XSG Sent Library

Version 20.1 – June 2020

Embedded Success 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

Contents

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_SENT.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 a free programmable dSPACE FPGA Board:

- MATLAB & Simulink
- Xilinx ISE including XSG
- dSPACE Release

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

RTI FPGA Programming Blockset	dSPACE Release	Operating System	MATLAB	Xilinx Design Tools
3.9	2020-A (64-bit)	Windows 7 Windows 10 (64-bit)	R2018b R2019a R2019b (64-bit)	Vivado 2019.2 (64-bit)

Figure 1: Compatibility matrix for XSG Sent 20.1

dSPACE Release	Operating System	MATLAB
2020-A	Windows 7 Windows 10 (64-bit)	R2018b R2019a R2019b R2020a (64-bit)

Figure 2: Compatibility matrix for XSG Sent Interface 20.1

The XSG Sent Library is requires Xilinx Design Tools which will not be sold by dSPACE

It is extremely recommended to secure that only one XSG Sent library version is used with one Matlab!

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.

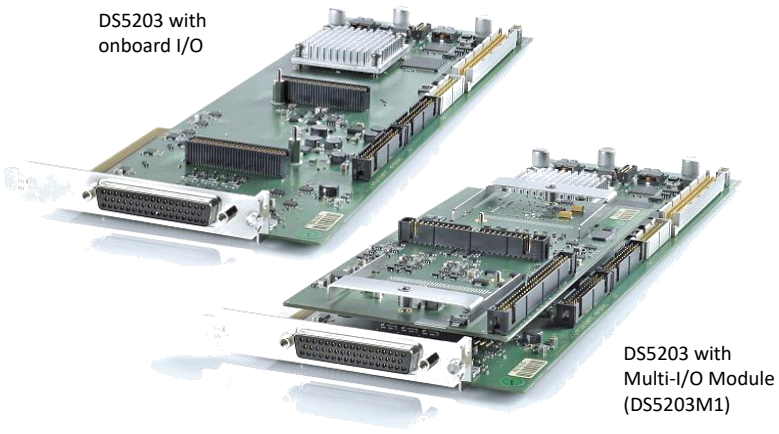


Figure 3: DS5203 Boards (PHS-based)

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 which can be programmed via the Xilinx synthesis tool called **Vivado**:

- Kintex7 K410 (more FPGA recourses than Kintex7 K325)
- Kintex7 K325
-

	The XSG_SENT Solution has been specially been designed, based on IO and performance, for DS5203 7K325. By taking the differences of IO and FPGA clock rate into account it can be used on other free programmable FPGA boards of dSPACE.

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, 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.

The DS1552 I/O module features the following I/O:

- 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

	The XSG_SENT Solution has been specially been designed, based on IO and performance, for DS5203 7K325. By taking the differences of IO and FPGA clock rate into account it can be used on other free programmable FPGA boards of dSPACE.

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.

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

	The XSG_SENT Solution has been specially been designed, based on IO and performance, for DS5203 7K325. By taking the differences of IO and FPGA clock rate into account it can be used on other free programmable FPGA boards of dSPACE.

DS2655 FPGA Board + DS2655M1 I/O Module

Board description

With the DS2655 FPGA base module and the DS2655M1 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.

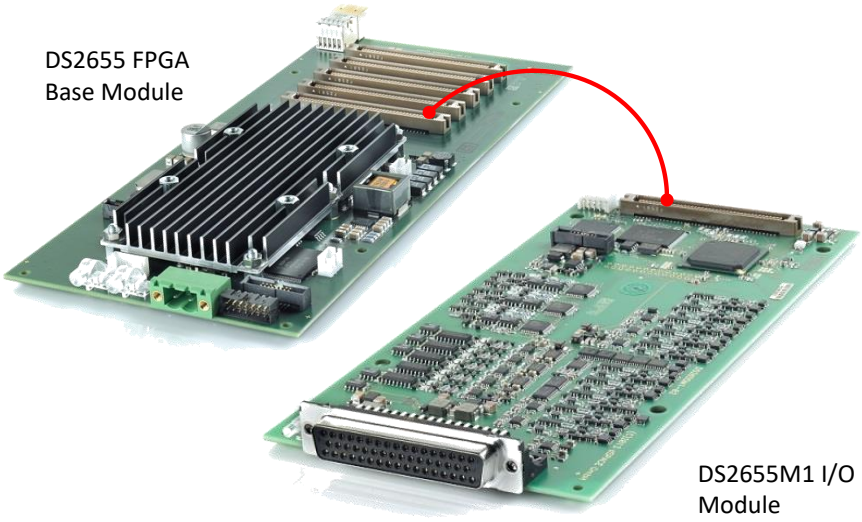


Figure 6: DS2655 FPGA base module and the DS2655M1 I/O module (SCALEXIO)

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. The first module available for the DS2655 is the DS2655M1 and features the following I/O:

- 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. However, the processing of SSI sensors is not possible with the DS2655M1, as it required RS485 or RS422 digital I/O drivers. An additional I/O module, providing those drivers, has to be plugged in order to perform SSI processing.

	The XSG_SENT Solution has been specially been designed, based on IO and performance, for DS5203 7K325. By taking the differences of IO and FPGA clock rate into account it can be used on other free programmable FPGA boards of dSPACE.

Simulink Model Structure

General Description

Description

This chapter describes the user interface for accessing the FPGA Board from Simulink. In general, there are two different ways to access the boards, either via PHS bus (for DS5203) / IOCNET (for DS2655) 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 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 or model port 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.

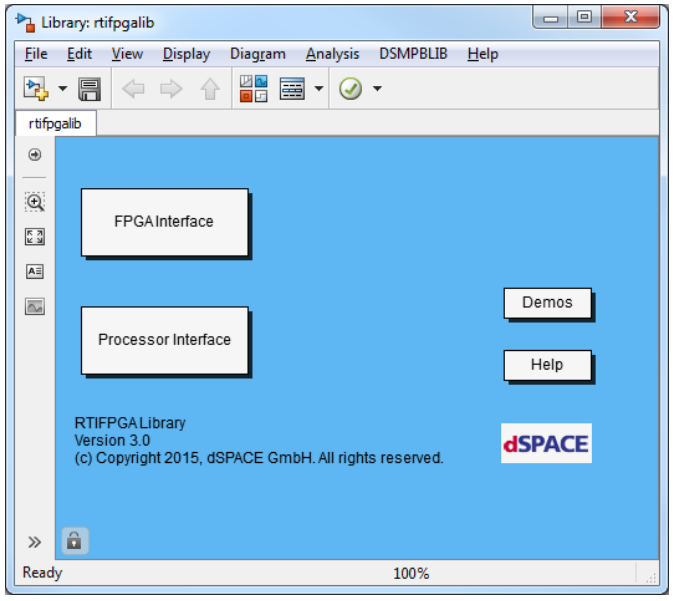


Figure 7: RTIFPGA Library

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

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 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 registers
5. **Processor in interface** (<component>_in (Processor Interface))
   The merged signals from the FPGA side are decoded on the processor side

Example of Interfacing

Example: Register Access

The illustration below shows a structural example of how to implement the interfaces, between Processor and FPGA, with usage of register access of RTI. The red blocks (1-5) are components from the dSPACE XSG Sent Library. The blue blocks (6-9) are RTI interface register blocks that define the basic communication between FPGA, processor and I/O.

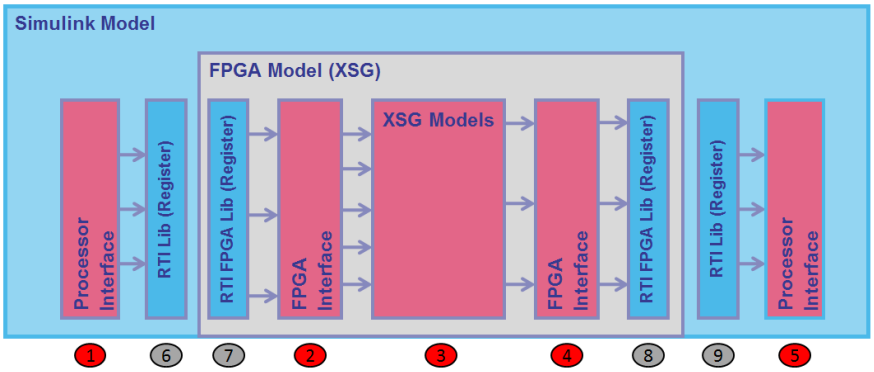


Figure 8: Using library components with register access

A detailed view of the RTI interface is shown in the next figure below. In this example, the values calculated by the processor are written via PHS bus to the FPGA (on the left-hand side) and the values calculated on the FPGA are written back to the processor (right-hand side). The I/O access is also shown (ADC1, DAC1).

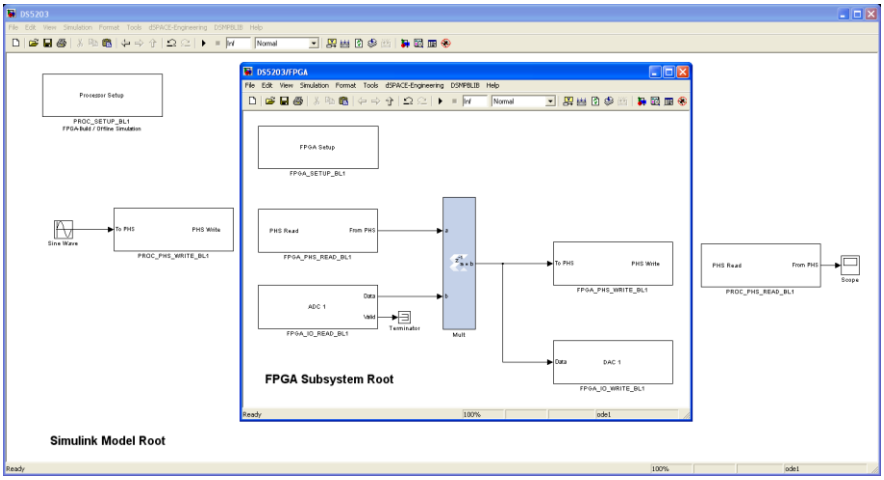


Figure 9: RTI interface usage with register access

The Processor Setup Block

Features

The Processor Setup block has to be located on the model root. All processor interface blocks can be placed anywhere inside the Simulink model except the FPGA subsystem. This block is only necessary for PHS-Bus based systems.

The number of FPGA boards which are used in the application has to be defined on the unit page, which is called double-clicking the processor setup block PROC_SETUP_BL1. Up to 16 FPGA boards can be handled. The subsystem, or precompiled binary file, has to be specified for each board. On the FPGA side the configuration can be stored in either the RAM or the flash memory.

The Interface lets you generate the interface blocks on the processor side automatically. Therefore only the interface blocks on the FPGA side have to be configured.

The Model Configuration page lets you switch between the FPGA Build and the Processor Build mode. For offline simulation, the FPGA build option must be used. Changes take effect when the Switch model mode button is clicked. If the Processor Build mode is selected, the FPGA subsystem is removed from the model file and stored inside a separate model file. When you switch back to FPGA Build mode, the FPGA subsystem is copied back to the application model.

When performing the build process on the processor side, it is mandatory to activate the Processor Build mode.

On the Advanced page, precompiled FPGA applications (*.ini) can be added. After adding them there, you can select the application on the Unit page. The FPGA application is added to the generated processor files during the processor build process.

	Because of the long synthesis time and the huge resource consumption of the FPGA build process, it is recommended to use a separate PC. The synthesis result (*.ini file), can be linked to the application during a separate processor build on the modeling PC.

The FPGA Setup Block

Features

	All blocks belonging either to the Xilinx System Generator (XSG) or to the RTI FPGA Programming Blockset have to be placed in one Simulink subsystem if they are assigned to the same FPGA board. The root directory of the FPGA subsystem must contain an FPGA Setup block from the RTIFPGA Library

The Unit page, which is opened by double-clicking the FPGA setup block, displays the board number on which the application is stored. You must also select the framework and the piggyback.

The Parameter page shows the clock period of the FPGA. The parameters for the down-sample factor and the offline simulation period can be specified here. To start a timing analysis, an FPGA Build process or a HDL simulation, click the Execute button.

	Before starting an FPGA Build process (synthesis), it is recommended to perform a timing analysis in advance to ensure that the modeled design fits the timing constraints and the resources of the target FPGA. It is also recommended to verify changes in the offline simulation before starting the synthesis
	***If inserting interface blocks designed by dSPACE manually, ensure that all write/read registers are in an unsigned fixed format with a binary point of zero (UFix32_0).***

XSG Sent Library

Library Contents

Description / Overview	The XSG Sent Library provides general functionalities which are necessary for setting up a sent communication interface.
Quickstart	Brief description from starting the software installation process up to the download of the FPGA configuration into the real time hardware.
SENT_TX	Sent sensor simulation SAE J2716 (Revised JAN2010), respectively Sent-Slave (SPC)
SENT_RX	Sent sensor processing SAE J2716 (Revised JAN2010), respectively Sent-Master (SPC)
ADC_2_6Proc	Measurement unit for all analog inputs of the DS5203 baseboard
Proc_2_6DAC	Setting unit for all analog outputs of the DS5203 baseboard
Small Apps	The library also contains elements without an interface to the processor model. This is because **the elements' inputs do not need to be adjustable during run time or are of a small bit width**, so that it is inefficient to use an entire 32-bit register. If a connection to the processor model is required, you have to design the interface yourself. There are different kinds of frequently used applications available: ▪ Edge detection (rising, falling, both) ▪ Relay ▪ Compare to constant ▪ Switch-off delay ▪ Multifunction block (delay and shift operations) ▪ Valid edge detection
Demos	The XSG Sent Library also contains two separate demo models to illustrate an exemplary model structure of an FPGA application. Next to the Simulink models and the pre-build FPGA configuration, the demos also come with an experimental project for ControlDesk. Therefore the following demos are prepared: ▪ STD The demonstration model contains the following components: ○ 16 SENT TX channels ○ 8 SENT RX channels ○ Reading 6 ADC channels

- Driving 6 ADC channels
- Version Info

Description / Overview

Objective

The following section describes the general structure and features of the XSG Sent Library

Library Structure

General information

The XSG Sent Library is divided into:

- XSG Sent Library (FPGA-based blocks)
- XSG Sent Interface Library (processor-based blocks)

as shown in the following figure.

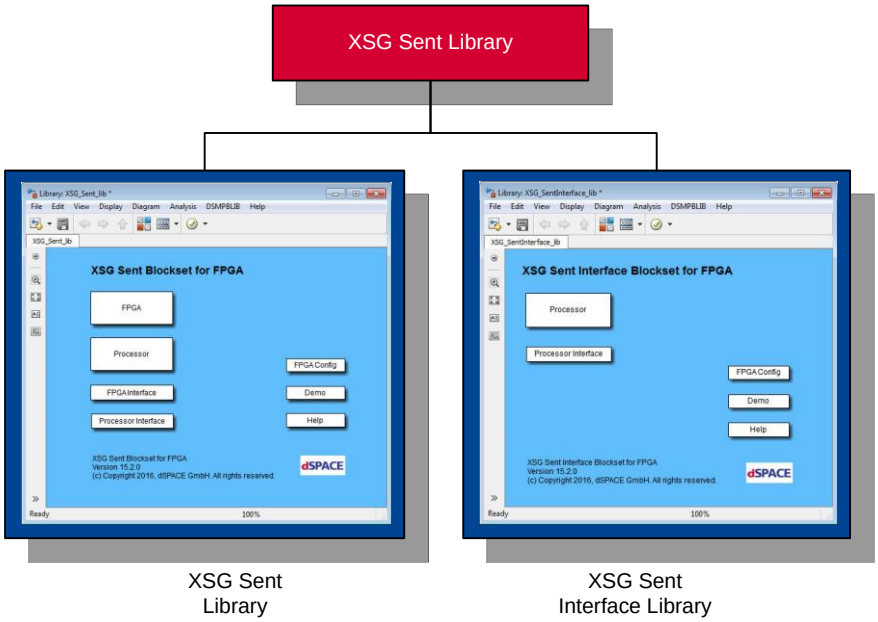


Figure 10: XSG Sent Library

XSG Sent Library

The XSG Sent Library provides easy access to FPGA based models for hardware in the loop simulation, including IO access. The library is divided into FPGA / Processor Interface and FPGA components as shown in the figure below. To open the library, type “XSG_Sent_lib” in the MATLAB Command Window.

	If the XSG Sent library will load for the first time, demos and documentation will be installed to: My Documents\ dSPACE\ XSG_SENT\20.1

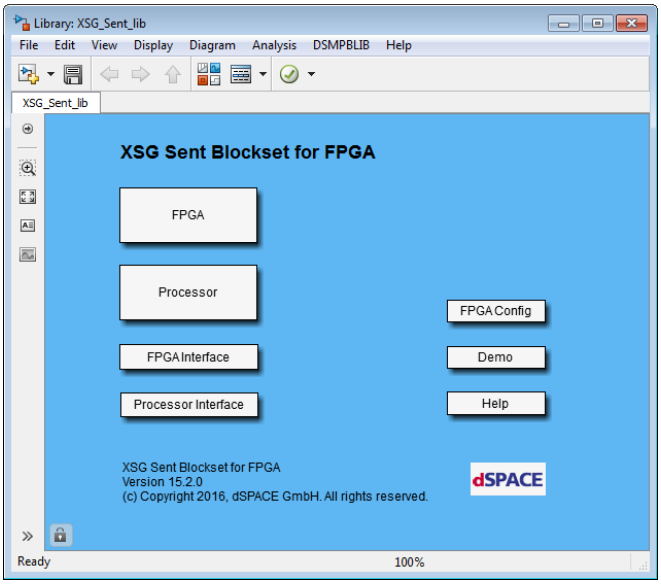


Figure 11: XSG Sent Library

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 “TX”, “RX”, etc. All **processor** blocks are located in the XSG Sent Interface Library.

	The XSG Sent Library contains the XSG Sent Interface Library.

Please refer to the Demos subsystem to get an example for the general structure of a designed FPGA (Simulink) model.

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.

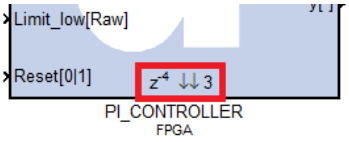


Figure 12: Block information example (PI_Controller): Latency: 4; internal model down sampling: 3

If there are no 2 down arrows in the mask of the FPGA block, the block runs with the FPGA sampling rate (e.g. 10ns for DS5203).

XSG Sent Interface Library

If no user-defined adjustments have to be made in the FPGA code, for example when the FPGA configuration (*.ini file) is pre-compiled. Then the processor interface is only required for **online** simulation. It is separately located inside the XSG Sent Interface

Library as shown in the figure below. To open the Processor Interface Library, type "XSG_SentInterface_lib" in the MATLAB Command Window or navigate via the Simulink Library Browser.

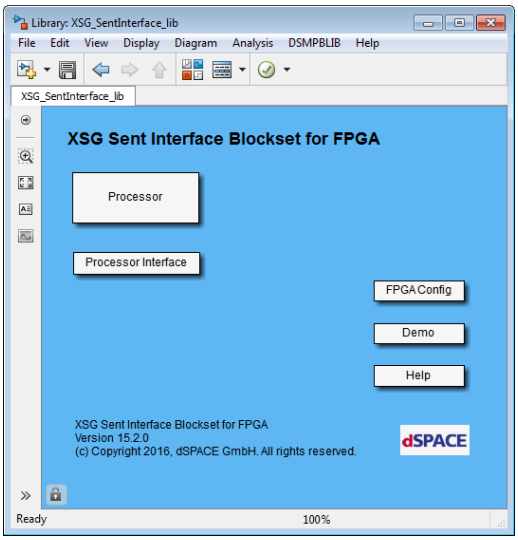


Figure 13: XSG Sent Interface Library

The internal structure of the library is the same as the structure of the XSG Sent Library, except that the interface library only contains of standard Simulink instead of XSG blocks.

Quickstart

Objective

See the following sections for a short example of building user-defined FPGA code for an SENT TX channel.

The chapter is subdivided into the following subchapters:

- Installation Guide / Procedure
- How to Generate an FPGA Model
- FPGA Timing Analysis
- FPGA Build Process
- Processor Build Process
- Offline Simulation
- Automatic Interface Generation

Installation Guide / Procedure

General information

To avoid errors in the installation of MATLAB, dSPACE and Xilinx software, you must install them in the following order:

- MATLAB
- dSPACE Release with RTI FPGA Programming Blockset
- Xilinx ISE Design Suite
- XSG Sent library

For software version combinations and their compatibility, refer to Requirements chapter.

How to Generate an FPGA Model

General procedure

To generate a user-defined FPGA model, you can use the following example which explains the procedure for a PHS-Bus based model. For models which made for the DS2655 the procedure can differ.

- Start MATLAB with Vivado XY.Z (the version XY.Z depends on your Vivado version) by using the System Generator token (Start\All Programs\Xilinx Design Tools\Vivado XY.Z\ System Generator\System Generator) and generate a new model file (e.g. Example_MDL.mdl)
- Open the RTI FPGA Library by typing “rtifpgalib” in the Command Window and drag the Processor Setup block from the Processor Interface subsystem to your model.
- Create a subsystem for the FPGA- based model part (e.g., named FPGA)

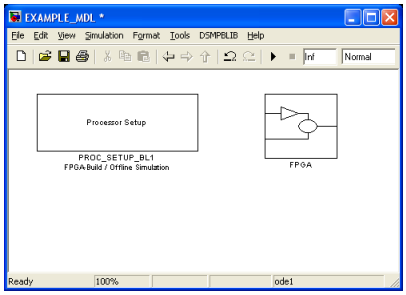


Figure 14: EXAMPLE_MDL

- Drag the FPGA Setup block from the FPGA Interface subsystem in the library to the FPGA subsystem.
- Open the Processor Setup block and on the Unit page, select the FPGA subsystem as the FPGA model for the first DS5203 board.

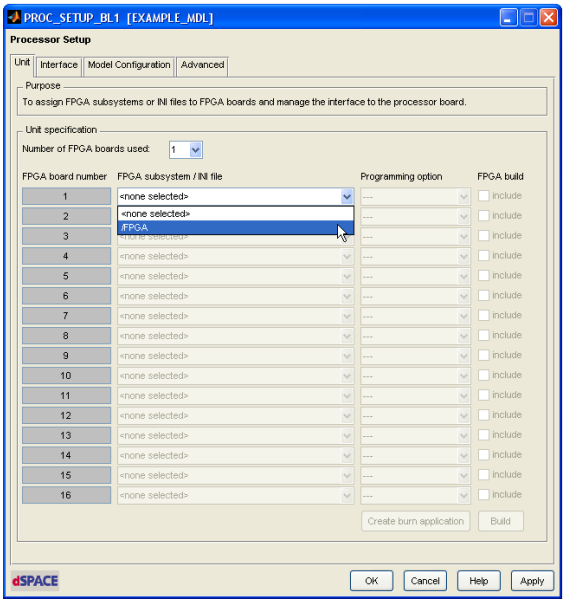


Figure 15: EXAMPLE_MDL Processor Setup block

- Open the FPGA Setup block and on the Unit page, select the FPGA chip and board for the FPGA model. In this case a DS5203 with an 7K325 FPGA is selected.

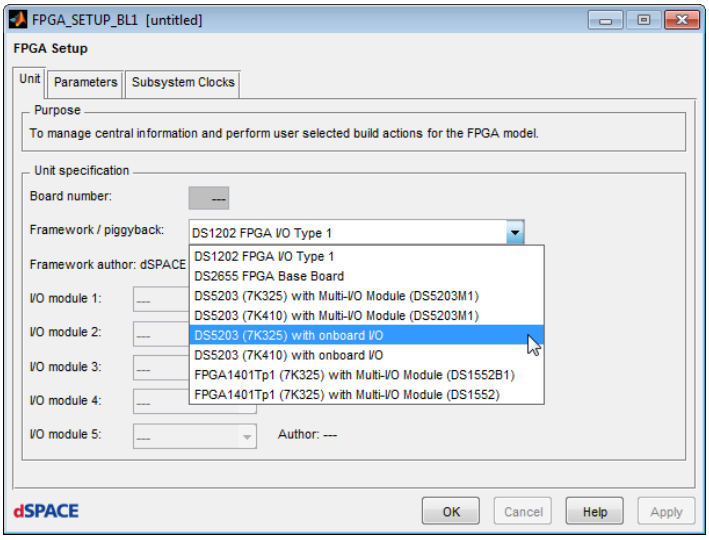


Figure 16: EXAMPLE_MDL FPGA Setup block, Page: Unit

- Configure the FPGA sample time on the Parameters page. In this case an FPGA sample rate of 10e-9 s is parameterized.

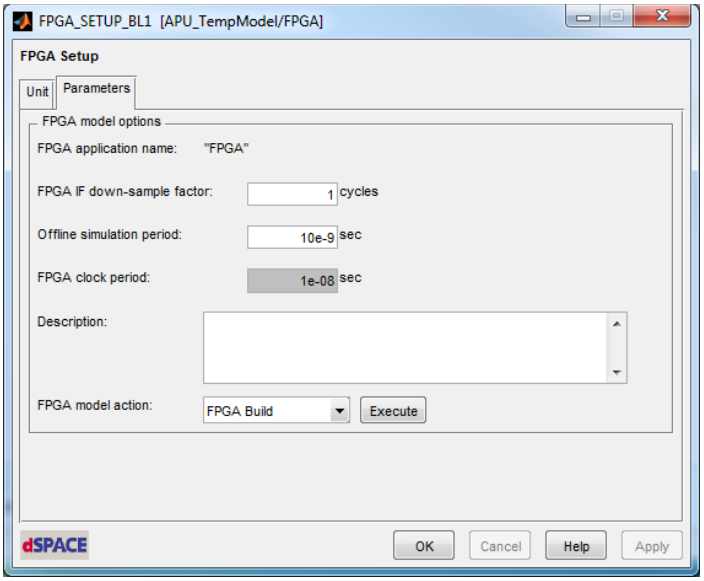


Figure 17: EXAMPLE_MDL FPGA Setup block, Page: Parameter

- After performing the changes, a system generator token is inserting in the subsystem FPGA.
- Insert the Simulink structure in the FPGA subsystem as shown in the figure below. The SENT TX and the interface blocks are located in the **XSG Sent Library**.

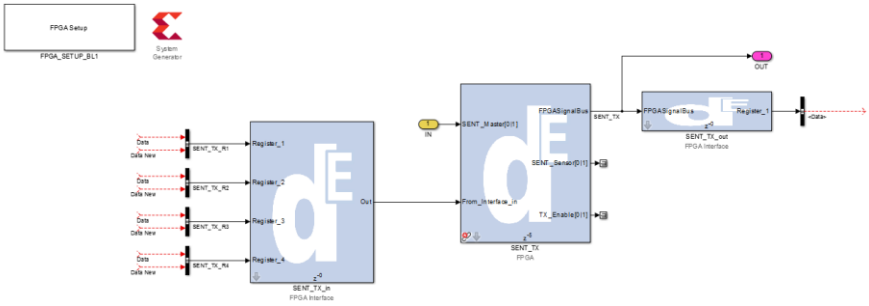


Figure 18: EXAMPLE_MDL FPGA model structure

- To generate processor-to-FPGA communication, insert an FPGA_XDATA_READ block from the RTI FPGA library in the FPGA subsystem and open the block.
- Configure the following parameters as shown below (for all 4 registers):
    - Unit Page:
        - Access type: Register
        - Channel Number: 1…4
        - Channel Name: SENT_TX_in_R1
    - Parameters:
        - Binary Point: 0
        - Format: unsigned
        - Simulation ports sample time: 10e-9
- After performing the changes, you can connect the SENT_TX_in block with the FPGA_XDATA_READ block.
- To generate FPGA-to- processor communication, insert an FPGA_XDATA_WRITE block from the RTI FPGA library in the FPGA subsystem and open the block.
- Configure the following parameters as shown below:
    - Unit Page:
        - Access type: Register
        - Channel Number: 1
        - Channel Name: SENT_TX_out_R1
    - Parameters:
        - Binary Point: 0
        - Format: unsigned
        - Simulation ports sample time: 10e-9
- After performing the changes, you can connect the SENT_TX_out block with the FPGA_XDATA_WRITE block to produce the FPGA-based model structure shown in the figure below.

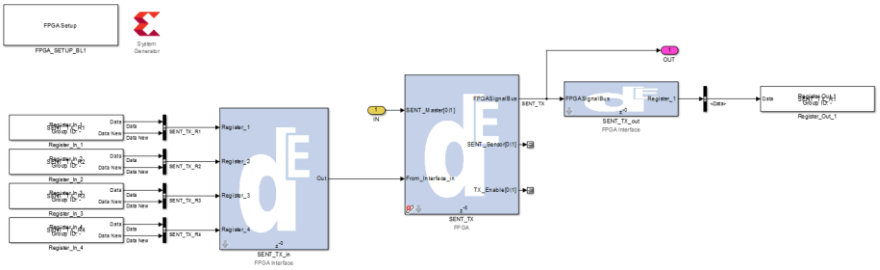


Figure 19: EXAMPLE_MDL complete FPGA model structure

- In the next step, you can insert the processor-based model part by dragging the SENT_TX_in and SENT_TX_out blocks from the **XSG Sent Interface Library**.

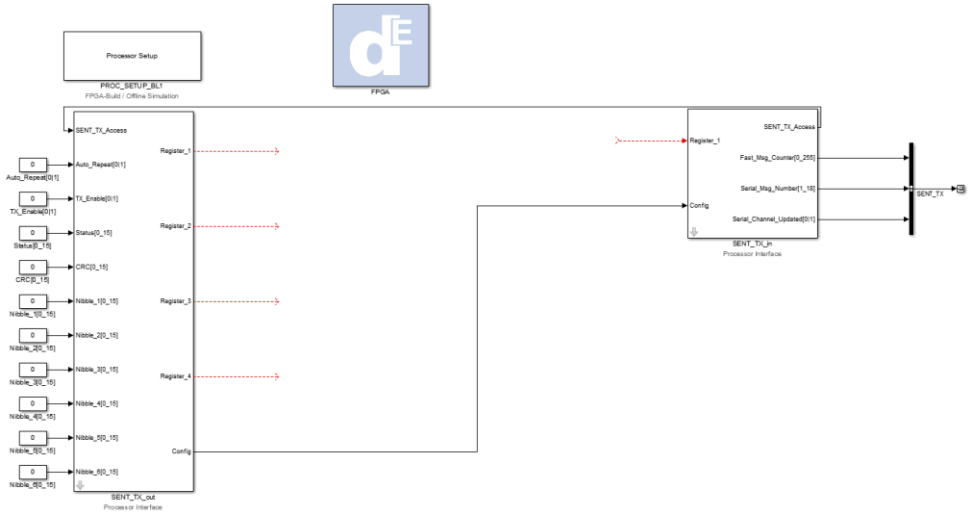


Figure 20: EXAMPLE_MDL processor model structure

- To generate processor-to-FPGA communication, insert a FPGA_XDATA_WRITE block from the RTI FPGA library and open the block.
- Configure the following parameters as shown below and connect the block to the SENT_TX_out block (for all 4 blocks):
    - Unit Page:
        - Access type: Register
        - Channel Number: 1…4
- To generate FPGA-to- processor communication, insert a FPGA_XDATA_READ block from the RTI FPGA library and open the block.
- Configure the following parameters as shown below and connect the block to the SENT_TX_in block:
    - Unit Page:
        - Access type: Register
        - Channel Number: 1
- The complete processor model structure is shown in the figure below.

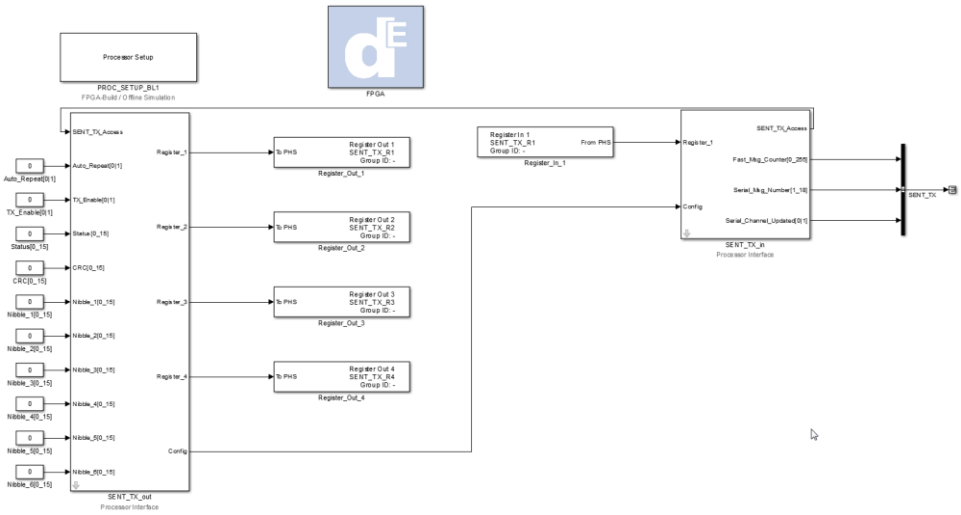


Figure 21: EXAMPLE_MDL complete processor model structure

When you have performed these steps, the processor model structure for a single sent tx channel is complete.

	The processor and the FPGA-based model structure can also be built with an **automatic** function. For detailed information, refer to the Automatic Interface Generation chapter.

FPGA Timing Analysis

General procedure

Before starting an FPGA build, it is advisable to generate a timing analysis. The following procedure can be used for this:

- Set the simulation solver options type in the Configurations Parameters to Variable-step.
- Open the FPGA Setup block which is located in the FPGA subsystem and in the FPGA model, select Timing Analysis and then click Execute.
- The timing analysis starts.

After a successful timing analysis the following window opens and the result is displayed.

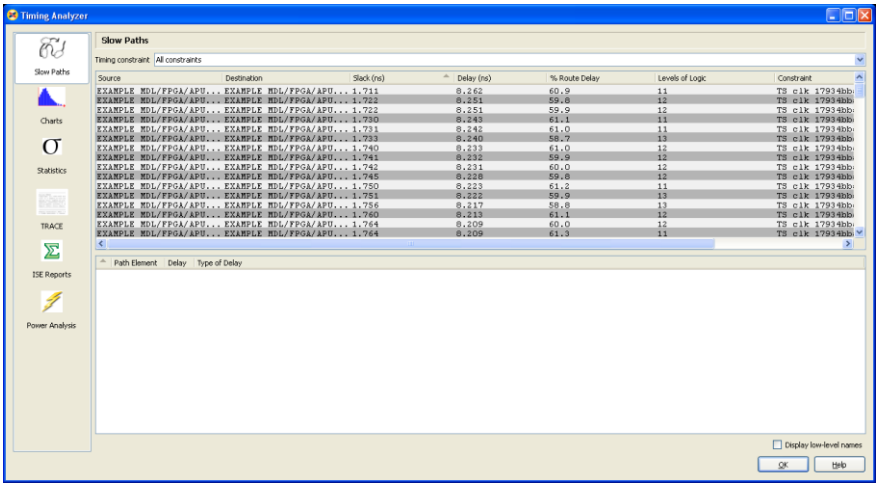


Figure 22: EXAMPLE_MDL Result of the timing analysis

Each necessary delay of each path of the FPGA-based model and the current slack is displayed. If a path delay is greater than the FPGA sample time, the current path is marked in red, the design has to be modified for example by adding an additional delay inside the longest the path. Perform another Timing Analysis to check if your design now fulfills the constrains for online simulation.

FPGA Build Process

General procedure

If an FPGA application matches the timing analysis or the timing is known, you can start a build process as follows:

- Set the simulation solver options type in the Configurations Parameters to Variable-step.
- Open the FPGA Setup block which is located in the FPGA subsystem and in the FPGA model, select Timing Analysis and then click Execute.
- The FPGA build starts.

After a successful build, the following text is displayed in the Command Window:

```
Starting synthesis of System Generator output files...
Packing netlists into one netlist file (NGC)...

Starting synthesis...
Starting translation (NGD)...
Starting mapping...
Starting place-and-route...
Starting generation of bitstream file...

WORK   DIRECTORY: X:\Appl_Intern\E_Drive_Team\Solutions\DS5203\infos\Doku\EXAMPLE_MDL
BUILD  DIRECTORY: X:\Appl_Intern\E_Drive_Team\Solutions\DS5203\infos\Doku\EXAMPLE_MDL\EXAMPLE_MDL_rtiFPGA\FPGA_672F57301D178D
RESULT FILE:      X:\Appl_Intern\E_Drive_Team\Solutions\DS5203\infos\Doku\EXAMPLE_MDL\EXAMPLE_MDL_rtiFPGA\ini\FPGA_672F57301D178D.ini

FPGA Build Done
Elapsed time is 716.815880 seconds.
>>
```

Figure 23: EXAMPLE_MDL Result of the build process

The result of the build process in this example is the binary file FPGA_672F57301D178D.ini (renaming this file is allowed).

Because an FPGA build process needs a lot of time, it is assumed that an offline check was performed successfully **before** an FPGA build process is started. An additional PC is recommended for the FPGA build process.

Processor Build Process

General procedure

If the build process is successful or an INI file is available, you can start the processor build process as follows:

- Open the Processor Setup block, which is located on the first layer of the model, and select the INI file from the build process by clicking Add on the Advanced page.
- Select the added INI file on the Unit page and set the programming option, e.g. to “into ram”, as shown in the figure below.

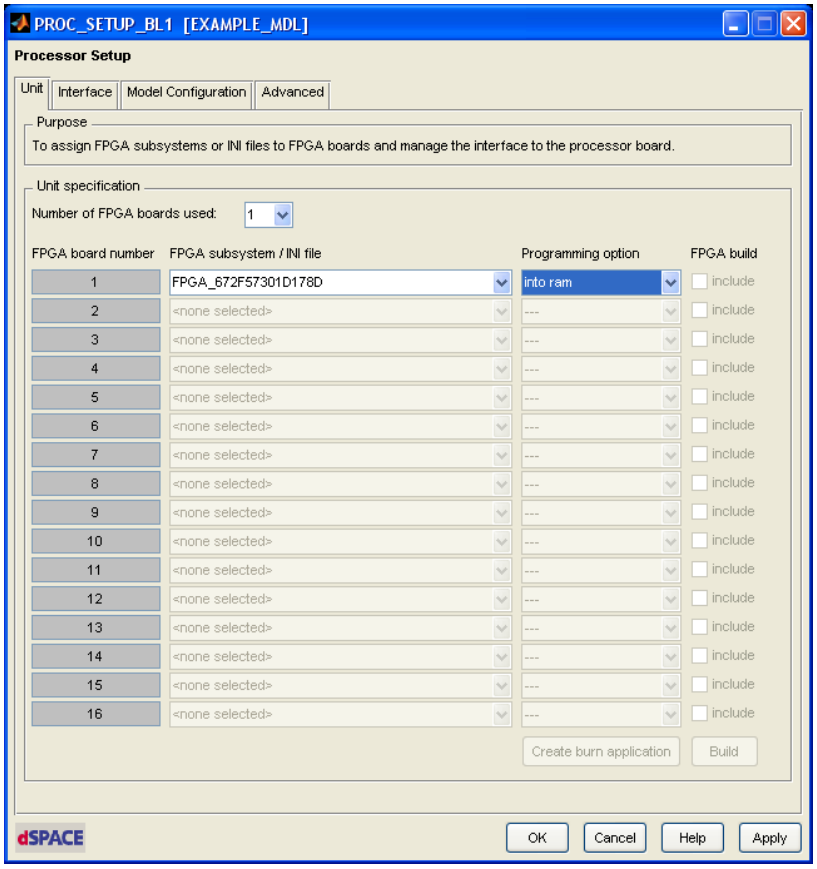


Figure 24: EXAMPLE_MDL Processor setup block

- Switch the model mode on the Model Configuration page from FPGA-Build / Offline Simulation to Processor-Build and then click the Switch model mode button.
- The model is copied to the EXAMPLE_MDL_rtiFPGASeparationFile.mdl model and the FPGA subsystem is cut off.

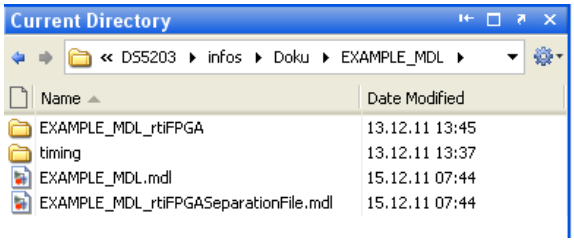


Figure 25: EXAMPLE_MDL Current directory of MATLAB after switching the model mode

- Set the simulation solver options type in the Configurations Parameters to -Fixed-step and configure the sample time of the processor, e.g., 0.001s.
- Start the processor build process by pressing Ctrl+ B.

Offline Simulation

General procedure

To generate a user offline simulation of the FPGA and the processor model, the following procedure can be used:

- Set the simulation solver options type in the Configurations Parameters to Variable-step.
- Open the Processor Setup block which is located on the first layer of the model and switch the model mode to FPGA-Build / Offline Simulation.
- Select the FPGA subsystem on the Unit page (in this case "/FPGA")
- Configure the stop time and start the offline simulation by clicking the Run button.

Mapping of FPGA signals from / to Simulink blocks

If standard Simulink blocks are connected to Xilinx blocks, you have to insert in between the following blocks from Xilinx Blockset / Basic Elements:

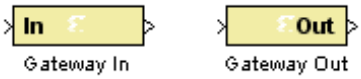


Figure 26: Gateway In and Gateway Out blocks

⚠	The Gateway In block, has to be configured for the data format of the fixed point output and the offline sample period, like regular XSG blocks.
⚠	Before starting a Timing Analysis or FPGA Build all Gateway In/Out blocks have to be deleted from your model!

The EXAMPLE_MDL model is used gives you a better overview.

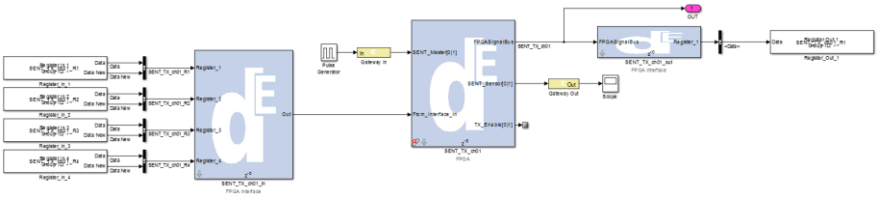


Figure 27: Gateway In and Gateway Out blocks

In the example, the SENT_Master[0|1] signal gets the source of a Simulink Pulse Generator block, and SENT_Sensor[0|1] is connected to a Scope block.

Automatic Interface Generation

Objective

An automatic interface generation tool is available for quick, easy use of the Sent Library. The supported FPGA main components have a dialog as explained in the next section.

Example SENT_TX

The following example gives a better overview of the functionality of automatic interface generation. Open a new model and insert a component, e.g., the FPGA main component of the sent message transmission (SENT_TX), from the library as shown in the following figure.

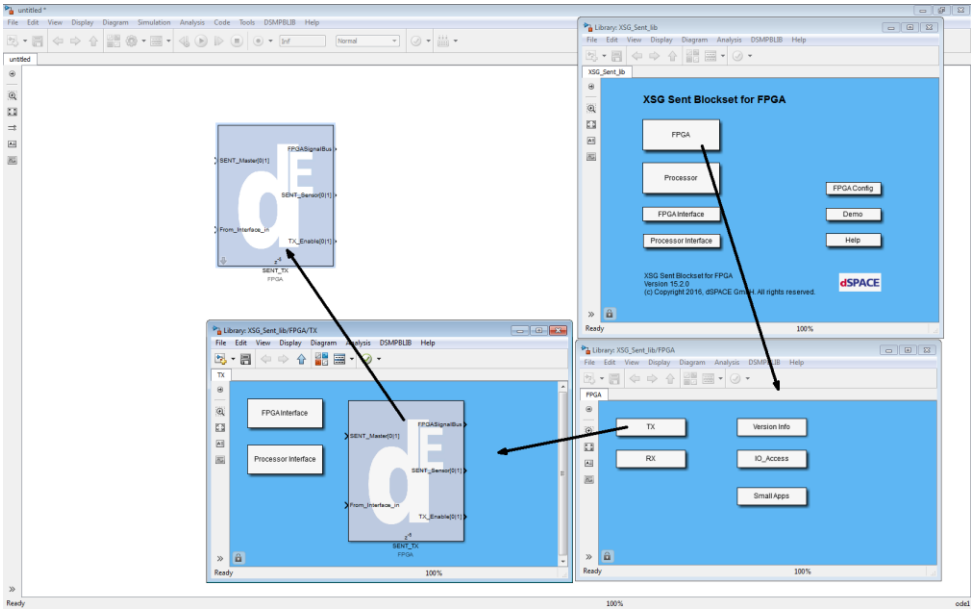


Figure 28: Tool example of automatic interface generation

After inserting the FPGA main component block of the SENT_TX, you can open the dialog by double-clicking the block. The TX's dialog parameters are shown in the figure below.

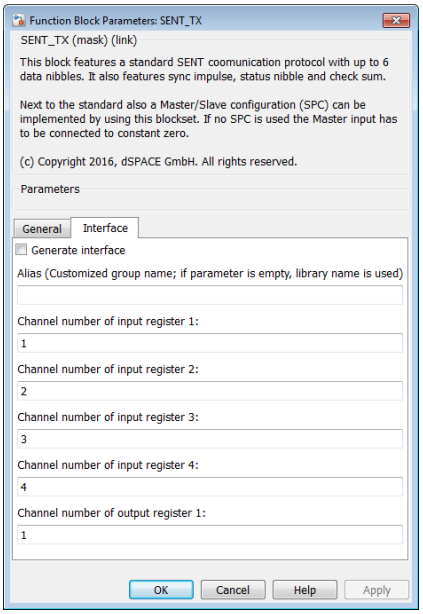


Figure 29: FPGA main component SENT_TX: Parameter dialog

Each dialog that supports automatic interface generation lets you parameterize the user-defined input and output channel numbers (Register channel numbers for the corresponding interface elements). Additional to that a customized group name (parameter Alias) can be defined. If this parameter is empty, the original library name is used. To start automatic interface generation, enable the “Settings for interface generation” and select the “Generate interface” checkbox. At first the actual framework of the model will be analyzed. Because no FPGA framework is defined in the actual new model the following popup will opened.

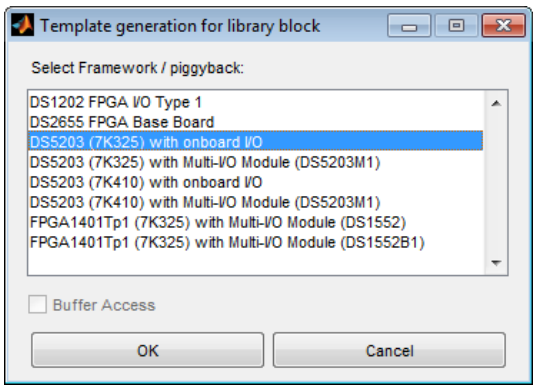


Figure 30: Selection Popup of the framework of the build process

The destination framework can be selected and the automatic interface generation continuous. If a FPGA framework is already defined in the model during the start of the automatic interface generation, the algorithm analyzed the overall model and opens a popup in which the destination submodel can be selected.

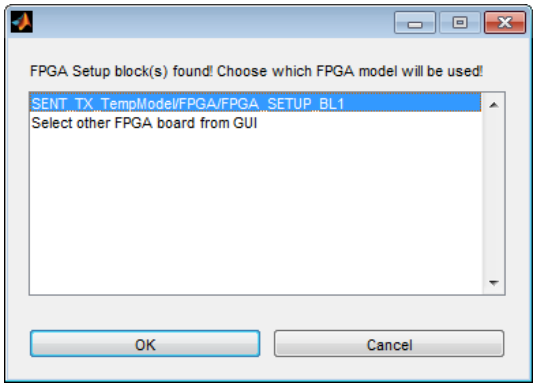


Figure 31: Selection Popup of the destination submodel of the build process

If a new framework / submodel needs to be used the Tab “Select other FPGA board from GUI” can be selected. In this case the destination framework popup from Figure 30 opens and the destination framework can be adjusted.

When an interface channel which is defined in the GUI (here, input ch.1 or output ch.1) is already used inside your source model the following message will occur:

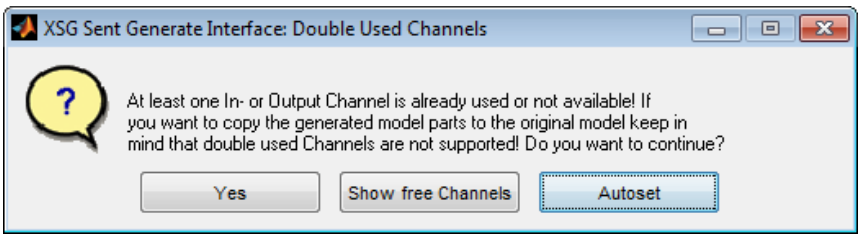


Figure 32: Warning automatic interface generation

- Yes: The template model will be generated with the specified channel numbers defined in the GUI
- Show free Register: A dialog appears which shows the free input and output channels, so that the GUI entries can be correctly adapted
- Autoset: Replace of the double used register(s) by the first free ones

When the “Show free Register” or “Autoset” button is selected the actual model is also automatically checked for double used interface channels. If a channel is used more than once the following dialog appears:

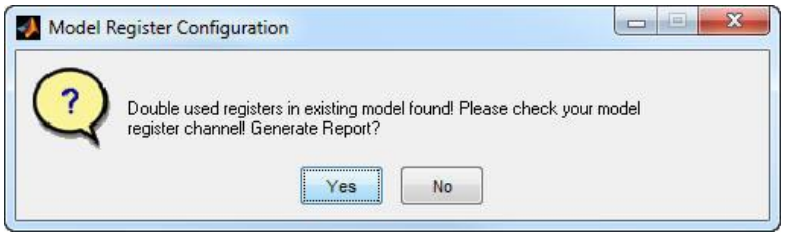


Figure 33: Double used channels

When pressing the “Yes” button an html report will be generated from which related are accessible via hyperlink.

When all settings are correct the template model generation starts immediately. The result of the generation process is a template model of the SENT_TX with a standard interface configuration and a standard Simulink structure for FPGA models. This model differs, which depends on the settings on the selection popup as shown in figures below. The following subchapters described in detail the specific output of the “Automatic Interface Generation” function.

If during the automatic interface generation the question of Buffer Acces raises, the option “**No**” has to be choosen, since the XSG_Sent is designed for register access only!

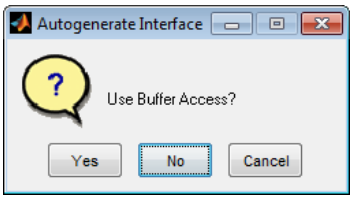


Figure 34: NO Buffer Access

	Buffer access is not supported for XSG_Sent

Template model

After the automatic interface generation has fished. A template model has been generated. Two windows for this model will appear on the screen, the left one shows the processor and the right one the FPGA model view, like shown in the figure bellow.

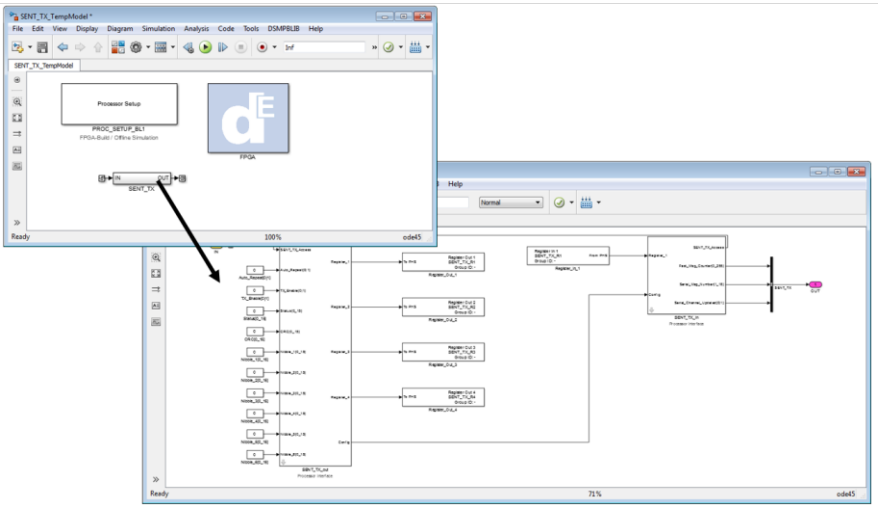


Figure 35: SENT_TX template model (processor view)

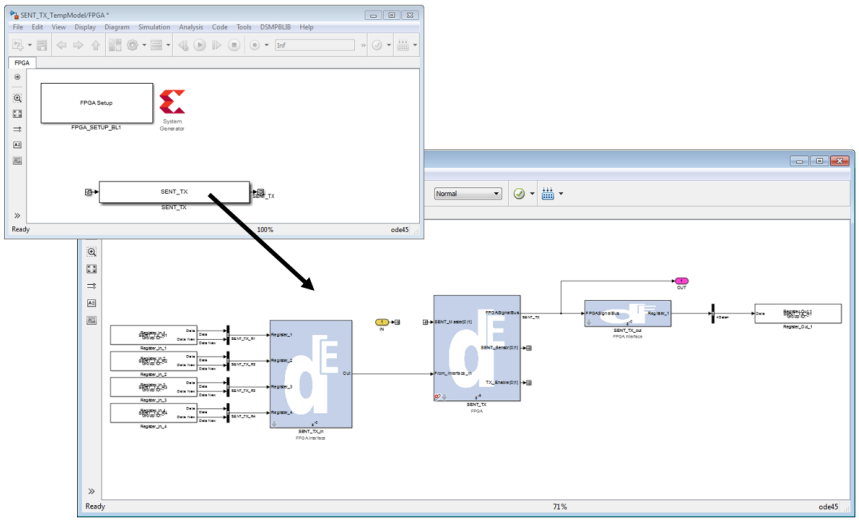


Figure 36: SENT_TX template model (FPGA view)

SENT_TX

Objective

The sent transmission implementation is based on SAE J2716 (Revised JAN2010). It supports serial message, short and enhanced format, as well as pause pulse functionality with constant pause length or constant message length option and a message counter for fast messages can be activated.

Next to the standard frame, this block can also be configured for bidirectional SPC (Master/Slave) configuration.

The blockset contains the following elements:

- Processor Interface: SENT_TX_out (Processor Interface)
- FPGA Interface: SENT_TX _in (FPGA Interface)
- FPGA: SENT_TX (FPGA Main Component)
- FPGA Interface: SENT_TX _out (FPGA Interface)
- Processor Interface: SENT_TX _in (Processor Interface)

General behavior

According to the GUI settings (SENT_TX_out), a SENT transmission frame configuration is transported to the FPGA where it is calculated. Related to GUI settings (SENT_TX_out) the block itself modifies its amount of inputs, e.g. number of data nibles, serial channel config, etc.

Processor Output

Block

Merges the processor signals and writes them to the FPGA. Here with the GUI settings for standard SENT without serial message transmission.

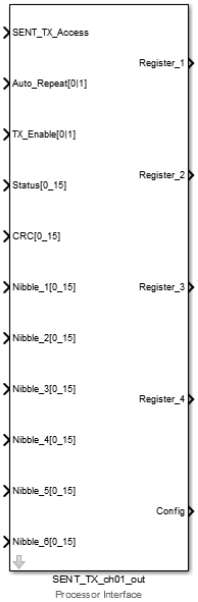


Figure 37: SENT_TX_out block

Block Dialog

The dialog for a standard SENT frame configuration without serial message and pause pulse is displayed here:

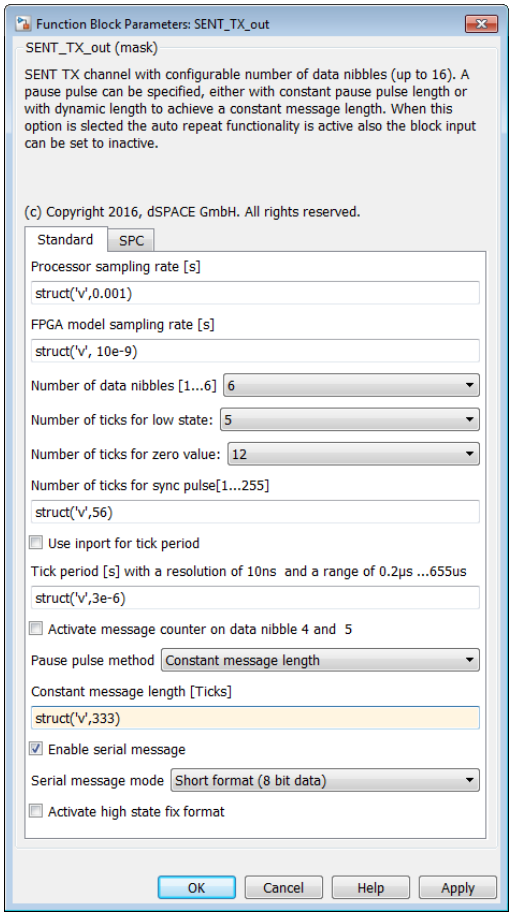


Figure 38: SENT_TX_out GUI Standard tab

The GUI comes with following parameters:

Name	Unit	Description	Format
Processor sampling rate	[s]	Sampling rate of this Simulink block on the processor side	-
FPGA model sampling rate	[s]	Sampling rate of the corresponding main component block on the FPGA side	-
Number of data nibble	[-]	Amount of data nibble for a single sent frame	1 … 6
Number of ticks for low state	[-]	Amount of ticks which stay constant low for a single nibble. Must be smaller than the amount of zero ticks	1 … 14
Number of ticks for zero value	[-]	Amount of ticks for a nibble which represents the value 0	1 …15
Number of ticks for sync pulse	[-]	Amount of ticks for the synchronization pulse at the beginning of a SENT frame	1 … 255
Use inport for tick period	[0|1]	If checkbox is activated an inport will added, so that the tick period can be modified dynamically during runtime	Bool
Tick period	[s]	If checkbox is deactivated the tick period has to be entered over here	(0 … 65535)*10e-9 = 0 … 655.35µs
Activate message counter	[0|1]	When activating the message counter on data nibble 4 & 5 an additional input port for resetting the counter on the FPGA will be available. Since the message counter itself is located directly on the FPGA, **the entered data values for nibble 4 & 5 will be ignored in this configuration**. **In addition, the CRC is directly calculated on the FPGA, so that the processor input value is not taken into account within this configuration.** The most significant data will be transmitted on data nibble 4 (MSN) and the least significant data on data nibble 5 (LSN).	Bool

Data Nibble 5				Data Nibble 4			
8	4	2	1	8	4	2	1
8	4	2	1	128	64	32	16
			LSB	**MSB**			

Name	Unit	Description	Range
Pause pulse method	[-]	**No pause pulse**: End the end of a SENT TX frame no additional pause will be added, only the amount of low ticks, before the next frame starts **Constant pause pulse** At the end of the TX frame a constant pause (low pulse) is added **Constant message length** A pause pulse is dynamically calculated to achieve a constant frame length	(0 … 32767)*10e-9 = 0 … 327.67µs
Enable serial message	[0|1]	If checkbox is activated 3 types of serial messages can be selected 1. Short 2. Enhanced 12 Bit data 3. Enhanced 16 Bit data	0|1

Serial message mode	Description
Short format	4 bit ID 8 bit data 4 bit CRC
Enhanced format Config bit = 0	8 bit ID 12 bit data 6 bit CRC
Enhanced format Config bit = 1	4 bit ID 16 bit data 6 bit CRC

When the serial message mode has been enabled, the following three additional inports appear to block:

- Slow_ID
- Slow_Data
- Slow_CRC

The units of the block are adapted automatically according to the specified serial mode format

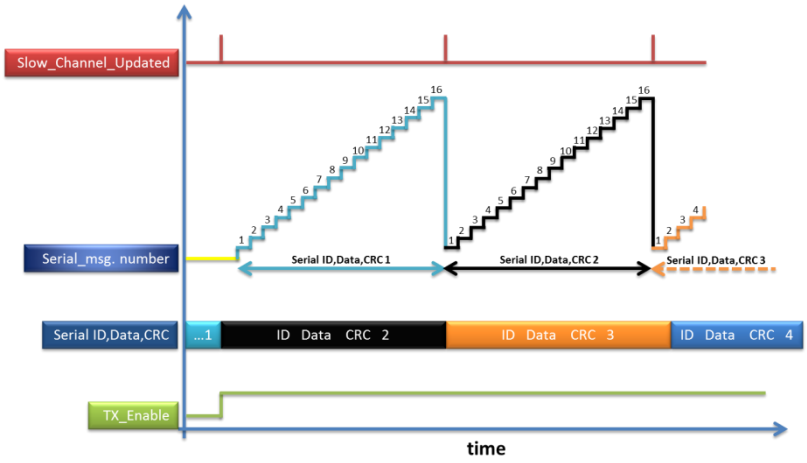


Figure 39: Serial message data transfer

When the ID, the data and the CRC of the slow (serial) message is latched on the FPGA the Slow_Channel_Updated[0|1] flag is set to 1 for one processor sample step. Then the new data for the next following slow message can be applied to the block.

Name	Unit	Description	Range
Activate high state fix	[0|1]	If checkbox is activated the number of low ticks are converted to number of high ticks which stay constant. The data nibble value only effects the length of the low time	Bool

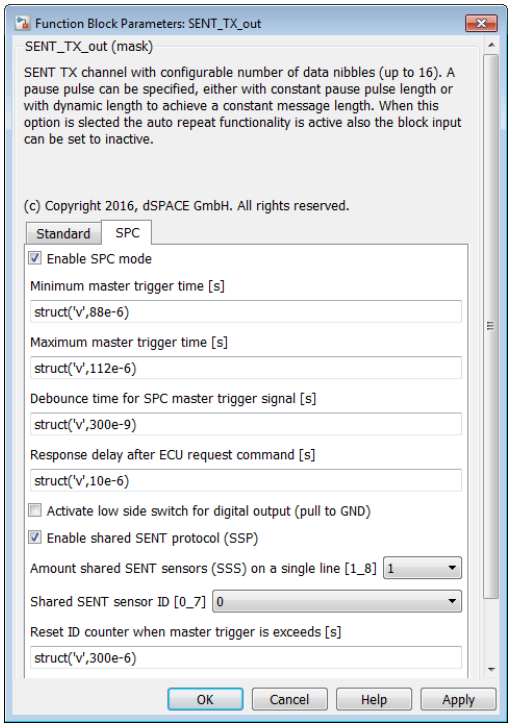


Figure 40: SENT_TX_out GUI SPC tab

Name	Unit	Description	Range
Enable SPC mode	[-]	If checkbox is activated bidirectional SPC (Master/Slave) mode is activated. Additional GUI parameters are available	0|1
Minimum master trigger time	[s]	Minimum time of master trigger pulse. The measured master trigger pulse has to be inside the range between minimum and maximum trigger time for being valid (generating a sent frame answer)	(0 … 65535)*10e-9 = 0 … 655.35µs
Maximum master trigger time	[s]	Maximum time of master trigger pulse. The measured master trigger pulse has to be inside the range between minimum and maximum trigger time for being valid (generating a sent frame answer)	(0 … 65535)*10e-9 = 0 … 655.35µs
Debounce time	[s]	The debounce time prevents that noise signals are recognized as valid falling edges. High or low times below this value are ignored.	(0…1023)*10e-9 = 0 … 10.23µs
Response delay time	[s]	After a valid master trigger, the SENT transmission starts after the specified response delay time.	(0…4095)*40e-9 = 0 … 163.83µs

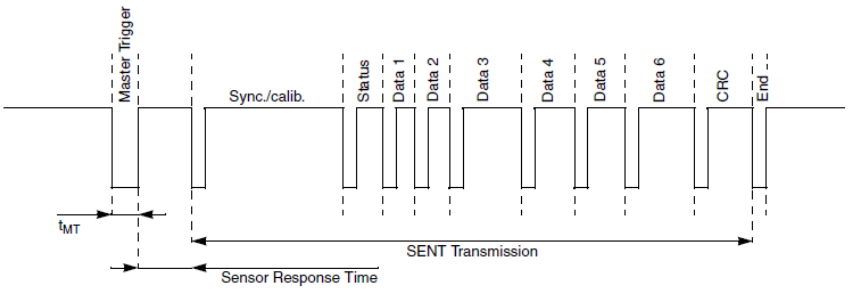


Figure 41: Standard SENT frame configuration plus SPC (Master/Slave) extension

Name	Unit	Description	Range
Activate low side switch	[0|1]	In regular applications, the ECU (SENT Master or SENT RX channels) provides the high potential of the SENT line and the sensor (SENT TX channel) is pulling to GND. This functionality can be enabled by activating the check for the low side switch (0:open state | 1:pull to GND), otherwise push pull mode is activated (0:pull to GND | 1:push to 5V)	Bool
Enable SSP	[0|1]	0: shared SENT protocol (SSP) deactivated 1: SSP activated SSP means that on a single physical line several sensor are connected. Each sensor has its own unique ID. A counter increases the actual ID (the sensor which is allowed to send its data) after every master impulse and wraps around after the highest ID sent the data. The ID counter resets when the master impulse over exceeds the spefied time	Bool
Amount SSS	[1_8]	Amount of shared SENT sensors (SSS) which are connected on a single physical line.	1 … 8
SSS ID	[0_7]	ID of the shared SENT sensor	0 … 7
Reset ID Counter	[s]	Shared SENT ID counter reset when the master impulse exceeds this value	(0 … 65535)*10e-9 = 0 … 655.35µs

Input

The SENT_TX_out block has the following inputs, which are permanent, available:

Name	Unit	Description	Range
SENT_TX_Access	[-]	Bus which provides information coming from the SENT_TX_in, respectively FPGA	
Auto_Repeat	[0|1]	If checkbox is activated, at the end of a sent frame the transmission continuous with a new sent frame when the **TX_Enable[0|1]** port is **activated**.	bool
TX_Enable	[0|1]	If checkbox is activated one SENT frame will be transmitted at every rising edge of the signal. Continuous transmission in conjunction with activated Auto_Repeat[0|1]	0|1
Status	[0_15]	Value of status nibble for non-serial message transmission	0…15
CRC	[0_15]	Value of checksum nibble at the end of the frame	0…15
Nibble_1…6	[0_15]	Data nibbles 1 to 6, amount of nibbles depends on the GUI settings	[0_15]

The SENT_TX_out block has the following inputs, which appear depending on the GUI settings:

Name	Unit	Description	Range
Const_Tick_Period	[s]	Tick period (smallest time interval) of the SENT frame	(0 … 65535)*10e-9 = 0 … 655.35µs
Reset_Msg_Counter	[0|1]	If message counter is activated, the counter value can be reset	bool
Serial_ID	[-]	Identifier of the serial message	0 …255
Serial_Data	[0_15]	Data value of serial message	0 … 65535
Serial_CRC	[0_15]	Checksum of serial message	0…64

FPGA Main Component

Block

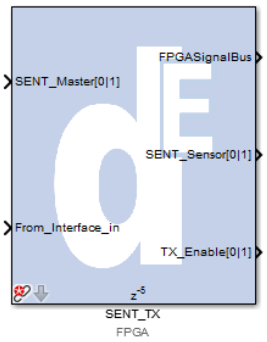


Figure 42: SENT_TX FPGA Main block

Block Dialog

The FPGA main blockset contains the following dialog.

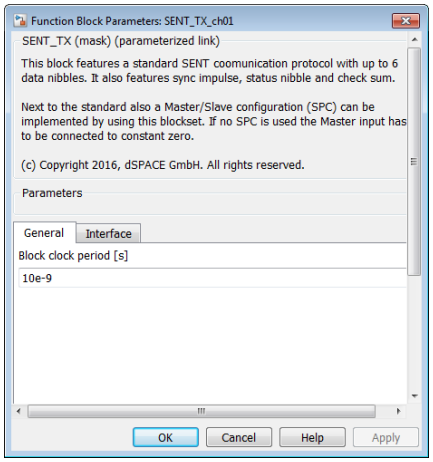


Figure 43: SENT_TX FPGA Main block dialog

The dialog blockset has the following parameters:

Name	Unit	Description	Format
Block clock period	[s]	Sampling rate of the FPGA model part, usually the FPGA clock rate. PHS 10e-9, SCLX 8e-9	-

	The parameters of the input and output registers can be defined for automatic interface generation (When checkbox is enabled) on the page Interface. The dialog also contains an additional button to start interface generation. For more details about automatic interface generation, refer to Automatic Interface Generation chapter.

Input

The main block has the following inputs:

Name	Unit	Description	Format
SENT_Master	[0|1]	Interface connection to the SENT master. Has to be connected to a digital input. Only required for SPC mode, otherwise connect a constant with the value 0 to this port.	UFix_1_0
From_Interface_in	[-]	Bus which contains all relevant information which configure the SENT frame	Bus

Output

The SENT_TX main block has the following outputs:

Name	Unit	Description	Format
FPGASingalBus	Bus	Bus containing the most significant signals of this block	-
SENT_Sensor	[0|1]	Actual level of the SENT signal. Has to be connected to a digital output (Data port).	UFix_1_0
TX_Enable	[0|1]	Enables the digital output channel in standard SENT mode. In SPC mode it enables to switch between open state und pull to GND. Has to be connected to a digital output (Enable port)	Bool

Processor Input

Block

Feedback information from the FPGA are available at the blocks outputs. The FPGA provides information via register in a raw format (UFix32_0), these raw values are reinterpreted inside this block.

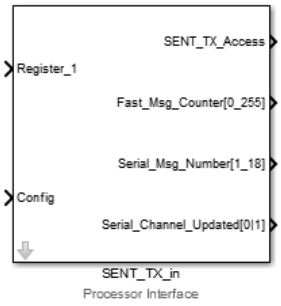


Figure 44: SENT_TX_in block

Block Dialog

The dialog provides only a short block description.

Input

The processor input block set has the following inputs:

Register 1: Provides the raw data coming from the FPGA

Config: Settings from the SENT_TX_out block

Output

The processor input block has the following outputs:

Name	Unit	Description	Range
SENT_TX_Access	[-]	Bus which provides information coming from the SENT_TX_in, respectively FPGA	
Fast_Msg_Counter	[-]	Counter for fast messages. Is incremented after every end of SENT frame	0 …255
Serial_Msg_Number	[-]	Actual serial message number, either up to 16 or 18 depending on the serial message configuration	0 … 18
Serial_Channel_Updated	[0|1]	When the FPGA is ready to receive new serial channel data, the output is set to high state for a single sample step	Bool

Interface Examples

Processor blocks

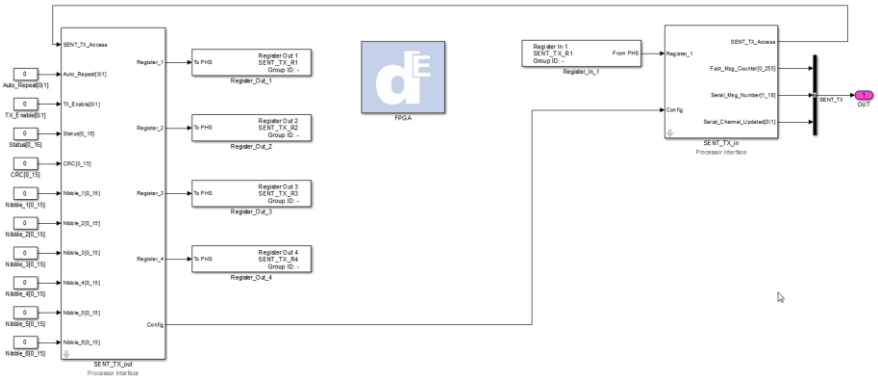


Figure 45: Processor interface

FPGA block

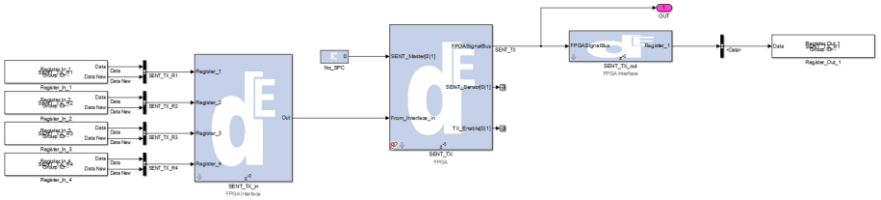


Figure 46: FPGA interface

SENT_RX

Objective

The sent transmission implementation is based on SAE J2716 (Revised JAN2010). It supports serial message, short and enhanced format, as well as pause pulse functionalty..

Next to the standard frame, this block can also be configured for bidirectional SPC (Master/Slave) configuration. Where this block can simulate the master functionality, inclusive master pulse generation.

The blockset contains the following elements:

- Processor Interface: SENT_RX_out (Processor Interface)
- FPGA Interface: SENT_RX _in (FPGA Interface)
- FPGA: SENT_RX (FPGA Main Component)
- FPGA Interface: SENT_RX _out (FPGA Interface)
- Processor Interface: SENT_RX _in (Processor Interface)
-

General behavior

According to the GUI settings (SENT_RX_out), a SENT receiving frame configuration is transported to the FPGA where it is calculated. Related to GUI settings (SENT_TX_in) the block itself modifies its amount of inputs, e.g. number of data nibles, etc.

The time resolution of the measured signal is automatically decreased to 16*FPGA sampling rate (e.g. 160ns @ Ts = 10ns) when the specified tick period is bigger than 5µs

Processor Output

Block

Merges the processor signals and writes them to the FPGA. Here with the GUI settings for standard SENT without serial message transmission.

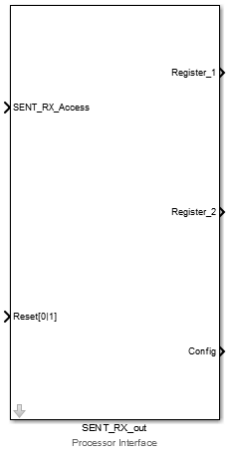


Figure 47: SENT_RX_out block

Block Dialog

The dialog for a standard SENT frame configuration without serial message and activated SPC mode is displayed below:

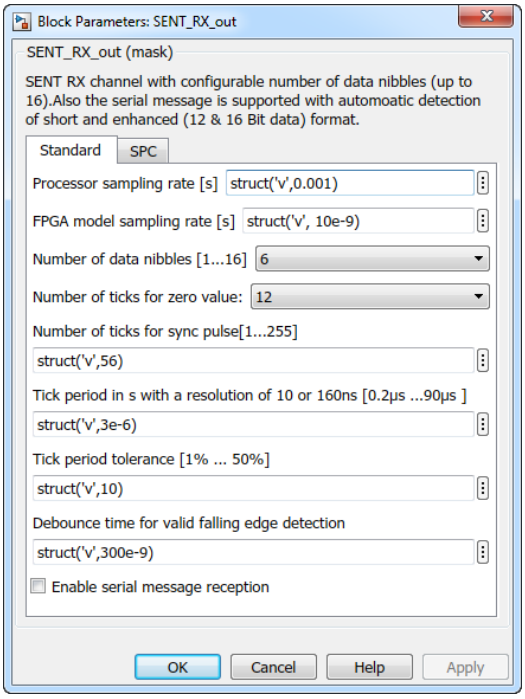


Figure 48: SENT_RX_out GUI Standard tab

The GUI comes with following parameters:

Name	Unit	Description	Format
Processor sampling rate	[s]	Sampling rate of this Simulink block on the processor side	-
FPGA model sampling rate	[s]	Sampling rate of the corresponding main component block on the FPGA side	-
Number of data nibble	[-]	Amount of data nibble for a single sent frame	1 … 6
Number of ticks for zero value	[-]	Amount of ticks for a nibble which represents the value 0	1 …15
Number of ticks for sync pulse	[-]	Amount of ticks for the synchronization pulse at the beginning of a SENT frame	1 … 255
Tick period	[s]	Expected tick period which is the smallest unit inside a SENT frame	0 … 90µs
Tick Period Tolerance	[%]	Tolerance range, in which the measured tick period can differ from the specified one	0 … 50
Debounce time	[s]	The debounce time prevents that noise signals are recognized as valid falling edges. High or low times below this value are ignored.	(0…1023)*10e-9 = 0 … 10.23µs
Enable serial message	[0|1]	If checkbox is activated, the received serial message mode is detected automically	Bool

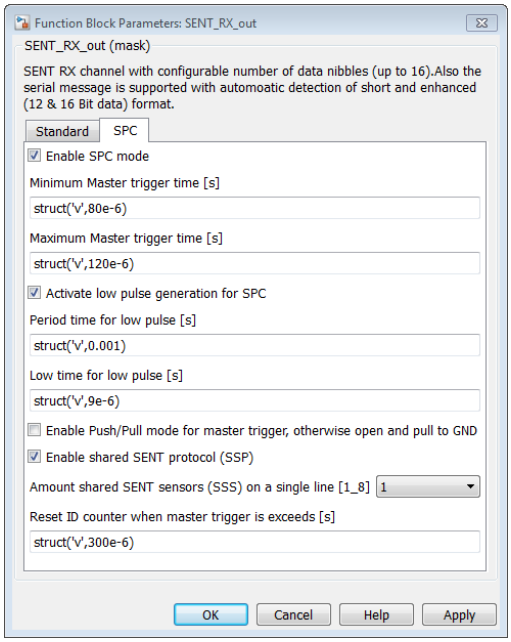


Figure 49: SENT_RX_out GUI SPC tab

Name	Unit	Description	Range
Enable SPC mode	[0|1]	If checkbox is activated bidirectional SPC (Master/Slave) mode is active.	Bool
Minimum master trigger time	[s]	Minimum time of master trigger pulse. The measured master trigger pulse has to be inside the range between minimum and maximum trigger time	(0 … 65535)*10e-9 = 0 … 655.35µs
Maximum master trigger time	[s]	Maximum time of master trigger pulse. The measured master trigger pulse has to be inside the range between minimum and maximum trigger time	(0 … 65535)*10e-9 = 0 … 655.35µs
Activate low pulse	[0|1]	The master impulse (low pulse) will be generated by this block	Bool
Period time	[s]	Period time of the master (low) pulse, e.g. 1ms. Time between 2 consecutive master trigger pulses.	(0… 1048576)*10e-9 = 0…10.5ms
Low time	[s]	Length of the low time for the master (low) pulse, e.g. 80µs	(0… 1048576)*10e-9 = 0…10.5ms
Enable Push/Pull	[0|1]	If checkbox is activated, the digital output for the master trigger is configured in Push (5V)/Pull (GND) mode. Otherwise it is configured in Open/Pull (GND) mode. In this case the sensor supply has to be connected via pull up resistance to the digital output	Bool

Enable SSP	[0|1]	0: shared SENT protocol (SSP) deactivated 1: SSP activated SSP means that on a single physical line several sensor are connected. Each sensor has its own unique ID. A counter increases the actual ID (the sensor which is allowed to send its data) after every master impulse and wraps around after the highest ID sent the data. The ID counter resets when the master impulse over exceeds the spefied time	Bool
Amount SSS	[1_8]	Amount of shared SENT sensors (SSS) which are connected on a single physical line.	1 … 8
Reset ID Counter	[s]	Shared SENT ID counter reset when the master impulse exceeds this value	(0 … 65535)*10e-9 = 0 … 655.35µs

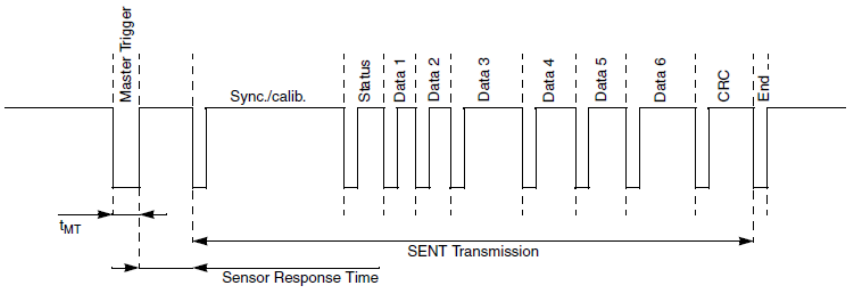


Figure 50: Standard SENT frame configuration plus SPC (Master/Slave) extension

Input

The SENT_RX_out block has the following inputs, which are permanent, available:

Name	Unit	Description	Range
SENT_RX_Access	[-]	Bus which provides information coming from the SENT_TX_in, respectively FPGA	
Reset	[0|1]	Resets the received values on the FPGA	bool

FPGA Main Component

Block

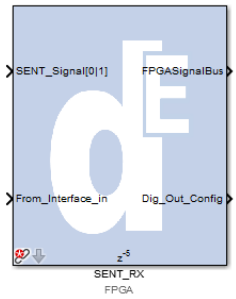


Figure 51: SENT_RX FPGA Main block

Block Dialog

The FPGA main blockset contains the following dialog.

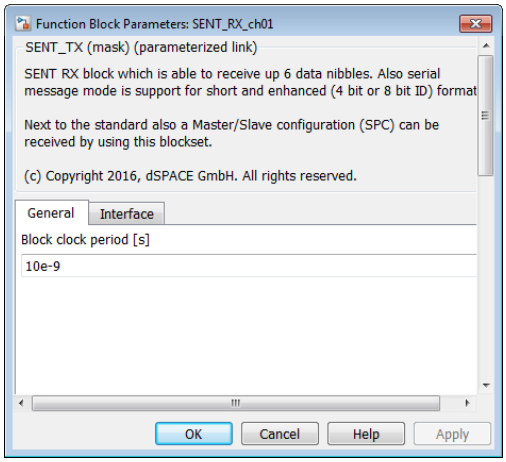


Figure 52: SENT_RX FPGA Main block dialog

The dialog blockset has the following parameters:

Name	Unit	Description	Format
Block clock period	[s]	Sampling rate of the FPGA model part, usually the FPGA clock rate. PHS 10e-9, SCLX 8e-9	-

	The parameters of the input and output registers can be defined for automatic interface generation (When checkbox is enabled) on the page Interface. The dialog also contains an additional button to start interface generation. For more details about automatic interface generation, refer to Automatic Interface Generation chapter.

Input

The main block has the following inputs:

Name	Unit	Description	Format
SENT_Signal	[0|1]	Interface connection to the SENT sensor. Has to be connected to a digital input.	UFix_1_0
From_Interface_in	[-]	Bus which contains all relevant information which configure the SENT frame	Bus

Output

The SENT_RX main block has the following outputs with a maximal latency of 5:

Name	Unit	Description	Format
FPGASingalBus	Bus	Bus containing the most significant signals of this block	-
Dig_Out_Cpnfig	Bus	Configuraiton settings for the digital output. Only required for SPC (Master/Slave) mode	UFix_1_0

Processor Input

Block

Feedback information from the FPGA are available at the blocks outputs. The FPGA provides information via register in a raw format (UFix32_0), these raw values are reinterpreted inside this block.

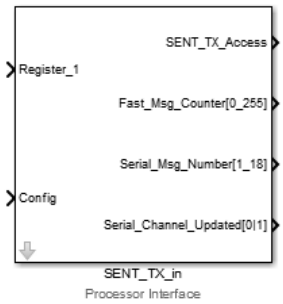


Figure 53: SENT_TX_in block

Block Dialog

The dialog provides only a short block description.

Input

The processor input block set has the following inputs:

Register 1: Provides the raw data coming from the FPGA

Config: Settings from the SENT_TX_out block

Output

The processor input block has the following outputs:

Name	Unit	Description	Range
SENT_TX_Access	[-]	Bus which provides information coming from the SENT_TX_in, respectively FPGA	
Fast_Msg_Counter	[-]	Counter for fast messages. Is incremented after every end of SENT frame	0 …255
Serial_Msg_Number	[-]	Actual serial message number, either up to 16 or 18 depending on the serial message configuration	0 … 18
Serial_Channel_Updated	[0|1]	When the FPGA is ready to receive new serial channel data, the output is set to high state for a single sample step	Bool

Interface Examples

Processor blocks

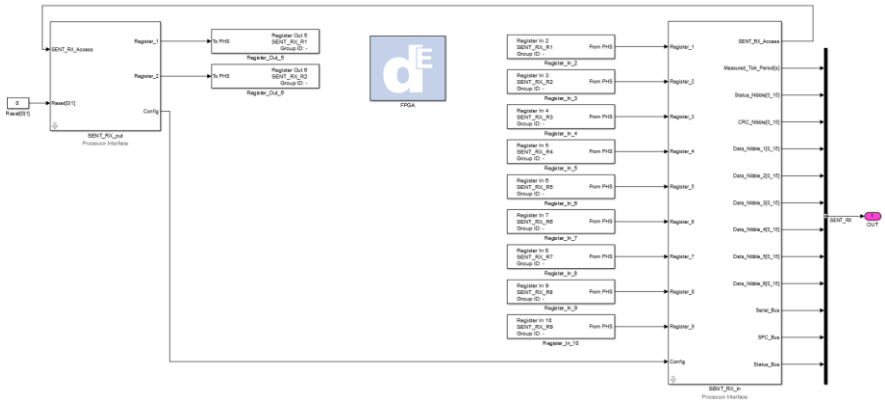


Figure 54: Processor interface

FPGA block

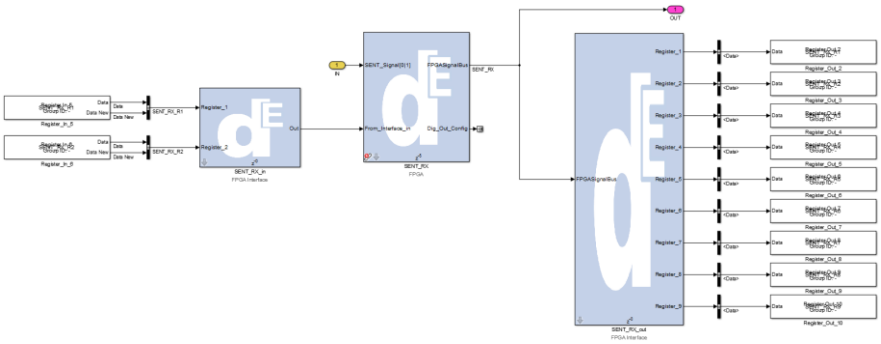


Figure 55: FPGA interface

ADC_2_6Proc

Objective

Provides the functionality to access **six** ADCs via processor interface. This block set is always used for those ADCs which are not connected to the dynamic FPGA model, so that they can be used like standard ADCs (like those from the DS2211 for example).

. 	The FPGA ADC scaling mode must be set to binary format (Bit)

The blockset contains the following elements:

- FPGA: ADC_2_6Proc (FPGA Main Component)
  FPGA Interface: ADC_2_6Proc_out (FPGA Interface)
  Processor Interface: ADC_2_6Proc_in (Processor Interface)

FPGA Main Component

Block

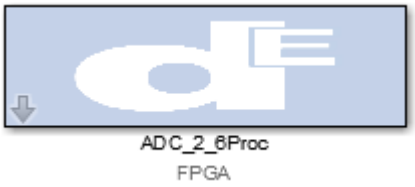


Figure 56: ADC_2_6Proc FPGA Main block

Block Dialog

The FPGA main blockset contains the following dialog.

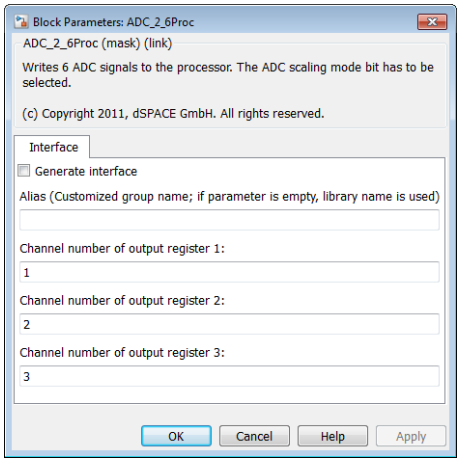


Figure 57: ADC_2_6Proc FPGA Main block dialog

The dialog blockset has no specific parameters only those for automatic interface generation.

	The parameters of the input and output registers can be defined for automatic interface generation (When checkbox is enabled) on the page Interface. The dialog also contains an additional button to start interface generation. For more details about automatic interface generation, refer to Automatic Interface Generation chapter.

Input The main block has no inputs:

Output The main has no output.

Processor Input

Block Adapts the FPGA raw signals for the processor side and converts them into voltages.

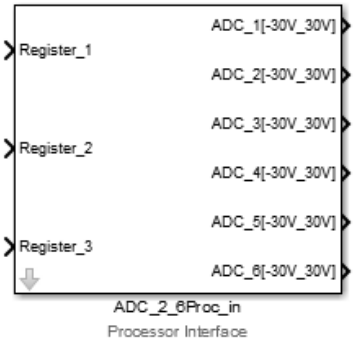


Figure 58: ADC_2_6Proc_in block

Block Dialog The dialog provides only a short block description.

Input The processor input block set has the following inputs:

Register 1_3: Provides the raw data coming from the FPGA

Output The processor input block provides on its outputs the measured voltages of the ADCs 1 to 6 coming from the DS5203 base board, with a resolution 14 Bit and a range of +-30V.

	The analog input channel on the FPGA has to be configured with the voltage range of +-30V and the scaling mode “Bit”!

Interface Examples

Processor blocks

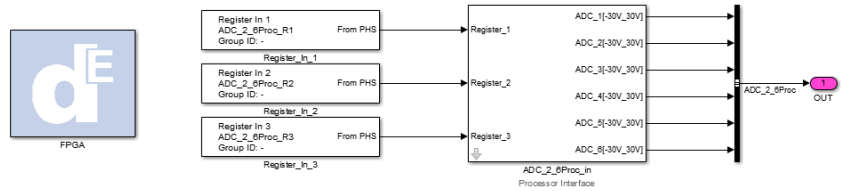


Figure 59: Processor interface

FPGA block

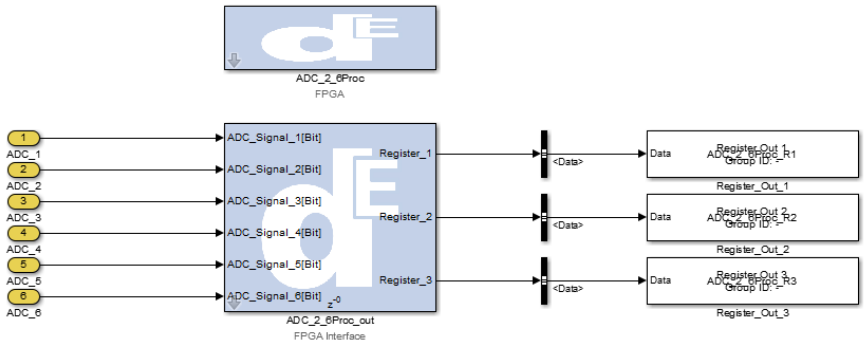


Figure 60: FPGA interface

Proc_2_6DAC

Objective

Provides the functionality to access **six** DACs via processor interface. This block set is always used for those DACs which are not connected to the dynamic FPGA model, so that they can be used like standard DACs (like those from the DS2211 for example).

. 	The FPGA DAC scaling mode must be set to binary format (Bit)

The blockset contains the following elements:

- Processor Interface: Proc_2_6DAC_out (Processor Interface)
- FPGA: Proc_2_6DAC (FPGA Main Component)
  FPGA Interface: Proc_2_6DAC_in (FPGA Interface)

Processor Output

Block

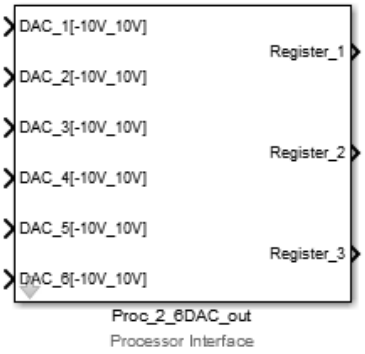


Figure 61: Proc_2_6DAC_out block

Block Dialog

The processor output blockset contains the following dialog:

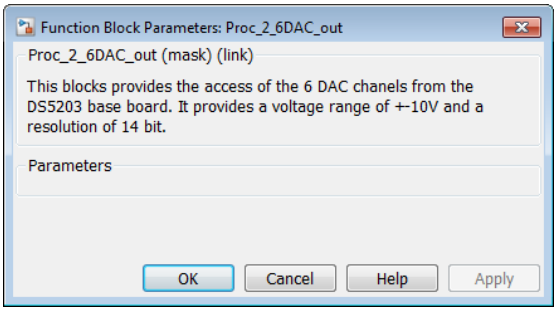


Figure 62: Proc_2_6DAC_out dialog

Input

The Proc_2_DAC_out block has the 6 set values for the DACs as input.

Output The processor out block provides the register raw data for all 6 channels.

FPGA Main Component

Block

Figure 63: Proc_2_6DAC FPGA Main block

Block Dialog The FPGA main blockset contains the following dialog.

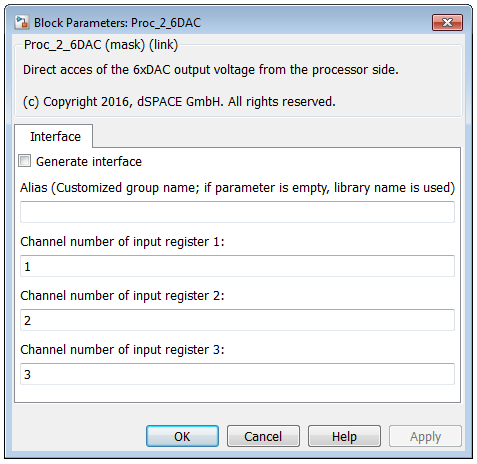


Figure 64: ADC_2_6Proc FPGA Main block dialog

The dialog blockset has no specific parameters only those for automatic interface generation.

	The parameters of the input and output registers can be defined for automatic interface generation (When checkbox is enabled) on the page Interface. The dialog also contains an additional button to start interface generation. For more details about automatic interface generation, refer to Automatic Interface Generation chapter.

Input The main block has no inputs:

Output The main has no output.

Interface Examples

Processor blocks

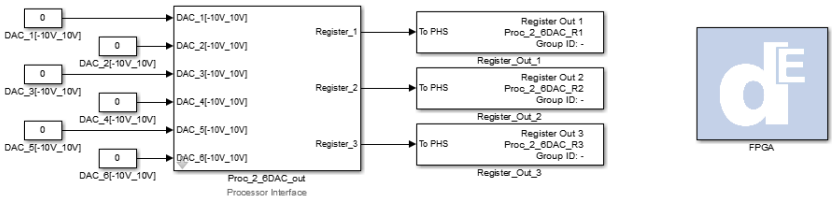


Figure 65: Processor interface

FPGA block

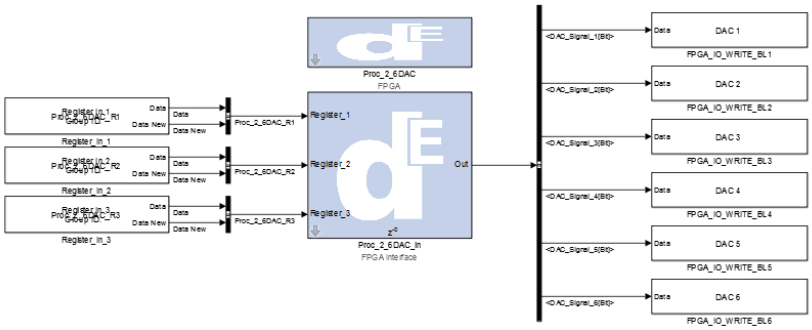


Figure 66: FPGA interface

Small APPS

Content

See the following sections for information about the blocks of the subsystem SMALL APPS. These blocks located provide an easy solution for the most used functions, such as edge detection, clock recovery, timing counter, switch off delay, etc. These blocks do not provide a complete interface blockset, but are designed to simplify the implementation of user-defined functionality in user-defined FPGA code.

EDGE_DETECTION

Description / Overview

The EDGE_DETECTION block detects a falling or rising edge or both (change of input) and generates an output trigger impulse. Additional to that, the block detects if a signal has changed in relation to a specified previous sample (the delay can be specified, that is the very important point). This can for example be used to compare time-multiplexed signals.

Block

Figure 67: EDGE_DETECTION block for Comparision Sample setting = 1

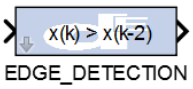


Figure 68: EDGE_DETECTION block for Comparision Sample setting > 1

Block Dialog

The FPGA block contains the following dialog.

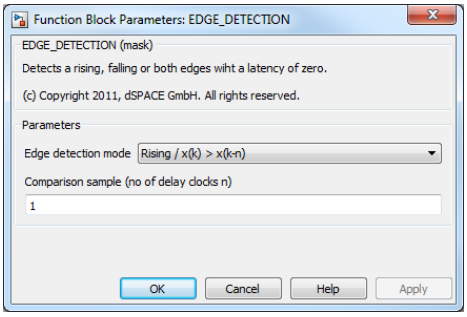


Figure 69: EDGE_DETECTION dialog

The dialog blockset has the following parameters:

Name	Unit	Description	Range
Edge detection mode	[-]	Lets you select the edge detection mode	- Falling / x(k) < x(k-n) - Rising / x(k) > x(k-n) - Both / x(k) != x(k-n)

Input

The main block has the following inputs:

Name	Unit	Description	Format
In	[-]	Input of the edge detection	User-defined

Output

The EDGE_DETECTION block has the following outputs:

Name	Unit	Description	Format
Out	[0|1]	Output of the edge detection block	Bool

Workflow / Example

The following example shows the transfer behavior of the FPGA block.

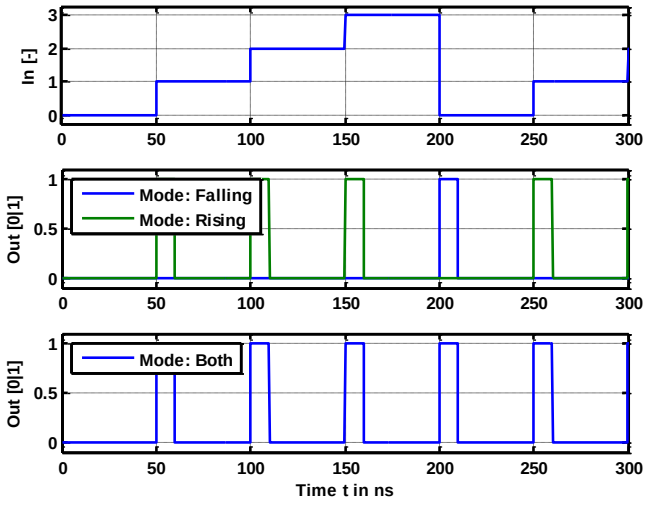


Figure 70: Workflow of the EDGE_DETECTION block

RELAY

Description / Overview

The RELAY block detects when the input signal crosses the on value in positive direction (rising edge), the output state is set to high state. When the input signal crosses the off value in negative direction (falling edge), the output state is set to low state.

Block

Figure 71: RELAY block

Block Dialog

The FPGA block contains the following dialog.

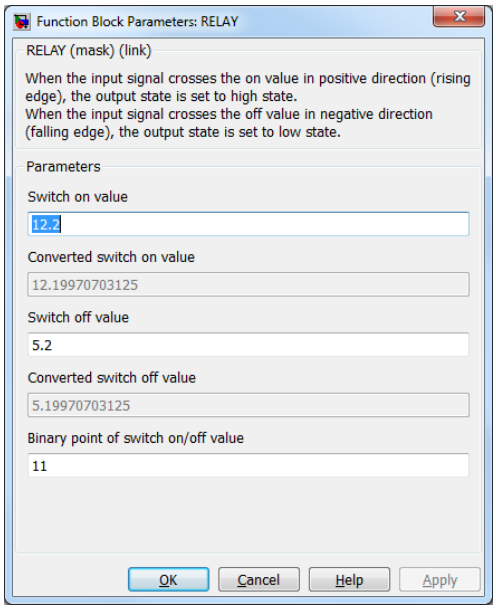


Figure 72: RELAY dialog

The dialog blockset has the following parameters:

Name	Unit	Description	Range
Switch on value	[-]	Switch on value	-
Converted switch on value	[-]	Converted switch on value in fixed-point format	-
Switch off value	[-]	Switch off value	-
Converted switch off value	[-]	Converted switch off value in fixed-point format	-
Binary point of switch on/off value	[-]	Binary point of switch on/off value	-

Input

The main block has the following inputs:

Name	Unit	Description	Format
In	[-]	Input	User-defined

Output

The RELAY block has the following outputs:

Name	Unit	Description	Format
Out	[0|1]	Output	Bool

COMP2CONST

Description / Overview

The Comp2Const block compares the input value for different modes to a user defined constant value.The number of bits and the binary point of the constant value can be user defined.

Block

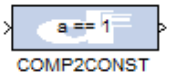


Figure 73: COMP2CONST block

Block Dialog

The FPGA block contains the following dialog.

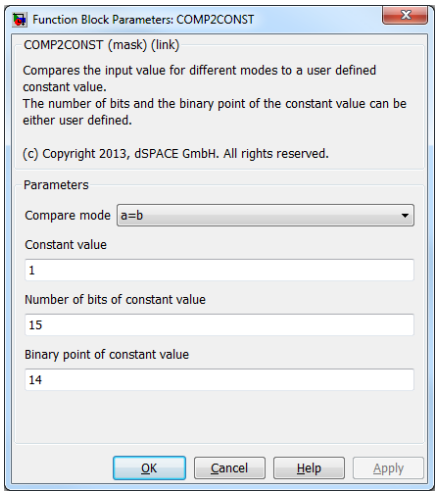


Figure 74: COMP2CONST dialog

The dialog blockset has the following parameters:

Name	Unit	Description	Range
Compare mode	[-]	Define the actual compare mode; available modes: a = b a! = b a < b a > b a <= b a >= b	-
Const value	[-]	Constant value	-
Number of bits	[-]	Number of bits of the constant value	-
Binary point	[-]	Binary point of the constant value	-

Input

The main block has the following inputs:

Name	Unit	Description	Format
In	[-]	Input	User-defined

Output

The COMP2CONST block has the following outputs:

Name	Unit	Description	Format
Out	[-]	Output	UFix_1_0

SWITCH OFF DELAY

Description / Overview

Delays a Boolean input signal with the number of parameterized delay steps with an additional latency of zero.

Block

Figure 75: SWITCH_OFF_DELAY block

Block Dialog

The FPGA block contains the following dialog.

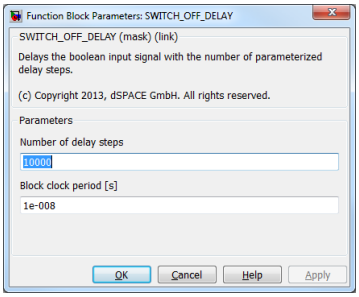


Figure 76: SWITCH_OFF_DELAY dialog

The dialog blockset has the following parameters:

Name	Unit	Description	Range
Number of Delay Steps	[-]	Number of delay steps.	0 … 2^10-1
FPGA clock period	[s]	Clock period of the FPGA	-

Input

The main block has the following inputs:

Name	Unit	Description	Format
In	[-]	Input	Bool

Output

The SWITCH_OFF_DELAY block has the following outputs:

Name	Unit	Description	Format
Out	[-]	Delayed output	Bool

Workflow / Example

The following example shows the transfer behavior of the FPGA block.

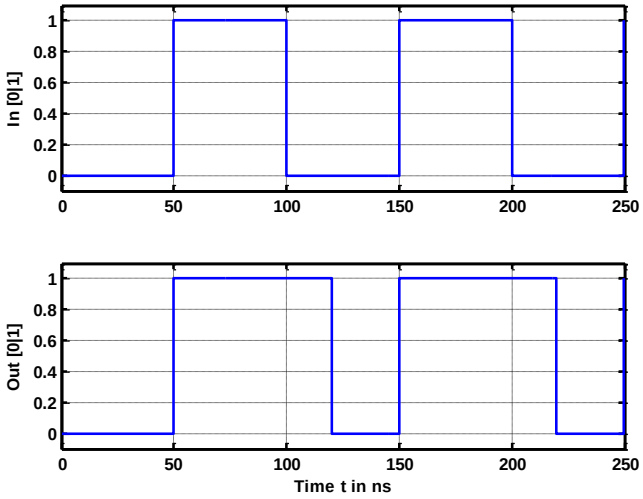


Figure 77: Workflow of the ABS block

MULTIFUNCTION BLOCK

Description / Overview

The MULTIFUNCTION_BLOCK combines the functionalities of a Delay, a Switch Left and a Switch Right function.

Be careful of the bit length and the binary point of this block's input and output. The output has the same bit length and binary point as the input.

Block

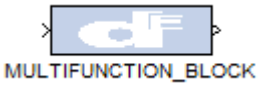


Figure 78: MULTIFUNCTION_BLOCK block

Block Dialog

The FPGA block contains the following dialog.

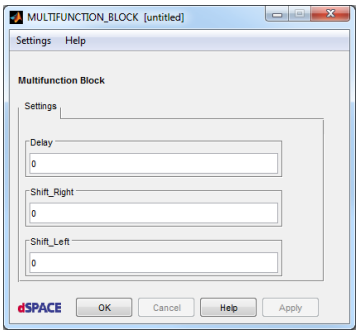


Figure 79: MULTIFUNCTION_BLOCK dialog

The MULTIFUNCTION_BLOCK blockset has the following parameters:

Name	Unit	Description	Range
Delay	[-]	Number of delay steps	User-defined
Shift Right	[-]	Number of shift bits to the right	User-defined
Shift Left	[-]	Number of shift bits to the left	User-defined

Input

The main block has the following inputs:

Name	Unit	Description	Format
In	[-]	Input	Unsigned and signed values

Output

The MULTIFUNCTION_BLOCK block has the following outputs:

Name	Unit	Description	Format
Out	[-]	Output	Unsigned and signed values

Workflow / Example

The following example shows the transfer behavior of the FPGA block with a delay of three and a right shift of one bit.

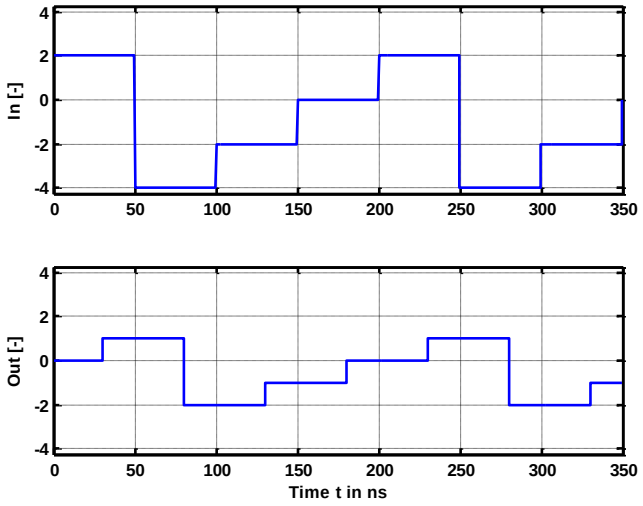


Figure 80: Workflow of the MULTIFUNCTION_BLOCK block

VALID_EDGES

Description / Overview

The VALID_EDGES block outputs a valid edge if the edge is set to level zero or one for a defined length of FPGA sample steps.

Block

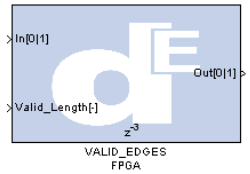


Figure 81: VALID_EDGES block

Block Dialog

The dialog only provides a short block description with a parameter to configure the clock period of the FPGA.

Input

The main block has the following inputs:

Name	Unit	Description	Format
In	[-]	Input	U_Fix_1_0
Valid_Length	[-]	Detection of the valid length of edge which is interpreted as a valid edge	U_Fix_x_0

Output

The VALID_EDGES block has the following outputs:

Name	Unit	Description	Format
Out	[-]	Valid edges	Bool

Workflow / Example

The following example shows the transfer behavior of the FPGA block with a valid length of 10 FPGA samples.

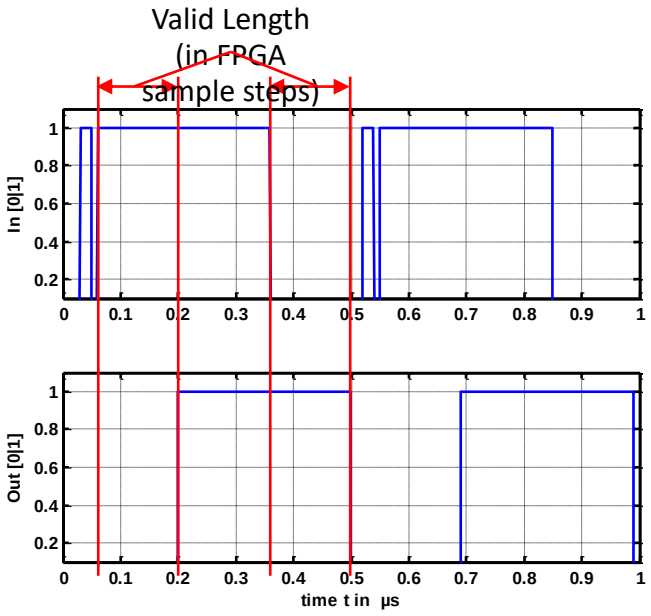


Figure 82: Workflow of the VALID_EDGES block

Processor based parts

Overview

Overview

The XSG Sent library also contains processor-based items. The following processor parts are available:

- CRC (Checksum calculation)

The following section describes the different functionalities in detail.

CRC

Content

See the following sections for information about the blocks of the subsystem CRC. These blocks are used for checksum calculation.

CRC

Description / Overview

The data nibbles 1 to 6 (vector) are fed to this block in a multiplexed structure (using Simulink Mux block). The output of this block is calculated checksum which depends on the data nibble values.

Block

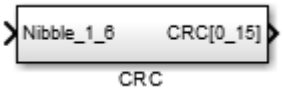


Figure 83: CRC block

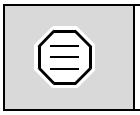
Demo

Overview

Overview

The XSG Sent library provides two demo models.

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



The XSG Sent Demos require only the XSG Sent Interface library.

The demos are precompiled for the following hardware configurations:

Processor Model	FPGA Model
DS1006	DS5203 (7K325)

In case that a different hardware configuration should be used, e.g. DS5203 (7K410) with mounted multi IO module (DS5203M1), it is necessary to rebuild the **FPGA** model.

In case that different processing units should be used (e.g. DS1007) it is necessary to rebuild the **processor** model

Demo Installation

Initial Installation

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

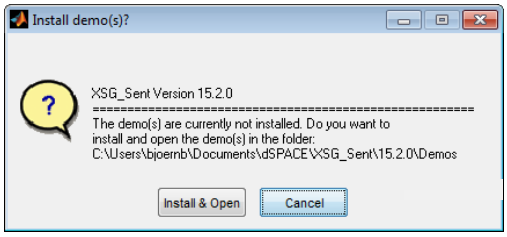


Figure 84: Demo installation dialog (first installation)

The Demo models and experiments will be installed to: My Documents\ dSPACE\ XSG_SENT\<Version_Number>. Hence, the demo folder installation path, shown in the demo installation dialog will adjust, according to the user's login name.

Open Demo

After the Demos have been installed, the user can choose which demo shall be opend.

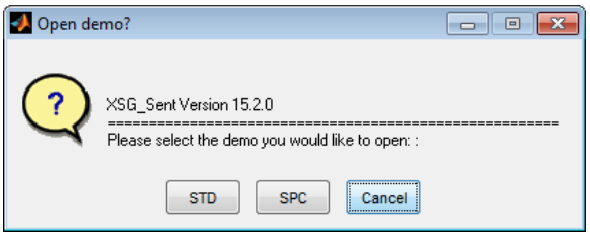


Figure 85: Select Demo

STD: Demo including standard SENT implementation after SAE J2716 (Revised JAN2010).

SPC: Demo including bidirectional SPC (Master/Slave) communication.

	Both demos are available for DS5203 7K325 and DS1006 only! When a different processor platform shall be used (e.g. DS1007), the processor model must be recompiled for the desired target platform. If the DS5203 7 K 325 stays unchanged, no FPGA build is required.

Demo Folder Structure

The demos are copied, using the following folder structure:

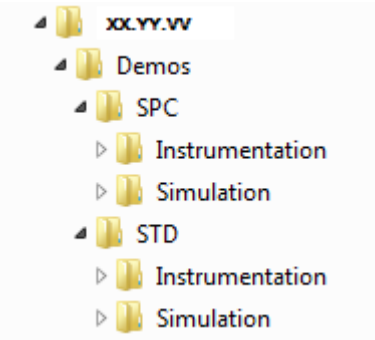


Figure 86: Demo folder structure after demo model installation

The Simulation folder contains the Matlab Simulink model.

The Instrumentation folder contains the ControlDesk 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 models, or to reinstall the demo, as shown in the figure below.

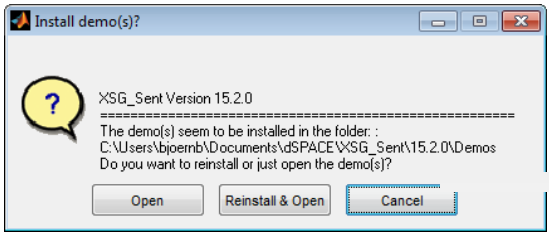


Figure 87: Demo installation dialog (after initial installation)

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

STD Demo

Overview

After the demos are installed, the actual used platform (PHS-Bus based) is used and the specific model will be loaded.

The Sent demo shows an exemplary integration of 16 SENT transmission channels and 8 receiving channels. Where the first TX channel (SENT_TX_ch01) and the first receiving channel (SENT_RX_ch01) are configured for serial message transmission respectively reception. TX channel 1 is sending is sending on the first 3 data nibbles (1-3) a ramp signal (from 0 to 4095) and the last 3 data nibbles (4-6) the inverted signal (from 4095 to 0). The checksum is calculated separately at every processor cycle. In this demonstration the data from the serial channel is not dynamically changing, but can be modified directly via ControlDesk.

The available analog onboard IO of the DS5203 can be accessed via the ADC_2_6Proc respectively Proc_2_6DAC interface.

Since the TX channels (1-8) are connected to the same IO pins like the RX channels (1-8) it is not necessary connect them hardware wire to achieve a valid communication line.

Processor Model

Here the model root of the demo is displayed:

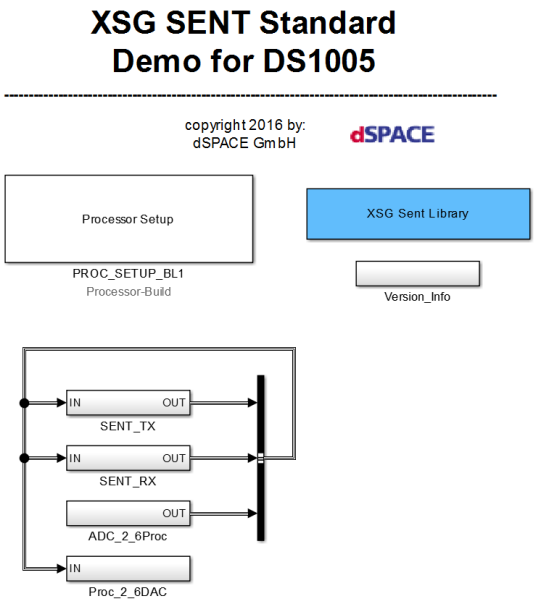


Figure 88: Sent STD demo Simulink model root

The processor setup block is located in the top left corner of the model. Here you can find the Link to the actual FPGA configuration which will be downloaded to the target FPGA

The FPGA model is directly placed below the link to the XSG_SENT library.

Furthermore, the processor interfaces for the FPGA model are located on the model root and are divided into subsystem for the corresponding component. It includes SENT_TX (16 interface channels for SENT TX), SENT_RX (16 interface channels for SENT RX), ADC_2_6Proc and Proc_2_6DAC.

FPGA Model

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

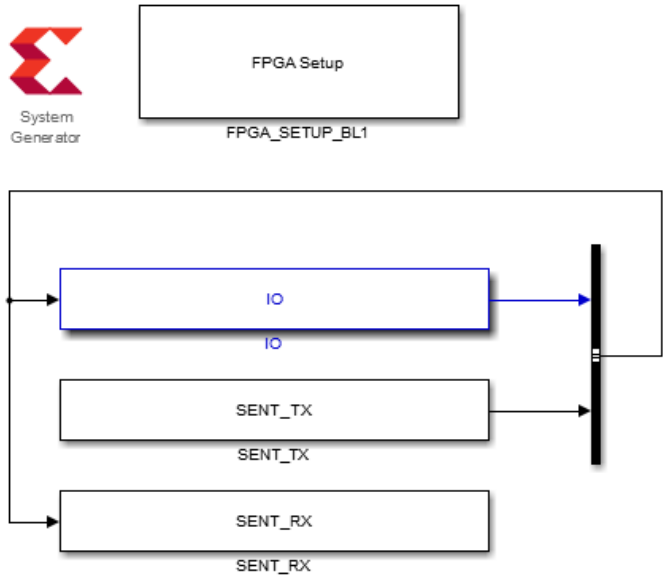


Figure 89: Sent STD demo FPGA model

The FPGA setup block is located in the top of the FPGA model. Here the specification of desired hardware platform (framework) has to be made. All hardware in- and outputs are located in the subsystem IO. All necessary model inputs from the PHS-Bus, which define the structure and the content of a the SENT frame, are located in subsystems SENT_TX/RX individually for each channel. Model variables (e.g. the SENT TX signal) which are necessary from one subsystem (SENT_TX) to another subsystems (IO) are routed over the overall Bus. The behavior for the SENT_RX implementation, here the signal is coming from the IO, received inside the SENT_RX_ch_xx subsystem and send its information via PHS bus to the processor.

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 to 6	No special function, can be freely supplied by the user
DAC 1 to 6	No special function, can be freely supplied by the user
Digital Output 1 to 16	SENT_TX channel 1 to 16
Digital Input 1 to 8	SENT_RX channel 1 to 8
Digital Input 9 to 16	Not used

ControlDesk Environment

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

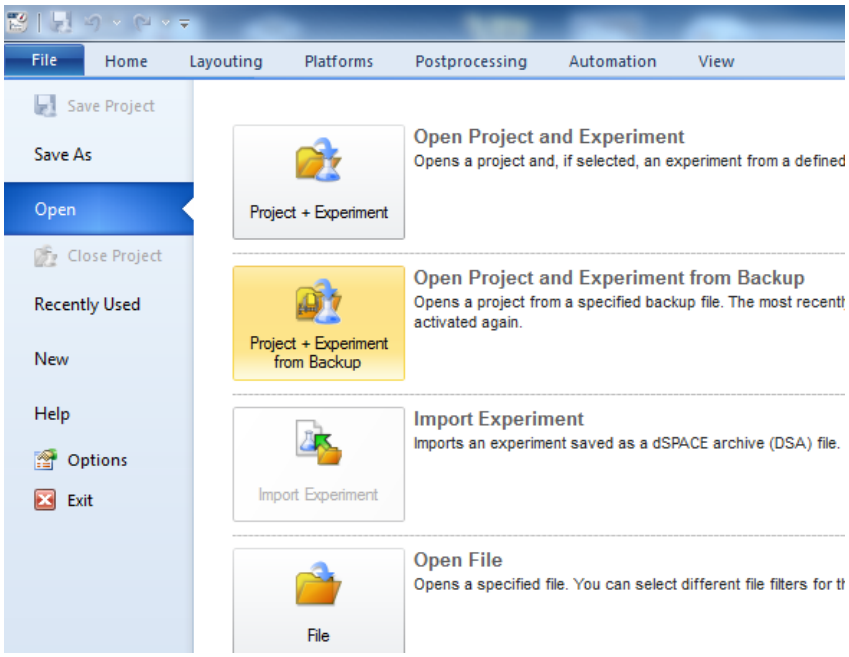


Figure 90: Open ControlDesk backup

Navigate to the demo installation directory at: My Documents\ dSPACE\ XSG_Sent\<Version>\ Demos\ STD \ Instrumentation\ and select the sent_std_demo.ZIP file.

After the backup is loaded, the following layouts is shown:

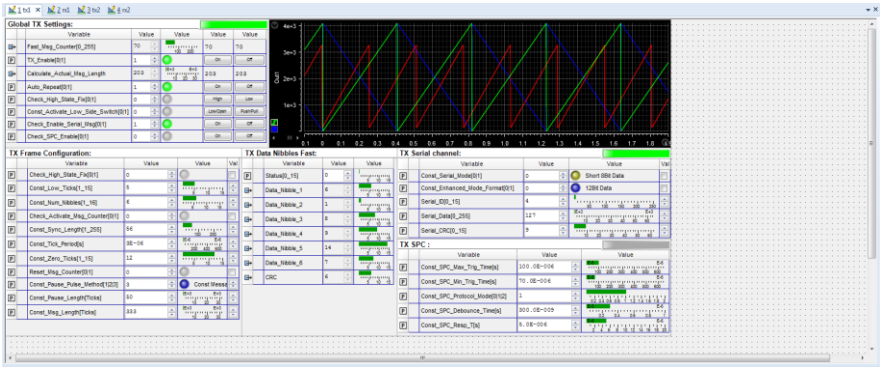


Figure 91: ControlDesk Sent STD, tx1

Now, register the processor board and start the measurement. The real-time application will be downloaded.

After the real-time application is downloaded and the real-time process is executed, the XSG Sent Lib Version Info will show the recognized versions numbers for the FPGA and Processor model.
The “tx1” layout is organized into function groups. On the upper left hand side the “Gloabal TX Frame Settings” are available, like e.g. enabling the communication, is standard SENT or SPC, does it transport serial message, etc …
The “TX Frame Configuration” describes the SENT frame itself, like tick period, number of data nibbles, synchronization length, etc. The variable array “TX Data Nibbles Fast” display the actual value of each data nibble, where its value is dynamically changing since a ramp generator is connected to it. The “Serial channel” provides the modification ability to change the serial transmission mode, e.g. from short serial message to enhanced serial message.
This layout provides also the possibility to change the parameters for SPC (Master/Slave) communication, this function is deactivated inside the demo (indicated also by the LED in the upper right corner).

The plotter shows the actual transmitted values for the data nibbles 1 to 3 (green), the data nibbles 4 to 6 (blue) and the message counter for the fast messages (red).

The “rx1” layout is displaying the received data from the SENT channel no. 1.

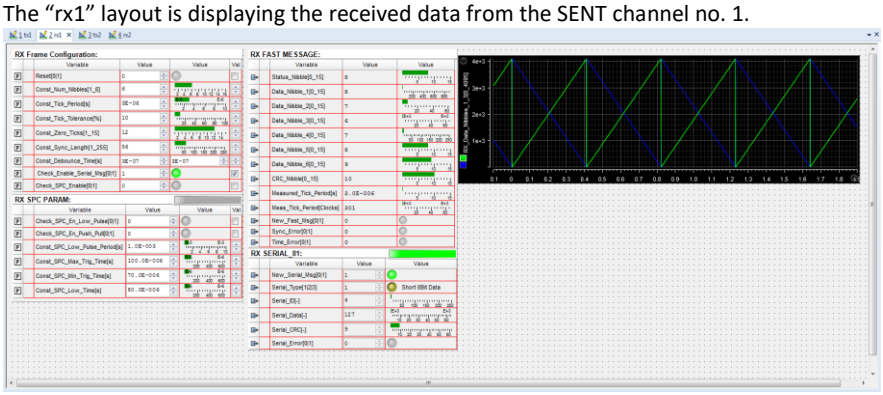


Figure 92: ControlDesk Sent STD, rx1

The layout “tx2” has no dynamic model inputs. The values for data nibble 1to 6 as well as the checksum can be defined by the user individually.

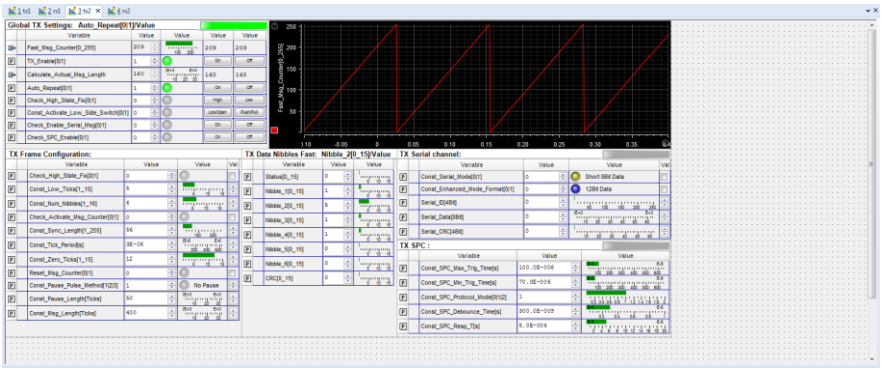


Figure 93: ControlDesk Sent STD Demo, tx2

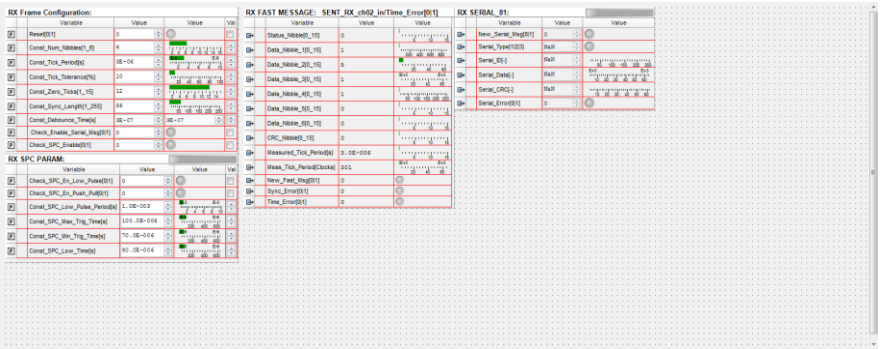


Figure 94: ControlDesk Sent STD Demo, rx2

SPC Demo

Overview

After the demos are installed, the actual used platform (PHS-Bus based) is used and the specific model will be loaded.

The Sent demo shows an exemplary integration of 8 SENT transmission channels (Slave) and 8 receiving channels (Master), which are configured in SPC mode. In SPC mode the Master (in automotive area the ECU) and the Slave (in automotive area the sensor) get the high voltage level (Vcc, e.g. 5V) on the signal line via pull up resistance (R e.g. 1k Ohm) where the Master and Slave itself actively pull to ground. The schematic below shows the SPC hardware behavior:

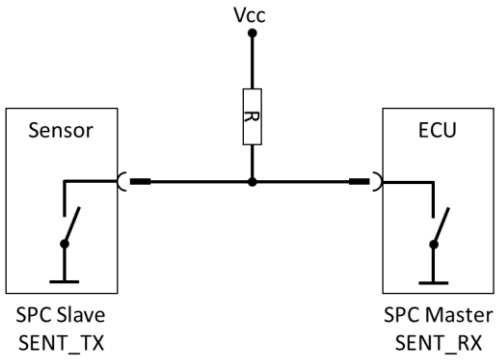


Figure 95: External supply SPC mode

The first TX channel (SENT_TX_ch01) and the first receiving channel (SENT_RX_ch01) are configured for serial message transmission respectively reception. TX channel 1 is sending is sending on the first 3 data nibbles (1-3) a ramp signal (from 0 to 4095) and the last 3 data nibbles (4-6) the inverted signal (from 4095 to 0). The checksum is calculated separately at every processor cycle. In this demonstration the data from the serial channel is not dynamically changing, but can be modified directly via ControlDesk.

The available analog onboard IO of the DS5203 can be accessed via the ADC_2_6Proc respectively Proc_2_6DAC interface.

Since SPC mode requires the supply to Vcc the wiring in drawing above has to be realized for a valid SENT communication. The SPC Master is simulated via “SENT_RX_ch01” (Dig_IO 9) and the SPC slave is simulated via “SENT_TX_ch01” (Dig_IO 1).

Processor Model

Here the model root of the demo is displayed:

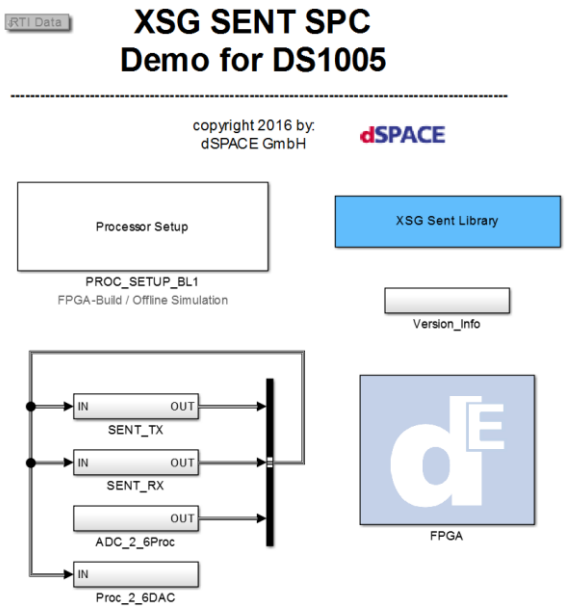


Figure 96: Sent SPC demo Simulink model root

The processor setup block is located in the top left corner of the model. Here you can find the Link to the actual FPGA configuration which will be downloaded to the target FPGA

The FPGA model is directly placed below the link to the XSG_SENT library.

Furthermore, the processor interfaces for the FPGA model are located on the model root and are divided into subsystem for the corresponding component. It includes SENT_TX (8 interface channels for SENT TX), SENT_RX (8 interface channels for SENT RX), ADC_2_6Proc and Proc_2_6DAC.

FPGA Model

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

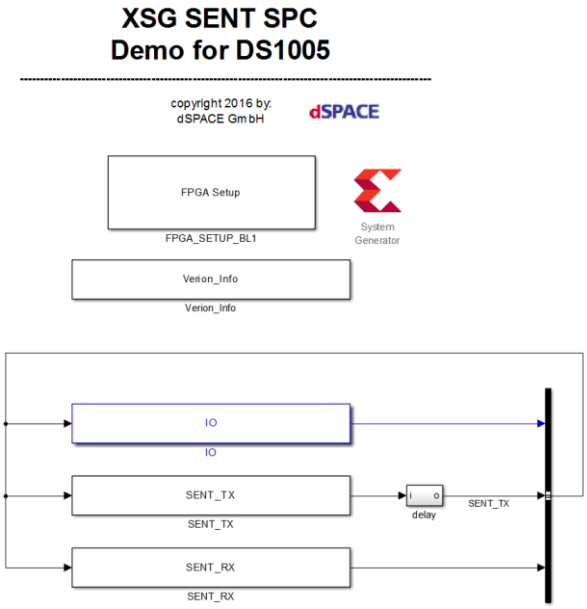


Figure 97: Sent SPC demo FPGA model

The FPGA setup block is located in the top of the FPGA model. Here the specification of desired hardware platform (framework) has to be made. All hardware in- and outputs are located in the subsystem IO. All necessary model inputs from the PHS-Bus, which define the structure and the content of a the SENT frame, are located in subsystems SENT_TX/RX individually for each channel. Model variables (e.g. the SENT TX signal) which are necessary from one subsystem (SENT_TX) to another subsystems (IO) are routed over the overall Bus. The behavior for the SENT_RX implementation, here the signal is coming from the IO, received inside the SENT_RX_ch_xx subsystem and send its information via PHS bus to the processor.

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 to 6	No special function, can be freely supplied by the user
DAC 1 to 6	No special function, can be freely supplied by the user
Digital Input Output 1 to 8, bidirectional	SENT_TX channel 1 to 8
Digital Input Output 9 to 16, bidirectional	SENT_RX channel 1 to 8

ControlDesk Environment

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

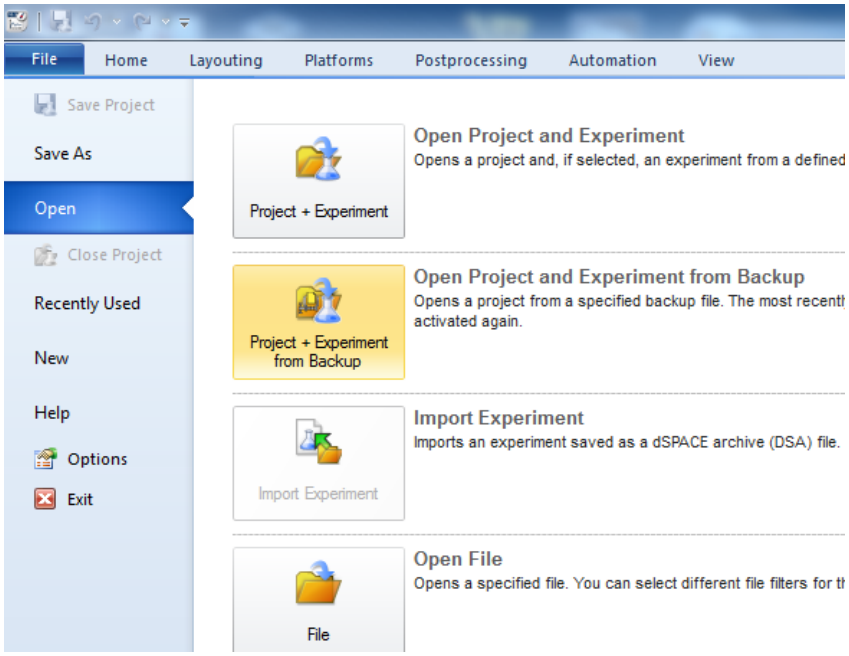


Figure 98: Open ControlDesk backup

Navigate to the demo installation directory at: My Documents\ dSPACE\ XSG_Sent\<Version>\ Demos\ SPC \ Instrumentation\ and select the sent_spc_demo.ZIP file.

After the backup is loaded, the following layouts is shown:

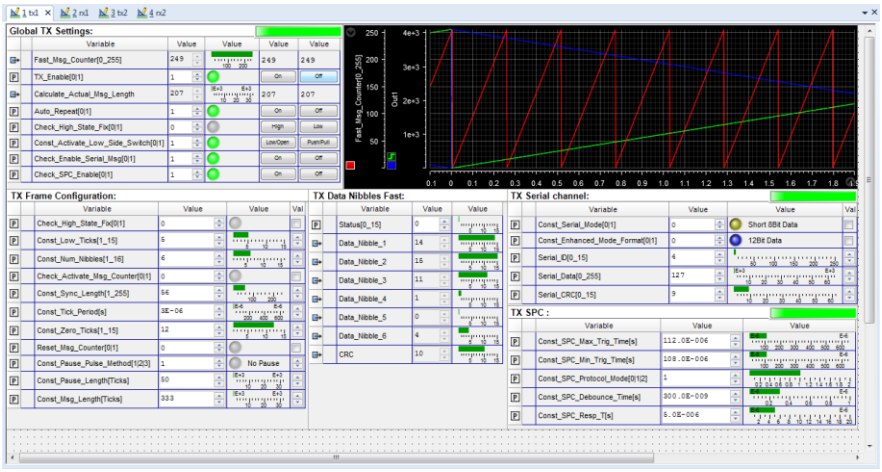


Figure 99: ControlDesk Sent SPC, tx1

Now, register the processor board and start the measurement. The real-time application will be downloaded.

After the real-time application is downloaded and the real-time process is executed, the XSG Sent Lib Version Info will show the recognized versions numbers for the FPGA and Processor model.

The “tx1” layout is organized into function groups. On the upper left hand side the “Gloabal TX Frame Settings” are available, like e.g. enabling the communication, is standard SENT or SPC, does it transport serial message, etc …
The “TX Frame Configuration” describes the SENT frame itself, like tick period, number of data nibbles, synchronization length, etc. The variable array “TX Data Nibbles Fast” display the actual value of each data nibble, where its value is dynamically changing since a ramp generator is connected to it. The “Serial channel” provides the modification ability to change the serial transmission mode, e.g. from short serial message to enhanced serial message.
The SPC (Master/Slave) specific settings, like reacting on a master pulse with a specific length, response time, etc. are accessible on the lower right variable array.

The plotter shows the actual transmitted values for the data nibbles 1 to 3 (green), the data nibbles 4 to 6 (blue) and the message counter for the fast messages (red).

The “rx1” layout is displaying the received data from the SENT channel no. 1. Next to this the settings for SPC Master impulse (low pulse) generation is accessible via “RX SPC PARAM”

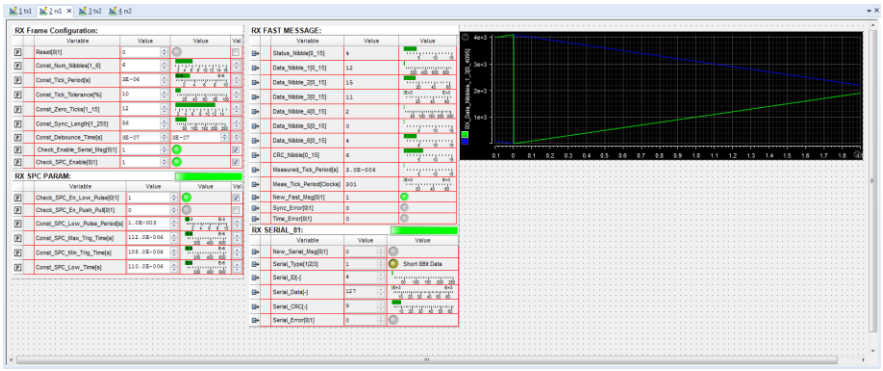


Figure 100: ControlDesk Sent SPC, rx1

The layout “tx2” has no dynamic model inputs. The values for data nibble 1to 6 as well as the checksum can be defined by the user individually.

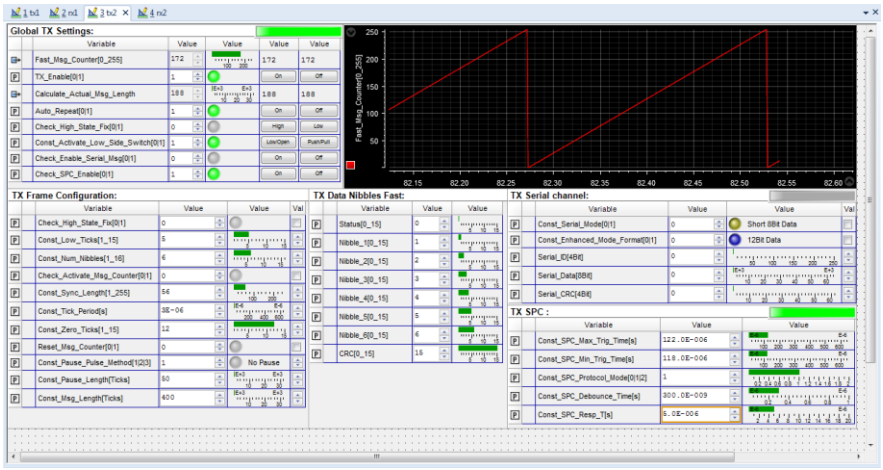


Figure 101: ControlDesk Sent SPC Demo, tx2

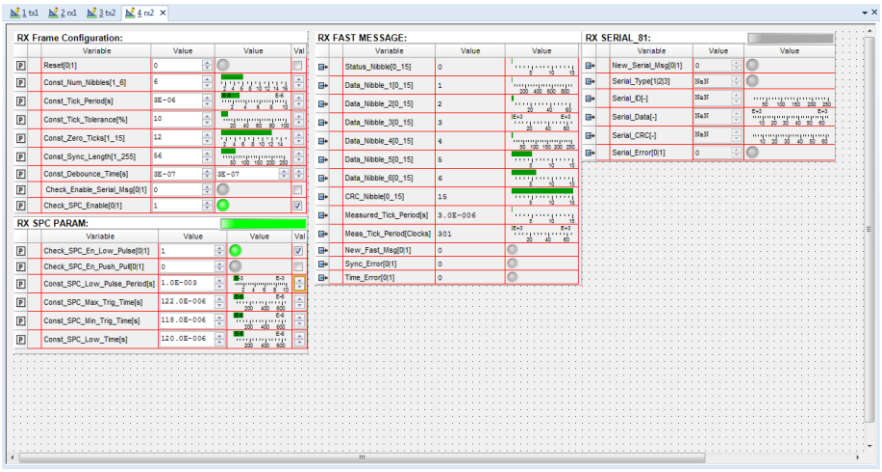


Figure 102: ControlDesk Sent SPC Demo, rx2

Working with a fix FPGA configuration

Overview

Overview

For those who do not want to program the FPGA on their own there are 2 precompiled FPGA configurations available. These FPGA configuration are also available inside the Demos, for further information please refer to the chapter Demo.

Working with a precompiled FPGA configuration (*.ini) requires only:

XSG Sent Interface Library (processor-based blocks)

dSPACE Release	Operating System	MATLAB	dSPACE HW Platform
2019-B (64-bit)	Windows 7 Windows 10 (64-bit)	R2018a R2018b R2019a R2019b (64-bit)	DS5203 7K325

Figure 103: Compatibility matrix for XSG Sent Interface 20.1

FPGA Configuration

The FPGA configurations contain below features:

1. **std_sent_16tx_8rx_7K325_V01.ini**
    a. Standard SENT SAE J2716 (Revised JAN2010)
    b. 16 TX channels (Dig_IO 1-16)
    c. 8 RX channels (Dig_IO 1-8)
        i. TX channel 1-8 share the same IO like RX channel 1-8
    d. Measurement of ADC 1-6
    e. Control of DAC 1-6
2. **sent_spc_8tx_8rx_7K325_V01.ini**
    a. Standard SENT SAE J2716 (Revised JAN2010 + SPC
    b. 8 TX channels

c. 8 RX channels

d. Measurement of ADC 1-6

e. Control of DAC 1-6

The XSG_SENT_Interface library can be opened by calling “XSG_SentInterface_lib” inside the Matlab command window or via library browser. The FPGA configurations can be easily accessed via double click on the “FPGA Config” subsystem.

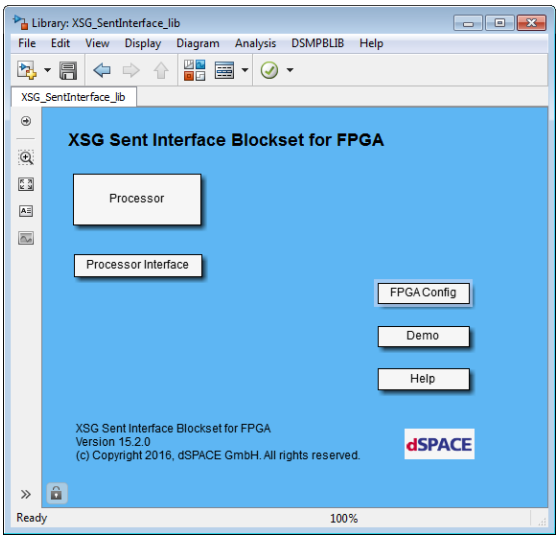


Figure 104: Functions of the XSG Sent library

This will open an explorer window which points to the FPGA configurations, or navigate to the folder MyDocuments\dSPACE\XSG_Sent\<Version_Number>\FPGA_Configuration.

Setting up a SENT model from the scratch

Integration of a FPGA Configuration

To be able to download this configuration to the target platform (DS5203 7K325), a **Processor Setup** block from the RTIFPGA library has to be inserted inside the Simulink model on model root. The library can be opened by using the command “rtifpga” in the Matlab Command Window or by using the Simulink Library Browser→dSPACE RTIFPGA Programming Blockset→PROCESSOR INTERFACE.

Open the Setup GUI, navigate to the Advanced page and add e.g. the “spc_sent_8tx_8rx_7K325_V01” file

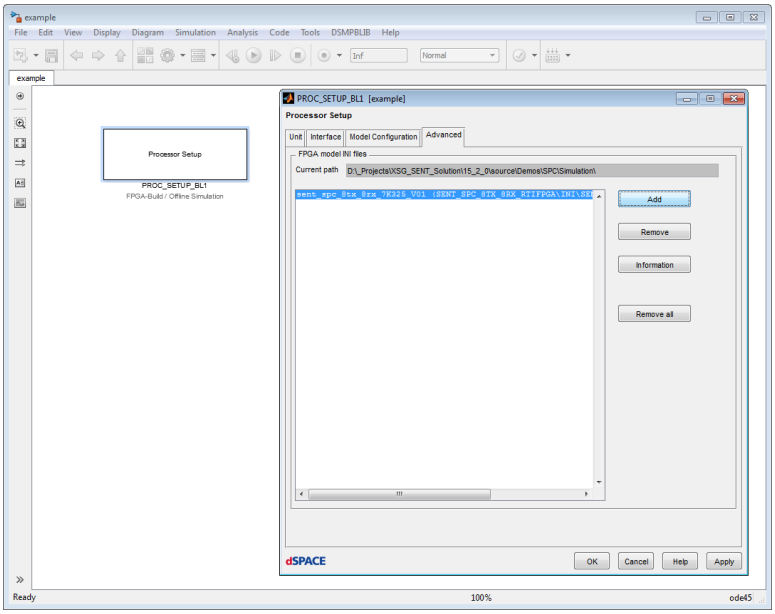


Figure 105: Adding FPGA configuration to Simulink mdl

Navigate to the Unit page and specify the SENT configuration for the desired FPGA board.

Choose the right Programming option, e.g. into RAM.

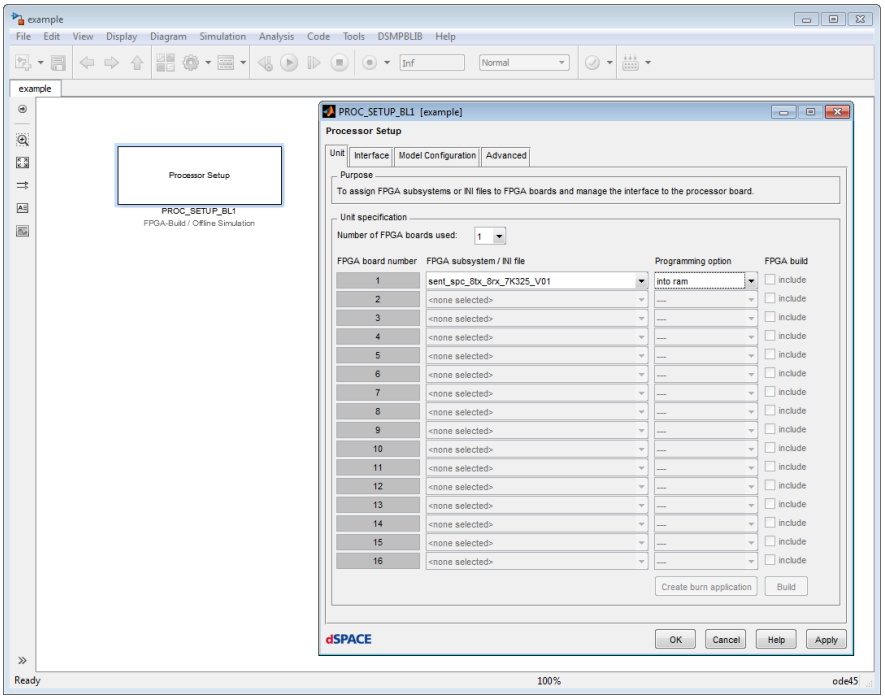


Figure 106: Specification of programming option

Navigate to the “Model Configuration” tab and switch model mode to “Processor-Build”.

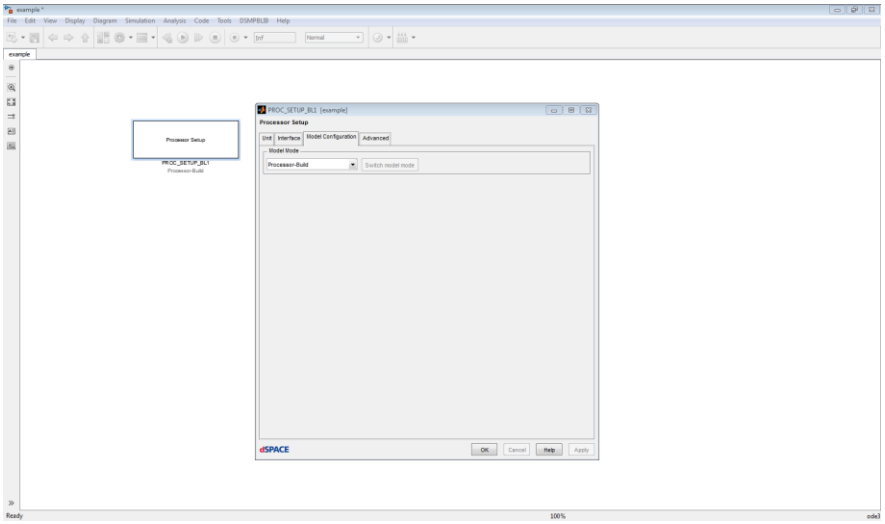


Figure 107: Model mode: Processor-Build

Below the implementation is shown as an example for a SENT sensor (SENT_TX), for a receiving channel the workflow is completely the same.

Open the XSG Sent Interface Library, navigate to “Processor Interface/TX” and copy the SENT_TX_out & SENT_TX_in blocks inside your Simulink model. Ensure that the connection on the “SENT_TX_Access” and “Config” ports is established, like shown below:

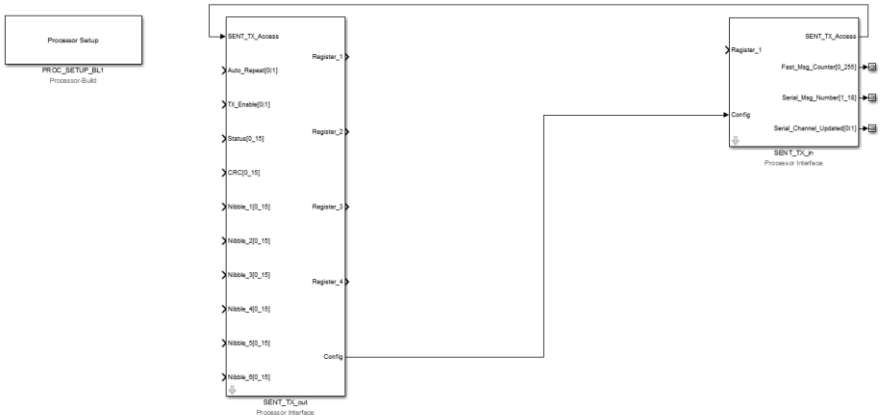


Figure 108: Specification of programming option

Integrate “PROC_XDATA_READ_BL1” & “PROC_XDATA_WRITE_BL1” blocks from dSPACE RTIFPGA Programming Blockset→PROCESSOR INTERFACE inside the model. Configure them as registers with the right channel number and connect them to the ports of the processor interface.

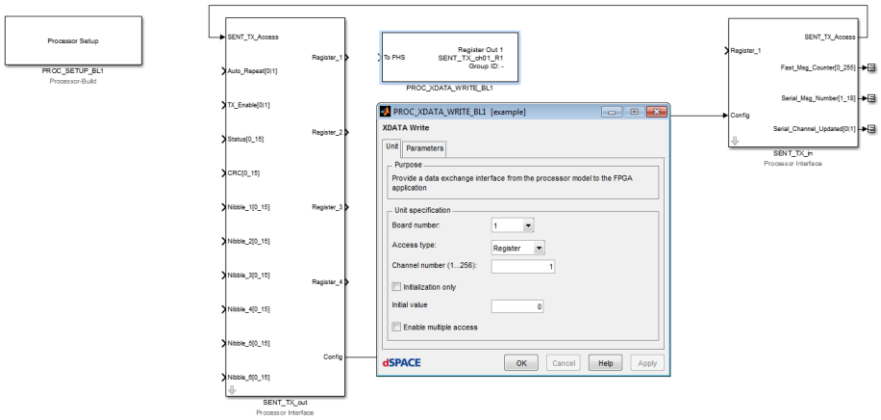


Figure 109: PROC_XDATA_WRITE settings

The tables below show the mapping structure for the SENT_TX/RX channels for both FPGA configurations.

SENT channel	XDATA Write channel number	XDATA Read channel number	IO
TX_ch_01	1-4	1	Digital_IO 1
TX_ch_02	5-8	2	Digital_IO 2
TX_ch_03	9-12	3	Digital_IO 3
TX_ch_04	13-16	4	Digital_IO 4
TX_ch_05	17-20	5	Digital_IO 5
TX_ch_06	21-24	6	Digital_IO 6
TX_ch_07	25-28	7	Digital_IO 7
TX_ch_08	29-32	8	Digital_IO 8
RX_ch_01	65-66	17-25	Digital_IO 9
RX_ch_02	67-68	26-34	Digital_IO 10
RX_ch_03	69-70	35-43	Digital_IO 11
RX_ch_04	71-72	44-52	Digital_IO 12
RX_ch_05	73-74	53-61	Digital_IO 13
RX_ch_06	75-76	62-70	Digital_IO 14
RX_ch_07	77-78	71-79	Digital_IO 15
RX_ch_08	79-80	80-88	Digital_IO 16
ADC_2_6Proc	-	89-91	ADC 1-6
Proc_2_6DAC	81-83	-	DAC 1-6
Version_Info	33	9	-

Figure 110: Mapping table for: spc_sent_8tx_8rx_7K325_V01.ini

SENT channel	XDATA Write channel number	XDATA Read channel number	IO
TX_ch_01	1-4	1	Digital_Out 1
TX_ch_02	5-8	2	Digital_Out 2
TX_ch_03	9-12	3	Digital_Out 3
TX_ch_04	13-16	4	Digital_Out 4
TX_ch_05	17-20	5	Digital_Out 5
TX_ch_06	21-24	6	Digital_Out 6
TX_ch_07	25-28	7	Digital_Out 7
TX_ch_08	29-32	8	Digital_Out 8
TX_ch_09	33-36	8	Digital_Out 9
TX_ch_10	37-40	8	Digital_Out 10
TX_ch_11	41-44	8	Digital_Out 11
TX_ch_12	45-48	8	Digital_Out 12
TX_ch_13	49-52	8	Digital_Out 13
TX_ch_14	53-56	8	Digital_Out 14
TX_ch_15	57-60	8	Digital_Out 15
TX_ch_16	61-64	8	Digital_Out 16
RX_ch_01	65-66	17-25	Digital_In 1
RX_ch_02	67-68	26-34	Digital_In 2
RX_ch_03	69-70	35-43	Digital_In 3
RX_ch_04	71-72	44-52	Digital_In 4
RX_ch_05	73-74	53-61	Digital_In 5
RX_ch_06	75-76	62-70	Digital_In 6
RX_ch_07	77-78	71-79	Digital_In 7
RX_ch_08	79-80	80-88	Digital_In 8
ADC_2_6Proc	-	89-91	ADC 1-6
Proc_2_6DAC	81-83	-	DAC 1-6
Version_Info	84	92	-

Figure 111: Mapping table for: std_sent_16tx_8rx_7K325_V01.ini

After all registers have been mapped correctly, constants have been connected to the interface out block the model for a SENT sensor simulation completed right now and able to build.

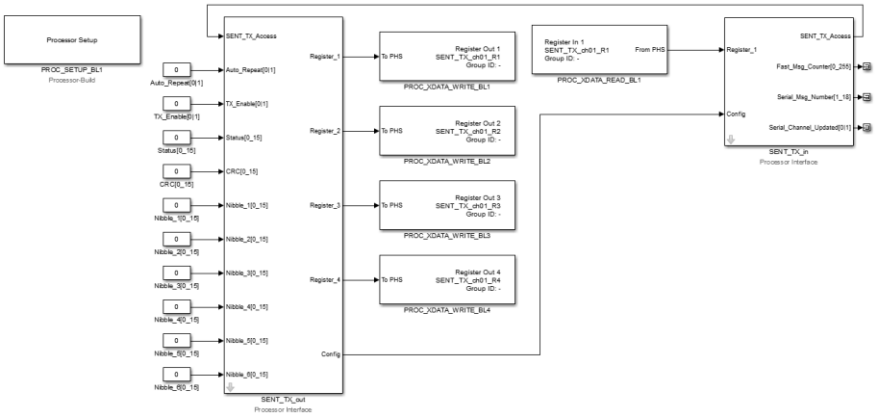


Figure 112: Simulink model containing 1 SENT sensor

	Instead of following all these steps manually, you can also copy the desired SENT communication channel into your target model