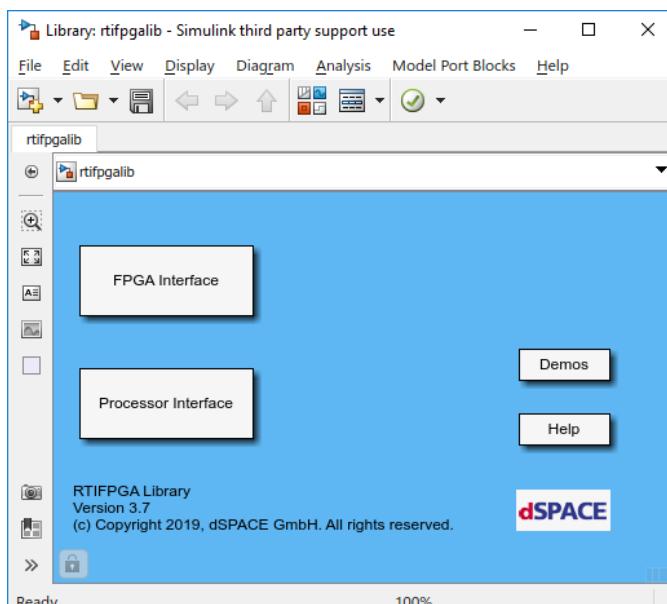


Figure 7: RTIFPGA Library

	<p>Separating these two libraries makes it possible to use precompiled FPGA applications without buying an extra license for FPGA modeling.</p>
-------------------------------------------------------------------------------------	-------------------------------------------------------------------------------------------------------------------------------------------------

The XSG Electric Library contents are designed to give you easy access to create the model that you require. Each main component blockset consists of five elements:

1. **Processor out interface** (<component>_out (Processor Interface))
All the required parameters and actual values are merged into the smallest necessary number of PHS-bus registers or buffers.
2. **FPGA in interface** (<component>_in (FPGA Interface))
The merged signals from the processor side are decoded on the FPGA side
3. **FPGA main component** (<component>_FPGA (FPGA))
Provides the actual model functionality of the component
4. **FPGA out interface** (<component>_out (Processor Interface))
All the required signals to be returned to the processor are merged here to utilize the smallest possible amount of PHS bus registers
5. **Processor in interface** (<component>_in (Processor Interface))
The merged signals from the FPGA side are decoded on the processor side

For further information about the interfacing structure, automatic interface generation please refer to the documentation of the XSG Utils library.

XSG Electric Library

Library Contents

Description / Overview	The XSG Electric Library is designed to give you easy access to perform e-motor HIL real time simulation based on a FPGA platform, either on signal or on power level.
Sensors	<p>Position sensor simulation for the following encoders:</p> <ul style="list-style-type: none"> ▪ Incremental encoder (TTL) ▪ Sine encoder ▪ Hall encoder ▪ Resolver ▪ SSI Encoder ▪ EnDat Encoder ▪ HIPERFACE Encoder
Electric Machines	<p>The following motor models are available:</p> <ul style="list-style-type: none"> ▪ Brushless DC Motor (BLDC) ▪ Permanent Magnetized Synchronous Machine (PMSM) ▪ Separately Exited DC Machine ▪ Squirrel Cage Induction Machine (SCIM)
Power Electronics	<p>Three-phase inverter models:</p> <ul style="list-style-type: none"> ▪ BLDC Inverter ▪ PMSM Inverter ▪ DCM Inverter
Utils	<p>The library also contains a separate library called XSG_Utils. This library collected often used elements (e.g. angular processing unit, scaling- and scope functionalities, wave-table encoder) which are not especially necessary to simulate an electric motor. To get a quick and easy usability of these tools, all FPGA main components are supported with specially designed and optimized interface to the processor model. For further information about these blocks, please find the documentation of the XSG Utils library.</p>
Processor based model parts	<p>The XSG Electric Components library also contains several processor based model parts for full support of e-motor simulation. Therefore the following processor model parts are available:</p> <ul style="list-style-type: none"> ▪ Motor inertia ▪ Motor position integration ▪ Clarke / Park transformation ▪ Park Modulator

- Discrete PT1
- Discrete Integrator
- PI Controller
- PMSM Controller
- Space Vector Modulation
- Star / Delta conversion

Demos

The XSG Electric Components library also contains a separate demo model to illustrate an exemplary model structure of an FPGA application.

Description / Overview

Objective	The following section describes the general structure and features of the XSG Electric Library.
------------------	-------------------------------------------------------------------------------------------------

Library Structure

General information	The XSG Electric Library is divided into:
----------------------------	-------------------------------------------

- XSG Electric Components Library (FPGA-based blocks)
- XSG Electric Components Interface Library (processor-based blocks)

The XSG Electric Library contains always the separate XSG Utils Library. The following figure illustrates the overall structure of the solution package.

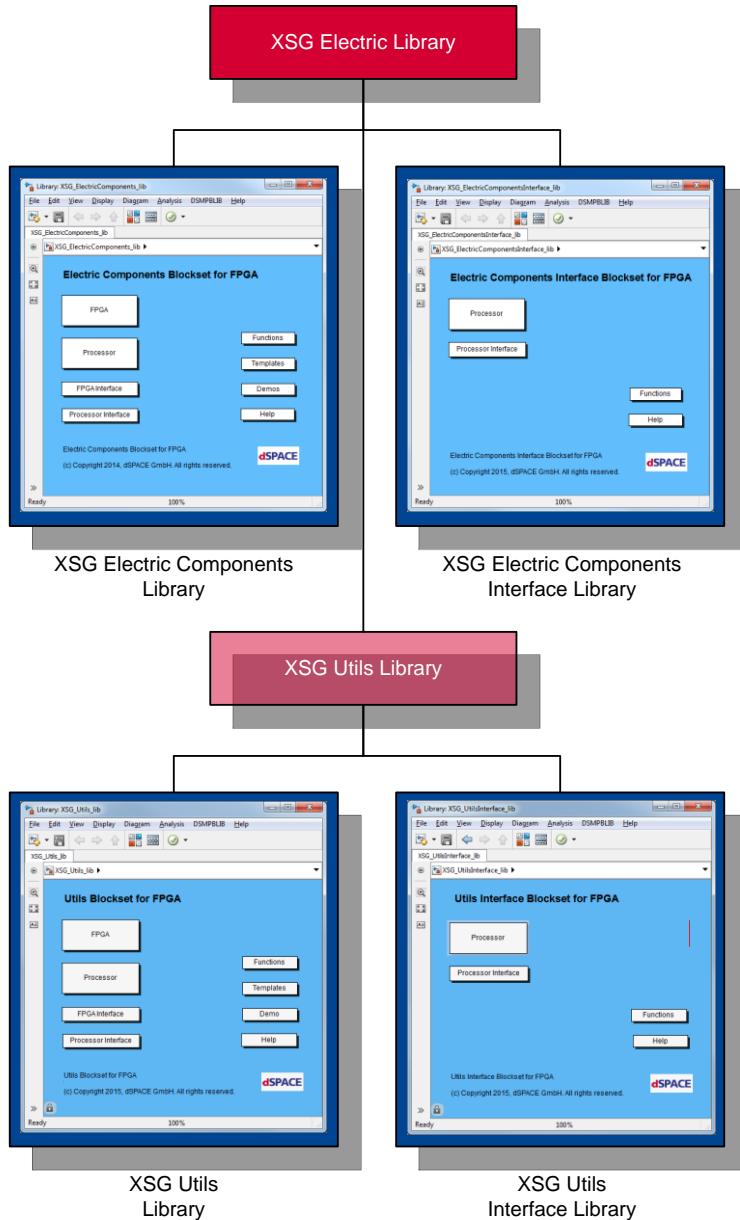


Figure 8: XSG Electric Library

To realize an easy user access to the different functionalities of the XSG Utils Library from the XSG Electric Library hyperlinks are used. For further information about the blocks of the XSG Utils Library please find the XSG Utils Library documentation.

XSG Electric Components Library

The XSG Electric Components Library provides easy access to FPGA based e-motor simulation and allows easy, user-specific e-motor HIL simulation, either on signal or on power level, including IO functionalities. The library is divided into FPGA / Processor Interface and FPGA components as shown in the figure below. To open the library, type "XSG_ElectricComponents_Lib" in the MATLAB Command Window.

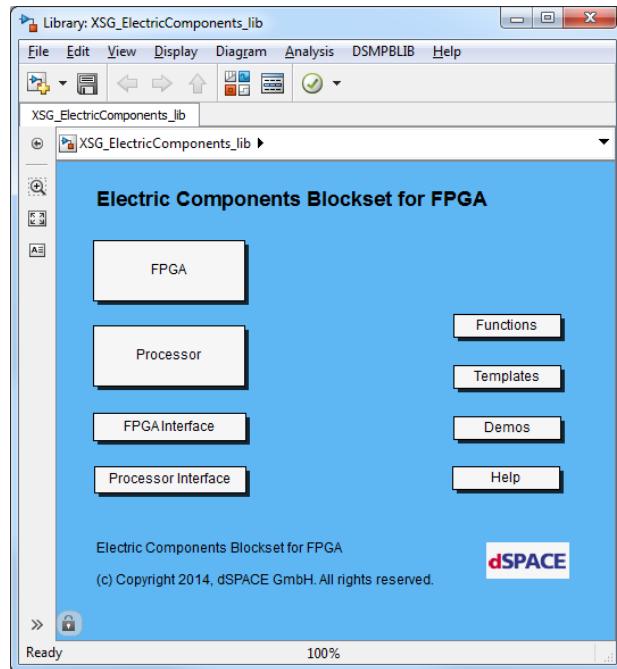


Figure 9: XSG Electric Components Library

The following illustration gives you an overview of the internal structure of the library.

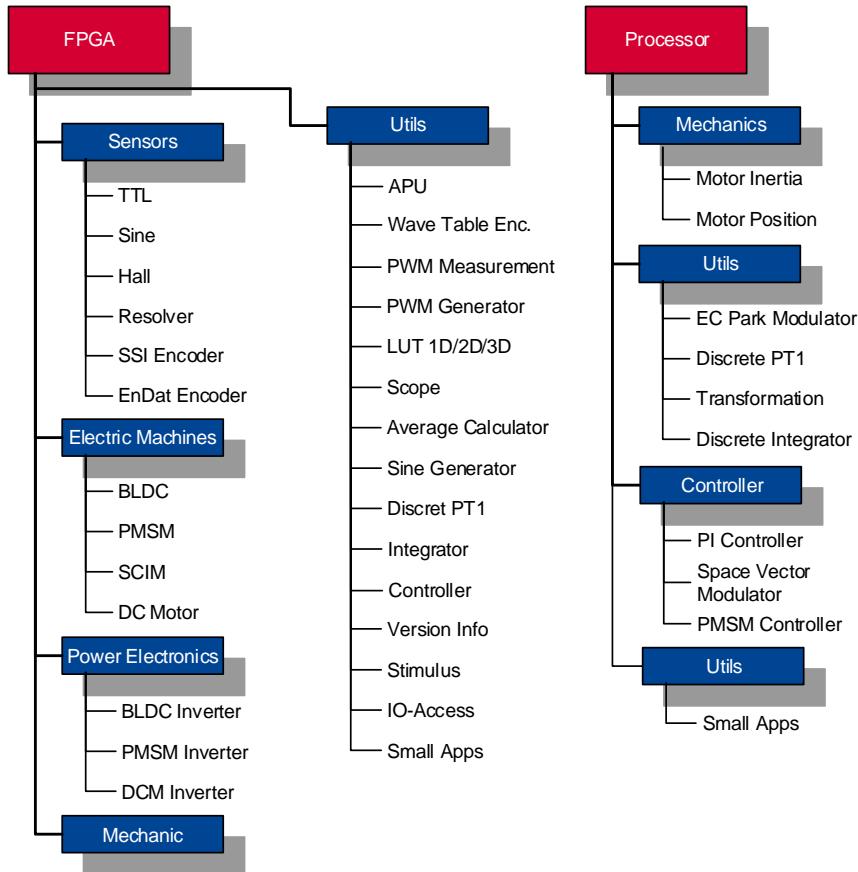


Figure 10: Overview of the XSG Electric Components Library

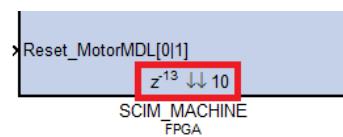
The FPGA-based blocks are located in the FPGA subsystem. To simplify the realization of an interface for sending processor-based parameters to the FPGA and back, optimized FPGA and processor interface blocks are located in the subsystems **FPGA Interface** and **Processor Interface**. All the systems are organized in logical groups like “**Sensors**”, “**Electric Machines**”, etc. Additional processor models like Park Clarke transformations, discrete integrator or motor inertia are in the **Processor** subsystem. All processor blocks are located in the **XSG Electric Components Interface Library** and are linked with the **XSG Electric Component Library**.



The XSG Electric Component Library also contains the XSG Electric Components Interface Library.

For an example of the general structure of an FPGA programming model, refer to the **Demos** subsystem.

To simplify the overall timing of the library blocks in a customer model, all FPGA based library blocks have a block description. This description gives information about the overall block latency and the down sampling of the internal model structure as shown in the figure below.



**Figure 11: Block information example (SCIM):
Latency: 13; internal model down sampling: 10**

XSG Electric Components Interface Library

If no user-defined adjustments have to be made in the FPGA code, the processor interface is separately located in the XSG Electric Components Interface Library as shown in the figure below. To open the Processor Interface Library, type "XSG_ElectricComponentsInterface_lib" in the MATLAB Command Window.

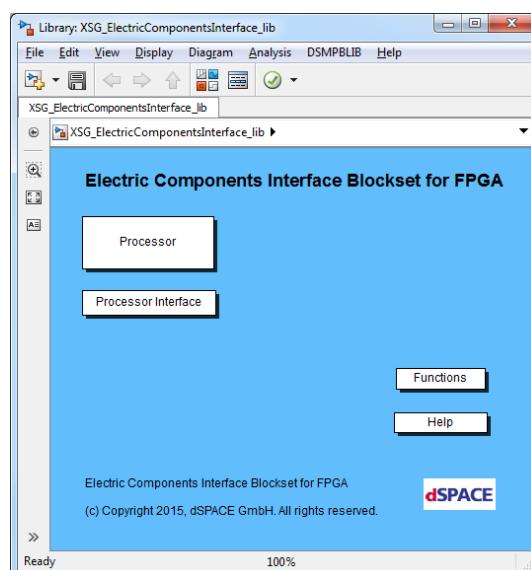


Figure 12: XSG Electric Components Interface Library

The internal structure of the library is the same as the structure of the XSG Electric Component Interface Library, except that the interface library only contains the processor blocks.

Sensors: Incremental Encoder (TTL)

Objective

The TTL encoder is a rotary relative sensor with a square wave output format. The phase output magnitude can be set to 3.3V or 5V, depending on the DAC settings. The encoder can either be used as single ended (3 signals) or as double ended (6 signals) sensor. The number of lines per revolution is online tunable (up to 16383). The encoder FPGA main component has to be connected to the APU interface signals.

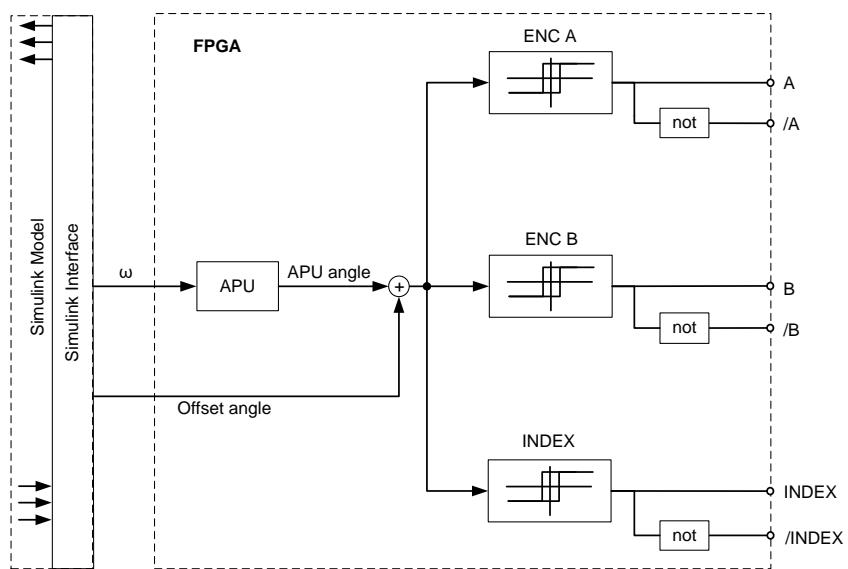


Figure 13: Equivalent circuit incremental encoder

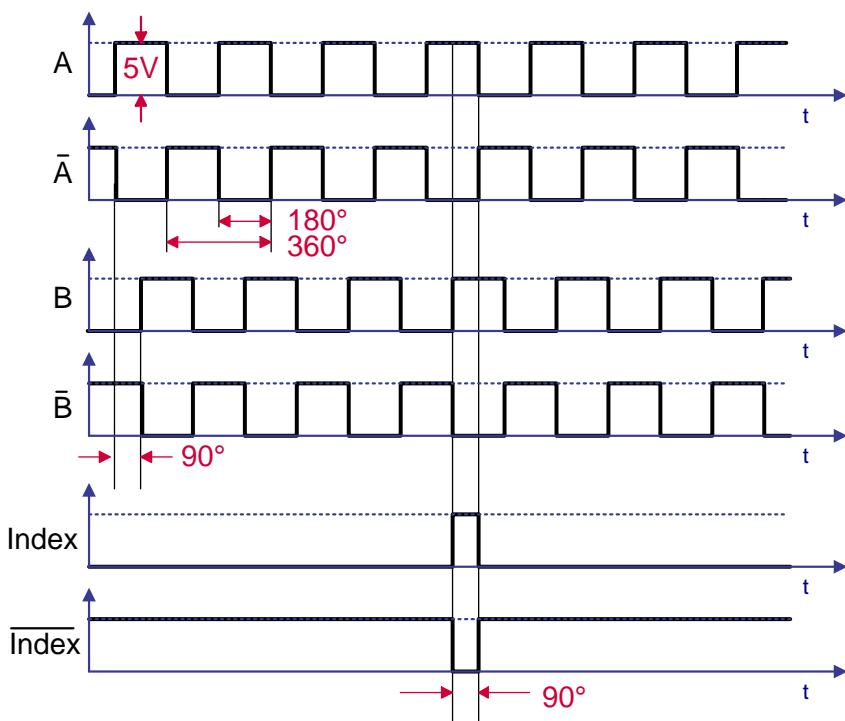


Figure 14: Incremental encoder shape

The blockset contains the following elements:

- Processor Interface: TTL_ENCODER_out (Processor Interface)
- FPGA Interface: TTL_ENCODER_in (FPGA Interface)
- FPGA: TTL_ENCODER (FPGA Main Component)

Processor Output

Block	Merges the processor signals and writes them to the FPGA
--------------	----------------------------------------------------------

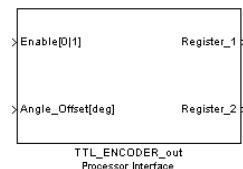


Figure 15: TTL_ENCODER_out block

Block Dialog	The processor output block set contains the following dialog.
---------------------	---------------------------------------------------------------

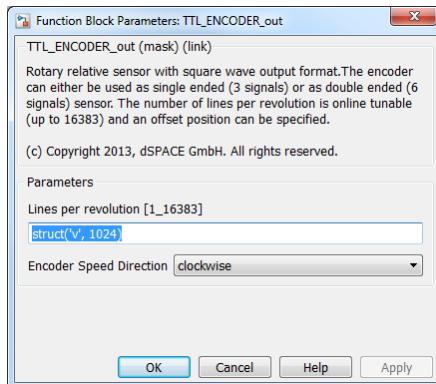


Figure 16: TTL_ENCODER_out dialog

The blockset dialog has the following parameters:

Name	Unit	Description	Range
Lines per revolution	[-]	Number of lines of the incremental encoder.	1 ... 16383; resolution: 1
Encoder Speed Direction	[-]	Configure the direction of the input speed (clockwise / counter-clockwise)	

Input

The TTL_ENCODER_out block has the following inputs:

Name	Unit	Description	Range
Enable	[-]	Sensor signal enable	0 1
Angle_Offset	[deg]	Offset angle for incremental encoder	0° ... 360°; resolution: 1.37E-3

Output

The Processor Out block provides two input registers, whose sectioning is shown below:

Register 1

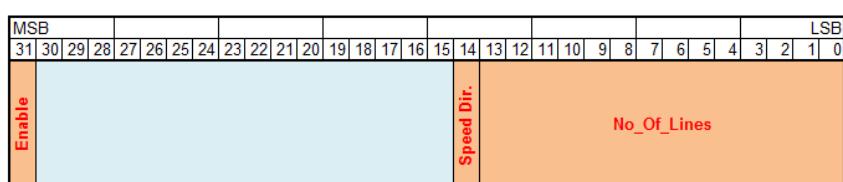


Figure 17: TTL_ENCODER_out Register 1

Name	Used Bits	Description	Range
No_Of_Lines	13 ... 0	Lines per revolution in range 1 ... 16383 and a resolution of 1	1 ... 16383; resolution: 1
Speed Direction	14	Configure the speed direction (clockwise / counter-clockwise)	0 1
SPARE	30 ... 15		
Enable	31	Sensor signal enable	0 1

Register 2

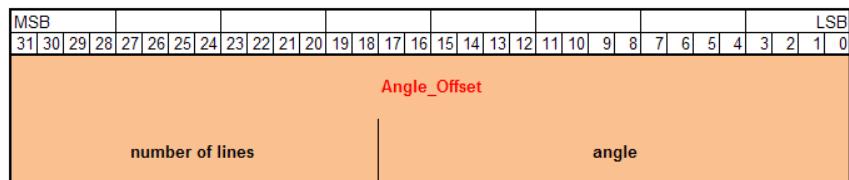


Figure 18: TTL_ENCODER_out Register 2

Name	Used Bits	Description	Range
Angle_Offset	31 ... 0	The offset angle represents the mechanical angle of the rotator, scaled to the number of lines of the encoder.	0...2^18-1 ≈ 360° electrical angle

FPGA Main Component

Block

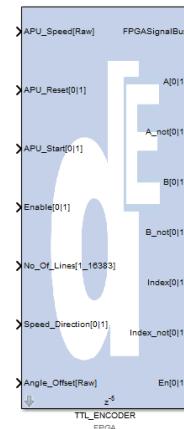


Figure 19: TTL_ENCODER FPGA Main block

Block Dialog

The FPGA main blockset contains the following dialog.

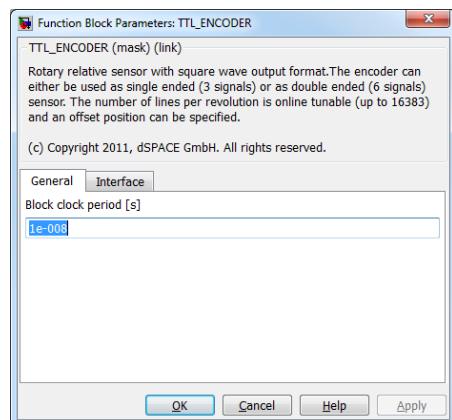


Figure 20: TTL_ENCODER FPGA Main block dialog

On the page Interface you can define the parameters of the input and output registers for automatic interface generation. The dialog also contains a button to start interface generation. For more details about automatic interface generation, refer to the XSG Utils documentation chapter Automatic Interface Generation. The FPGA clock period can be configured too.

Input

The main block has the following inputs:

Name	Unit	Description	Format
APU_Speed	[Raw]	Actual speed	Fix_30_0
APU_Reset	[-]	Reset on high state	Bool
APU_Start	[-]	Start on high state	UFix_1_0
Enable	[-]	Sensor signal enable	UFix_1_0
No_of_Lines	[-]	Number of lines of the incremental encoder.	UFix_14_0
Speed_Direction	[-]	Configure the speed direction (clockwise / counter-clockwise)	UFix_1_0
Angle_Offset	[Raw]	Offset angle for incremental encoder	UFix_32_0



If more than one encoder is mounted on a shaft, keep in mind that any modifications of the parameter Speed_Direction and No_of_Lines during runtime need a reset of the overall angular processing units (APUs).

Output

The TTL_ENCODER main block has the following outputs:

Name	Unit	Description	Format
FPGASignalBus	Bus	Bus containing the block's most significant signals	-
A	[0 1]	Track A	Bool
\bar{A}	[0 1]	Track \bar{A}	Bool
B	[0 1]	Track B	Bool
\bar{B}	[0 1]	Track \bar{B}	Bool
Index	[0 1]	Track Index	Bool
Index	[0 1]	Track <u>Index</u>	Bool
Enable	[0 1]	Enable signal output	Bool

Processor Input

Block

There is no processor input available for this blockset.

Interface Examples

Processor blocks

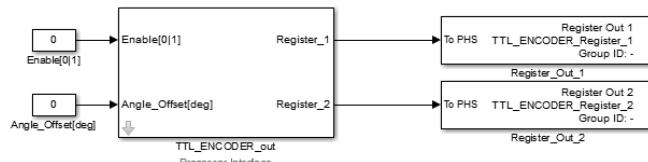


Figure 21: Processor interface

FPGA block

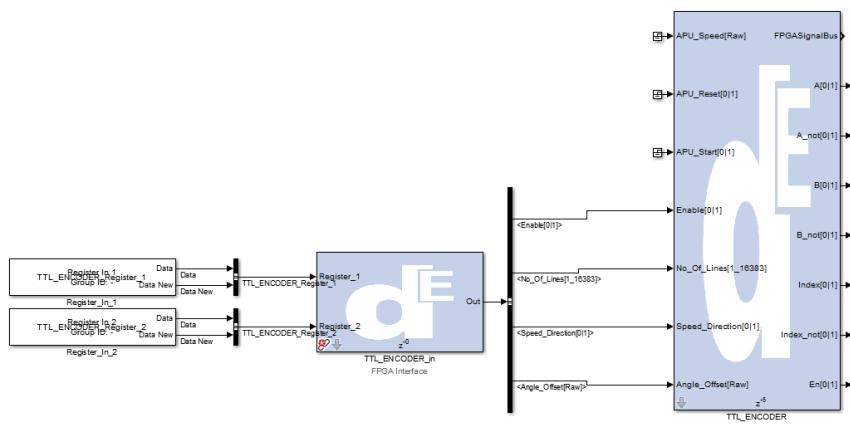


Figure 22: FPGA interface

Sensors: Sine Encoder

Objective

Sine encoders are a further development of pure TTL encoders. The optical system can be designed to modulate the light transition between light and dark sections. An additional resolution of up to n bit can then be obtained by using an analog track evaluation. You can set additional functionalities such as user-defined voltage amplitudes and offsets of the sine/cosine and the index track, and select the form (sine or square) and the length (90 deg electric or 1 deg mechanic) of the index track. The encoder FPGA main component has to be connected to the APU interface signals.

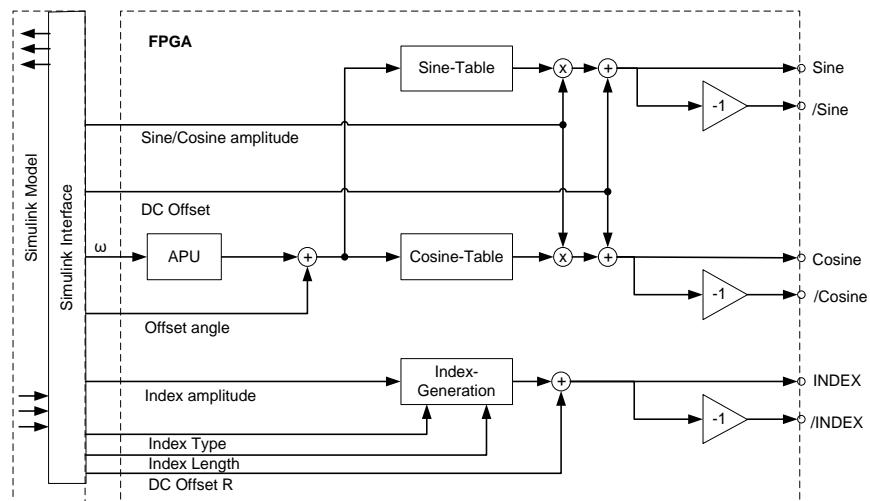


Figure 23: Equivalent circuit sine encoder

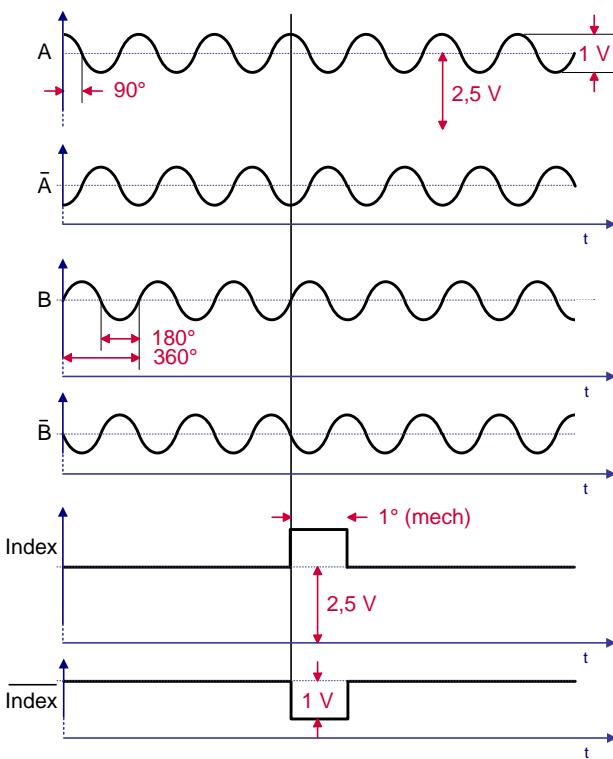


Figure 24: Sine encoder shape

The blockset contains the following elements:

- Processor Interface: SINE_ENCODER_out (Processor Interface)
- FPGA Interface: SINE_ENCODER_in (FPGA Interface)
- FPGA: SINE_ENCODER (FPGA Main Component)

Processor Output

Block

Merges the processor signals and writes them to the FPGA

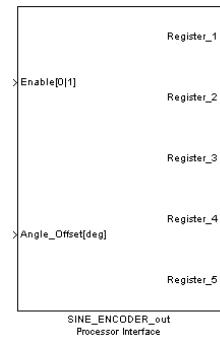


Figure 25: SINE_ENCODER_out block

Block Dialog

The processor output block set contains the following dialog.

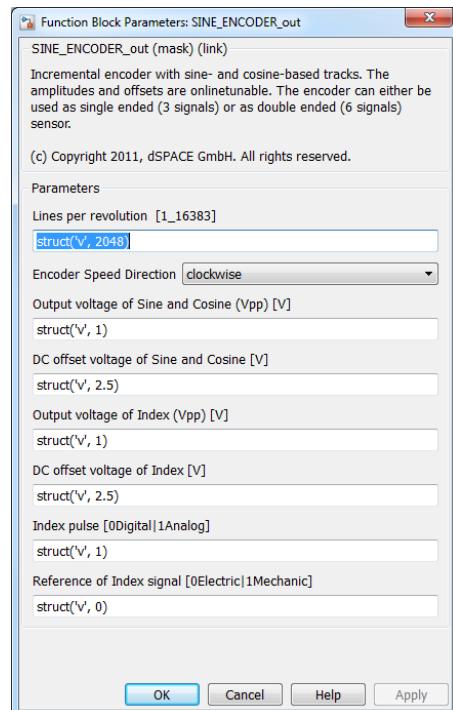


Figure 26: SINE_ENCODER_out dialog

The dialog blockset has the following parameters:

Name	Unit	Description	Range
Lines per revolution	[-]	Number of lines of the incremental encoder.	1 ... 16383; resolution: 1
Encoder Speed Direction	[-]	Configure the direction of the input speed (clockwise / counter-clockwise)	
Output voltage of Sine and Cosine	[V]	Peak-to-peak voltage of the sine and cosine track	0... 10V; resolution: 610E-6
DC offset voltage of Sine and Cosine	[V]	Offset voltage of the sine and cosine track	-10 ... 10V; resolution: 1.22E-3
Output voltage of Index	[V]	Pea- to-peak voltage of the index track	0... 10V; resolution: 610E-6
DC offset voltage of Index	[V]	Offset voltage of the index track	-10 ... 10V; resolution: 1.22E-3
Index pulse	[-]	Form of the index track; 0: digital behavior 1: analog behavior	0 1
Reference of Index signal	[-]	Length of the generated index signal 0: depends on the electrical angle 1: depends on the mechanical angle	0 1

Input

The SINE_ENCODER_out block has the following inputs:

Name	Unit	Description	Range
Enable	[-]	Sensor signal enable	0 1
Angle_Offset	[deg]	Offset angle for incremental encoder	0° ... 360°; resolution: 1.37E-3

Output

The Processor Out block provides five input registers whose sectioning is shown below:

Register 1

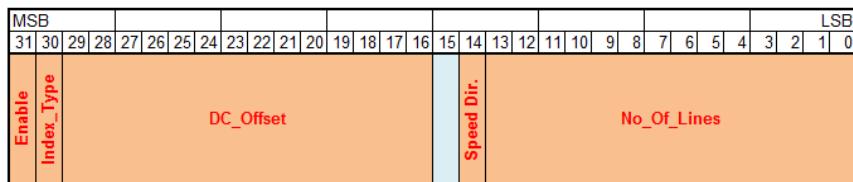


Figure 27: SINE_ENCODER_out Register 1

Name	Used Bits	Description	Range
No_Of_Lines	13 ... 0	Lines per revolution in range 1 ... 16383 and a resolution of 1	1 ... 16383; resolution: 1
Speed Direction	14	Configure the speed direction (clockwise / counter-clockwise)	0 1
SPARE	15		
DC_Offset	29 ... 16	Offset voltage of encoder output signals	0...2^12 ≈ positive voltage; 2^12+1...2^13 ≈ negative voltage (two's complement)
Index_Type	30	Form of the index track	0 1
Enable	31	Enable signal output	0 1

Register 2

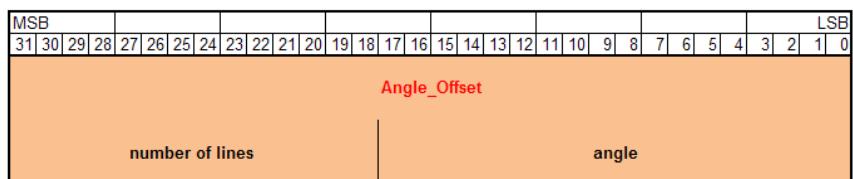


Figure 28: SINE_ENCODER_out Register 2

Name	Used Bits	Description	Range
Angle_Offset	31 ... 0	The offset angle represents the mechanical angle of the rotor, scaled to the number of lines of the encoder.	0...2^18-1 ≈ 360° electrical angle

Register 3

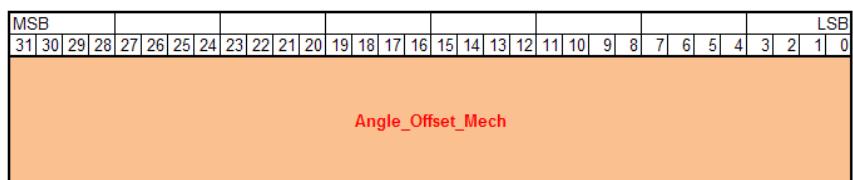


Figure 29: SINE_ENCODER_out Register 3

Name	Used Bits	Description	Range
Angle_Offset_Mech	31 ... 0	The offset angle represents the mechanical angle of the rotor.	0...2^32-1 ≈ 360° resolution: 83.8E-9

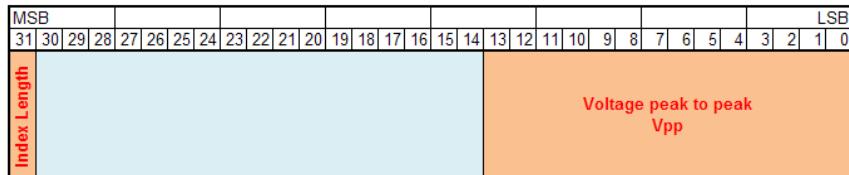
Register 4

Figure 30: SINE_ENCODER_out Register 4

Name	Used Bits	Description	Range
Voltage peak to peak Vpp	13 ... 0	Peak-to-peak voltage of the sine and cosine track	0... $2^{13} \hat{=} 0 \dots 10V$
SPARE	30 ... 14		
Index_Lengt h	31	Length of the generated index signal	0 1

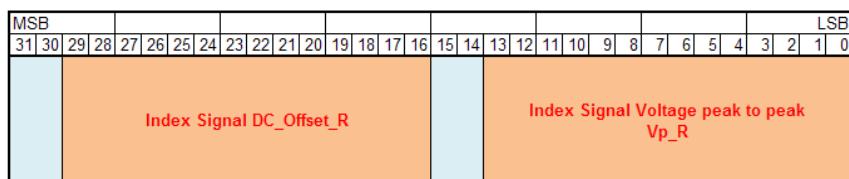
Register 5

Figure 31: SINE_ENCODER_out Register 5

Name	Used Bits	Description	Range
Index Signal Voltage Vp_R	13 ... 0	Peak-to-peak voltage of the index track	0... $2^{13} \hat{=} 0 \dots 10V$
SPARE	15 ... 14		
Index Signal DC_Offset	29 ... 16	Offset voltage of index track	0... $2^{12} \hat{=} \text{positive voltage}$; $2^{12+1} \dots 2^{13} \hat{=} \text{negative voltage}$ (two's complement)
SPARE	31 ... 29		

FPGA Main Component**Block**

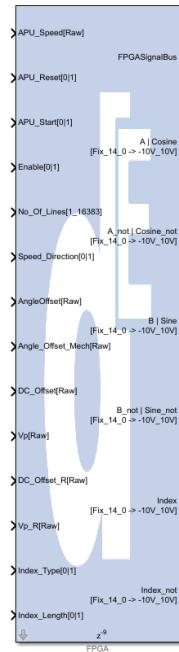


Figure 32: SINE_ENCODER FPGA Main block

Block Dialog

The FPGA main blockset contains the following dialog.

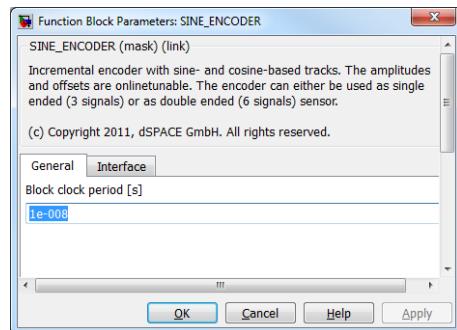


Figure 33: SINE_ENCODER FPGA Main block dialog

On the page Interface you can define the parameters of the input and output registers for automatic interface generation. The dialog also contains a button to start interface generation. For more details about automatic interface generation, refer to the XSG Utils documentation chapter Automatic Interface Generation. The FPGA clock period can be configured too.

Input

The main block has the following inputs:

Name	Unit	Description	Format
APU_Speed	[Raw]	Actual speed	Fix_30_0
APU_Reset	[-]	Reset on high state	Bool
APU_Start	[-]	Start on high state	UFix_1_0
Enable	[-]	Sensor signal enable	UFix_1_0
No_of_Lines	[-]	Number of lines of the incremental encoder.	UFix_14_0
Speed_Direction	[-]	Configure the speed direction (clockwise / counter-clockwise)	UFix_1_0
Angle_Offset	[Raw]	Offset angle for incremental encoder	UFix_32_0
Angle_Offset_Mech	[Raw]	Mechanical offset angle of the rotor shaft	UFix_32_0
DC_Offset	[Raw]	Offset voltage of the sine and cosine track	UFix_14_0
Vp	[Raw]	Pea- to-peak voltage of the sine and cosine track	UFix_14_0
DC_Offset_R	[Raw]	Offset voltage of the index track	UFix_14_0
Vp_R	[Raw]	Peak-to-peak voltage of the index track	UFix_14_0
Index_Type	[-]	Form of the index track	Bool
Index_Length	[-]	Length of the generated index signal	UFix_1_0



If more than one encoder is mounted on a shaft, keep in mind that any modifications of the parameter Speed_Direction and No_of_Lines during runtime need a reset of the overall angular processing units (APUs).

Output

The SINE_ENCODER main block has the following outputs:

Name	Unit	Description	Format
FPGASingalBus	Bus	Bus containing the block's most significant signals	-
A	[Bit]	Track cosine	Fix_14_0
\bar{A}	[Bit]	Track $\bar{\text{cosine}}$	Fix_14_0
B	[Bit]	Track sine	Fix_14_0
\bar{B}	[Bit]	Track $\bar{\text{sine}}$	Fix_14_0
Index	[Bit]	Track Index	Fix_14_0
$\bar{\text{Index}}$	[Bit]	Track $\bar{\text{Index}}$	Fix_14_0

Processor Input

Block

There is no processor input available for this blockset.

Interface Examples

Processor blocks

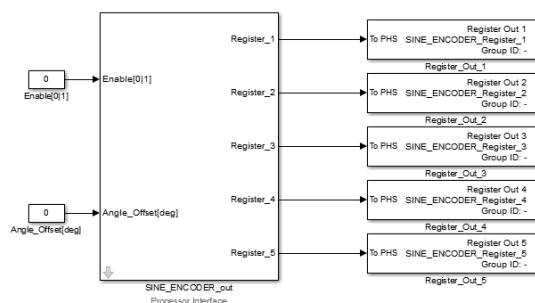


Figure 34: Processor interface

FPGA block

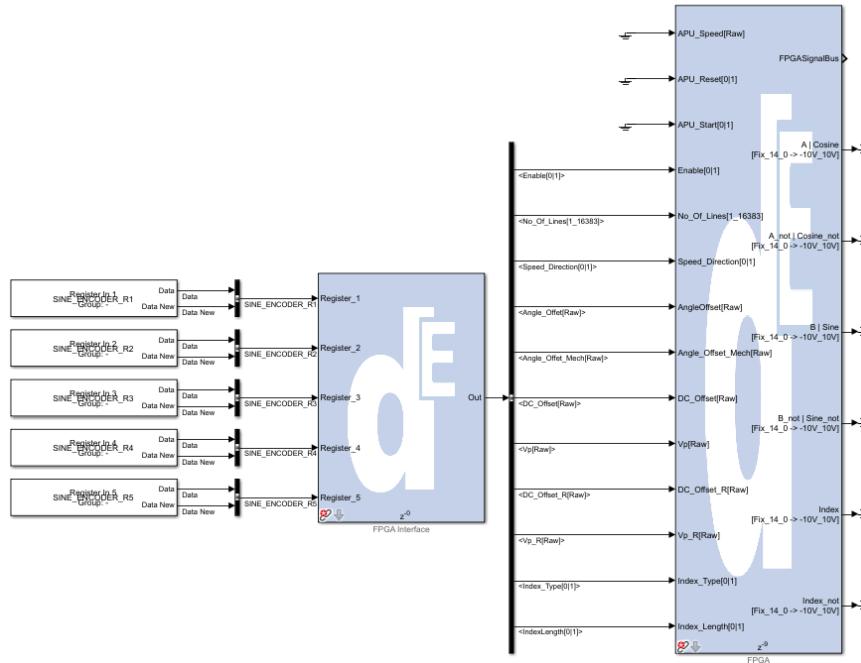


Figure 35: FPGA interface

Sensors: Hall Encoder

Objective

The hall encoder unit (HALL_ENCODER) is designed to simulate a Hall-based encoder. The on/off switching behavior can be defined for each track individually as a function of the current angle. An offset angle can be added and the amount of pole pairs is also adjustable. The FPGA main component has to be connected to the APU interface signals.

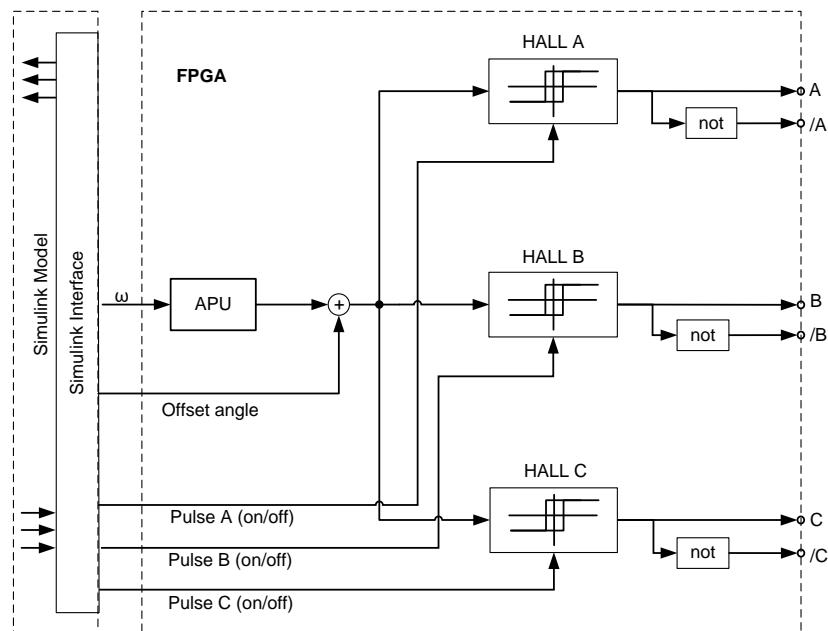


Figure 36: Equivalent circuit hall encoder

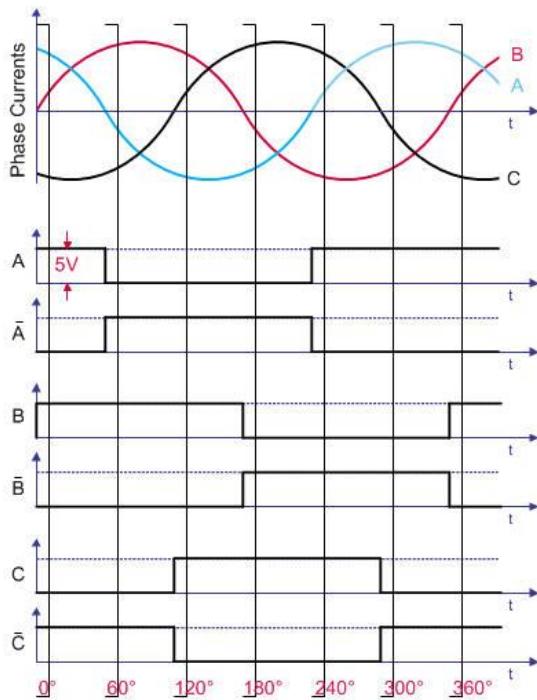


Figure 37: Hall encoder shape

The blockset contains the following elements:

- Processor Interface: HALL_ENCODER_out (Processor Interface)
- FPGA Interface: HALL_ENCODER_in (FPGA Interface)
- FPGA: HALL_ENCODER (FPGA Main Component)

Processor Output

Block

Merges the processor signals and writes them to the FPGA

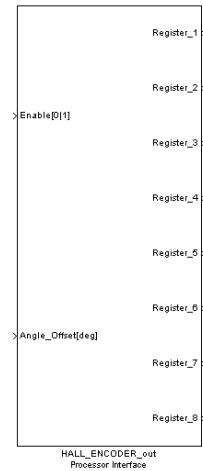


Figure 38: HALL_ENCODER_out block

Block Dialog

The processor output blockset contains the following dialog.

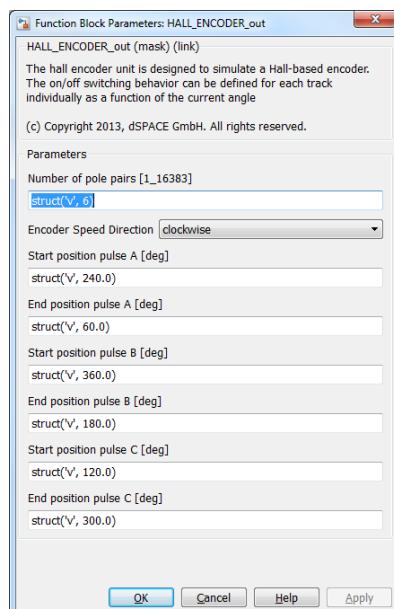


Figure 39: HALL_ENCODER_out dialog

The dialog contains the following parameters:

Name	Unit	Description	Range
Number of pole pairs	[-]	Number of pole pairs of the Hall encoder.	1 ... 16383; resolution: 1
Encoder Speed Direction	[-]	Configure the direction of the input speed (clockwise / counter-clockwise)	
Start position pulse A	[deg]	Switch on angle of pulse A	0° ... 360°; resolution: 1.37E-3
End position pulse A	[deg]	Switch off angle of pulse A	0° ... 360°; resolution: 1.37E-3
Start position pulse B	[deg]	Switch on angle of pulse B	0° ... 360°; resolution: 1.37E-3
End position pulse B	[deg]	Switch off angle of pulse B	0° ... 360°; resolution: 1.37E-3
Start position pulse C	[deg]	Switch on angle of pulse C	0° ... 360°; resolution: 1.37E-3
End position pulse C	[deg]	Switch off angle of pulse C	0° ... 360°; resolution: 1.37E-3

Input

The HALL_ENCODER_out block has the following inputs:

Name	Unit	Description	Range
Enable	[-]	Sensor signal enable	0 1
Angle_Offset	[deg]	Offset angle for Hall encoder	0° ... 360°; resolution: 1.37E-3

Output

The Processor Out block provides eight input registers whose sectioning is shown below:

Register 1

Figure 40: HALL_ENCODER_out Register 1

Name	Used Bits	Description	Range
Phase_A_on	31 ... 0	Switch on angle of phase A for Hall encoders; represents the mechanical angle of the rotor, scaled to the number of lines of the encoder	0...2^32-1 ≈ 360° electrical angle

Register 2

Figure 41: HALL_ENCODER_out Register 2

Name	Used Bits	Description	Range
Phase_A_off	31 ... 0	Switch off angle of phase A for Hall encoders; represents the mechanical angle of the rotor, scaled to the number of lines of the encoder	0...2^32-1 ≈ 360° electrical angle

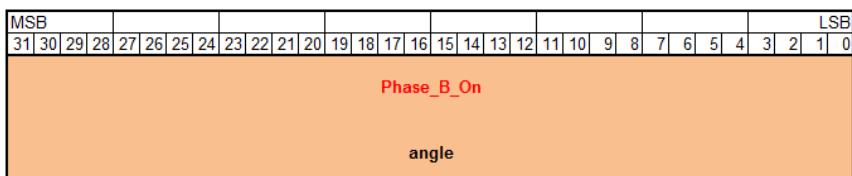
Register 3

Figure 42: HALL_ENCODER_out Register 3

Name	Used Bits	Description	Range
Phase_B_on	31 ... 0	Switch on angle of phase B for Hall encoders; represents the mechanical angle of the rotor, scaled to the number of lines of the encoder	0... $2^{32}-1 \hat{=} 360^\circ$ electrical angle

Register 4

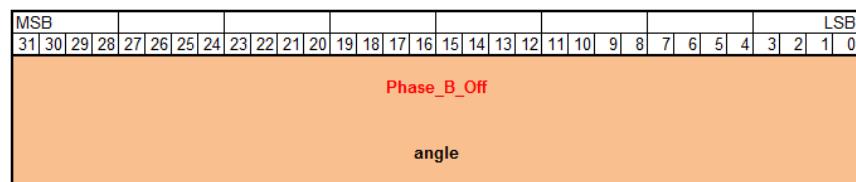


Figure 43: HALL_ENCODER_out Register 4

Name	Used Bits	Description	Range
Phase_B_off	31 ... 0	Switch off angle of phase B for Hall encoders; represents the mechanical angle of the rotor, scaled to the number of lines of the encoder	0... $2^{32}-1 \hat{=} 360^\circ$ electrical angle

Register 5

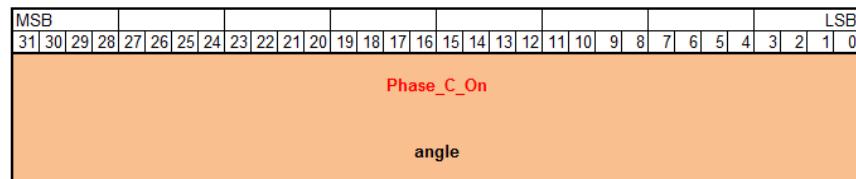


Figure 44: HALL_ENCODER_out Register 5

Name	Used Bits	Description	Range
Phase_C_on	31 ... 0	Switch on angle of phase C for Hall encoders; represents the mechanical angle of the rotor and scaled to the number of lines of the encoder	0... $2^{32}-1 \hat{=} 360^\circ$ electrical angle

Register 6

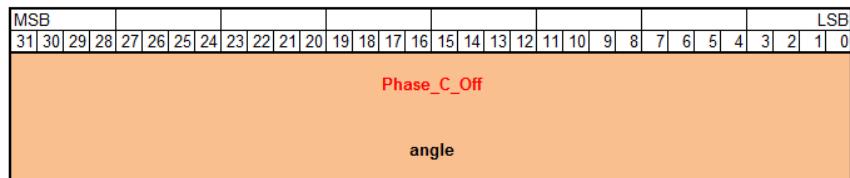


Figure 45: HALL_ENCODER_out Register 6

Name	Used Bits	Description	Range
Phase_C_off	31 ... 0	Switch off angle of phase C for Hall encoders; represents the mechanical angle of the rotor scaled to the number of lines of the encoder	0... $2^{32}-1 \hat{=} 360^\circ$ electrical angle

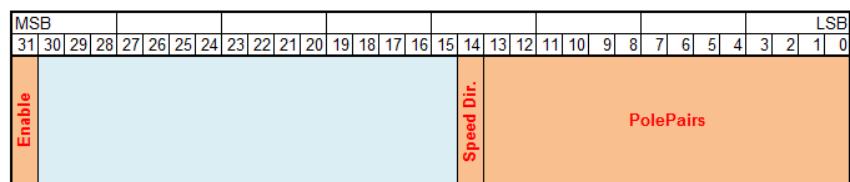
Register 7

Figure 46: HALL_ENCODER_out Register 7

Name	Used Bits	Description	Range
PolePairs	13 ... 0	Number of pole pairs	1 ... 16383
Speed Direction	14	Configure the speed direction (clockwise / counter-clockwise)	0 1
SPARE	30 ... 14		
Enable	31	Enable signal output	0 1

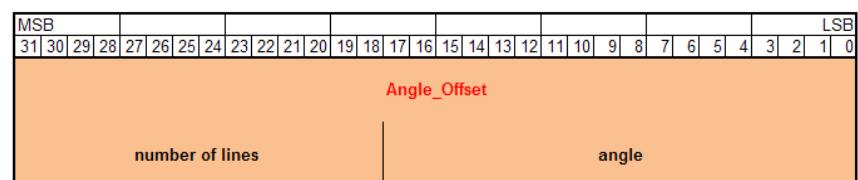
Register 8

Figure 47: HALL_ENCODER_out Register 8

Name	Used Bits	Description	Range
Angle_Offset	31 ... 0	The offset angle represents the mechanical angle of the rotor, scaled to the number of pole pairs of the encoder.	0... $2^{18}-1 \hat{=} 360^\circ$ electrical angle

FPGA Main Component

Block



Figure 48: HALL_ENCODER FPGA Main block

Block Dialog

The FPGA main blockset contains the following dialog.

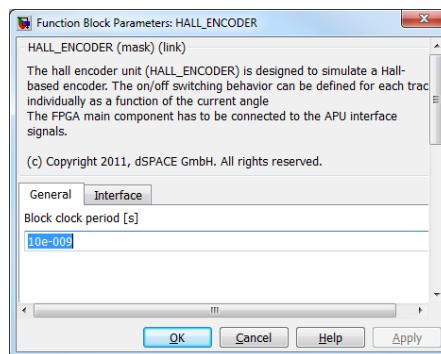


Figure 49: HALL_ENCODER FPGA Main block dialog

On the page Interface you can define the parameters of the input and output registers for automatic interface generation. The dialog also contains a button to start interface generation. For more details about automatic interface generation, refer to the XSG Utils documentation chapter Automatic Interface Generation. The FPGA clock period can be configured too.

Input

The main block has the following inputs:

Name	Unit	Description	Format
APU_Speed	[Raw]	Actual speed	Fix_30_0
APU_Reset	[-]	Reset on high state	Bool
APU_Start	[-]	Start on high state	UFix_1_0
Enable	[-]	Sensor signal enable	UFix_1_0
PolePairs	[-]	Pole pairs of the Hall encoder	UFix_14_0
Speed_Direction	[-]	Configure the speed direction (clockwise / counter-clockwise)	UFix_1_0
Angle_Offset	[Raw]	Offset angle for Hall encoder	UFix_32_0
Phase_A_on	[Raw]	Switch on angle for phase A	UFix_32_0
Phase_A_off	[Raw]	Switch off angle for phase A	UFix_32_0
Phase_B_on	[Raw]	Switch on angle for phase B	UFix_32_0
Phase_B_off	[Raw]	Switch off angle for phase B	UFix_32_0
Phase_C_on	[Raw]	Switch on angle for phase C	UFix_32_0
Phase_C_off	[Raw]	Switch off angle for phase C	UFix_32_0

	If more than one encoder is mounted on a shaft, keep in mind that any modifications of the parameter Speed_Direction and PolePairs during runtime need a reset of the overall angular processing units (APUs).
-------------------------------------------------------------------------------------	----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

Output

The HALL_ENCODER main block has the following outputs:

Name	Unit	Description	Format
FPGASingalBus	Bus	Bus containing the block's most significant signals	-
A	[-]	Track A	Bool
\bar{A}	[-]	Track \bar{A}	Bool
B	[-]	Track B	Bool
\bar{B}	[-]	Track \bar{B}	Bool
C	[-]	Track C	Bool
\bar{C}	[-]	Track \bar{C}	Bool
Enable	[-]	Enable signal output	Bool

Processor Input

Block

There is no processor input available for this blockset.

Interface Examples

Processor blocks

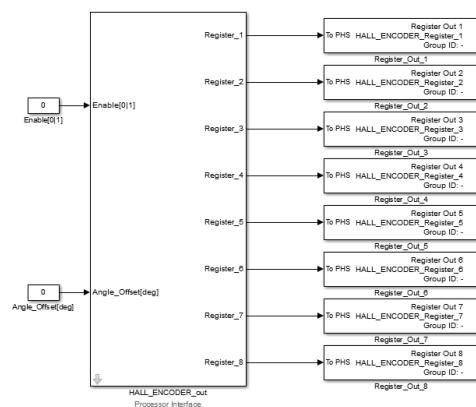


Figure 50: Processor interface

FPGA block

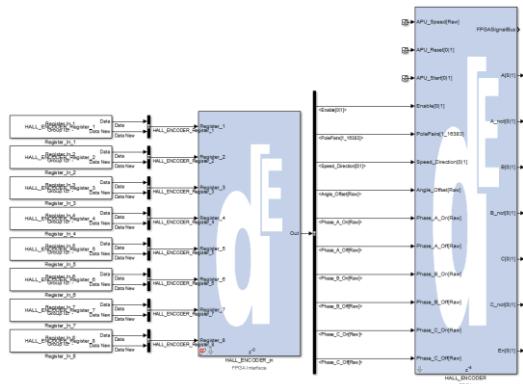


Figure 51: FPGA interface

Sensors: Resolver

Objective

Resolver position sensors are types of rotary electrical transformers used for measuring degrees of rotation. They are mainly based on the concept of encoding the shaft angle into sine and cosine angles. It features the online tune ability for the number of pole pairs, the transformation ratio, position offset and a dynamic delay for the excitation. Next to this also fault simulation for the sine/cosine magnitude and position is supported. The FPGA main component has to be connected to the APU interface signals.

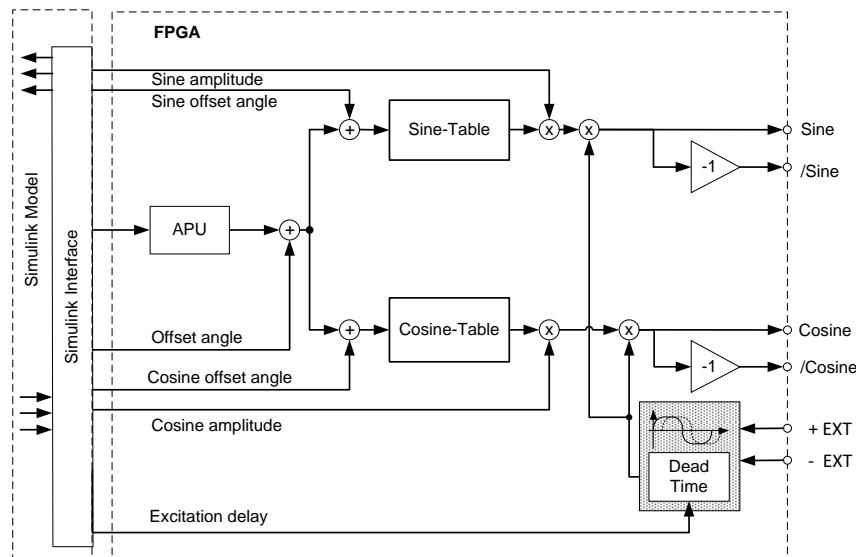


Figure 52: Equivalent circuit resolver

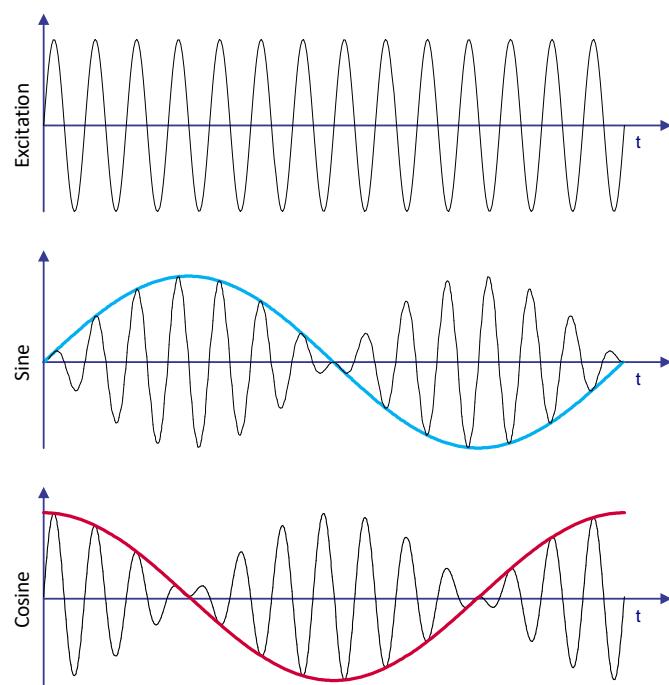


Figure 53: Resolver shape over 360 deg electric

The blockset contains the following elements:

- Processor Interface: RESOLVER_out (Processor Interface)
- FPGA Interface: RESOLVER_in (FPGA Interface)
- FPGA: RESOLVER (FPGA Main Component)

Processor Output

Block

Merges the processor signals and writes them to the FPGA



Figure 54: RESOLVER_out block

Block Dialog

The processor output block set contains the following dialog.

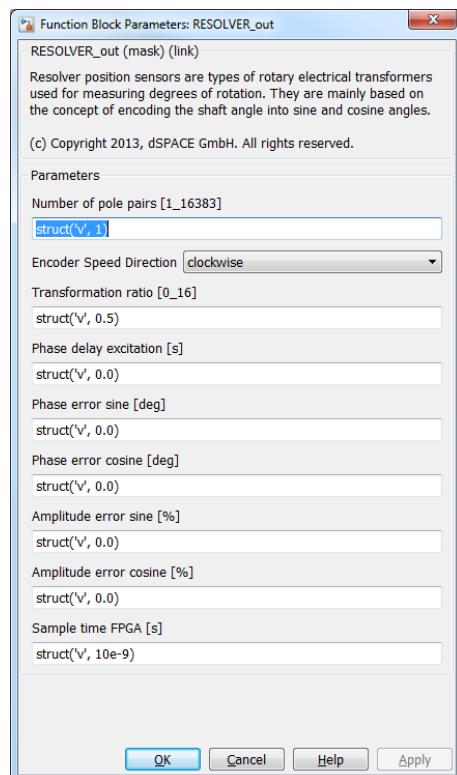


Figure 55: RESOLVER_out dialog

The dialog blockset has the following parameters:

Name	Unit	Description	Range
Number of pole pairs	[-]	Number of pole pairs of the resolver	1 ... 16383; resolution: 1
Encoder Speed Direction	[-]	Configure the direction of the input speed (clockwise / counter-clockwise)	
Transformation Ratio	[-]	Transformation ratio for the resolver sine and cosine output signals	0 ... 16 resolution: 244E-6
Phase delay excitation	[s]	Delay time for external excitation signal	0 ... 409.5E-6 s resolution: 100E-9
Phase error sine	[deg]	Phase error of the sine signal	0° ... 360° resolution: 5.49E-6
Phase error cosine	[deg]	Phase error of the cosine signal	0° ... 360° resolution: 5.49E-6
Amplitude error sine	[%]	Output factor of the sine signal	-100% ... 100% resolution: 3.05E-6
Amplitude error cosine	[%]	Output factor of the cosine signal	-100% ... 100% resolution: 3.05E-6
Sample time FPGA	[s]	Sample time of the FPGA	-

Input

The RESOLVER_out block has the following inputs:

Name	Unit	Description	Range
Enable	[-]	Sensor signal enable	0 1
Angle_Offset	[deg]	Offset angle for incremental encoder	0° ... 360°; resolution: 1.37E-3

Output

The processor out block provides five input registers whose sectioning is shown below:

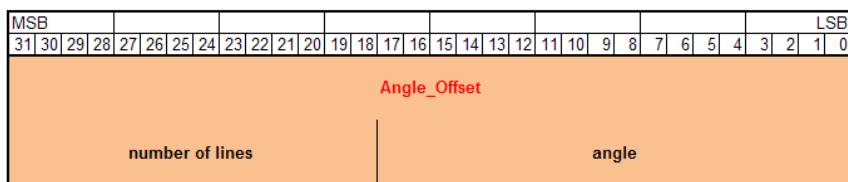
Register 1

Figure 56: RESOLVER_out Register 1

Name	Used Bits	Description	Range
Angle_Offset	31 ... 0	The offset angle represents the mechanical angle of the rotor, scaled to the number of pole pairs	0...2^18-1 ≈ 360° electrical angle

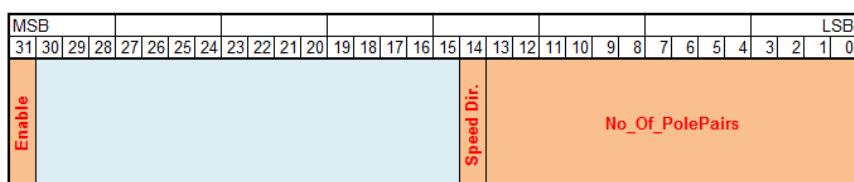
Register 2

Figure 57: RESOLVER_out Register 2

Name	Used Bits	Description	Range
PolePairs	13 ... 0	Number of pole pairs in range 1...16383 and a resolution of 1	1 ... 16383
Speed Direction	14	Configure the speed direction (clockwise / counter-clockwise)	0 1
SPARE	30 ... 14		
Enable	31	Enable signal output	0 1

Register 3

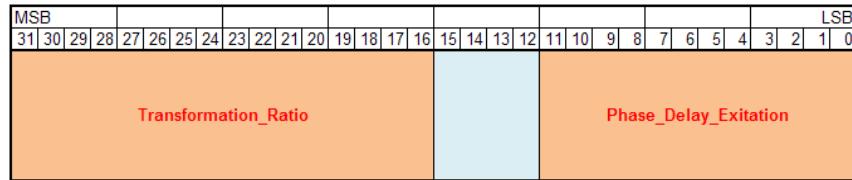


Figure 58: RESOLVER_out Register 3

Name	Used Bits	Description	Range
Phase_Delay_Excitation	11 ... 0	Delay time for external excitation signal	0 ... 4096; One bit represents one FPGA sample time step
SPARE	15 ... 12		
Transformation_Ratio	31 ... 16	Transformation ratio for the resolver sine and cosine output	0 ... 2^16

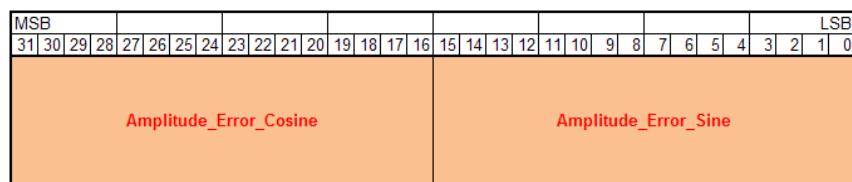
Register 4

Figure 59: RESOLVER_out Register 4

Name	Used Bits	Description	Range
Amplitude_Error_Sine	15 ... 0	Output factor for the resolver sine signal in % with respect to its output signal	0...2^14 ≈ positive percent; 2^14+1...2^15 ≈ negative percent (two's complement)
Amplitude_Error_Cosine	31 ... 16	Output factor for the resolver cosine signal in % with respect to its output signal	0...2^14 ≈ positive percent; 2^14+1...2^15 ≈ negative percent (two's complement)

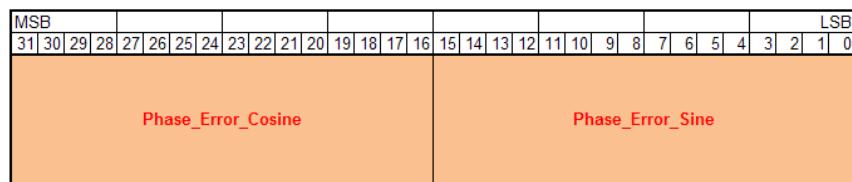
Register 5

Figure 60: RESOLVER_out Register 5

Name	Used Bits	Description	Range
Phase_Error_Sine	15 ... 0	Phase error for the sine signal	0 ... 2^16-1
Phase_Error_Cosine	31 ... 16	Phase error for the cosine signal	0 ... 2^16-1

FPGA Main Component

Block

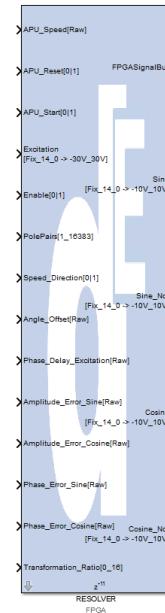


Figure 61: RESOLVER FPGA Main block

Block Dialog

The FPGA main blockset contains the following dialog.

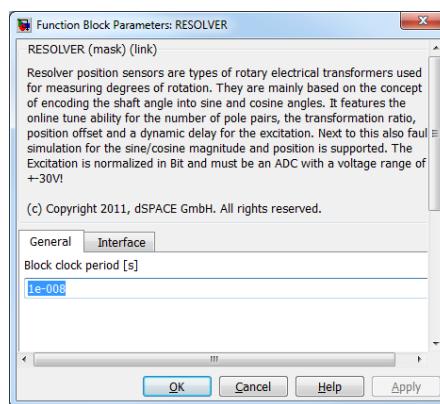


Figure 62: RESOLVER FPGA Main block dialog

On the page Interface you can define the parameters of the input and output registers for automatic interface generation. The dialog also contains a button to

start interface generation. For more details about automatic interface generation, refer to the XSG Utils documentation chapter Automatic Interface Generation. The FPGA clock period can be configured too.

Input

The main block has the following inputs:

Name	Unit	Description	Format
APU_Speed	[Raw]	Actual speed	Fix_30_0
APU_Reset	[-]	Reset on high state	Bool
APU_Start	[-]	Start on high state	UFix_1_0
Excitation	[Bit]	Excitation of the resolver (direct connection to an analog input possible)	Fix_14_0
Enable	[-]	Sensor signal enable	UFix_1_0
PolePairs	[-]	Pole pairs of the resolver	UFix_14_0
Speed_Direction	[-]	Configure the speed direction (clockwise / counter-clockwise)	UFix_1_0
Angle_Offset	[Raw]	Offset angle for encoder	UFix_32_0
Phase_Delay_Excitation	[Raw]	Delay time for external excitation signal	UFix_12_0
Amplitude_Error_Sine	[Raw]	Amplitude error of the sine track	Fix_16_12
Amplitude_Error_Cosine	[Raw]	Amplitude error of the cosine track	Fix_16_12
Phase_Error_Sine	[Raw]	Phase error for the sine signal	UFix_16
Phase_Error_Cosine	[Raw]	Phase error for the cosine signal	UFix_16
Transformation_Ratio	[-]	Transformation ratio	UFix_16_12



If more than one encoder is mounted on a shaft, keep in mind that any modifications of the parameter Speed_Direction and PolePairs during runtime need a reset of the overall angular processing units (APUs).

Output

The RESOLVER main block has the following outputs:

Name	Unit	Description	Format
FPGASingalBus	Bus	Bus containing the block's most significant signals	-
Sine	[Bit]	Track Sine	Fix_14_0
Sine	[Bit]	Track Sine	Fix_14_0
Cosine	[Bit]	Track Cosine	Fix_14_0
Cosine	[Bit]	Track Cosine	Fix_14_0

Processor Input

Block There is no processor input available for this blockset.

Interface Examples

Processor blocks

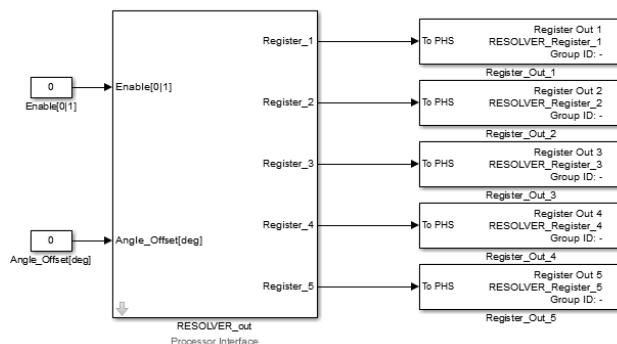


Figure 63: Processor interface

FPGA block

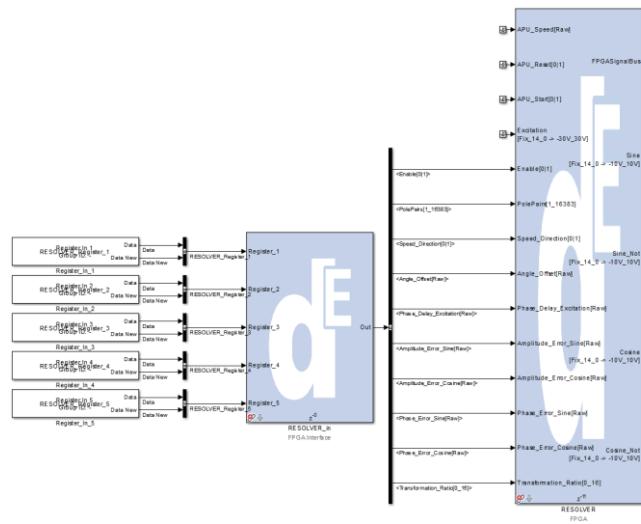


Figure 64: FPGA interface

Sensors: SSI Encoder

Objective

Position sensors often use a serial sensor interface (SSI) for communication with the overall controller. To realize a SSI data frame the following blocks are located in the library.

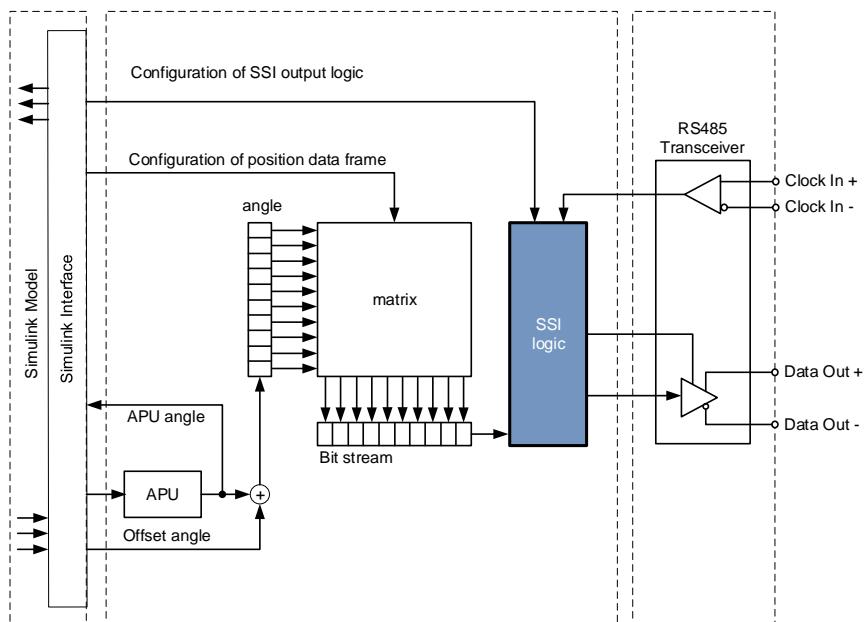
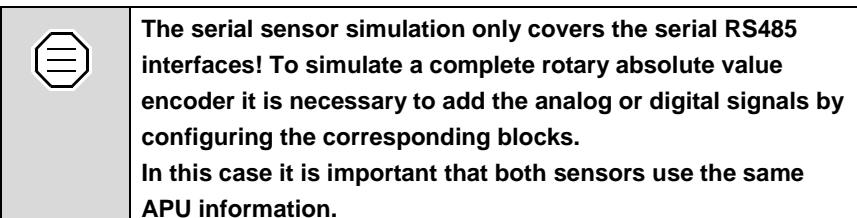


Figure 65: Implemented schematic of SSI encoder

The blockset contains the following elements:

- Processor Interface: SSI_ENCODER_out (Processor Interface)
- FPGA Interface: SSI_ENCODER_in (FPGA Interface)
- FPGA: SSI_ENCODER (FPGA Main Component)

Processor Output

Block

Merges the processor signals and writes them to the FPGA



Figure 66: SSI_ENCODER_out block

Block Dialog

The processor output block set contains two dialogs. In the first one, the general parameters can be configured.

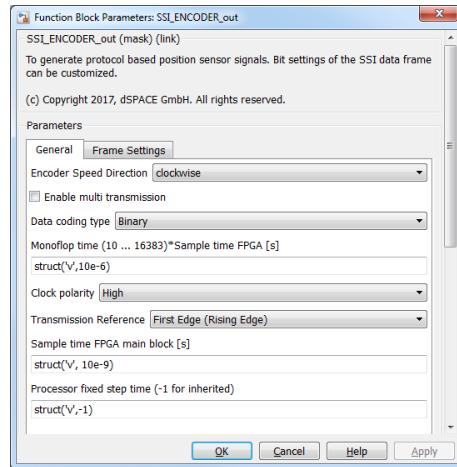


Figure 67: SSI_ENCODER_out dialog; page “General”

The dialog blockset has the following parameters:

Name	Unit	Description	Range
Encoder Speed Direction	[-]	Configure the direction of the input speed (clockwise / counter-clockwise)	
Enable multi transmission	[-]	Used to select between "Disabled (Insert zero bits)" or "Enabled (Ring buffer operation)"	
Data coding type	[-]	Used to select the bit code: - Binary code - Gray code (Payload) - Gray code (Data Frame)	
Monoflop time	[s]	Used to enter the monoflop time in seconds. When the least significant bit is received by the controller, the pulse stream is terminated (logic 0). After the monoflop time interval the data line output returns to logic 1.	(10 ... 16383) * Ts_FPGA
Clock polarity	[-]	Used to configure the polarity of the clock signal (High or Low)	
Transmission Reference	[-]	Define the reference for transmission (rising or falling edge)	
Sample time FPGA	[s]	Sample time of the FPGA	-
Processor fixed step time	[s]	Step time of the processor task; a different sample time can be set in offline simulation to simulate the processor interface communication behavior	-

On the second page the data frame can be configured.

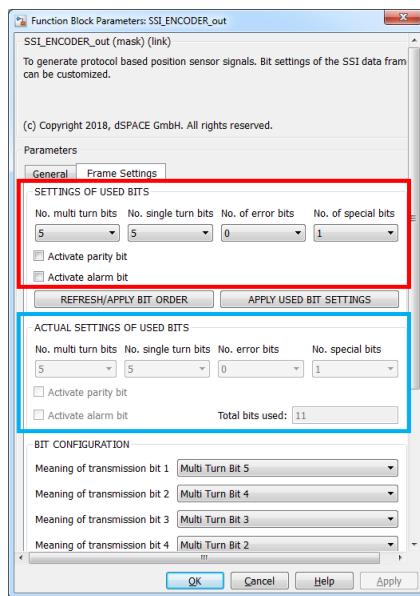


Figure 68: SSI_ENCODER_out dialog; page “Frame Settings”

In the GUI the actual used bit settings of the data frame is shown (marked in blue in the figure above). To configure other data frames the actual bit settings can be configured with the edit ports which are marked with red. After configuration of the data frame the settings can be assumed with the button “APPLY USED BIT SETTINGS”. The block GUI will be modified and the actual possible popups for the bit configuration will be activated. With the button “REFRESH/APPLY BIT ORDER” the actual bit settings will be updated and the default bit configuration will be set. After applying the changes of the GUI with the button “Apply” the internal simulink block configuration will be modified.

Input

The SSI_ENCODER_out block has the following inputs:

Name	Unit	Description	Range
Enable	[-]	Sensor signal enable	0 1
Angle_Offset	[deg]	Offset angle for incremental encoder	0° ... 2^18 * 360°; resolution: 1.37E-3
Following settings depends on GUI settings:			
Error_Bit_x	[-]	Error bit x; 0 ≤ x ≤ 5, if the number of bits are set to zero no input ports are available	0 1
Special_Bit_x	[-]	Special bit x; 0 ≤ x ≤ 5, if the number of bits are set to zero no input ports are available	0 1
Alarm_Bit	[-]	Alarm bit; if the alarm bit is deactivated no input ports are available	0 1

Output

The processor out block provides the following registers whose sectioning is shown below:

Register 1



Figure 69: SSI_ENCODER_out Register 1

Name	Used Bits	Description	Range
Number of Bits	5 ... 0	Number of actual used bits	0...2^5-1
Bit configuration (data)	11 ... 6	Bit configuration of data frame with respect to actual address; mapping of vector 0: Single Turn MSB -17 1: Single Turn MSB -16 ... 17: Single Turn MSB 18: Multi Turn Bit 1 ... 35: Multi Turn Bit 18 36: Parity Bit 37: Error Bit 1 ... 41: Error Bit 5 42: Alarm Bit 43: Special Bit 1 ... 50: Special Bit 8 51: Constant 0 52: Constant 1	0...2^5-1
Bit configuration (address)	17 ... 12	Actual address of committed bit configuration data	0...2^5-1
Configuration changed	18	Flag which starts the update of the internal bit configuration of the data frame	0 1
Angle Offset	22 ... 19	Angle offset (Bit 32 – 35)	0 ... 2^4-1
SPARE	23		
Speed Direction	24	Configure the speed direction (clockwise / counter-clockwise)	0 1
Transmission Ref	25	Define transmission reference 0: first edge 1: following edge	0 1
Parity	26	Definition of parity bit 0: odd 1: even	0 1
Multi transmission	27	Enables multi transmission of data frame	0 1
Gray Coding	29 ... 28	Definition of gray coding 0: binary 1: gray (only of payload; no error, special, alarm bits) 2: gray (whole data frame)	0 1 2

Clock Polarity	30	Polarity of clock signal 0: idle low 1: idle high	0 1
Enable	31	Enable signal output	0 1

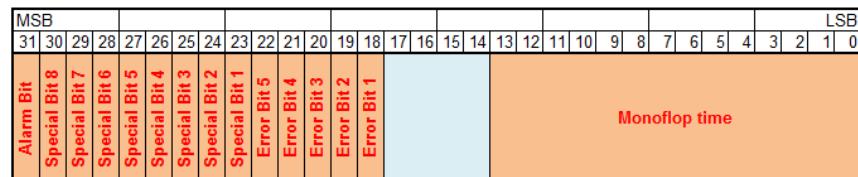
Register 2

Figure 70: SSI_ENCODER_out Register 2

Name	Used Bits	Description	Range
Monoflop time	13 ... 0	Actual monoflop time	0 ... 2^14-1
SPARE	17 ... 14		
Error Bit 1	18	Error bit 1	0 1
...			
Error Bit 5	22	Error bit 5	0 1
Special Bit 1	23	Special bit 1	0 1
...			
Special Bit 8	30	Special bit 8	0 1
Alarm Bit	31	Alarm Bit	0 1

Register 3

Figure 71: SSI_ENCODER_out Register 3

Name	Used Bits	Description	Range
Angle Offset	31 ... 0	Angle offset (Bit 0 – 31)	0 ... 2^32-1

FPGA Main Component

Block

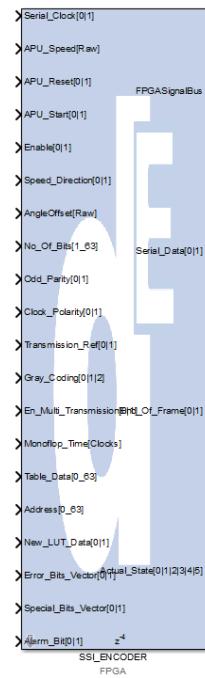


Figure 72: SSI_ENCODER FPGA Main block

Block Dialog

The FPGA main blockset contains the following dialog.

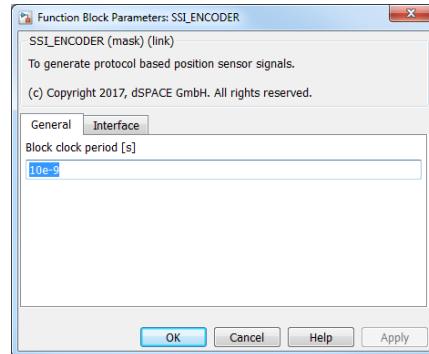


Figure 73: SSI_ENCODER FPGA Main block dialog

On the page Interface you can define the parameters of the input and output registers for automatic interface generation. The dialog also contains a button to start interface generation. For more details about automatic interface generation, refer to the XSG Utils documentation chapter Automatic Interface Generation. The FPGA clock period can be configured too.

Input

The main block has the following inputs:

Name	Unit	Description	Format
Serial_Clock	[0 1]	Clock signal of SSI	Bool or UFix_1_0
APU_Speed	[Raw]	Actual speed	Fix_30_0
APU_Reset	[-]	Reset on high state	Bool
APU_Start	[-]	Start on high state	UFix_1_0
Enable	[-]	Sensor signal enable	UFix_1_0
Speed_Direction	[-]	Configure the speed direction (clockwise / counter-clockwise)	UFix_1_0
Angle_Offset	[Raw]	Offset angle for encoder 18 Bits Multiturn (35 – 18) 18 Bits Singleturn (17 – 0)	UFix_36_0
No_Of_Bits	[-]	Number of actual used bits	UFix_6_0
Odd_Parity	[-]	Definition of parity bit 0: odd 1: even	UFix_1_0
Clock_Polarity	[-]	Polarity of clock signal 0: idle low 1: idle high	UFix_1_0
Transmission_Ref	[-]	Define transmission reference 0: first edge 1: following edge	UFix_1_0
Gray_Coding	[-]	Definition of gray coding 0: binary 1: gray (only of payload; no error, special, alarm bits) 2: gray (whole data frame)	UFix_2_0
En_Multi_Transmission	[-]	Enables multi transmission of data frame	UFix_1_0
Monoflop_Time	[clocks]	Actual monoflop time	UFix_14_0

Table_Data	[-]	Bit configuration of data frame with respect to actual address; mapping of vector 0: Single Turn MSB -17 1: Single Turn MSB -16 ... 17: Single Turn MSB 18: Multi Turn Bit 1 ... 35: Multi Turn Bit 18 36: Parity Bit 37: Error Bit 1 ... 41: Error Bit 5 42: Alarm Bit 43: Special Bit 1 ... 50: Special Bit 8 51: Constant 0 52: Constant 1	UFix_6_0
Address	[-]	Actual address of committed bit configuration data	UFix_6_0
New_LUT_Data	[-]	Flag which starts the update of the internal bit configuration of the data frame	UFix_1_0
Error_Bits_Vecotr	Vector	Vector of error bits	[5x1] vector Bool
Special_Bits_Vector	Vector	Vector of special bits	[8x1] vector Bool
Alarm_Bit	[-]	Alarm Bit	Bool



If more than one encoder is mounted on a shaft, keep in mind that any modifications of the parameter Speed_Direction during runtime need a reset of the overall angular processing units (APUs).

Output

The SSI_ENCODER main block has the following outputs:

Name	Unit	Description	Format
FPGASingalBus	Bus	Bus containing the block's most significant signals	-
Serial_Data	[Bit]	Serial data of encoder	0 1
End_Of_Frame	[Bit]	Flag which marks the end of the data frame	UFix_1_0
Actual_State	[Bit]	Actual state of transmission; 0: Not Defined 1: Slave Disabled 2: Waiting for Clock 3: Transmitting Data 4: MF_Time 5: MT Active	UFix_5_0

Processor Input

Block

There is no processor input available for this blockset.

Interface Examples

Processor blocks

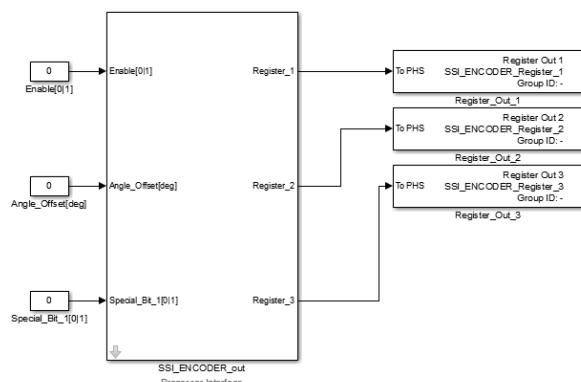


Figure 74: Processor interface

FPGA block

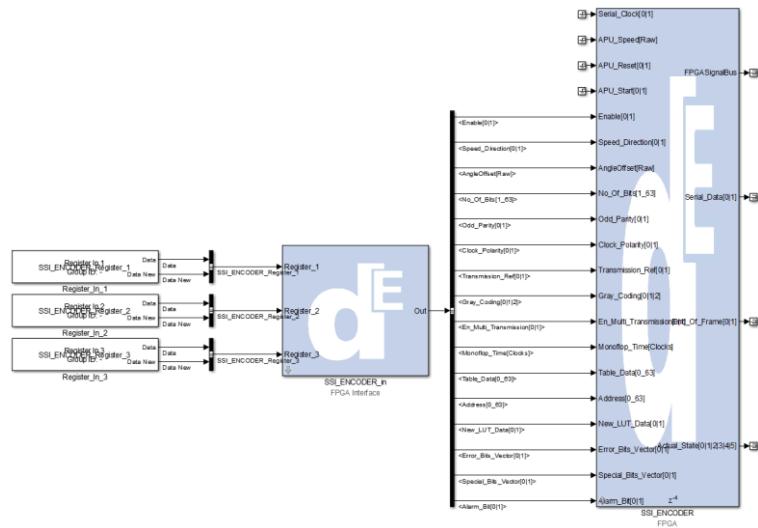


Figure 75: FPGA interface

Sensors: EnDat Encoder

Objective

Position sensors often use a digital parameter interface (like EnDat) for communication with the overall controller. To realize an EnDat data frame the following blocks are located in the library.

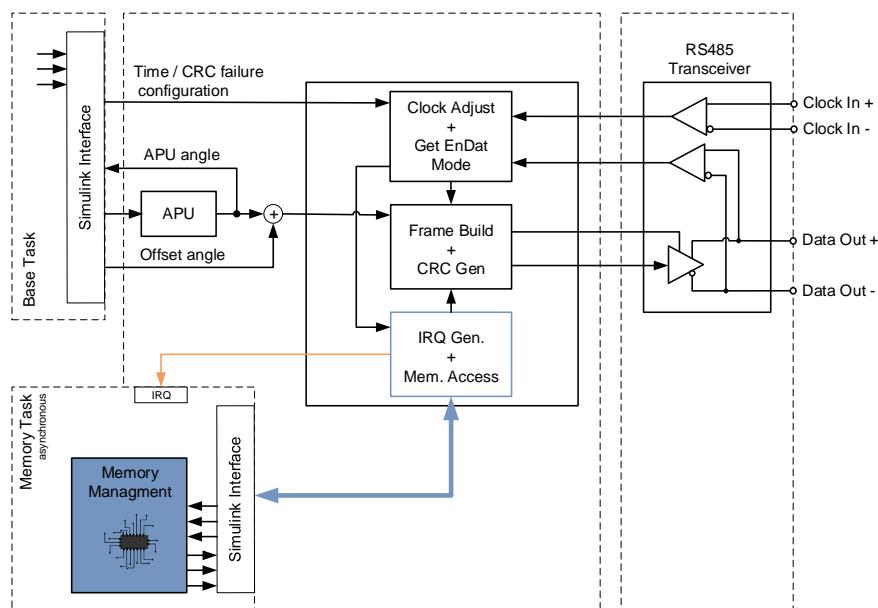
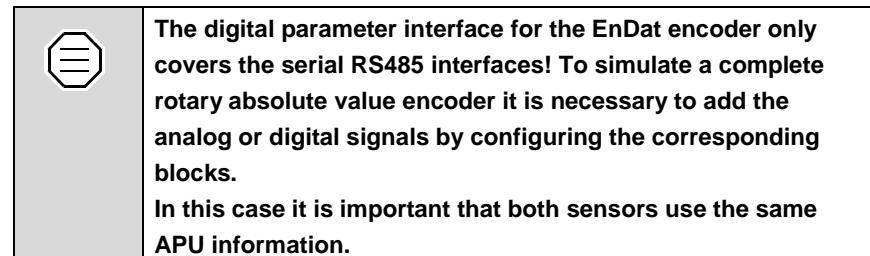


Figure 76: Implemented schematic of EnDat encoder

The blockset contains the following elements:

- Processor Interface: ENDAT_ENCODER_out (Processor Interface)
- FPGA Interface: ENDAT_ENCODER_in (FPGA Interface)
- FPGA: ENDAT_ENCODER (FPGA Main Component)

To ensure time requirements of memory access between the master and the encoder the memory management is realized in an asynchronous task. This guarantee a synchronous called task with respect to the memory access request from the connected master. Additionally, a frequently interrupt with respect to the main task can be also parameterized. This functionality is realized with the following elements:

- Processor Interface: ENDAT_ENCODER_MEMORY_out
(Processor Interface)
- FPGA Interface: ENDAT_ENCODER_MEMORY_in
(FPGA Interface)
- FPGA Interface: ENDAT_ENCODER_MEMORY_out
(FPGA Interface)
- Processor Interface: ENDAT_ENCODER_MEMORY_in
(Processor Interface)

	<p>To get an easy startup with the overall model implementation, please refer to the implementation example. The model can be opened via double clicking on the “Open Implementation Example” or via the Matlab command:</p> <p>“xsg_electric_DEMO_installDemo('Template_ImpMDL\EnDat');”</p>
	<p>Please use a high priority task for the memory access of the encoder.</p>

Processor Output (Base Task)

Block	Merges the processor signals e.g. time requirements, failure and error bits in the base task and writes them to the FPGA
--------------	--------------------------------------------------------------------------------------------------------------------------



Figure 77: ENDAT_ENCODER_out block

Block Dialog	The processor output block set contains two dialogs. In the first one, the general parameters can be configured.
---------------------	------------------------------------------------------------------------------------------------------------------

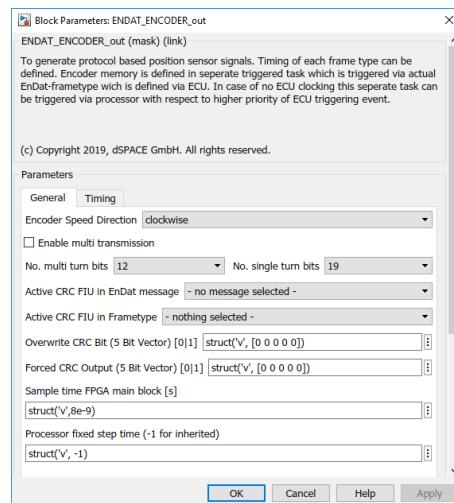


Figure 78: ENDAT_ENCODER_out dialog; page “General”

The dialog blockset has the following parameters:

Name	Unit	Description	Range
Encoder Speed Direction	[-]	Configure the direction of the input speed (clockwise / counter-clockwise)	
Enable multi transmission	[-]	Used to select between “Disabled (Insert zero bits)” or “Enabled (Ring buffer operation)”	
Number of multi turn bits	[-]	Number of multi turn bits	0 ... 22
Number of single turn bits	[-]	Number of single turn bits	0 ... 42
Activate CRC FIU on EnDat message	[-]	Signal selector on which type of EnDat dataframe the CRC failure vector will be enabled	
Activate CRC FIU Frametype	[-]	Specify the type of frame in which the CRC FIU Bit pattern will be integrated	
Overwrite CRC Bit	[-]	Specify which CRC bit will be overwritten with the forced CRC Bit	5 Bit Vector
Forced CRC Bit	[-]	Forced CRC Bit which will be used for output	5 Bit Vector
Sample time FPGA	[s]	Sample time of the FPGA	-
Processor fixed step time	[s]	Step time of the processor task; a different sample time can be set in offline simulation to simulate the processor interface communication behavior	-

On the second page the timing behavior can be configured.

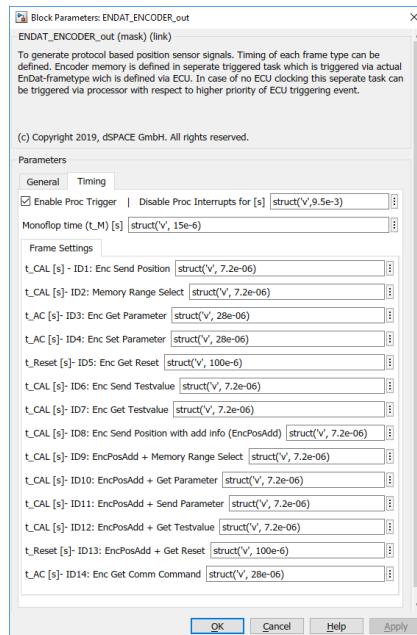


Figure 79: ENDAT_ENCODER_out dialog; page “Timing”

The dialog have the following parameters:

Name	Unit	Description	Range
Enable Proc Trigger	[-]	Enables the interrupt generation synchronous to the base task	on off
Disable Proc Interrupts for	[s]	If the timing requirements of the encoder – master communication is nearly the maximum performance, it can lead to processor overload and overruns. To minimize the overall processor load the spare time between the last and the next interrupt (which will be generated from the processor) can be parameterized	1 ... 10e-6 resolution: Ts_FPGA * 2^7
Monoflop time	[s]	Specified monoflop time which detects end of dataframe	0 ... (2^25-1) * Ts_FPGA resolution: Ts_FPGA
Frame depending settings:			
t_x [s] - IDx	[s]	In this group the calculation, reset and ac time can be set for each frame type of the EnDat protocol	0 ... (2^25-1) * Ts_FPGA resolution: Ts_FPGA

Input

The ENDAT_ENCODER_out block has the following inputs:

Name	Unit	Description	Range
Enable	[-]	Sensor signal enable	0 1
Angle_Offset	[deg]	Offset angle for incremental encoder	0° ... 2^18 * 360°; resolution: 1.37E-3
Error_Bit_1	[-]	Error bit 1 0: no error 1: error	0 1
Error_Bit_2	[-]	Error bit 2; only used in EnDat 2.2 frames 0: error 1: no error	0 1
WRN_Bit	[-]	Warning bit	0 1
BUSY_Bit	[-]	Bit which detects the encoder is busy	0 1
RM_Busy	[-]	Bit which detects the reference position was reached	0 1
Test_Values	[-]	Test values of the encoder	0 ... 2^40-1

Output

The processor out block provides multiplexed and non-multiplexed signals. Each register build-in is not part of the documentation.

Processor Output (Memory Access Task)

Block

Implementation of the encoder memory and logic. Encoder memory can be defined via parameter vector and write/read process is realized via buffer access to the FPGA.

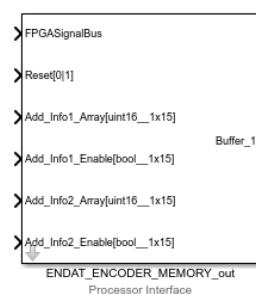


Figure 80: ENDAT_ENCODER_MEMORY out block

Block Dialog

The processor output block set contains the following dialog.

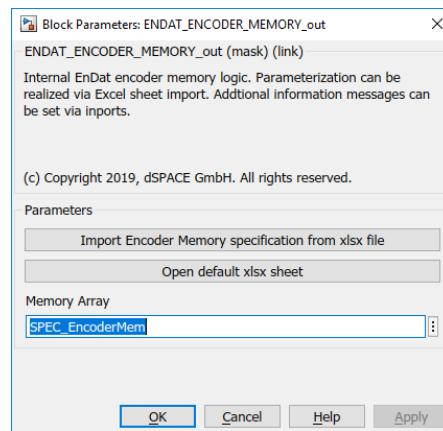


Figure 81: ENDAT_ENCODER_MEMORY_out dialog

The dialog blockset has the following parameters/buttons:

Name	Unit	Description	Range
Button: Import Encoder Memory specification from xlsx file	[-]	The encoder memory of an EnDat encoder is implemented as a superset of the specification of EnDat. This array can be defined in a xlsx file can be imported.	
Open default xlsx sheet	[-]	Opens the default parameter sheet of the encoder type EQI1331 (manufacturer: HEIDENHAIN)	
Memory Array	Array	Memory array of the encoder	

Input

The ENDAT_ENCODER_MEMORY_out block has the following inputs:

Name	Unit	Description	Range
FPGASignalBus	Bus	Feedback signal bus of the ENDAT_ENCODER_MEMORY_out block	
Reset	[-]	Reset of the internal encoder memory → initial parameterization will be loaded	0 1
Add_Info1_Array	Vector	Feedback Array of all possible additional information 1 messages	Uint16 – 1x15 vector
Add_Infor1_Enable	Vector	Enable vector of all possible additional information 1 messages; if the message is disabled the master cannot book the specify message	Bool- 1x15 vector
Add_Info2_Array	Vector	Feedback Array of all possible additional information 2 messages	Uint16 – 1x15 vector
Add_Infor2_Enable	Vector	Enable vector of all possible additional information 2 messages; if the message is disabled the master cannot book the specify message	Bool- 1x15 vector

Output

The processor out block realize the FPGA feedback via one specific buffer.

FPGA Main Component

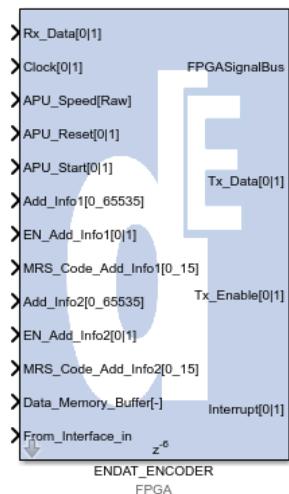
Block

Figure 82: ENDAT_ENCODER FPGA Main block

Block Dialog

The FPGA main blockset contains the following dialog.

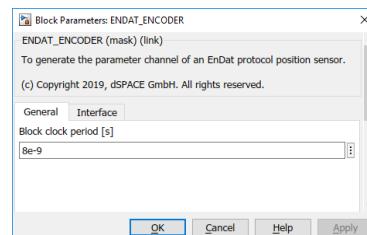


Figure 83: ENDAT_ENCODER FPGA Main block dialog

On the page Interface you can define the parameters of the input and output registers for automatic interface generation. The dialog also contains a button to start interface generation. For more details about automatic interface generation, refer to the XSG Utils documentation chapter Automatic Interface Generation. The FPGA clock period can be configured too.

Input

The main block has the following inputs:

Name	Unit	Description	Format
Rx_Data	[0 1]	Rx of the data channel	Bool or UFix_1_0
Clock	[0 1]	Clock signal	Bool or UFix_1_0
APU_Speed	[Raw]	Actual speed	Fix_30_0
APU_Reset	[-]	Reset on high state	Bool
APU_Start	[-]	Start on high state	UFix_1_0
Signals which are connected to the ENDAT_ENCODER_MEMORY_in block to the FPGA			
Add_Info1	[-]	Additional information 1 of memory access	UFix_16_0
EN_Add_Info1	[-]	Additional information 1 is enabled	UFix_1_0
MRS_Code_Add_Info1	[-]	MRS code of the actual additional information 1	UFix_4_0
Add_Info2	[-]	Additional information 2 of memory access	UFix_16_0
EN_Add_Info2	[-]	Additional information 2 is enabled	UFix_1_0
MRS_Code_Add_Info2	[-]	MRS code of the actual additional information 2	UFix_4_0
Data_Memory_Buffer	[-]	Data feedback of the selected encoder memory	UFix_32_0
From_Interface_in	[Bus]	SignalBus which is connected to the ENDAT_ENCODER_in block	



If more than one encoder is mounted on a shaft, keep in mind that any modifications of the parameter Speed_Direction during runtime need a reset of the overall angular processing units (APUs).

Output

The ENDAT_ENCODER main block has the following outputs:

Name	Unit	Description	Format
FPGASingalBus	Bus	Bus containing the block's most significant signals	-
Tx_Data	[Bit]	Tx data of the data channel of the encoder	UFix_1_0
Tx_Enable	[Bit]	Flag which control the output driver of the Tx driver	UFix_1_0
Interrupt	[Bit]	Interrupt which controls the memory access task of the encoder; direct connected the interrupt block of rtifpga	UFix_1_0

Processor Input (Base Task)

Block

There is no processor input available for this blockset.

Processor Input (Memory Access Task)

Block

All necessary feedback for the memory logic (implemented in the ENDAT_ENCODER_MEMORY_out block) is realized via the ENDAT_ENCODER_MEMORY_in block. No specific settings are necessary.

Interface Examples

Implementation Example

To get a better overview of the necessary model structure for the EnDat simulation please refer to the implementation example via the Matlab command:

```
"xsg_electric_DEMO_installDemo('Template_ImpMDL\EnDat');"
```

Sensors: HIPERFACE® Encoder

Objective

This block simulates RS485 digital parameter (UART based) channel of an encoder with HIPERFACE protocol. The functionality of the block includes standard HIPERFACE commands (Read Position, Set Position, Read Analog Value, Read Counter, Increment Counter, Reset Counter, Read Data, Store Data, Data Field Status, Create Data Field, Memory Status, Set Access Code, Read Encoder Status, Read Type Label, Encoder Reset, Set Encoder Address, Read Version and Set Serial Interface). Initial content of encoder memory is specified in Excel file which is imported in the model. Aside from normal operation, error/warning states and forcing bits in output checksum can be simulated. The block also detects protocol and data errors (e.g. parity error, checksum error, wrong data field number, etc.) and responds with the corresponding error code. Full error diagnostics is available on processor side.

The sine/cosine analog process channel of encoder should be added using SINE_ENCODER block from XSG Electric Library (refer to Physical Layer – Sine/Cosine Process Channel in official HIPERFACE documentation).

Position sensors often use a digital parameter interface (like HIPERFACE) for communication with the overall controller. To realize a HIPERFACE data frame the following blocks are located in the library.

	<p>The digital parameter interface for the HIPERFACE encoder only covers the serial RS485 interfaces! To simulate a complete rotary absolute value encoder, it is necessary to add the analog or digital signals by configuring the corresponding blocks.</p> <p>In this case it is important that both sensors use the same APU information.</p> <p>For detailed information about the HIPERFACE® definition please refer to the SICK homepage (Link).</p>
-------------------------------------------------------------------------------------	--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

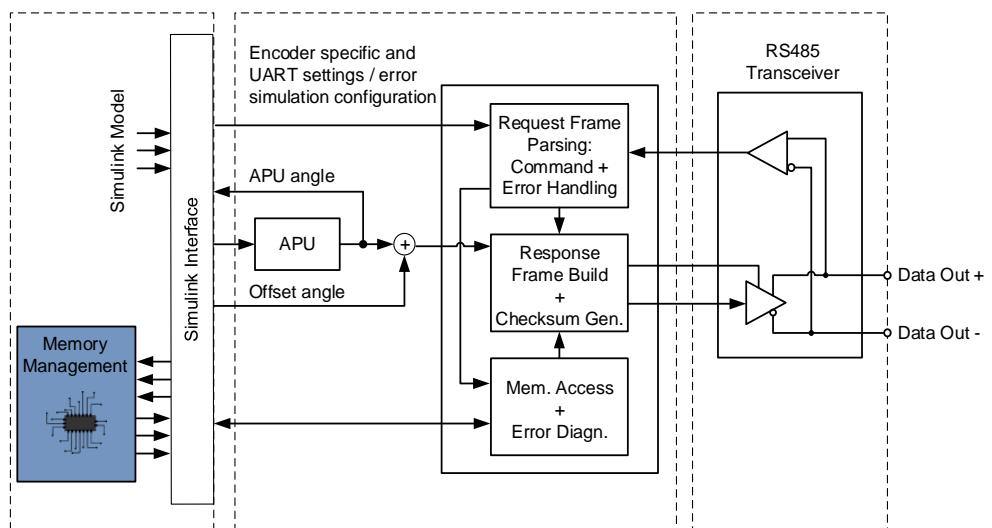


Figure 84: Implemented schematic of HIPERFACE encoder

The blockset contains the following elements:

- Processor Interface: HIPERFACE_ENCODER_out (Processor Interface)
- Processor Interface: HIPERFACE_ENCODER_in (Processor Interface)
- FPGA Interface: HIPERFACE_ENCODER_in (FPGA Interface)
- FPGA Interface: HIPERFACE_ENCODER_out (FPGA Interface)
- FPGA: HIPERFACE_ENCODER (FPGA Main Component)



To get an easy startup with the overall model implementation, please refer to the implementation example. The model can be opened via double clicking on the “Open Implementation Example” or via the Matlab command:

`“xsg_electric_DEMO_installDemo('Template_ImpMDL\Hiperface');”`

Processor Output

Block	Merges the processor signals e.g. general settings (number of single/multi turn bits, encoder type ID, etc.), UART settings (data rate, parity, timeout, Tx driver delays), error simulation settings (error/warning state, error codes), etc. and writes them to the FPGA. The block specifies user-available encoder memory size and performs memory initialization which is specified in Excel file. Additionally, the block handles protocol commands related to encoder memory (reading/storing data, creating data fields, reading memory status) or encoder settings (UART settings, encoder address, access codes) and generates Slave response frame for those commands.
-------	-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

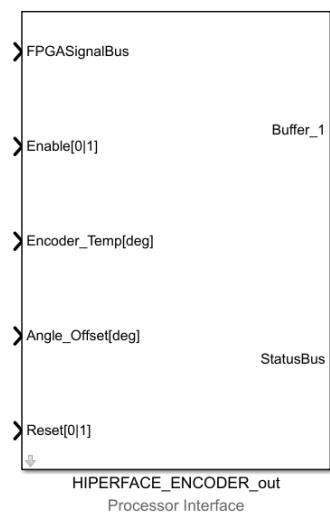


Figure 85: HIPERFACE_ENCODER_out block

Block Dialog

The processor output block set contains four dialogs. In the first one, the general parameters can be configured.

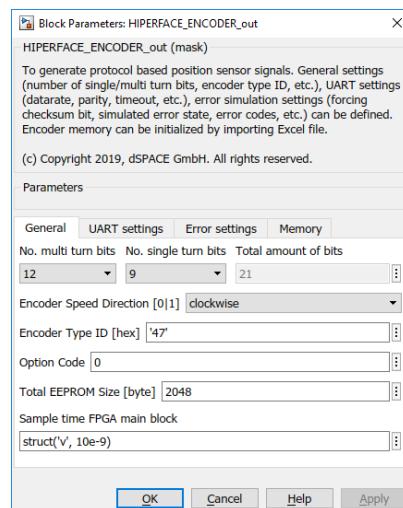


Figure 86: HIPERFACE_ENCODER_out dialog; page “General”

The first dialog has the following parameters:

Name	Unit	Description	Range
No. multi turn bits	[-]	Number of multi turn bits	0 ... 22
No. single turn bits	[-]	Number of single turn bits	0 ... 32
Encoder Speed Direction	[-]	Configure the direction of the input speed (clockwise / counter-clockwise)	0 1
Encoder Type ID	[-]	Code which defines encoder type ID (specified in datasheet in hexadecimal format)	
Option Code	[-]	Code which specifies special encoder function (usually specified in additional documentation)	
Total EEPROM Size	[byte]	Size of total encoder memory (specified in encoder datasheet)	
Sample time FPGA main block	[s]	Sample time of the FPGA	

Number of single turn bits is calculated based on number of sine/cosine periods per revolution, considering that resolution of one sine/cosine period is 5 bits for all HIPERFACE encoders. For example, if number of sine/cosine periods per revolution is equal to 16 (4 bits), with additional 5 bits per period, the number of single turn bits is equal to 9. Encoder type ID and total EEPROM size can be found in encoder datasheet, while option code designates specific encoder function (usually available in additional encoder documentation, by default equal to 0).

In the second dialog UART settings can be configured.

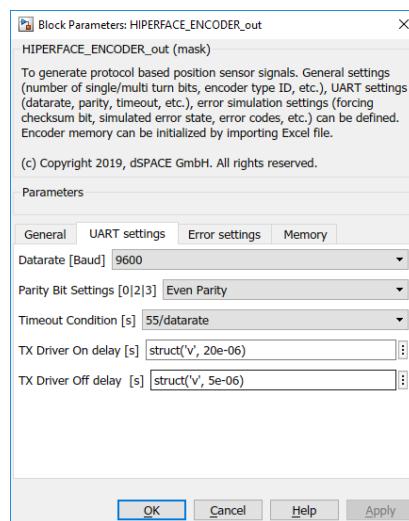


Figure 87: HIPERFACE_ENCODER_out dialog; page “UART Settings”

The second dialog has the following parameters:

Name	Unit	Description	Range
Datarate	[Baud]	Data rate of the UART communication (reception and transmission): 0: 600 1: 1200 2: 2400 3: 4800 4: 9600 5: 19200 6: 38400	0 ... 6
Parity Bit Settings	[]	Parity of the UART communication: 0: No parity bit is used 2: Even parity 3: Odd parity	0 2 3
Timeout Condition	[s]	Timeout condition which specifies end of Master request frame: 0: 22/data rate 1: 55/data rate	0 1
Tx Driver On delay [s]	[s]	Period between activation of Tx driver and start of Slave response frame transmission	0 ... (2^15-1) * Ts_FPGA resolution: Ts_FPGA
Tx Driver Off delay [s]	[s]	Period between end of Slave response frame transmission and deactivation of Tx driver	0 ... (2^13-1) * Ts_FPGA resolution: Ts_FPGA

In the third dialog Error settings can be configured.

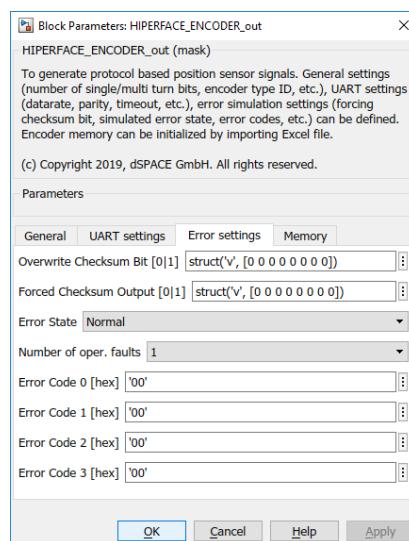


Figure 88: HIPERFACE_ENCODER_out dialog; page “Error Settings”

The third dialog has the following parameters:

Name	Unit	Description	Range
Overwrite Checksum Bit	[-]	Specifies which checksum bit will be overwritten with the forced checksum bit	8 Bit vector
Forced Checksum Bit	[-]	Forced checksum bit which will be used for output	8 Bit vector
Error State	[-]	Simulation of error/warning state: 0: normal 1: error 2: warning	0 1 2
Number of operational faults	[-]	Number of operational faults which must be read out by Master in warning state	1 ... 4
Error Code 0-4	[-]	Error codes for simulated error state (Error Code 0) or warning state (Error Code 0-4, depends on specified number of operational faults). Values are specified in hexadecimal format.	

By choosing which bits are overwritten and forcing desired bits in the output checksum, original checksum for Slave response frame (XOR function of all bytes in frame) can be modified. If simulated error state is chosen (Error State = 1), the encoder responds to every Master request frame with Read Encoder Status (50h) command and Error Code 0. If simulated warning state is chosen (Error State = 2), the encoder normally responds to every Master request frame with warning bit (bit 7 of command identifier byte) set to high state. Each time correct Master request frame with Read Encoder Status (50h) command is received, one (out of maximum four) error codes is read out and sent via Slave response frame. When all error codes are read out, the warning bit is set to low state and normal state becomes active. Regardless of the chosen simulated state, if correct Master request frame with Encoder Reset (53h) command is received or input Reset is set to high state, simulated error/warning state is reset (normal state becomes active).

In the fourth dialog Memory can be initialized.

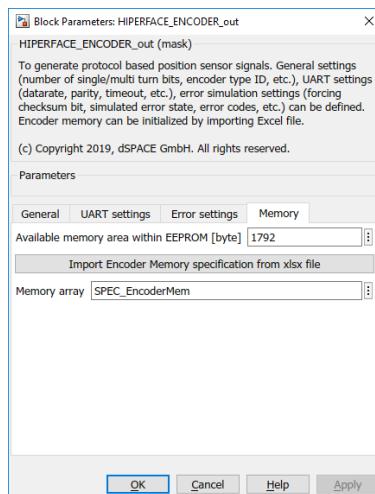


Figure 89: ENDAT_ENCODER_out dialog; page “Memory”

The fourth dialog has the following parameters:

Name	Unit	Description	Range
Available memory area within EEPROM	[byte]	User-available memory size of the encoder (specified in encoder datasheet). User can create data fields inside this memory for reading/storing data.	
Button: Import Encoder Memory specification from xlsx file	[-]	The initial encoder memory specified in xlsx file can be imported and stored in array	
Memory array	[-]	Memory array of the encoder	

Excel file which contains initial encoder memory should be imported before model compilation. In the xlsx file data fields and their content can be defined. Data field status of each data field should be defined (refer to Create Data Field (4Dh) command in official HIPERFACE documentation). The following parameters should be defined for each created data field: data field size, code disabled/enabled, used access code, read-only/write-enabled flag and exist flag. Additionally, content of each byte in memory of created data fields can be specified. New created data field should always be the next available and size of data field should not exceed user-available memory. Also, for encoders with encoder type ID equal to FFh, extended type label can be specified in xlsx file (refer to Extended Type Label Specification in official HIPERFACE documentation). Additionally, serial number, firmware version and firmware date (ASCII coding) can be specified (refer to Read Version (56h) command in official HIPERFACE documentation).

Input

The HIPERFACE_ENCODER_out block has the following inputs:

Name	Unit	Description	Range
FPGASignalBus	Bus	Feedback signal bus of the HIPERFACE_ENCODER_in block	
Enable	[-]	Sensor signal enable	0 1
Encoder_Temp	[deg]	Encoder temperature	
Angle_Offset	[deg]	Offset angle for incremental encoder	0° ... 2^18 * 360°; resolution: 1.37E-3
Reset	[-]	Sensor reset signal (same functionality as Encoder Reset (53h) command)	0 1

Output

The processor out block has two outputs: Buffer_1 (values which are written to buffer) and StatusBus (important diagnostic signals). The output Buffer_1 contains encoder parameters specified in mask and block inputs, but also Slave response frame for encoder memory or encoder settings related protocol commands. The Slave response frame is located on the first 34 elements, while parameters are located on the last 8 elements (buffer depth is equal to 42). The output StatusBus contains important signals such as error diagnostics (protocol and data errors), simulated error state, access codes, address, UART settings, etc.

FPGA Main Component

Block

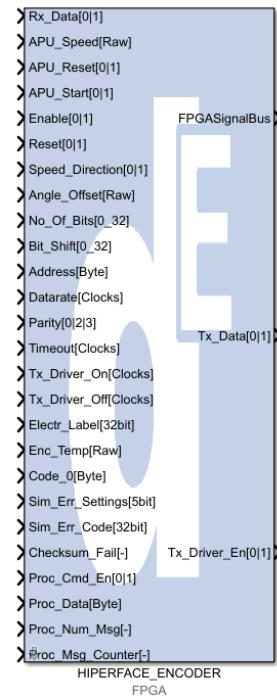


Figure 90: HIPERFACE_ENCODER FPGA Main block

Block Dialog

The FPGA main blockset contains the following dialog.

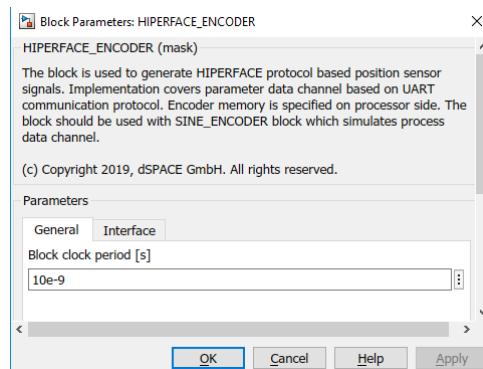


Figure 91: HIPERFACE_ENCODER FPGA Main block dialog

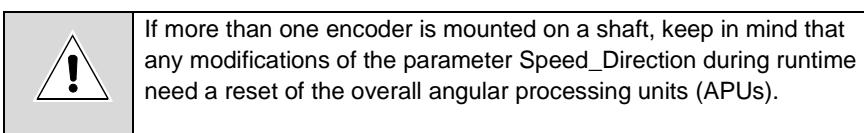
On the page Interface you can define the parameters of the input and output buffers for automatic interface generation. The dialog also contains a button to start interface generation. For more details about automatic interface generation, refer to the XSG Utils documentation chapter Automatic Interface Generation. The FPGA clock period can be configured too.

Input

The main block has the following inputs:

Name	Unit	Description	Format
Rx_Data	[0 1]	Rx of the data channel	Bool or UFix_1_0
APU_Speed	[Raw]	Actual speed	Fix_30_0
APU_Reset	[-]	Reset on high state	Bool
APU_Start	[-]	Start on high state	UFix_1_0
Signals which are connected to the HIPERFACE_ENCODER_in block to the FPGA			
Enable	[-]	Sensor signal enable	UFix_1_0
Reset	[-]	Sensor signal reset	UFix_1_0
Speed_Direction	[-]	Configure the speed direction (clockwise / counter-clockwise)	UFix_1_0
Angle_Offset	[Raw]	Offset angle for encoder 18 Bits Multiturn (35 – 18) 18 Bits Singleturn (17 – 0)	UFix_36_0
No_Of_Bits	[-]	Total number of bits (single turn + multi turn)	UFix_6_0
Bit_Shift	[-]	Number of bits by which absolute position is shifted to right	UFix_6_0
Address	[-]	Address byte of Slave response frame	UFix_8_0
Datarate	[Clocks]	Data rate of UART communication	UFix_19_0
Parity	[-]	Parity of UART communication	UFix_2_0
Timeout	[Clocks]	Timeout condition (end of Master request frame)	UFix_25_0
Tx_Driver_On	[Clocks]	Tx output driver enable delay	UFix_15_0
Tx_Driver_Off	[Clocks]	Tx output driver disable delay	UFix_13_0
Electr_Label	[-]	Electronic type label which contains 4 bytes (UART settings, encoder type ID, total EEPROM size and option code)	UFix_32_0
Enc_Temp	[Raw]	Encoder temperature which contains two 16-bit words (48h and F0h channel)	UFix_32_0
Code_0	[-]	Access Code 0 (used for protection of some HIPERFACE commands)	UFix_8_0
Sim_Err_Settings	[-]	Settings of error simulation (simulated state (2 bit), number of operational faults (3 bit))	UFix_5_0

Sim_Err_Code	[-]	Simulated error codes (up to 4 error codes)	UFix_32_0
Checksum_Fail	[-]	Specifies which checksum bit will be overwritten (8 bit) and forced checksum output (8 bit)	UFix_16_0
Proc_Cmd_En	[-]	Enable signal for Slave response frame from processor side	Bool
Proc_Data	[-]	Slave response frame from processor side	UFix_8_0
Proc_Num_Msg	[-]	Total number of bytes in Slave response frame from processor side	UFix_8_0
Proc_Msg_Counter	[-]	Counter of bytes in Slave response frame from processor side	UFix_8_0



Inside the main FPGA block, additional response delay for each HIPERFACE command can be specified by defining number of FPGA clocks in constant block (subsystem Response_Delay_Definition). In such a way, Slave response frame for specific commands can be additionally delayed by the defined number of FPGA clocks.

Output

The HIPERFACE_ENCODER main block has the following outputs:

Name	Unit	Description	Format
FPGASignalBus	Bus	Bus containing the block's most significant signals	-
TX_Data	[Bit]	Tx data of the data channel of the encoder	UFix_1_0
Tx_Output	[Bit]	Flag which controls the output driver of the Tx driver	Bool

Processor Input

Block

Master request frame (address, command identifier, data bytes and checksum) with additional information such as simulated error state status and detected errors in request frame (parity error, checksum error, wrong access code, wrong

command argument, unknown command error or wrong command length) is received through buffer (buffer depth is equal to 35). Relevant parts of received frame are extracted in HIPERFACE_ENCODER_in block. The extracted signals are merged into FPGASignalBus and routed to HIPERFACE_ENCODER_out block which handles received command. No specific settings are necessary.

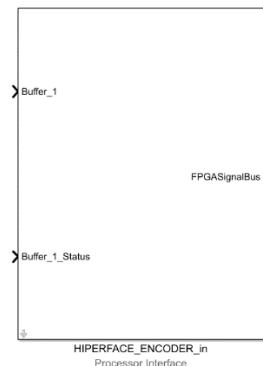


Figure 92: HIPERFACE_ENCODER_in block

Interface Examples

Implementation Example

To get a better overview of the necessary model structure for the Hiperface simulation please refer to the implementation example via the Matlab command:

`“xsg_electric_DEMO_installDemo('Template_ImpMDL\Hiperface');”`

Processor blocks

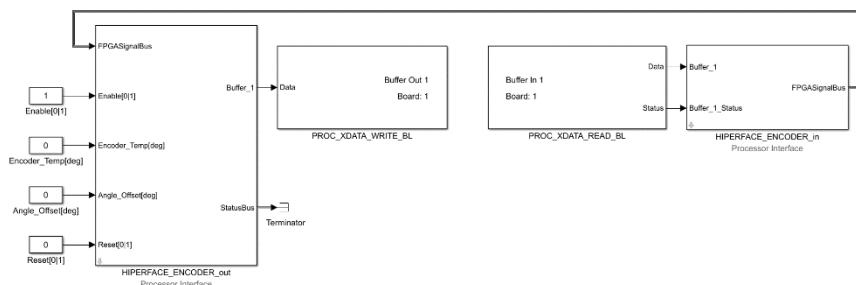


Figure 93: Processor interface

FPGA block

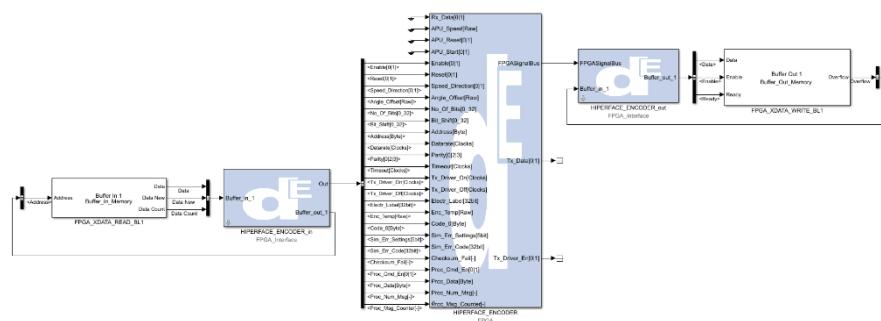


Figure 94: FPGA interface

Sensors: BiSS Encoder

Objective

Position sensors often use a Bidirectional Serial Synchronous (BiSS) interface for communication with the overall controller. This block simulates encoder with BiSS-C (Continuous) protocol. The functionality of the block covers point-to-point configuration scenario with the single data channel; Transmission of position sensor Single Cycle Data (SCD) and Control Data (CD) can be simulated. Initial content of encoder memory is specified in Excel file which is imported in the model. Aside from normal operation, CRC error states can be simulated by forcing CRC bits in SCD and register communication frame. The block also detects CRC errors in register communication (slave addressing, writing to slave registers and protocol commands). To realize a BiSS-C data frame the following blocks are located in the library.

 For detailed information about BiSS protocol refer to BiSS-Interface homepage ([Link](#)).

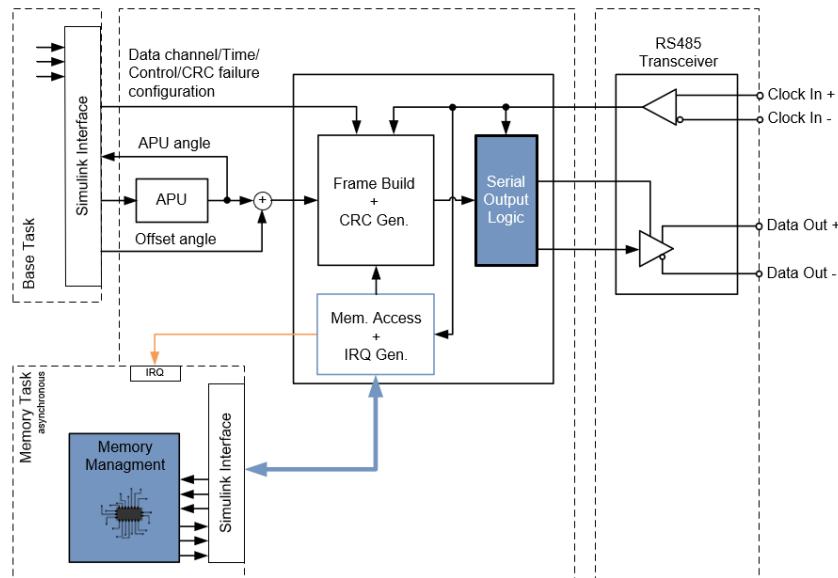


Figure 95: Implemented schematic of BiSS encoder

The blockset contains the following elements:

- Processor Interface: BISS_ENCODER_out (Processor Interface)
- FPGA Interface: BISS_ENCODER_in (FPGA Interface)
- FPGA: BISS_ENCODER (FPGA Main Component)

To ensure time requirements of memory access between the master and the encoder the memory management is realized in an asynchronous task. This guarantee a synchronous called task with respect to the memory access request from the connected master. Additionally, a frequently interrupt with respect to the main task can be also parameterized. This functionality is realized with the following elements:

- Processor Interface: BISS_ENCODER_MEMORY_out
(Processor Interface)
- FPGA Interface: BISS_ENCODER_MEMORY_in
(FPGA Interface)
- FPGA Interface: BISS_ENCODER_MEMORY_out
(FPGA Interface)
- Processor Interface: BISS_ENCODER_MEMORY_in
(Processor Interface)

	To get an easy startup with the overall model implementation, please refer to the implementation example. The model can be opened via double clicking on the “Open Implementation Example” or via the Matlab command: “xsg_electric_DEMO_installDemo('Template_ImpMDL\BiSS');”
	Please use a high priority for the memory access of the encoder.

Processor Output (Base Task)

Block

Merges the processor signals e.g. General Settings (SCD and register access control, BiSS timeout, etc.), Frame Settings (number of multi/single turn and diagnosis bits, resolution, CRC polynome, etc.) and CRC Error Settings signals (CRC failure type, forcing CRC bits, etc.) and writes them to the FPGA.

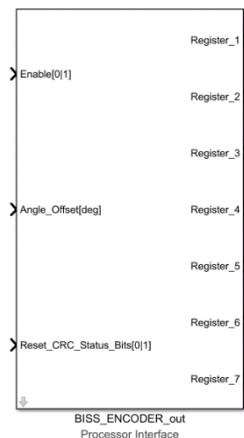


Figure 96: BISS_ENCODER_out block

Block Dialog

The processor output block set contains three dialogs. In the first one, the general parameters can be configured.

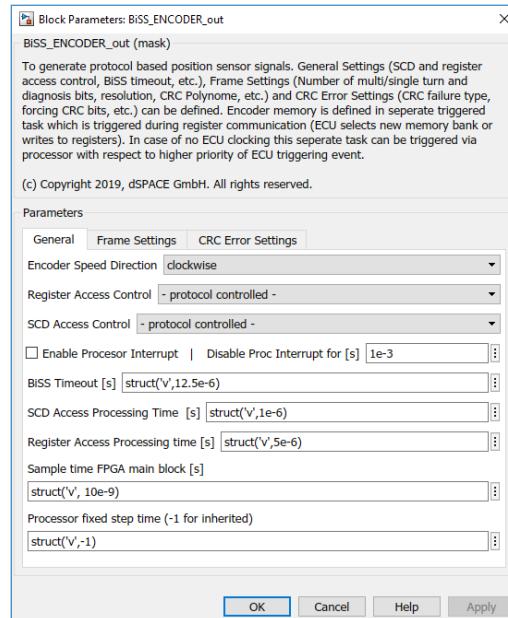


Figure 97: BISS_ENCODER_out dialog; page “General”

The first dialog has the following parameters:

Name	Unit	Description	Range
Encoder Speed Direction	[-]	Configure the direction of the input speed (clockwise / counter-clockwise)	0 1
Register Access Control	[-]	Configure register access state 0: protocol - controlled (Protocol commands broadcast CMD="01" and addressed CMD="01" are detected, which enable and disable register access respectively. Register access is enabled by default.) 1: forced enable 2: forced disable	0 1 2
SCD Access Control	[-]	Configure SCD access state 0: protocol - controlled (Protocol commands broadcast CMD="00" and addressed CMD="00" are detected, which de- and activate SCD in the data channel respectly. SCD access is enabled by default.) 1: forced enable 2: forced disable	0 1 2
Enable Proc Interrupt	[-]	Enables the interrupt generation synchronous to the base task	0 1
Disable Proc Interrupt for	[s]	If the timing requirements of the encoder – master communication is nearly the maximum performance, it can lead to processor overload and overruns. To minimize the overall processor load the spare time between the last and the next interrupt (which will be generated from the processor) can be parameterized	10e-6...1 resolution: Ts_FPGA

BiSS Timeout	[s]	Specified time period for detecting end of SCD frame	12.5e-6 ... 40e-6 resolution: Ts_FPGA
SCD Access Processing Time	[s]	Simulated processing time for SCD response; Defines time from the first rising edge of clock up to moment when sensor is ready to output the data	0 ... 40e-6 resolution: Ts_FPGA
Register Access Processing Time	[s]	Simulated processing time for register access; Defines time from the moment when slave has been addressed up to moment when slave is ready to output the register data	0 ... 20e-3 resolution: Ts_FPGA
Sample time FPGA	[s]	Sample time of the FPGA	-
Processor fixed step time	[s]	Step time of the processor task; a different sample time can be set in offline simulation to simulate the processor interface communication behavior	-

NOTE: Beside simulated processing time on the SCD and register access, there should be noted that position sensor model have real internal processing time which depends on used platform and simulated memory area. SCD access processing time is same for all platforms and it depends on used CRC polynome; maximum processing time for SCD access is 1.6 us. Register access processing time depends on used platform and simulated memory area. Note that overall processing time of position sensor model is over when real internal and simulated processing times are both over. This means that simulated processing time will have effect on slave response time only if it is greater then internal processing time of position sensor model!

On the second dialog frame settings can be configured.

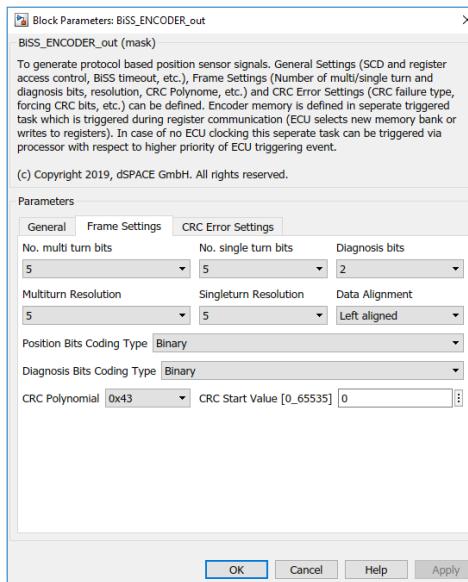


Figure 98: BISS_ENCODER_out dialog; page “Frame Settings”

The second dialog has the following parameters:

Name	Unit	Description	Range
No. Multi Turn Bits	[-]	Number of multi turn bits	0 ... 22
No. Single Turn Bits	[-]	Number of single turn bits	0 ... 34
No. Diagnosis Bits	[-]	Number of diagnosis bits	0 ... 8
Multi Turn Resolution	[-]	Multi turn resolution	0 ... 22
Single Turn Resolution	[-]	Single turn resolution	0 ... 34
Data Alignment	[-]	Specify type of data alignment in SCD frame: 0: left alignment (position bits in the most significant bits) 1: right alignment (diagnosis bits in the most significant bits)	0 1
Position Bits Coding Type	[-]	Specify coding type of position bits (binary / gray)	0 1
Diagnosis Bits Coding Type	[-]	Specify coding type of diagnosis bits (binary / gray)	0 1
CRC Polynomial	[-]	Specify CRC polynomial for SCD frame protection (0x00/0x25/0x43/0x190D9)	0 1 2 3

CRC Start Value	[-]	Specify CRC start value	0 ... 65535
-----------------	-------	-------------------------	-------------

In the third dialog CRC Error settings can be configured.

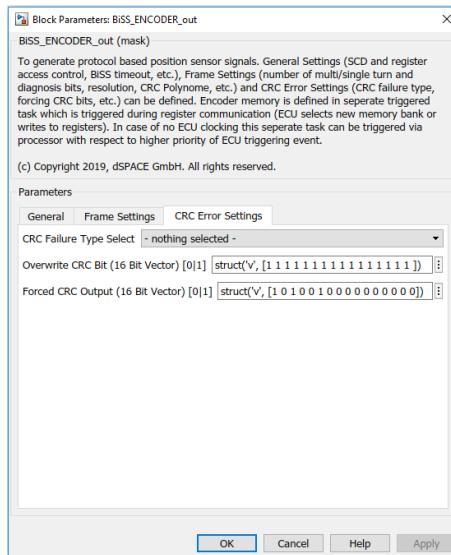


Figure 99: BISS_ENCODER_out dialog; page “CRC Error Settings”

The third dialog has the following parameters:

Name	Unit	Description	Range
CRC Failure Type Select	[-]	Specify type of CRC failure (nothing selected / SCD frame CRC / register data CRC)	0 1 2
Overwrite CRC Bit	[-]	Specify which CRC bits will be overwritten with the forced CRC Bit	16 Bit vector
Forced CRC Bit	[-]	Forced CRC Bits which will be used for output	16 Bit Vector

By choosing which CRC bits are overwritten and forcing desired CRC bits in the output frame, original CRC bits for slave response frame can be modified. If CRC failure type “SCD frame CRC” is selected, CRC bits at the end of SCD frame can be changed in order to simulate error (Number of modified CRC bits which is taken into account depends on the selected CRC polynome result length, starting from last significant CRC bit – 5 bits for 0x25, 6 for 0x43 and 16 bits for 0x190D9 CRC polynome). If CRC failure type “Register data CRC” is selected, CRC bits at the end of register communication frame can be changed in order to simulate error during slave register readout (Here only first 4 bits are taken into account, since used polynome for register communication protection is fixed - 0x13).

NOTE: CRC bits are transmitted inverted on the serial output line!

Input

The BISS_ENCODER_out block has the following inputs:

Name	Unit	Description	Range
Enable	[-]	Sensor signal enable	0 1
Angle_Offset	[deg]	Offset angle for incremental encoder	0° ... 2^18 * 360°; resolution: 1.37E-3
Diagnosis_Bit_1	[-]	Diagnosis bit 1	0 1
...			
Diagnosis_Bit_8	[-]	Diagnosis bit 8	0 1
Reset_CRC_Status_Bits[0 1]	[-]	CRC status bits reset	0 1

Diagnosis bits inputs can be used for controlling diagnosis bits which are selected to be part of SCD frame. Number of diagnosis bits taken into SCD frame depends on the parameter “No. Diagnosis Bits” value, starting from last significant diagnosis bit – Diagnosis bit 1.

Output

The processor out block provides multiplexed and non-multiplexed signals. Each register build-in is not part of the documentation.

Processor Output (Memory Access Task)

Block

Implementation of the encoder memory and logic. Encoder memory can be defined via parameter vector and write/read process is realized via buffer access to the FPGA.

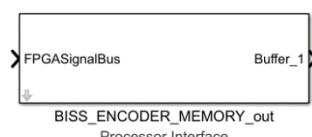


Figure 100: BISS_ENCODER_MEMORY_out block

Block Dialog

The processor output block set contains the following dialog.

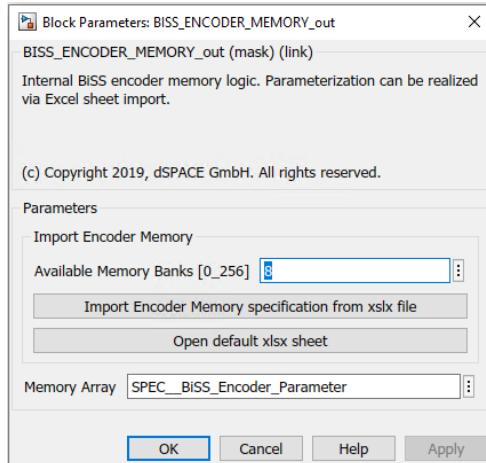


Figure 101: BISS_ENCODER_MEMORY_out dialog

The dialog blockset has the following parameters/buttons:

Name	Unit	Description	Range
Available Memory Banks	[-]	Number of User-available memory banks of the encoder (specified in encoder datasheet), which will be imported from xlsx file	0 ... 256
Button: Import Encoder Memory specification from xlsx file	[-]	The initial encoder memory specified in xlsx file can be imported and stored in array	
Open default xlsx sheet	[-]	Opens the default parameter sheet based on iC-MHM X5 encoder (manufacturer: iC-Haus)	
Memory Array	Array	Memory array of the encoder	

Excel file which contains initial encoder memory should be imported before model compilation. In the xlsx file data fields under “Encoder specification” columns can be defined and modified; Parameter values are entered under “Parameter [Hex]” column in hex format, while their description can be optionally added. Note that in direct access bank specific fields (EDS bank, Profile ID, etc.) only parameter values can be modified, while “Bank select” parameter value is not editable and it has always starting value of 0. Register Protection Level (RPL) column is fully editable, except for “Bank select” parameter, which RPL field must always contain value of 2. It is recommended that number of used memory banks (Defined with “Available Memory Banks” parameter) match the number of defined memory banks in xlsx file, since higher number of imported memory banks will increase register communication processing time of position sensor model!

NOTE: Memory defined in xlsx file is intended to be used for the simulation of register communication; Changes made under configuration parameter fields in xlsx file will not have effect on simulated BiSS position sensor model configuration!

Input

The BISS_ENCODER_MEMORY_out block has the following inputs:

Name	Unit	Description	Range
FPGASignalBus	Bus	Feedback signal bus of the BISS_ENCODER_MEMORY_out block	

Output

The processor out block realize the FPGA feedback via one specific buffer. Buffer contains encoder parameters for direct access and selected memory bank, which are packed together with RPL data (encoder parameters are packed on the first 32 registers, while RPL data is packed on the last 8 registers).

FPGA Main Component

Block

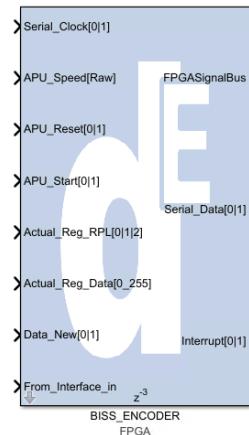


Figure 102: BISS_ENCODER FPGA Main block

Block Dialog

The FPGA main blockset contains the following dialog.

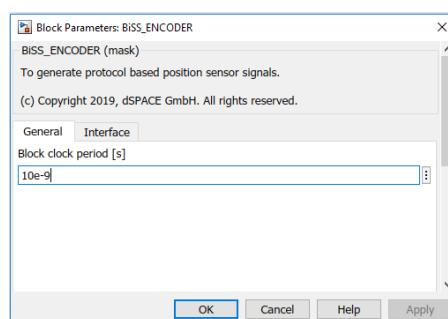


Figure 103: BISS_ENCODER FPGA Main block dialog

On the page Interface you can define the parameters of the input and output registers for automatic interface generation. The dialog also contains a button to start interface generation. For more details about automatic interface generation, refer to the XSG Utils documentation chapter Automatic Interface Generation. The FPGA clock period can be configured too.

Input

The main block has the following inputs:

Name	Unit	Description	Format
Serial Clock	[0 1]	Clock signal	Bool or UFix_1_0
APU_Speed	[Raw]	Actual speed	Fix_30_0
APU_Reset	[-]	Reset on high state	Bool
APU_Start	[-]	Start on high state	UFix_1_0
Signals which are connected to the BISS_ENCODER_MEMORY_in block to the FPGA			
Actual_Reg_RPL	[-]	RPL for selected register	UFix_2_0
Actual_Reg_Data	[-]	Data content for selected register	UFix_8_0
Data_New	[-]	New buffer data trigger	Bool
From_Interface_in	[Bus]	SignalBus which is connected to the BISS_ENCODER_in block	



If more than one encoder is mounted on a shaft, keep in mind that any modifications of the parameter Speed_Direction during runtime need a reset of the overall angular processing units (APUs).

Output

The BISS_ENCODER main block has the following outputs:

Name	Unit	Description	Format
FPGASingalBus	Bus	Bus containing the block's most significant signals	-
Serial_Data	[Bit]	Serial data of the encoder	UFix_1_0
Interrupt	[Bit]	Interrupt which controls the memory access task of the encoder; direct connected the interrupt block of rtifpga	UFix_1_0

Processor Input (Base Task)

Block

There is no processor input available for this blockset.

Processor Input (Memory Access Task)

Block

All necessary feedback for the memory logic (implemented in the BISS_ENCODER_MEMORY_out block) is realized via the BISS_ENCODER_MEMORY_in block. No specific settings are necessary.

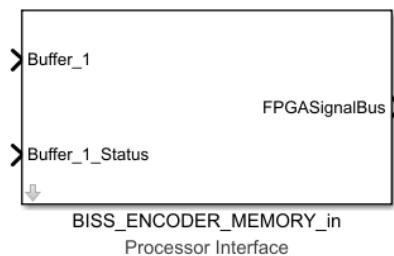


Figure 104: BISS_ENCODER_MEMORY_in block

Interface Examples

Implementation Example

To get a better overview of the necessary model structure for the BiSS simulation please refer to the implementation example via the Matlab command:

```
"xsg_electric_DEMO_installDemo('Template_ImpMDL\BiSS');"
```

Processor blocks

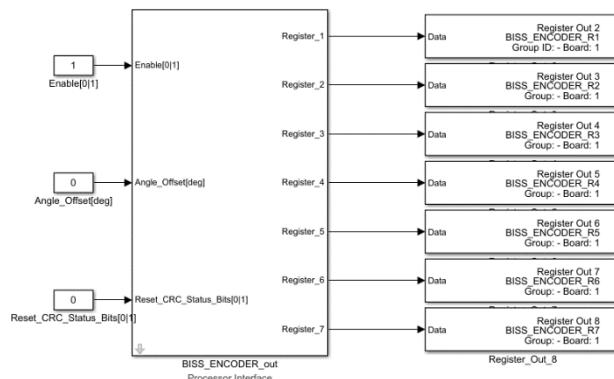


Figure 105: Processor (Base Task) interface

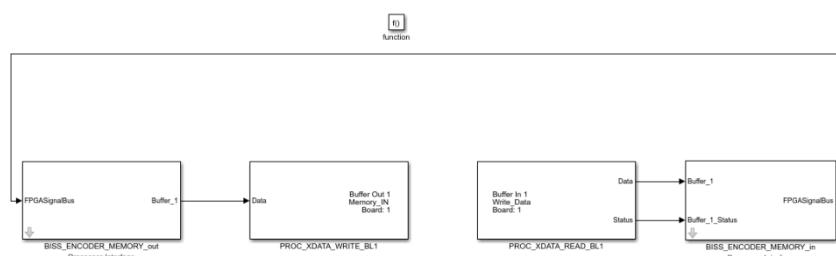
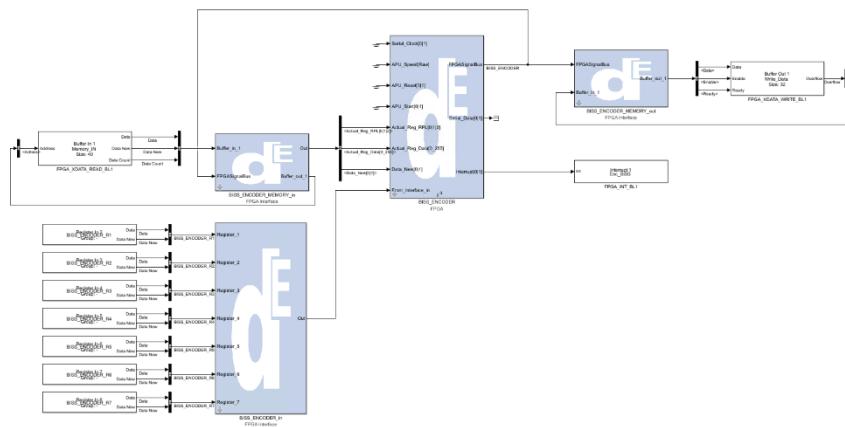


Figure 106: Processor (Memory Task) interface

FPGA block**Figure 107: FPGA interface**

Electric Machines: Brushless DC Motor (BLDC)

Objective

This block simulates a Brushless DC Machine. The Back EMF can be set with three tables individually. Therefor the machine is modeled inside the alpha beta-coordinates (also known as stator reference frame).

Next to the parameterization of the motor specific parameters, the motor can also be set into stimulus mode, too just set one of the phase currents to a specific value (to check if the current scaling fits to the ECU). The motor model also calculates the actual processor synchronous average values for the currents and the torque, where a downsampling might be useful to cover as well slow processor clock rates, but keep in mind that the resolution might decrease. An additional, reset functionality of the average function and the motor integrators is also implemented.

	If the standard interface of the XSG Electric Library is used, make sure that the BLDC_MACHINE_in and BLDC_MACHINE_out block of the processor interface is located in the same sampled task. This ensures that the timing behavior of the included average functionality of the FPGA main block is correct. Infractions of this requirement produce fail behavior in value conversion and calculation on the processor interface.
	If the machine configuration is set to delta, the maximum voltage input voltage of the motor is reduced to $1024V / \sqrt{3}$.

All motor parameters can be adjusted in the Processor Output block dialog. User-defined curves of the back EMF can be also parameterized as a function of the angle. To synchronize this table data with the actual motor angle, you can configure an additional angle offset.

The next figure shows the equivalent circuit diagram of the model in alpha- and beta- axis with arbitrarily back EMF.

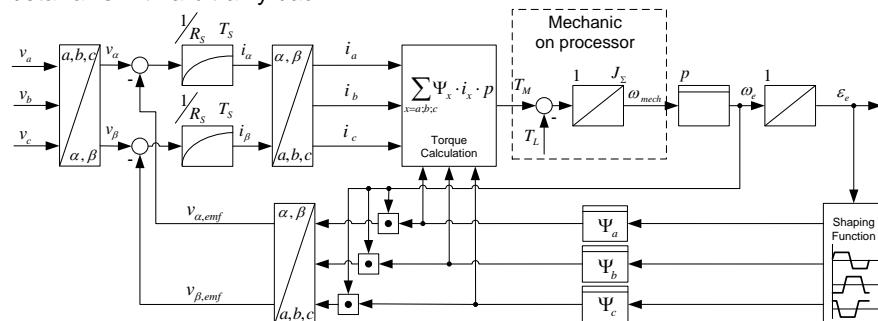


Figure 108: Circuit diagram of the BLDC motor model

The blockset contains the following elements:

- Processor Interface: BLDC_MACHINE_out (Processor Interface)
- FPGA Interface: BLDC_MACHINE_in (FPGA Interface)
- FPGA: BLDC_MACHINE (FPGA Main Component)

- FPGA Interface: BLDC_MACHINE_out (FPGA Interface)
- Processor Interface: BLDC_MACHINE_in (Processor Interface)

Processor Output

Block

Merges the processor signals and writes them to the FPGA.



Figure 109: BLDC_MACHINE_out block

Block Dialog

The processor output block contains several dialogs. All general parameters of the block can be configured in the page “General”.

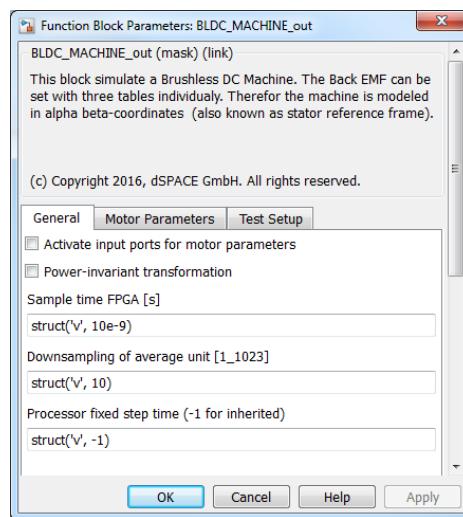


Figure 110: BLDC_MACHINE_out dialog; page “General”

The page has the following parameters:

Name	Unit	Description	Range
Activate input ports	[-]	Activate input ports for the relevant motor parameters	-
Power-invariant transformation	[-]	Activate power invariant transformation for clark transformation	-
Sample time FPGA	[s]	Sample time of the FPGA	-
Downsampling of average unit	[-]	Downsampling factor of the average functionality	1 ... 1023; resolution: 1
Processor fixed step time	[s]	Step time of the processor task; a different sample time can be set in offline simulation to simulate the processor interface communication behavior	-

All specific motor parameters can be configured in the page "Motor Parameters".

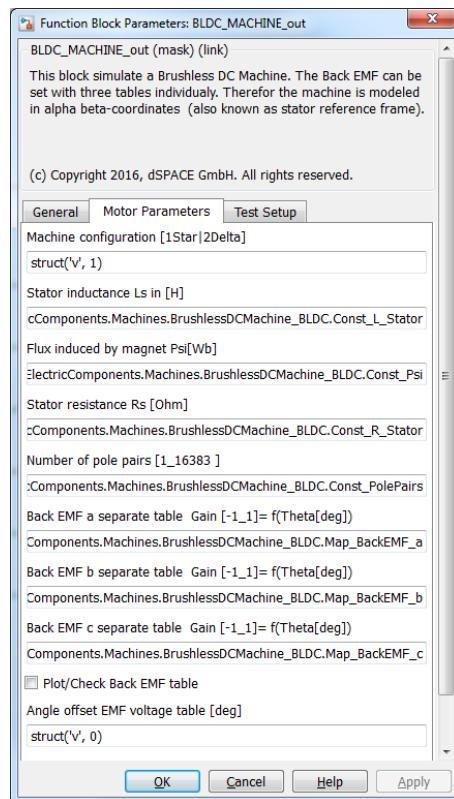


Figure 111: BLDC_MACHINE_out dialog, page "Motor Parameters"

The page has the following parameters:

Name	Unit	Description	Range
Machine configuration	[·]	Select the machine configuration; 1: star 2: delta	-
Stator inductance Ls	[H]	Stator inductance of the BLDC motor	-
Flux induced by magnet Psi	[Wb]	Magnetic flux of the motor magnets	0 ... 1 Wb; resolution: 15.2E-6
Stator resistance Rs	[Ohm]	Stator resistance in the range 0.00025...16 Ohm	244E-6 ... 16 Ohm; resolution: 244E-6
Number of pole pairs	[·]	Number of motor pole pairs	0 ... 16383; resolution: 1
Back EMF a separate table	[·]	Represents the back EMF of phase a with a separate gain table related to stator a The table data must have a structure of var.x for the angle vector and var.v for the back EMF factor. The angle must be set from 0 to 360°	-1 ... 1; resolution: 59.6E-9
Back EMF b separate table	[·]	Represents the back EMF of phase a with a separate gain table related to stator b. The table data must have a structure of var.x for the angle vector and var.v for the back EMF factor. The angle must be set from 0 to 360°	-1 ... 1; resolution: 59.6E-9
Back EMF c separate table	[·]	Represents the back EMF of phase c with a separate gain table related to stator c The table data must have a structure of var.x for the angle vector and var.v for the back emf factor. The angle must be set from 0 to 360°	-1 ... 1; resolution: 59.6E-9
Plot / Check Back EMF table	[·]	Let the user plot the actual display of the back EMF table	[·]

Angle offset EMF voltage table	[deg]	Offset angle between the rotor angle and the back EMF table data	0° ... 360°; resolution: 1.37E-3
OperationMode	[-]	Selection of the Operation Mode of the motor model; 1: Open Loop test 0: Closed Loop test	0 1
Open Loop Current Selector	[-]	Selector which current is stimulated if the operation mode is set to open loop test; 1: i_a 2: i_b 3: i_c	-
Open Loop Current	[A]	Value for the stimulated open loop current	A

Specific settings for testing the outputs of the machine can be configured on the page “Test Setup”.

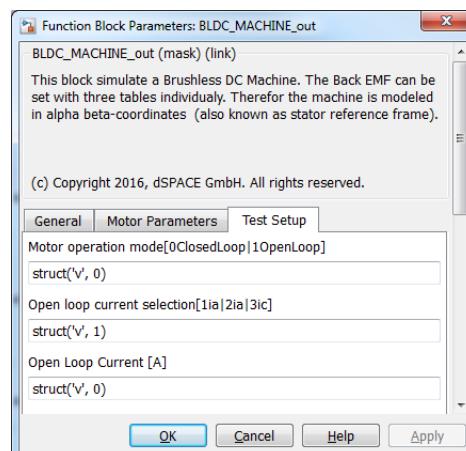


Figure 112: BLDC_MACHINE_out dialog; page “Test Setup”

The page has the following parameters:

Name	Unit	Description	Range
OperationMode	[-]	Selection of the Operation Mode of the motor model; 1: Open Loop test 0: Closed Loop test	0 1
Open Loop Current Selector	[-]	Selector which current is stimulated if the operation mode is set to open loop test; 1: i_a 2: i_b 3: i_c	-
Open Loop Current	[A]	Value for the stimulated open loop current	A

Input

The BLDC_MACHINE_out block has the following inputs as listed below. If input ports for the motor parameters are activated via the GUI checkbox, additional ports for the relevant motor parameters are added.

Name	Unit	Description	Range
FPGASignalBus	Bus	Feedback bus of the BLDC Machine. Connect this input with the associated output of the processor input block BLDC_MACHINE_in	-
Reset_Average	[-]	Reset function of the implemented FPGA average functionality	0 1
Reset_Motor MDL	[-]	Reset function of the motor model on FPGA	0 1

Output

The Processor Out block provides the following input registers whose sectioning is shown below:

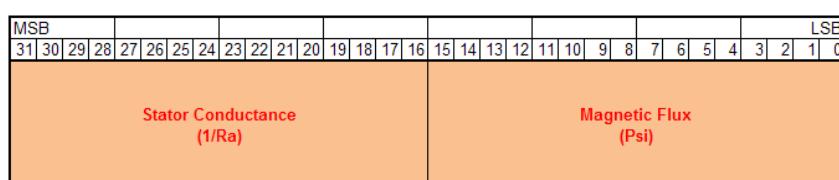
Register 1

Figure 113: BLDC_MACHINE_out Register 1

Name	Used Bits	Description	Range
Magnetic Flux Psi	15 ... 0	Magnetic flux induced by the magnet	0 ... 2^16-1
Stator Conductance 1/Ra	31 ... 16	Stator conductance of a phase in [S]	0 ... 2^16-1 represents 16 ... 250E-6 Ohm

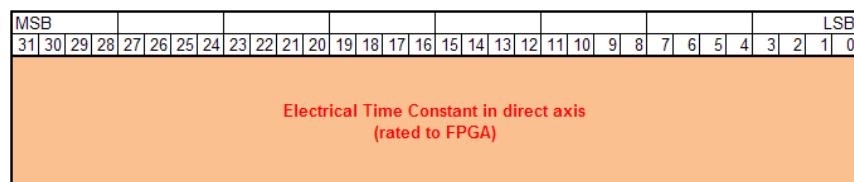
Register 2

Figure 114: BLDC_MACHINE_out Register 2

Name	Used Bits	Description	Range
Electrical time constant in direct axis	31 ... 0	Electrical motor time constant in direct axis with reference to the FPGA sample time ($T = T_s_{FPGA} * 10^6 L/R$)	0 ... 2^32-1 represents 0 ... 1; resolution: 232E-12

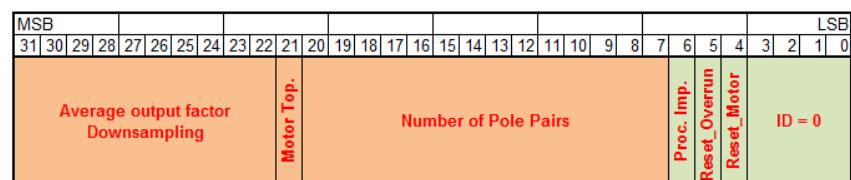
Register 3; ID = 0

Figure 115: BLDC_MACHINE_out Register 3; ID = 0

Name	Used Bits	Description	Range
Counter ID	3 ... 0	ID to select which register coding is activated via Processor	0 ... 1
Reset_Motor MDL	4	Reset flag for the motor model on the FPGA	0 1
Reset_OVERRUN	5	Reset flag to reset an overrun of the average functionality	0 1
Proc. Sync. Impulse	6	Processor synchronous toggle bit	0 1
Number of Pole Pairs	13 ... 0	Number of pole pairs	1 ... 16363
Motor Topology	30	Specify the motor topology	0 1
Average output factor Down-sampling	9 ... 0	Downsampling factor for average functionality of the outputs to the processor interface	1 ... 1023

Register 4; ID = 1

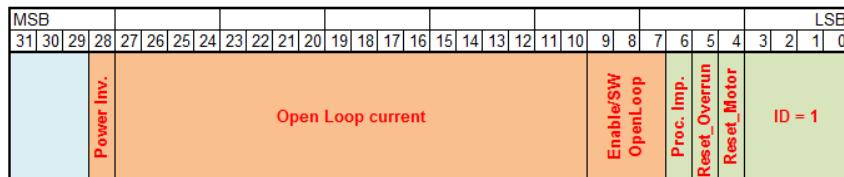


Figure 116: BLDC_MACHINE_out Register 3; ID = 1

Name	Used Bits	Description	Range
Continuous Data	6 ... 0	Definition same as ID = 0	
Enable/SW Open Loop	9 ... 7	Switch and Enable of the open loop stimulation	0 ... 3
OpenLoop Current	27 ... 10	Open Loop current for open loop stimulation	-1024 ... 1024
Power Invariant	28	Switch to activate power invariant transformation	0 1
SPARE	31 ... 29		

Register 4

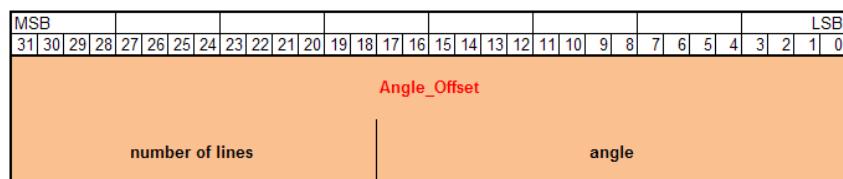


Figure 117: BLDC_MACHINE_out Register 4

Name	Used Bits	Description	Range
Angle_Offset	31 ... 0	The offset angle represents the mechanical angle of the rotator, scaled to the number of pole pairs	0...2^18-1 ≈ 360° electrical angle

Register 5

Figure 118: BLDC_MACHINE_out Register 5

Name	Used Bits	Description	Range
LUT Table Data A	23 ... 0	LUT Table Data A: Represents the back EMF with a separate gain table related to stator a	0 ... 2^23-1 Represents 0...2^23 ≈ 0...1 2^23+1...2^24 ≈ -1...0 (two's complement)
Actual LUT Write Address	31 ... 24	Actual LUT Write Address (Bit 0 – 7)	0 ... 2^8-1

Register 6

Figure 119: BLDC_MACHINE_out Register 6

Name	Used Bits	Description	Range
LUT Table Data B	23 ... 0	LUT Table Data B: Represents the back EMF with a separate gain table related to stator b	0 ... 2^23-1 Represents 0...2^23 $\hat{=}$ 0...1 $2^{23+1}...2^{24} \hat{=}$ - 1...0 (two's complement)
Actual LUT Write Address	31 ... 24	Actual LUT Write Address (Bit 8 – 15)	0 ... 2^8-1

Register 7

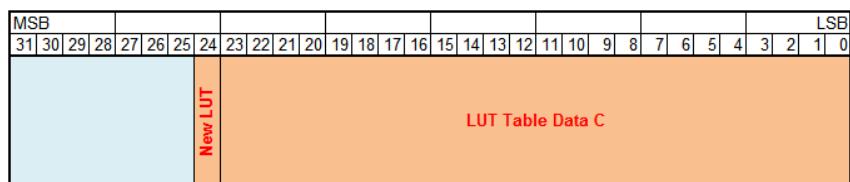


Figure 120: BLDC_MACHINE_out Register 7

Name	Used Bits	Description	Range
LUT Table Data C	23 ... 0	LUT Table Data C: Represents the back EMF with a separate gain table related to stator c	0 ... 2^23-1 Represents 0...2^23 $\hat{=}$ 0...1 $2^{23+1}...2^{24} \hat{=}$ - 1...0 (two's complement)
New LUT	24	Flag which detects new data for lookup table	
SPARE	31 ... 25		

FPGA Main Component

Block



Figure 121: BLDC_MACHINE FPGA Main block

Block Dialog

The FPGA main blockset contains the following dialog.

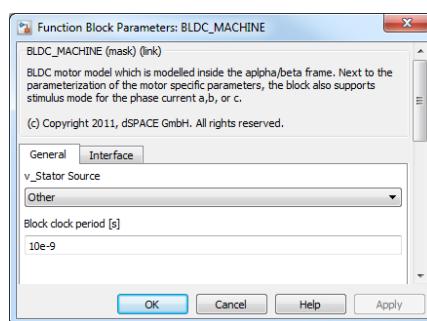


Figure 122: BLDC_MACHINE FPGA Main block dialog

On the page General the connected stator voltage can be selected. Please choose the connected inverter model via the popup.

On the page Interface you can define the parameters of the input and output registers for automatic interface generation. The dialog also contains a button to start interface generation. For more details about automatic interface generation,

refer to the XSG Utils documentation chapter Automatic Interface Generation. The FPGA clock period can be configured too.

Input

The main block has the following inputs:

Name	Unit	Description	Format
v_Stator	[V] (vector)	Stator voltage vector of the motor phase	Fix_16_7
v_floating	[-] (vector)	Floating detection vector of the motor phase	UFix_1_0
APU_Speed	[Raw]	Actual speed	Fix_30_0
APU_Reset	[-]	Reset on high state	Bool
APU_Start	[-]	Start on high state	UFix_1_0
Angle_Offset	[Raw]	Offset angle between the rotor angle and the back EMF table data	UFix_32_0
Motor_Topology	[-]	Specify the motor topology	UFix_1_0
Power_Invariant	[-]	Specify the internal clark transformation	UFix_1_0
Psi	[Wb]	Magnetic flux of the motor magnets	UFix_16_16
R_inv	[S]	Stator conductance	UFix_16_4
PolePairs	[-]	Pole pairs of the resolver	UFix_14_0
Ts/(L/R)	[-]	Electrical motor time constant with reference to the FPGA sample time	UFix_32_32
Reset_OVERRUN	[-]	Reset flag to reset an overrun of the average functionality	Bool
Downsampling	[-]	Downsampling factor for average functionality of the outputs to the processor interface	UFix_10_0
1 Max_Phase_Current	[1/A]	Maximal phase current of one motor phase (prepared for external scaling in next releases)	UFix_14_14
Table_Data_A	[-]	LUT Table Data A: Represents the back EMF with a separate gain table related to stator a	UFix_24_0
Table_Data_B	[-]	LUT Table Data B: Represents the back EMF with a separate gain table related to stator b	UFix_24_0
Table_Data_C	[-]	LUT Table Data C: Represents the back EMF with a separate gain table related to stator c	UFix_24_0

New_LUT_Data	[-]	Table Data flag; Flag is set to one if new table date is sent over the Table_Data inputs. Each value of the table data is stored synchronously with the processor toggle bit.	UFix_1_0
Address	[Raw]	LUT table address of the stored table data	UFix_16_0
Proc_Sync_Imp	[-]	Processor synchronize toggle bit	UFix_1_0
SW_Select_Open_Loop	[-]	Switch and Selector for the Open Loop Control	UFix_3_0
Open_Loop_Current	[A]	Open Loop current	UFix_18_7
Reset_Motor_MDL	[-]	Reset flag to the motor model	Bool

Output

The BLDC_MACHINE main block has the following outputs:

Name	Unit	Description	Format
FPGASignalBus	Bus	Bus containing the block's most significant signals	-
i_a	[A]	Current in phase a	Fix_18_7
i_b	[A]	Current in phase b	Fix_18_7
i_c	[A]	Current in phase c	Fix_18_7
i_stator_alpha	[A]	Current in the α axis	Fix_18_7
i_stator_beta	[A]	Current in the β axis	Fix_18_7
Trq	[Nm]	Torque of the motor	Fix_25_14
v_emf	[V](Bus)	Back EMF voltage	Fix_18_7
v_Feedback	[V](Bus)	Feedback voltage to the inverter model	Fix_18_7
Shape	[-]Bus	Actual back EMF factors of the table data	-

Processor Input

Block

Adapts the FPGA signals for the processor side

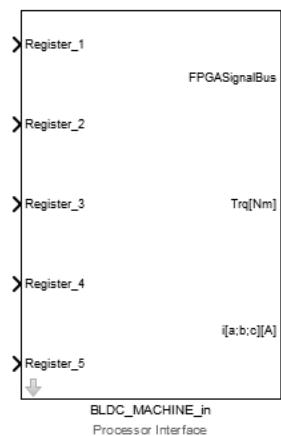


Figure 123: BLDC_MOTOR_in block

Block Dialog

The dialog only provides a short block description.

Input

The processor input block has the following inputs:

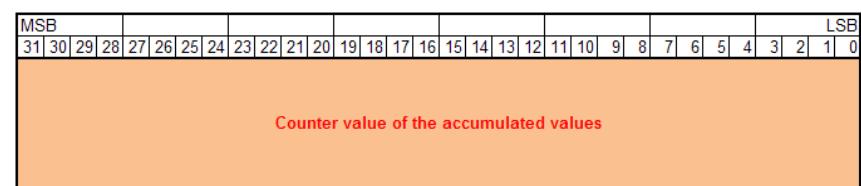
Register 1

Figure 124: BLDC_MOTOR_in Register 1

Name	Used Bits	Description	Range
Accu	31 ... 0	Counter value of the accumulated values; if an overrun is detected the value $2^{32}-1$ will be sent	0 ... $2^{32}-1$

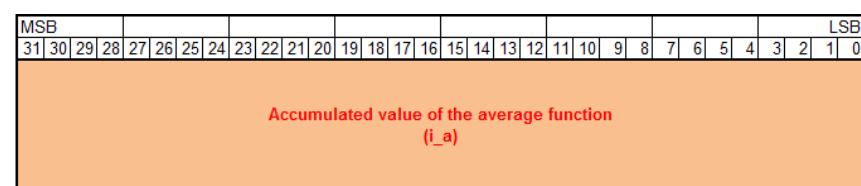
Register 2

Figure 125: BLDC_MOTOR_in Register 2

Name	Used Bits	Description	Range
Accu	31 ... 0	Accumulated value of the average function; if an overrun is detected the value 2^31-1 will be sent Signal: i_a	0 ... 2^32-1

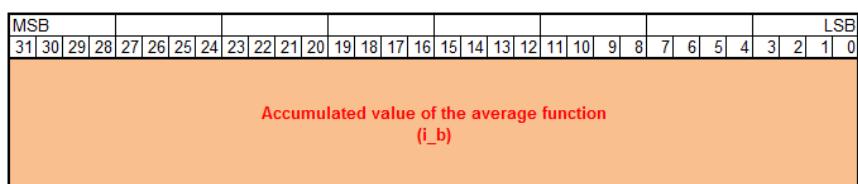
Register 3

Figure 126: BLDC_MOTOR_in Register 3

Name	Used Bits	Description	Range
Accu	31 ... 0	Accumulated value of the average function; if an overrun is detected the value 2^31-1 will be sent Signal: i_b	0 ... 2^32-1

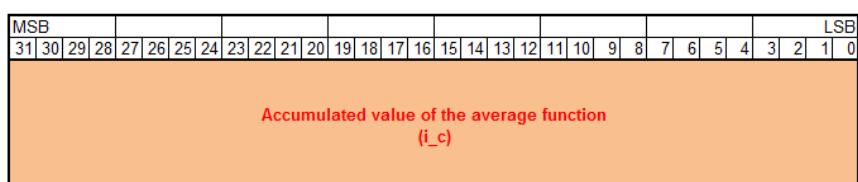
Register 4

Figure 127: BLDC_MOTOR_in Register 4

Name	Used Bits	Description	Range
Accu counter	31 ... 0	Accumulated value of the average function; if an overrun is detected the value 2^31-1 will be sent Signal: i_c	0 ... 2^32-1

Register 5

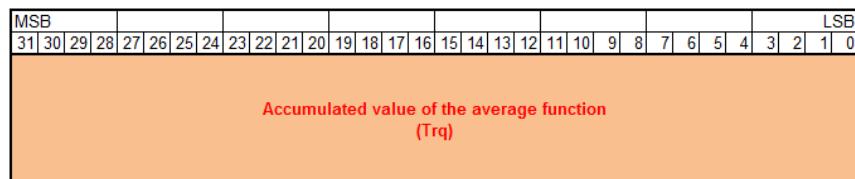


Figure 128: BLDC_MOTOR_in Register 5

Name	Used Bits	Description	Range
Accu	31 ... 0	Accumulated value of the average function; if an overrun is detected the value 2^31-1 will be sent Signal: Trq	0 ... 2^32-1

Output

The processor input block has the following outputs:

Name	Unit	Description	Range
FPGASign alBus	Bus	Feedback Bus of the BLDC Motor. Connect this output with the associated input of the processor input block BLDC_MACHINE_out	-
Trq	[Nm]	Torque of the motor	-1024 ... 1024.99 Nm
i	[A]Bus	Phase current in each motor phase (a,b,c)	-1024 ... 1023.99 A
Average_ Overrun_ Bus	[-]Bus	Overrun feedback from the average function	0 1

Block additional feedback

Adapts the FPGA signals for additional feedback of the FPGA to the processor side (like power consumption,...).



Figure 129: BLDC_MOTOR_additional_in block

Block Dialog

The dialog only provides a short block description.

Input

The processor input block has the following inputs:

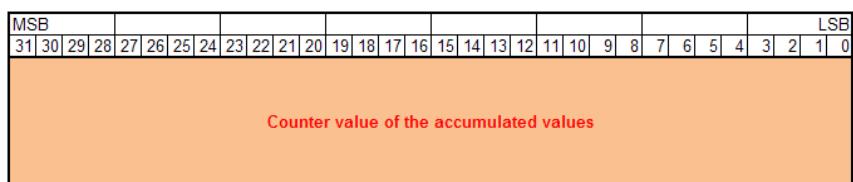
Register 1

Figure 130: BLDC_MOTOR_additional_in Register 1

Name	Used Bits	Description	Range
Accu	31 ... 0	Counter value of the accumulated values; if an overrun is detected the value $2^{32}-1$ will be sent	0 ... $2^{32}-1$

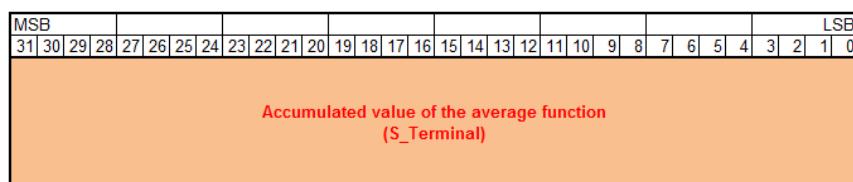
Register 2

Figure 131: BLDC_MOTOR_additional_in Register 2

Name	Used Bits	Description	Range
Accu	31 ... 0	Accumulated value of the average function; if an overrun is detected the value $2^{31}-1$ will be sent Signal: input power consumption S_Terminal [VA]	0 ... $2^{32}-1$

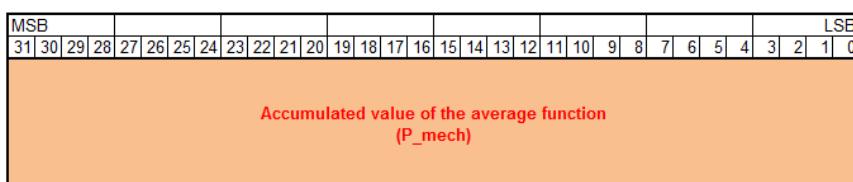
Register 3

Figure 132: BLDC_MOTOR_additional_in Register 3

Name	Used Bits	Description	Range
Accu	31 ... 0	Accumulated value of the average function; if an overrun is detected the value 2^31-1 will be sent Signal: mechanical output power P_mech [W]	0 ... 2^32-1

Output

The processor input block has the following outputs:

Name	Unit	Description	Range
FPGASignalBus	Bus	Bus containing the most significant signals of this block	-

Interface Examples

Processor blocks

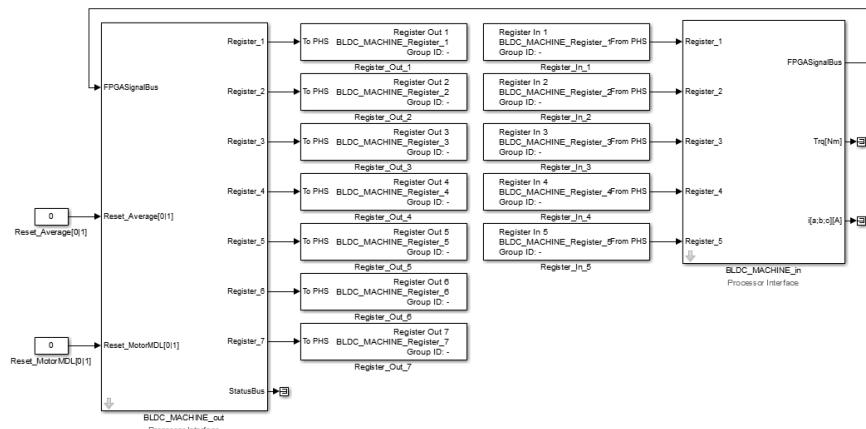


Figure 133: Processor interface

FPGA block

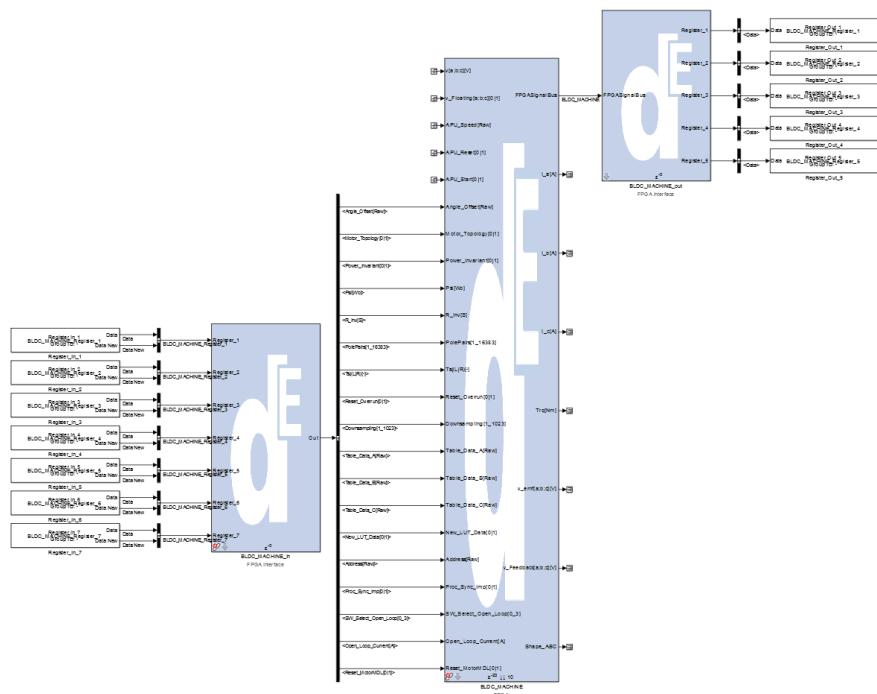


Figure 134: FPGA interface

Electric Machines: Permanent Magnetized Synchronous Machine (PMSM)

Objective

In this model a permanent magnet synchronous machine is modeled. The generated back EMF from the PMSM has sinusoidal shape, so that the machine is modeled in dq-coordinates (also known as rotor reference frame). All motor parameters are adjustable from the processor during runtime with respect to the sampling rate of the processor.

Next to the parameterization of the motor specific parameters, the motor can also be set into stimulus mode, too just set one of the phase currents (a, b, c) or one of the transformed currents (d, q) to a specific value (to check if the current scaling fits to the ECU). The motor model also calculates the actual processor synchronous average values for the currents and the torque, where a downsampling might be useful to cover as well slow processor clock rates, but keep in mind that the resolution might decrease. An additional, reset functionality of the average function and the motor integrators is also implemented.

	If the standard interface of the XSG Electric Library is used, make sure that the PMSM_MACHINE_in and PMSM_MACHINE_out block of the processor interface is located in the same sampled task. This ensures that the timing behavior of the included average functionality of the FPGA main block is correct. Infractions of this requirement produce fail behavior in value conversion and calculation on the processor interface!
	If the machine configuration is set to delta, the maximum voltage input voltage of the motor is reduced to $1024V / \sqrt{3}$.

All motor parameters can be adjusted (during runtime) in the processor output block dialog. The next figure shows the equivalent circuit diagram of the model in d- and q- axis.

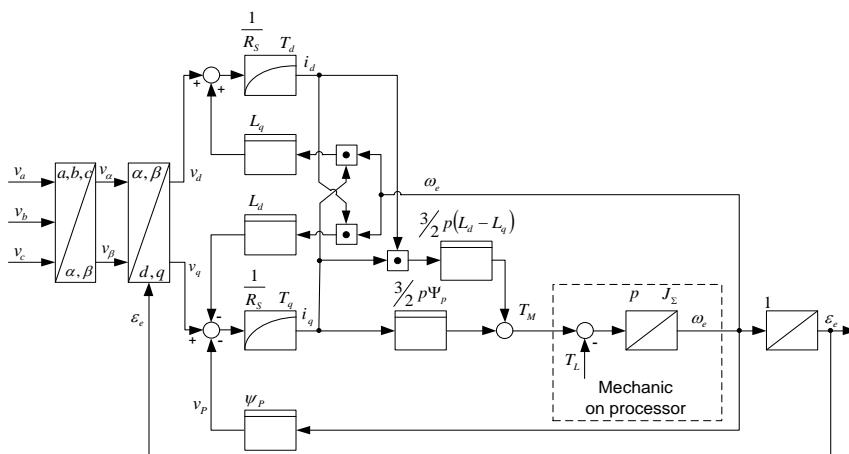


Figure 135: Circuit diagram of the PMSM motor model

The PMSM is modeled using dq notation in the synchronously rotating reference frame. The following equation describes the motor mathematical model.

$$\begin{aligned} L_d \frac{di_d}{dt} &= v_d - R_s i_d + L_q \omega_e i_q \\ L_q \frac{di_q}{dt} &= v_q - R_s i_q - L_d \omega_e i_d - \psi_p \omega_e \\ Trq &= \frac{3}{2} p [\psi_p i_q + (L_d - L_q) i_d i_q] \end{aligned}$$

L_d	Direct axis inductance
L_q	Quadrature axis inductance
R_s	Stator winding resistance
ψ_p	Flux induced by the magnet
p	Number of pole pairs

The blockset contains the following elements:

- Processor Interface: PMSM_MACHINE_out (Processor Interface)
- FPGA Interface: PMSM_MACHINE_in (FPGA Interface)
- FPGA: PMSM_MACHINE (FPGA Main Component)
- FPGA Interface: PMSM_MACHINE_out (FPGA Interface)
- Processor Interface: PMSM_MACHINE_in (Processor Interface)

Processor Output

Block

Merges the processor signals and writes them to the FPGA

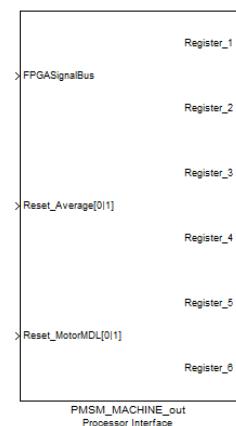


Figure 136: PMSM_MACHINE_out block

Block Dialog

The processor output block contains several dialogs. All general parameters of the block can be configured in the page “General”.

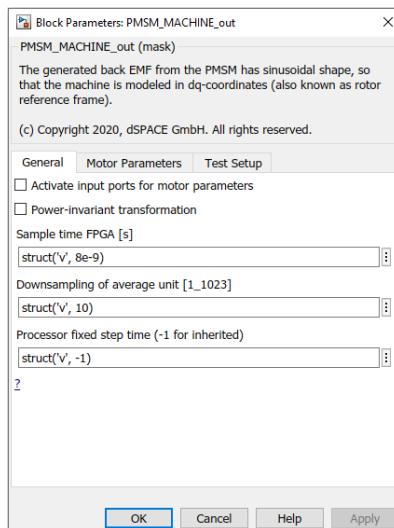


Figure 137: PMSM_MACHINE_out dialog; page “General”

The page has the following parameters:

Name	Unit	Description	Range
Activate input ports	[-]	Activate input ports for the relevant motor parameters	-
Power-invariant transformation	[-]	Activate power invariant transformation for clark transformation	-
Sample time FPGA	[s]	Sample time of the FPGA	-
Downsampling of average unit	[-]	Downsampling factor of the average functionality	1 ... 1023; resolution: 1
Processor fixed step time	[s]	Step time of the processor task; a different sample time can be set in offline simulation to simulate the processor interface communication behavior	-

All specific motor parameters can be configured in the page “Motor Parameters”.

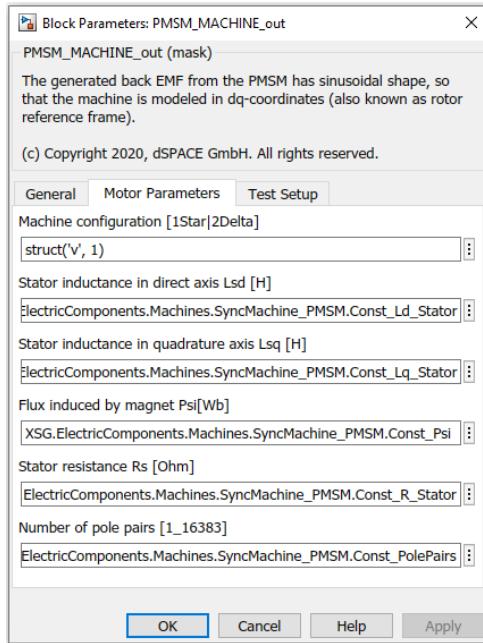


Figure 138: PMSM_MACHINE_out dialog; page “Motor Parameters”

The page has the following parameters:

Name	Unit	Description	Range
Machine configuration	[-]	Select the machine configuration; 1: star 2: delta	-
Stator inductance in direct axis Lsd	[H]	Stator inductance in the direct axis	0 ... 0.25 H; resolution: 3.81E-6
Stator inductance in quadrature axis Lsq	[H]	Stator inductance in the quadrature axis	0 ... 0.25 H; resolution: 3.81E-6
Flux induced by magnet Psi	[Wb]	Magnetic flux of the motor magnets	0 ... 1 Wb; resolution: 15.2E-6
Stator resistance Rs	[Ohm]	Stator resistance	244E-6 ... 16 Ohm; resolution: 244E-6
Number of pole pairs	[-]	Number of motor pole pairs	0 ... 16383; resolution: 1

Specific settings for testing the outputs of the machine can be configured on the page “Test Setup”.

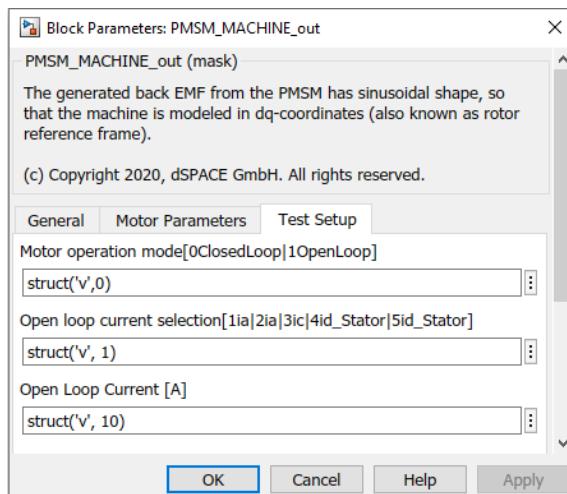


Figure 139: PMSM_MACHINE_out dialog; page “Test Setup”

The page has the following parameters:

Name	Unit	Description	Range
OperationMode	[-]	Selection of the Operation Mode of the motor model; 1: Open Loop test 0: Closed Loop test	0 1
Open Loop Current Selector	[-]	Selector which current is stimulated if the operation mode is set to open loop test; 1: i_a 2: i_b 3: i_c 4: i_d_Stator 5: i_q_Stator	-
Open Loop Current	[A]	Value for the stimulated open loop current	A

Input

The PMSM_MACHINE_out block has the following inputs. If input ports for the motor parameters are activated via the GUI checkbox, additional ports for the relevant motor parameters are added.

Name	Unit	Description	Range
FPGASignalBus	Bus	Bus containing the most significant feedback signals from the processor input block	-
Reset_Average	[-]	Reset function of the implemented FPGA average functionality	0 1
Reset_Motor MDL	[-]	Reset function of the motor model on FPGA	0 1

FPGA Main Component

Block

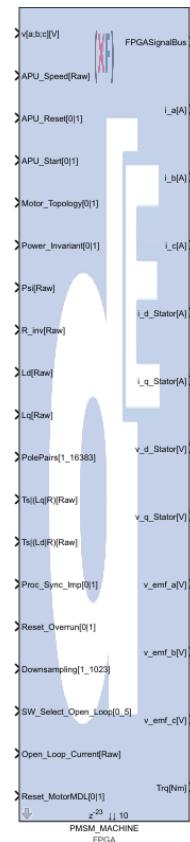


Figure 140: PMSM_MACHINE FPGA Main block for fixed point

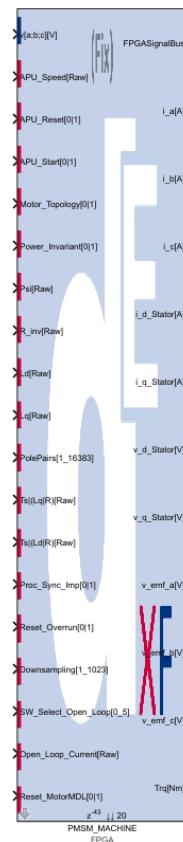


Figure 141: PMSM_MACHINE FPGA Main block for floating point

Block Dialog

The FPGA main blockset contains the following dialog.

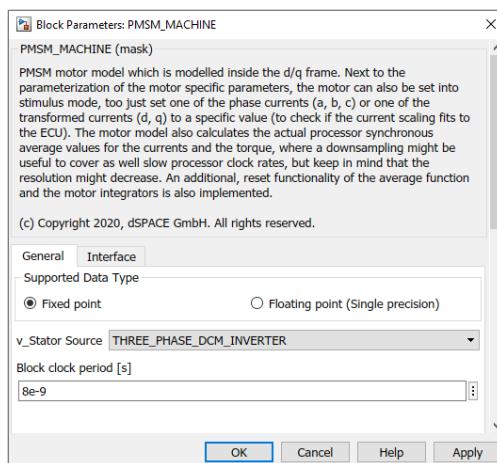


Figure 142: PMSM_MACHINE FPGA Main block dialog

On the page General the connected stator voltage can be selected. Please choose the connected inverter model via the popup. In addition, the input datatype can be selected as well (Fixed/Floating point).

On the page Interface you can define the parameters of the input and output registers for automatic interface generation. The dialog also contains a button to start interface generation. For more details about automatic interface generation, refer to the XSG Utils documentation chapter Automatic Interface Generation. The FPGA clock period can be configured too.

Input

The main block has the following inputs:

Name	Unit	Description	Format
v	[V] (vector)	Stator voltage vector of the motor phase	User defined
APU_Speed	[Raw]	Actual speed	Fix_30_0
APU_Reset	[‐]	Reset on high state	Bool
APU_Start	[‐]	Start on high state	UFix_1_0
Motor_Topology	[‐]	Specify the motor topology	UFix_1_0
Power_Invariant	[‐]	Specify the internal clark transformation	UFix_1_0
Psi	[Raw]	Magnetic flux of the motor magnets	UFix_31_0
R_inv	[Raw]	Stator conductance	UFix_31_0
Ld	[Raw]	Stator inductance in the direct axis	UFix_32_0
Lq	[Raw]	Stator inductance in the quadrate axis	UFix_32_0
PolePairs	[‐]	Pole pairs of the resolver	UFix_14_0
Ts/(Ld/R)	[Raw]	Electrical motor time constant in direct axis with reference to the FPGA sample time	UFix_32_0
Ts/(Lq/R)	[Raw]	Electrical motor time constant in quadrate axis with reference to the FPGA sample time	UFix_32_0
Proc_Sync_Imp	[‐]	Processor synchronize toggle bit	UFix_1_0
Reset_OVERRUN	[‐]	Reset flag to reset an overrun of the average functionality	Bool
Downsampling	[‐]	Downsampling factor for average functionality of the outputs to the processor interface	UFix_10_0
1 Max_Phase_Current	[1/A]	Maximal phase current of one motor phase (prepared for external scaling in next releases)	UFix_14_14
SW_Select_Open_Loop	[‐]	Switch and Selector for the Open Loop Control	UFix_4_0
Open_Loop_Current	[Raw]	Open Loop current	UFix_32_0
Reset_Motor_MDL	[‐]	Reset flag to the motor model	Bool

Output

The PMSM_MACHINE main block has the following outputs with a maximal latency of thirteen:

Name	Unit	Description	Format
FPGASignalBus	Bus	Bus containing the most significant signals of this block	-
i_a	[A]	Current in phase a	User defined
i_b	[A]	Current in phase b	User defined
i_c	[A]	Current in phase c	User defined
i_d_Stator	[A]	Current in the direct axis	User defined
i_q_Stator	[A]	Current in the quadrature axis	User defined
v_d_Stator	[V]	Voltage in direct axis	User defined
v_q_Stator	[V]	Voltage in quadrature axis	User defined
v_emf_a	[V]	Back EMF Voltage in phase a	User defined
v_emf_b	[V]	Back EMF Voltage in phase b	User defined
v_emf_c	[V]	Back EMF Voltage in phase c	User defined
Trq	[Nm]	Torque of the motor	User defined

Processor Input

Block

Adapts the FPGA signals for the processor side.

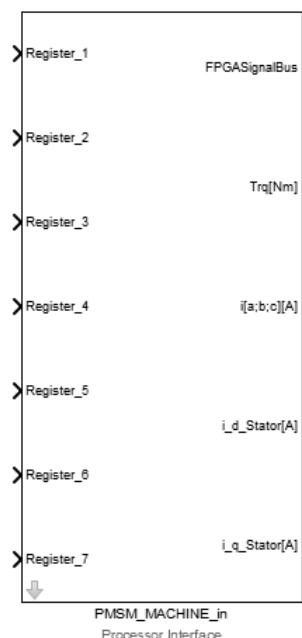


Figure 143: PMSM_MACHINE_in block

Block Dialog

The dialog only provides a short block description.

Output

The processor input block has the following outputs:

Name	Unit	Description	Range
FPGASignalBus	Bus	Bus containing the most significant signals of this block	-
Trq	[Nm]	Torque of the motor	-
i	[A]Bus	Phase current in each motor phase (a,b,c)	-
i_d_Stator	[A]	Current in direct axis	-
i_q_Stator	[A]	Current in quadrature axis	-

Block additional feedback

Adapts the FPGA signals for additional feedback of the FPGA to the processor side (like power consumption,...).

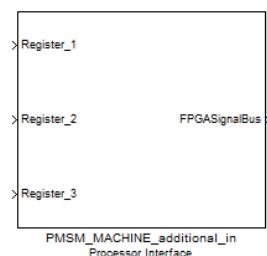


Figure 144: PMSM_MACHINE_additional_in block

Block Dialog

The dialog only provides a short block description.

Output

The processor input block has the following outputs:

Name	Unit	Description	Range
FPGASignalBus	Bus	Bus containing the most significant signals of this block	-

Interface Examples

Processor blocks

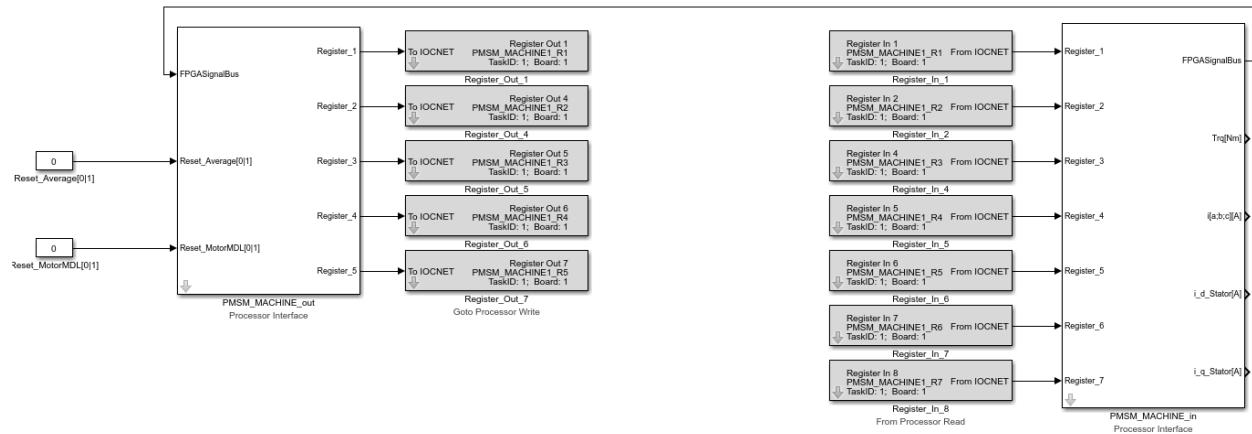
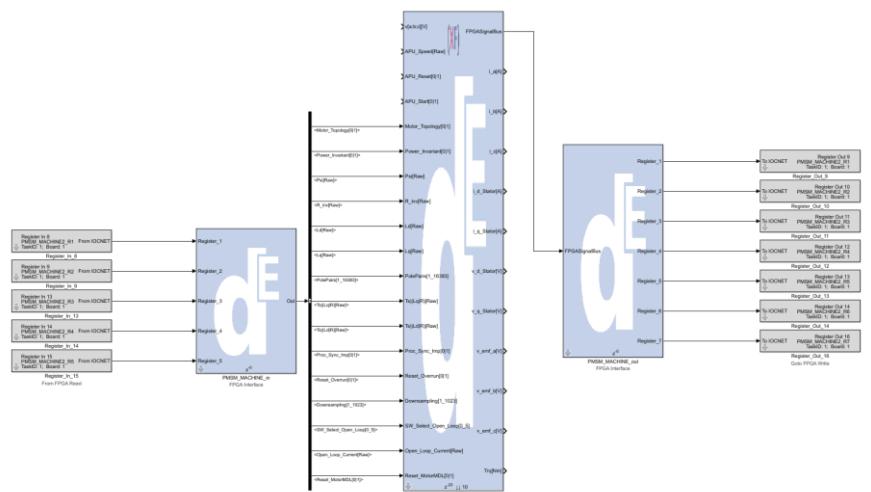


Figure 145: Processor interface

FPGA block



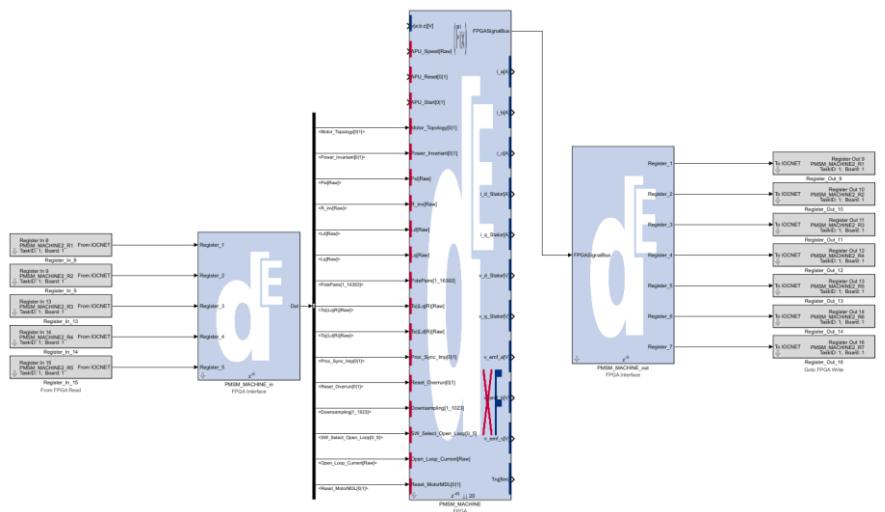


Figure 147: FPGA interface for floating point

Electric Machines: Table based Permanent Magnetized Synchronous Machine (PMSM Table based)

Objective

In this model a permanent magnet synchronous machine with table based motor parameters is integrated. In comparison to the model of the PMSM (explained in the chapter before) this model have integrated tables to simulate working point inductance of L_d , L_q and magnetic flux depending on the actual currents i_d and i_q . To simulate this dependency a normed table is integrated which contains the factors for the motor parameters which are sent from the processor. The internal normed table data and the overall model behavior (table based motor parameters or fix values) can be set during runtime.

The generated back EMF from the PMSM has sinusoidal shape, so that the machine is modeled in dq-coordinates (also known as rotor reference frame). All motor parameters are adjustable from the processor during runtime with respect to the sampling rate of the processor.

Next to the parameterization of the motor specific parameters, the motor can also be set into stimulus mode, too just set one of the phase currents (a , b , c) or one of the transformed currents (d , q) to a specific value (to check if the current scaling fits to the ECU). The motor model also calculates the actual processor synchronous average values for the currents and the torque, where a downsampling might be useful to cover as well slow processor clock rates, but keep in mind that the resolution might decrease. An additional, reset functionality of the average function and the motor integrators is also implemented.

	If the standard interface of the XSG Electric Library is used, make sure that the PMSM_TABLE_BASED_in and PMSM_TABLE_BASED_out block of the processor interface is located in the same sampled task. This ensures that the timing behavior of the included average functionality of the FPGA main block is correct. Infractions of this requirement produce fail behavior in value conversion and calculation on the processor interface!
	If the machine configuration is set to delta, the maximum voltage input voltage of the motor is reduced to $1024V / \sqrt{3}$.

All motor parameters can be adjusted (during runtime) in the processor output block dialog. To get a better overview of the adjustability of the motor parameters and the internal motor model an equivalent circuit diagram is shown below.

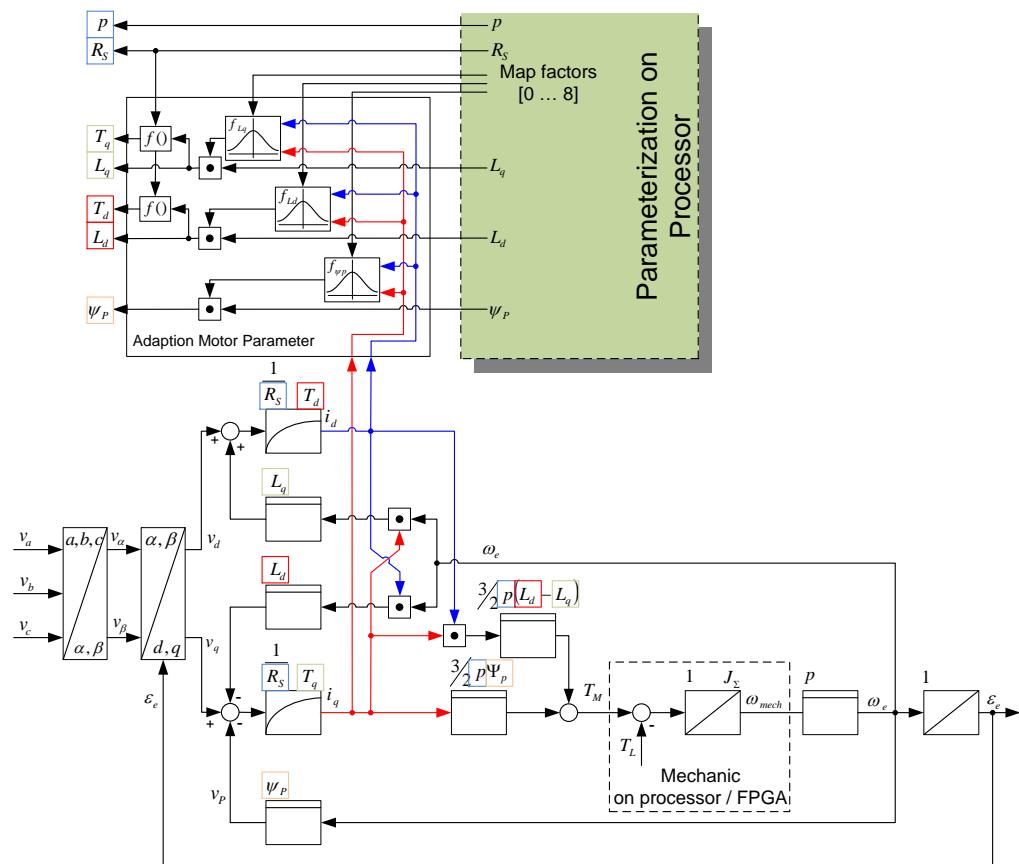


Figure 148: Circuit diagram of the PMSM table based motor model

The PMSM is modeled using dq notation in the synchronously rotating reference frame. The following equation describes the motor mathematical model.

$$\begin{aligned}
 L_d \frac{di_d}{dt} &= v_d - R_S i_d + L_q \omega_e i_q \\
 L_q \frac{di_q}{dt} &= v_q - R_S i_q - L_d \omega_e i_d - \psi_p \omega_e \\
 Trq &= \frac{3}{2} p [\psi_p i_q + (L_d - L_q) i_d i_q]
 \end{aligned}$$

L_d	Direct axis inductance
L_q	Quadrature axis inductance
R_S	Stator winding resistance
ψ_p	Flux induced by the magnet
p	Number of pole pairs

The blockset contains the following elements:

- Processor Interface: PMSM_TABLE_BASED_out (Processor Interface)

- FPGA Interface: PMSM_TABLE_BASED_in
(FPGA Interface)
- FPGA: PMSM_TABLE_BASED
(FPGA Main Component)
- FPGA Interface: PMSM_TABLE_BASED_out
(FPGA Interface)
- Processor Interface: PMSM_TABLE_BASED_in
(Processor Interface)

Processor Output

Block

Merges the processor signals and writes them to the FPGA

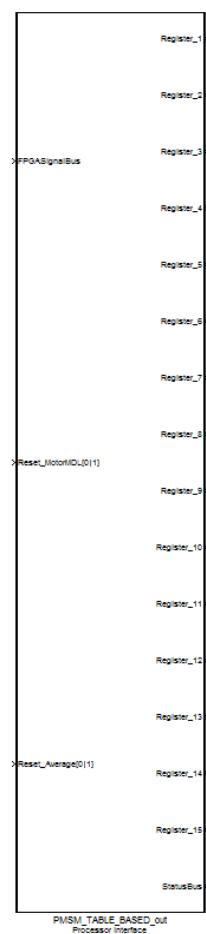


Figure 149: PMSM_TABLE_BASED_out block

Block Dialog

The processor output block contains several dialogs. All general parameters of the block can be configured in the page "General".

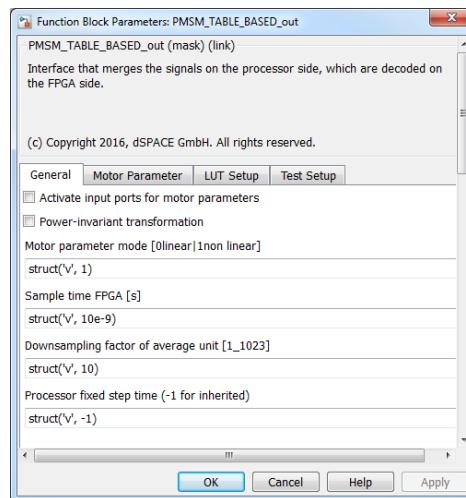


Figure 150: PMSM_TABLE_BASED_out dialog; page “General”

The page has the following parameters:

Name	Unit	Description	Range
Activate input ports	[-]	Activate input ports for the inductance, magnetic flux and stator resistance	-
Power-invariant transformation	[-]	Activate power invariant transformation for clark transformation	-
Motor parameter mode	[-]	Specify the motor parameter mode 0: linear 1: non linear	0 1
Sample time FPGA	[s]	Sample time of the FPGA	
Downsampling of average unit	[-]	Downsampling factor of the average functionality in the range 1...1023	1 ... 1023; resolution: 1
Processor fixed step time	[s]	Step time of the processor task. For offline simulation a different sample time can be set to simulate the processor interface communication behavior.	-

The motor specific parameters can be configured on the page “Motor Parameter” as show in the figure below.

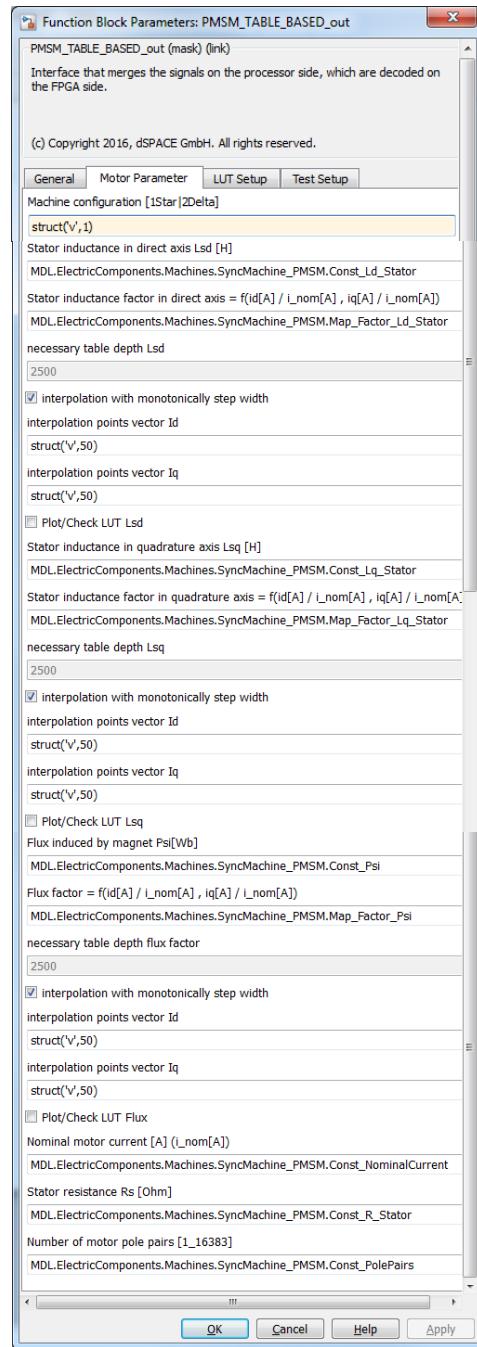


Figure 151: PMSM_TABLE_BASED_out dialog; page “Motor Parameters”

The page has the following parameters:

Name	Unit	Description	Range
Machine configuration	[-]	Select the machine configuration; 1: star 2: delta	-
Stator inductance in direct axis Lsd	[H]	Stator inductance in the direct axis; this field is deactivated if the “activate input ports” checkbox on page “General” is set to on	0 ... 0.25 H; resolution: 3.81E-6
Stator inductance factor in direct axis	[-]	Stator inductance factor of the direct axis normalized with the working point current i_norm	0 ... 4
Necessary table depth Lsd	[-]	Information about the necessary table depth of the factor table on the FPGA	
Interpolation with monotonically step width	[-]	Specify if the factor table should be internal interpolated with the actual number of interpolation points	[-]
Interpolation points vector Id	[-]	Specify the number of points in direct axis for the internal interpolation functionality; this field is deactivated if the interpolation checkbox is set to off	[-]
Interpolation points vector Iq	[-]	Specify the number of points in the quadrature axis for the internal interpolation functionality; this field is deactivated if the interpolation checkbox is set to off	[-]
Plot / Check LUT Lsd	[-]	Let the user plot the actual display of the table factor	[-]
Stator inductance in quadrature axis Lsq	[H]	Stator inductance in the quadrature axis; this field is deactivated if the “activate input ports” checkbox on page “General” is set to on	0 ... 0.25 H; resolution: 3.81E-6

Stator inductance factor in quadrature axis	[]	Stator inductance factor of the quadrature axis normalized with the working point current <i>i_norm</i>	0 ... 4
Necessary table depth Lsq	[]	Information about the necessary table depth of the factor table on the FPGA	
Interpolation with monotonically step width	[]	Specify if the factor table should be internal interpolated with the actual number of interpolation points	[]
Interpolation points vector Id	[]	Specify the number of points in direct axis for the internal interpolation functionality; this field is deactivated if the interpolation checkbox is set to off	[]
Interpolation points vector Iq	[]	Specify the number of points in the quadrature axis for the internal interpolation functionality; this field is deactivated if the interpolation checkbox is set to off	[]
Plot / Check LUT Lsq	[]	Let the user plot the actual display of the table factor	[]
Flux induced by magnet Psi	[Wb]	Magnetic flux of the motor magnets; this field is deactivated if the “activate input ports” checkbox on page “General” is set to on	0 ... 1 Wb; resolution: 15.2E-6
Magnetic flux factor	[]	Magnetic flux factor of the normalized with the working point current <i>i_norm</i>	0 ... 4
Necessary table depth Psi	[]	Information about the necessary table depth of the factor table on the FPGA	

Interpolation with monotonically step width	[-]	Specify if the factor table should be internal interpolated with the actual number of interpolation points	[-]
Interpolation points vector Id	[-]	Specify the number of points in direct axis for the internal interpolation functionality; this field is deactivated if the interpolation checkbox is set to off	[-]
Interpolation points vector Iq	[-]	Specify the number of points in the quadrature axis for the internal interpolation functionality; this field is deactivated if the interpolation checkbox is set to off	[-]
Plot / Check LUT Flux	[-]	Let the user plot the actual display of the table factor	[-]
Nominal motor current	[-]	Nominal motor current used for normalization input currents of the table factor	1 ... $2^{14}-1$ A
Stator resistance Rs	[Ohm]	Stator resistance; this field is deactivated if the "activate input ports" checkbox on page "General" is set to on	244E-6 ... 16 Ohm; resolution: 244E-6
Number of pole pairs	[-]	Number of motor pole pairs	0 ... 16383; resolution: 1

General parameters of the LUT parameters are located on the page "LUT Setup".

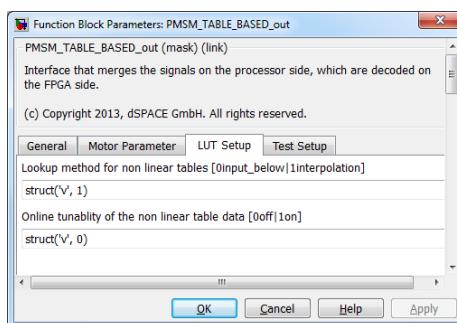


Figure 152: PMSM_TABLE_BASED_out dialog; page "LUT Setup"

The page has the following parameters:

Name	Unit	Description	Range
Lookup method for non linear tables	[-]	Specify if the output of the table data interpolated or input below	0 1
Online tuneability of the non linear table data	[-]	Specify if the table data is changed during runtime on the FPGA; Keep in mind that every download of new table data assumed a download time; If this function is enabled, the turnaround time of your real-time model will increase.	0 1

To enable the motor model in test mode the parameters of page “Test Setup” are available.

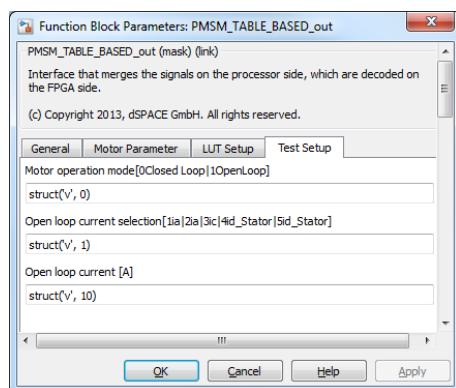


Figure 153: PMSM_TABLE_BASED_out dialog; page “Test Setup”

The page has the following parameters:

Name	Unit	Description	Range
Operation Mode	[-]	Selection of the Operation Mode of the motor model; 1: Open Loop test 0: Closed Loop test	0 1
Open Loop Current Selector	[-]	Selector which current is stimulated if the operation mode is set to open loop test; 1: i_a 2: i_b 3: i_c 4: i_d_Stator 5: i_q_Stator	-

Input

The PMSM_MACHINE_out block has the following inputs if the “Activate input ports” are disabled on page “General”:

Name	Unit	Description	Range
FPGASignalBus	Bus	Bus containing the most significant feedback signals from the processor input block	-
Reset_Average	[-]	Reset function of the implemented FPGA average functionality	0 1
Reset_Motor MDL	[-]	Reset function of the motor model on FPGA	0 1

If the checkbox “Activate input ports” is disabled on the page “General” the block has the following outputs:

Name	Unit	Description	Range
FPGASignalBus	Bus	Bus containing the most significant feedback signals from the processor input block	-
Reset_Average	[-]	Reset function of the implemented FPGA average functionality	0 1
Reset_Motor MDL	[-]	Reset function of the motor model on FPGA	0 1
Stator inductance in direct axis Lsd	[H]	Stator inductance in the direct axis	0 ... 0.25 H; resolution: 3.81E-6
Stator inductance in quadrature axis Lsq	[H]	Stator inductance in the quadrature axis	0 ... 0.25 H; resolution: 3.81E-6
Flux induced by magnet Psi	[Wb]	Magnetic flux of the motor magnets	0 ... 1 Wb; resolution: 15.2E-6
Stator resistance Rs	[Ohm]	Stator resistance	244E-6 ... 16 Ohm; resolution: 244E-6

Output

The processor out block provides 15 output registers. The registers are internally grouped in the following sections:

Register 1 ... 3: Parameterization of the map table of the inductance in direct axis. For detailed information about the data section of each register, please refer to the 2-D LUT documentation of the XSG Utils library.

Register 4 ... 6: Parameterization of the map table of the inductance in quadrature axis. For detailed information about the data section of each register, please refer to the 2-D LUT documentation of the XSG Utils library.

Register 7 ... 9: Parameterization of the map table of the magnetic flux. For detailed information about the data section of each register, please refer to the 2-D LUT documentation of the XSG Utils library.

Register 10...14: Variables for parameterization of a standard PMSM model. For detailed information about the data section of each register, please refer to the chapter “Electric Machines: Permanent Magnetized Synchronous Machine (PMSM)”. The only difference is the Nominal Current in this model is used over the section of the inverse current.

StatusBus: Is used for internal data transfer and hand-shaking protocols between the FPGA and the Processor blocks to manage the download and update process of the internal map tables. For detailed information about the data section of each register, please refer to the 2-D LUT documentation of the XSG Utils library.

FPGA Main Component

Block

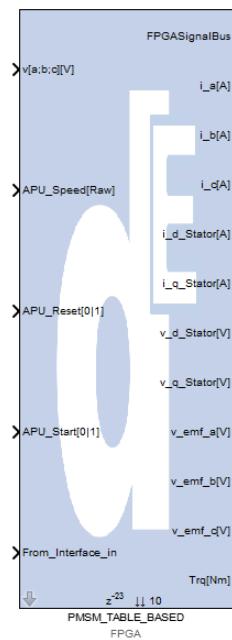


Figure 154: PMSM_TABLE_BASED FPGA Main block

Block Dialog

The FPGA main blockset contains several dialogs. All general parameters of the block can be configured in the page “General”.

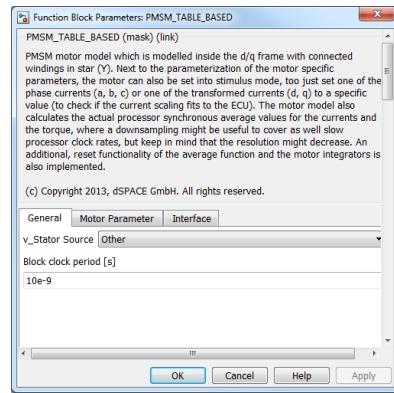


Figure 155: PMSM_TABLE_BASED FPGA Main block dialog; page: “General”

On this page the FPGA block clock period can be defined. Additional to that the connected stator voltage can be selected. Please choose the connected inverter model via the popup.

The motor specific parameters can be configured on the page “Motor Parameter” as show in the figure below.

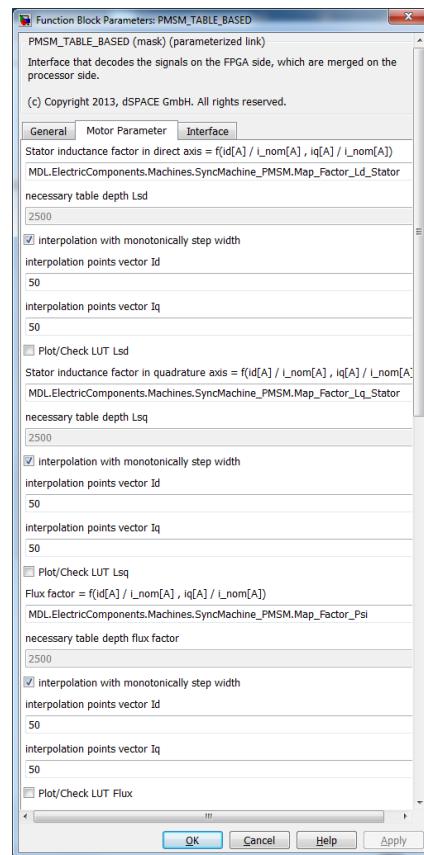


Figure 156: PMSM_TABLE_BASED FPGA Main block dialog; page: “Motor Parameter”

The page has the following parameters:

Name	Unit	Description	Range
Stator inductance factor in direct axis	[-]	Stator inductance factor of the direct axis normalized with the working point current i_norm	0 ... 8
Necessary table depth Lsd	[-]	Information about the necessary table depth of the factor table on the FPGA	
Interpolation with monotonically step width	[-]	Specify if the factor table should be internal interpolated with the actual number of interpolation points	[-]
Interpolation points vector Id	[-]	Specify the number of points in direct axis for the internal interpolation functionality; this field is deactivated if the interpolation checkbox is set to off	[-]
Interpolation points vector Iq	[-]	Specify the number of points in the quadrature axis for the internal interpolation functionality; this field is deactivated if the interpolation checkbox is set to off	[-]
Plot / Check LUT Lsd	[-]	Let the user plot the actual display of the table factor	[-]
Stator inductance factor in quadrature axis	[-]	Stator inductance factor of the quadrature axis normalized with the working point current i_norm	0 ... 8
Necessary table depth Lsq	[-]	Information about the necessary table depth of the factor table on the FPGA	
Interpolation with monotonically step width	[-]	Specify if the factor table should be internal interpolated with the actual number of interpolation points	[-]

Interpolation points vector Id	[-]	Specify the number of points in direct axis for the internal interpolation functionality; this field is deactivated if the interpolation checkbox is set to off	[-]
Interpolation points vector Iq	[-]	Specify the number of points in the quadrature axis for the internal interpolation functionality; this field is deactivated if the interpolation checkbox is set to off	[-]
Plot / Check LUT Lsq	[-]	Let the user plot the actual display of the table factor	[-]
Magnetic flux factor	[-]	Magnetic flux factor of the normalized with the working point current i_{norm}	0 ... 8
Necessary table depth Psi	[-]	Information about the necessary table depth of the factor table on the FPGA	
Interpolation with monotonically step width	[-]	Specify if the factor table should be internal interpolated with the actual number of interpolation points	[-]
Interpolation points vector Id	[-]	Specify the number of points in direct axis for the internal interpolation functionality; this field is deactivated if the interpolation checkbox is set to off	[-]
Interpolation points vector Iq	[-]	Specify the number of points in the quadrature axis for the internal interpolation functionality; this field is deactivated if the interpolation checkbox is set to off	[-]
Plot / Check LUT Flux	[-]	Let the user plot the actual display of the table factor	[-]

On the page Interface you can define the parameters of the input and output registers for automatic interface generation. The dialog also contains a button to start interface generation. For more details about automatic interface generation, refer to the XSG Utils documentation chapter Automatic Interface Generation. The FPGA clock period can be configured too.

Input

The main block has the following inputs:

Name	Unit	Description	Format
v	[V] (vector)	Stator voltage vector of the motor phase	Fix_16_7
APU_Speed	[Raw]	Actual speed	Fix_30_0
APU_Reset	[-]	Reset on high state	Bool
APU_Start	[-]	Start on high state	UFix_1_0
From_Interface_in	Bus	Because it is not useful to separate and modify any variables of the FPGA interface_in block the FPGA main component block only supports the whole data bus of the interface block	-

Output

The PMSM_TABLE_BASED main block has the following outputs with a maximal latency of thirteen:

Name	Unit	Description	Format
FPGASignalBus	Bus	Bus containing the most significant signals of this block	-
i_a	[A]	Current in phase a	Fix_18_7
i_b	[A]	Current in phase b	Fix_18_7
i_c	[A]	Current in phase c	Fix_18_7
i_d_Stator	[A]	Current in the direct axis	Fix_18_7
i_q_Stator	[A]	Current in the quadrature axis	Fix_18_7
v_d_Stator	[V]	Voltage in direct axis	Fix_18_7
v_q_Stator	[V]	Voltage in quadrature axis	Fix_18_7
v_emf_a	[V]	Back EMF Voltage in phase a	Fix_18_7
v_emf_b	[V]	Back EMF Voltage in phase b	Fix_18_7
v_emf_c	[V]	Back EMF Voltage in phase c	Fix_18_7
Trq	[Nm]	Torque of the motor	Fix_25_14

Processor Input

Block

Adapts the FPGA signals for the processor side.



Figure 157: PMSM_TABLE_BASED_in block

Block Dialog

The dialog only provides a short block description.

Input

The processor out block provides 15 output registers. The registers are internally grouped in the following sections:

Register 1 ... 3: Feedback of the map table of the inductance in direct axis. For detailed information about the data section of each register, please refer to the 2-D LUT documentation of the XSG Utils library.

Register 4 ... 6: Feedback of the map table of the inductance in quadrature axis. For detailed information about the data section of each

register, please refer to the 2-D LUT documentation of the XSG Utils library.

Register 7 ... 9: Feedback of the map table of the magnetic flux. For detailed information about the data section of each register, please refer to the 2-D LUT documentation of the XSG Utils library.

Register 10...16: Feedback of a standard PMSM model. For detailed information about the data section of each register, please refer to the chapter “Electric Machines: Permanent Magnetized Synchronous Machine (PMSM)”.

Output

The processor input block has the following outputs:

Name	Unit	Description	Range
FPGASignalBus	Bus	Bus containing the most significant signals of this block	-
Trq	[Nm]	Torque of the motor	-1024 ... 1023.99 Nm
i	[A]Bus	Phase current in each motor phase (a,b,c)	-1024 ... 1023.99 A
i_d_Stator	[A]	Current in direct axis	-1024 ... 1023.99 A
i_q_Stator	[A]	Current in quadrature axis	-1024 ... 1023.99 A

Block additional feedback

Adapts the FPGA signals for additional feedback of the FPGA to the processor side (like power consumption,...).

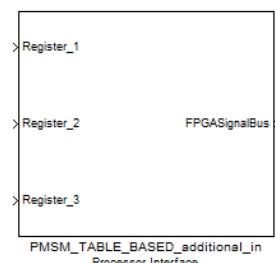


Figure 158: PMSM_TABLE_BASED_additional_in block

Block Dialog

The dialog only provides a short block description.

Input

The processor input block has the following inputs:

Register 1

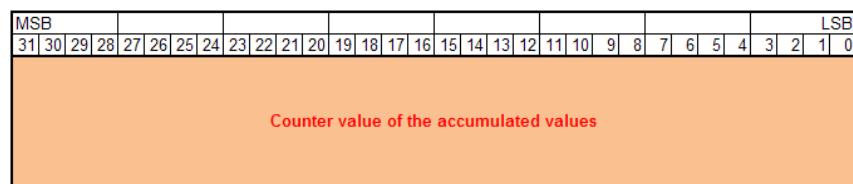


Figure 159: PMSM_TABLE_BASED_additional_in Register 1

Name	Used Bits	Description	Range
Accu	31 ... 0	Counter value of the accumulated values; if an overrun is detected the value $2^{32}-1$ will be sent	0 ... $2^{32}-1$

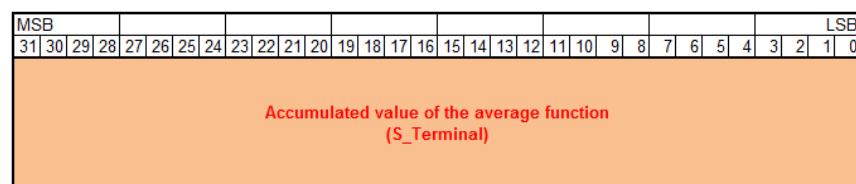
Register 2

Figure 160: PMSM_TABLE_BASED_additional_in Register 2

Name	Used Bits	Description	Range
Accu	31 ... 0	Accumulated value of the average function; if an overrun is detected the value $2^{31}-1$ will be sent Signal: input power consumption S_Terminal [VA]	0 ... $2^{32}-1$

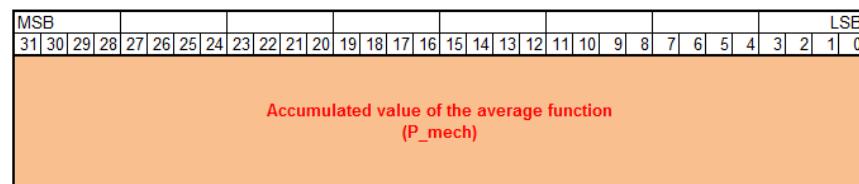
Register 3

Figure 161: PMSM_TABLE_BASED_additional_in Register 3

Name	Used Bits	Description	Range
Accu	31 ... 0	Accumulated value of the average function; if an overrun is detected the value 2^31-1 will be sent Signal: mechanical output power P_mech [W]	0 ... 2^32-1

Output

The processor input block has the following outputs:

Name	Unit	Description	Range
FPGASignalBus	Bus	Bus containing the most significant signals of this block	-

Interface Examples

Processor blocks

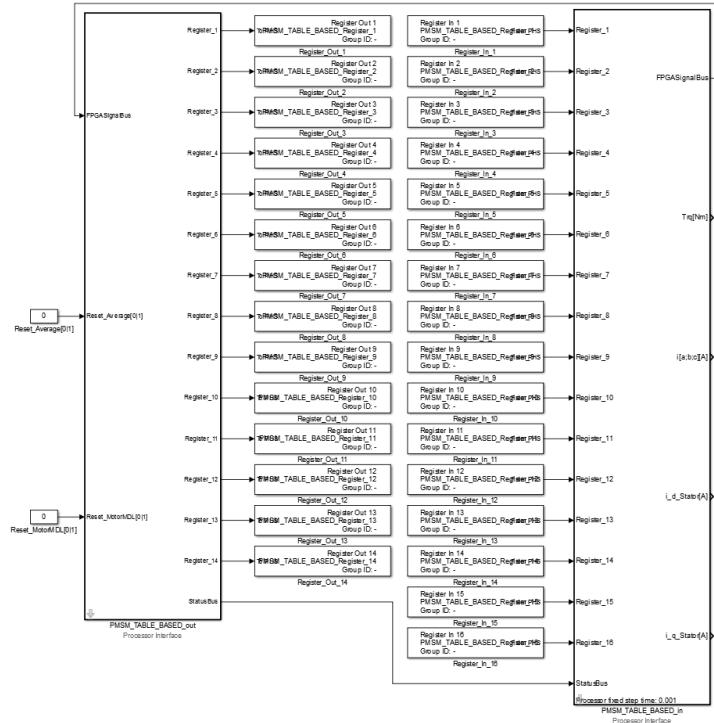


Figure 162: Processor interface

FPGA block

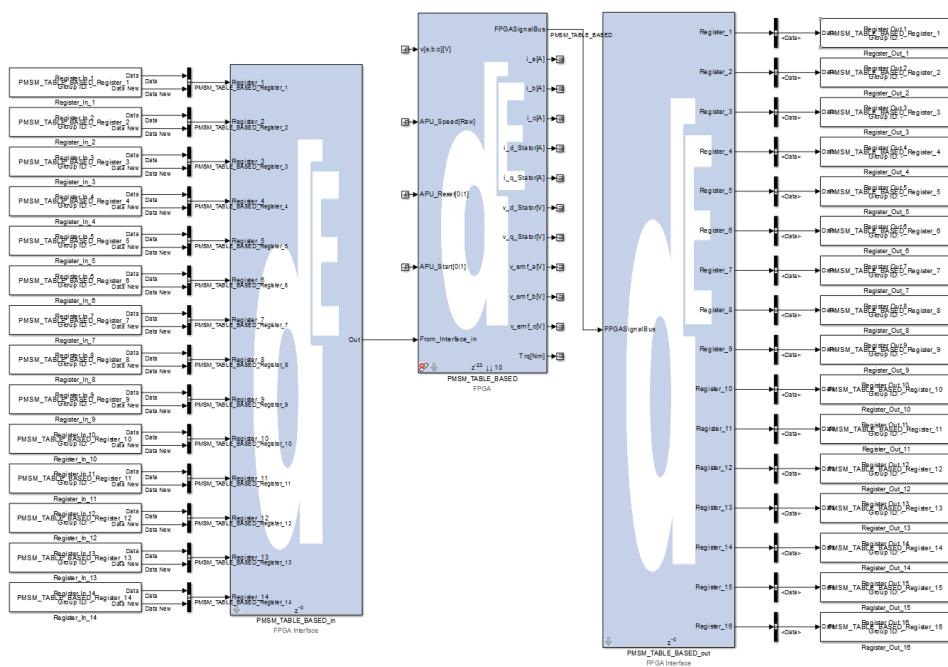


Figure 163: FPGA interface

Electric Machines: Squirrel Cage Induction Machine (SCIM)

Objective

In this model a squirrel cage induction machine is integrated. Next to the parameterization of the motor specific parameters, the motor can also be set into stimulus mode, too just set one of the phase currents (a , b , c) to a specific value (to check if the current scaling fits to the target ECU). The motor model also calculates the actual processor synchronous average values for the currents and the torque, where a downsampling might be useful to cover as well slow processor clock rates, but keep in mind that the resolution might decrease. An additional reset functionality of the average function and the motor integrators is also implemented. Due to faster calculation time and FPGA recourse consumption the motor model is designed inside the stator reference frame (alpha/beta).

	If the standard interface of the XSG Electric Library is used, make sure that the SCIM_MACHINE_in and SCIM_MACHINE_out block of the processor interface is located in the same sampled task. This ensures that the timing behavior of the included average functionality of the FPGA main block is correct. Infractions of this requirement produce fail behavior in value conversion and calculation on the processor interface!
	If the machine configuration is set to delta, the maximum voltage input voltage of the motor is reduced to $1024V / \sqrt{3}$.

All motor parameters can be adjusted (during runtime) in the processor output block dialog. The squirrel cage asynchronous machine is modeled using alpha beta-coordinates in the stator reference frame. The next figure shows the equivalent circuit diagram of the model.

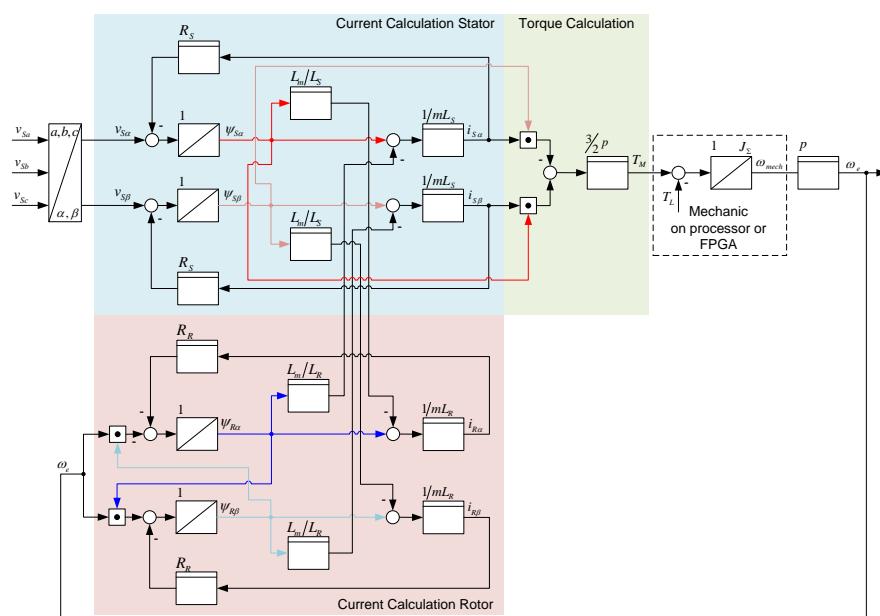


Figure 164: Circuit diagram of the SCIM motor model

The following equation describes the motor mathematical model.

Calculation of stator current / magnetic flux in stator reference frame:

$$\frac{d\psi_{S\alpha}}{dt} = -R_s \cdot i_{S\alpha} + u_{S\alpha}$$

$$\frac{d\psi_{S\beta}}{dt} = -R_s \cdot i_{S\beta} + u_{S\beta}$$

$$i_{S\alpha} = \frac{1}{m \cdot L_s} \left(\psi_{S\alpha} - \frac{L_m}{L_R} \psi_{R\alpha} \right)$$

$$i_{S\beta} = \frac{1}{m \cdot L_s} \left(\psi_{S\beta} - \frac{L_m}{L_R} \psi_{R\beta} \right)$$

Calculation of rotor current / magnetic flux in stator reference frame:

$$\frac{d\psi_{R\alpha}}{dt} = -R_R \cdot i_{R\alpha} - \omega_e \cdot \psi_{R\beta}$$

$$\frac{d\psi_{R\beta}}{dt} = -R_R \cdot i_{R\beta} + \omega_e \cdot \psi_{R\alpha}$$

$$i_{R\alpha} = \frac{1}{m \cdot L_R} \left(\psi_{R\alpha} - \frac{L_m}{L_R} \psi_{S\alpha} \right)$$

$$i_{R\beta} = \frac{1}{m \cdot L_R} \left(\psi_{R\beta} - \frac{L_m}{L_R} \psi_{S\beta} \right)$$

With respect to abstraction of m:

$$m = 1 - \frac{L_m^2}{L_R L_S}$$

The actual torque of the motor is calculated with:

$$Trq = \frac{3}{2} p (\psi_{S\alpha} i_{S\beta} - \psi_{S\beta} i_{S\alpha})$$

Legend of formula symbols:

$\psi_{S\alpha}$	Stator magnetic flux in alpha coordinates
$\psi_{S\beta}$	Stator magnetic flux in beta coordinates
$\psi_{R\alpha}$	Rotor magnetic flux in alpha coordinates
$\psi_{R\beta}$	Rotor magnetic flux in beta coordinates
R_s	Stator winding resistance
R_R	Rotor winding resistance
L_s	Stator inductance
L_R	Rotor inductance

$$\begin{aligned} L_m & \quad \text{Mutual inductance} \\ P & \quad \text{Number of pole pairs} \end{aligned}$$

The blockset contains the following elements:

- Processor Interface: SCIM_MACHINE_out (Processor Interface)
- FPGA Interface: SCIM_MACHINE_in (FPGA Interface)
- FPGA: SCIM_MACHINE (FPGA Main Component)
- FPGA Interface: SCIM_MACHINE_out (FPGA Interface)
- Processor Interface: SCIM_MACHINE_in (Processor Interface)

Processor Output

Block	Merges the processor signals and writes them to the FPGA
--------------	----------------------------------------------------------



Figure 165: SCIM_MACHINE_out block

Block Dialog	The processor output block contains several dialogs. All general parameters of the block can be configured in the page "General".
---------------------	-----------------------------------------------------------------------------------------------------------------------------------

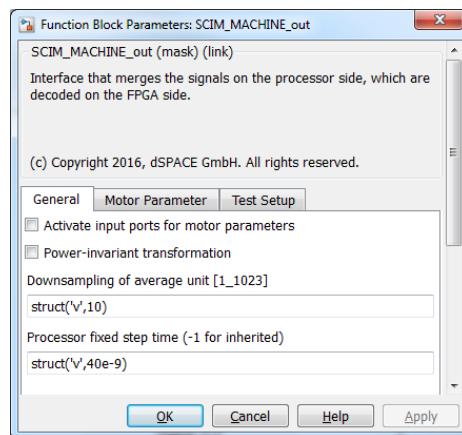


Figure 166: SCIM_MACHINE_out dialog; page "General"

The page has the following parameters:

Name	Unit	Description	Range
Activate input ports	[-]	Activate input ports for the inductance, magnetic flux and stator resistance	-
Power-invariant transformation	[-]	Activate power invariant transformation for clark transformation	-
Downsampling of average unit	[-]	Downsampling factor of the average functionality in the range 1...1023; resolution: 1	1 ... 1023; resolution: 1
Processor fixed step time	[s]	Step time of the processor task. For offline simulation a different sample time can be set to simulate the processor interface communication behavior.	-

The motor specific parameters can be configured on the page "Motor Parameter" as show in the figure below.

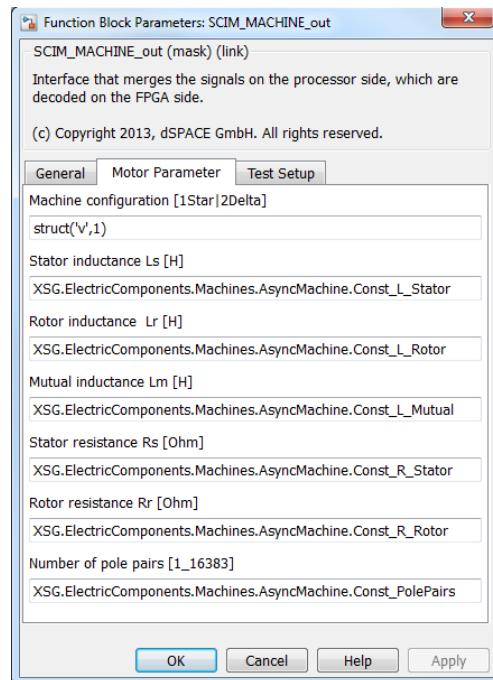


Figure 167: SCIM_MACHINE_out dialog; page "Motor Parameter"

The page has the following parameters:

Name	Unit	Description	Range
Stator inductance Ls	[H]	Stator inductance	Depends on the combination of Ls Lr and Lm
Rotor inductance Lr	[H]	Rotor inductance	Depends on the combination of Ls Lr and Lm
Mutal inductance Lm	[H]	Mutal inductance	Depends on the combination of Ls Lr and Lm
Stator resistance Rs	[Ohm]	Stator resistance	244E-6 ... 16 Ohm; resolution: 244E-6
Rotor resistance Rr	[Ohm]	Rotor resistance	244E-6 ... 16 Ohm; resolution: 244E-6
Number of pole pairs	[-]	Number of motor pole pairs	0 ... 16383; resolution: 1

Specific settings for testing the outputs of the machine can be configured on the page “Test Setup”.

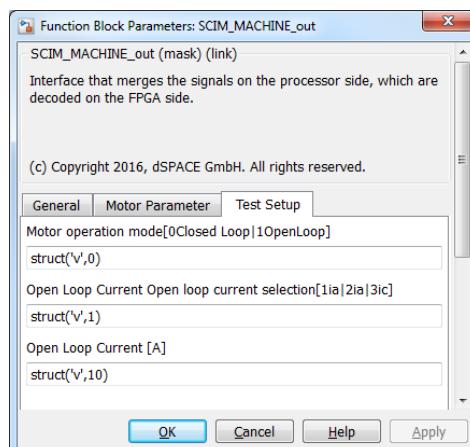


Figure 168: SCIM_MACHINE_out dialog; page “Test Setup”

The page has the following parameters:

Name	Unit	Description	Range
OperationMode	[-]	Selection of the Operation Mode of the motor model; 1: Open Loop test 0: Closed Loop test	0 1
Open Loop Current Selector	[-]	Selector which current is stimulated if the operation mode is set to open loop test; 1: i_a 2: i_b 3: i_c	-
Open Loop Current	[A]	Value for the stimulated open loop current	A

Input

The SCIM_MACHINE_out block has the following inputs. If input ports for the motor parameters are activated via the GUI checkbox, additional ports for the relevant motor parameters are added.

Name	Unit	Description	Range
FPGASignalBus	Bus	Bus containing the most significant feedback signals from the processor input block	-
Reset_Average	[-]	Reset function of the implemented FPGA average functionality	0 1
Reset_Motor MDL	[-]	Reset function of the motor model on FPGA	0 1

Output

The processor out block provides five input registers whose sectioning is shown below:

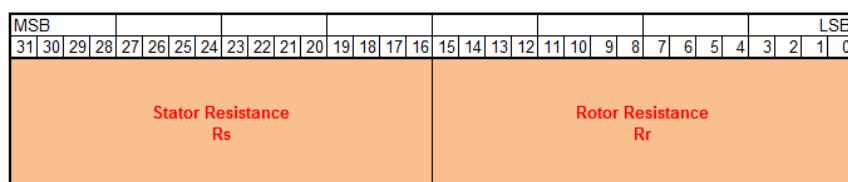
Register 1

Figure 169: SCIM_MACHINE_out Register 1

Name	Used Bits	Description	Range
Rotor Resistance Rr	15 ... 0	Rotor resistance Rr	0 ... 2^16-1 represents 250E-6 ...16 Ohm
Stator Resistance Rs	31 ... 16	Stator resistance Rs	0 ... 2^16-1 represents 250E-6 ...16 Ohm

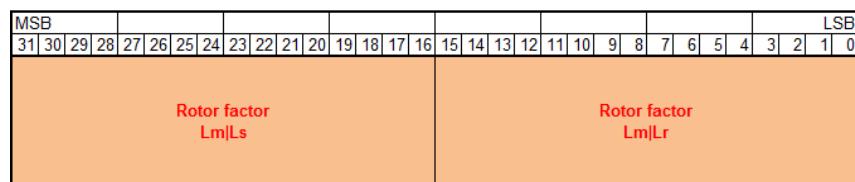
Register 2

Figure 170: SCIM_MACHINE_out Register 2

Name	Used Bits	Description	Range
Rotor factor Lm Lr	15 ... 0	Calculated rotor factor Lm Lr	0 ... 2^16-1 represents 0 ... 1
Rotor factor Lm Ls	31 ... 16	Calculated rotor factor Lm Ls	0 ... 2^16-1 represents 0 ... 1

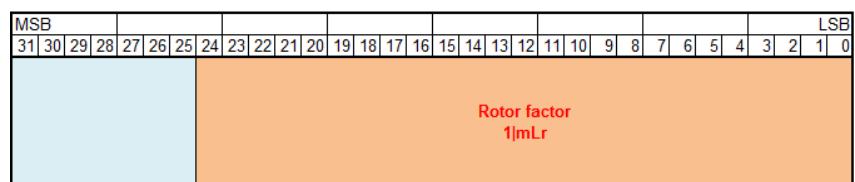
Register 3

Figure 171: SCIM_MACHINE_out Register 3

Name	Used Bits	Description	Range
Stator factor 1 mLr	24 ... 0	Calculated stator factor 1 mLr	0 ... 2^25-1 represents 0 ... 262144 1 H
SPARE	31 ... 25		

Register 4

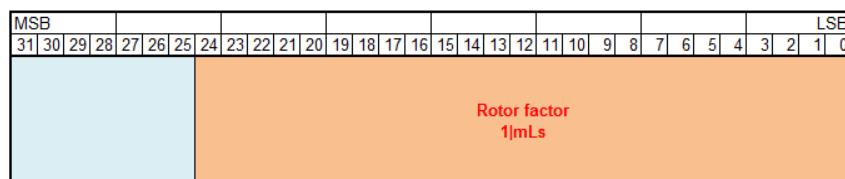


Figure 172: SCIM_MACHINE_out Register 4

Name	Used Bits	Description	Range
Stator factor 1 mLs	24 ... 0	Calculated stator factor 1 mLs	0 ... 2^25-1 represents 0 ... 262144 1 H
SPARE	31 ... 25		

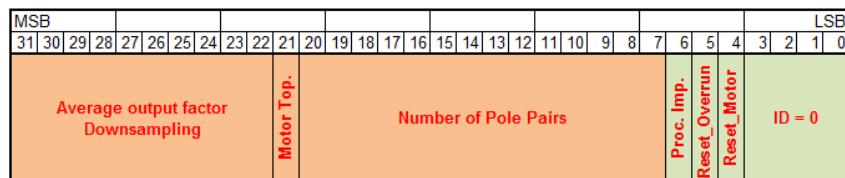
Register 5; ID = 0

Figure 173: SCIM_MACHINE_out Register 5; ID = 0

Name	Used Bits	Description	Range
Counter ID	3 ... 0	ID to select which register coding is activated via Processor	0 ... 1
Reset_Motor MDL	4	Reset flag for the motor model on the FPGA	0 1
Reset_OVERRUN	5	Reset flag to reset an overrun of the average functionality	0 1
Proc. Sync. Impulse	6	Processor synchronous toggle bit	0 1
Number of Pole Pairs	13 ... 0	Number of pole pairs	1 ... 16363
Motor Topology	30	Specify the motor topology	0 1
Average output factor Down-sampling	9 ... 0	Downsampling factor for average functionality of the outputs to the processor interface	1 ... 1023

Register 5; ID = 1



Figure 174: SCIM_MACHINE_out Register 5; ID = 1

Name	Used Bits	Description	Range
Continuous Data	6 ... 0	Definition same as ID = 0	
Enable/SW Open Loop	9 ... 7	Switch and Enable of the open loop stimulation	0 ... 3
OpenLoop Current	27 ... 10	Open Loop current for open loop stimulation	-1024 ... 1024
Power Invariant	28	Switch to activate power invariant transformation	0 1
SPARE	31 ... 29		

FPGA Main Component

Block

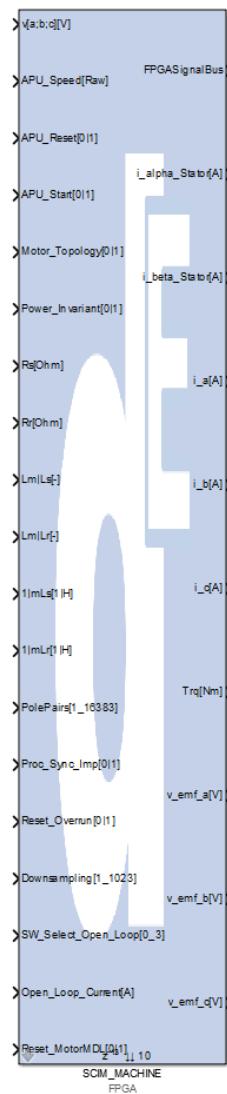


Figure 175: SCIM_MACHINE FPGA Main block

Block Dialog

The FPGA main blockset contains the following dialog.

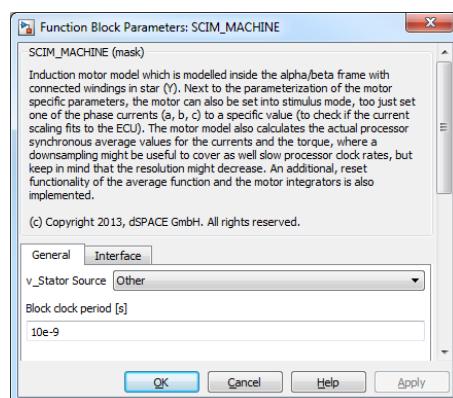


Figure 176: SCIM_MACHINE FPGA Main block dialog

On the page General the connected stator voltage can be selected. Please choose the connected inverter model via the popup.

On the page Interface you can define the parameters of the input and output registers for automatic interface generation. The dialog also contains a button to start interface generation. For more details about automatic interface generation, refer to the XSG Utils documentation chapter Automatic Interface Generation. The FPGA clock period can be configured too.

Input

The main block has the following inputs:

Name	Unit	Description	Format
v	[V] (vector)	Stator voltage vector of the motor phase	Fix_16_7
APU_Speed	[Raw]	Actual speed	Fix_30_0
APU_Reset	[‐]	Reset on high state	Bool
APU_Start	[‐]	Start on high state	UFix_1_0
Motor_Topology	[‐]	Specify the motor topology	UFix_1_0
Power_Invariant	[‐]	Specify the internal clark transformation	UFix_1_0
Rs	[Ohm]	Stator resistance	UFix_16_16
Rr	[Ohm]	Rotor resistance	UFix_16_12
Lm Ls	[‐]	Stator factor Lm Ls	UFix_16_6
Lm Lr	[‐]	Rotor factor Lm Lr	UFix_16_6
1 mLs	[1 H]	Stator factor 1 mLs	UFix_25_7
1 mLr	[1 H]	Rotor factor 1 mLr	UFix_25_7
PolePairs	[‐]	Pole pairs of the resolver	UFix_14_0
Proc_Sync_Imp	[‐]	Processor synchronize toggle bit	UFix_1_0
Reset_OVERRUN	[‐]	Reset flag to reset an overrun of the average functionality	Bool
Downsampling	[‐]	Downsampling factor for average functionality of the outputs to the processor interface	UFix_10_0
SW_Select_Open_Loop	[‐]	Switch and Selector for the Open Loop Control	UFix_4_0
Open_Loop_Current	[A]	Open Loop current	UFix_18_7
Reset_Motor_MDL	[‐]	Reset flag to the motor model	Bool

Output

The SCIM_MACHINE main block has the following outputs:

Name	Unit	Description	Format
FPGASignalBus	Bus	Bus containing the most significant signals of this block	-
i_alpha_Stator	[A]	Current in phase a	Fix_18_7
i_beta_Stator	[A]	Current in phase b	Fix_18_7
i_a	[A]	Current in phase a	Fix_18_7
i_b	[A]	Current in phase b	Fix_18_7
i_c	[A]	Current in phase c	Fix_18_7
Trq	[Nm]	Torque of the motor	Fix_25_14
v_emf_a	[V]	Back EMF Voltage in phase a	Fix_18_7
v_emf_b	[V]	Back EMF Voltage in phase b	Fix_18_7
v_emf_c	[V]	Back EMF Voltage in phase c	Fix_18_7

Processor Input

Block Adapts the FPGA signals for the processor side.

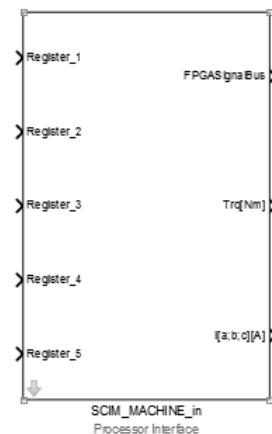


Figure 177: SCIM_MACHINE_in block

Block Dialog The dialog only provides a short block description.

Input The processor input block has the following inputs:

Register 1

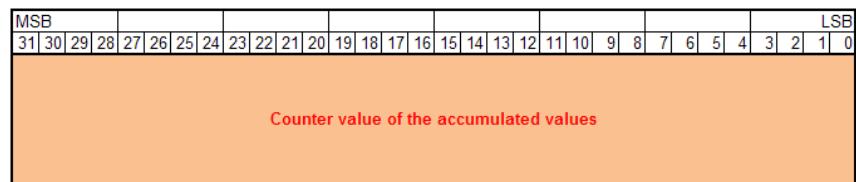


Figure 178: SCIM_MACHINE_in Register 1

Name	Used Bits	Description	Range
Accu	31 ... 0	Counter value of the accumulated values; if an overrun is detected the value $2^{32}-1$ will be sent	0 ... $2^{32}-1$

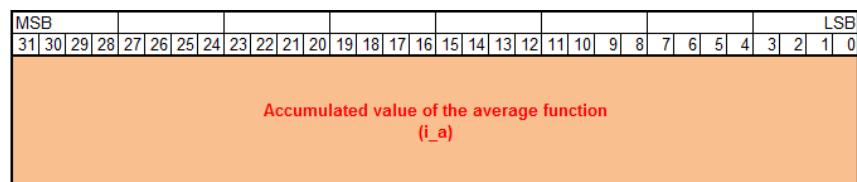
Register 2

Figure 179: SCIM_MACHINE_in Register 2

Name	Used Bits	Description	Range
Accu	31 ... 0	Accumulated value of the average function; if an overrun is detected the value $2^{31}-1$ will be sent Signal: i_a	0 ... $2^{32}-1$

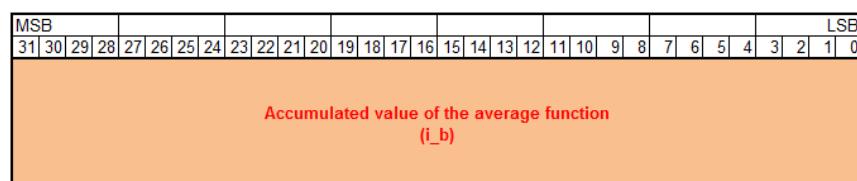
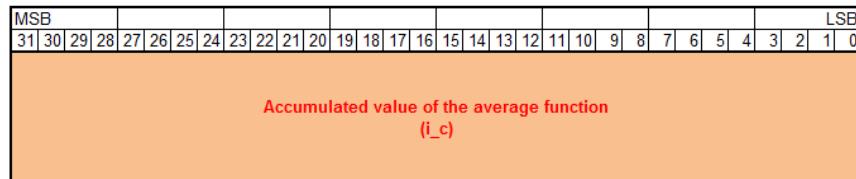
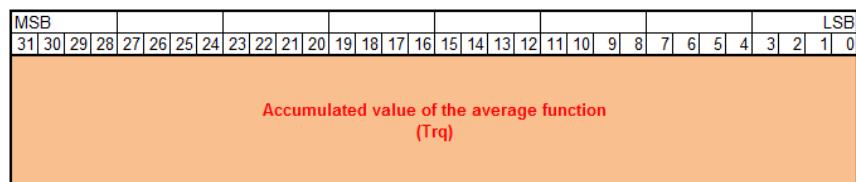
Register 3

Figure 180: SCIM_MACHINE_in Register 3

Name	Used Bits	Description	Range
Accu	31 ... 0	Accumulated value of the average function; if an overrun is detected the value $2^{31}-1$ will be sent Signal: i_b	0 ... $2^{32}-1$

Register 4**Figure 181: SCIM_MACHINE_in Register 4**

Name	Used Bits	Description	Range
Accu counter	31 ... 0	Accumulated value of the average function; if an overrun is detected the value 2^31-1 will be sent Signal: i_c	0 ... 2^32-1

Register 5**Figure 182: SCIM_MACHINE_in Register 5**

Name	Used Bits	Description	Range
Accu	31 ... 0	Accumulated value of the average function; if an overrun is detected the value 2^31-1 will be sent Signal: Trq	0 ... 2^32-1

Output

The processor input block has the following outputs:

Name	Unit	Description	Range
FPGASignalBus	Bus	Bus containing the most significant signals of this block	-
Trq	[Nm]	Torque of the motor	-1024 ... 1023.99 Nm
i	[A]Bus	Phase current in each motor phase (a,b,c)	-1024 ... 1023.99 A

Block additional feedback	Adapts the FPGA signals for additional feedback of the FPGA to the processor side (like power consumption,...).
----------------------------------	-----------------------------------------------------------------------------------------------------------------

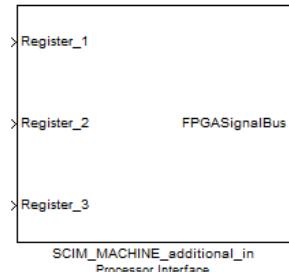


Figure 183: SCIM_MACHINE_additional_in block

Block Dialog	The dialog only provides a short block description.
---------------------	-----------------------------------------------------

Input	The processor input block has the following inputs:
--------------	-----------------------------------------------------

Register 1

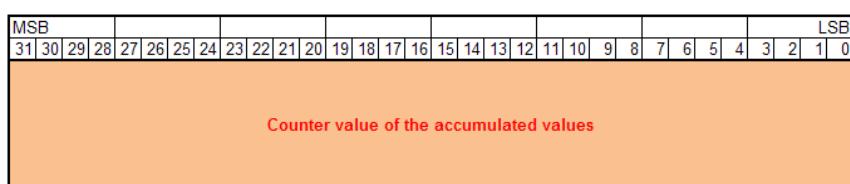


Figure 184: SCIM_MACHINE_additional_in Register 1

Name	Used Bits	Description	Range
Accu	31 ... 0	Counter value of the accumulated values; if an overrun is detected the value $2^{32}-1$ will be sent	0 ... $2^{32}-1$

Register 2

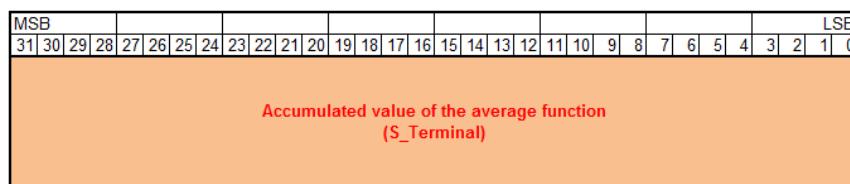


Figure 185: SCIM_MACHINE_additional_in Register 2

Name	Used Bits	Description	Range
Accu	31 ... 0	Accumulated value of the average function; if an overrun is detected the value 2^31-1 will be sent Signal: input power consumption S_Terminal [VA]	0 ... 2^32-1

Register 3

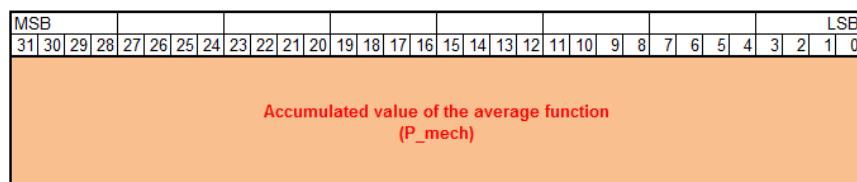


Figure 186: SCIM_MACHINE_additional_in Register 3

Name	Used Bits	Description	Range
Accu	31 ... 0	Accumulated value of the average function; if an overrun is detected the value 2^31-1 will be sent Signal: mechanical output power P_mech [W]	0 ... 2^32-1

Output

The processor input block has the following outputs:

Name	Unit	Description	Range
FPGASignalBus	Bus	Bus containing the most significant signals of this block	-

Interface Examples

Processor blocks

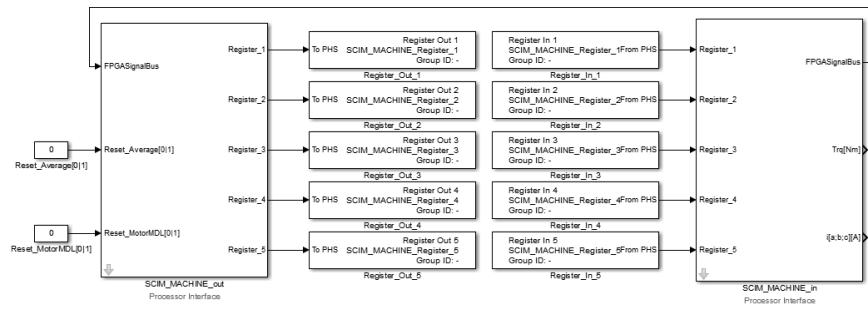


Figure 187: Processor interface

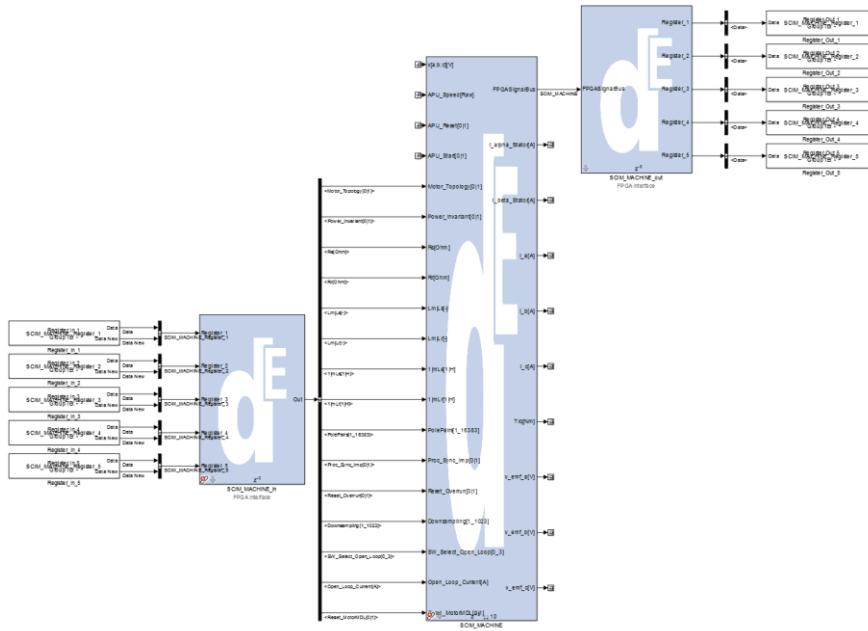
FPGA block

Figure 188: FPGA interface

Electric Machines: Squirrel Cage Induction Machine DQ (SCIM_MACHINE_DQ)

Objective

In this model a squirrel cage induction machine is integrated. Next to the parameterization of the motor specific parameters, the motor can also be set into stimulus mode, too just set one of the phase currents (a, b, c) to a specific value (to check if the current scaling fits to the target ECU). The motor model also calculates the actual processor synchronous average values for the currents and the torque, where a downsampling might be useful to cover as well slow processor clock rates, but keep in mind that the resolution might decrease. An additional reset functionality of the average function and the motor integrators is also implemented. Often the target ECU control the motors by the field orientated control algorithm, therefore the SCIM is modeled inside the dq frame. Due to a division and more calculations on the FPGA more chip resources and RAM of the build PC will be bounded. This block is runs with a downsampling factor of 20, this means every 20th clock a new set of data will be available on the blocks output (FPGA).

	If the standard interface of the XSG Electric Library is used, make sure that the SCIM_MACHINE_DQ_in and SCIM_MACHINE_DQ_out block of the processor interface is located in the same sampled task. This ensures that the timing behavior of the included average functionality of the FPGA main block is correct. Infractions of this requirement produce fail behavior in value conversion and calculation on the processor interface!
	If the machine configuration is set to delta, the maximum voltage input voltage of the motor is reduced to $1024V / \sqrt{3}$.

All motor parameters can be adjusted (during runtime) in the processor output block dialog. The squirrel cage asynchronous machine is modeled using field orientated coordinates (dq frame). The next figure shows the equivalent circuit diagram of the model.

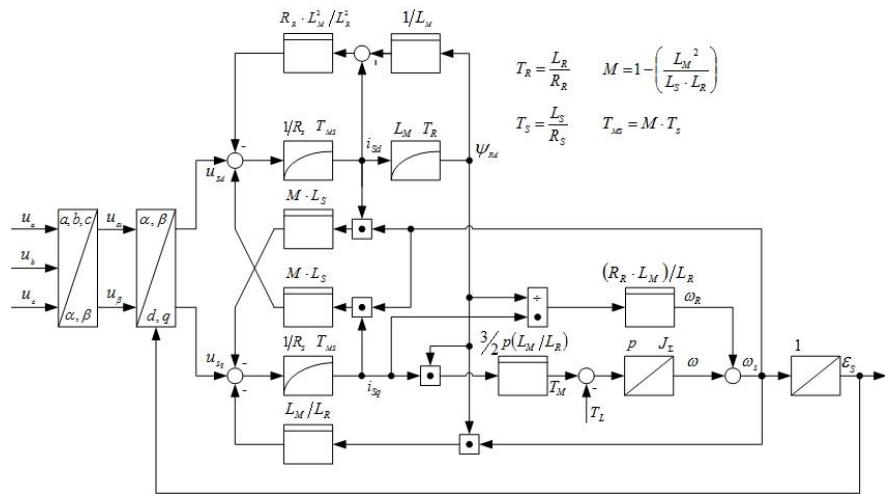


Figure 189: Circuit diagram of the SCIM_MACHINE_DQ model

The following equations describe the motor model mathematically inside state space.

Calculation of stator current / magnetic flux in stator reference frame:

$$\begin{aligned}\frac{di_{Sd}}{dt} &= -\left(\frac{1}{M \cdot T_S} + \frac{1-M}{M \cdot T_R}\right)i_{Sd} + \omega_s \cdot i_{Sq} + \frac{1-M}{M \cdot T_R} \cdot \psi'_{Rd} + \frac{1-M}{M} \cdot \omega_R \cdot \psi'_{Rq} + \frac{1}{M \cdot L_S} \cdot u_{Sd} \\ \frac{di_{Sq}}{dt} &= -\left(\frac{1}{M \cdot T_S} + \frac{1-M}{M \cdot T_R}\right)i_{Sq} + \omega_s \cdot i_{Sd} + \frac{1-M}{M \cdot T_R} \cdot \psi'_{Rq} - \frac{1-M}{M} \cdot \omega_R \cdot \psi'_{Rd} + \frac{1}{M \cdot L_S} \cdot u_{Sq}\end{aligned}$$

$$\begin{aligned}\frac{d\psi'_{Rd}}{dt} &= \frac{1}{T_R} i_{Sd} - \frac{1}{T_R} \cdot \psi'_{Rd} + (\omega_s - \omega_R) \cdot \psi'_{Rq} \\ \frac{d\psi'_{Rq}}{dt} &= \frac{1}{T_R} i_{Sq} - \frac{1}{T_R} \cdot \psi'_{Rq} - (\omega_s - \omega_R) \cdot \psi'_{Rd}\end{aligned}$$

The actual torque of the motor is calculated with:

$$Trq = \frac{3}{2} p [(1-M) \cdot L_S \cdot \psi'_{Rd} \cdot i_{Sq}]$$

Legend of formula symbols:

ψ'_{Rd}	Rotor magnetic flux in direct axis
ψ'_{Rq}	Rotor magnetic flux in quadrature axis
ω_s	Stator electrical angular speed
ω_R	Rotor electrical angular speed
R_s	Stator winding resistance
R_R	Rotor winding resistance
L_s	Stator inductance
L_R	Rotor inductance

$$\begin{array}{ll} L_m & \text{Mutal inductance} \\ P & \text{Number of pole pairs} \end{array}$$

The blockset contains the following elements:

- Processor Interface: SCIM_MACHINE_DQ_out
(Processor Interface)
- FPGA Interface: SCIM_MACHINE_DQ_in
(FPGA Interface)
- FPGA: SCIM_MACHINE_DQ
(FPGA Main Component)
- FPGA Interface: SCIM_MACHINE_DQ_out
(FPGA Interface)
- Processor Interface: SCIM_MACHINE_DQ_in
(Processor Interface)

Processor Output

Block

Merges the model parameterization signals and writes them to the FPGA

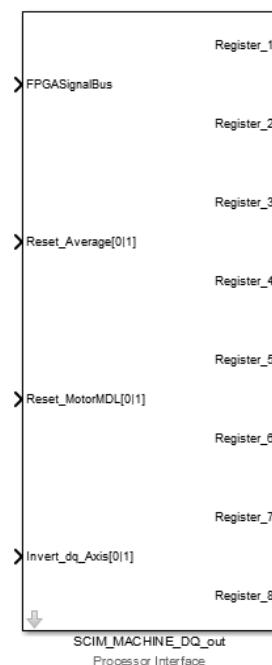


Figure 190: SCIM_MACHINE_DQ_out block

Block Dialog

The processor output block contains several dialogs. All general parameters of the block can be configured in the page "General".

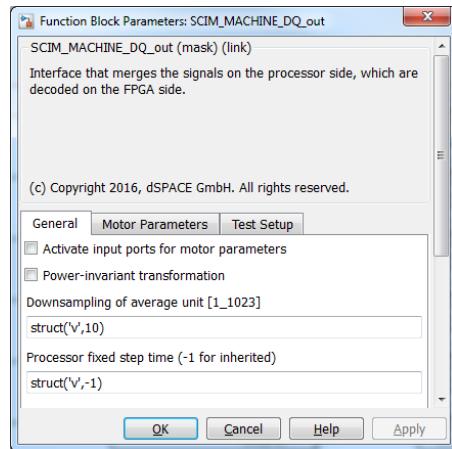


Figure 191: SCIM_MACHINE_DQ_out dialog; page “General”

The page has the following parameters:

Name	Unit	Description	Range
Activate input ports	[-]	Activate input ports for the inductance, magnetic flux and stator resistance	-
Power-invariant transformation	[-]	Activate power invariant transformation for clark transformation	-
Downsampling of average unit	[-]	Downsampling factor of the average functionality in the range 1...1023	1 ... 1023; resolution: 1
Processor fixed step time	[s]	Step time of the processor task. For offline simulation a different sample time can be set to simulate the processor interface communication behavior.	-

The motor specific parameters can be configured on the page “Motor Parameter” as show in the figure below.

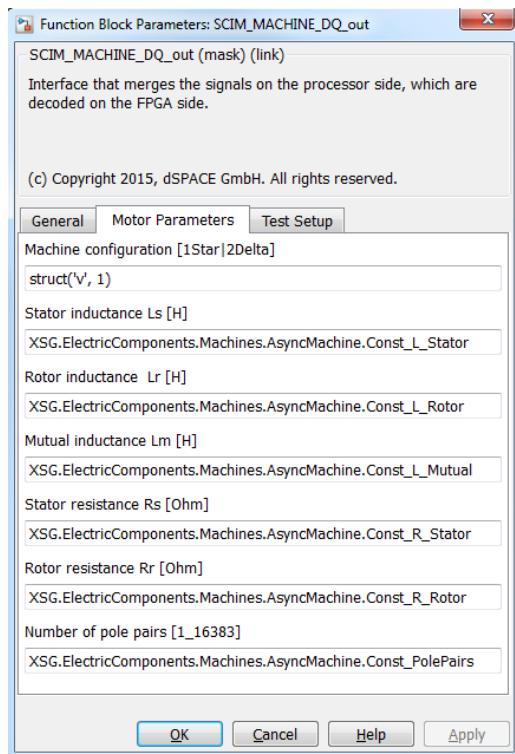


Figure 192: SCIM_MACHINE_DQ_out dialog; page “Motor Parameters”

The page has the following parameters:

Name	Unit	Description	Range
Machine configuration	[-]	Select the machine configuration; 1: star 2: delta	-
Stator inductance Ls	[H]	Stator inductance	0 ... 1H; resolution: 953E-9
Stator resistance Rs	[Ohm]	Stator resistance	122E-6 ... 512 Ohm; resolution: 122E-6
Rotor inductance Lr	[H]	Rotor inductance	3.81E-6 ... 32 H; resolution: 3.81E-6
Rotor resistance Rr	[Ohm]	Rotor resistance	3.81E-6 ... 16 Ohm; resolution: 3.81E-6
Mutal inductance Lm	[H]	Mutal inductance	0 ... 1H; resolution: 29.8E-9
Number of pole pairs	[-]	Number of motor pole pairs	0 ... 16383; resolution: 1
OperationMode	[-]	Selection of the Operation Mode of the motor model; 1: Open Loop test 0: Closed Loop test	0 1
Open Loop Current Selector	[-]	Selector which current is stimulated if the operation mode is set to open loop test; 1: i_a 2: i_b 3: i_c 4: i_d_Stator 5: i_q_Stator	1 2 3 Resolution: 1
Open Loop Current	[A]	Value for the stimulated open loop current	-1024 ... 1024 A Resolution: 0.0078

Specific settings for testing the outputs of the machine can be configured on the page “Test Setup”.

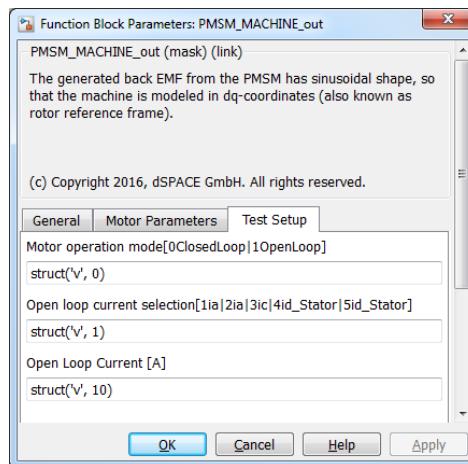


Figure 193: PMSM_MACHINE_out dialog; page “Test Setup”

The page has the following parameters:

Name	Unit	Description	Range
OperationMode	[-]	Selection of the Operation Mode of the motor model; 1: Open Loop test 0: Closed Loop test	0 1
Open Loop Current Selector	[-]	Selector which current is stimulated if the operation mode is set to open loop test; 1: i_a 2: i_b 3: i_c 4: i_d_Stator 5: i_q_Stator	-
Open Loop Current	[A]	Value for the stimulated open loop current	A

Input

The SCIM_MACHINE_DQ_out block has the following inputs. If input ports for the motor parameters are activated via the GUI checkbox, additional ports for the relevant motor parameters are added.

Name	Unit	Description	Range
FPGASignalBus	Bus	Bus containing the most significant feedback signals from the processor input block	-
Reset_Average	[-]	Reset function of the implemented FPGA average functionality	0 1
Reset_Motor MDL	[-]	Reset function of the motor model on FPGA	0 1
Invert_dq_Axis	[0 1]	Adds an offset of 180 deg to the electrical transformation angle to invert dq currents from negative to positive. Only supported when slip and mechanical speed is 0!	0 1

Output

The processor out block provides the following input registers whose sectioning is shown below:

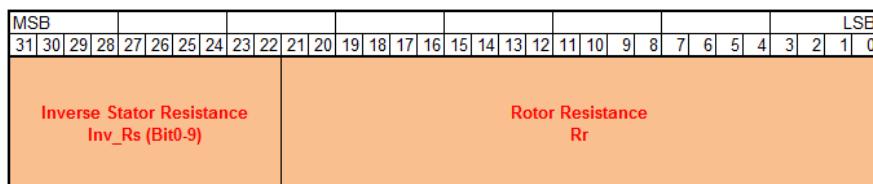
Register 1

Figure 194: SCIM_MACHINE_DQ_out Register 1

Name	Used Bits	Description	Range
Rotor Resistance Rr	21 ... 0	Rotor resistance Rr	0 ... 2^21-1 represents 3.81E-6 ... 16 Ohm
Inverse stator Resistance Rs	31 ... 22	Inverse stator resistance 1/Rs Part 1	0 ... 8192

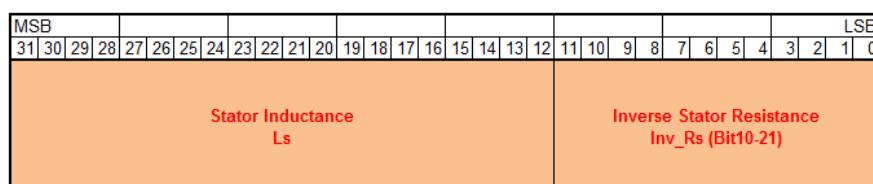
Register 2

Figure 195: SCIM_MACHINE_DQ_out Register 2

Name	Used Bits	Description	Range
Inverse stator Resistance Rs	11 ... 0	Inverse stator resistance 1/Rs Part 2	0 ... 8192
Stator inductance Ls	31 ... 12	Calculated rotor factor Lm Ls	0 ... 2^20-1 represents 0 ... 1

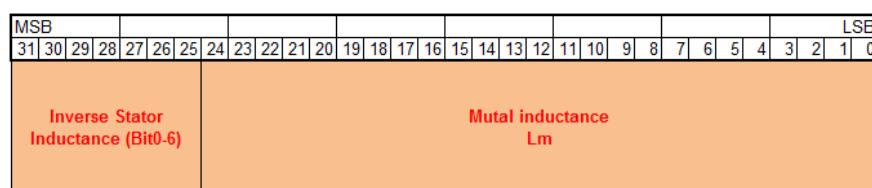
Register 3

Figure 196: SCIM_MACHINE_DQ_out Register 3

Name	Used Bits	Description	Range
Mutal inductance Lm	24 ... 0	Mutal inductance	0 ... 2^25-1 represents 0 ... 1 H
Inverse stator inductance 1/Ls	31 ... 25	Inverse stator inductance 1/Ls Part 1	0 ... 2^7

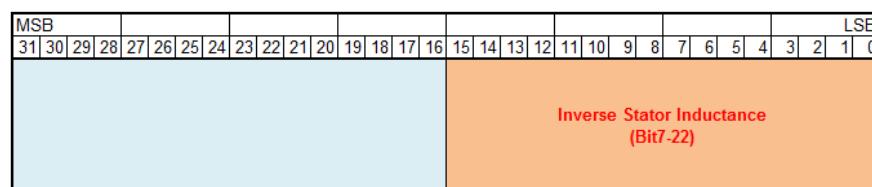
Register 4

Figure 197: SCIM_MACHINE_DQ_out Register 4

Name	Used Bits	Description	Range
Inverse stator inductance 1/Ls	15 ... 0	Inverse stator inductance 1/Ls Part 2	0 ... 2^15-1
SPARE	31 ... 16		

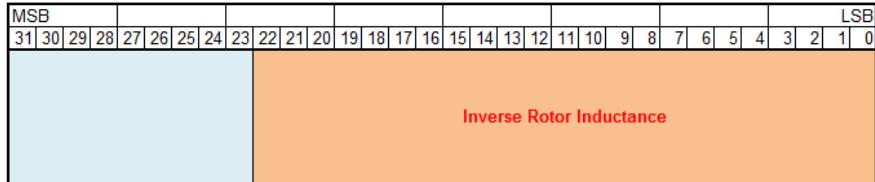
Register 5

Figure 198: SCIM_MACHINE_DQ_out Register 5

Name	Used Bits	Description	Range
Inverse rotor inductance 1/Lr	22 ... 0	Inverse rotor inductance	0 ... 2^22-1
SPARE	31 ... 23		

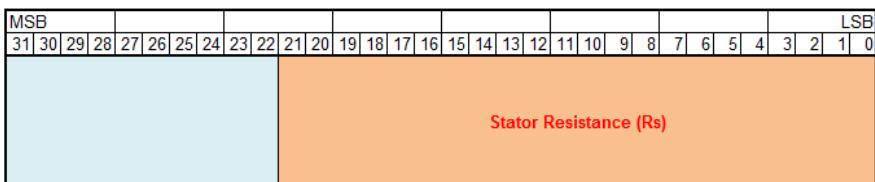
Register 6

Figure 199: SCIM_MACHINE_DQ_out Register 6

Name	Used Bits	Description	Range
Stator Resistance	21 ... 0	Stator Resistance (rs)	0 ... 16
SPARE	31 ... 22		

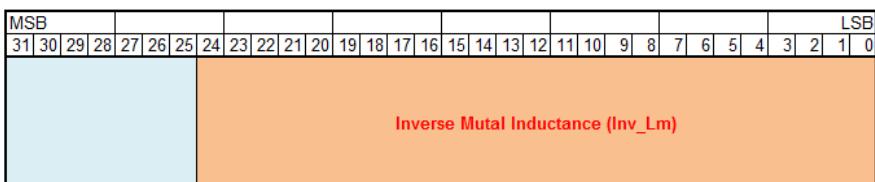
Register 7

Figure 200: SCIM_MACHINE_DQ_out Register 7

Name	Used Bits	Description	Range
Inverse Mutual Inductance	24 ... 0	Inverse value of mutual inductance (Lm)	0 ... 2^18-1
SPARE	31 ... 25		

Register 8; ID = 0

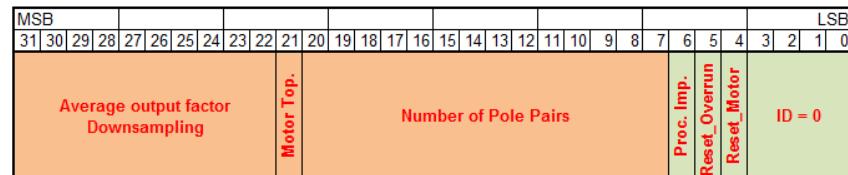


Figure 201: PMSM_MACHINE_out Register 8; ID = 0

Name	Used Bits	Description	Range
Counter ID	3 ... 0	ID to select which register coding is activated via Processor	0 ... 1
Reset_Motor MDL	4	Reset flag for the motor model on the FPGA	0 1
Reset_Overrun	5	Reset flag to reset an overrun of the average functionality	0 1
Proc. Sync. Impulse	6	Processor synchronous toggle bit	0 1
Number of Pole Pairs	13 ... 0	Number of pole pairs	1 ... 16363
Motor Topology	30	Specify the motor topology	0 1
Average output factor Down-sampling	9 ... 0	Downsampling factor for average functionality of the outputs to the processor interface	1 ... 1023

Register 8; ID = 1

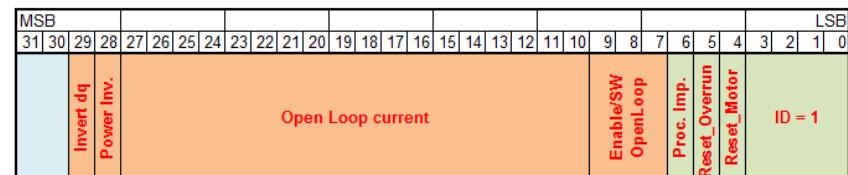


Figure 202: PMSM_MACHINE_out Register 8; ID = 1

Name	Used Bits	Description	Range
Continuous Data	6 ... 0	Definition same as ID = 0	
Enable/SW Open Loop	9 ... 7	Switch and Enable of the open loop stimulation	0 ... 5
OpenLoop Current	27 ... 10	Open Loop current for open loop stimulation	-1024 ... 1024
Power Invariant	28	Switch to activate power invariant transformation	0 1
Invert dq	29	Switch to invert d and q axis	0 1
SPARE	31 ... 30		

FPGA Main Component

Block



Figure 203: SCIM_MACHINE_DQ FPGA Main block

Block Dialog

The FPGA main blockset contains the following dialog.

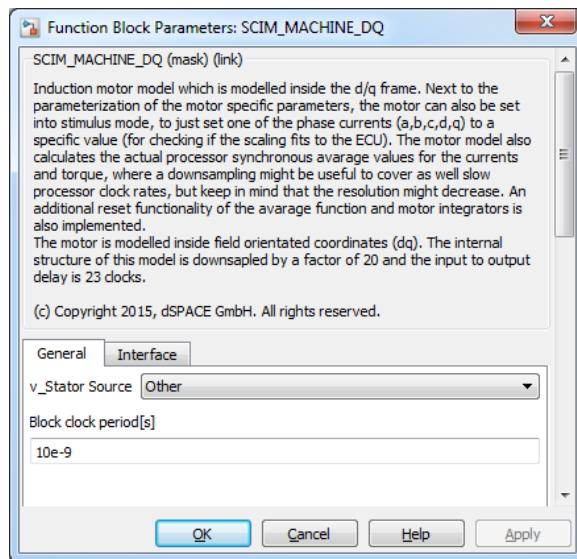


Figure 204: SCIM_MACHINE_DQ FPGA Main block dialog

On the page General the connected stator voltage can be selected. Please choose the connected inverter model via the popup.

On the page Interface you can define the parameters of the input and output registers for automatic interface generation. The dialog also contains a button to start interface generation. For more details about automatic interface generation, refer to the XSG Utils documentation chapter Automatic Interface Generation. The FPGA clock period can be configured too.

Input

The main block has the following inputs:

Name	Unit	Description	Format
v	[V] (vector)	Stator voltage vector of the motor phase	Fix_18_7
APU_Speed	[Raw]	Actual speed	Fix_30_0
APU_Reset	[‐]	Reset on high state	Bool
APU_Start	[‐]	Start on high state	UFix_1_0
Motor_Topology	[‐]	Specify the motor topology	UFix_1_0
Power_Invariant	[‐]	Specify the internal clark transformation	UFix_1_0
Rr	[Ohm]	Rotor resistance	UFix_22_18
Rs	[Ohm]	Stator resistance	UFix_22_18
Inv_Rs	[S]	Inverse stator resistance	UFix_22_9
Ls	[H]	Stator inductance	UFix_20_20
Inv_Ls	[1 H]	Inverse stator inductance	UFix_23_5
Inv_Lr	[1 H]	Inverse rotor inductance	UFix_23_5
Lm	[H]	Mutal inductance	UFix_25_25
PolePairs	[‐]	Pole pairs of the motor	UFix_14_0
Invert_dq_Axis	[0 1]	Negate d&q currents	UFix_1_0
Proc_Sync_Imp	[‐]	Processor synchronize toggle bit	UFix_1_0
Reset_OVERRUN	[‐]	Reset flag to reset an overrun of the average functionality	Bool
Downsampling	[‐]	Downsampling factor for average functionality of the outputs to the processor interface	UFix_10_0
SW_Select_Open_Loop	[‐]	Switch and Selector for the Open Loop Control	UFix_4_0
Open_Loop_Current	[A]	Open Loop current	UFix_18_7
Reset_Motor_MDL	[‐]	Reset flag to the motor model	Bool

Output

The SCIM_MACHINE_DQ main block has the following outputs:

Name	Unit	Description	Format
FPGASignalBus	Bus	Bus containing the most significant signals of this block	-
Trq	[Nm]	Generated motor torque	Fix_25_14
i_d_Stator	[A]	Current in phase d axis	Fix_18_7
i_q_Stator	[A]	Current in phase q axis	Fix_18_7
i_a	[A]	Current in phase a	Fix_18_7
i_b	[A]	Current in phase b	Fix_18_7
i_c	[A]	Current in phase c	Fix_18_7
Psi_d_Rotor	[Wb]	Magnetic flux in d axis	Fix_25_22
v_emf_a	[V]	Back EMF Voltage in phase a	Fix_18_7
v_emf_b	[V]	Back EMF Voltage in phase b	Fix_18_7
v_emf_c	[V]	Back EMF Voltage in phase c	Fix_18_7

Processor Input

Block

Adapts the FPGA signals for the processor side.



Figure 205: SCIM_MACHINE_DQ_in block

Block Dialog

The dialog only provides a short block description.

Input

The processor input block has the following inputs:

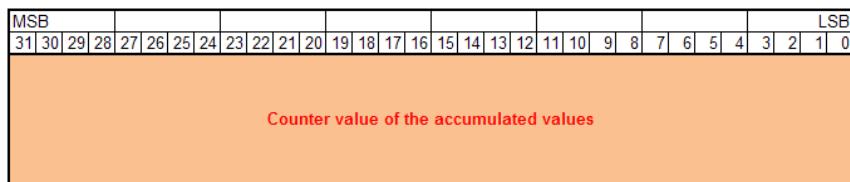
Register 1

Figure 206: SCIM_MACHINE_DQ_in Register 1

Name	Used Bits	Description	Range
Accu	31 ... 0	Counter value of the accumulated values; if an overrun is detected the value $2^{32}-1$ will be sent	0 ... $2^{32}-1$

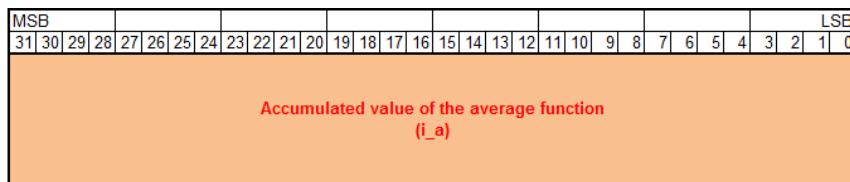
Register 2

Figure 207: SCIM_MACHINE_DQ_in Register 2

Name	Used Bits	Description	Range
Accu	31 ... 0	Accumulated value of the average function; if an overrun is detected the value $2^{31}-1$ will be sent Signal: i_a	0 ... $2^{32}-1$

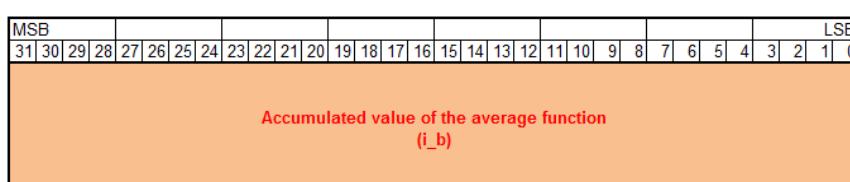
Register 3

Figure 208: SCIM_MACHINE_DQ_in Register 3

Name	Used Bits	Description	Range
Accu	31 ... 0	Accumulated value of the average function; if an overrun is detected the value 2^31-1 will be sent Signal: i_b	0 ... 2^32-1

Register 4

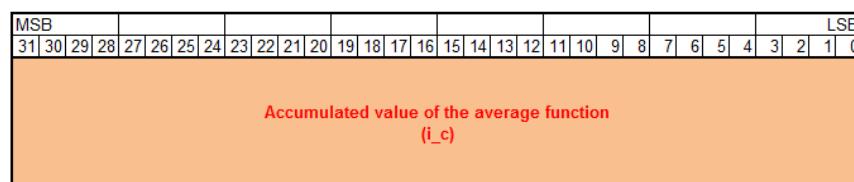


Figure 209: SCIM_MACHINE_DQ_in Register 4

Name	Used Bits	Description	Range
Accu counter	31 ... 0	Accumulated value of the average function; if an overrun is detected the value 2^31-1 will be sent Signal: i_c	0 ... 2^32-1

Register 5

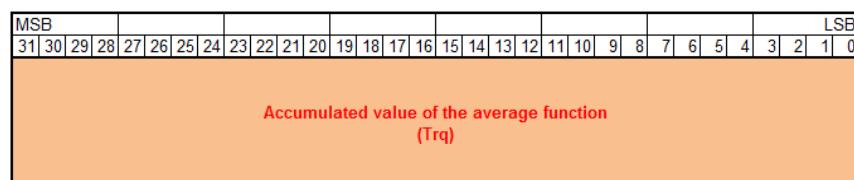


Figure 210: SCIM_MACHINE_DQ_in Register 5

Name	Used Bits	Description	Range
Accu	31 ... 0	Accumulated value of the average function; if an overrun is detected the value 2^31-1 will be sent Signal: Trq	0 ... 2^32-1

Register 6

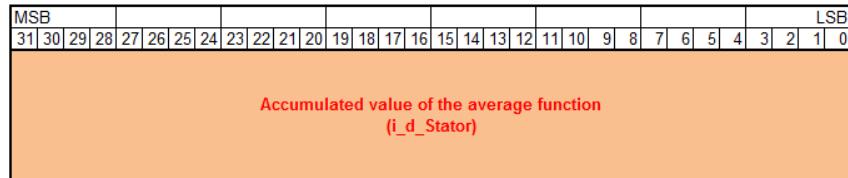


Figure 211: SCIM_MACHINE_DQ_in Register 6

Name	Used Bits	Description	Range
Accu	31 ... 0	Accumulated value of the average function; if an overrun is detected the value 2^31-1 will be sent Signal: i_d_Stator	0 ... 2^32-1

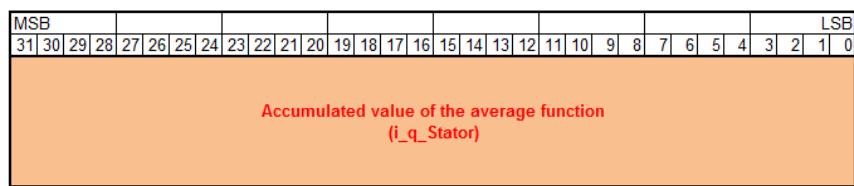
Register 7

Figure 212: SCIM_MACHINE_DQ_in Register 7

Name	Used Bits	Description	Range
Accu	31 ... 0	Accumulated value of the average function; if an overrun is detected the value 2^31-1 will be sent Signal: i_q_Stator	0 ... 2^32-1

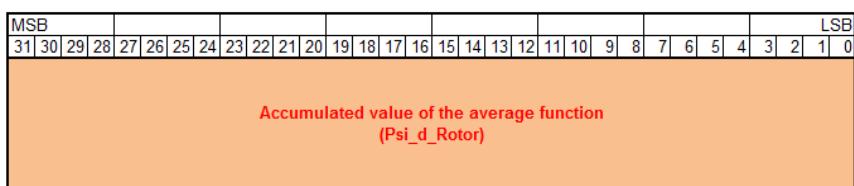
Register 8

Figure 213: SCIM_MACHINE_DQ_in Register 8

Name	Used Bits	Description	Range
Accu	31 ... 0	Accumulated value of the average function; if an overrun is detected the value 2^31-1 will be sent Signal: Psi_d_Rotor	0 ... 2^32-1

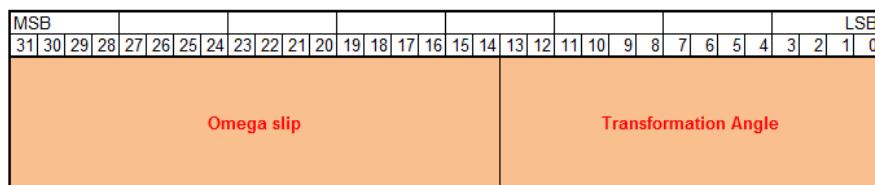
Register 9

Figure 214: SCIM_MACHINE_DQ_in Register 9

Name	Used Bits	Description	Range
Transformation Angle	13 ... 0	Transformation angle for dq transformation	0 ... 2^14-1
Omega slip	31...14	Slip speed	0 ... 2^18-1

Output

The processor input block has the following outputs:

Name	Unit	Description	Range
FPGASignalBus	Bus	Bus containing the most significant signals of this block	-
Trq	[Nm]	Torque of the motor	-1024 ... 1023.99 Nm
i	[A]Bus	Phase current in each motor phase (a,b,c)	-1024 ... 1023.99 A
i_d_Stator	[A]	Current in direct axis	-1024 ... 123.99 A
i_q_Stator	[A]	Current in quadrature axis	-1024 ... 123.99 A
Psi_d_Rotor	[Wb]	Magnetic flux in direct axis	-4 ... 3.99 Wb
Transformation Angle	[deg]	Transformation Angle for dq transformation	0 ... 359,99 deg
Omega Slip el	[rpm]	Electrical slip speed	-4096 ... 4095.97 rpm

Block additional feedback

Adapts the FPGA signals for additional feedback of the FPGA to the processor side (like power consumption,...).

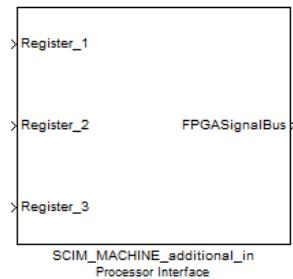


Figure 215: SCIM_MACHINE_additional_in block

Block Dialog

The dialog only provides a short block description.

Input

The processor input block has the following inputs:

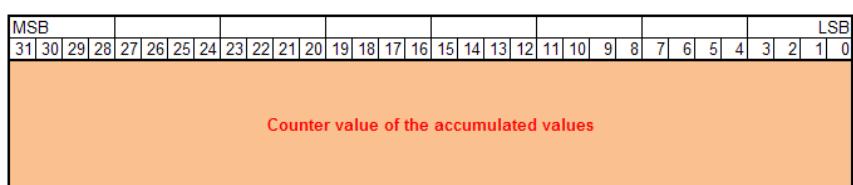
Register 1

Figure 216: SCIM_MACHINE_additional_in Register 1

Name	Used Bits	Description	Range
Accu	31 ... 0	Counter value of the accumulated values; if an overrun is detected the value $2^{32}-1$ will be sent	0 ... $2^{32}-1$

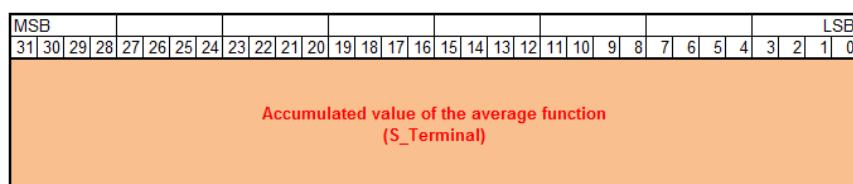
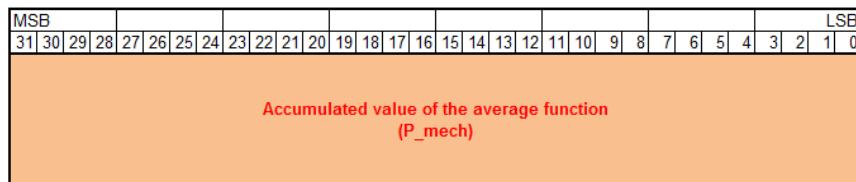
Register 2

Figure 217: SCIM_MACHINE_additional_in Register 2

Name	Used Bits	Description	Range
Accu	31 ... 0	Accumulated value of the average function; if an overrun is detected the value $2^{31}-1$ will be sent Signal: input power consumption S_Terminal [VA]	0 ... $2^{32}-1$

Register 3**Figure 218: SCIM_MACHINE_additional _in Register 3**

Name	Used Bits	Description	Range
Accu	31 ... 0	Accumulated value of the average function; if an overrun is detected the value 2^31-1 will be sent Signal: mechanical output power P_mech [W]	0 ... 2^32-1

Output

The processor input block has the following outputs:

Name	Unit	Description	Range
FPGASignalBus	Bus	Bus containing the most significant signals of this block	-

Interface Examples

Processor blocks

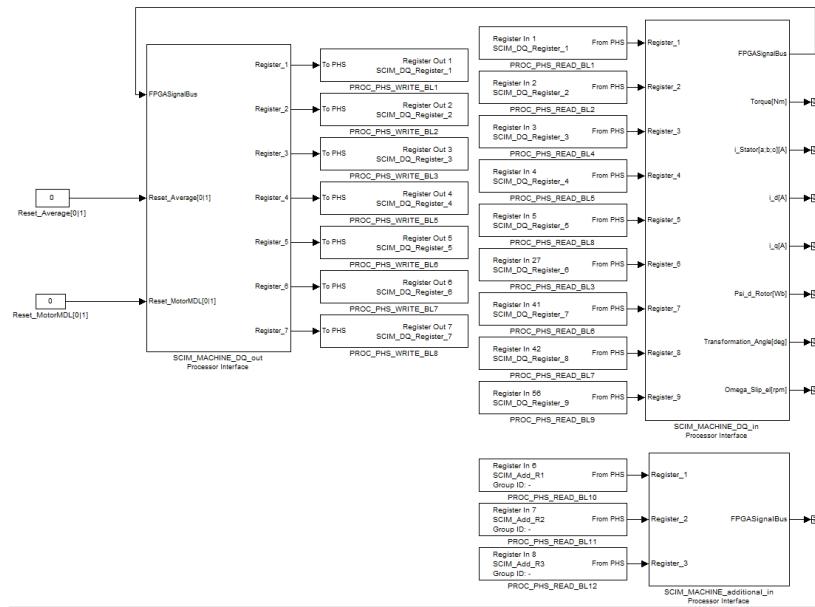


Figure 219: Processor interface

FPGA block

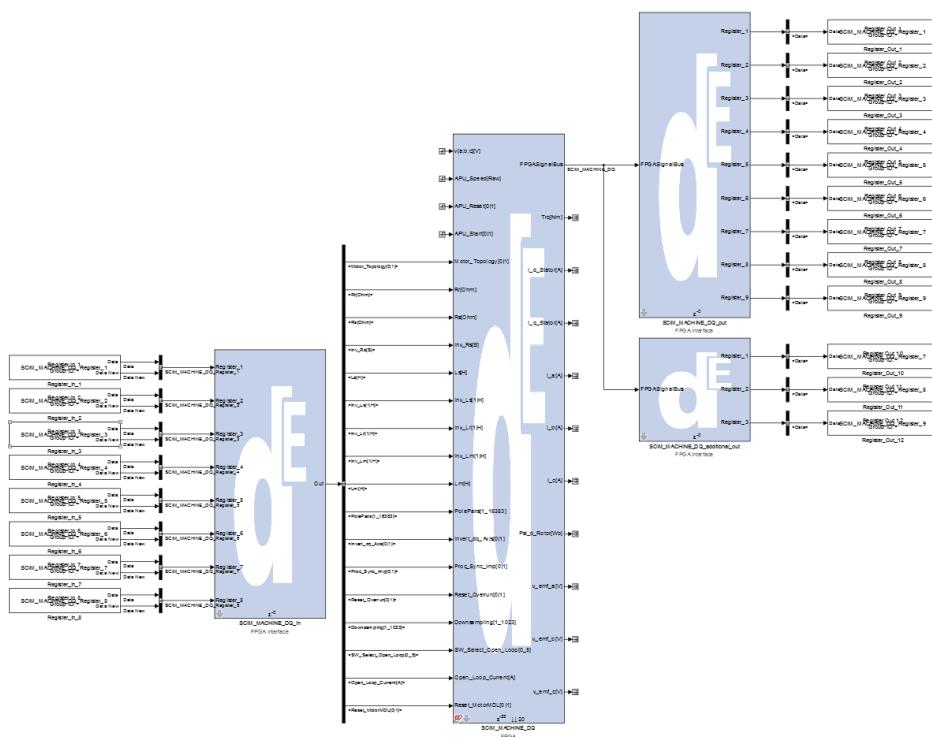


Figure 220: FPGA interface

Electric Machines: Separately Exited DC machine

Objective

Next to the parameterization of the motor specific parameters, the motor can also be set into open loop mode, too just set one of the currents (armature, field) to a specific value (to check if the current scaling fits to the ECU). The motor model also calculates the actual processor synchronous average values for the currents and the torque, where a downsampling might be useful to cover as well slow processor clock rates. But keep in mind that the resolution might decrease. An additional reset functionality of the average function and the motor integrators is also implemented.



If the standard interface of the XSG Electric Library is used, make sure that the SEPARATELY_EXCITED_DC_MACHINE_in and SEPARATELY_EXCITED_DC_MACHINE_out block of the processor interface is located in the same sampled task. This ensures that the timing behavior of the included average functionality of the FPGA main block is correct. **Infractions of this requirement produce fail behavior in value conversion and calculation on the processor interface!**

All motor parameters can be adjusted in the processor output block dialog.

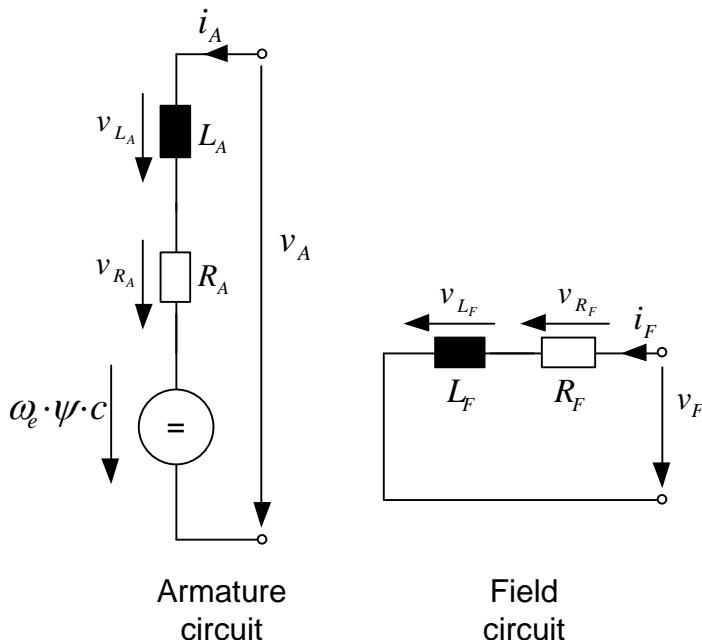


Figure 221: Equivalent circuit dc machine

The blockset contains the following elements:

- Processor Interface: SEPARATELY_EXCITED_DC_MACHINE_out (Processor Interface)
- FPGA Interface: SEPARATELY_EXCITED_DC_MACHINE_in (FPGA Interface)

- FPGA: SEPARATLY_EXCITED_DC_MACHINE
(FPGA Main Component)
- FPGA Interface: SEPARATLY_EXCITED_DC_MACHINE_out
(FPGA Interface)
- Processor Interface: SEPARATLY_EXCITED_DC_MACHINE_in
(Processor Interface)

Processor Output

Block

Merges the processor signals and writes them to the FPGA

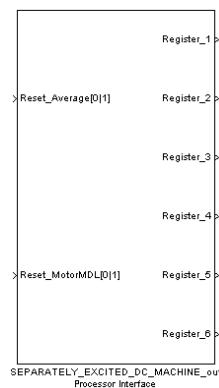


Figure 222: SEPARATLY_EXCITED_DC_MACHINE_out block

Block Dialog

The processor output blockset contains the following dialog.

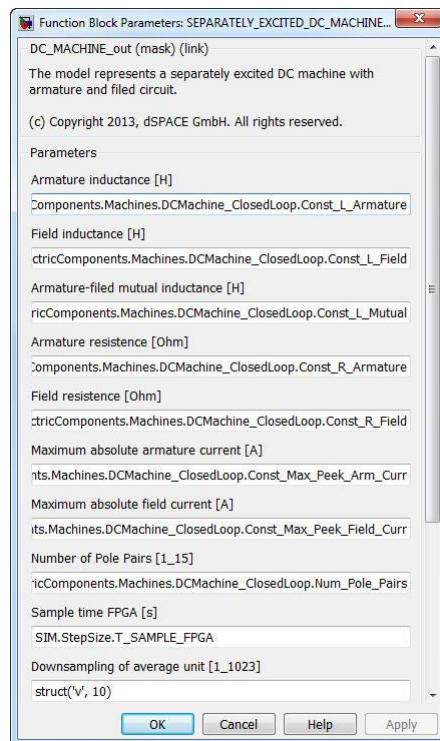


Figure 223: SEPARATELY_EXCITED_DC_MACHINE_out dialog

The dialog blockset has the following parameters:

Name	Unit	Description	Range
Armature inductance	[H]	Armature inductance	-
Field inductance	[H]	Field inductance	-
Armature-field mutual inductance	[H]	Armature-field mutual inductance	0 ... 16 H; resolution: 244E-6
Armature resistance	[Ohm]	Armature resistance	244E-6 ... 16 Ohm; resolution: 244E-6
Field resistance	[-]	Field resistance	244E-6 ... 16 Ohm; resolution: 244E-6
Maximum absolute armature current	[A]	Maximal current of the armature (prepared for external scaling in next releases)	1 ... 16384 A; resolution: 1
Maximum absolute field current	[A]	Maximal current of the field (prepared for external scaling in next releases)	1 ... 16384 A; resolution: 1
Number of Pole Pairs	[-]	Number of pole pairs	1 ... 15; resolution: 1
Sample time FPGA	[s]	Sample time of the FPGA	-
Downsampling of average unit	[-]	Downsampling factor of the average functionality	1 ... 1023 resolution: 1
Processor fixed step time	[s]	Step time of the processor task. For offline simulation a different sample time can be set to simulate the processor interface communication behavior.	-
OperationMode	[-]	Selection of the Operation Mode of the motor model; 1: Open Loop test 0: Closed Loop test	0 1
Open Loop Current Selector	[-]	Selector which current is stimulated if the operation mode is set to open loop test; 1: i_field 2: i_armature	-
Open Loop Current	[A]	Value for the stimulated open loop current	A

Input

The SEPARATLY_EXCITED_DC_MACHINE _out block has the following inputs:

Name	Unit	Description	Range
Reset_Average	[-]	Reset function of the implemented FPGA average functionality	0 1
Reset_Motor MDL	[-]	Reset function of the motor model on FPGA	0 1

Output

The processor out block provides six input registers whose sectioning is shown below:

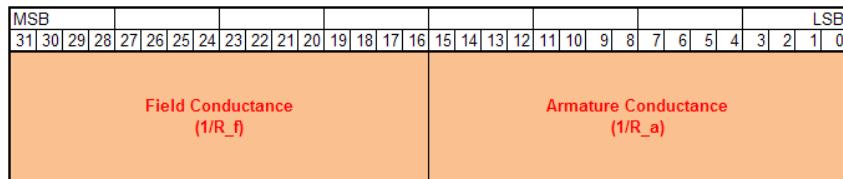
Register 1

Figure 224: SEPARATLY_EXCITED_DC_MACHINE_out Register 1

Name	Used Bits	Description	Range
Armature Conductance Ra_inv	15 ... 0	Armature conductance of a phase in [S]	0 ... 2^16-1 represents 16 ... 250E-6 Ohm
Field Conductance Ra_inv	31 ... 16	Field conductance of a phase in [S]	0 ... 2^16-1 represents 16 ... 250E-6 Ohm

Register 2

Figure 225: SEPARATLY_EXCITED_DC_MACHINE_out Register 2

Name	Used Bits	Description	Range
Mutual inductance Laf	15 ... 0	Armature-field mutual inductance represents 0 ... 1 H	0 ... 2^16-1
Max Peak Current Field	22 ... 16	Maximum peak current for field winding	0 ... 2^7
Max Peak Current Armature	29 ... 13	Maximum peak current for armature winding	0 ... 2^7
SPARE	30		
Proc.Sync. Impulse	31	Processor synchronous toggle bit	0 1

Register 3

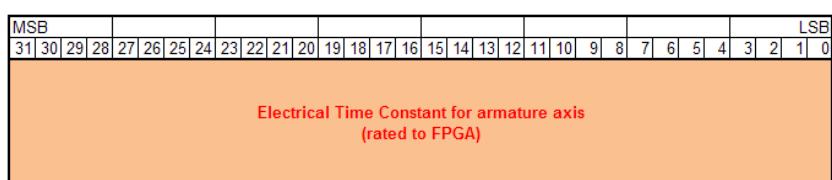


Figure 226: SEPARATLY_EXCITED_DC_MACHINE_out Register 3

Name	Used Bits	Description	Range
Electric time constant for armature axis	31 ... 0	Electrical motor time constant for armature axis with reference to the FPGA sample time ($T = Ts_FPGA * Ld/R$)	0 ... 2^32-1 represents 0 ... 1s resolution: 232E-12

Register 4

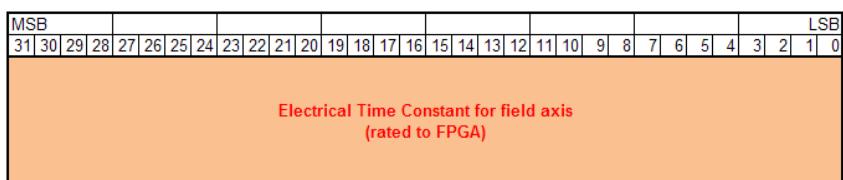


Figure 227: SEPARATLY_EXCITED_DC_MACHINE_out Register 4

Name	Used Bits	Description	Range
Electric time constant for field axis	31 ... 0	Electrical motor time constant for field axis with reference to the FPGA sample time ($T = Ts_FPGA * Ld/R$)	0 ... 2^32-1 represents 0 ... 1s resolution: 232E-12

Register 5

MSB	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	LSB
Number of Pole Pairs	Inverse maximum armature current										Inverse maximum field current																						

Figure 228: SEPARATLY_EXCITED_DC_MACHINE_out Register 5

Name	Used Bits	Description	Range
Inverse maximum field current	13 ... 0	Inverse of the maximal phase current in field winding	0 ... 2^14-1
Inverse maximum armature current	27 ... 14	Inverse of the maximal phase current in armature winding	0 ... 2^14-1
Number of Pole Pairs	31 ... 28	Number of pole pairs	1 ... 15

Register 6

MSB	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	LSB
Reset_Overrun	Open Loop current										Average output factor Downsampling																						
Reset_Motor																																	

Figure 229: SEPARATLY_EXCITED_DC_MACHINE_out Register 6

Name	Used Bits	Description	Range
Average output factor Downsampling	9 ... 0	Downsampling factor for average functionality of the outputs to the processor interface	1 ... 1023
OpenLoop Current	27 ... 10	Open Loop current for open loop stimulation	-1024 ... 1024
Enable/SW Open Loop	29 ... 28	Switch and Enable of the open loop stimulation	0 ... 3
Reset_MotorMDL	30	Reset flag for the motor model on the FPGA	0 1
Reset_Overrun	31	Reset flag to reset an overrun of the average functionality	0 1

FPGA Main Component

Block

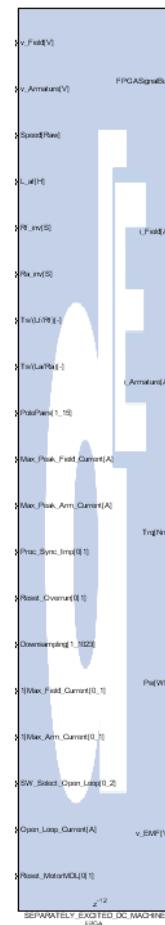


Figure 230: SEPARATLY_EXCITED_DC_MACHINE FPGA Main block

Block Dialog

The FPGA main blockset contains the following dialog.

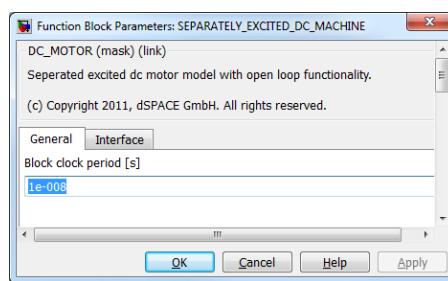


Figure 231: SEPARATLY_EXCITED_DC_MACHINE FPGA Main block dialog

On the page Interface you can define the parameters of the input and output registers for automatic interface generation. The dialog also contains a button to start interface generation. For more details about automatic interface generation, refer to the XSG Utils documentation chapter Automatic Interface Generation. The FPGA clock period can be configured too.

Input

The main block has the following inputs:

Name	Unit	Description	Format
v_Field	[V]	Field voltage of the motor	Fix_18_7
v_Armature	[V]	Armature voltage of the motor	Fix_18_7
APU_Speed	[Raw]	Actual speed	Fix_30_0
Laf	[H]	Armature-field mutual inductance	UFix_16_12
Rf_inv	[S]	Field conductance	UFix_16_8
Ra_inv	[S]	Armature conductance	UFix_16_8
Ts/(Lf/Rf)	[·]	Electrical motor time constant in field axis with reference to the FPGA sample time	UFix_32_32
Ts/(La/Ra)	[·]	Electrical motor time constant in armature axis with reference to the FPGA sample time	UFix_32_32
Pole Pairs	[·]	Number of pole pairs	UFix_4_0
Max peak current field	[A]	Maximal current of the field	UFix_7_0
Max peak current armature	[A]	Maximal current of the armature	UFix_7_0
Proc_Sync_Imp	[0 1]	Processor synchronize toggle bit	UFix_1_0
Reset_OVERRUN	[0 1]	Reset flag to reset an overrun of the average functionality	Bool
Downsampling	[·]	Downsampling factor for average functionality of the outputs to the processor interface	UFix_10_0
1 Max_Field_Current	[1/A]	Maximal field current (prepared for external scaling in next releases)	UFix_14_14
1 Max_Armature_Current	[1/A]	Maximal armature current (prepared for external scaling in next releases)	UFix_14_14
SW_Select_Open_Loop	[·]	Switch and Selector for the Open Loop Control	UFix_3_0
Open_Loop_Current	[A]	Open Loop current	UFix_18_7
Reset_Motor_MDL	[·]	Reset flag to the motor model	Bool

Output

The BLDC_MACHINE main block has the following outputs with a maximal latency of seventeen:

Name	Unit	Description	Format
FPGASignalBus	Bus	Bus containing the block's most significant signals	-
i_Field	[A]	Current in phase a	Fix_25_14
i_Armature	[A]	Current in phase b	Fix_25_14
Trq	[Nm]	Torque of the motor	Fix_25_15
Psi	[Wb]	Magnetic flux	Fix_18_15
v_EMF	[V]	Back EMF Voltage	Fix_18_7

Processor Input

Block

Adapts the FPGA signals for the processor side

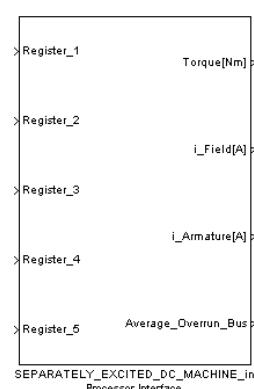


Figure 232: SEPARATELY_EXCITED_DC_MACHINE_in block

Block Dialog

The dialog only provides a short block description.

Input

The processor input block has the following inputs:

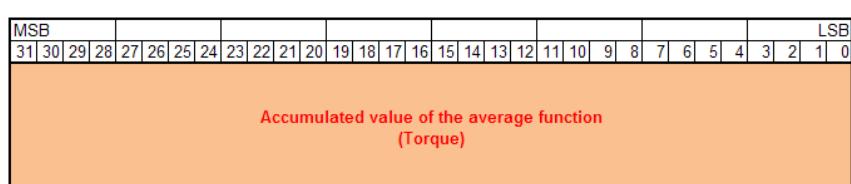
Register 1

Figure 233: SEPARATELY_EXCITED_DC_MACHINE_in Register 1

Name	Used Bits	Description	Range
Accu Torque	31 ... 0	Accumulated value of the average function; Signal: Trq (actual motor torque)	0 ... 2^32-1

Register 2

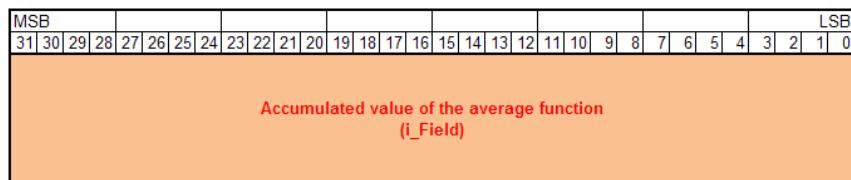


Figure 234: SEPARATLY_EXCITED_DC_MACHINE_in Register 2

Name	Used Bits	Description	Format
Accu i_Field	31 ... 0	Accumulated value of the average function; Signal: i_Field (actual current in field axis)	0 ... 2^32-1

Register 3

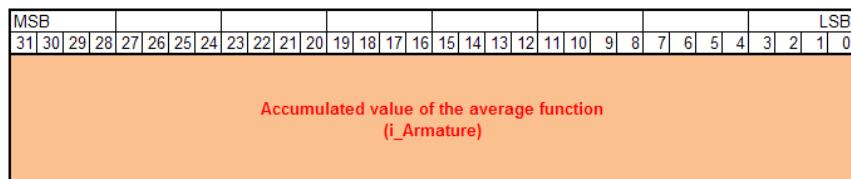


Figure 235: SEPARATLY_EXCITED_DC_MACHINE_in Register 3

Name	Used Bits	Description	Range
Accu i_Armature	31 ... 0	Accumulated value of the average function; Signal: i_Armature (actual current in armature axis)	0 ... 2^32-1

Register 4

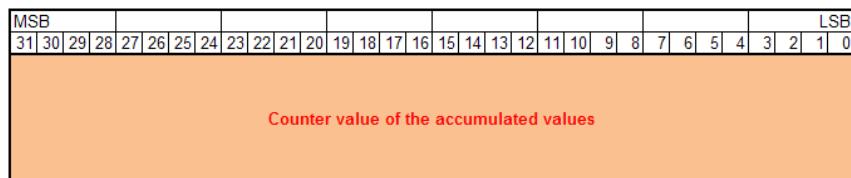
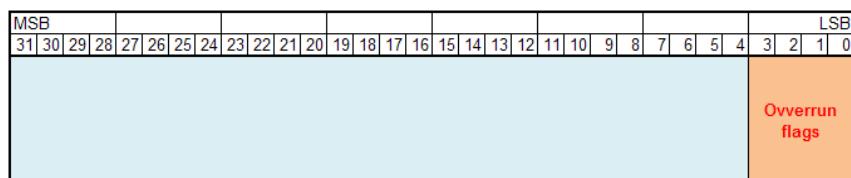


Figure 236: SEPARATLY_EXCITED_DC_MACHINE_in Register 4

Name	Used Bits	Description	Range
Accu counter	31 ... 0	Counter value of the accumulated values	0 ... $2^{32}-1$

Register 5**Figure 237: SEPARATLY_EXCITED_DC_MACHINE_in Register 5**

Name	Used Bits	Description	Range
Overrun flags	3 ... 0	Overrun feedback of the average function Bit 0 = overrun of average function; signal: Trq Bit 1 = overrun of average function; signal: i_Field Bit 2 = overrun of average function; signal: i_Armature Bit 3 = overrun of counter value	0 ... 15
SPARE	31 ... 4		

Output

The processor input block has the following outputs:

Name	Unit	Description	Range
Torque	[Nm]	Torque of the motor	-128 ... 127.99 Nm
i_Field	[A]	Current in field axis	-1024 ... 1023.99 A
I_Armature	[A]	Current in armature axis	-1024 ... 1023.99 A
Average_Overrun_Bus	[Bus]	Overrun feedback from the average function Average_Overrun_Trq = overrun of signal Trq Average_Overrun_i_fFeld= overrun of signal i_Field Average_Overrun_i_Armature= overrun of signal i_Armature	0 1

Interface Examples

Processor blocks

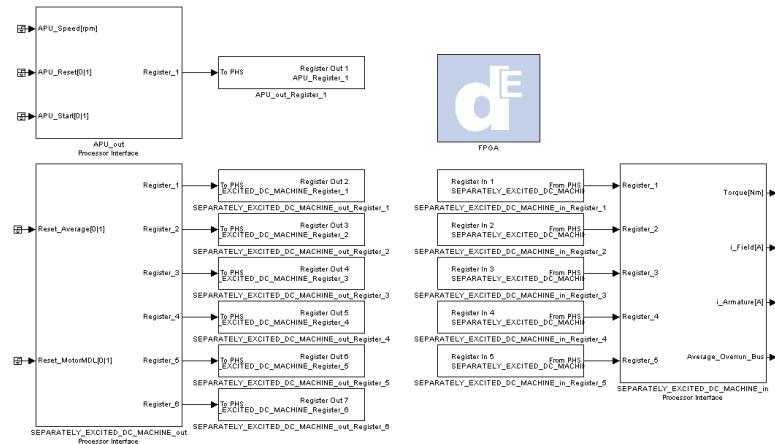


Figure 238: Processor interface

FPGA block

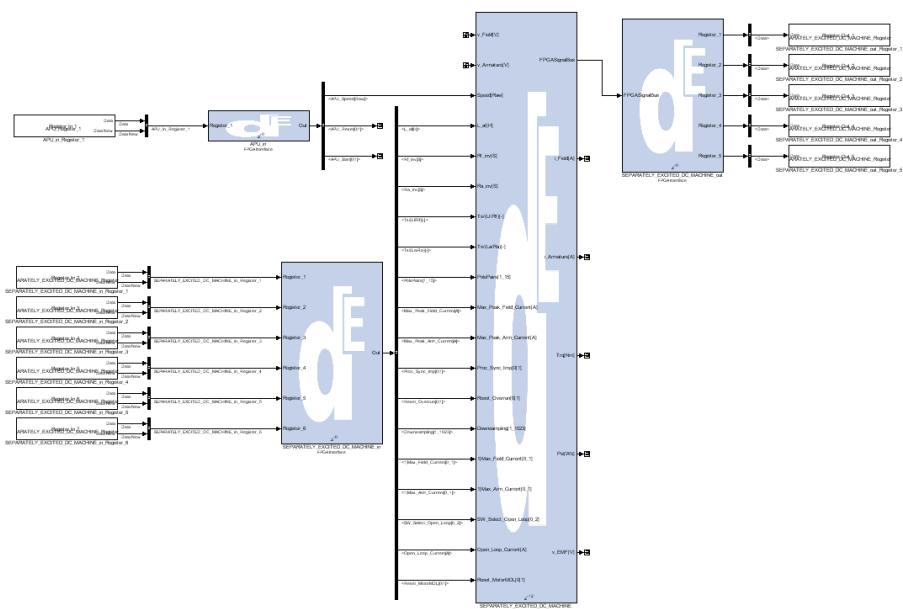


Figure 239: FPGA interface

Power Electronics: DCM Inverter

Objective

See the following sections for information on the power electronics unit (THREE_PHASE_DCM_INVERTER). The THREE_PHASE_DCM_INVERTER block implements a three-phase power converter that consists of six power switches (IGBT and body diodes connected in a bridge configuration). A precise simulation considers natural switching effects, such as the free-wheeling diode, and changing energy flow such as passive energy recovery into battery.

To enable the most interesting motor model parameters to be reused in a superior processor model, a processor synchronous average calculation of the motor torque and the stator currents is also included. A bit toggles with every sample step (processor synchronous impulse: PROC_SYNC_IMP) on the processor side, and on the FPGA side, the time is measured and the signal values are accumulated between two edges of the toggle bit. Downsampling is also possible to enlarge the capture time, but keep in mind that the resolution might decrease. An additional reset functionality of the average function is also implemented.



If the standard interface of the XSG Electric Library is used, make sure that the THREE_PHASE_DCM_INVERTER_in and THREE_PHASE_DCM_INVERTER_out block of the processor interface is located in the same sampled task. This ensures that the timing behavior of the included average functionality of the FPGA main block is correct. **Infractions of this requirement produce fail behavior in value conversion and calculation on the processor interface!**

All inverter parameters can be adjusted in the processor output block dialog.

The blockset contains the following elements:

- Processor Interface: THREE_PHASE_DCM_INVERTER_out
(Processor Interface)
- FPGA Interface: THREE_PHASE_DCM_INVERTER_in
(FPGA Interface)
- FPGA: THREE_PHASE_DCM_INVERTER
(FPGA Main Component)
- FPGA Interface: THREE_PHASE_DCM_INVERTER_out
(FPGA Interface)
- Processor Interface: THREE_PHASE_DCM_INVERTER_in
(Processor Interface)

Processor Output

Block

Merges the processor signals and writes them to the FPGA



Figure 240: THREE_PHASE_DCM_INVERTER_out block

Block Dialog

The processor output block contains several dialogs. All general parameters of the block can be configured in the page “General”.

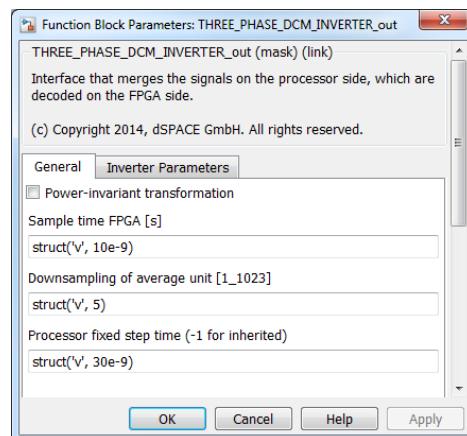


Figure 241: THREE_PHASE_DCM_INVERTER_out dialog; page “General”

The page has the following parameters:

Power-invariant transformation	[<input type="checkbox"/>]	Activate power invariant transformation for clark transformation	-
Downsampling of average unit	[<input type="checkbox"/>]	Downsampling factor of the average functionality in the range 1...1023	1 ... 1023; resolution: 1
Processor fixed step time	[<input type="checkbox"/> s]	Step time of the processor task. For offline simulation a different sample time can be set to simulate the processor interface communication behavior.	-

The inverter specific parameters can be configured on the page “Inverter Parameter” as show in the figure below.

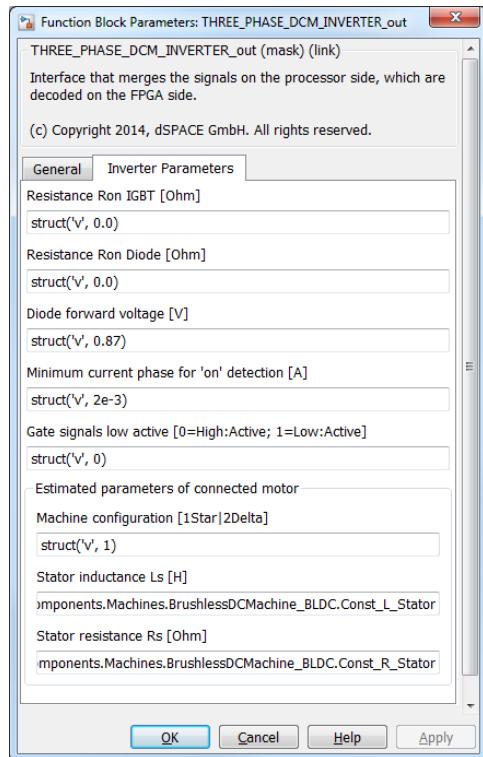


Figure 242: THREE_PHASE_DCM_INVERTER_out dialog; page “Inverter Parameters”

The dialog blockset has the following parameters:

Name	Unit	Description	Range
Resistance Ron IGBT	[Ohm]	Resistance of the IGBT in on-mode	0 ... 1 Ohm; resolution: 15.2E-6
Resistance Ron Diode	[Ohm]	Resistance of the Diode in on-mode	0 ... 1 Ohm; resolution: 15.2E-6
Diode forward voltage	[V]	Forward voltage of the diodes of the inverter	0 ... 4V; resolution: 16E-3
Minimium current phase for 'on' detection	[A]	Minimum phase current for the floating detection of each inverter leg	0 ... 1A; resolution: 15.2E-6
Gate signals low active	[-]	Inversion of the gate signals (HSD → LSD and LSD → HSD)	0 1
Machine configuration	[-]	Select the machine configuration; 1: star 2: delta	-
Stator inductance	[H]	Stator inductance of the connected motor	0 ... 0.25 H; resolution: 3.81E-6
Stator resistance	[Ohm]	Stator resistance of the connected motor	244E-6 ... 16 Ohm; resolution: 244E-6

Input

The THREE_PHASE_DCM_INVERTER_out block has the following inputs:

Name	Unit	Description	Range
FPGASignalBus	Bus	Bus containing the most significant feedback signals from the processor input block	-
Enable	[-]	Enable output voltage	0 1
V_DCLink	[V]	DC link voltage of the inverter	0 ... 1024V; resolution: 3.9E-3
Reset_Average	[-]	Reset function of the implemented FPGA average functionality	0 1

Output

The processor out block provides five input registers whose sectioning is shown below:

Register 1

Figure 243: THREE_PHASE_DCM_INVERTER_out Register 1

Name	Used Bits	Description	Range
DC Link Voltage V_DC_Link	17 ... 0	DC link voltage of the inverter	0 ... 2^17-1 represents 0... 1024V
Diode Forward Voltage V_Diode_Fw	25 ... 18	Diode forward voltage of the internal diodes of the inverter	0 ... 2^8-1 represents 0 ... 4V
Ls Ts-Rs	29 ... 26	Constant Ls Ts-Rs (Bit 16-19)	
SPARE	31 ... 30		

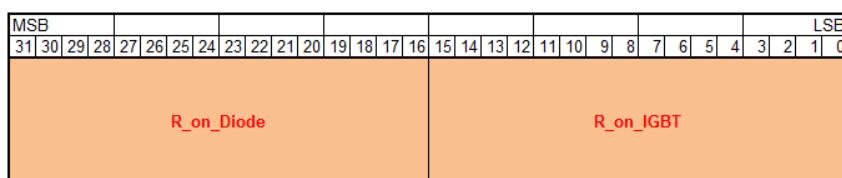
Register 2

Figure 244: THREE_PHASE_DCM_INVERTER_out Register 2

Name	Used Bits	Description	Range
Ron IGBT	15 ... 0	Resistance of the IGBT in on-mode	0 ... 2^16-1 represents 0 ... 1 Ohm
Ron Diode	31 ... 16	Resistance of the Diode in on-mode	0 ... 2^16-1 represents 0 ... 1 Ohm

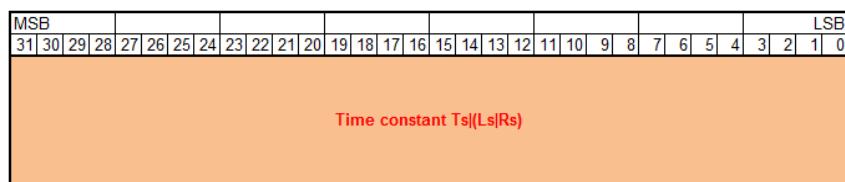
Register 3

Figure 245: THREE_PHASE_DCM_INVERTER_out Register 3

Name	Used Bits	Description	Range
Time Constant	31 ... 0	Time constant of the connected motor $Ts (Ls Rs)$	0 ... 2^32-1

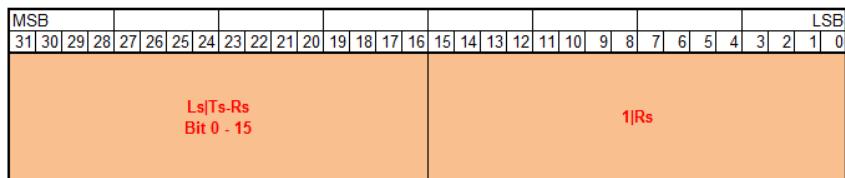
Register 4

Figure 246: THREE_PHASE_DCM_INVERTER_out Register 4

Name	Used Bits	Description	Range
1 Rs	15 ... 0	Inverse resistance	0 ... 2^16-1 Represents an inverse resistance with a format of UFix_16_4
Ls Ts-Rs	31 ... 16	Constant Ls Ts-Rs (Bit 0-15)	

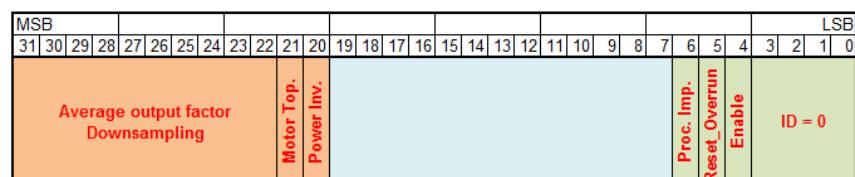
Register 5; ID = 0

Figure 247: THREE_PHASE_DCM_INVERTER_out Register 5; ID = 0

Name	Used Bits	Description	Range
Counter ID	3 ... 0	ID to select which register coding is activated via Processor	0 ... 1
Enable	29	Automatic generated enable of the gate signals	0 1
Reset_Overrun	5	Reset flag to reset an overrun of the average functionality	0 1
Proc. Sync. Impulse	6	Processor synchronous toggle bit	0 1
SPARE	19 ... 7		
Power Invariant	20	Switch to activate power invariant transformation	0 1
Motor Topology	21	Specify the motor topology	0 1
Average output factor Down-sampling	31 ... 22	Downsampling factor for average functionality of the outputs to the processor interface	1 ... 1023

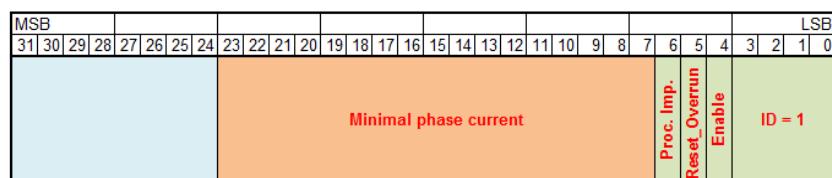
Register 5; ID = 1

Figure 248: THREE_PHASE_DCM_INVERTER_out Register 5; ID = 1

Name	Used Bits	Description	Range
Continuous Data	6 ... 0	Definition same as ID = 0	
Minimum Current I_Min_abc	23 ... 7	Minimum current in [A] of an absolute current; if below this value, a floating phase is detected.	0 ... 2^16-1 represents 0 ... 1A
SPARE	31 ... 24		

FPGA Main Component

Block

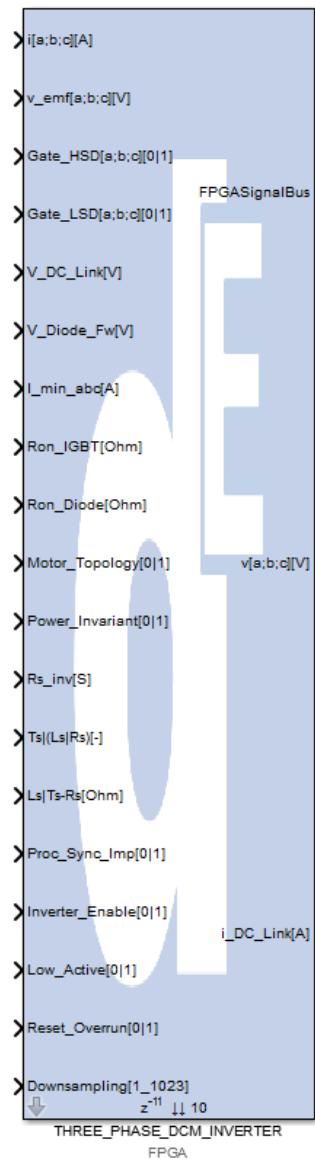


Figure 249: THREE_PHASE_DCM_INVERTER FPGA Main block

Block Dialog

The FPGA main blockset contains the following dialog.

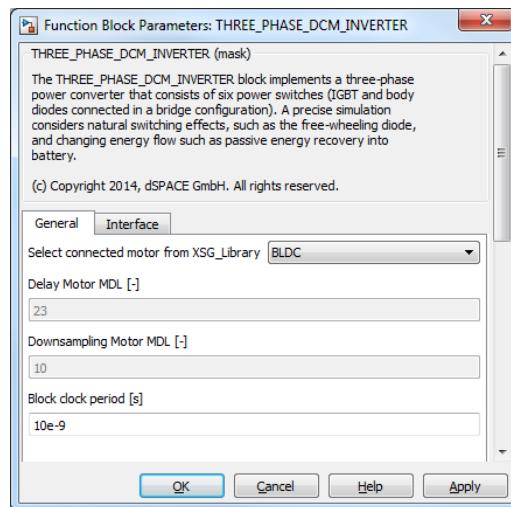


Figure 250: THREE_PHASE_DCM_INVERTER FPGA Main block dialog

The page “General” has the following parameters:

Name	Unit	Description	Range
Connected Motor from XSG EC Library	[-]	Specify the connected motor to the DCM Inverter model from the XSG ElectricComponents library	
Delay Motor MDL	[-]	Delay of the connected motor model.	
Downsampling Motor MDL	[-]	Downsampling of the connected motor model	Multiple of 10

For further information of the delay and the downsampling of the connected motor model please refer to the corresponding FPGA main block. On the block mask the specific parameters are displayed.

On the page Interface you can define the parameters of the input and output registers for automatic interface generation. The dialog also contains a button to start interface generation. For more details about automatic interface generation, refer to the XSG Utils documentation chapter Automatic Interface Generation. The FPGA clock period can be configured too.

Input

The main block has the following inputs:

Name	Unit	Description	Format
i	[A]	Vector of three stator currents of the connected motor	i_a, i_b; i_c: Fix_28_17
v_emf	[V]	Vector of the induced voltage of the PMSM motor	v_emf_a; v_emf_b; v_emf_c: Fix_18_7
Gate_HSD	[0 1]	Vector of the highside gates of the inverter	UFix_1_0 or Bool
Gate_LSD	[0 1]	Vector of the lowside gates of the inverter	UFix_1_0 or Bool
V_DC_Link	[V]	Internal DC link voltage of the inverter	UFix_18_7
V_Diode_Fw	[V]	Forward voltage of the internal inverter diodes	UFix_8_6
I_min_abc	[A]	Minimum current of an absolute current below; this value leads to detection of a floating phase.	UFix_16_16
Ron_IGBT	[Ohm]	Resistance of the IGBT in on-mode	UFix_16_16
Ron_Diode	[Ohm]	Resistance of the Diode in on-mode	UFix_16_16
Motor_Topology	[-]	Specify the motor topology	UFix_1_0
Power_Invariant	[-]	Specify the internal clark transformation	UFix_1_0
Rs_inv	[S]	Inverse resistance	UFix_16_4
Ts (Ls Rs)	[-]	Time constant of connected motor model	UFix_32_32
Ls Ts-Rs	[Ohm]	Resistance factor	UFix_28_17
Proc_Sync_Imp	[0 1]	Processor synchronize toggle bit	UFix_1_0
Inverter_Enable	[-]	Enable of the gate signals of the inverter model	Bool
Low_Active	[-]	Inversion of the gate signals	UFix_1_0
Reset_OVERRUN	[0 1]	Reset flag to reset an overrun of the average functionality	Bool
Downsampling	[-]	Downsampling factor for average functionality of the outputs to the processor interface	UFix_10_0

Output

The THREE_PHASE_DCM_INVERTER main block has the following outputs with a maximal latency of three:

Name	Unit	Description	Format
FPGASignalBus	Bus	Bus containing the most significant signals of this block	-
v	[-]	Stator voltage vector	Fix_18_7
i_DC_Link	[A]	DC link current of the inverter	Fix_20_7

Processor Input

Block Adapts the FPGA signals for the processor side.



Figure 251: THREE_PHASE_DCM_INVERTER_in block

Block Dialog The dialog only provides a short block description.

Input The processor input block has the following inputs:

Register 1

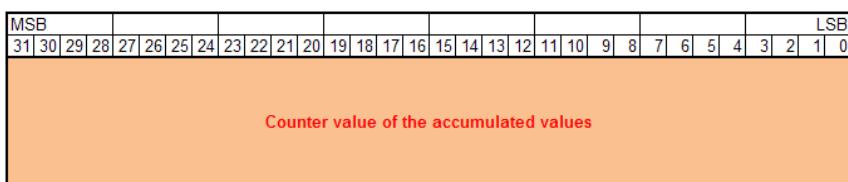
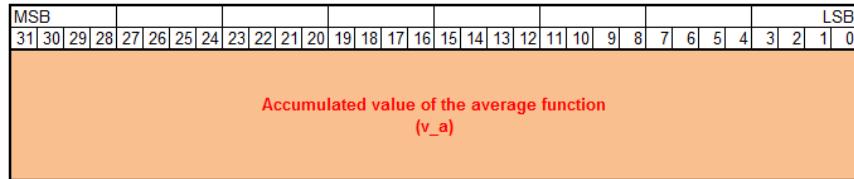
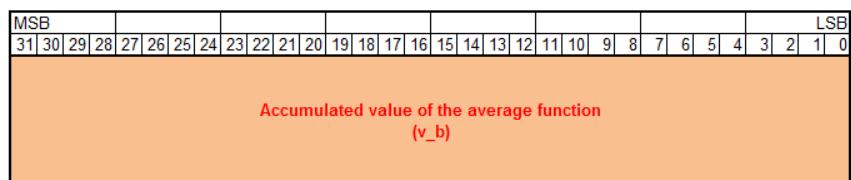


Figure 252: THREE_PHASE_DCM_INVERTER_in Register 1

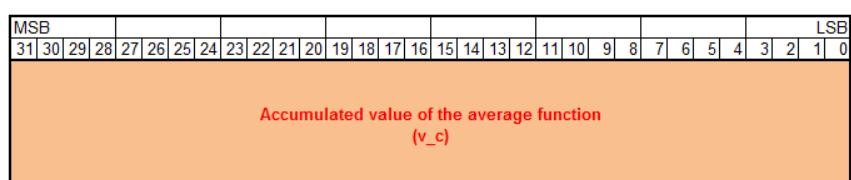
Name	Used Bits	Description	Range
Accu counter	31 ... 0	Counter value of the accumulated values; if an overrun is detected the value $2^{32}-1$ will be sent	0 ... $2^{32}-1$

Register 2**Figure 253: THREE_PHASE_DCM_INVERTER_in Register 2**

Name	Used Bits	Description	Format
Accu	31 ... 0	Accumulated value of the average function; if an overrun is detected the value $2^{31}-1$ will be sent Signal: v_a	0 ... $2^{32}-1$

Register 3**Figure 254: THREE_PHASE_DCM_INVERTER_in Register 3**

Name	Used Bits	Description	Range
Accu	31 ... 0	Accumulated value of the average function; if an overrun is detected the value $2^{31}-1$ will be sent Signal: v_b	0 ... $2^{32}-1$

Register 4**Figure 255: THREE_PHASE_DCM_INVERTER_in Register 4**

Name	Used Bits	Description	Range
Accu counter	31 ... 0	Accumulated value of the average function; if an overrun is detected the value 2^31-1 will be sent Signal: v_c	0 ... 2^32-1

Register 5

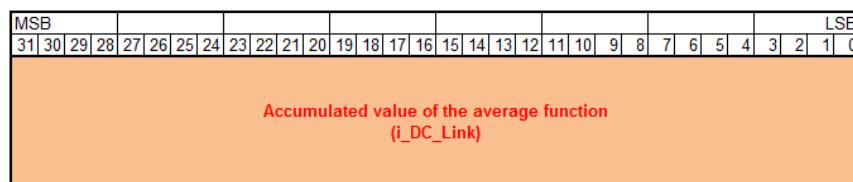


Figure 256: THREE_PHASE_DCM_INVERTER_in Register 5

Name	Used Bits	Description	Range
Accu	31 ... 0	Accumulated value of the average function; if an overrun is detected the value 2^31-1 will be sent Signal: i_DC_Link	0 ... 2^32-1

Output

The processor input block has the following outputs:

Name	Unit	Description	Range
FPGASignalBus	Bus	Bus containing the most significant signals of this block	-
v	[V]	Voltage bus of the a,b,c phase	-1024 ... 1023.99 V
I_DC_Link	[A]	DC link current of the inverter	-4096 ... 4095.99 A

Block additional feedback

Adapts the FPGA signals for additional feedback of the FPGA to the processor side (like power consumption,...).

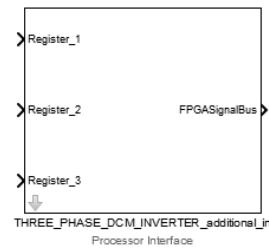


Figure 257: THREE_PHASE_DCM_INVERTER_additional_in block

Block Dialog

The dialog only provides a short block description.

Input

The processor input block has the following inputs:

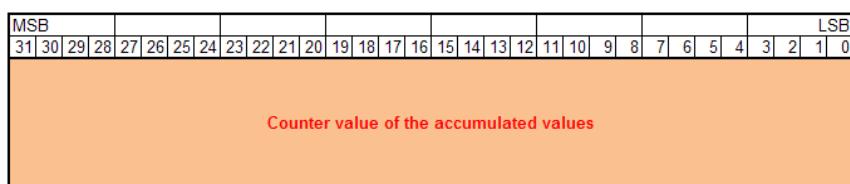
Register 1

Figure 258: THREE_PHASE_DCM_INVERTE_additional_in Register 1

Name	Used Bits	Description	Range
Accu	31 ... 0	Counter value of the accumulated values; if an overrun is detected the value $2^{32}-1$ will be sent	0 ... $2^{32}-1$

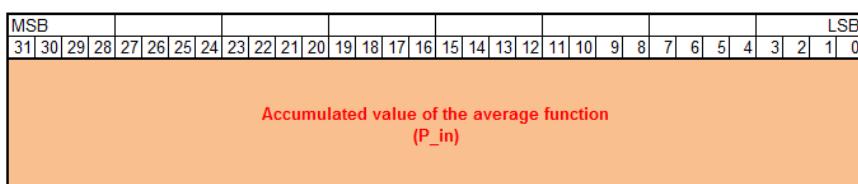
Register 2

Figure 259: THREE_PHASE_DCM_INVERTER_additional_in Register 2

Name	Used Bits	Description	Range
Accu	31 ... 0	Accumulated value of the average function; if an overrun is detected the value $2^{31}-1$ will be sent Signal: input power consumption P_in [W]	0 ... $2^{32}-1$

Register 3

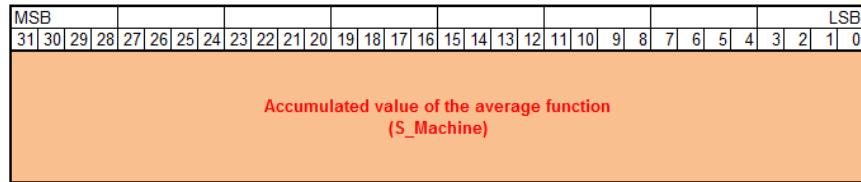


Figure 260: THREE_PHASE_DCM_INVERTER_additional_in Register 3

Name	Used Bits	Description	Range
Accu	31 ... 0	Accumulated value of the average function; if an overrun is detected the value 2^31-1 will be sent Signal: output power S_Machine [VA]	0 ... 2^32-1

Output

The processor input block has the following outputs:

Name	Unit	Description	Range
FPGASignalBus	Bus	Bus containing the most significant signals of this block	-

Interface Examples

Processor blocks

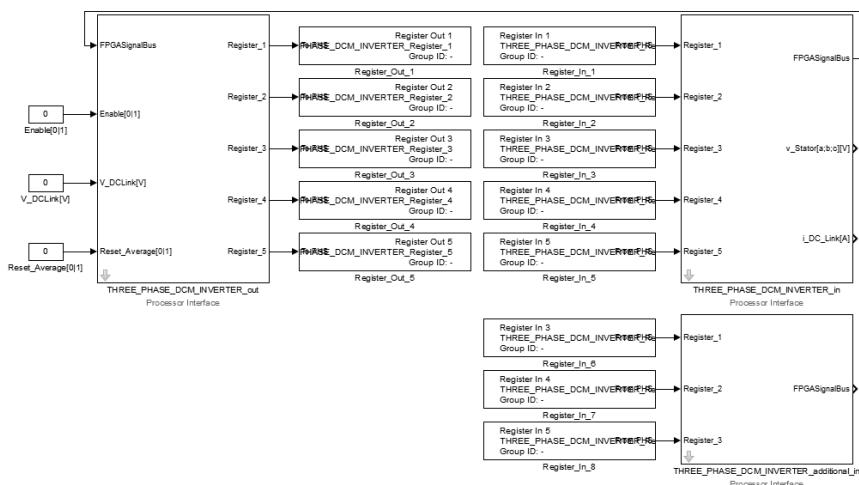


Figure 261: Processor interface

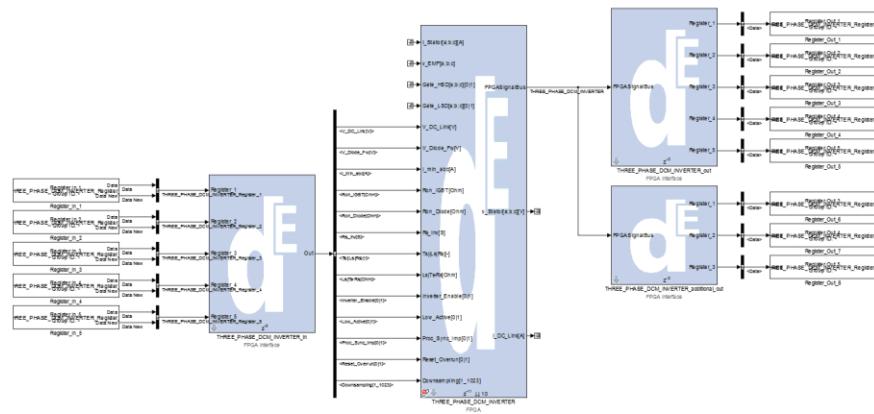
FPGA block

Figure 262: FPGA interface

Mechanic

Description / Overview

Provides the opportunity to calculate a mechanic model with a small inertia on the FPGA. The mechanic blockset represent a functional bundle of the processor model functions "Motor Inertia" and Motor Position" (described in the chapter "Processor based model parts"). Speed depending damping coefficient, load torque and open loop functionalities can be set from the processor during runtime. The following schematic shows the internal functionality of the mechanic block.

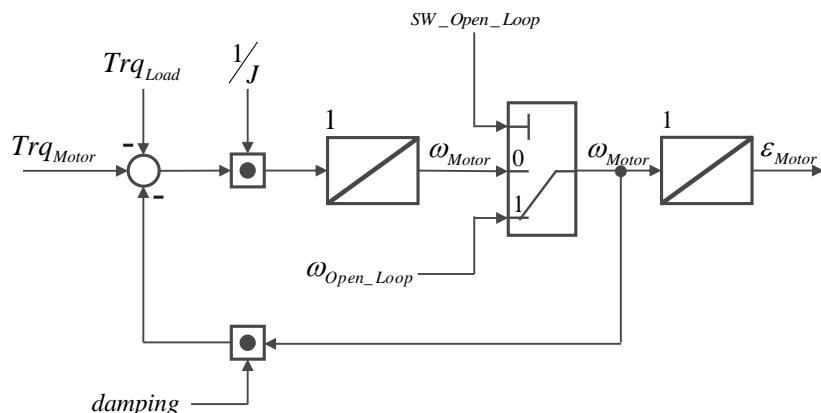


Figure 263: Schematic of MECHANIC block

The equation of the system can be written as (damping = 0; SW_Open_Loop = 0):

$$\omega_{Motor} = \frac{1}{J_{Motor} + J_{Load}} \int (Trq_{Motor} - Trq_{Load}) dt$$

$$\epsilon_{Motor} = \int (\omega_{Motor}) dt$$

ω_{Engine}	Engine speed
ϵ_{Engine}	Engine position
ω_{Motor}	Engine speed
Trq_{Motor}	Engine torque
Trq_{Load}	Load torque
J_{Motor}	Engine shaft inertia
J_{Load}	Load inertia

The blockset contains the following elements:

- Processor Interface: MECHANIC_out (Processor Interface)
- FPGA Interface: MECHANIC_in (FPGA Interface)
- FPGA: MECHANIC (FPGA Main Component)
- FPGA Interface: MECHANIC_out (FPGA Interface)
- Processor Interface: MECHANIC_in (Processor Interface)

Processor Output

Block

Merges the processor signals and writes them to the FPGA

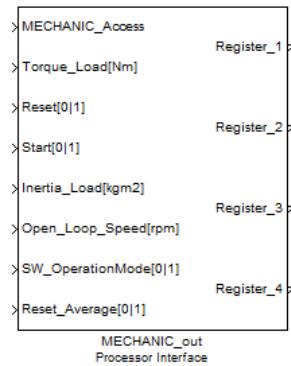


Figure 264: MECHANIC_out block

Block Dialog

The processor output blockset contains the following dialog.

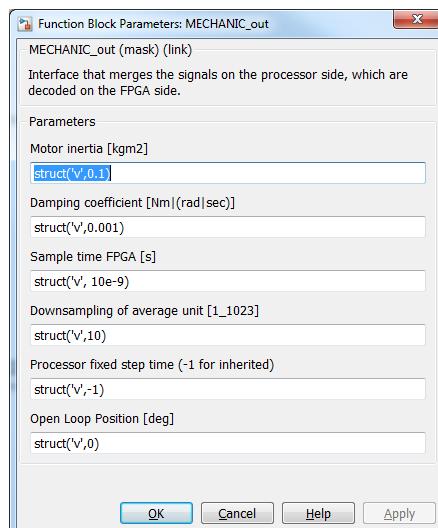


Figure 265: MECHANIC_out dialog

The dialog blockset has the following parameters:

Name	Unit	Description	Range
Motor inertia	[kgm2]	Motor inertia	Motor inertia + Inertia Load = 0.5e-6 ... 1
Damping coefficient	[Nm](rad s)]	Speed depending damping coefficient	0 ... 0.25
Sample time FPGA	[s]	Sample time of the FPGA	
Downsampling of average unit	[‐]	Downsampling factor of the average functionality in the range 1...1023	1 ... 1023; resolution: 1
Processor fixed step time	[s]	Step time of the processor task. For offline simulation a different sample time can be set to simulate the processor interface communication behavior.	-
Open Loop Position	[deg]	Open Loop Position; in SW_Operation = 2 the internal APU of the Mechanic will be controlled to the open loop position	-

Input

The MECHANIC _out block has the following inputs:

Name	Unit	Description	Range
MECHANIC_Access	BUS	Has to be connected to MECHANIC_in block	
Torque Load	[Nm]	Load torque of the mechanic	Internal normed to the input bitwidth of the motor torque of the FPGA main component; resolution: 32 Bit
Reset	[-]	Reset of the mechanic model	0 1
Start	[-]	Start / Enable of the mechanic model	0 1
Inertia_Load	[kgm ²]	Inertia of the external load	Motor inertia + Inertia Load = 0.5e-6 ... 1
Open_Loop_Speed	[rpm]	APU speed in open loop mode	(-732421 ... 732421) · 10E-9/ Ts_FPGA; resolution: 1.36E-3
SW_Operation_Mode	[-]	Switch to control the operation mode 0: closed loop mode (mechanic + APU) 1: open loop mode speed (only APU; mechanic is in reset mode) 2: open loop mode position (internal APU will be controlled to the position setpoint)	0 1 2
Reset_Average	[-]	Reset function of the implemented FPGA average functionality	0 1

FPGA Main Component

Block

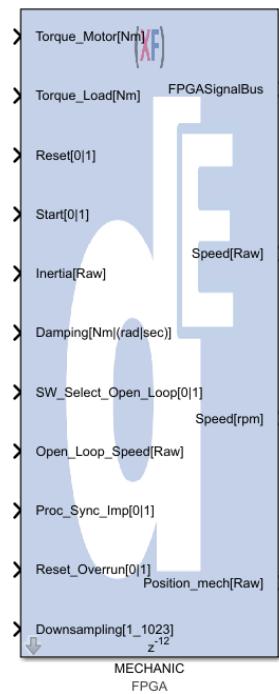


Figure 266: MECHANIC FPGA Main block for fixed point

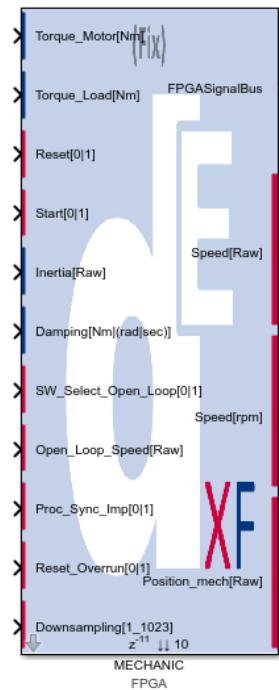


Figure 267: MECHANIC FPGA Main block for floating point

Block Dialog

The FPGA main blockset contains the following dialog.

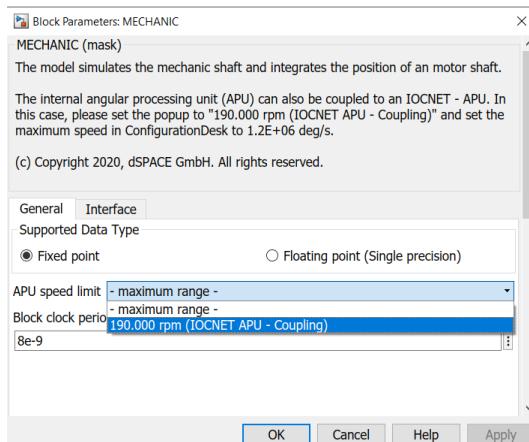


Figure 268: MECHANIC FPGA Main block dialog

The page “General” has the following parameters:

Name	Unit	Description	Range
Connected Motor from XSG EC Library	[-]	Defintion of the maximum speed of the internal APU “- maximum range -“ “190.000 rpm”	

For detailed information about the APU coupling between different IO-Boards, please refer to the XSG Utils documentation.

On the page Interface you can define the parameters of the input and output registers for automatic interface generation. The dialog also contains a button to start interface generation. For more details about automatic interface generation, refer to the XSG Utils documentation chapter Automatic Interface Generation. The FPGA clock period can be configured too.

Input

The main block has the following inputs:

Name	Unit	Description	Format
Torque_Motor	[Nm]	Motor Torque	user defined
Torque_Load	[Nm]	Load Torque	Must be the same as Torque_Motor
Reset	[0 1]	Reset of the mechanic model	UFix_1_0 or Bool
Start	[0 1]	Start / Enable of the mechanic model	UFix_1_0 or Bool
Inertia	[Raw]	Inertia motor + load (precompiled for integrator on FPGA)	UFix_31_26 or XFloat_8_24
Damping	[Nm](rad sec)]	Speed depending damping coefficient	UFix_22_22 or XFloat_8_24
SW_Select_Open_Loop	[0 1]	Switch to control the operation mode 0: closed loop mode (mechanic + APU) 1: open loop mode (only APU; mechanic is in reset mode)	UFix_1_0
Open_Loop_Speed	[Raw]	Open-Loop speed	Fix_30_0
Proc_Sync_Imp	[0 1]	Processor synchronize toggle bit	UFix_1_0
Reset_OVERRUN	[0 1]	Reset flag to reset an overrun of the average functionality	Bool
Downsampling	[-]	Downsampling factor for average functionality of the outputs to the processor interface	UFix_31_0

Output

The MECHANIC main block has the following outputs:

Name	Unit	Description	Format
FPGASignalBus	Bus	Bus containing the most significant signals of this block	-
Speed	[Raw]	Actual speed in precompiled format for an APU	Fix_30_0
Speed	[rpm]	Actual speed	Fix_31_10
Position_mech	[Raw]	Actual position 41 ... 0 Bit: Single turn bits 64 ... 42 Bit: Multi turn bits	UFix_64_0

Processor Input

Block

Adapts the FPGA signals for the processor side.

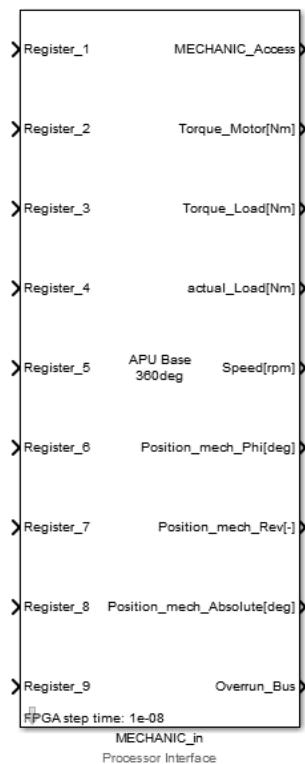


Figure 269: MECHANIC_in block

Block Dialog

The dialog only provides a short block description.

Output

The processor input block has the following outputs:

Name	Unit	Description	Range
MECHANIC_Access	BUS	Has to be connected to MECHANIC_out block	
Torque_Motor	[Nm]	input motor torque	Depends on user setting of the input motor torque of the FPGA main component (Fix/XFloat_8_24)
Torque_Load	[Nm]	input load torque	Depends on user setting of the input motor torque of the FPGA main component (Fix/XFloat_8_24)
actual_Load	[Nm]	load torque at the actual speed	Depends on user setting of the input motor torque of the FPGA main component (Fix/XFloat_8_24)
Speed	[rpm]	Speed of the mechanic	(-732421 ... 732421) · 10E-9/ Ts_FPGA
Position_mech_Phi	[deg]	Position of the motor shaft with respect to the APU base	0 ... 360°
Position_mech_Rev	[-]	Revolution of the motor shaft with respect to the APU base	
Position_mech_Absolute	[deg]	Absolute position of the motor shaft	
Overrun_Bus	Bus	Overrun feedback from the average function and the internal Integrator	

Interface Examples

Processor blocks

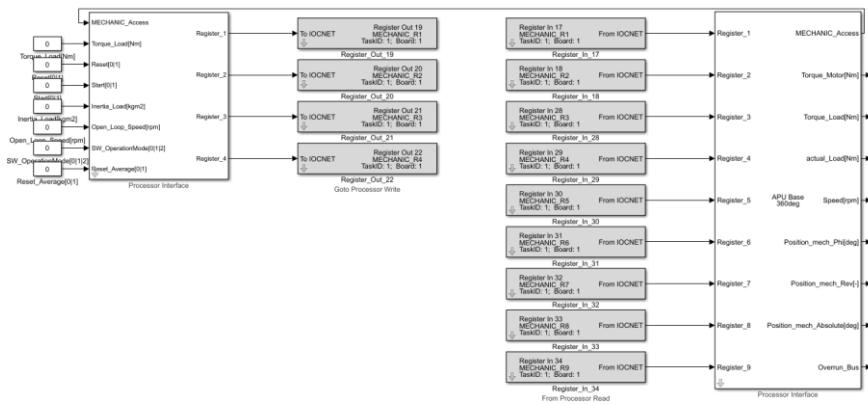


Figure 270: Processor interface

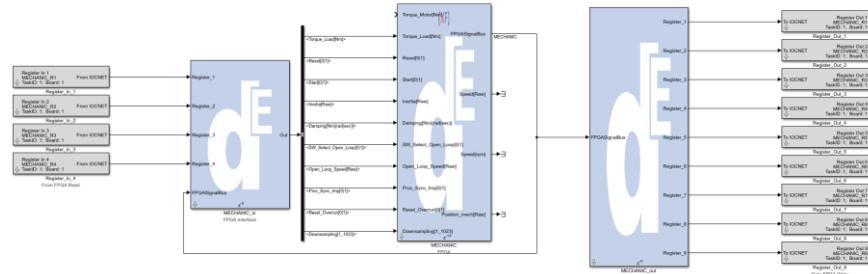
FPGA block

Figure 271: FPGA interface for fixed point

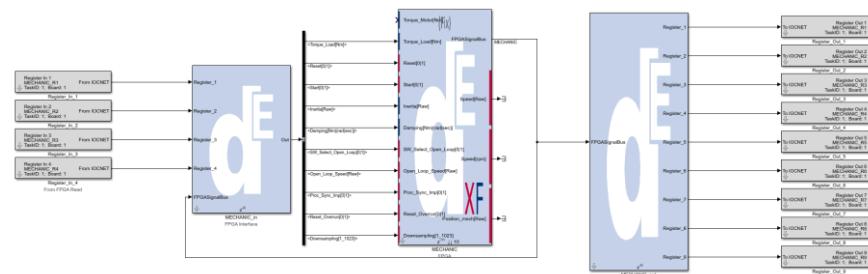


Figure 272: FPGA interface for floating point

Processor based model parts

Overview

Overview

The XSG Electric Components library also contains several processor based model parts for full support of e-motor simulation. Therefore the following processor model parts are available:

- Motor inertia
- Motor position integration
- Clarke / Park transformation
- Star / Delta conversion
- Park Modulator
- Discrete PT1
- Discrete Integrator
- PI Controller
- PMSM Controller
- Space Vector Modulation

The following section describes the different functionalities in detail.

Motor Inertia

Objective

The model simulates the mechanic of a motor shaft.

The next illustration shows the Simulink representation of the motor inertia.

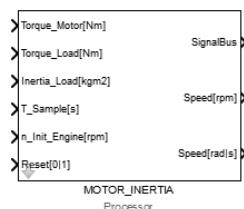


Figure 273: MOTOR_INERTIA block

The equation of the system can be written as

$$\omega_{Motor} = \frac{1}{J_{Motor} + J_{Load}} \int (Trq_{Motor} - Trq_{Load}) dt$$

ω_{Motor}	Engine speed
Trq_{Motor}	Engine torque
Trq_{Load}	Load torque
J_{Motor}	Engine shaft inertia
J_{Load}	Load inertia

Block Dialog

The blockset contains the following dialog.

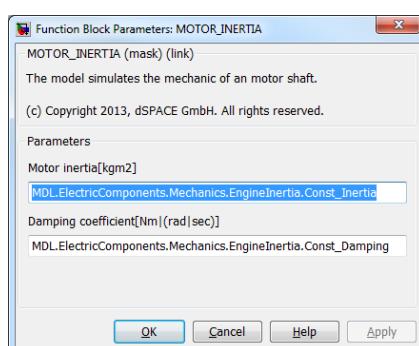


Figure 274: MOTOR_INERTIA dialog

The dialog has the following parameters:

Name	Unit	Description
Motor inertia	[kgm ²]	Motor inertia
Damping coefficient	[Nm/(rad s)]	Damping coefficient

Input

The MOTOR_INERTIA block has the following inputs:

Name	Unit	Description
Torque_Motor	[Nm]	Motor torque
Torque_Load	[Nm]	Load torque
Inertia_Load	[kgm ²]	Load inertia
n_Init_Engine	[rpm]	Initial engine speed
T_Sample	[s]	Sample time
Reset	[-]	Reset functionality

Output

The block has the following outputs:

Name	Unit	Description
SignalBus	Bus	Bus containing the most significant signals of this block
Speed	[rpm]	Actual motor speed in rpm
Speed	[rad s]	Actual motor speed in rad s

Motor Position

Objective

The model simulates the position integration of a motor shaft.

The next illustration shows the Simulink representation of the motor position.

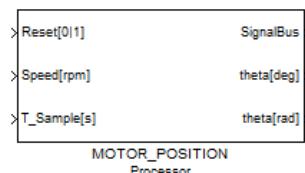


Figure 275: MOTOR_POSITION block

The equation of the system can be written as

$$\varepsilon_{Engine} = \int (\omega_{Engine}) dt$$

ω_{Engine}	Engine speed
ε_{Engine}	Engine position

Block Dialog

The dialog provides only a short block description.

Input

The MOTOR_INERTIA block has the following inputs:

Name	Unit	Description
Reset	[-]	Reset functionality
Speed	[rpm]	Motor speed
T_Sample	[s]	Sample time

Output

The block has the following outputs:

Name	Unit	Description
SignalBus	Bus	Bus containing the most significant signals of this block
theta	[deg]	Actual angle in deg
theta	[rad]	Actual angle in rad

Park Modulator

Objective

The system calculates the sinus and cosines of the angle input to be used as a signal source for the transformation block.

The next illustration shows the Simulink representation of the park modulation.

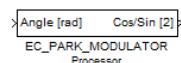


Figure 276: EC_PARK_MODULATOR block

The equation of the system can be written as

$$out_{\text{sinus}} = \sin(ep_s)$$

$$out_{\text{cosinus}} = \cos(ep_s)$$

Block Dialog

The dialog provides only a short block description.

Input

The MOTOR_INERTIA block has the following inputs:

Name	Unit	Description
Angle	[rad]	Actual angle

Output

The block has the following outputs:

Name	Unit	Description
Cos/Sin	[-] Vector	Sinus and Cosines of the input angle

Clarke / Park transformation

Objective

The system represents a Clarke and/or Park transformation for electric motor applications depending on the mask configuration.

The equation of the system can be written as

Clarke-transformation:

The equation of the system can be written as Clarke-transformation:

Orthogonal components for three-phase system:

$$\begin{bmatrix} x_\alpha \\ x_\beta \\ x_0 \end{bmatrix} = \begin{bmatrix} \frac{2}{3} & -\frac{1}{3} & -\frac{1}{3} \\ 0 & \frac{1}{\sqrt{3}} & -\frac{1}{\sqrt{3}} \\ \frac{1}{3} & \frac{1}{3} & \frac{1}{3} \end{bmatrix} \cdot \begin{bmatrix} x_a \\ x_b \\ x_c \end{bmatrix}$$

This form is invariant with respect to reference values: $x_\alpha = x_a$ when $x_0 = 0$. Moreover,

$$x_0 = \frac{1}{3} \sum_{i=a,b,c} x_i$$

is the average value of components x_a , x_b and x_c . The corresponding inverse Clarke-transformation is

$$\begin{bmatrix} x_a \\ x_b \\ x_c \end{bmatrix} = \begin{bmatrix} 1 & 0 & 1 \\ -\frac{1}{2} & \frac{\sqrt{3}}{2} & 1 \\ -\frac{1}{2} & -\frac{\sqrt{3}}{2} & 1 \end{bmatrix} \cdot \begin{bmatrix} x_\alpha \\ x_\beta \\ x_0 \end{bmatrix}$$

A different form of Clarke-transformation is invariant with respect to power:

$$\begin{bmatrix} x_\alpha \\ x_\beta \\ x_0 \end{bmatrix} = \sqrt{\frac{3}{2}} \cdot \begin{bmatrix} \frac{2}{3} & -\frac{1}{3} & -\frac{1}{3} \\ 0 & \frac{1}{\sqrt{3}} & -\frac{1}{\sqrt{3}} \\ \frac{\sqrt{2}}{3} & \frac{\sqrt{2}}{3} & \frac{\sqrt{2}}{3} \end{bmatrix} \cdot \begin{bmatrix} x_a \\ x_b \\ x_c \end{bmatrix}$$

The corresponding inverse Clarke-transformation is

$$\begin{bmatrix} x_a \\ x_b \\ x_c \end{bmatrix} = \sqrt{\frac{2}{3}} \begin{bmatrix} 1 & 0 & \frac{1}{\sqrt{2}} \\ -\frac{1}{2} & \frac{\sqrt{3}}{2} & \frac{1}{\sqrt{2}} \\ -\frac{1}{2} & -\frac{\sqrt{3}}{2} & \frac{1}{\sqrt{2}} \end{bmatrix} \cdot \begin{bmatrix} x_\alpha \\ x_\beta \\ x_0 \end{bmatrix}$$

Park-transformation:

Transformation of orthogonal components to a reference system which is turned by a certain angle. The amount (geometric mean value) is unchanged:

$$\begin{bmatrix} x_a \\ x_b \end{bmatrix} = \begin{bmatrix} \cos(\varepsilon) & \sin(\varepsilon) \\ -\sin(\varepsilon) & \cos(\varepsilon) \end{bmatrix} \cdot \begin{bmatrix} x_\alpha \\ x_\beta \end{bmatrix}$$

The corresponding inverse Park-transformation is

$$\begin{bmatrix} x_\alpha \\ x_\beta \end{bmatrix} = \begin{bmatrix} \cos(\varepsilon) & -\sin(\varepsilon) \\ \sin(\varepsilon) & \cos(\varepsilon) \end{bmatrix} \cdot \begin{bmatrix} x_a \\ x_b \end{bmatrix}$$

The next illustration shows the Simulink representation of the transformation block.

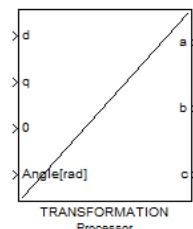


Figure 277: TRANSFORMATION block

Block Dialog

The blockset contains the following dialog.

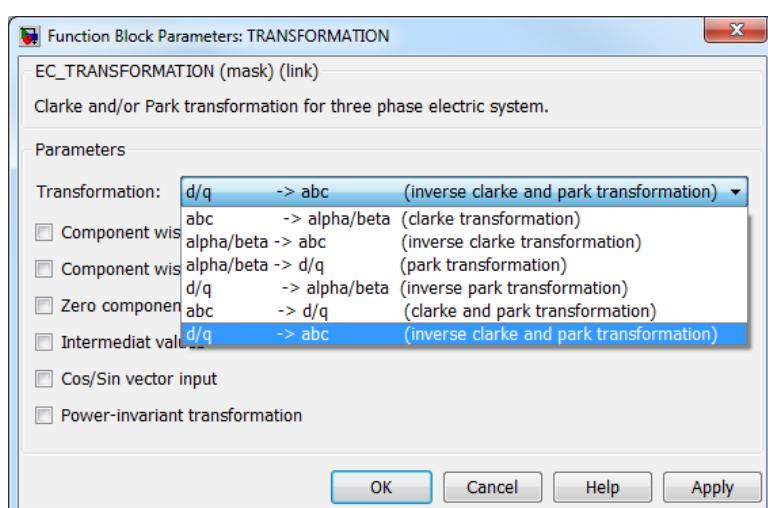


Figure 278: Transformation dialog

The dialog has the following parameters:

Name	Unit	Description
Transformation	[-]	abc → alpha/beta (clarke transformation) alpha/beta → abc (inverse clarke transformation) alpha/beta → d/q (park transformation) d/q → alpha/beta (inverse park transformation) abc → d/q (clarke and park transformation) d/q → abc (inverse clarke and park transformation)
Component wise input	[-]	If selected the inputs are component wise, else the Signals are muxed
Component wise output	[-]	If selected the outputs are component wise, else the Signals are muxed
Zero component	[-]	Generate the Zero component output
intermediate values	[-]	Internal values are routed to the output port
Cos/Sin vector input	[-]	Generate sin und cos input, so turnaround times can be save, because you have to calc the sin and cos functions only at one point with the Park Modulator
Power-invariant transformation	[-]	A different form of Clarke-Transformation is invariant with respect to power (German: leistungsinvariante Transformation):

Inputs and outputs are variable depending on the block configuration.

Star / Delta conversion

Objective

The system represents a current (or voltage) conversion from three-phase supply variables to three-phase machine variables (in both directions) depending on how the machine is configured.

The following illustration shows the two main configurations:

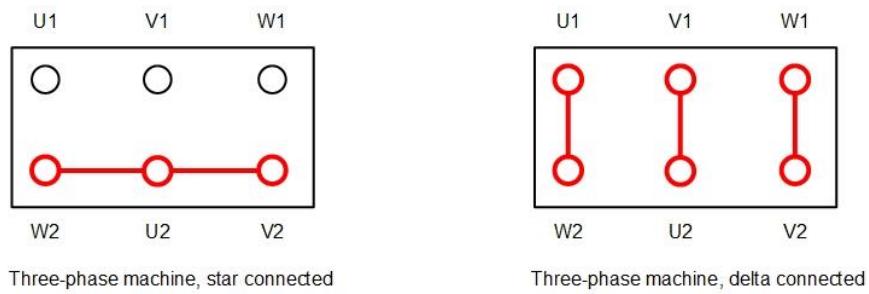


Figure 279: Machine configuration

For the star connected machine, the terminals U_2 , V_2 , W_2 are interconnected in the terminal box on the machine.

For the delta connected machine, the terminal pairs (U_1, W_2) , (V_1, U_2) , (W_1, V_2) are interconnected (clockwise) and the terminal pairs (U_1, V_2) , (V_1, W_2) , (W_1, U_2) are interconnected (counterclockwise) in the terminal box on the machine.

Current and voltage conversion of star connected machine

For the case where the input and the output variables are star, the machine phases have a common neutral point as shown in Figure below.

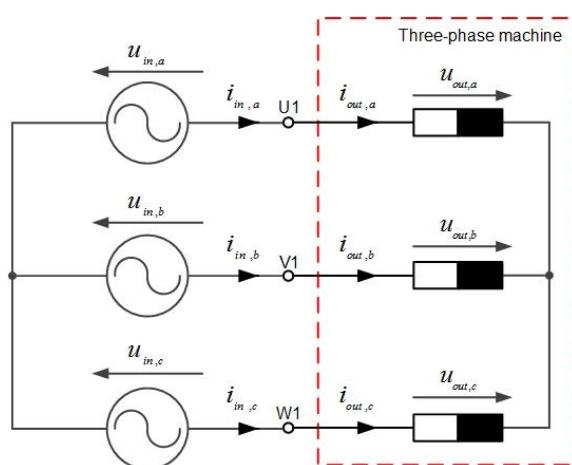


Figure 280: Machine configuration star

With respect to those variables the following expressions are valid

$$\begin{bmatrix} u_{out,a} \\ u_{out,b} \\ u_{out,c} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} u_{in,a} \\ u_{in,b} \\ u_{in,c} \end{bmatrix} \quad \begin{bmatrix} i_{out,a} \\ i_{out,b} \\ i_{out,c} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} i_{in,a} \\ i_{in,b} \\ i_{in,c} \end{bmatrix}$$

Note that the zero sequence voltage is not considered here in the transformation.

The inverse are given by

$$\begin{bmatrix} u_{in,a} \\ u_{in,b} \\ u_{in,c} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} u_{out,a} \\ u_{out,b} \\ u_{out,c} \end{bmatrix} \quad \begin{bmatrix} i_{in,a} \\ i_{in,b} \\ i_{in,c} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} i_{out,a} \\ i_{out,b} \\ i_{out,c} \end{bmatrix}$$

Current and voltage conversion of delta connected machine

For the case where the input variables are star and the output variables are delta (clockwise), the machine phases have not a common neutral point as shown in Figure below.

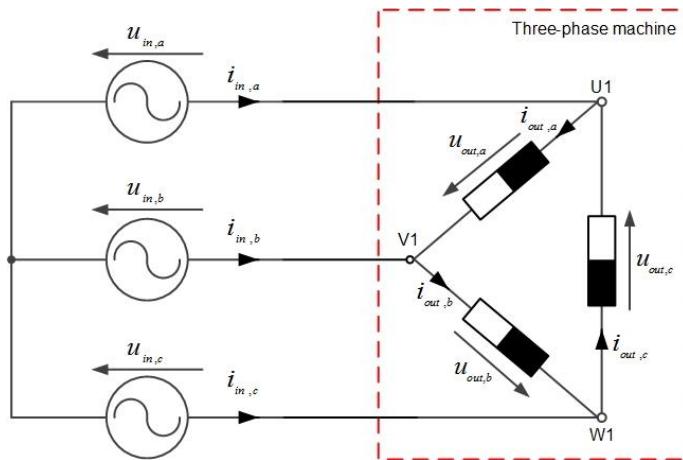


Figure 281: Machine configuration delta

With respect to those variables the following expressions are valid

$$\begin{bmatrix} u_{out,a} \\ u_{out,b} \\ u_{out,c} \end{bmatrix} = \begin{bmatrix} 1 & -1 & 0 \\ 0 & 1 & -1 \\ -1 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} u_{in,a} \\ u_{in,b} \\ u_{in,c} \end{bmatrix} \quad \begin{bmatrix} i_{out,a} \\ i_{out,b} \\ i_{out,c} \end{bmatrix} = \begin{bmatrix} \frac{1}{3} & -\frac{1}{3} & 0 \\ 0 & \frac{1}{3} & -\frac{1}{3} \\ -\frac{1}{3} & 0 & \frac{1}{3} \end{bmatrix} \cdot \begin{bmatrix} i_{in,a} \\ i_{in,b} \\ i_{in,c} \end{bmatrix}$$

Note that the zero sequence voltage is not considered here in the transformation.

The inverse are given by

$$\begin{bmatrix} u_{in,a} \\ u_{in,b} \\ u_{in,c} \end{bmatrix} = \begin{bmatrix} \frac{1}{3} & 0 & -\frac{1}{3} \\ -\frac{1}{3} & \frac{1}{3} & 0 \\ \frac{1}{3} & 0 & \frac{1}{3} \end{bmatrix} \cdot \begin{bmatrix} u_{out,a} \\ u_{out,b} \\ u_{out,c} \end{bmatrix} \quad \begin{bmatrix} i_{in,a} \\ i_{in,b} \\ i_{in,c} \end{bmatrix} = \begin{bmatrix} 1 & 0 & -1 \\ -1 & 1 & 0 \\ 0 & -1 & 1 \end{bmatrix} \cdot \begin{bmatrix} i_{out,a} \\ i_{out,b} \\ i_{out,c} \end{bmatrix}$$

Conversion Block

The next illustration shows the Simulink representation of the transformation block.

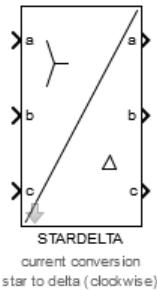


Figure 282: Conversion block

Block Dialog

The blockset contains the following dialog.

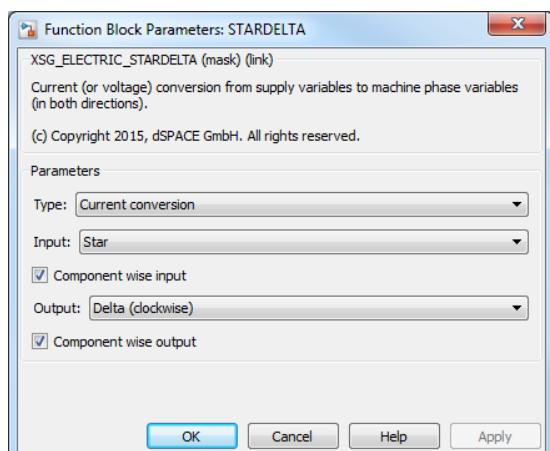


Figure 283: StarDelta conversion dialog

The dialog has the following parameters:

Name	Unit	Description
Type	[·]	Type of conversion: <ul style="list-style-type: none">• Current conversion• Voltage conversion
Input	[·]	Input signals need to be transformed: <ul style="list-style-type: none">• Star• Delta (clockwise)• Delta (counterclockwise)
Component wise input	[·]	If selected the output signals are component wise, else they are combined into a vector
Output	[·]	Into which quantities the output signals are transformed: <ul style="list-style-type: none">• Star• Delta (clockwise)• Delta (counterclockwise)
Component wise output	[·]	If selected the output signals are component wise, else they are combined into a vector

Discrete PT1

Objective

The system simulates a discrete first order lag element. The implemented integration method is Backward Euler.

The equation of the system can be written as

$$y(k) = \frac{K \cdot T_{Sample}}{(T_{Sample} + T)} \cdot e(k) + \frac{T}{T_{Sample} + T} \cdot y(k-1)$$

The next illustration shows the Simulink representation of the integrator block.

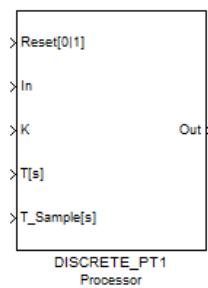


Figure 284: DISCRETE_PT1 block

Block Dialog

The blockset contains the following dialog.

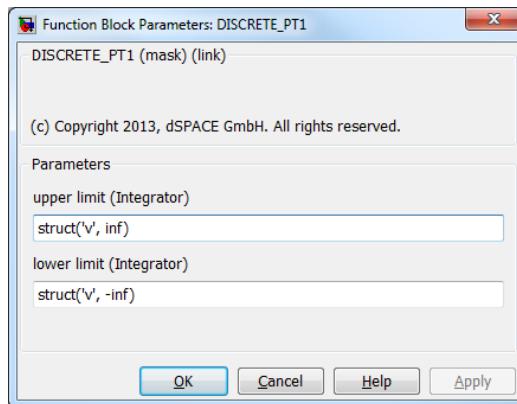


Figure 285: DISCRETE_PT1 dialog

The dialog has the following parameters:

Name	Unit	Description
upper_limit	[-]	Upper limit for integrator saturation
lower_limit	[-]	Lower limit for integrator saturation

Input

The DISCRETE_PT1 block has the following inputs:

Name	Unit	Description
Reset	[-]	Reset the integrator
In	[-]	Input value
Initial	[-]	Initial value
T_Sample	[s]	Sample time

Output

The block has the following outputs:

Name	Unit	Description
Out	[-]	Time discrete integration of the input signal

Discrete Integrator

Objective

The system performs a discrete-time integration of its input signal. The implemented integration method is Backward Euler.

The equation of the system can be written as

$$y(k) = T_{Sample} \cdot e(k) + y(k-1) + y_0$$

The next illustration shows the Simulink representation of the integrator block.

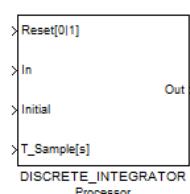


Figure 286: DISCRETE_INTEGRATOR block

Block Dialog

The blockset contains the following dialog.

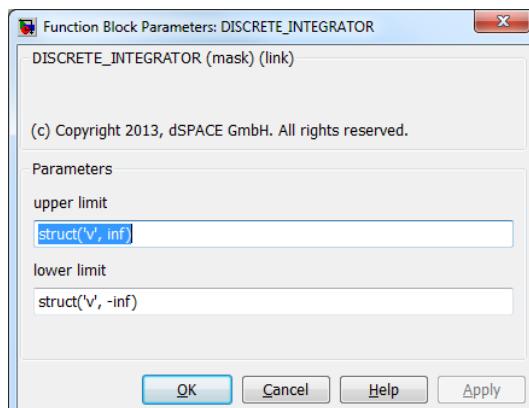


Figure 287: DISCRETE_INTEGRATOR dialog

The dialog has the following parameters:

Name	Unit	Description
upper_limit	[-]	Upper limit for integrator saturation
lower_limit	[-]	Lower limit for integrator saturation

Input

The DISCRETE_INTEGRATOR block has the following inputs:

Name	Unit	Description
Reset	[-]	Reset the integrator
In	[-]	Input value
Initial	[-]	Initial value
T_Sample	[s]	Sample time

Output

The block has the following outputs:

Name	Unit	Description
Out	[-]	Time discrete integration of the input signal

PI controller

Objective

A proportional–integral controller (PI controller) is a control loop feedback mechanism widely used in electrical drive control systems.

A PI controller attempts to correct the error between a measured variable and a desired set point by calculating and then outputting a corrective action.

The next figure shows the principle diagram of the model.

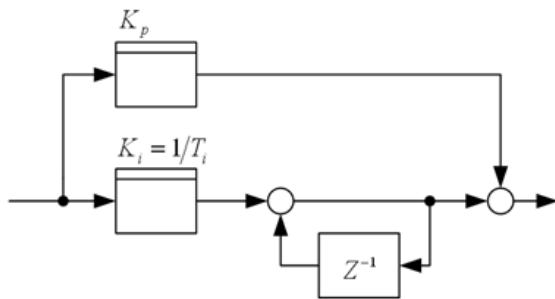


Figure 288: principle diagram of a PI Controller

The implemented PI Controller has an internal saturation with anti-wind up. The minimal and maximal can be set in the block mask. The procedure here use, at the end of the PI algorithm examined, whether the PI controller is overridden. If so, the proportion of the I- controller will calculate that the PI controller just to control the border.

The next illustration shows the Simulink representation of PI Controller.

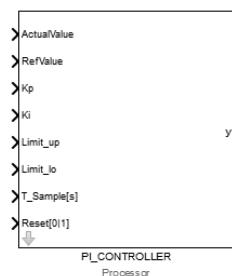


Figure 289: PI_CONTROLLER block

Block Dialog

The dialog provides only a short block description.

Input

The PI_CONTROLLER block has the following inputs:

Name	Unit	Description
ActualValue	[-]	Actual value
RefValue	[-]	Reference value
Kp	[-]	Proportional factor
Ki	[-]	Integral factor
Limit_up	[-]	Upper output limit
Limit_lo	[-]	Lower output limit
T_Sample	s	Sample time
Reset	[-]	Reset the controller

Output

The block has the following outputs:

Name	Unit	Description
y	[-]	Control variable

PI Controller with external saturation

Objective

This Controller works like the controller described above. The only difference is that the saturation must be designed externally from the PI controller (For example vector voltage limitation by using the PI controller as a d- or q- current controller. The procedure here uses, at the end of the PI algorithm examined, whether the PI controller is overridden. If so, the proportion of the I-Controller will calculate that the PI controller just controls the border.

The next illustration shows the Simulink representation of PI controller.

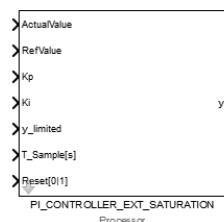


Figure 290: PI_CONTROLLER_EXT_SATURATION block

Block Dialog

The dialog provides only a short block description.

Input

The PI_CONTROLLER_EXT_SATURATION block has the following inputs:

Name	Unit	Description
ActualValue	[-]	Actual value
RefValue	[-]	Reference value
Kp	[-]	Proportional factor
Ki	[-]	Integral factor
y_limited	[-]	Control variable after external limit
T_Sample	[s]	Sample time
Reset	[-]	Reset the controller

Output

The block has the following outputs:

Name	Unit	Description
y	[-]	Control variable

Space vector modulation

Objective

This block presents a space vector modulator e.g. used in PMSM Controller, to generate the duty cycles for an inverter bridge as shown in the figure bellow.

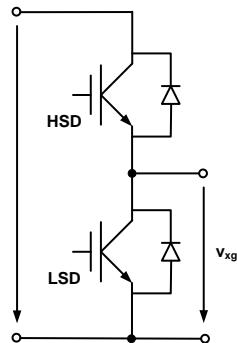


Figure 291: Schematic of the inverter bridge for the SPACE_VECTOR_MODULATOR block

The next illustration shows the Simulink representation of space vector modulation.

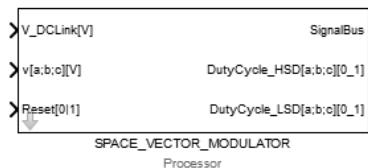


Figure 292: SPACE_VECTOR_MODULATOR block

Block Dialog

The dialog provides only a short block description.

Input

The SPACE_VECTOR_MODULATOR block has the following inputs:

Name	Unit	Description
V_DCLink	[V]	DC Link Voltage
v	[V]	Voltages of the three phases
Reset	[-]	Reset the output

Output

The block has the following outputs:

Name	Unit	Description
SignalBus	Bus	Bus containing the most significant signals of this block
DutyCycle_HSD[a;b;c]	[-]	Duty cycle of the high side driver (phase a;b;c)
DutyCycle_LSD[a;b;c]	[-]	Duty cycle of the low side driver (phase a;b;c)

Three Level Space vector modulation

Objective

This presents a space vector modulator, to generate the duty cycles for an three level inverter bridge as shown in the figure bellow.

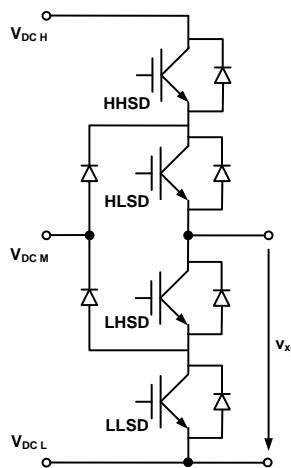


Figure 293: Schematic of the three level inverter bridge for the THREE_LEVEL_SPACE_VECTOR_MODULATOR block

The next illustration shows the Simulink representation of three level space vector modulation.

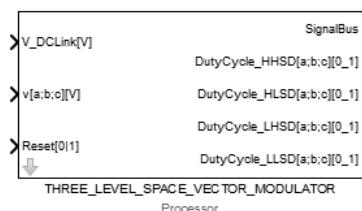


Figure 294: THREE_LEVEL_SPACE_VECTOR_MODULATOR block

Block Dialog

The dialog provides only a short block description.

Input

The THREE_LEVEL_SPACE_VECTOR_MODULATOR block has the following inputs:

Name	Unit	Description
V_DCLink	[V]	DC Link Voltage
v	[V]	Voltages of the three phases
Reset	[-]	Reset the output

Output

The block has the following outputs:

Name	Unit	Description
SignalBus	Bus	Bus containing the most significant signals of this block
DutyCycle_HHSD[a;b;c]	[-]	Duty cycle of the high side driver (phase a;b;c)
DutyCycle_HLSD[a;b;c]	[-]	Duty cycle of the upper low side driver (phase a;b;c)
DutyCycle_LHSD[a;b;c]	[-]	Duty cycle of the lower high side driver (phase a;b;c)
DutyCycle_LLSD[a;b;c]	[-]	Duty cycle of the low side driver (phase a;b;c)

PMSM Controller Basic

Objective

This controller is design to control a PMSM machine. It contains 2 current controllers (d-axis and q-axis), a speed controller and a field weakening controller. The PMSM Controller can be configured in different ways:

- Voltage control
- Current control
- Speed control

All parameters and settings can be configured by input ports.

The next illustration shows the Simulink representation of PMSM controller basic.

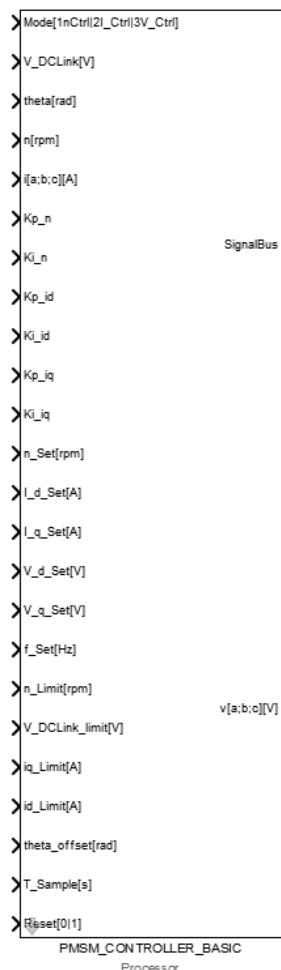


Figure 295: PMSM_CONTROLLER_BASIC block

Block Dialog

The blockset contains the following dialog.

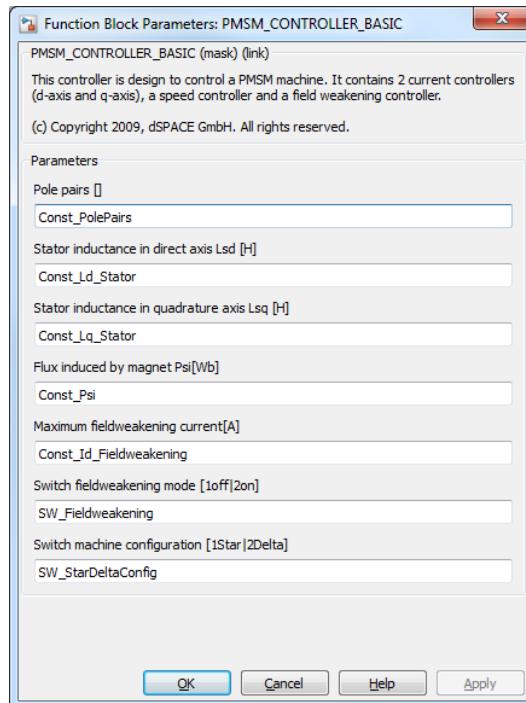


Figure 296: PMSM_CONTROLLER_BASIC dialog

The dialog has the following parameters:

Name	Unit	Description
Const_PolePairs	[-]	Pole pairs
Const_Ld_Stator	[H]	Stator inductance in direct axis Lsd
Const_Lq_Stator	[H]	Stator inductance in quadrature axis Lsq
Const_Psi	[Wb]	Flux induced by magnet Psi
Const_Id_Fieldweakening	[A]	Maximum fieldweakening current
SW_Fieldweakening	[-]	Switch fieldweakening mode [1off 2on]
SW_StarDeltaConfig	[-]	Switch machine configuration [1star 2delta]

Input

The PMSM_CONTROLLER_BASIC block has the following inputs:

Name	Unit	Description
Mode	[-]	Switch between the three control modes [1nCtrl 2I_Ctrl 3V_Ctrl]
V_DClink	[V]	DC link voltage
theta	[rad]	Mechanic motor angle
n	[rpm]	Motor speed
i[a;b;c]	[A]	Currents of the controlled motor
Kp_n	[-]	Kp n controller
Ki_n	[-]	Ki n controller
Kp_id	[-]	Kp id controller
Ki_id	[-]	Ki id controller
Kp_iq	[-]	Kp iq controller
Ki_iq	[-]	Ki iq controller
n_Set	[rpm]	Setpoint in speed control mode
I_d_set	[A]	Setpoint in current control mode
I_q_set	[A]	Setpoint in current control mode
V_d_set	[V]	Setpoint in voltage control mode
V_q_set	[V]	Setpoint in voltage control mode
f_Set	[Hz]	Frequency for outputvoltages (V_Ctrl mode)
n_Limit	[rpm]	Speed limit
V_DClink_Limit	[V]	DC Link limit
Limit_id	[A]	Maximum current in direct axis
Limit_iq	[A]	Maximum current in quadrature axis
theta_offset	[rad]	Theta offset
T_Sample	[s]	Sample time
Reset	[-]	Reset the output

Output

The block has the following outputs:

Name	Unit	Description
SignalBus	Bus	Bus containing the most significant signals of this block
v[a;b;c]	[V]	Motor voltages

PMSM Controller

Objective

The PMSM controller combines the PMSM controller basic and the space vector modulator.

For information please refer to PMSM controller basic and space vector modulator.

The next illustration shows the Simulink representation of PMSM Controller.

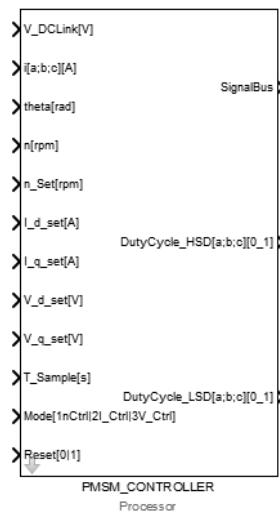


Figure 297: PMSM_CONTROLLER block

Block Dialog

The blockset contains the following dialog.

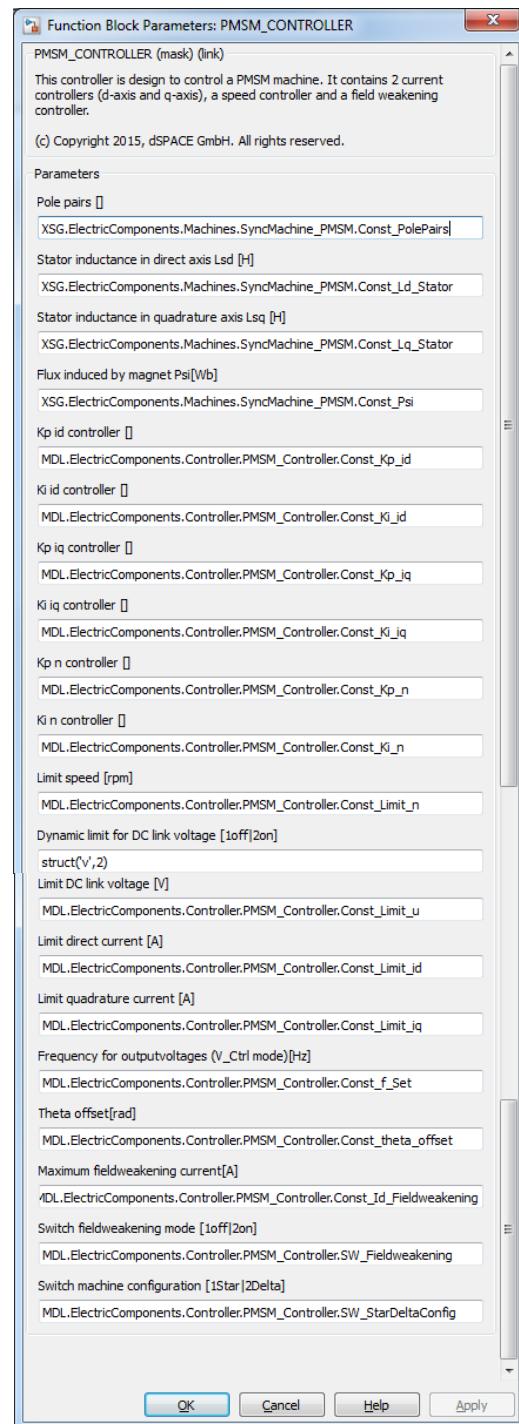


Figure 298: PMSM_CONTROLLER dialog

The dialog has the following parameters:

Name	Unit	Description
Const_PolePairs	[-]	Pole pairs
Const_Ld_Stator	[H]	Stator inductance in direct axis Lsd
Const_Lq_Stator	[H]	Stator inductance in quadrature axis Lsq
Const_Psi	[Wb]	Flux induced by magnet Psi
Const_Kp_id	[-]	Kp id controller
Const_Ki_id	[-]	Ki id controller
Const_Kp_iq	[-]	Kp iq controller
Const_Ki_iq	[-]	Ki iq controller
Const_Kp_n	[-]	Kp n controller
Const_Ki_n	[-]	Ki n controller
Const_Limit_n	[rpm]	Speed limit
Dynamic limit for DC link voltage	[-]	Specify if the limit for the maximum DC link voltage depends on the input port V_DCLink (maximum DC Link voltage is set to 4/5 of the input port) or fix value [1off 2on]
Limit DC link voltage	[V]	Limit of the maximum DC Link voltage (fix value)
Const_Limit_id	[A]	Maximum current in direct axis
Const_Limit_iq	[A]	Maximum current in quadrature axis
Const_Limit_u	[V]	Maximum output voltage
Const_f_Set	[Hz]	Frequency for outputvoltages (V_Ctrl mode)
Const_theta_offset	[rad]	Theta offset
Const_Id_Fieldweakening	[A]	Maximum fieldweakening current
SW_Fieldweakening	[-]	Switch fieldweakening mode [1off 2on]
SW_StarDeltaConfig	[-]	Switch machine configuration [1star 2delta]

Input

The PMSM_CONTROLLER block has the following inputs:

Name	Unit	Description
V_DCLink	[V]	DC link voltage
i[a;b;c]	[A]	Currents of the controlled motor
theta	[rad]	Mechanic motor angle
n	[rpm]	Motor speed
n_Set	[rpm]	Setpoint in speed control mode
I_d_set	[A]	Setpoint in current control mode
I_q_set	[A]	Setpoint in current control mode
V_d_set	[V]	Setpoint in voltage control mode
V_q_set	[V]	Setpoint in voltage control mode
T_Sample	[s]	Sample time
Mode	[-]	Switch between the three control modes [1nCtrl 2I_Ctrl 3V_Ctrl]
Reset	[-]	Reset the output

Output

The block has the following outputs:

Name	Unit	Description
SignalBus	Bus	Bus containing the most significant signals of this block
DutyCycle_HSD[a;b;c]	[-]	Duty cycle of the high side driver (phase a;b;c)
DutyCycle_LSD[a;b;c]	[-]	Duty cycle of the low side driver (phase a;b;c)

Three Level PMSM Controller

Objective

The PMSM controller combines the PMSM controller basic and the three level space vector modulator.

For information please refer to PMSM controller basic and three level space vector modulator.

The next illustration shows the Simulink representation of PMSM Controller.

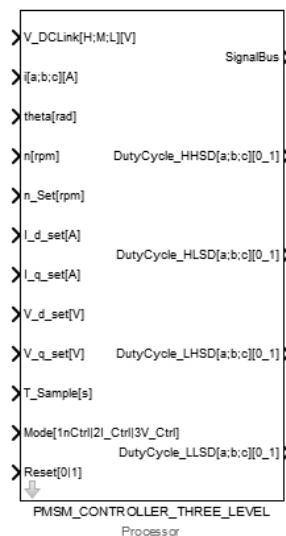


Figure 299: PMSM_CONTROLLER_THREE_LEVEL block

Block Dialog

The blockset contains the same dialog than the PMSM_CONTROLLER. For detailed information please refer to the PMSM_CONTROLLER.

Input

The PMSM_CONTROLLER_THREE_LEVEL block has the following inputs:

Name	Unit	Description
V_DClink[H;M;L]	[V]	DC link voltage of the three levels
i[a;b;c]	[A]	Currents of the controlled motor
theta	[rad]	Mechanic motor angle
n	[rpm]	Motor speed
n_Set	[rpm]	Setpoint in speed control mode
I_d_set	[A]	Setpoint in current control mode
I_q_set	[A]	Setpoint in current control mode
V_d_set	[V]	Setpoint in voltage control mode
V_q_set	[V]	Setpoint in voltage control mode
T_Sample	[s]	Sample time
Mode	[-]	Switch between the three control modes [1nCtrl 2l_Ctrl 3V_Ctrl]
Reset	[-]	Reset the output

Output

The block has the following outputs:

Name	Unit	Description
SignalBus	Bus	Bus containing the most significant signals of this block
DutyCycle_HHSD[a;b;c]	[-]	Duty cycle of the high side driver (phase a;b;c)
DutyCycle_HLSD[a;b;c]	[-]	Duty cycle of the upper low side driver (phase a;b;c)
DutyCycle_LHSD[a;b;c]	[-]	Duty cycle of the lower high side driver (phase a;b;c)
DutyCycle_LLSD[a;b;c]	[-]	Duty cycle of the low side driver (phase a;b;c)

Demo

Overview

Overview

The XSG Electric Components library provides a demo of a permanent magnet synchronous machine (PMSM) in closed loop operation.

It consists of a Matlab Simulink model and a ControlDesk Next Generation project backup.



The XSG Electric Components Demo requires the XSG Electric Components Interface library and the XSG Electric Components library.

The demo is precompiled for the following hardware configurations:

Processor Model	FPGA Model
DS1006	
DS1007	
DS2502	DS2655 (7K160) + DS2655M1 Module

In case that a different hardware configuration should be used, e.g. DS5203 (7K325) with mounted multi IO module (DS5203M1), it is necessary to rebuild the FPGA model.



Note that the demo model is configured for an offline simulation. For starting a FPGA build process please refer to the XSG Utils documentation chapter offline simulation.

Demo Installation

Initial Installation

The Demo Model setup is separated from the XSG Electric Components installation. To install the demo it is necessary to open the XSG Electric Components library and double click the demo block. The following dialog will appear:



Figure 300: Demo installation dialog (first installation)

The Demo model and experiment will be installed to: My Documents\ dSPACE\ XSG_EC\ 20.1. Hence, the demo folder installation path, shown in the demo installation dialog will adjust, according to the user's login name.

Demo Folder Structure

The demo is copied, using the following folder structure:

- XX.YY.Z
- Demos
- PHS
- Instrumentation
- Simulation
- SCLX
- Configuration
- Instrumentation
- Simulation

Figure 301: Demo folder structure after demo model installation

The Simulation folder contains the Matlab Simulink model.

The Instrumentation folder contains the Control Desk Project Backup.

If Scalexio platform is used an additional folder Configuration contains the Configuration Desk Project Backup.

Reinstallation

After the initial demo installation is done, the install demo dialog will change. The dialog now provides the options to open the demo model, or to reinstall the demo, as shown in the figure below.

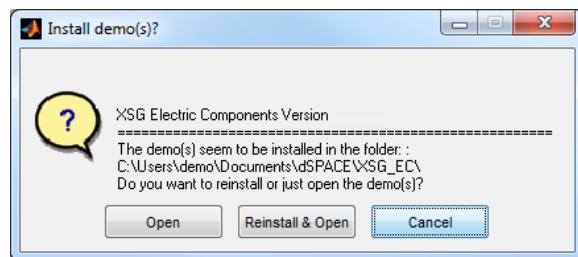


Figure 302: Demo installation dialog (after initial installation)

	Any changes on the demo model and experiment will be lost, when reinstalling the demo!
-----------------------------------------------------------------------------------	----------------------------------------------------------------------------------------

PMSM Closed Loop Demo

Overview

The PMSM Closed Loop demo shows an exemplary integration of a permanent magnet synchronous motor simulation utilizing the XSG Electric Components library.

For this demo implementation, different main components from the XSG Electric Components library are used. These are the APU, Resolver, the PMSM motor model, the special designed inverter for the PMSM motor, a PWM measurement, a PWM generation and to capture various kinds of signals of the FPGA with the Multi-Scope function. To realize a scaling of the DAC signals on runtime the Multi-Scale-DAC function is also implemented.

Processor Model

To get a better overview of the general model structure the following schematic illustration and the Simulink model root is shown in the next figures.

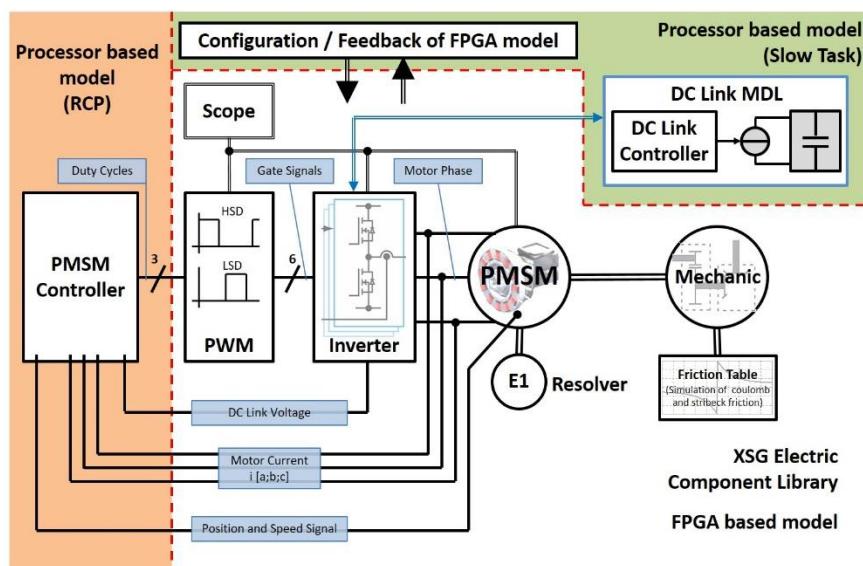


Figure 303: Schematic overview of the model structure

XSG Electric Components Library Electric Drive PMSM Demo for PHS-Bus System

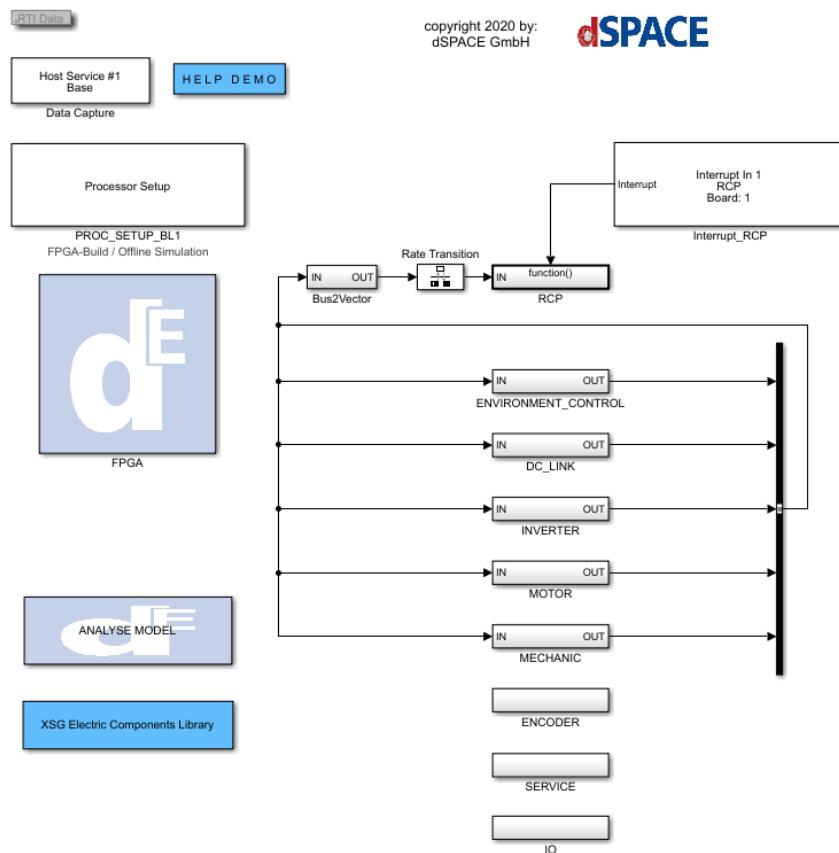


Figure 304: PMSM demo Simulink model root

The processor setup block is located in the top left corner of the model.

The FPGA model is placed below the processor setup block.

Some useful functions, like a shortcut to change the solver type from fix to variable step, the analyse model function or some shortcuts to libraries are located below the FPGA model.

The overall model is separated in the parts RCP, Environment Control, DC_Link, and eDrive components.

In the part Environment control only control variables like the setpoint of the DC link voltage, the setpoint of speed, current and the selection of the mechanical model (simulation on processor or FPGA) is included. If the mechanic model is switched the overall model will be set to initial state.

In the next part (subsystem DC_Link) a DC Link model is included.

In the last part the eDrive components (Inverter, Motor, Encoder, IO and Multiscope) are included.

In this model different timer and interrupt triggered tasks are included. The model parts which are located at the bottom are sampled in the slow task.

Each model has its own subsystem with the specific processor <-> FPGA model interface to configure the model parameters. Additional subsystems like capturing internal variables or the scaling of the IO channels are also included. If the RTI process is not running, the scaling functionality detects a non-running application on the real time processor and disables the output of the DS5203 board. The actual result of this detection is given out by the LED of the DS5203.

In addition, the Library_Version_Info block is used to ensure that processor and FPGA model are built with the same XSG Electric Components library version. The Library_Version_Info block is located in the subsystem SERVICE.

Model variables (e.g. the Reset signal of the outer subsystem Environment Control) which are necessary from one to another subsystem (e.g. Average) are routed over the overall Bus.

The implemented different timer and interrupt triggered tasks of the processor based part of the model carries the following functionalities:

- PMSM Controller Task (RCP):

The PMSM Controller task (RCP) includes the PMSM controller of the motor. The internal structure of the subsystem is shown in the figure below.

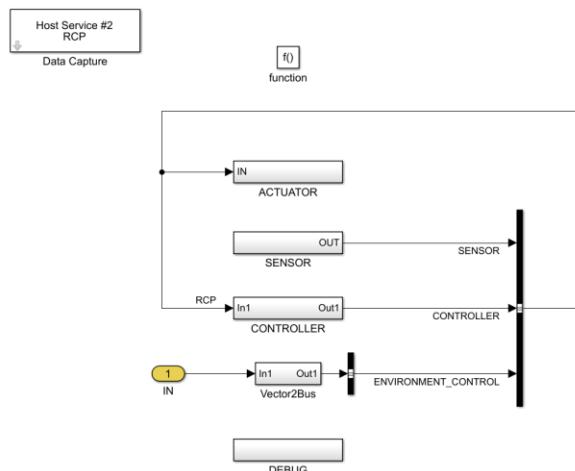


Figure 305: Model root of the subsystem RCP

In this subsystem a simulation of a current measurement and a simulated sensor feedback for speed and position are included. The Controller measures with these inputs the actual motor current, the actual speed and position of the motor and provides the necessary duty cycles of each motor phase. To generate a PWM signal from these duty cycle the PWM-Generator (located in the subsystem THREE_PHASE_PWM_GENERATOR) of the XSG Utils library is used.

To illustrate the possibility of an implementation of a frequency derating the setpoint of the PWM frequency of the controller depends in this case on the actual

speed of the motor. Implementation of user defined derating functions are also possible.

If no PWM is generated by the PWM generator block the RCP task is triggered with a default interrupt with a frequency of 1 kHz. The available control modes of the PMSM controller are current and speed control.

FPGA Model

The FPGA model is located on the PMSM demo model root, below the processor setup block. The FPGA model is shown in the following figure:

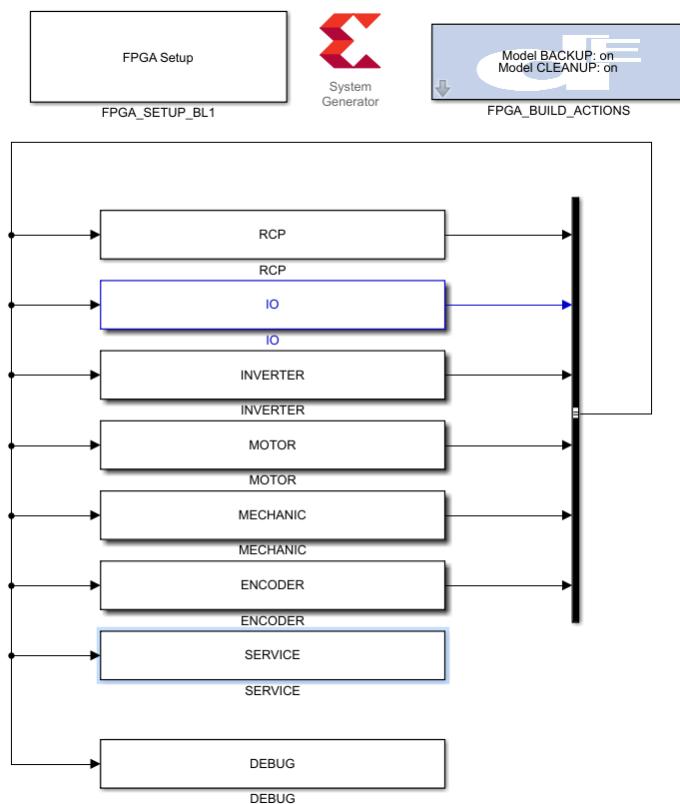


Figure 306: PMSM demo FPGA model

The FPGA setup block is in the top left corner of the FPGA model. Here the specification of desired hardware platform (framework) has to be made. All hardware in- and outputs are in the subsystem IO. Each subsystem represents a fully implemented function like the MECHANIC, a MOTOR, ... Model variables (e.g. the speed of the internal APU of the MECHANIC) which are necessary from one subsystem (e.g. MECHNAINC) to another subsystems (e.g. ENCODER) and the hardware IO signals are routed over the overall Bus.

DS5203 IO Mapping

The following table summarizes the model signals, which are mapped to DS5203 IO channels:

DS5203 IO Channel	Model Signal / Function
ADC 1	Resolver Excitation Input / FPGA Sine Generator out
DAC 1	Motor current phase a
DAC 2	Motor current phase b
DAC 3	Motor current phase c
DAC 4	Resolver Sine Output
DAC 5	Resolver Cosine Output
DAC 6	DC Link voltage
DIG IO 1	Digital Input: high side of PWM phase a (external ECU)
DIG IO 2	Digital Input: low side of PWM phase a (external ECU)
DIG IO 3	Digital Input: high side of PWM phase b (external ECU)
DIG IO 4	Digital Input: low side of PWM phase b (external ECU)
DIG IO 5	Digital Input: high side of PWM phase c (external ECU)
DIG IO 6	Digital Input: low side of PWM phase c (external ECU)
LED out	<p>Red: no model is running on the FPGA</p> <p>Green: model is running on the FPGA and realtime processor model is also running</p> <p>Yellow: Scaling functionality detects a non-running processor model → output ports of the DS5203 are set to logical TTL level zero</p>

DS2655 IO Mapping

The following table summarizes the model signals, which are mapped to DS2655 M1 IO channels:

DS2655 M1 IO Channel	Model Signal / Function
Analog In – Ch: 11 [Mod: 1]	Resolver Excitation Input / FPGA Sine Generator out
Analog Out – Ch: 16 [Mod: 1]	Motor current phase a
Analog Out – Ch: 17 [Mod: 1]	Motor current phase b
Analog Out – Ch: 18 [Mod: 1]	Motor current phase c
Analog Out – Ch: 19 [Mod: 1]	Resolver Sine Output
Analog Out – Ch: 20 [Mod: 1]	Resolver Cosine Output
Digital InOut – Ch: 1 [Mod: 1]	Digital Input: high side of PWM phase a (external ECU)
Digital InOut – Ch: 2 [Mod: 1]	Digital Input: low side of PWM phase a (external ECU)
Digital InOut – Ch: 3 [Mod: 1]	Digital Input: high side of PWM phase b (external ECU)
Digital InOut – Ch: 4 [Mod: 1]	Digital Input: low side of PWM phase b (external ECU)
Digital InOut – Ch: 5 [Mod: 1]	Digital Input: high side of PWM phase c (external ECU)
Digital InOut – Ch: 6 [Mod: 1]	Digital Input: low side of PWM phase c (external ECU)
Digital Out – Ch: 10 [Mod: 1]	Feedback if the realtime processor model is running: logical TTL level 1: model is running; logical TTL level 0: model is not running
LED out	Red: no model is running on the FPGA Green: model is running on the FPGA and realtime processor model is also running Yellow: Scaling functionality detects a non-running processor model → output ports of the DS5203 are set to logical TTL level zero

ControlDesk Environment

To open the prepared ControlDesk backup, first start ControlDesk. Then, select File - Open – Project + Experiment from Backup.

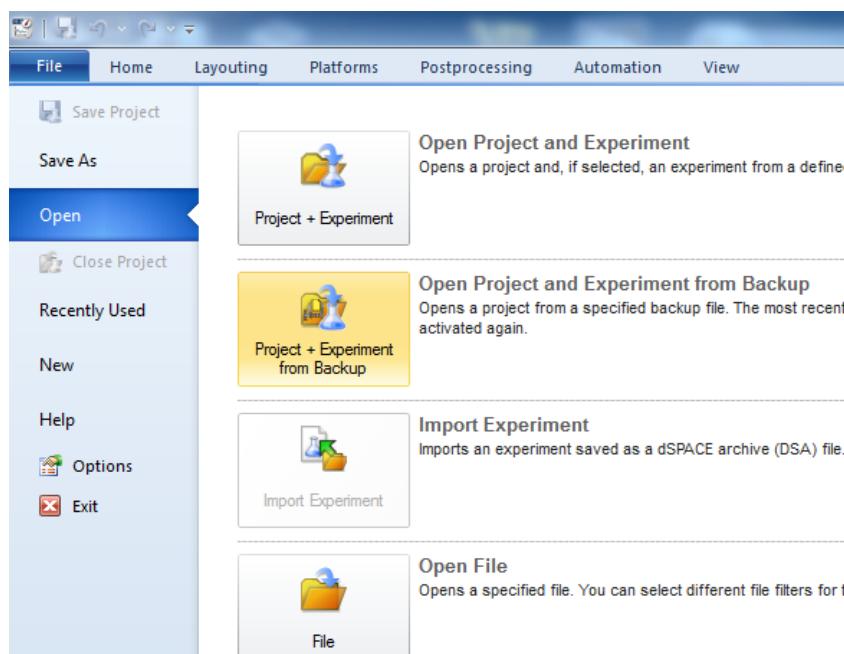


Figure 307: Open ControlDesk backup

Navigate to the demo installation directory at: My Documents\ dSPACE\ XSG_EC\ 20.1\ Demos\ <(PHS/SCLX)>\ Instrumentation\ and select the PMSM_<(PHS/SCLX)>_Demo.ZIP file.

After the backup is loaded, the following layouts of the inverter, motor and mechanical model and a configuration layout of the internal FPGA based scope is selectable.

Now, register the processor board and start the measurement. The real-time application should be downloaded. The different layouts show the specific parameters of each model. An overall quick control field is implemented on nearly every layout. The Controller is designed to control the motor in the following settings:

	<p>Note that the settings for the control mode and the specific setpoints of the PMSM motor controller can be set on nearly each. The maximum values of the implemented motor are:</p> <p>Current Control:</p> <ul style="list-style-type: none"> - maximum setpoint of the current in direct axis: +- 300 A - maximum speed: +- 3000 rpm <p>Speed Control:</p> <ul style="list-style-type: none"> - maximum setpoint of speed: +- 3000 rpm - maximum load torque: +- 250 Nm
-------------------------------------------------------------------------------------	--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

- Layout: Overview

To get a quick overview of the model the layout “Overview” can be used, as shown in the figure below.

To reach the specific function groups, (like Inverter; Controller; Motor; Scope; ...) Shortcuts are implemented in the layout. By double clicking on the shortcuts the specific layout of the section will be opened.

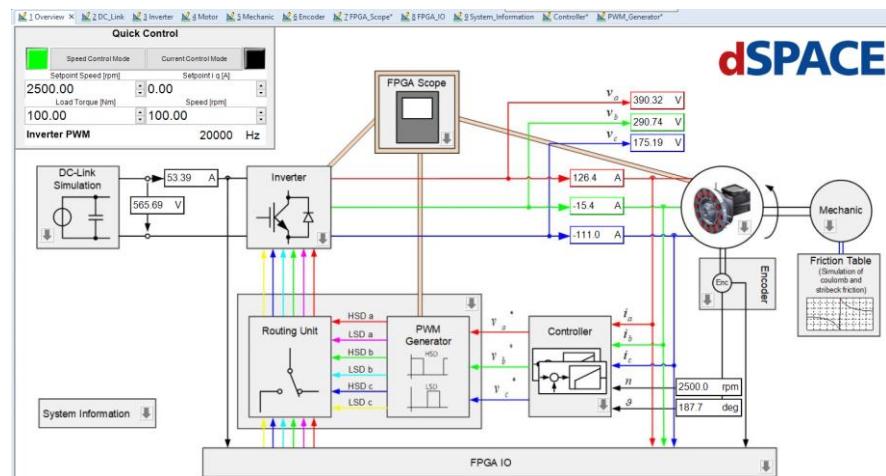


Figure 308: Layout: Overview

- Layout / Function Group: DC-Link Simulation

The layout of the DC-Link simulation is shown in the following figure.

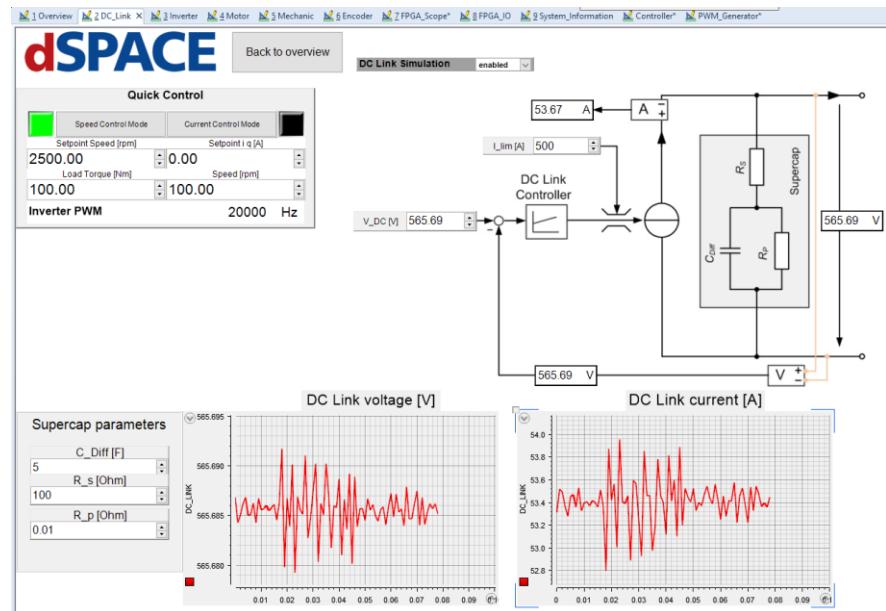


Figure 309: Layout: DC-Link Simulation

On left side of the layout the inverter model can be en- or disabled. The dc link voltage and current and a simplified model are plotted and presented. The dc link model can be switched direct via the popup. The settings of the implemented supercap condensator can be controlled or configured by the edit fields in the lower left side. By clicking on the Button "Back to overview" the active layout can be switched to the layout "Overview".

- Layout / Function Group: Inverter

The DC-Link supplies the implemented inverter as shown in the overview. By double clicking on the shortcut the Inverter layout will opened as shown in the figure below.

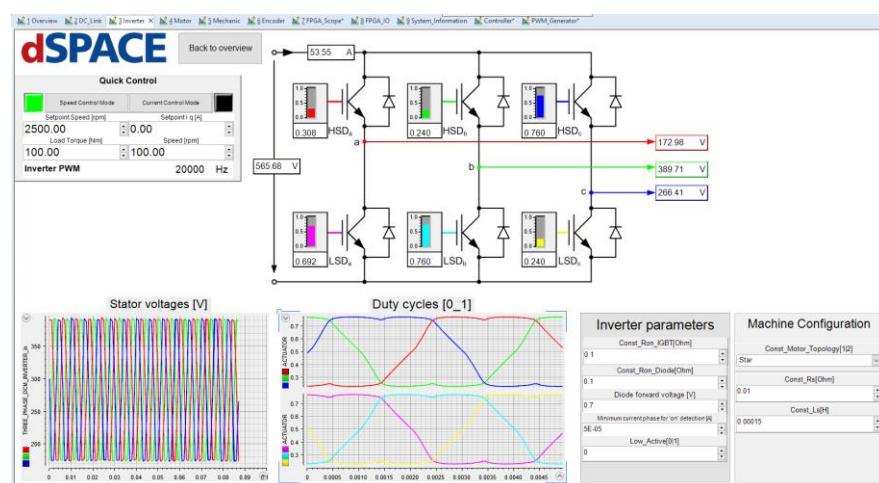


Figure 310: Layout: Inverter

On the upper left side, a simplified model is used to control the duty cycles and the corresponding output voltages of the inverter. The lower plotter shows the calculated average value of the duty cycles of the controller and the phase voltages. Here, the variation of the PWM signals, which is reasoned by the controller, directly takes effect to voltages, so that they seem to be noisy. By clicking on the Button "Back to overview" the active layout can be switched to the layout "Overview".

- Layout / Function Group: Motor

The implemented motor is connected to the inverter. By double clicking on the shortcut the Motor layout will opened as shown in the figure below.

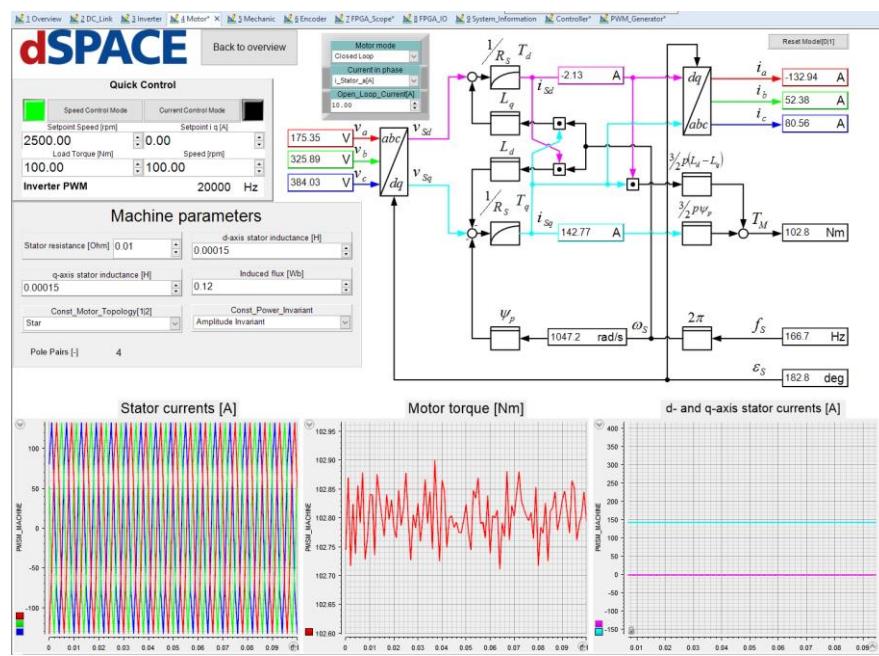


Figure 311: Layout: Motor

On the upper left side, a schematic of the implemented model is presented. The actual current in direct and quadrature axis, the motor torque, the electrical speed, the electrical circuit frequency, the position and the input voltages are shown in the schematic model structure. To check possible external wiring an additional OpenLoop switch can be activated which is located in the upper left corner.

On the middle and lower section of the layout the motor parameters like the magnetic flux, inductance or the resistance can be configured. The actual curve of the motor torque, the current in direct and quadrature axis is also presented in the plots. By clicking on the Button "Back to overview" the active layout can be switched to the layout "Overview".

- Layout / Function Group: Load

The motor generates a torque which accelerates a connected mechanic / load. By double clicking on the shortcut the Motor layout will open as shown in the figure below.

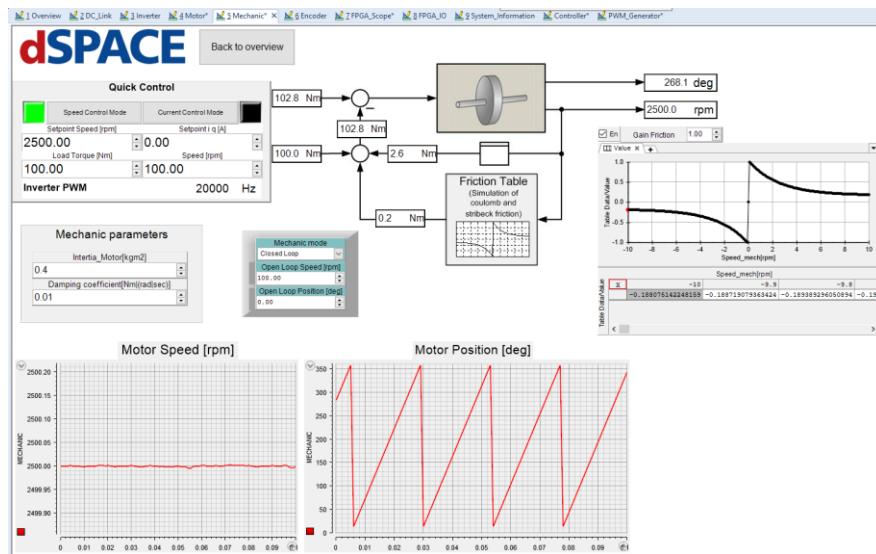


Figure 312: Layout: Load

In the left upper corner, the position of the simulated mechanic can be switched from a Processor or FPGA based. The schematic of the mechanic will be updated for both simulation types (Processor or FPGA). For both models the inertia parameters like the motor inertia and the damping constant can be configured. Additional to the mechanic model on the FPGA a friction table is inserted to simulate the coulomb, stribbeck and static friction effects which can be en- or disabled on the FPGA. The actual curve of the motor angle and motor speed is also presented in the plots in the right corner. By clicking on the Button "Back to overview" the active layout can be switched to the layout "Overview".

- Layout / Function Group: Encoder

To realize a feedback to the overall controller three encoder are implemented on the motor shaft. By double clicking on the shortcut the Encoder layout will open as shown in the figure below.

The Encoder have specific groups for configuration. Available configurations can be set in the specific edit fields.

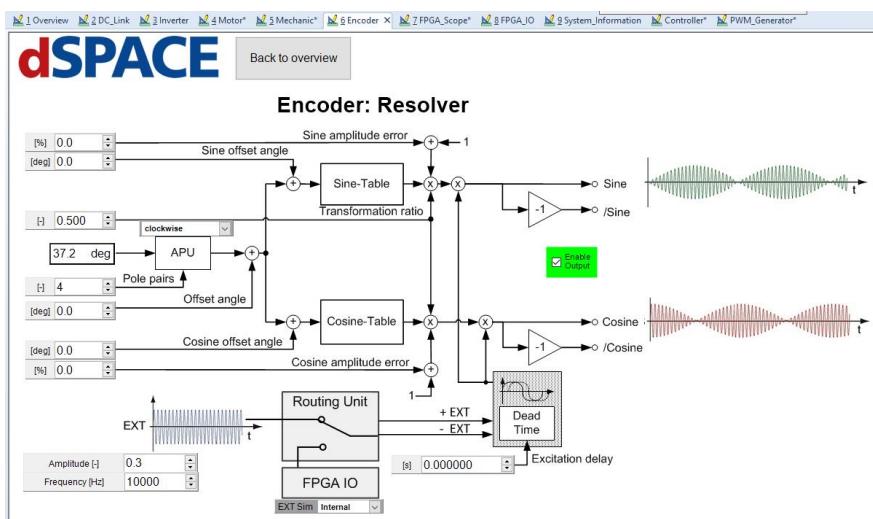


Figure 313: Layout: Encoder

- Layout / Function Group: Controller

With the position feedback of the motor shaft and the feedback of the actual phase currents of the motor the implemented controller realizes in different control modes of the motor. By double clicking on the shortcut the Controller layout will open as shown in the figure below.

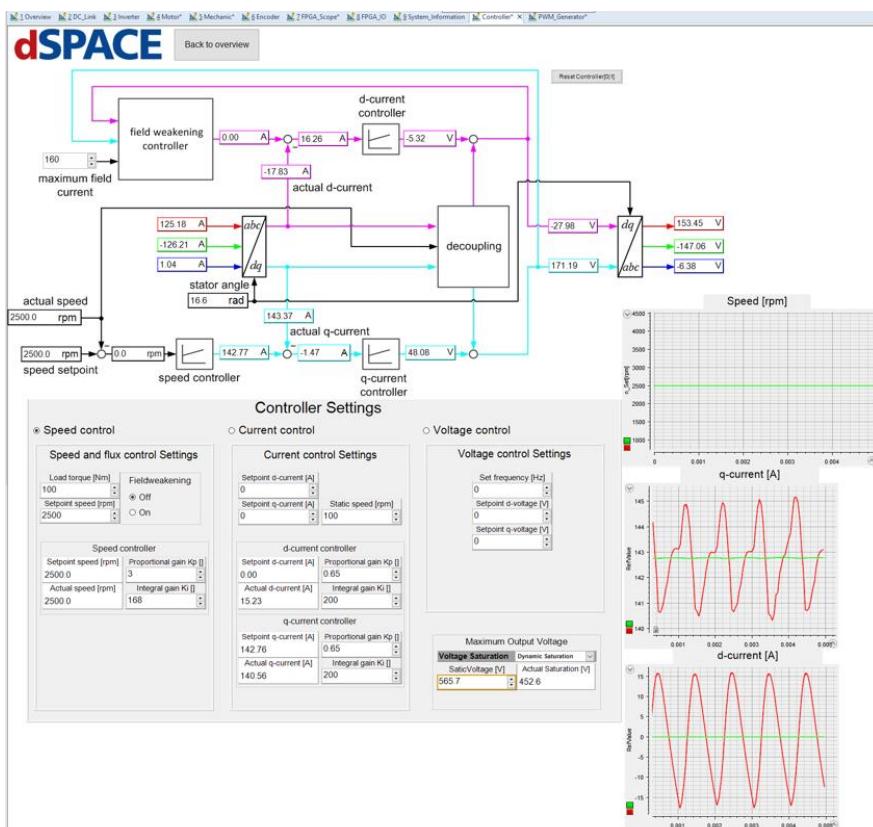


Figure 314: Layout: Controller

To get an easy access to the implemented controller a schematic is available in the upper left corner. The different control modes can be configured via the function groups in the lower middle of the layout. The actual speed and direct and quadrature axis currents are plotted in the lower right corner. By clicking on the Button "Back to overview" the active layout can be switched to the layout "Overview".

- Layout / Function Group: PWM Generator – Routing Unit

The duty cycle setpoints are sending to a PWM Generator which realizes the specific gate signals. In the connected Routing Unit the simulation of an external connected Motor controller can be switched. By double clicking on the shortcut the specific layout will open as shown in the figure below.

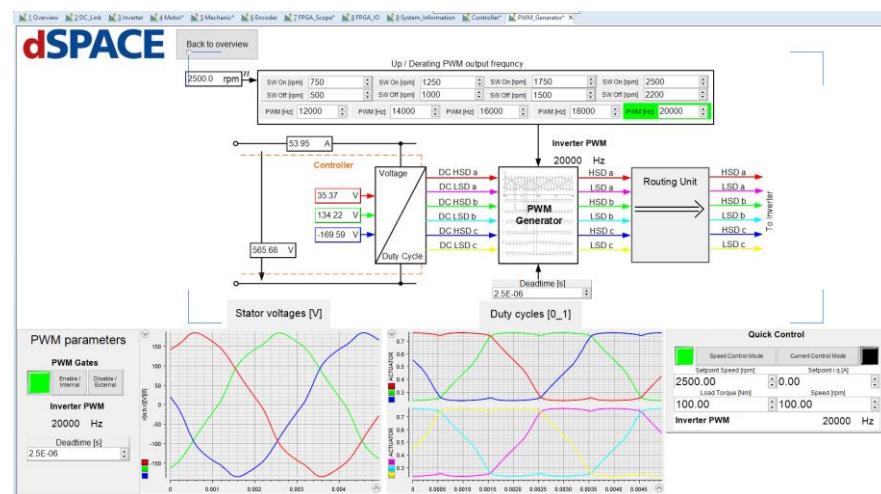


Figure 315: Layout: PWM Generator - Routing Unit

The PWM Generator simulates a speed depending derating or uprating of the PWM frequency. The levels of switching can be configured in the upper schematic of the layout. The generated gate signals are read back by the inverter and the path of the closed loop simulation is done. By clicking on the Button "Back to overview" the active layout can be switched to the layout "Overview".

Layout / Function Group: FPGA Scope

All specific signals can be captured via a scope which is implemented on the FPGA as shown in the following figure.

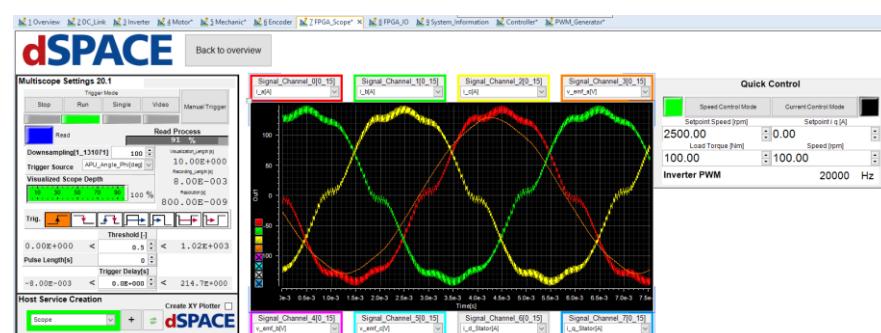


Figure 316: Layout: Scope

The layout consists of one XY-Plotter, which are triggered from the MultiScope block. Around the plotter the selection boxes are placed where the signals which will be displayed can be chosen (1 out of up to 16). On the lower left side, the settings of the scope are placed. It gives information about the actual status and provides the opportunity to specify the trigger condition. Either the trigger can be performed manually or it is related to one input variable of the FPGA. In this example the scope starts recording when the current in phase a crosses the value 0 in a positive direction. There is no pre- or post-trigger specified and the values are captured every 1 μ s (100 * 10ns). Also, the time to capture the data on the FPGA is displayed here (can be changed by adapting the downsample factor) as well as the time which is required to visualize the recorded data (duration time for the plotter capture service).

Layout / Function Group: FPGA IO

To change the scaling of the analog in- and outputs a function group FPGA IO is inserted as shown in the figure below.

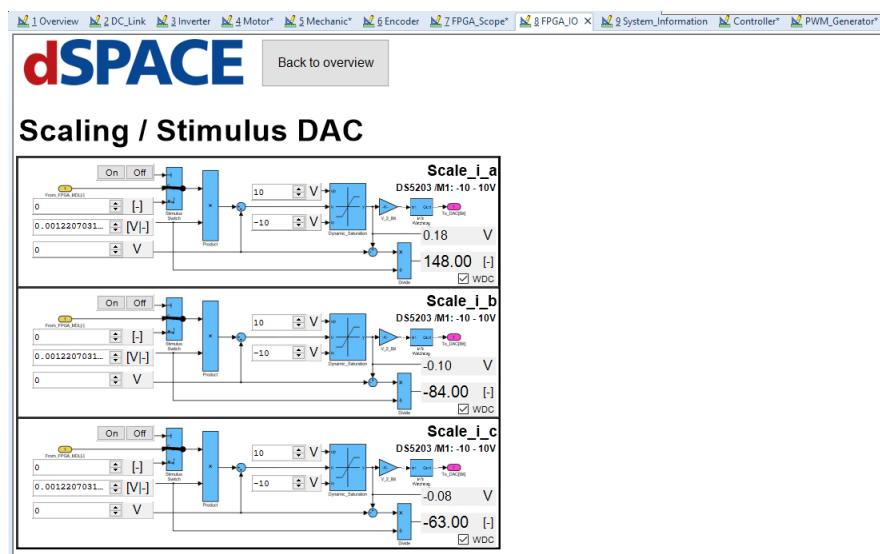


Figure 317: Layout: FPGA IO

Layout / Function Group: System Information

To check the used software version of the FPGA and the processor library and the turnaround time of the different tasks the layout of the system information can be used.

ConfigurationDesk Environment

In case of using an IOCNET system (SCALEXIO) an additional ConfigurationDesk environment is also available. To open the prepared ConfigurationDesk backup, first start ConfigurationDesk. Then, select File - Open – Project + Experiment from Backup.

Navigate to the demo installation directory at: My Documents\ dSPACE\ XSG_EC\ 20.1\ PMSM_ClosedLoop \SCLX\ Configuration\ and select the PMSM_ClosedLoop_SCLX_Demo.ZIP file.

Functions

Overview

Overview

The XSG Electric library provides several functions to simplify the work with an FPGA model. These functions are not linked to a specific processor or FPGA model part and can be only offline used.

The different functions of the both library parts are shown in the figure below.

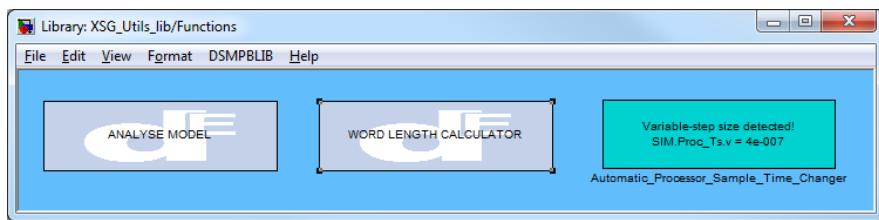


Figure 318: Functions of the XSG Electric library (shared with XSG Utils)

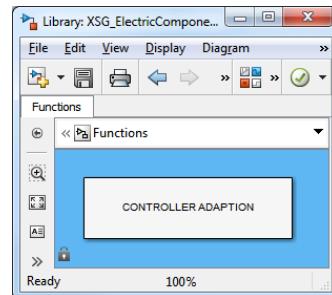


Figure 319: Functions of the XSG Electric Interface library

Additional to these, helpful matlab functions are also included which can be called from user code or the command window.

dSPACE Controller Adaption

Purpose	The controller adaption GUI is used to tune a controller for a known plant by three different approaches.
----------------	-----------------------------------------------------------------------------------------------------------

Introduction

Overview The controller can be adapted according to the following approaches:

- Amount optimum
- Symmetrical optimum
- Ziegler-Nichols (step response method)

The controller can be designed for the following plant types:

- First-order lag element (P-T1)
- Second-order lag element (P-T2)
- Delayed integrator (I-T1)
- First-order lag with dead time (P-T1-Tt)

Not all plant types can be adapted with each tuning method. The table below gives an overview of the valid tuning approaches for the plant types:

Plant type	Amount optimum	Symmetrical optimum	Ziegler-Nichols approach
P-T1	x		
P-T2	x	x*	x*
I-T1	x*	x	
P-T1-Tt	x*	x*	x*

* Certain conditions have to be fulfilled to apply the method for the given plant.

The controller tuning rules and the conditions to be fulfilled are described in the *Tuning Methods* chapter.

The controller is assumed as the PI controller. This guide refers to the following form:

$$G_C(s) = \frac{K_R}{s} \cdot \frac{T_R s + 1}{1}$$

Plant Types

Overview

This chapter describes the transfer functions and its parameters for each plant type.

The following plant types can be chosen:

- First-order lag element (P-T1)
- Second-order lag element (P-T2)
- Delayed integrator (I-T1)
- First-order lag element with dead time (P-T1-Tt)

P-T1

The transfer function of the first-order lag element in the Laplace domain is given as:

$$G(s) = \frac{K}{Ts + 1}$$

P-T2

There are two different options to be chosen for the parameterization of a second-order lag element. The transfer function parameters can be entered either as "real" P-T2 or by the parameters of a composed P-T2 by two P-T1 elements.

The transfer function in the Laplace domain of a P-T2 has the following form:

$$G(s) = \frac{K}{T^2 s^2 + 2dTs + 1}$$

The parameter d describes the damping of the plant. If the damping is equal to or larger than 1, the second-order lag element can be described by two first-order lag elements in series (composed P-T2). For damping lower than 1 the poles of the plant are conjugated complex.

The composed second-order lag element is determined by:

$$G(s) = \frac{K}{T_1 s + 1} \cdot \frac{1}{T_2 s + 1}$$

I-T1

The delayed integrator is composed by a first-order lag element and an integrator in series. The transfer function in the Laplace domain is given as:

$$G(s) = \frac{K}{T_1 s} \cdot \frac{1}{T_2 s + 1}$$

P-T1-Tt

The P-T1-Tt element is composed by a first-order lag element and a dead time in series. The transfer function in the Laplace domain is given as:

$$G(s) = \frac{K}{T_1 s + 1} \cdot e^{-sT_t}$$

Tuning Methods

Overview

There are three different controller tuning options which can be chosen:

1. Amount optimum
2. Symmetrical optimum
3. Ziegler-Nichols (step response method)

The tuning rule for each applicable plant type is described in the following.

1. Amount Optimum

Overview

The amount optimum tuning approach is applicable for all plant types listed in the previous chapter. The following sections describe the calculation rules for each plant type.

P-T1

Transfer function plant:

$$G(s) = \frac{K}{Ts + 1}$$

Transfer function controller:

$$G_c(s) = \frac{K_R}{s} \cdot \frac{T_R s + 1}{1}$$

Calculation rule:

$$\begin{aligned} T_R &= T \\ K_R &= \frac{10}{2 \cdot T \cdot K} \end{aligned}$$



K_R can be chosen arbitrarily. The closed-loop system is stable for any K_R . The K_R is determined without an exact calculation rule with regard to the amount optimum method.

P-T2

Transfer function plant:

$$G(s) = \frac{K}{T^2 s^2 + 2dTs + 1}$$



The damping of the plant has to be sufficient. If the damping d is lower than 1, the amount optimum method might fail and lead to instability. (Foellinger, 1994)

Transfer function controller:

$$G_C(s) = \frac{K_R}{s} \frac{T_R s + 1}{1}$$

Case 1:

If the transfer function is composed by two first-order lag elements in series and the large time constant is 5 times larger than the smaller time constant, the controller is tuned according to the rule below.

Tuning condition (Schroeder, 2009):

$$T_1 > 4T_2$$

Calculation rule (Foellinger, 1994):

$$T_R = T_1$$

$$K_R = \frac{1}{2 \cdot K \cdot T_2}$$



If the larger time constant is more than 4 times larger than the smaller time constant, it is recommended to use the symmetrical optimum instead. (Schroeder, 2009)

Case 2:

If the condition above is not fulfilled, the controller parameters are determined by another tuning rule. Therefore, the plant parameters have to be converted into the following form (Foellinger, 1994):

$$G(s) = \frac{1}{a_2 s^2 + a_1 s + a_0}$$

The parameters are calculated by (Foellinger, 1994):

$$r_0 = a_0 \cdot \frac{a_1^2 - a_0 a_2}{a_1 a_2}$$

$$r_1 = a_1 \cdot \frac{a_1^2 - a_0 a_2}{a_1 a_2} - a_0$$

$$T_R = \frac{r_1}{r_0}$$

$$K_R = \frac{r_0}{2}$$

$$G(s) = \frac{K}{T_1 s} \cdot \frac{1}{T_2 s + 1}$$

Transfer function controller:

$$G_C(s) = \frac{K_R}{s} \cdot \frac{T_R s + 1}{1}$$



The tuning condition below has to be fulfilled to apply the amount optimum for an I-T1 plant.

Tuning condition (Schroeder, 2009):

$$\frac{T_1}{K T_2} \gg 1$$

The plant parameters have to be converted into the following form (Foellinger, 1994):

$$G(s) = \frac{1}{a_2 s^2 + a_1 s + a_0}$$

The parameters are calculated by (Foellinger, 1994):

$$r_0 = 2 \cdot 10^{-6} \quad (\text{chosen to avoid division through zero})$$

$$r_1 = a_1 \cdot \frac{a_1^2 - a_0 a_2}{a_1 a_2} - a_0$$

$$T_R = \frac{r_1}{r_0}$$

$$K_R = \frac{r_0}{2}$$

The resulting controller is a P controller due to neglectable I-part (10^{-6}).

P-T1-Tt

Transfer function:

$$G(s) = \frac{K}{T_1 s + 1} \cdot e^{-s T_t}$$

Transfer function controller:

$$G_C(s) = \frac{K_R}{s} \cdot \frac{T_R s + 1}{1}$$



The tuning condition below has to be fulfilled to apply the amount optimum for a P-T1-Tt plant.

Tuning condition:

$$T_t < T_1$$

Tuning rule (Schroeder, 2009):

$$T_R = T_1$$

$$K_R = \frac{T_1}{2 \cdot K \cdot T_t}$$

2. Symmetrical Optimum

Overview

The amount optimum tuning approach is applicable for all plant types listed in the previous chapter, except for the first-order lag element. The following sections describe the calculation rules for each plant type.

P-T2

Transfer function plant:

$$G(s) = \frac{K}{T^2 s^2 + 2dT s + 1}$$



The damping of the plant has to be larger than 1. Otherwise, the plant will have conjugated complex poles and the symmetrical optimum approach will not be applicable. (Foellinger, 1994)

Tuning condition:

$$d \geq 1$$

Therefore, the plant has to be composed by two first-order lag elements and the transfer function can be rewritten as:

$$G(s) = \frac{K}{T_1 s + 1} \cdot \frac{1}{T_2 s + 1}$$

Transfer function controller:

$$G_C(s) = \frac{K_R}{s} \cdot \frac{T_R s + 1}{1}$$

It is assumed that T_1 is the larger time constant.



If the larger time constant is less than 4 times larger than the smaller time constant, it is recommended to use the amount optimum instead. (Schroeder, 2009)

Tuning rule (Schroeder, 2009):

$$T_R = k_1 \cdot 4 \cdot T_2$$

$$K_R = k_2 \cdot \frac{T_1}{2 \cdot K \cdot T_2 \cdot T_R}$$

The correction factors consider the ratio between both time constants. They are determined by (Schroeder, 2009):

$$k_1 = \frac{1 + \left(\frac{T_2}{T_1}\right)^2}{\left(1 + \frac{T_2}{T_1}\right)^3}$$

$$k_2 = 1 + \left(\frac{T_2}{T_1}\right)^2$$

I-T1

Transfer function plant:

$$G(s) = \frac{K}{T_1 s} \cdot \frac{1}{T_2 s + 1}$$

Transfer function controller:

$$G_C(s) = \frac{K_R}{s} \cdot \frac{T_R s + 1}{1}$$

The parameters are calculated by (Schroeder, 2009):

$$T_R = 4 \cdot T_2$$

$$K_R = \frac{T_1}{2 \cdot K \cdot T_2}$$

P-T1-Tt

Transfer function:

$$G(s) = \frac{K}{T_1 s + 1} \cdot e^{-sT_t}$$

Transfer function controller:

$$G_C(s) = \frac{K_R}{s} \cdot \frac{T_R s + 1}{1}$$



The rising time has to be larger than the delay time. Otherwise, the Ziegler-Nichols approach will not be applicable (Foellinger, 1994).

Tuning condition:

$$T_t < T_1$$



If the larger time constant is less than 4 times larger than the smaller time constant, it is recommended to use the amount optimum instead. (Schroeder, 2009)

Calculation rule (Schroeder, 2009):

$$T_R = k_1 \cdot 4 \cdot T_t$$

$$K_R = k_2 \cdot \frac{T_1}{2KT_t T_R}$$

The correction factors consider the ratio between both time constants. They are determined by (Schroeder, 2009):

$$k_1 = \frac{1 + \left(\frac{T_t}{T_1}\right)^2}{\left(1 + \frac{T_t}{T_1}\right)^3}$$

$$k_2 = 1 + \left(\frac{T_t}{T_1}\right)^2$$

3. Ziegler-Nichols Method

Overview

The Ziegler-Nichols tuning approach is applicable for a delayed first-order lag element or a second-order lag element for certain conditions. The following sections describe the calculation rules for both of the plant types. There are two different methods to tune a controller with the Ziegler-Nichols approach. Here, only the step response method is used in accordance with (Foellinger, 1994).

P-T2

Transfer function plant:

$$G(s) = \frac{K}{T^2 s^2 + 2dT s + 1}$$



The damping of the plant has to be larger than 1. Otherwise, the plant Ziegler-Nichols approach will not be applicable (Foellinger, 1994).

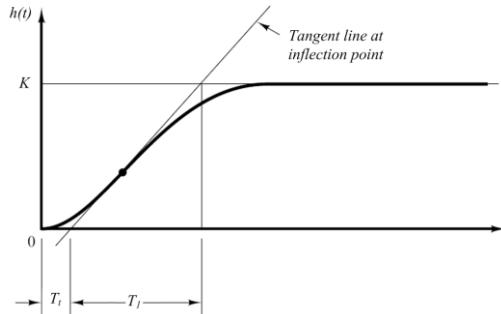
Tuning condition:

$$d \geq 1$$

Transfer function controller:

$$G_C(s) = \frac{K_R}{s} \cdot \frac{T_R s + 1}{1}$$

For the tuning controller the step response of the plant has to be recorded. A tangent at the inflection point of the step response has to be drawn to determine the rising time and the delay time for the controller tuning.



**Figure 320: Step-response method for the Ziegler-Nichols tuning approach
cf. (Ogata, 2010 (5th Edition))**



The rising time has to be larger than the delay time. Otherwise, the Ziegler-Nichols approach will not be applicable. (Foellinger, 1994)

Tuning condition:

$$T_1 > T_t$$

Tuning rule (Foellinger, 1994):

$$T_R = 3.33 \cdot T_t$$

$$K_R = 0.9 \cdot \frac{T_1}{T_t \cdot K \cdot T_R}$$

P-T1-Tt

Transfer function:

$$G(s) = \frac{K}{T_1 s + 1} \cdot e^{-sT_t}$$

Transfer function controller:

$$G_C(s) = \frac{K_R}{s} \cdot \frac{T_R s + 1}{1}$$



The rising time has to be larger than the delay time. Otherwise, the Ziegler-Nichols approach will not be applicable (Foellinger, 1994).

The following tuning condition has to be fulfilled (Foellinger, 1994):

$$T_t < T_1$$

Tuning rule (Foellinger, 1994):

$$T_R = 3.33 \cdot T_t$$

$$K_R = 0.9 \cdot \frac{T_1}{T_t \cdot K \cdot T_R}$$

Controller Tuning GUI

Overview

The controller tuning GUI can be opened by double clicking on the block CONTROLLER_ADAPTION as shown in the figure below.

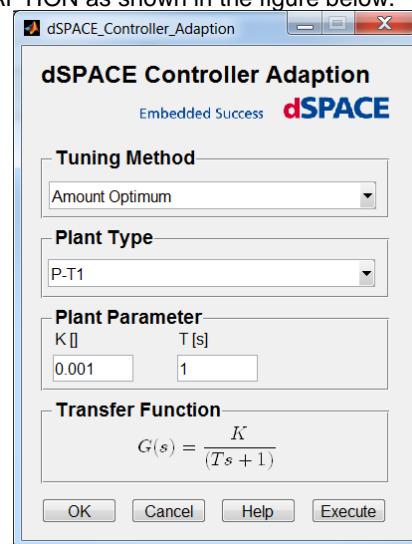


Figure 321: Controller Tuning GUI

Parameters

Tuning methods:

- Amount optimum
- Symmetrical optimum
- Ziegler-Nichols

Depending on the chosen plant type the plant parameters to be entered change dynamically. The following plant types can be chosen and the corresponding parameters have to be entered.

P-T1	K	T	
Real P-T2	K	T	d
Composed P-T2	K	T ₁	T ₂
I-T1	K	T ₁	T ₂

P-T1-Tt	K	T ₁	T _t
---------	---	----------------	----------------

The controller is tuned according to the chosen options by clicking on "Execute" or "OK". If no errors occur a report with the tuning results is generated. If "OK" is clicked the UI is closed and at "Execute" the GUI remains open. The help can be opened by the "Help" button und "Cancel" will close the UI without controller tuning.

Output report

The output report is generated automatically if the specific tuning conditions are fulfilled and no error occurs. It is saved under

Controller_Tuning_Year_Month_Day_Hour_Minute.

An example report is illustrated below:

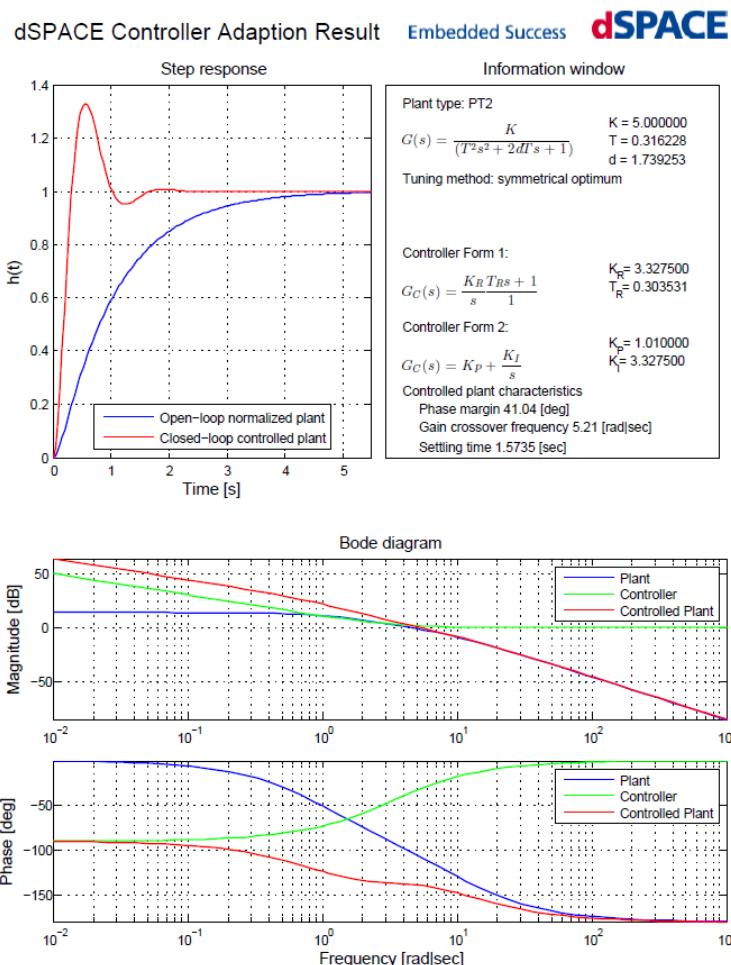


Figure 322: Output report with tuning results

The step response of the closed-loop controlled plant is shown in the top left as well as the normalized (with plant gain K normalized) open-loop step response of the plant. In the top right the information window provides information about the

plant transfer function, the tuning method, the controller parameters in two different forms and the characteristics of the controlled plant.

Additionally, there is an indication if a warning occurred at the controller tuning process. The warning message can be seen in detail in the MATLAB command window (see Figure 323):

```
Tuning warning: The plant is low damped. Amount optimum method can fail and lead to instability.
ft >>
```

Figure 323: Tuning warning example

The Bode diagram with the magnitude and phase are illustrated in the bottom half of Figure 322.

Example: Cascade Controller for a Permanent Magnet Synchronous Machine

Overview

This chapter shows an example to illustrate how a cascade controller structure of a permanent magnet synchronous motor (PMSM) can be tuned by the Controller Tuning GUI.

A cascade controller structure has to be tuned from the innermost control loop to the outermost one. In this case, the controller structure consists of an inner current control loop and the outer speed control loop.

Current control loop

The plant to be controlled by the current controller consists basically of the motor inductance (inductivity L and resistance R), which can be described as a first-order lag element. The parameters can be calculated by:

$$K = \frac{1}{R}$$

$$T = \frac{L}{R}$$

But there are often some further time constants (smaller than the time constant of the inductance) in the plant. Typically, the dead time of the inverter and the delay of the sample and hold element for the current measurement and possible measurement filter time constants have to be taken into account. Therefore, the current control plant can be approximated as P-T1-Tt or as composed P-T2, whereby the smaller time constants can be added up to one time constant (T2 or Tt).

The current controller can be tuned by the amount optimum method for good reference behavior or the symmetrical optimum method for good disturbance behavior. The Ziegler-Nichols approach can be applied also. For a further explanation of the current controller tuning see (Schroeder, 2009).

Speed control loop

The speed control plant consists of the closed current control loop and the rotational mechanical part in series. The mechanical part with the inertia and the friction can be described as a first-order lag element or as a simple integrator.

The first-order lag element is determined by:

$$K = \frac{1}{F}$$

$$T = \frac{J}{F}$$

If the mechanics are regarded as the integrator the time constant is calculated by:

$$T = J$$

The closed current control loop is the second part of the speed control plant. To keep the effort for the controller tuning low, the current control plant is regarded as a first-order lag element with one equivalent time constant T_{eq} .

$$G_{CurrentControlLoop}(s) = \frac{1}{T_{eq}s + 1}$$

To determine the equivalent time constant of the current control loop the settling time of the current control loop can be taken into account. It can be multiplied by a factor between 1 and 2. A higher factor results in a more stable behavior and a smaller factor in a faster system response.

$$T_{eq} = 1 \dots 2 T_{SettlingTime\ Current\ Control\ Loop}$$

Further time constants, such as measurement delay times or sample and hold dead times, can be added to the smaller time constant as well.

Additionally the motor constant, which describes the relation between the torque-generating current and the torque, has to be considered:

$$K_M = \frac{i}{T} = \frac{[A]}{[Nm]}$$

Thus, the speed control plant can be described as an I-T1 element or a P-T2 element (composed of two P-T1 elements).

$$G_{SpeedControlPlant}(s) = K_M \cdot G_{CurrentControlLoop}(s) \cdot G_{Mechanic}(s)$$

It is recommended to use the symmetrical optimum approach for the controller tuning to achieve a good disturbance behavior of the closed-loop plant.

Example

A PMSM with the following characteristics should be tuned:

$$L = 45.83mH$$

$$R = 3.48\Omega$$

Current measurement dead time:

$$T_t = 70\mu\text{s}$$

The current controller is tuned with the amount optimum method. The result report is illustrated in Figure 324.

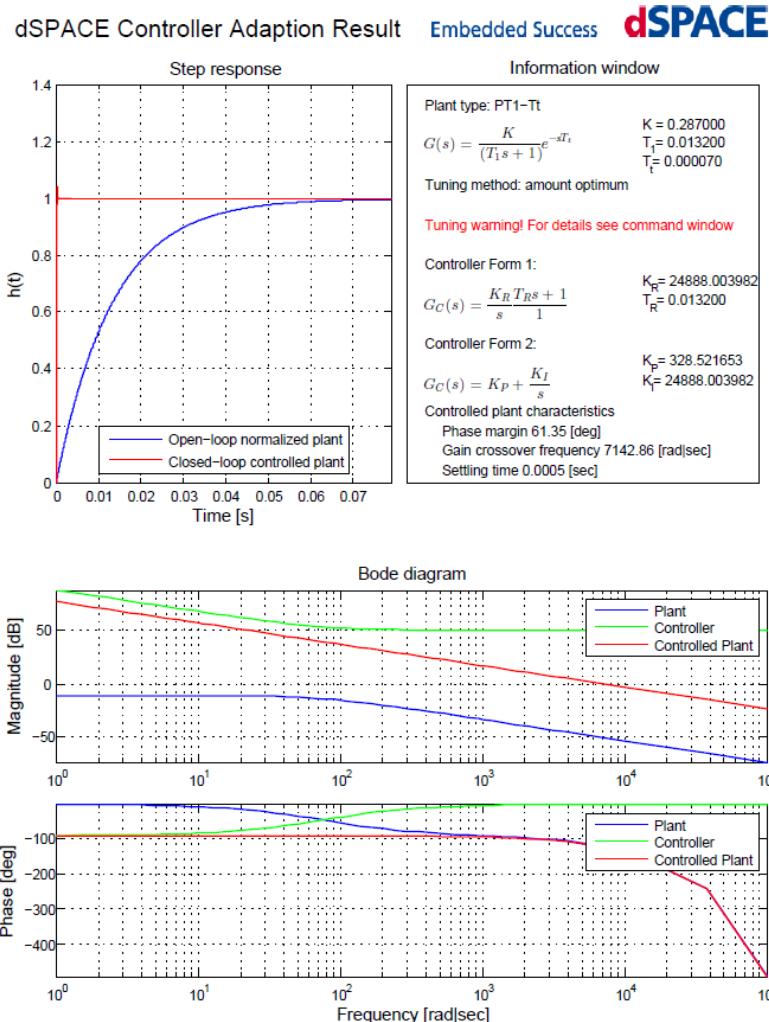


Figure 324: Current controller tuning

	A tuning warning is displayed in the report, because the larger time constant is more than 4 times larger than the smaller time constant. According to (Schroeder, 2009) it is recommended to apply the symmetrical optimum method. Nevertheless, the amount optimum method is used in this case. The tuning method has to be chosen individually for each use case, depending on the desired behavior.
	The inner current control loop should be tested on the real plant. The controller parameters can be optimized with regard to the desired behavior. It might be more appropriate to determine the

	settling time (used for tuning the speed controller) with the step response of the real current plant.
--	--------------------------------------------------------------------------------------------------------

The equivalent time constant of the closed-loop current control plant is calculated by the settling time of the current control closed-loop step response and a factor of 2 (can be chosen between 1...2):

$$G_{CurrentControlLoop}(s) = \frac{1}{0.001s + 1}$$

The inertia is given by:

$$J = 0.0046 \text{ kg m}^2$$

The ratio between the torque-generating current and the torque is determined as:

$$K_M = 5.1 \frac{[A]}{[Nm]}$$

The delay time of the speed measurement is given as:

$$T_t = 70\mu s$$

which can be added to the smaller time constant.

Thus, the simplified transfer function of the speed control can be written as:

$$G(s) = \frac{5.1}{0.0046s} \cdot \frac{1}{0.001s + 1}$$

In order to achieve good disturbance behavior, the symmetrical optimum method is used for the tuning:

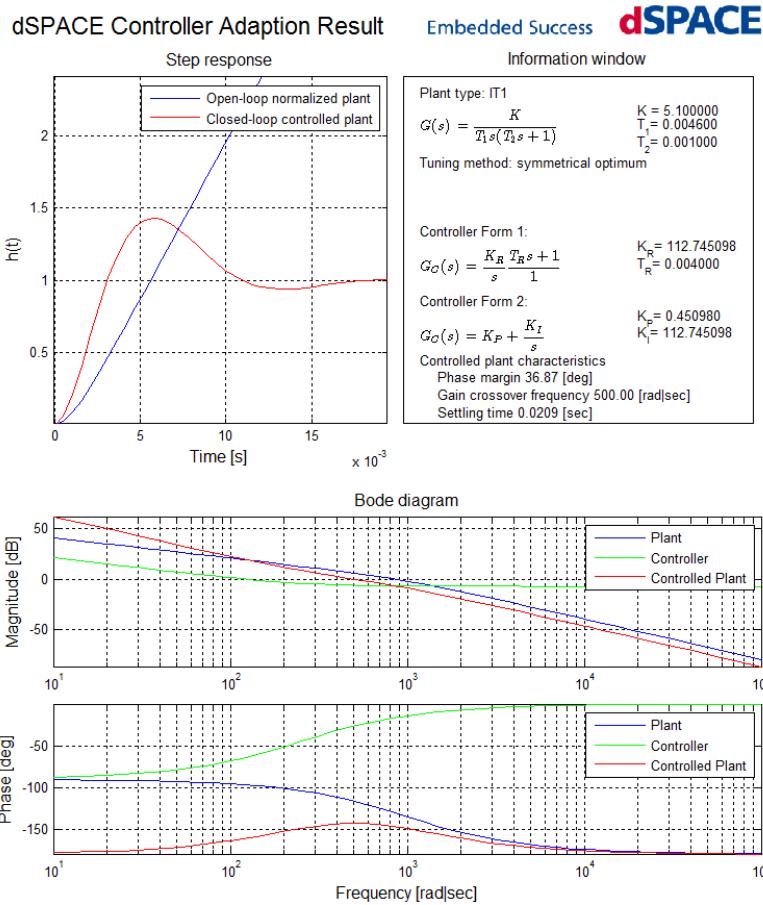


Figure 325: Speed controller tuning