

Board and Systems

Model-Based Design Guide

May 2016

Revision history

Revision	Date	Comments
0.1	January 2011	First edition.
1.1	May 2011	Update for release 5.2.0 of software tools
1.2	July 2011	Update for release 5.3.0 of software tools.
1.3	October 2011	Update for release 5.4.0 of software tools.
1.4	September 2012	Update of the page layout
2.0	April 2013	New software release. Corrected for AXI bus.
2.1	September 2013	Added ADAC250, ADC5000, MI125, MI250, Record and Playback host blocks
2.2	June 2014	Corrected the numbering of the figures. Added missing example descriptions (Mestor, Aurora, ADC5000, Radio420 Streaming). Added ADAC250 streaming example description. Added Aurora and Mestor blocks in miscellaneous library. Release 6.5
2.3	November 2014	Added MO1000 MDBK support Release 6.6
2.4	March 2015	Added RTDEx Sync Support Release 6.6.1
3.0	July 2015	New MBDK architecture Release 7
3.1	October 2015	New glossary
3.2	May 2016	Added Radio640, GPIO and Channel Synchronizer blocks

© Nutaq All rights reserved.

No part of this document may be reproduced or used in any form or by any means—graphical, electronic, or mechanical (which includes photocopying, recording, taping, and information storage/retrieval systems)—without the express written permission of Nutaq.

To ensure the accuracy of the information contained herein, particular attention was given to usage in preparing this document. It corresponds to the product version manufactured prior to the date appearing on the title page. There may be differences between the document and the product, if the product was modified after the production of the document.

Nutaq reserves itself the right to make changes and improvements to the product described in this document at any time and without notice.

Trademarks

Acrobat, Adobe, and Reader are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States and/or other countries. IBM is a registered trademark of International Business Machines Corporation in the United States, other countries, or both. Intel and Pentium are registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries. Microsoft, MS-DOS, Windows, Windows NT, and the Windows logo are either registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries. MATLAB, Simulink, and Real-Time Workshop are registered trademarks of The MathWorks, Inc. Xilinx, Spartan, and Virtex are registered trademarks of Xilinx, Inc. Texas Instruments, Code Composer Studio, C62x, C64x, and C67x are trademarks of Texas Instruments Incorporated. All other product names are trademarks or registered trademarks of their respective holders.

The ™ and ® marks have been omitted from the text.

WARNING

Do not use Nutaq products in conjunction with life-monitoring or life-critical equipment. Failure to observe this warning relieves Nutaq of any and all responsibility.

FCC WARNING

This equipment is intended for use in a controlled environment only. It generates, uses, and can radiate radio frequency energy and has not been tested for compliance with the limits of personal computers and peripherals pursuant to subpart J of part 15 of the FCC rules. These rules are designed to provide reasonable protection against radio frequency interference. Operating this equipment in other environments may cause interference with radio communications, in which case the user must, at his/her expense, take whatever measures are required to correct this interference.

This page was left intentionally blank.

Table of Contents

- 1 Introduction1**
 - 1.1 Conventions1
 - 1.2 Block Colors.....1
 - 1.3 Glossary3
 - 1.4 Technical Support4

- 2 Model-Based Design Overview5**
 - 2.1 Model-Based Design Approach.....5
 - 2.2 Model-Based Design Kit6
 - 2.3 MBDK FPGA Overview7
 - 2.3.1 FPGA Code Generation7

- 3 Targeting the FPGA.....9**
 - 3.1 MBDK FPGA Libraries and Blocks9
 - 3.1.1 Onboard and Miscellaneous Tools Libraries Organization10
 - 3.1.2 Add-On Hardware Modules Library Organization12
 - 3.1.3 Accessing Block Help12
 - 3.2 Targeting the FPGA with Simulink13
 - 3.2.1 MBDK FPGA Model Structure13
 - 3.2.2 Targeting the FPGA15

- 4 Hardware Functional Examples18**
 - 4.1 Locating the Example Models18

List of Figures and Tables

Figure 2-1	MBDK code generation flows and how it encapsulates BSDK software	6
Figure 2-2	Conceptual approach to the MBDK FPGA solution	7
Figure 3-1	Nutaq MBDK FPGA blockset	9
Figure 3-2	Onboard library	10
Figure 3-3	Miscellaneous tools library	11
Figure 3-4	Add-on hardware modules library	12
Figure 3-5	MBDK FPGA model	13
Figure 3-6	System Generator block dialog box	14
Figure 3-7	ADC5000 record example model	15
Figure 3-8	Compilation target in the System Generator block	16
Figure 3-9	Generation complete message	17
Figure 4-1	Locating example models	18
Table 1	Block background color code	1
Table 2	Block logo color code	2
Table 3	Typical block example	2
Table 4	Example of exceptions	2
Table 5	Glossary	4
Table 6	FPGA I/Os and corresponding MBDK FPGA blocks	15

1 Introduction

Congratulations on the purchase of the MBDK development environment for your system.

This document contains all the information necessary to understand and use MBDK. It should be read carefully before using the card and stored in a handy location for future reference.

1.1 Conventions

In a procedure containing several steps, the operations are numbered (1, 2, 3...). The diamond (♦) is used to indicate a procedure containing only one step, or secondary steps. Lowercase letters (a, b, c...) can also be used to indicate secondary steps in a complex procedure.

The abbreviation NC is used to indicate no connection.

Capitals are used to identify any term marked as is on an instrument, such as the names of connectors, buttons, indicator lights, etc. Capitals are also used to identify key names of the computer keyboard.

All terms used in software, such as the names of menus, commands, dialog boxes, text boxes, and options, are presented in bold font style.

The abbreviation N/A is used to indicate something that is not applicable or not available at the time of press.

Note:

The screen captures in this document are taken from the software version available at the time of press. For this reason, they may differ slightly from what appears on your screen, depending on the software version that you are using. Furthermore, the screen captures may differ from what appears on your screen if you use different appearance settings.

1.2 Block Colors

The blocks presented in this document all abide by a strict color code aimed at helping you indentify them at a glance.

Block background colors

A block background color is indicative of its intended target.



Background color	Block target
	FPGA
	Simulink

Table 1 Block background color code

Block logo colors

All Nutaq blocks contain a Nutaq infinity logo, its color indicative of the type of the block.

Logo color	Block type
------------	------------






Logo color	Block type
	Input/Output
	Control
	Configuration
	Probe
	Miscellaneous

Table 2 Block logo color code

Examples

Typical blocks

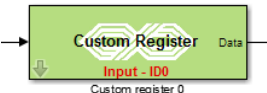

Block	Target	Type
	FPGA	Control
	Simulink	Miscellaneous

Table 3 Typical block example

Exceptions



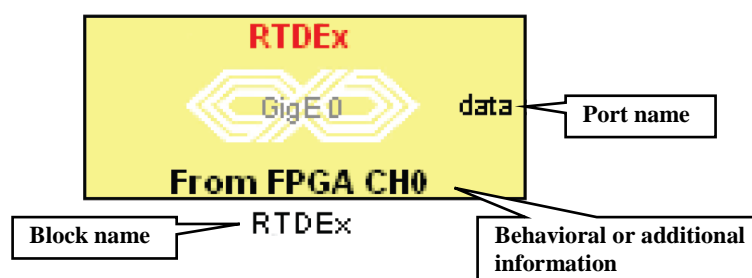
Color	Target	Type	Description
	FPGA	Configuration	The FPGA configuration block use Nutaq logo and the hardware platform name selected is printed at the bottom of icon (ex.: Perseus 601X).
	Simulink	Miscellaneous	All library blocks resemble Simulink blocks, with a red Nutaq infinity logo.

Table 4 Example of exceptions

Block nomenclature

Nutaq blocks are all constructed according to the same infrastructure and nomenclature, as illustrated below. Some of the indications may not appear in certain blocks when they are not relevant.



1.3 Glossary

This section presents a list of terms used throughout this document and their definition.

Term	Definition
Advanced Mezzanine Card (AMC)	AdvancedMC is targeted to requirements for the next generation of "carrier grade" communications equipment. This series of specifications are designed to work on any carrier card (primarily AdvancedTCA) but also to plug into a backplane directly as defined by MicroTCA specification.
Advanced Telecommunications Computing Architecture (or AdvancedTCA, ATCA)	AdvancedTCA is targeted primarily to requirements for "carrier grade" communications equipment, but has recently expanded its reach into more ruggedized applications geared toward the military/aerospace industries as well. This series of specifications incorporates the latest trends in high speed interconnect technologies, next-generation processors, and improved Reliability, Availability and Serviceability (RAS).
Application Programming Interface (API)	An application programming interface is the interface that a computer system, library, or application provides to allow requests for services to be made of it by other computer programs or to allow data to be exchanged between them.
Board Software Development Kit (BSDK)	The board software development kit gives users the possibility to quickly become fully functional developing C/C++ for the host computer and HDL code for the FPGA through an understanding of all Nutaq boards major interfaces.
Boards and Systems (BAS)	Refers to the division part of Nutaq which is responsible for the development and maintenance of the hardware and software products related to the different Perseus carriers and their different FMC daughter cards.
Carrier	Electronic board on which other boards are connected. In the FMC context, the FMC carrier is the board on which FMC connectors allow a connection between an FMC card and an FPGA. Nutaq has two FMC carriers, the Perseus601x (1 FMC site) and the Perseus611x (2 FMC sites).
Central Communication Engine (CCE)	The Central Communication engine (CCE) is an application that executes on a virtual processor called a MicroBlaze in the FPGA of the Perseus products. It handles all the behavior of the Perseus such as module initialization, clock management, as well as other tasks.
Chassis	Refers to the rigid framework onto which the CPU board, Nutaq development platforms, and other equipment are mounted. It also supports the shell-like case—the housing that protects all the vital internal equipment from dust, moisture, and tampering.
Command Line Interface (CLI)	The Command Line Interface (or CLI) is a basic client interface for Nutaq's FMC carriers. It runs on a host device. It consists of a shell where commands can be typed, interacting with the different computing elements connected to the system.
FPGA Mezzanine Card (FMC)	FPGA Mezzanine Card is an ANSI/VITA standard that defines I/O mezzanine modules with connection to an FPGA or other device with re-configurable I/O capability. It specifies a low profile connector and compact board size for compatibility with several industry standard slot card, blade, low profile motherboard, and mezzanine form factors.
HDL	Stands for hardware description language.
Host	A host is defined as the device that configures and controls a Nutaq board. The host may be a standard computer or an embedded CPU board in the same chassis system where the Nutaq board is installed. You can develop applications on the host for Nutaq boards through the use of an application programming interface (API) that comprises protocols and functions necessary to

Term	Definition
	build software applications. These API are supplied with the Nutaq board.
MicroTCA (or μ TCA)	The MicroTCA (μ TCA) specification is a PICMG Standard which has been devised to provide the requirements for a platform for telecommunications equipment. It has been created for AMC cards.
Model-Based Design	Refers to all the Nutaq board-specific tools and software used for development with the boards in MATLAB and Simulink and the Nutaq model-based design kits.
Model-Based Development Kit (MBDK)	The model-based development kit gives users the possibility to create FPGA configuration files, or bitstreams, without the need to be fluent in VHDL. By combining Simulink from Matlab, System Generator from Xilinx and Nutaq's tools, someone can quickly create fully-functional FPGA bitstreams for the Perseus platforms.
NTP	Network Time Protocol. NTP is a protocol to synchronize the computer time over a network.
Peer	A host peer is an associated host running RTDEx on either Linux or Windows. An FPGA peer is an associated FPGA device.
PicoDigitizer / PicoSDR Systems	Refers to Nutaq products composed of Perseus AMCs and digitizer or SDR FMCs in a table top format.
PPS	Pulse per second. Event to indicate the start of a new second.
Reception (Rx)	Any data received by the referent is a reception.
Reference Design	Blueprint of an FPGA system implemented on Nutaq boards. It is intended for others to copy and contains the essential elements of a working system (in other words, it is capable of data processing), but third parties may enhance or modify the design as necessary.
Transmission (Tx)	Any data transmitted by the referent is a transmission. Abbreviated TX.
μ Digitizer / μ SDR Systems	Any Nutaq system composed of a combination of μ TCA or ATCA chassis, Perseus AMCs and digitizer or SDR FMCs.
VHDL	Stands for VHSIC hardware description language.

Table 5 Glossary

1.4 Technical Support

Nutaq is firmly committed to providing the highest level of customer service and product support. If you experience any difficulties using our products or if it fails to operate as described, first refer to the documentation accompanying the product. If you find yourself still in need of assistance, visit the technical support page in the Support section of our Web site at www.nutaq.com.

2 Model-Based Design Overview

This chapter offers you an overview of model-based design with the Nutaq model-based design kit (or MBDK) in the MATLAB/Simulink environment. Model-based design, as you may already know, allows you to deploy and validate algorithms on actual hardware more rapidly and efficiently than through a traditional language-based approach.

2.1 Model-Based Design Approach

In a traditional design flow featuring embedded signal processing components, most errors arise from the development phase. Unfortunately, most of these errors are only detected later, during the design of system components. Even more unfortunate is that detecting and fixing design error at the product development stage is very expensive as it forces sometimes major portions of system to be reengineered—in extreme cases, it can even lead to products becoming outdated before they even reach the market!

To overcome this, more and more developers are making the move towards more open and flexible model-based design environments that encompass all the major stages of embedded signal processing development—design, simulation, code generation, validation, and implementation—which is proving more logical and economical.

Most companies and individuals designing systems this way keep their intellectual property (IP) in the form of C code or MATLAB code, and most model-based design environments allow developers to easily incorporate and reuse this (IP).

Also, by performing detailed simulations of a new system's behavior under various conditions—the bedrock of embedded system development—developers are able to identify, pinpoint, and resolve design problems very quickly because it is easier to differentiate design problems from programming problems. By working at the system level with model-based design tools, developers and designers are certain that problems detected at the development stage are design flaws instead of programming flaws.

The following sections delve deeper into the mechanics used by Nutaq to integrate its platforms to the model-based design environment MATLAB/Simulink.

2.2 Model-Based Design Kit

The following figure illustrates the FPGA code generation flows of the MBDK.

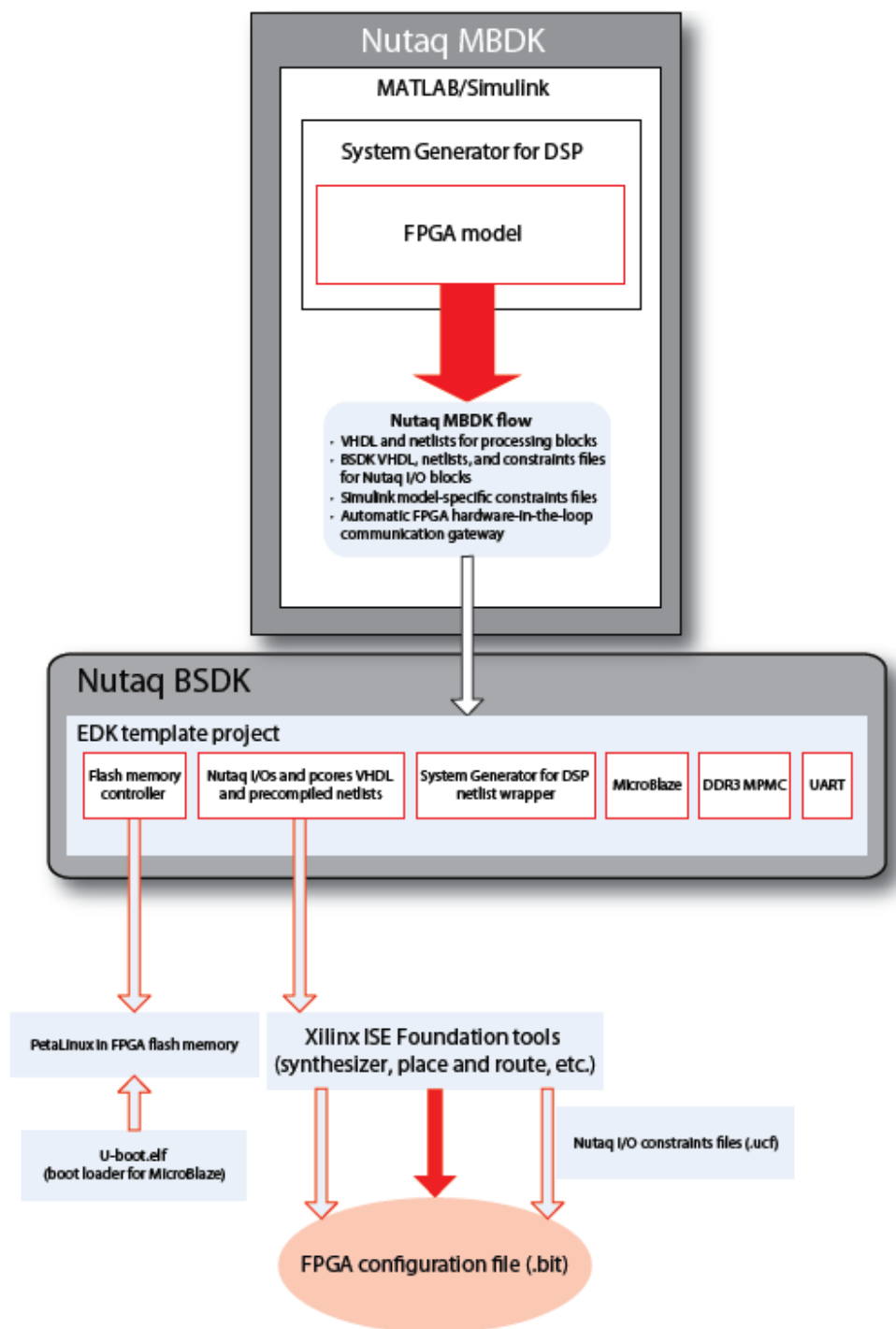


Figure 2-1 MBDK code generation flows and how it encapsulates BSDK software

2.3 MBDK FPGA Overview

This section outlines the basics of FPGA code generation. For more in depth explanations, see following chapters.

2.3.1 FPGA Code Generation

The MBDK allows you to use MATLAB, Simulink, and System Generator for DSP to generate an FPGA configuration file (.bit) for the FPGA in your system within a Simulink model. The MBDK FPGA flow uses a Nutaq EDK template project as top entity that contains the platform I/Os and peripheral cores that are modified according to the contents of the System Generator for DSP user design. The EDK project also includes a black box that holds the user model, also called sysgen_hw_cosim interface, used during the FPGA configuration file generation.

A model can only describe the behavior of one FPGA. If two FPGA require different behavior at build time, one model per FPGA needs to be created.

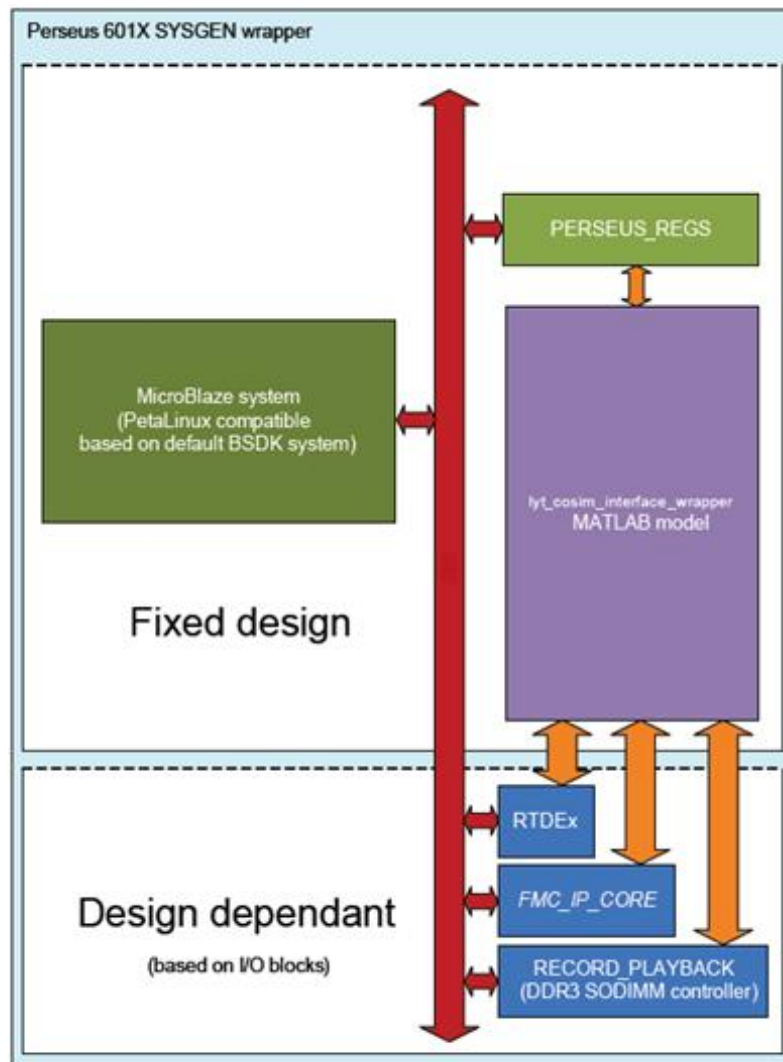


Figure 2-2 Conceptual approach to the MBDK FPGA solution

System Generator for DSP EDK project

The System Generator for DSP EDK project contains the platform I/O and peripheral cores. Some of these cores (green, above) are always present in projects while others are added (blue, above). They are incorporated in the

EDK project when they are present in your System Generator for DSP model. This EDK project is thus a template that is modified by the MBDK flow.

Onboard operating system

The EDK project is designed to run an operating system (PetaLinux) and the central communication engine in its onboard flash memory. For more details, refer to your platform User's Guide.

SYSGEN wrapper

FPGA interfaces communicate with your Simulink model through I/O blocks. The Nutaq I/O blocks allow logic to communicate with the SYSGEN wrapper (yellow, above).

The EDK project also includes a black box core that contains the SYSGEN wrapper files generated by System Generator for DSP's xflow. This is what corresponds to your System Generator for DSP model (purple, above).

The specific functionalities contained in the wrapper depend on the Nutaq I/O blocks used in your design.

Anatomy of a Nutaq block for Simulink

All Nutaq MBDK FPGA blocks are based on System Generator for DSP blocks from Xilinx, such as the Gateway In, Gateway Out, Register, and Convert blocks. In a model, FPGA blocks provide a transformation path between the Simulink blocks and the System Generator for DSP blocks that compose your model. When the model is compiled as an FPGA bitstream, FPGA blocks allow you to access the FPGA peripherals and resources of your platform.

Clocks and sample periods

All the System Generator for DSP blocks that you include in your model run at the design clock's rate or at a slower rate, using a clock enable. There are several possible sources for the design clock such as a reference clock (200 MHz), a AXI clock (100 MHz) and FMC's clocks.

The sample period that you can configure in FPGA blocks gives an indication of a block's relative sample period relative to other blocks, while the FPGA Clock Period specified in the System Generator block is the timing constraint of your platform's clock for Xilinx tools.

For details about FPGA code generation, see Targeting the FPGA.

3 Targeting the FPGA

As explained in the previous chapter, the MBDK is responsible for generating code for Nutaq development platform equipped with FPGAs. More accurately, it acts as a Xilinx System Generator for DSP add-in, making it possible to generate code that can be compiled.

This chapter introduces the FPGA block set and explains in detail how to use Simulink to generate code for the your platform FPGA.

3.1 MBDK FPGA Libraries and Blocks

The FPGA libraries are separated into three subcategories: the *Onboard* library, the *Add-on hardware modules* library, and the *Miscellaneous tools* library.

The Onboard library regroups blocks that relate directly to the carrier board. The Add-on hardware modules library contains blocks that relate to optional FMCs for the Perseus. The Miscellaneous tools library contains blocks used to perform general tasks like I/O interfacing.

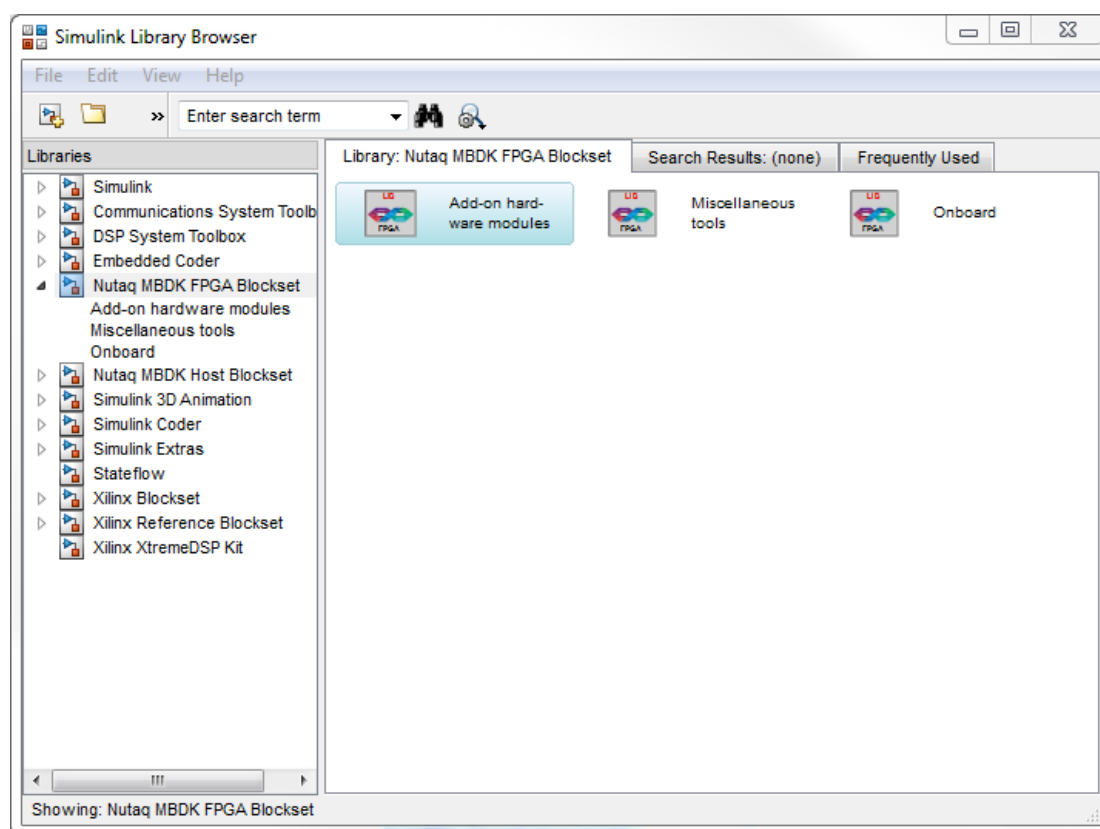


Figure 3-1 Nutaq MBDK FPGA blockset

3.1.1 Onboard Library

The Perseus MBDK FPGA Onboard library contains the following blocks.



Figure 3-2 Onboard library

Onboard library blocks

The Onboard library is composed of the following blocks:

- Custom register
- MBDK FPGA Configuration
- Record/Playback
- RTDEx
- RTDEx Configuration

3.1.2 Miscellaneous Tools Library

The Perseus MBDK FPGA Miscellaneous Tools library contains the following blocks.

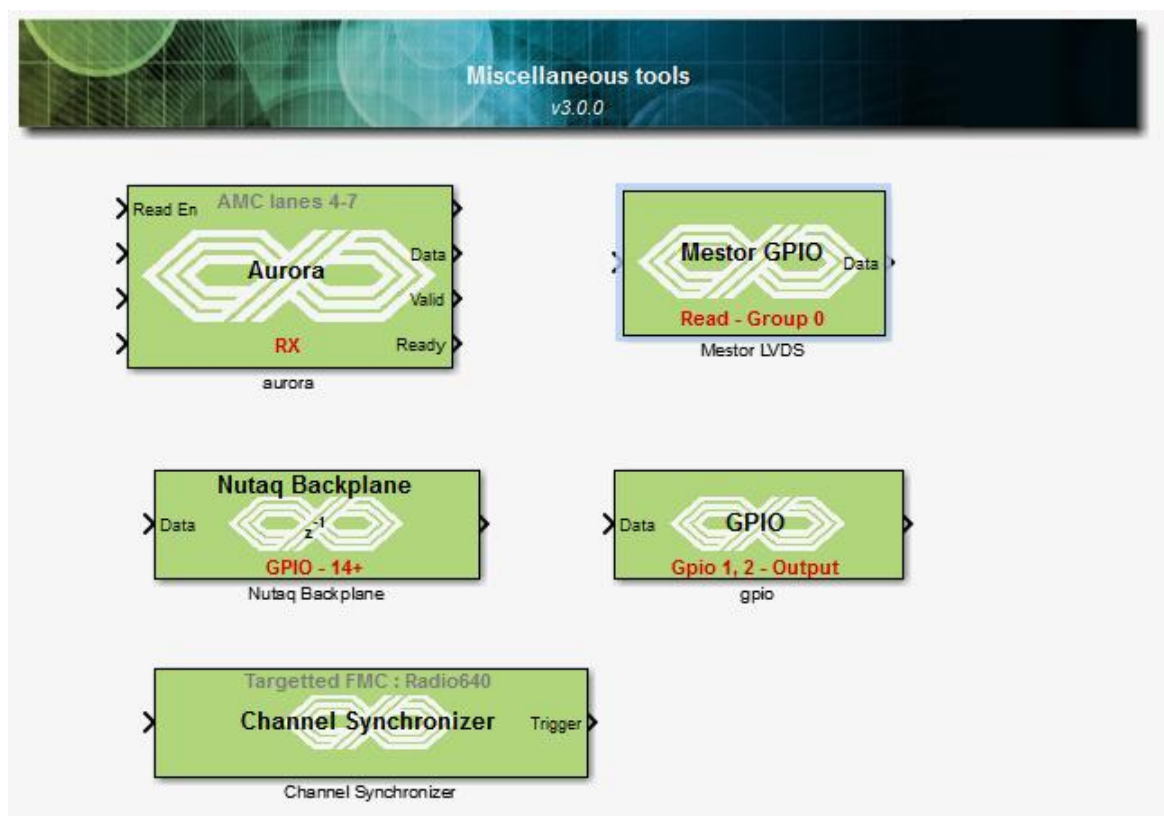


Figure 3-3 Miscellaneous tools library

Miscellaneous tools library blocks

The miscellaneous library is composed of the following blocks:

- Mestor LVDS
- Aurora
- Nutaq Backplane
- Channel synchronizer
- GPIO

3.1.3 Add-On Hardware Modules Library Organization

The carrier board can be equipped with a variety of add-on FMCs such as the ADAC250 and the Radio420X. To communicate with these modules, Nutaq supplies MBDK blocks.

Note:

Even if the blocks are present in your Simulink libraries, FMC must be installed on the Perseus before you can use the blocks in your models.

The FPGA Add-on hardware modules library contains the following blocks.

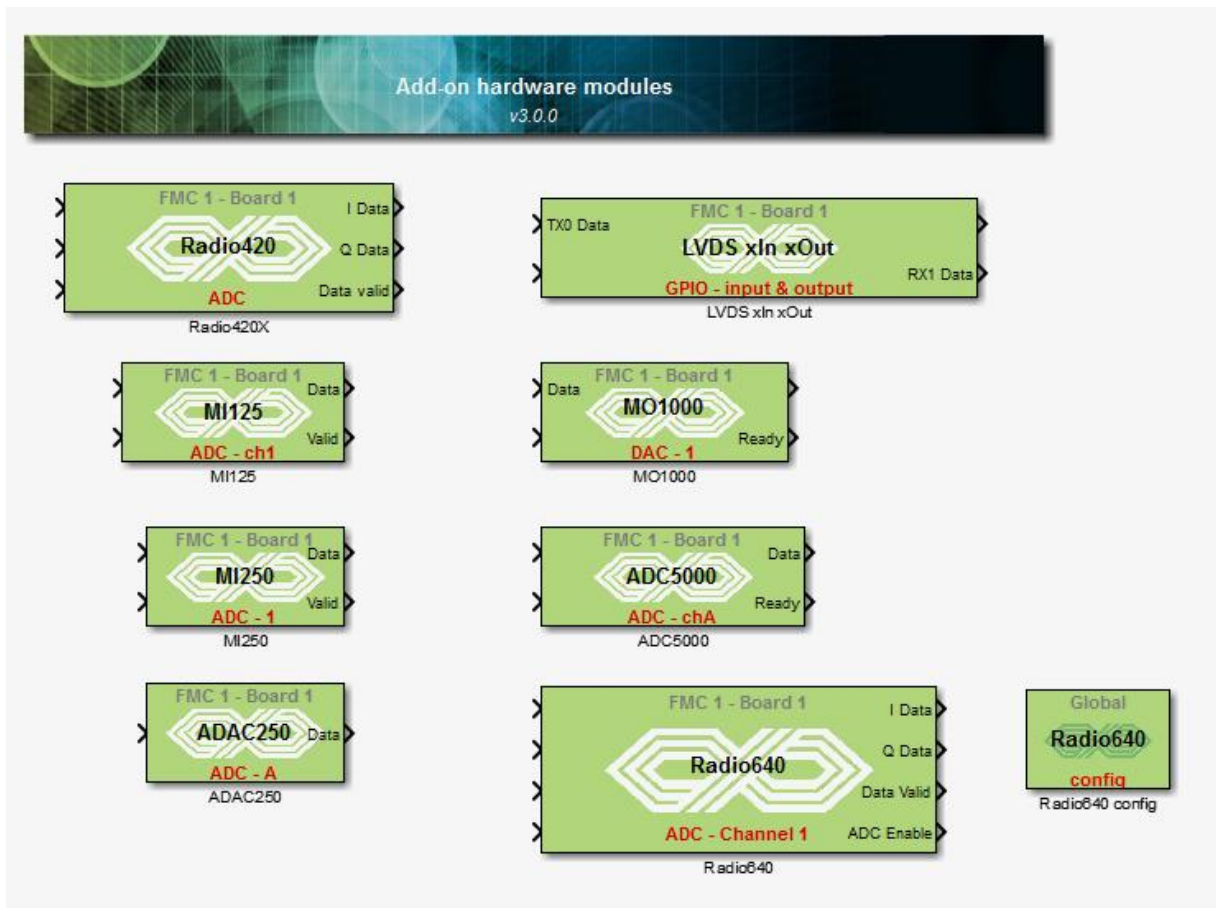


Figure 3-4 Add-on hardware modules library

Add-on hardware modules library blocks

The Add-on hardware modules library is composed of the ADAC250, ADC5000, Radio420X, MI250, MI125, MO1000, LVDS xIn xOut and Radio640 blocks. Other blocks will be added in future releases as other FMCs become available.

3.1.4 Accessing Block Help

To access block help:

1. Open the Nutaq block about which you want information.
2. Click **Help**.

The appropriate Help appears in a separate window, detailing the block.

3.2 Targeting the FPGA with Simulink

3.2.1 MBDK FPGA Model Structure

From Simulink point of view, a Nutaq MBDK FPGA model is exactly the same as any other System Generator for DSP model. The only difference is that MBDK FPGA models use an FPGA board configuration block to instruct Simulink to use the MBDK FPGA flow when compiling.

The figure below illustrates an MBDK FPGA model. Such models have two main sections—the configuration section (highlighted, below) and the rest of the model, the processing section.

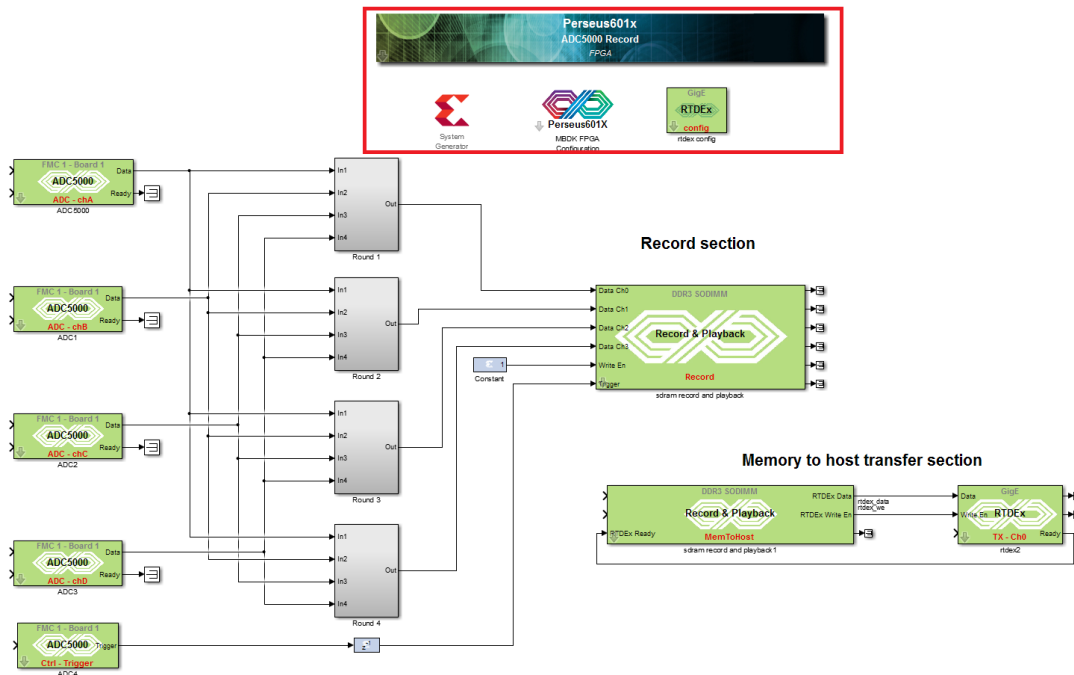


Figure 3-5 MBDK FPGA model

Configuration section

The configuration in an MBDK FPGA model is very straightforward—there are no special scripts to run or configurations to change. The only requirement is that the model contains the System Generator block from Xilinx and the MBDK FPGA Configuration block from Nutaq. The System Generator block instructs Simulink to use the Xilinx System Generator for DSP compilation flow, while the MBDK FPGA Configuration block allows compilation parameter modifications.

Processing section

The processing section of an FPGA model is where users place blocks to design their algorithm. This section is composed of Xilinx blocks, Nutaq I/O blocks, and Simulink blocks (white, for simulation only).

FPGAs process data as fixed-point data, but Simulink uses a double data type. This is why a Gateway out block is used to convert the Xilinx data type to the Simulink data type to display data in Simulink scope. The same Gateway out/Gateway in are present in Nutaq I/O blocks such as the RTDEx block. Nutaq I/O ports that do not have names are the Simulink (simulation) version of the System Generator (FPGA implementation) ports at the opposite side of the block.

System Generator block

As mentioned earlier, the System Generator block must be present in your FPGA model to use it. The following figure illustrates the main parameters that must be configured in the System Generator block.

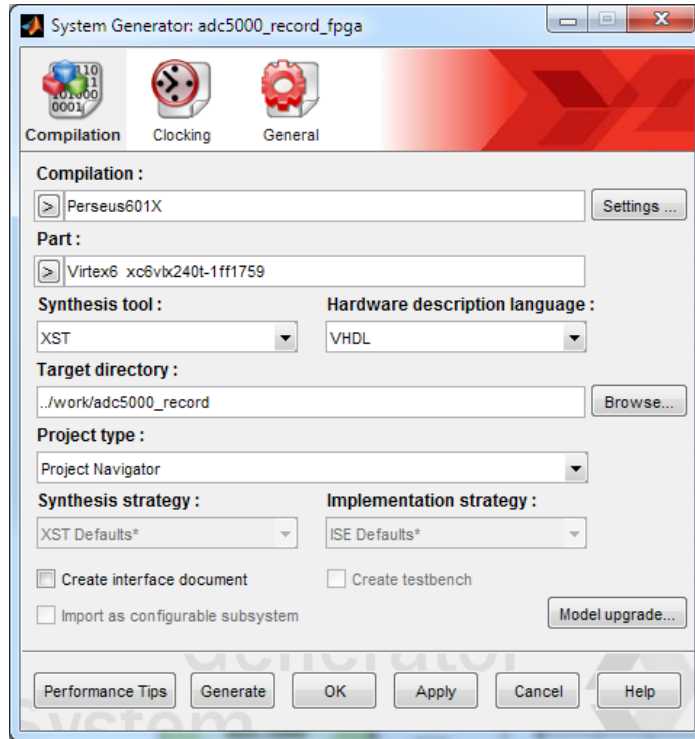


Figure 3-6 System Generator block dialog box

Compilation list

Specify the targeted system on this list. This instructs Simulink what design flow it must use. For the Nutaq's carrier board, point to Hardware Co-simulation, Nutaq, and select your carrier board.

Part list

This list allows you to select the type of FPGA installed on the carrier board. The available choices depend on the select carrier board. A specific FPGA exist in various speed grades. Make sure to select the speed grade that match your hardware (speed grade "-1" by default).

Target directory text box

Specify the folder where the compiler writes its temporary files when it performs the build operation in this text box. The complete compilation path must not contain space.

Synthesis tool and Hardware description language lists

At this time, MBDK only supports XST with VHDL.

FPGA clock period and Simulink system period text boxes

These parameters are necessary to inform System Generator for DSP of the timing constraints of the FPGA design. More specifically, these parameters inform System Generator for DSP at what frequency each block in the FPGA model is running. System Generator for DSP uses this information during the place and route phase of the bitstream generation process (performed by ISE Foundation).

- **FPGA clock period:** Informs System Generator for DSP of the time period that corresponds to the real-time clock driving your design in nanoseconds. For example, if you use an ADAC250, the period is configured to your sampling frequency. If in your design it is possible that the ADAC250 can run at 100 MHz and at 250 MHz, it is important to specify the worst case constraint. In this example, the worst case is 250 MHz and a value of 4 ns must be specified.
- **Simulink system period:** This parameter informs System Generator for DSP of the sample period related to the real-time clock value. It also relates each Xilinx and Nutaq FPGA blocks' sample periods to the real-time clock. For example, if the Simulink sample period is 1, then all the blocks with a Sample period of 1 run at the real-time clock value (200 MHz if the Reference clock is selected). Blocks with a Sample period of 2 run at half the rate of the real-time clock value (100 MHz if the Reference clock is selected).

Note:

Refer to System Generator for DSP documentation for details about this block.

MBDK FPGA Configuration block

The MBDK FPGA Configuration block specifies which carrier board is used and which clock is driving the model. In addition, the block specified which lanes are used when PCIe is instantiated.

3.2.2 Targeting the FPGA

Binary configuration files are created to represent the MBDK FPGA design. These FPGA files are referred to as bitstreams and have a .bit file extension.

FPGA real-time design

FPGA real-time design refers to the fact that the FPGA needs a physical clock to drive the data in the FPGA. More precisely, the design is implemented so that the FPGA sends or receives data to or from one of the FPGA peripheral or I/O. As an example, the following are I/O interfaces available to communicate with the FPGA.

I/Os	MBDK FPGA block used
Control and access internal registers from a host application.	Custom register block
Real-time data exchange (RTDEx) between a host computer and the FPGA user logic.	RTDEx block

Table 6 FPGA I/Os and corresponding MBDK FPGA blocks

Opening a model

To open an FPGA model, type its name at the MATLAB command prompt. For demonstration purposes, we used the ADC5000 record example.

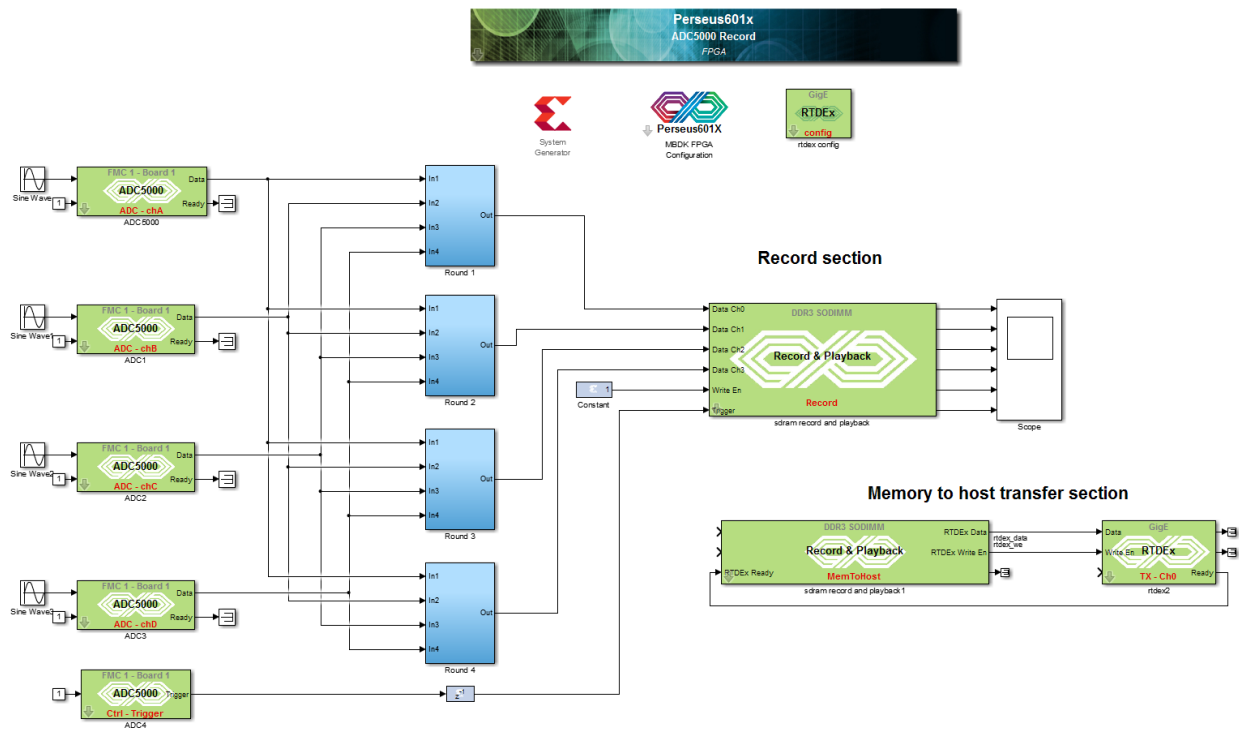


Figure 3-7 ADC5000 record example model

A real-time FPGA Simulink model should always contain the following sections:

- The System Generator block and the MBDK FPGA Configuration block.
- Xilinx processing blocks (light blue ones).
- Simulink simulation blocks (white ones).
- FPGA I/O and configuration blocks (light green ones), which have two purposes:
 - When you simulate the FPGA model, the blocks interface Xilinx processing blocks with Simulink I/O blocks. This is necessary because Simulink and Xilinx processing blocks do not use the same data format. Simulink blocks mainly use floating-point precision data types, while System Generator for DSP blocks use fixed-point data format. In the example above, the **ADC5000** block route data from the Simulink sine wave source to the FPGA user logic. The **Record** block passes the processed data in the FPGA environment to the Simulink environment for data display.

Note:

The Nutaq MBDK FPGA I/O blocks perform this conversion by encapsulating the Xilinx **Gateway In** and **Gateway Out** blocks, in the Xilinx blockset.

- When building a bitstream, the blocks interface the Xilinx processing blocks with the platform hardware. In the example above, the **ADC5000** blocks route the data from the A/D converters to the FPGA user logic described inside the System Generator model. The **Record** block route the data from the user logic described inside the System Generator model to the instantiated memory controller core.

Configuring a model

Before you can target the platform, make sure that the Nutaq's MBDK add-in is installed in System Generator for DSP. Refer to the software installation guide for details.

When the add-in is successfully installed, if you open a provided MBDK mode, you can see a Nutaq's carrier board appears as the compilation target in the System Generator block, as shown below.

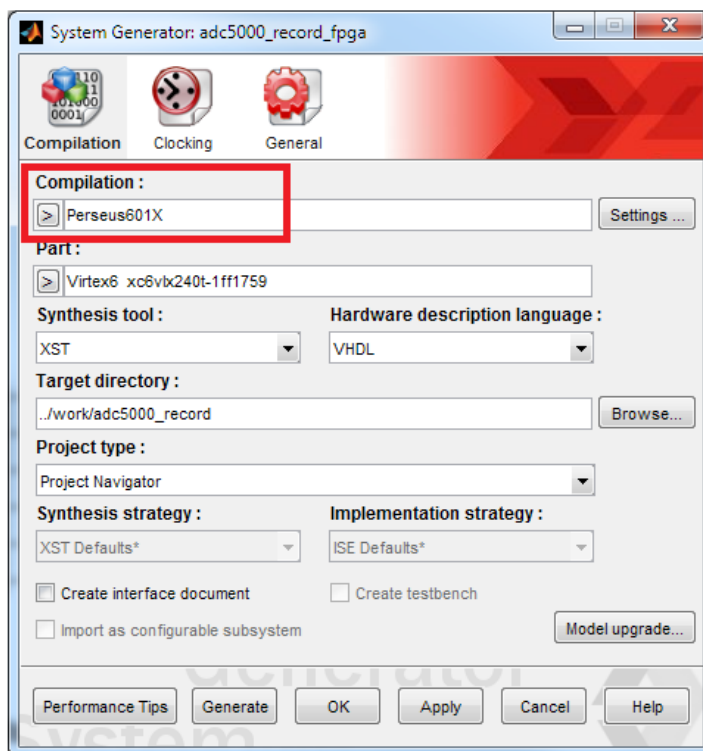


Figure 3-8 Compilation target in the System Generator block

Building a model

Before you can build an FPGA model, you must provide System Generator for DSP with the information about how to build the FPGA model.

To provide information about the FPGA model:

1. Select the hardware clock used by the System Generator for DSP design on the **Clock Source** list of the MBDK **FPGA Configuration** block.

2. Instruct the **System Generator** block in how to build the FPGA model:

On the **Compilation** list, point to **Hardware Co-Simulation**, **Nutaq** and select your carrier board.

On the **Part** list, select the type of FPGA that match your hardware configuration.

In the **Target directory** text box, specify the folder where the generated FPGA code must be created.

This folder will also contain the bitstream corresponding to your model, when the building process is complete. The name of the bitstream is *name_of_fpga_model.bit*, where the variable *name_of_fpga_model* corresponds to the name of your FPGA model.

On the **Synthesis tools** list, select **XST**.

On the **Hardware description language** list, select **VHDL**.

In the **FPGA clock period** and **Simulink system period** text boxes, specify valid values.

3. Instruct the **System Generator** block to build the bitstream by clicking **Generate**. The code generation process starts for each block in the FPGA model (except for Simulink I/O blocks, only present for simulation purposes). Once the code is generated, the **System Generator** block starts a Perl session and invokes ISE Foundation tools to create a bitstream from the model generated code. When the building process is complete, the following message box appears.

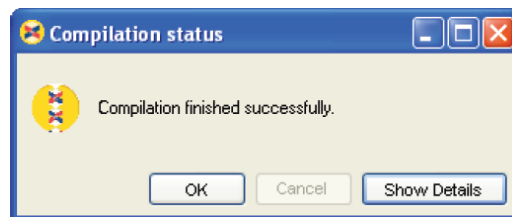


Figure 3-9 Generation complete message

Running the model

Running the model is different according to the interface used in the model. See the different possibilities by trying Nutaq examples.

4 Hardware Functional Examples

The Perseus FMC carriers come with a series of hardware functional examples that you can access as described below. They describe how to configure and communicate with the various hardware and software modules of the Perseus. Hardware functional examples are designed to help you use the FPGA interfaces. All the hardware functional examples use FPGA Simulink models.

4.1 Locating the Example Models

To locate the example:

1. At the MATLAB command prompt, type **doc -classic**.
MATLAB Help starts.
2. Expand **Examples**.
3. Expand **Nutaq**.

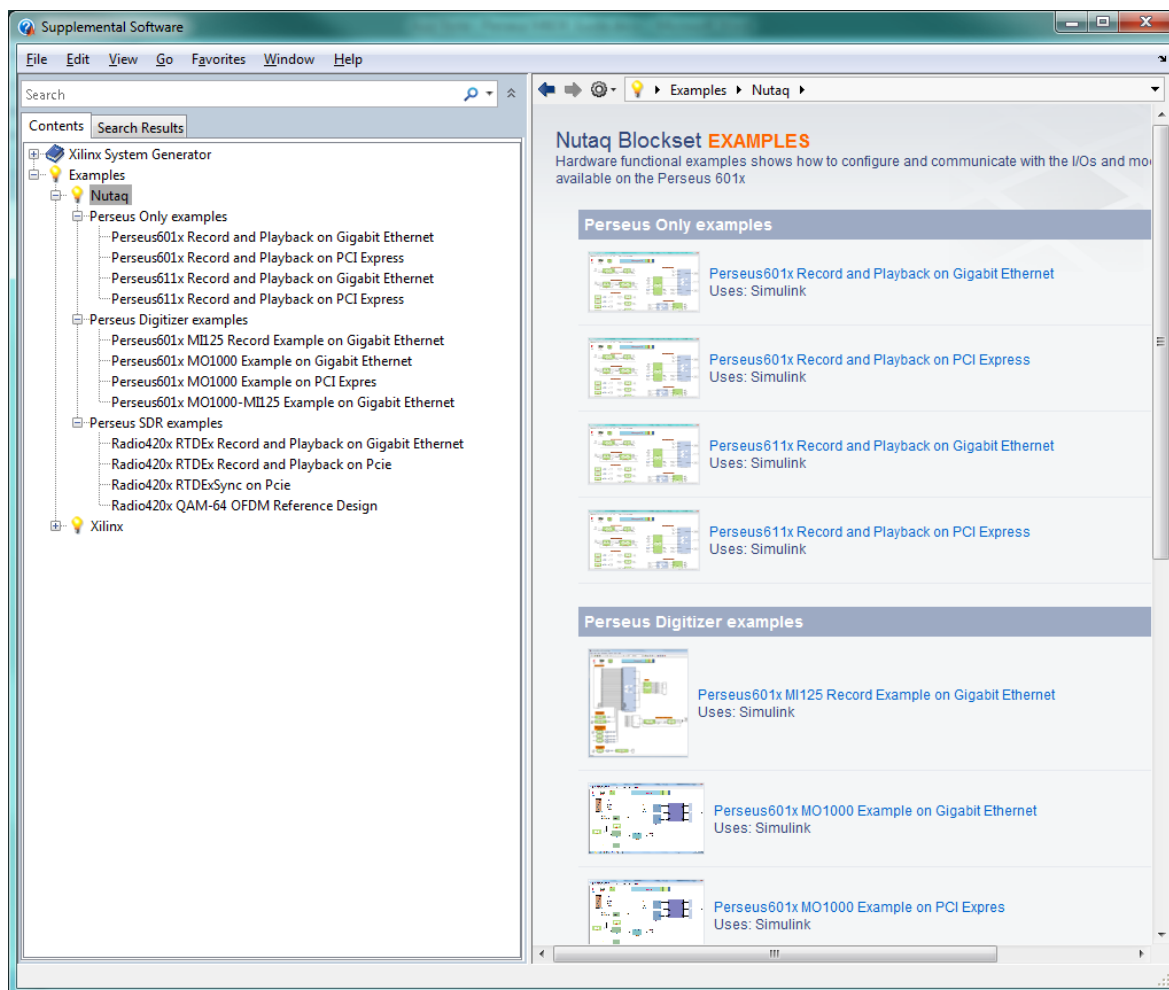


Figure 4-1 Locating example models

We strongly recommend that you have a basic knowledge of Simulink and System Generator for DSP before you continue. To avoid overwriting original files, we also recommend that you copy the hardware functional example models to a working folder before attempting to use them.