Configuring Custom HDL Models using Simulink

Prerequisites:

Recommended versions: Vivado 2021.1 – Matlab R2021B_U4

Recommended terminal for Windows: Cygwin

Make sure that the Vitis 2021.1 is installed.

The latest branch:

https://github.com/analogdevicesinc/HighSpeedConverterToolbox/tree/cn0585_v1

1. Instructions to build the toolbox from terminal:

>git clone <u>https://github.com/analogdevicesinc/HighSpeedConverterToolbox.git</u>

> cd HighSpeedConverterToolbox

../HighSpeedConverterToolbox> git submodule update --init --recursive

../HighSpeedConverterToolbox > git checkout cn0585_v1

To avoid tool mismatches, before opening MATLAB set this variable in the terminal:

../HighSpeedConverterToolbox> export ADI_IGNORE_VERSION_CHECK=TRUE

../HighSpeedConverterToolbox > cd CI/scripts

../HighSpeedConverterToolbox/CI/scripts > make build HDLBRANCH=cn0585_v1

This will also make a clone of the HDL repo and checkout the desired branch.

In Matlab current folder list select navigate to the folder where the files had been copied from previous step.

Launch MATLAB in the root of the HighSpeedConverterToolbox folder:

../HighSpeedConverterToolbox/CI/scripts > cd ../../

../HighSpeedConverterToolbox > matlab .



Figure 1. High Speed Converter Toolbox Sources

right click on test -> Add to Path -> Selected folders and subfolders

right click on hdl -> Add to Path -> Selected folders and subfolders

In the Matlab command window set the path to Vivaldo installation folder. The tool path should be replaced with the user's Vivado path.

e.g. hdlsetuptoolpath('ToolName', 'Xilinx Vivado', 'ToolPath', '</opt/Xilinx/Vivado/2021.1/bin/vivado>')

Expand the test folder and double click on the desired Simulink test model, as shown in Figure 2.

Current Folder	Current Folder						
🗋 Name 🔨	Git						
README.md		-					
LICENSE							
🗋 JenkinsfileHW							
JenkinsfileCron							
🗋 Jenkinsfile							
🗋 info.xml							
🗉 🧰 test							
🔁 testModel_Tx16and8.slx		4.40					
慉 testModel_Rx64Tx64.slx	•	10000					
🔁 testModel_bitswitch.slx	•	182					
🔁 testModel.slx	۲						

Figure 2. Simulink Test Model

After opening the Simulink model, right click on the HDL_DUT and launch the HDL Workflow Advisor as shown in Figure 3, and Figure 4.



Figure 3. HDL Workflow Advisor Launching



Figure 4. Simulink Device Under Test

Close this expected warning that will appear, as shown in Figure 5.



Figure 5. Expected HDL Workflow Advisor Warning

1.1 Select IP Core Generation, choose the desired project and carrier from the dropdown list and check the Allow unsupported version box. Change the project folder name if desired. Finally press the Run this Task button.

	HDL Workflow Advisor - testModel_bitswitch/HDL_DUT	r –	- 0	×
File Edit Run Help				
Find: 🔹 🗢 💠				
B HDL Workflow Advisor B 1. Set Target	1.1. Set Target Device and Synthesis Tool Analysis (^Triggers Update Diagram)			
 ^ 1.1. Set Target Device and Synthesis Tool 1.2. Set Target Reference Design 1.3. Set Target Interface 	Set Target Device and Synthesis Tool for HDL code generation Input Parameters			
a 2. Prepare Model For HDL Code Generation a 3. HDL Code Generation	Target workflow: IP Core Generation		*	
 	Target platform: AnalogDevices LLDK ZED	✓ Launch Board Man		
	Synthesis tool: Xilinx Vivado - Tool version:	2021.1 C Allow unsupported version Re	fresh	
	Family: Zynq -	Device: xc7z020		
	Package: clg484	Speed: -1		
	Project folder: hdl_prj	Bro	wse	
	Run This Task			
	Result: 🥝 Passed			
	Warning Version 2021.1 of tool Xilinx Vivado is not supported in HDL Workflow A further tasks. Attempting to continue using this tool version.	Advisor. Using incompatible tool version, HDL Workflow Advisor may not be	able to run	
	Warning Embedded Coder Support Package for Xilinx Zynq Platform is not installe recommended to complete the installation before proceeding.	ed, which may be required to run task(s): Generate Simulink software interfac	e model It is	5
	Passed Set Target Device and Synthesis Tool.			
¢b		Hel	р Др	• ply

Figure 6. Set Target Device and Synthesis Tool

1.2 Choose the RX, TX, or RX&TX configuration, then run task:



Figure 7. Set Target Reference Design

The HDL DUT in Rx &Tx configuration will pe placed between the Tx and the Rx path, as shown in Figure 8.



Figure 8. HDL Block Design After Inserting the Simulink Block in Rx & Tx Configuration

The HDL DUT in Tx configuration will pe placed between the Tx and the Rx path, as shown in Figure 9.



Figure 9. HDL Block Design After Inserting the Simulink Block in Tx Configuration

The HDL DUT in Rx configuration will pe placed between the Tx and the Rx path, as shown in Figure 10.



Figure 10. HDL Block Design After Inserting the Simulink Block in Rx Configuration

1.3 At this point you need to assign every port, as shown in Figure 11 and Figure 12, and run the task.

nd:	9P						
A		1.3. Set Target Inter	face				
 HDL Workflow Advisor I. Set Target 	Device and Synthesis T Reference Design	Analysis (^Triggers Up Set target interface for Input Parameters Processor/FPGA synch	idate Diagram) HDL code gene ironization: Fre	eration ee running			
 A 1.3. Set Target I 2. Prepare Model For 3. UPL Code Convert 	nterface HDL Code Generation	Enable HDL DUT of Target platform interf	utput port gen ace table	eration for te	st points		
> Colle Generat	Integration	Port Name	Port Type	Data Type	Target Platform Interfaces	Interface Mapping	Interface Options
· · · · · · · · · · · · · · · · · · ·	mugrocom	in1	Inport	int16	IP Data 0 IN [0:15]	- [0:15]	
	in2	Inport	int16	IP Data 1 IN [0:15]	- [0:15]		
	in3	Inport	int16	IP Data 2 IN [0:15]	* [0:15]		
		in4	Inport	int16	IP Data 3 IN [0:15]	- [0:15]	
		in5	Inport	int16	LLDK ADC Data 0 IN [0:15]	* [0:15]	
		in6	Inport	int16	LLDK ADC Data 1 IN [0:15]	* [0:15]	
		in7	Inport	int16	LLDK ADC Data 2 IN [0:15]	- [0:15]	
		in8	Inport	int16	LLDK ADC Data 3 IN [0:15]	- [0:15]	
	validIn1	Inport	boolean	IP Valid Rx Data IN	• [0]		
	validIn2	Inport	boolean	IP Valid Tx Data IN	• [0]		
	in9	Inport	int16	No Interface Specified			
	in10	Inport	int16	No Interface Specified			
			41.515				

Figure 11. Set Target Interface Input Ports (Rx & Tx Reference Design)

The LLDK ADC DATA $\langle x \rangle$ IN is the data in offset binary format captured by the ADC interface IP. IP sends the data at a variable sample rate (default is 15MHz but can be changed using the IIO Oscilloscope/ Python) along with the validIn $\langle x \rangle$ signal which has the logic value 1 for a clock period (8.33ns) when the data has changed.

nd: 🗸 🖓 🖓						
HDL Workflow Advisor G 1. Set Target A1.1. Set Target Device and Synthesis T A1.2. Set Target Reference Design A1.2. Set Target Interface S @ 2. Prepare Model For HDL Code Generation	1.3. Set Target Interf Analysis (^Triggers Up Set target Interface for Input Parameters Processor/FPGA synch Enable HDL DUT of Target platform Interf	face date Diagram) HDL code generation: Free pronization: Free putput port generation ace table	eration ee running eration for te	st points		
Age 3. HDC Code Generation A. Embedded System Integration	Port Name validIn3 validIn4	Port Type Inport Inport	Data Type boolean boolean	Target Platform Interfaces No Interface Specified No Interface Specified	Interface Mapping .	Interface Options
	out1 out2	Outport Outport	int16 int16	IP Data 0 OUT [0:15] IP Data 1 OUT [0:15]	• [0:15] • [0:15]	
	out3 out4 out5	Outport Outport Outport	int16 int16 int16	IP Data 2 OUT [0:15] IP Data 3 OUT [0:15] LLDK DAC Data 0 OUT [0:15]	• [0:15] • [0:15] • [0:15]	
	out6 out7	Outport Outport	int16 int16	LLDK DAC Data 1 OUT [0:15] LLDK DAC Data 2 OUT [0:15]	- [0:15] - [0:15]	
	validOut1 validOut2	Outport Outport Outport	boolean boolean	IP Data Valid OUT IP Load Tx Data OUT	- [0] - [0]	

Figure 12. Set Target Interface Output Ports (Rx & Tx Reference Design)

IP DATA <x> OUT is the data in offset binary format sent to the DAC interface IP. Data must be sent at 15MSPS when both channels are enabled or at 30MSPS when only one channel is enabled. The validOut<x> signal should have the same behavior as validIn. If you make changes to the data captured by the adc (delay for 1 clock period) and want to send it to the dac output, make sure you delay the validOut signal at the same time. If the feedback resistors are placed in the default position, which is +/-10V, a 0000h code will represent -10.382V and a ffffh code will represent 10.380V as described in *Table 1*.

			Hardwar	e	AD3552	2R Register Settings	
	Output Span	V _{zs} (V)	V _{FS} (V)	Feedback Connection	CH0_CH1_OUTPUT_RANGE	CHx_GAIN_SCALING_N	CHx_OFFSET
	+/-10V(Default)	-10.382	10.380	DAC0 to DAC0_RFB0	0x100	0	-245
	+/- 5V	-5.165	5.166	DACO to DACO_RFBO_X2	0x011	0	-495
CHO	10V	-0.165 10.163		DACO to DACO_RFBO_X2	0x010	0	495
	5V	-0.078	5.077	DAC0 to DAC0_RFB0_X1	0x001	0	0
	2.5V	-0.198	2.701	DAC0 to DAC0_RFB0_X1	0x000	3	-48
	+/-10V(Default)	-10.382	10.380	DAC1 to DAC1_RFB1	0x100	0	-245
	+/- 5V	- <mark>5.1</mark> 65	5.166	DAC1 to DAC1_RFB1_X2	0x011	0	-495
CH1	10V	-0.165	10.163	DACO to DACO_RFBO_X2	0x010	0	495
	5V	-0.078	5.077	DAC1 to DAC1_RFB1_X1	0x001	0	0
	2.5V	-0.198	2.701	DAC1 to DAC1_RFB1_X1	0x000	3	-48

Table 1: AD3552R DAC Output Span Configuration

Table 2, Table 3 and Table 4 show port descriptions for HDL DUT in Tx, Rx and Tx & Rx Reference Designs.

Interface signal name	Width	Description
IP Data 0 IN	16	Custom IP data input signal. This signal is connected to the AXI_LTC2387_0 IP ADC DATA port in the ADI reference design.
IP Data 1 IN	16	Custom IP data input signal. This signal is connected to the AXI_LTC2387_1 IP ADC DATA port in the ADI reference design.
IP Data 2 IN	16	Custom IP data input signal. This signal is connected to the AXI_LTC2387_2 IP ADC DATA port in the ADI reference design.
IP Data 3 IN	16	Custom IP data input signal. This signal is connected to the AXI_LTC2387_3 IP ADC DATA port in the ADI reference design.
IP Valid Tx Data IN	1	Input signal that has logic 1 value for a clock cycle period when the data starts to be valid.
CN0585 DAC Data 0 OUT	16	AD3552R_0 DAC 0 channel data. To be used as input into the AD3552R interface IP.
CN0585 DAC Data 1 OUT	16	AD3552R_0 DAC 1 channel data. To be used as input into the AD3552R interface IP.
CN0585 DAC Data 2 OUT	16	AD3552R_1 DAC 0 channel data. To be used as input into the AD3552R interface IP.
CN0585 DAC Data 3 OUT	16	AD3552R_1 DAC 1 channel data. To be used as input into the AD3552R interface IP.
IP Load Tx Data OUT	1	Custom IP output signal used to notify the design that the IP is ready to receive new input data. Output signal that must be logic '1' for a clock cycle period when the data starts to be valid.

Table 2: HDL DUT Ports for Transmit Reference Design (Tx)

Interface signal name	Width	Description
CN0585 ADC Data 0 IN	16	Custom IP data input signal. This signal is connected to the AXI_LTC2387_0 IP ADC DATA port in the ADI reference design.
CN0585 ADC Data 1 IN	16	Custom IP data input signal. This signal is connected to the AXI_LTC2387_1 IP ADC DATA port in the ADI reference design.
CN0585 ADC Data 2 IN	16	Custom IP data input signal. This signal is connected to the AXI_LTC2387_2 IP ADC DATA port in the ADI reference design.
CN0585 ADC Data 3 IN	16	Custom IP data input signal. This signal is connected to the AXI_LTC2387_3 IP ADC DATA port in the ADI reference design.
IP Valid Rx Data IN	1	Input signal that has logic 1 value for a clock cycle period when the data starts to be valid.
IP Data 0 OUT	16	ADAQ23876 ADC 0 channel data. To be used as input for the ADC CPAK IP.
IP Data 1 OUT	16	ADAQ23876 ADC 1 channel data. To be used as input for the ADC CPAK IP.
IP Data 2 OUT	16	ADAQ23876 ADC 2 channel data. To be used as input for the ADC CPAK IP.
IP Data 3 OUT	16	ADAQ23876 ADC 3 channel data. To be used as input for the ADC CPAK IP.
IP Data Valid OUT	1	Output signal that has to be logic '1' for a clock cycle period when the data starts to be valid.

 Table 3: HDL DUT Ports for Receive Reference Design (Rx)

Interface signal name	Width	Description
IP Data 0 IN	16	Custom IP data input signal. This signal is connected to the AXI_LTC2387_0 IP ADC DATA port in the ADI reference design.
IP Data 1 IN	16	Custom IP data input signal. This signal is connected to the AXI_LTC2387_1 IP ADC DATA port in the ADI reference design.
IP Data 2 IN	16	Custom IP data input signal. This signal is connected to the AXI_LTC2387_2 IP ADC DATA port in the ADI reference design.
IP Data 3 IN	16	Custom IP data input signal. This signal is connected to the AXI_LTC2387_3 IP ADC DATA port in the ADI reference design.
CN0585 ADC Data 0 IN	16	Custom IP data input signal. This signal is connected to the AXI_LTC2387_0 IP ADC DATA port in the ADI reference design.
CN0585 ADC Data 1 IN	16	Custom IP data input signal. This signal is connected to the AXI_LTC2387_1 IP ADC DATA port in the ADI reference design.
CN0585 ADC Data 2 IN	16	Custom IP data input signal. This signal is connected to the AXI_LTC2387_2 IP ADC DATA port in the ADI reference design.
CN0585 ADC Data 3 IN	16	Custom IP data input signal. This signal is connected to the AXI_LTC2387_3 IP ADC DATA port in the ADI reference design.
IP Valid Rx Data IN	1	Input signal that has logic 1 value for a clock cycle period when the data starts to be valid.
IP Valid Tx Data IN	1	Input signal that has logic 1 value for a clock cycle period when the data starts to be valid.
IP Data 0 OUT	16	ADAQ23876 ADC 0 channel data. To be used as input for the ADC CPAK IP.
IP Data 1 OUT	16	ADAQ23876 ADC 1 channel data. To be used as input for the ADC CPAK IP.
IP Data 2 OUT	16	ADAQ23876 ADC 2 channel data. To be used as input for the ADC CPAK IP.
IP Data 3 OUT	16	ADAQ23876 ADC 3 channel data. To be used as input for the ADC CPAK IP.
CN0585 DAC Data 0 OUT	16	AD3552R_0 DAC 0 channel data. To be used as input into the AD3552R interface IP.
CN0585 DAC Data 1 OUT	16	AD3552R_0 DAC 1 channel data. To be used as input into the AD3552R interface IP.
CN0585 DAC Data 2 OUT	16	AD3552R_1 DAC 0 channel data. To be used as input into the AD3552R interface IP.
CN0585 DAC Data 3 OUT	16	AD3552R_1 DAC 1 channel data. To be used as input into the AD3552R interface IP.
IP Data Valid OUT	1	Output signal that has to be logic 'l' for a clock cycle period when the data starts to be valid.
IP Load Tx Data OUT	1	Custom IP output signal used to notify the design that the IP is ready to receive new input data. The duration must be 1 clock cycle.

 Table 4: HDL DUT Ports for Transmit & Receive Reference Design (Tx & Rx)

2.1 Run the task, as shown in Figure 13.

	HDL Workflow Advisor - testModel_bitswitch/HDL_DUT		٥	×
File Edit Run Help				
Find: 🔹 🔹 🗢				
 ♥ HDL Workflow Advisor ▼	2.1. Check Model Settings Analysis			
 ^1.1. Set Target Device and Synthesis Tool ^1.2. Set Target Reference Design ^1.3. Set Target Interface 2. Prepare Model For HDL Code Generation 2.1. Check Model Settings 4. EnDL Code Generation 4. Embedded System Integration 	Check basic model-level settings for HDL code generation. To run additional HDL code generation compatibility checks, click button below to launch the HDL Code Advisor (opens in a separate window). Input Parameters	k th	e	
	HDL Code Advisor			
	Result: 🥥 Passed			
	Passed To run additional compatibility checks, click the button to launch the HDL Code Advisor (opens in a separate window).			

Figure 13. Check Model Settings

3.1 Select Verilog for the HDL Code Generation Settings, then run task as shown in Figure 14.

		HDL Workflow Advisor - testModel_bitswitch/HDL_DUT		D	0
ile Edit Run Help					
ind: 🚽 🗣 🗇					
 HDL Workflow Advisor 1. Set Target 1.1. Set Target De 1.1. Set Target Re 1.3. Set Target Inf 2. Prepare Model For F 2.1. Check Model S 3. HDL Code Generatic 	evice and Synthesis Tool ference Design terface IDL Code Generation ettings on	3.1. Set HDL Options Analysis Set options for HDL code and testbench generation Input Parameters HDL Code Generation Settings			
3.1., Set HDL Option 3.2. Generate RTL Solver Data importExport Math and Data Types Diagnostics Hardware implementation Mode Generation Target Optimization	Configuration Parameters: testModel_bitswitch/Configuration (Active)				
	Solver Solver Data ImportExport Math and Data Types Diagnotics Hardware Implementation Model Referencing Simulation Target Code Generation Target Optimization	Set Basic Options Generate HDL, for: testModel_bitswitch/HDL_DUT HDL VHDL Verlog Restore Model Defaults Run Compatibility Ch. Generate	Browse		

Figure 14. Set HDL Options

3.2 Run the task, as shown in Figure 15.

	HDL Workflow Advisor - testModel_bitswitch/HDL_DUT	- a x
File Edit Run Help		
Find: 🔹 🗢 💠		
 WDL Workflow Advisor I Set Target 1. Set Target Device and Synthesis Tool 1.2. Set Target Reference Design 1.3. Set Target Interface 2. Propare Model For HDL Code Generation 3.1. Set HDL Options 3.1. Set HDL Options 3.2. Generate RHL Code and IP Core 4. Embedded System Integration 4.2. Greater Bistream 4.4. Program Target Device 	3.2. Generate RTL Code and IP Core Additional source files:	Add Source

Figure 15. Generate RTL Code and IP Core.

4.1 Run the task (this will create the Vivado block design in the hdl_prj/vivado_ip_prj folder, or the project folder name that was chosen in 1.1), as shown in Figure 16.

M Applications Places System 📄 🖬 🥪	HDI Workflow Advisor - testModel bitswitch/HDI DIIT
File Edit Run Help Find:	
 HDL Workflow Advisor 1. Set Target 1.1. Set Target Reference Design 1.2. Set Target Interface 2. Prepare Model For HDL Code Generation 2.1. Check Model Settings 3. HDL Code Generation 3.1. Set HDL Options 3.1. Set HDL Options 3.1. Set HDL Code Generation 4.1. Seter Project 4.2. Generate RTL Code and IP Core 4.1. Greate Project 4.2. Generate Project 4.3. Build PFOA Bitstream 4.4. Program Target Device 	4.1. Create Project Analysis Create project for embedded system tool Input Parameters Embedded system tool: Xilinx Vivado with IP Integrator Project folder: hdl_pri/Vivado_ip_pri Synthesis objective: None Enable IP caching Result: Passed Passed Passed Create Project: Task "create Project." String Xilink Vivado or Fri Jun 10:9:30:5 MOT 2021 **** SF Build 32:46:46 or Fri Jun 10:9:30:57 MOT 2021 **** SF Build 32:16:46 or Fri Jun 10:9:30:57 MOT 2021 **** SF Build 32:16:46 or Fri Jun 10:9:30:55 MOT 2021 ************************************

Figure 16. Create Project.

4.2 Run the task in Figure 17.



Figure 17. Generate Software Interface

4.3 Choose the "Custom" option for the Tcl file synthesis build, then Browse for the adi_build.tcl file located under HighSpeedConverterToolbox/CI/scripts, as shown in Figure 18:

	HDL Workflow Advisor - testModel_bitswitch/HDL_DUT	•		×
File Edit Run Help Find:				
 HDL Workflow Advisor HDL Workflow Advisor 1. Set Target 1.1. Set Target Device and Syntf 1.2. Set Target Reference Design 1.3. Set Target Interface 2.1. Check Model Settings 3. HDL Code Generation 3.1. Set HDL Options 3.2. Generate RTL Code and IP C 4.1. Create Project 4.2. Generate Software Interface 4.4. Program Target Device 	4.3. Build FPGA Bitstream Analysis Synthesis and generate bitstream for embedded system on FPGA			
	Run build process externally Tcl file for synthesis build: Custom eedConverterToolbox/Cl/scripts/adi_build.tcl Enable routed design checkpoint for build Routed design checkpoint file for build: Default Routed design checkpoint file: hdl_pri/checkpoint/system_routed.dcp Maximum number of cores for build: synthesis tool defau +	Brows	: e .	
	Run This Task Result: 🗉 Not Run			
	Click Run This Task.			
()))) ()) () () () () () ()	Ŀ	lelp	App	ly

Figure 18. Build FPGA Bitstream

A bash prompt will open, and you can see the entire build process log file, as shown in Figure 19, and Figure 20.

This step usually takes about an hour.



Figure 19. Build FPGA Bitstream Console Output

	HDL Workflow Advisor - testModel_bitswitch/HDL_DUT		×		
bash 😑					
32 Info., 0 Barnings, 0 Critical Marnings and 0 Errors encountered with_bithrem completed successfully writh_bithrems. Then (9): cps = 000040 : elagesd = 0000024, M, Ms = 4000.675 ; psin = 358.755 ; free physical = 11964 ; free virt INFD: [Common 17-206] Exiting Vivado at led Bec 14 1352:41 2022., [Wed Bec 14 1552:47 2022] incl = 1.1 finished wait.on.runt Time (9): cpu = 00000102; elapsed = 000701, Hemo- 1339.369 ; psin = 0.000; free physical = 14199 ; free virtual = INFD: [Vivado 12-4859] Creating Mardamer Platform: /home/spopMark. ndd_lastErg=/MingSSeedCommeter foolbohndl.psr/Vivado.jp.pr// INFD: [Ms 35:2053] elapsed time for repository (/ema/wediadata/ 04/2021_i/444/webeddom)) boding 1 seconds	(#B): peak = Idrebare Idrebare and generate bitstream for embedded system on FPGA weters x/11/m/Viva				
INGD: [Vivado 12:1264] The Hardware Platform can be used for Hard INGD: [Vivado 12:4685] Successfully created Hardware Platform: And Id&_rebased_on_hal_meeter_ref/NighipeedConverterToolbox/hal_er_j/vi Plasarb wait. bits may take a few minutes. Ebededde system build completed. Tow may close this shell. ENGD: [Common 17-206] Exiting Vivado at Wed Dec 14 13:54:03 2022.	<pre>we we w</pre>	Browse			
4.3. Build FPGA Bitstream	House design checkpoint file: hdl_prj/checkpoint/system_routed.dcp	Browse			
4.4. Program Target Device	Maximum number of cores for build: synthesis tool defau -				
	Run This Task				
	Result: 📀 Passed				
	Warning Run build process externally: The system build has been launched in an external shell, please check the external make sure the bitstream generation is completed. Once complete, the generated bitstream file is located at: "hdl_prj/vivado_jp_prj/vivado_prj.runs/impl_1/system_top.bit".	console to			
	Warning The system build may fail if the timing constraints are not met by the synthesis tool. If a timing failure occurs, the will be renamed to "bdll pri/vivado in pri/vivado pri runs/impl 1/system too timingfailure bit". For more advice on ho	he bitstream	+		
4		Help App	y		

Figure 20. Build FPGA Bitstream Task Complete Message

In the end you will get this message, and the generated BOOT.BIN file will be located here, Figure 21.



Figure 21. Location of the BOOT.BIN File

4.4 Program target device

This tab in the HDL Workflow Advisor is incompatible with The ADI SD card flow. Instead, choose one of the following methods to update the BOOT.BIN file on the SD card.

After the BOOT.BIN file is generated, you have 2 options:

- Copy the BOOT.BIN file on the SD Card directly.
- Send it via network using a terminal (CMD for Windows machine): 1. Go to the folder where the BOOT.BIN file is:

HighSpeedConverterToolbox/hdl_prj/vivado_ip_prj/boot

2. Run this command:

scp BOOT.BIN root@<your_board_ip>:/boot

Finally, reboot the board.