BASYS3 board tutorial

(Decoder design using Vivado 2015.1)

Note: you will need the Xilinx Vivado Webpack version installed on your computer (or you can use the department systems).

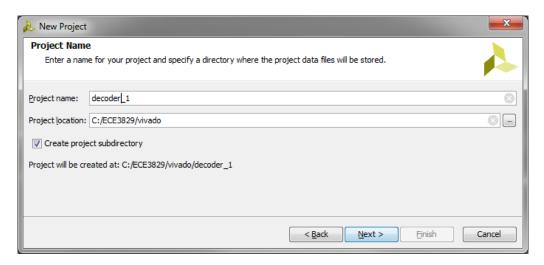
This tutorial shows how to create a simple combinational design (a 3 to 8 decoder using the slider switches and leds) that can be implemented on the Basys3 board.

Start Vivado Design Suite:



Select Create New Project.

Click Next and then enter a Project name and location for your project:



1

Click Next and select the RTL project type:

🚴 New Project	
Project Type Specify the type of project to create.	•
 <u>RTL Project</u> You will be able to add sources, create block designs in IP Integrator, generate IP, run RTL analysis, synthesis, implementation, design planning and analysis. <u>Do not specify sources at this time</u> <u>Post-synthesis Project:</u> You will be able to add sources, view device resources, run design analysis, planning and implementation. <u>Do not specify sources at this time</u> <u>I/O Planning Project</u> 	
< <u>Back</u> <u>Next</u> <u>Finish</u> Cancel]

Check the "Do not specify courses at this time" box and click Next:

Select the corect Xilinx FPGA that is on the Basys3 board (XC7A35T-1CPG236C)

🚴 New Project						1000		x
Default Part Choose a default Xil Select: Parts	inx part or bo Boards	ard for your proj	ject. This can b	e changed later.				
Product category:	All		-	Package:	cpg236		•	
Eamily:	Artix-7		Ŧ	Spee <u>d</u> grade:	-1		•	
Sub-Family:	Artix-7		*	Temp grade:	С	c 🗸		
				Si Revision:	All Remain	ning	•	
Search: Q-			Reset	All Filters		- ber		
Part	I/O Pin Count	Available IOBs	LUT Elements	FlipFlops	Block RAMs	DSPs	Gb Transceivers	G TI
🔷 xc7a15tcpg236-1	236	106	10400	20800	25	45	2	2
xc7a35tcpg236-1 xc7a50tcpg236-1	236 236	106 106	20800 32600	41600 65200	50 75	90 120	2 2	2
•	III							
				< <u>B</u> ack	<u>N</u> ext >	> <u>F</u> inish	Cance	

Click Next, and then Finish:

🚴 New Project	
	New Project Summary
VIVADO.	 A new RTL project named 'decoder_1' will be created.
	 The default part and product family for the new project: Default Part: xc7a35tcpg236-1 Product: Artix-7 Family: Artix-7 Package: cpg236 Speed Grade: -1
EXILINX ALL PROGRAMMABLE-	To create the project, dick Finish
	< Back Next > Einish Cancel

The Project window opens:

🚴 decoder_1 - [C:/ECE3829/vivado/d	decoder_1/decoder_1_xpr] - Vivado 2015.1	x					
Ele Edit Flow Tools Window Lagout Yew Help							
達 🔄 📾 🦇 🏬 📉 🔸 🔈 👌 🌀 🛞 🔽 🌀 🗄 Default.tayott 🔷 🎉 🛞 😵 😵 😵 Ready							
Flow Navigator V Project Hanager-decode_1 X							
오 🖫 🖨	Sources _ L L X D Project Summary X						
Project Manager	x = i and i	*					
St Add Sources V Language Templates	Constant Criterian Cr						
IP Integrator	Synthesis	*					
Create Block Design	Status: Not started Status: Not started						
Generate Blick Design	Messages: No errors or warnings Messages: No errors or warnings						
	Hierarchy Librais Comple Order Part: xc7a35tpg236-1 Part: xc7a35tpg236-1 Part: xc7a35tpg236-1						
 Simulation 	A Sources Templates Strategy: Windo Synthesis Defaults Incrementation Defaults Incrementation Defaults						
🔞 Simulation Settings	Properties _ L 2 ×						
(Run Simulation	← + 10 k DRC Volations & Timing	*					
# RTI Analysis		^					
C Elaboration Settings	Run Implementation to see DRC results Run Implementation to see timing results						
Open Elaborated Design	2 Utilization	*					
4 Synthesis	Select an object to see properties Utilization X Power	*					
🔞 Synthesis Settings	Run Synthesis to see utilization results Run Implementation to see power results						
🗞 Run Synthesis							
Open Synthesized Design							
4 Implementation							
(5) Implementation Settings	Design Runs L 2 Q Name Constraints Status Progress WNS THS WHS THS TPWS Failed Routes LUT % LUTs FF % FFs BRAM % BRAMs DSP % DSPs Start	_					
Run Implementation	Name Constraints Status Progress WINS THS WINS THS THYS Failed Routes LUT% LUT% FP % FPs BRAM % BRAM % BSP % DSP % DSP % Start Start Start	- 1					
Open Implemented Design	we mpi 1 constrs 1 Not started 0%						
4 December and Datum	3						
 Program and Debug Bitstream Settings 							
Cenerate Bitstream							
Open Hardware Manager		- 1					
	*	- 1					
		•					
	Td Console 🗋 Messages 🔄 Log 🗋 Reports 🔪 🗩 Design Runs	_					
l							

We now need to add a Verilog design source to describe our decoder operation.

Click on Add Sources in the left Project Manager window (or select the menu item File => Add Sources:

Add Sources	
VIVADO.	Add Sources This guides you through the process of adding and creating sources for your project
	 Add or greate constraints Add or create design sources
	 Add or create simulation sources Add or create DSP sources
	 Add existing block design sources Add existing IP
XILINX	To continue, click Next
	< Back Next > Einish Cancel

Select Next,

And then select Create File (click on the + symbol) and enter decoder for the file name:

🚴 Create So	urce File	×
Create a new	source file and add it to your project.	
<u>F</u> ile type:	🔞 Verilog	-
File name:	decoder	8
File location:	🛜 <local project="" to=""></local>	-
	ОК	ancel

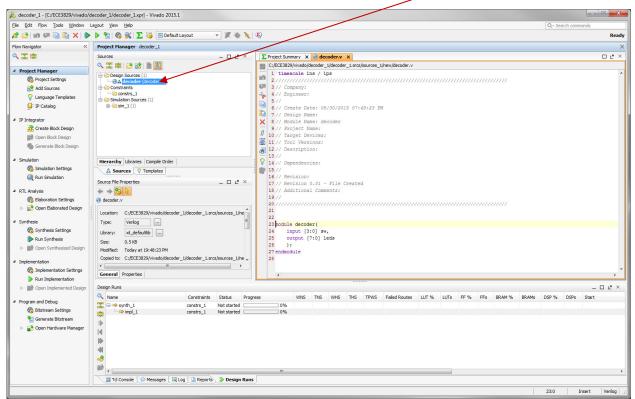
Then click OK and Finish.

We can now specify the inputs and outputs to create our 3 to 8 decoder (we will use three switches and eight LEDs):

For each p MSB an	nodule an port speci Id LSB vali	d specify I/O	ored unless i									×
	finition le name: ort Defini	L									8]
1/0 P												1
+	Port Na	me	Direction		Bus	MSB	LSB					
-	SW		input	•	V	2		0				
↑	led		output	•	1	7		0				
↓												
										ОК	Cance	

Click OK.

Back in the Project Manager Sources window double-click the new decoder.v file and you will then see the verilog file appear in the window on the right:



You should add your name and a description of this file to the header description.

We can now add the verilog statements to design our 3 to 8 decoder.

There are anumber of ways to design the decoder, below is an example using a case statament:

deco	oder.	v	×
8	C:/E	ECE3829/vivado/decoder_1/decoder_1.srcs/sources_1/new/decoder.v	
-	23	module decoder(
	24	input [2:0] sw,	
621	25	output reg [7:0] led // reg type required	
-20	26);	
Đ	27		
	28	always @ (sw) // we will use a case statement for this decoder	
1	29	case (sw)	
X	30	3'b000: led = 8'b00000001;	
11	31	3'b001: led = 8'b00000010;	
	32	3'b010: led = 8'b00000100;	
	33	3'b011: led = 8'b00001000;	
Æ	34	3'b100: led = 8'b00010000;	
	35	3'b101: led = 8'b00100000;	Ξ
	36	3'b110: led = 8'b01000000;	-
	37	3'b111: led = 8'b10000000;	
	38	endcase	
	39		
	40	endmodule	Ŧ
		۰ III ۲۰۰۰ ۲۰۰۰ ۲۰۰۰ ۲۰۰۰ ۲۰۰۰ ۲۰۰۰ ۲۰۰	

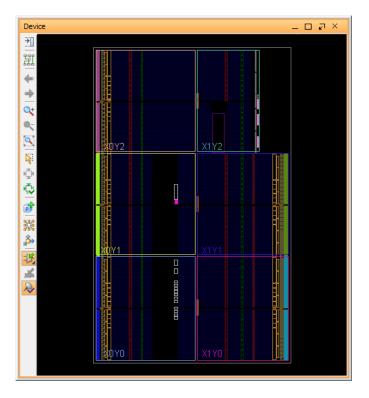
Now we can synthesize the design.

Click Run Synthesis in the Project Manager window.

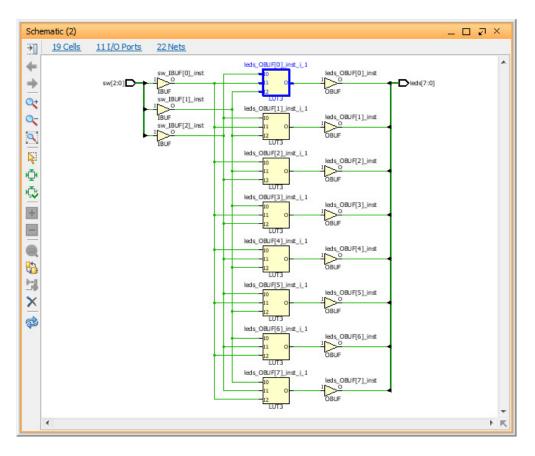
After synthesis is complete there should be no errors or warnings reported.

If you open the synthesized design you can see a device level representation (this is mostly empty since we just have a very simple design that only uses a small fraction of the available FPGA resources):

6



But you can also look at a schematic representation to see the input and output buffers and the LUTs used:



7

Before we can implement the design we need to specify the FPGA pins that will be used for the SW inputs and LED outputs.

Look at the Basys3 manual to determine the FPGA pins.

Next click Add Sources and select 'Add or create constraints':

Add Sources	
VIVADO.	Add Sources This guides you through the process of adding and creating sources for your project
	 Add or greate constraints Add or create design sources Add or create simulation sources Add or create DSP sources
E XILINX Al programmable.	 Add existing block design sources Add existing IP To continue, dick Next
	< <u>B</u> ack <u>Next ></u> Einish Cancel

We can name his decoder

🚴 Create Co	👃 Create Constraints File								
Create a new project	Create a new constraints file and add it to your project								
<u>File</u> type:	XDC	-							
File name:	decoder								
Fil <u>e</u> location:	🛜 <local project="" to=""></local>	•							
	ОК	Cancel							

In the souces window window select the constraints file by the hierarchy Constraints => constrs_1 => decoder.xdc. We can then add location constraints for all the inputs and outputs (you can download a copy of the Basys3 XDC constraints from the Digilent website – just copy the pins you are using for the design):

These constraints specify the pins to use for each signal and what type of interface.

decoder.xdc	_ 🗆 🖉 🗙
C:/ECE3829/vivado/decoder_1/decoder_1.srcs/constrs_1/new/decoder.xdc	
1 # Svitches	^
<pre>2 set_property PACKAGE_PIN V17 [get_ports {sw[0]}]</pre>	
3 set_property IOSTANDARD LVCMOS33 [get_ports {sw[0]}]	
4 set_property PACKAGE_PIN V16 [get_ports {sw[1]}]	
5 set_property IOSTANDARD LVCMOS33 [get_ports {sw[1]}]	
6 set_property PACKAGE_PIN W16 [get_ports {sw[2]}]	
7 set_property IOSTANDARD LVCMOS33 [get_ports {sw[2]}]	
× 8	
9 # LEDs 10 set property PACKAGE PIN U16 [get ports {led[0]}]	
11 set_property IOSTANDARD LVCMOS33 [get_ports {led[0]}] 12 set property PACKAGE_PIN E19 [get ports {led[1]}]	
12 set property PACKAGE_PIN EI9 [get_ports {led[1]}] 13 set property IOSTANDARD LVCMOS33 [get ports {led[1]}]	
<pre>13 Set_property PACKAGE_PIN U19 [get ports {led[2]}]</pre>	
15 set property IOSTANDARD LVCMOS33 [get ports {led[2]}]	
16 set property PACKAGE_PIN V19 [get ports {led[3]}]	
17 set property IOSTANDARD LVCMOS33 [get ports {led[3]}]	
18 set property PACKAGE_PIN W18 [get ports {led[4]}]	
<pre>19 set_property IOSTANDARD LVCMOS33 [get_ports {led[4]}]</pre>	
20 set_property PACKAGE_PIN U15 [get_ports {led[5]}]	
<pre>21 set_property IOSTANDARD LVCMOS33 [get_ports {led[5]}]</pre>	
22 set_property PACKAGE_PIN U14 [get_ports {led[6]}]	
<pre>23 set_property IOSTANDARD LVCMOS33 [get_ports {led[6]}]</pre>	
24 set_property PACKAGE_PIN V14 [get_ports {led[7]}]	
<pre>25 set_property IOSTANDARD LVCMOS33 [get_ports {led[7]}]</pre>	
26	-
•	Þ

Now we have specified the correct pins to use for the design (so it matches the Basys3 board layout) we can implement the design.

Click Run Implementation in the Project Manager window.

You will find there are two warnings after implementation. This is because we have not specified any timing constraints. For this simple combinational circuit we can ignore this.

& decoder_1 - [C:/ECE3829/vivado/d	decoder_1/decoder_1.xpr] - Vivado 2015.1						
File Edit Flow Tools Window L	Layout View Help		Q,- Search commands				
🯄 🔂 in 🕫 🏬 🐘 🗙 🔌	🕨 🎽 🊳 🐝 🔽 😼 🔚 Default Layout 💿 🗶 🔌 🏌	E)	Implementation Complete				
Flow Navigator «	Project Manager - decoder_1		×				
Q 🛣 🖶	Sources _ D L ^a ×	∑ Project Summary × @ decoder.v × b decoder.xdc ×	口 ピ ×				
4 Project Manager	🔍 🖾 🖨 📸 📓 📓	Project Settings	Edit 🛠				
Image: Register of the second sec		Project Name: decoder_1 Project location: Cr;ECE3829/wado/decoder_1 Product family: Artiv-7 Project part: <u>xc?a35ccre226-1</u> Top module name: <u>decoder</u>					
IP Integrator		Synthesis *	Implementation *				
 Create Block Design Open Block Design Generate Block Design Simulation Simulation Run Simulation 		Status: Complete Messages: No errors or warnings Part: xr-23Strop128-1 Strategy: <u>Wwado Symthese Defaults</u>	Status: ✓ Corplete Messages: ④ Zmarrings Part: v::Ch335rgc326-1 Stratey: Wrado Indementation Defaults Incremental concerts Issues Summary Route Status				
A RTL Analysis		DRC Violations *	Timing *				
RIL Analysis Genetic and Settings Joint Elaboration Settings Joint Elaborated Design Synthesis Synthesis Settings Nun Synthesis Joint Synthesis Dim Synthesized Design	Herarchy Loraris Comple Order Sources @ Tempotes Source Sie Propotes @ docoder.v Location: C.JECE302//mado/6coder_1(decoder_1.srcs/sources_1/ne Type: Verlog	Summary: (1) I warning	Vorst Negative Slack (VNIS): NA Total Negative Slack (TNIS): NA Namber of Fealing Endpoints: NA Total Namber of Endpoints: NA Indemented Timing Report Setup Hold Pulse Width				
4 Implementation		Utilization - Post-Implementation	Power \$				
@ Implementation Settings ▶ Run Implementation ▷ @ Open Implemented Design ▲ Program and Debug @ Bitstream Settings ③ Generate Bitstream	Lorany: W_Densulo Ster: 1.048 Modified: Today at 11:454 MM Coperto tto: (FLCB202)/wada/decoder_1/decoder_1.srcs/sources_1/re Read-only: No Brompted: No Gobal include	UT 1% 10 25 50 75 160 URitation (%) Graph Table	Total On-Chip Power: 3.996 W Junction Temperature: 45.0 °C Themail Margn: 40.0 °C (8.0 W) Effective DA: 5.0 °C/W Power supplied to off-thip devices: 0.W Confidence level: Lozs				
👂 📸 Open Hardware Managar		Post-Synthesis Post-Implementation	Summary On-Chip				
	Genhend Properties Mesopei ① 0.03 statues ⑦ 0 203 statues Brow A ◎ Tangeneration ② www.mays ③ 0.03 statues ◎ Route Design ③ www.mays ◎ Power 33-223 / Ho user defined dudes were flowed in the design ◎ Power 33-223 / Ho user defined dudes were flowed in the design ◎ Tangeneration ③ Power 33-223 / Ho user defined dudes were flowed in the design ◎ Tangeneration ③ Power 33-223 / Ho user defined dudes were flowed in the design ◎ Tangeneration ③ Power 33-223 / Ho user defined dudes were flowed in the design ◎ Tangeneration ③ Power 33-223 / Ho user defined dudes were flowed in the design						
			9:5 Insert XDC;				

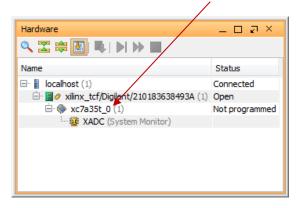
We can now generate the bitstream. Click on Generate Bitstream in the Project Manager window.

Next step is to program the FPGA with the bitsream.

On the Basys3 board make sure that JP1 is set to the JTAG mode and connect the USB cable to the board and turn on the power.

Select the Hardware Manager in the Project Manager and select Open Target and then Auto Connect.

You should see the Xilinx xc7a35t_0 in the Hardware Window:



Click on Program Device and select the decoder.bit bitstream (automatically filled in):

🚴 Program Device	×
	rogramming file and download it to your hardware device. You can optionally select a debug esponds to the debug cores contained in the bitstream programming file.
Bitstre <u>a</u> m file: Debu <u>q</u> probes file:	C:/ECE3829/vivado/decoder_1/decoder_1.runs/impl_1/decoder.bit
✓ Enable end of s	tartup check
	Program Cancel

Then select Program (ignore the warning about the missing debug core and the rule violation).

Programming should just take a few seconds and then the Done led on the Basys3 board will turn on.

You should now find that your decoder is implemented on the FPGA. Change the three slider switches (SW2, SW1, SW0) through all eight combinations and verify the correct corresponding LED turns on.

Congratulations!

You have entered a design and then Synthesized, Implemented, and programmed the FPGA with the generated bit file. This was a simple design example but the same steps are just repeated for any design.

Close the Hardware manager to go back to the Project Manager window.

Programming the Serial Flash

The FPGA is a volatile device and so the bit file will not be present after power cycling the board. We can load the QSPI serial flash on the Basys3 board so it loads the bit file from the flash on power up.

First we need to create a bin file to be able to program the serial flash.

Click on the Bitstream Settings in the Program and Debug section of the Project manager.

Select the bin_file option:

	Bitstream	
30	(i) Note: Additional bitstream settings will be available	once you open an implemented design
General	Write Bitstream (write_bitstream)	
	td.pre	
Simulation	td.post	
Sinuauon	-raw_bitfile	
	-mask_file	
Elaboration	-no_binary_bitfile	
	-bin_file	
>>	-readback_file	
Synthesis	-logic_location_file	
	-verbose	
	More Options	
mplementation		
2		
1010		
Bitstream		
=	-bin_file	
<u> </u>	Write a binary bit file without header (.bin).	
Īb		

Click OK.

Click on Generate Bitstream.

After generation, if you look in the decoder_1 => decoder_1.runs => impl_1 directory you will see a decoder.bit and a decoder.bin file.

Use the Hardware Manager to connect to the Basys3 board then right-mouse click on the FPGA and select Add Configuration Memory Device:

Hardware		_ 🗆 🖉 ×
옥 🛣 🖨 🛃 📭 🕨 🕨 🔳		
Name		Status
🖃 🚪 localhost (1)		Connected
🗄 📓 🤌 xilinx_tcf/Digilent/2101836384	1 93A	(1) Open
🗖 🔷 xc7a35t_0 (1)	6	Hardware Device Prop
🔤 XADC (System Monitor)	1	
	1	Program Device
	Ø	Refresh Device
	0	Add Configuration Mer
		Boot from Configuration

We now need to select the serial flash device that is on the Basys3 board.

Select Spansion as the Manufacturer, then select the 32Mb device:

Choose a confi	iguration memory (part. This can be ch	nanged later.						
vice: 🚸 xc7a35t_(D								
Manufacturer	Spansion		•			Туре	All		-
Density (Mb)	All		+			Width	All		+
			<u> </u>	<u>R</u> eset All Filter	-				
t Configuration Me Search: Q- Name	mory Part	Part	Manufacturer			Type	Density (Mb)	Width	
<u>S</u> earch: Q _. ✓			Manufacturer		Family	Туре	Density (Mb)		
Search: Q.	:1_x2_x4	Part <u>\$25fl032p</u> \$25fl064p				Type spi spi	Density (Mb) 32 64	x1_x2_x4	
Search: Q. Name S25fl032p-spi-x S25fl064p-spi-x	1_x2_x4 1_x2_x4	s25fl032p	Manufacturer Spansion Spansion		Family s25flxxxp	spi	32		
Search: Q- Name s25fl032p-spi-x s25fl064p-spi-x s25fl128sxxxxx	1_x2_x4 1_x2_x4 cx0-spi-x1_x2_x4	s25fl032p s25fl064p	Manufacturer Spansion Spansion Spansion	Alias	Family s25flxxxp s25flxxxp	spi spi	32 64	x1_x2_x4 x1_x2_x4	
Search: Q- Name s25fl032p-spi-x s25fl064p-spi-x s25fl128sxxxxx	1_x2_x4 1_x2_x4 cx0-spi-x1_x2_x4 cx1-spi-x1_x2_x4	s25fl032p s25fl064p s25fl128sxxxxx0 s25fl128sxxxxx1	Manufacturer Spansion Spansion Spansion	Alias	Family s25flxxxp s25flxxxp s25flxxxs	spi spi spi	32 64 128	x1_x2_x4 x1_x2_x4 x1_x2_x4	
Search: Q Name S25fl032p-spi-x S25fl064p-spi-x S25fl128sxxxxx S25fl128sxxxxx	1_x2_x4 1_x2_x4 xx0-spi-x1_x2_x4 xx1-spi-x1_x2_x4 1_x2_x4	s25fl032p s25fl064p s25fl128sxxxxx0 s25fl128sxxxxx1 s25fl132k	Manufacturer Spansion Spansion Spansion Spansion	Alias	Family s25flxxxp s25flxxxp s25flxxxs s25flxxxs	spi spi spi spi	32 64 128 128	x1_x2_x4 x1_x2_x4 x1_x2_x4 x1_x2_x4 x1_x2_x4	
Search: Q- Name s25f0032p-spi-x s25f064p-spi-x s25f128sxxxxx s25f128sxxxxx s25f128sxxxxx s25f132k-spi-x s25f164k-spi-x	1_x2_x4 1_x2_x4 x0-spi-x1_x2_x4 x1=spi-x1_x2_x4 1_x2_x4 1_x2_x4	s25fl032p s25fl064p s25fl128sxxxxx0 s25fl128sxxxxx1 s25fl132k	Manufacturer Spansion Spansion Spansion Spansion Spansion Spansion	Alias	Family s25flxxxp s25flxxxp s25flxxxs s25flxxxs s25flxxxs	spi spi spi spi spi	32 64 128 128 32	x1_x2_x4 x1_x2_x4 x1_x2_x4 x1_x2_x4 x1_x2_x4 x1_x2_x4	
Search: Q. Name S25fl032p-spi-x S25fl04p-spi-x S25fl128xxxxx S25fl128xxxxxx S25fl128xxxxxx S25fl128xxxxxx S25fl128xxxxxx S25fl164k-spi-x S25fl256xxxxxxx	1_x2_x4 1_x2_x4 x0-spi-x1_x2_x4 x1-spi-x1_x2_x4 1_x2_x4 1_x2_x4 xx0-spi-x1_x2_x4	s25fl032p s25fl064p s25fl128sxxxxx0 s25fl128sxxxxx1 s25fl12k s25fl132k s25fl164k	Manufacturer Spansion Spansion Spansion Spansion Spansion Spansion	Alias	Family s25flxxxp s25flxxxp s25flxxxs s25flxxxs s25flxxxs s25fl1 s25fl1	spi spi spi spi spi spi	32 64 128 128 32 64	x1_x2_x4 x1_x2_x4 x1_x2_x4 x1_x2_x4 x1_x2_x4 x1_x2_x4 x1_x2_x4	

Click OK

13



Click OK

Select the Configuration File (decoder.bin) in the impl_1 directory.

elect a configuration fil	e and set pro	gramming options.
Memory Device:		
Configuration file:		329/vivado/decoder_1/decoder_1.runs/impl_1/decoder.bin
State of non-config m	em I/O pins:	Pull-none 🔻
Program Operations		
Address Range:	Configuration	on File Only 👻
Erase		
Blank Check		
Program		
Verify		
Verify		
<u>▼</u> <u>V</u> erify		OK Cancel Apply

Click OK.

Note: this will erase any existing design in the QSPI flash (including the configuration file shipped with the Basys3 board).

The QSPI Flash will now be erased and then programmed with the decoder.bin file.

Once programmed, you can power off the Basys3 board and change the JP1 jumper mode from JTAG to QSPI.

Power back on the board and after a few seconds the Done LED will turn on indicating that your decoder design has been automatically loaded into the FPGA from the serial flash. You can verify the decoder design by moving the slider switches and observing the leds as before.