Vivado Simulation, Synthesis and Programming Tutorial for Basys3 board

This tutorial shows how to build a Vivado VHDL-project for Basys3 board. It is assumed to have Xilinx Vivado (WebPack) 2016.x installed. Although it is not mandatory, it is recommended to install Vivado board files for Digilent boards:

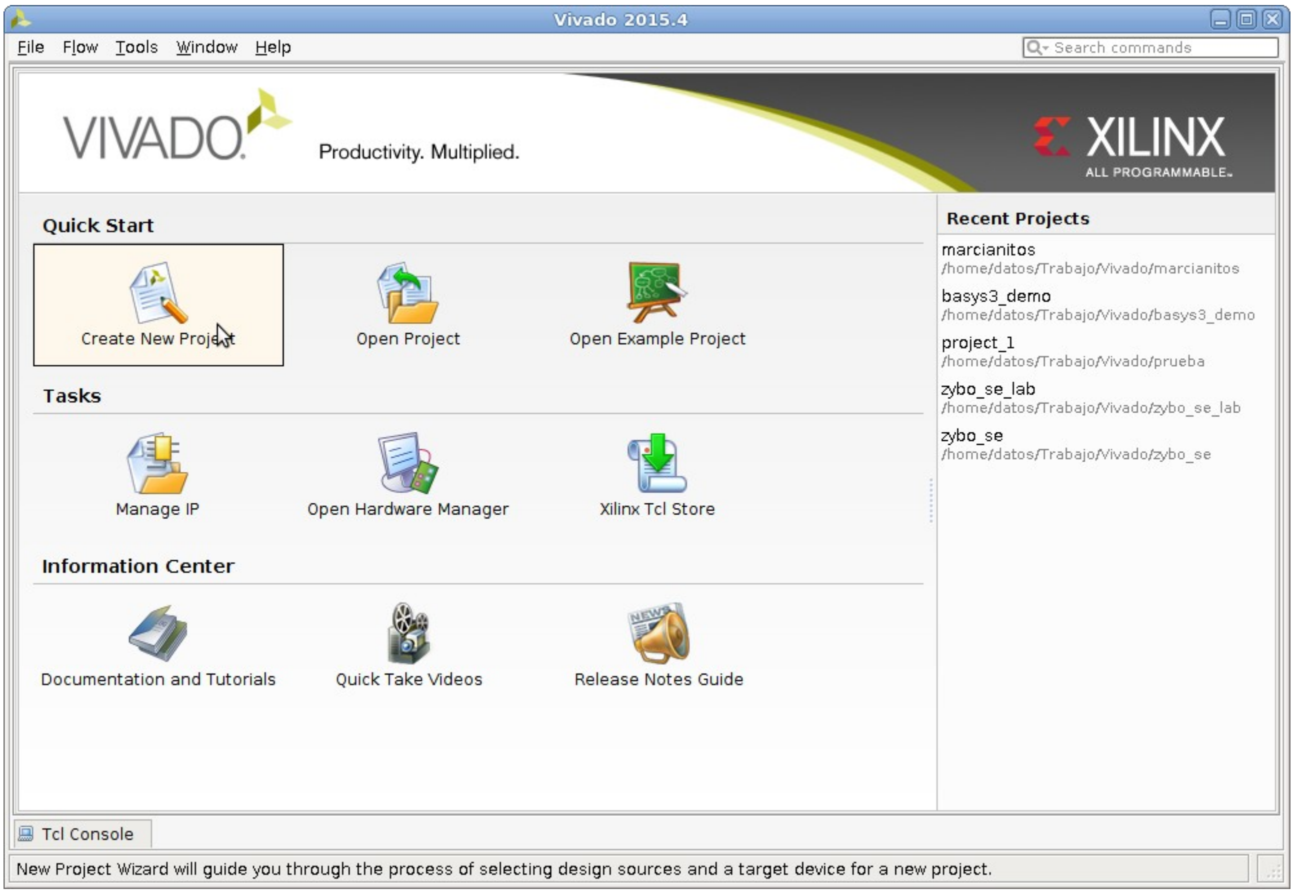
https://reference.digilentinc.com/learn/software/tutorials/vivado-board-files/start

The guide comprehends the following tasks:

- Project creation
- File addition: VHDL files and constraint file.
- Simulation
- Synthesis and implementation
- FPGA bitstream generation and download
- Report generation

Create a new project

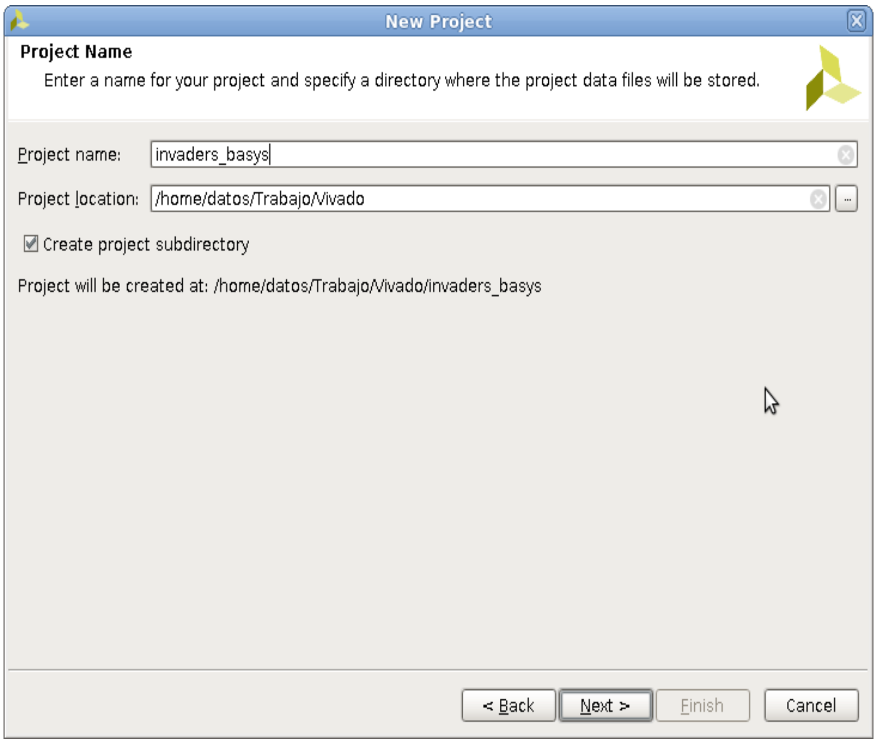
Start Vivado Design Suite, select Create New project and Click Next.



New Project Wizard will guide you through the process of selecting design sources and a target device for a new project.

Figure 1. Xilinx Vivado Design Suite

Specify project name and location. Be careful not to use spaces or non standard characters (like "ñ". Click Next.

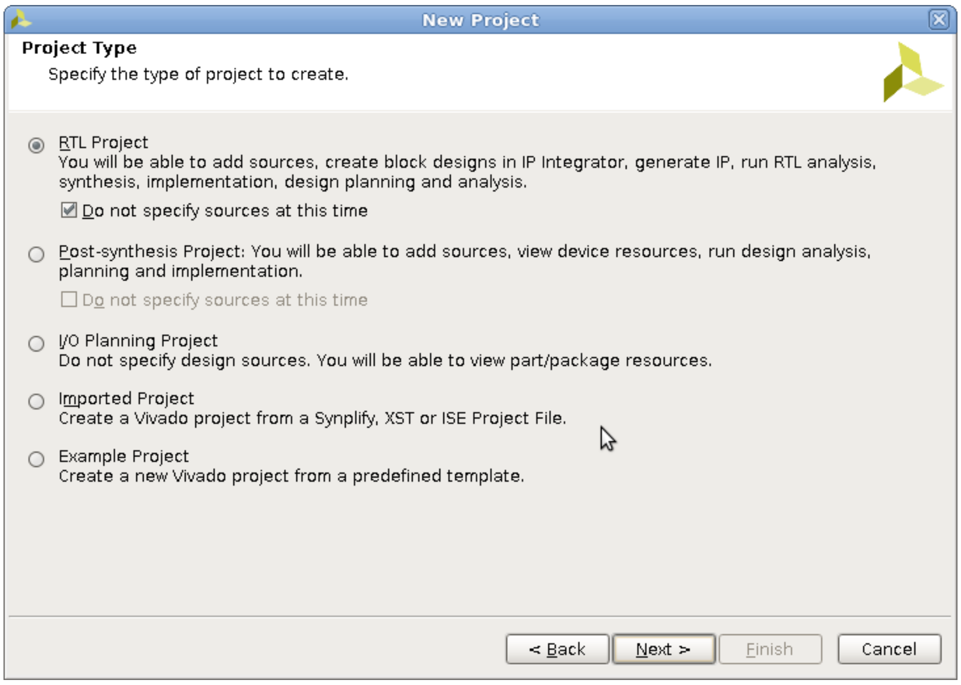


Project Name
Enter a name for your project and specify a directory where the project data files will be stored.

Project name: invaders_basys
Project location: /home/datos/Trabajo/Vivado
☑ Create project subdirectory
Project will be created at: /home/datos/Trabajo/Vivado/invaders_basys

Figure 2. Project name and location

Select Project Type: RTL Project. Activate "Do not specify sources at this time". Files will be added after project creation. Click Next.



Project Type
Specify the type of project to create.

● RTL Project
You will be able to add sources, create block designs in IP Integrator, generate IP, run RTL analysis, synthesis, implementation, design planning and analysis.
☑ Do not specify sources at this time

○ Post-synthesis Project: You will be able to add sources, view device resources, run design analysis, planning and implementation.
☐ Do not specify sources at this time

○ I/O Planning Project
Do not specify design sources. You will be able to view part/package resources.

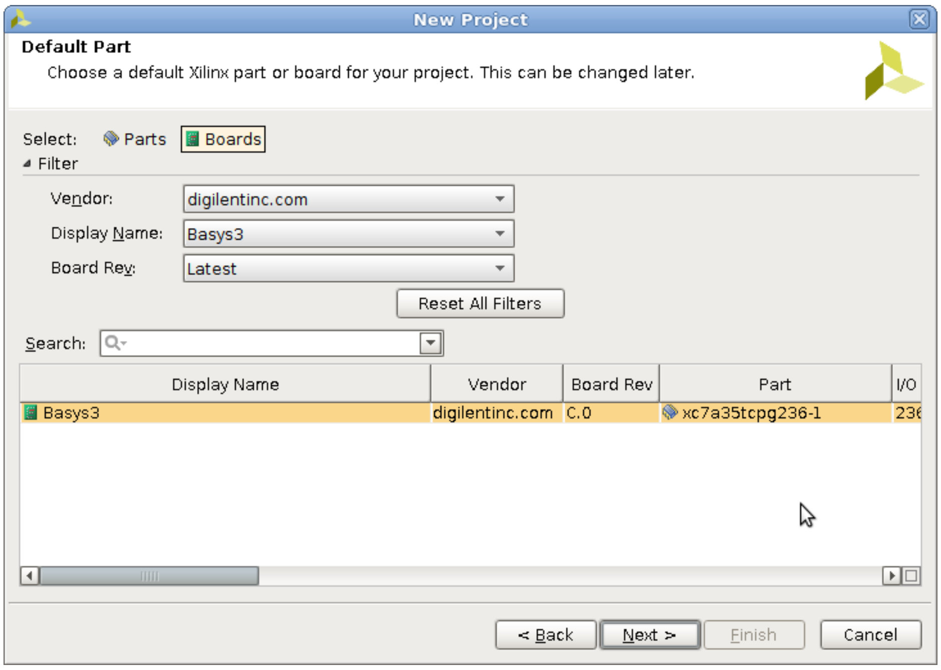
○ Imported Project
Create a Vivado project from a Synplify, XST or ISE Project File.

○ Example Project
Create a new Vivado project from a predefined template.

Figure 3. Project type

Next step can be fulfilled either by device or board selection. Board selection is preferrable, but Basys 3 board must be available for selection, as it is not installed with Xilinx software.

Board selection: Select “Boards” at the top.



Default Part

Choose a default Xilinx part or board for your project. This can be changed later.

Select: Boards

Vendor: digilentinc.com

Display Name: Basys3

Board Rev: Latest

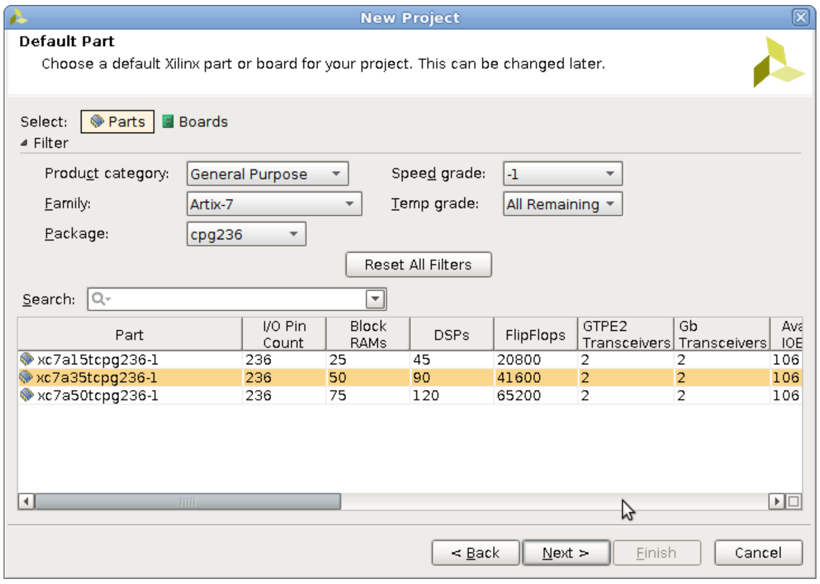
Search:

Display Name	Vendor	Board Rev	Part	I/O
Basys3	digilentinc.com	C.0	xc7a35tcpg236-1	236

Figure 4. Board selection

Select vendor “digilentinc.com” and display name “Basys3” and board revision “latest”. Part XC7A35TCPG236-1 should be selected as unique option. Click Next, review the briefing and click Finish.

If “digilenting.com” or “Basys3” do not appear as options, selection must be made by device. Select “Parts” at the top.



Default Part

Choose a default Xilinx part or board for your project. This can be changed later.

Select: Parts

Product category: General Purpose

Family: Artix-7

Package: cpg236

Speed grade: -1

Temp grade: All Remaining

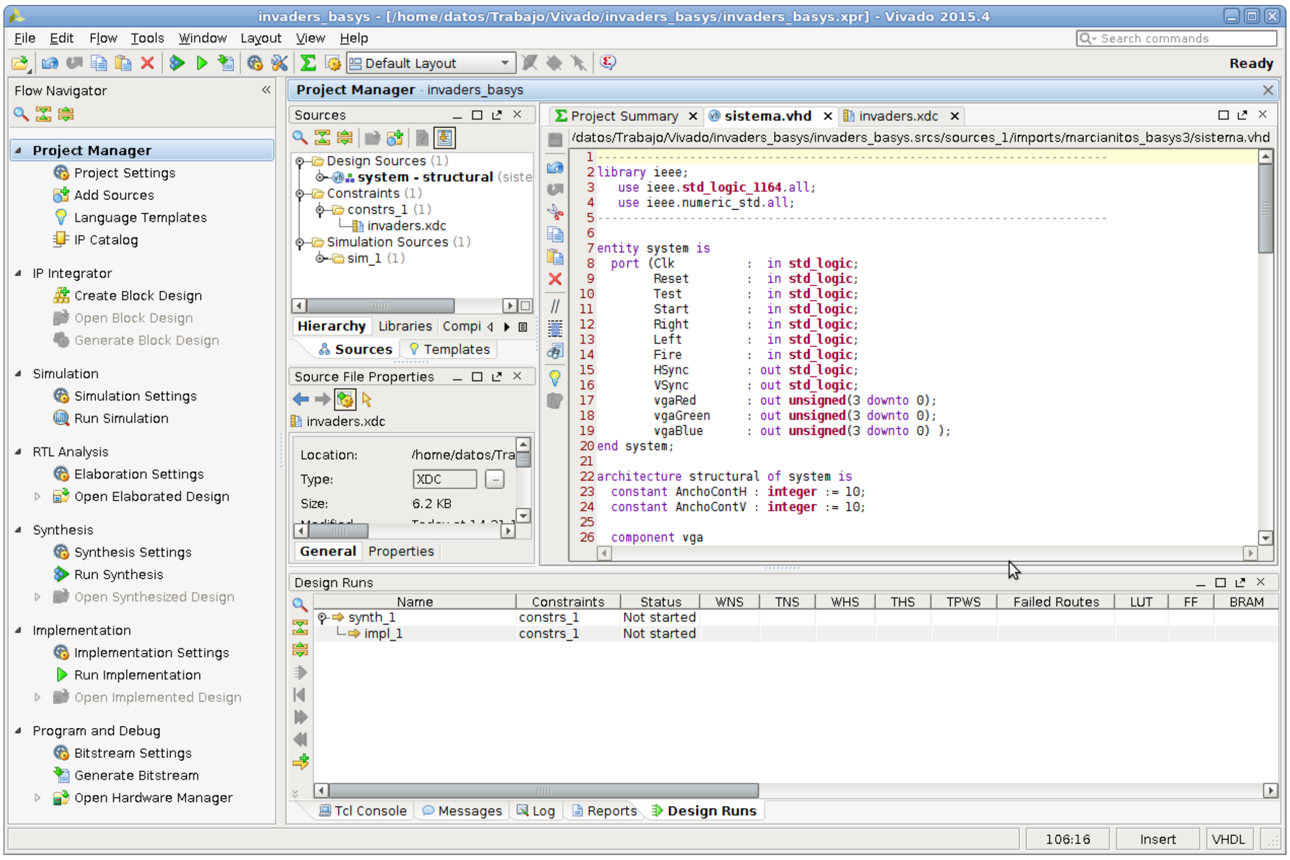
Search:

Part	I/O Pin Count	Block RAMs	DSPs	FlipFlops	GTPE2 Transceivers	Gb Transceivers	Ava IOE
xc7a15tcpg236-1	236	25	45	20800	2	2	106
xc7a35tcpg236-1	236	50	90	41600	2	2	106
xc7a50tcpg236-1	236	75	120	65200	2	2	106

Figure 5. Device selection (alternative to board selection)

Then select product category “General Purpose”, family “Artix-7”, “package cpg236” and speed grade “-1”. From the options available, select “XC7A35TCPG236-1” device. Click Next, review the briefing and click Finish.

After creating the project, the project window will open.



```
-------------------------------------------------------------------
library ieee;
   use ieee.std_logic_1164.all;
   use ieee.numeric_std.all;
-------------------------------------------------------------------

entity system is
  port (Clk          :  in std_logic;
        Reset        :  in std_logic;
        Test         :  in std_logic;
        Start        :  in std_logic;
        Right        :  in std_logic;
        Left         :  in std_logic;
        Fire         :  in std_logic;
        HSync        : out std_logic;
        VSync        : out std_logic;
        vgaRed       : out unsigned(3 downto 0);
        vgaGreen     : out unsigned(3 downto 0);
        vgaBlue      : out unsigned(3 downto 0) );
end system;

architecture structural of system is
  constant AnchoContH : integer := 10;
  constant AnchoContV : integer := 10;

  component vga
```

Name	Constraints	Status	WNS	TNS	WHS	THS	TPWS	Failed Routes	LUT	FF	BRAM
synth_1	constrs_1	Not started									
impl_1	constrs_1	Not started									

Figure 6. Project manager window

The project window is composed of menu bar, toolbar, flow navigator (left) and the main window.

Flow navigator shows the different design stages to obtain an FPGA configuration file (bitstream). The stages are:

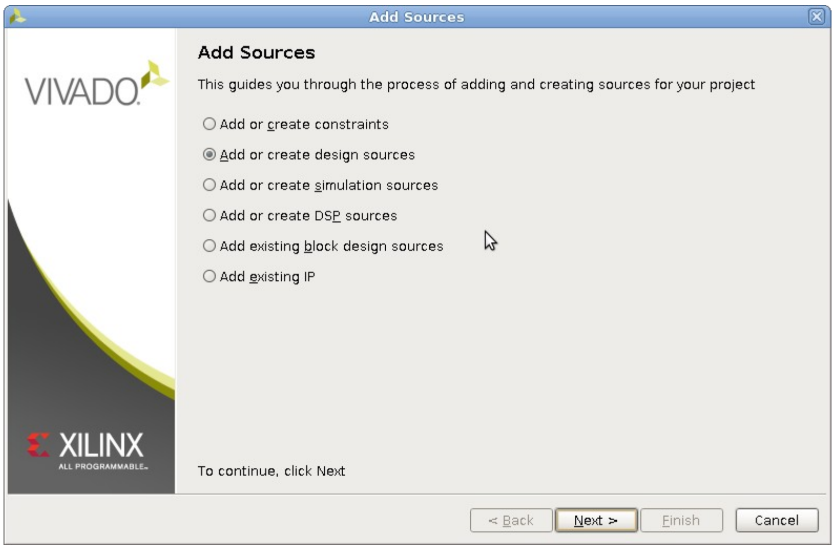
- Project Manager: add and manage source files.
- IP integrator: manage IPs, not used in this tutorial.
- Simulation: manage simulations
- RTL analysis: source code compilation and analysis.
- Synthesis: transform VHDL in a netlist
- Implementation: allocate cells and interconnect
- Program and debug: generate configuration file and download it to FPGA.

Depending on the stage selected in Flow navigator, menus, toolbars and windows change accordingly.

Add VHDL and constraint files

This tutorial assumes you have written your own VHDL files and XDC (constraint) files, or they have been provided elsewhere. For simulation, you need VHDL files describing both the circuit and the testbench. For synthesis, you need VHDL files describing the circuit and an XDC file.

Click on “Add Sources” under “Project manager” flow section.



Add Sources

This guides you through the process of adding and creating sources for your project

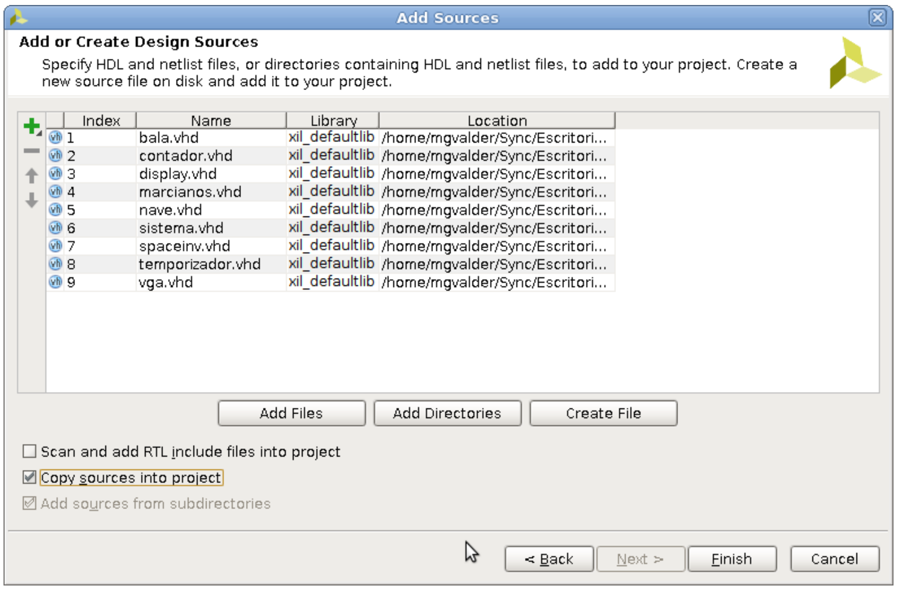
- Add or create constraints
- [✓ Add or create design sources]
- Add or create simulation sources
- Add or create DSP sources
- Add existing block design sources
- Add existing IP

To continue, click Next

Figure 7. Add sources dialog

Select “Add or create design sources”, click Next and use next window to add VHDL files (.vhd extension) to the design. Activate “Copy sources into project” if you want your files copied into the project. If you do not activate this option, files will be linked.

If you have VHDL files describing testbenches, you must repeat the process selecting “Add or create simulation sources”.



Add or Create Design Sources

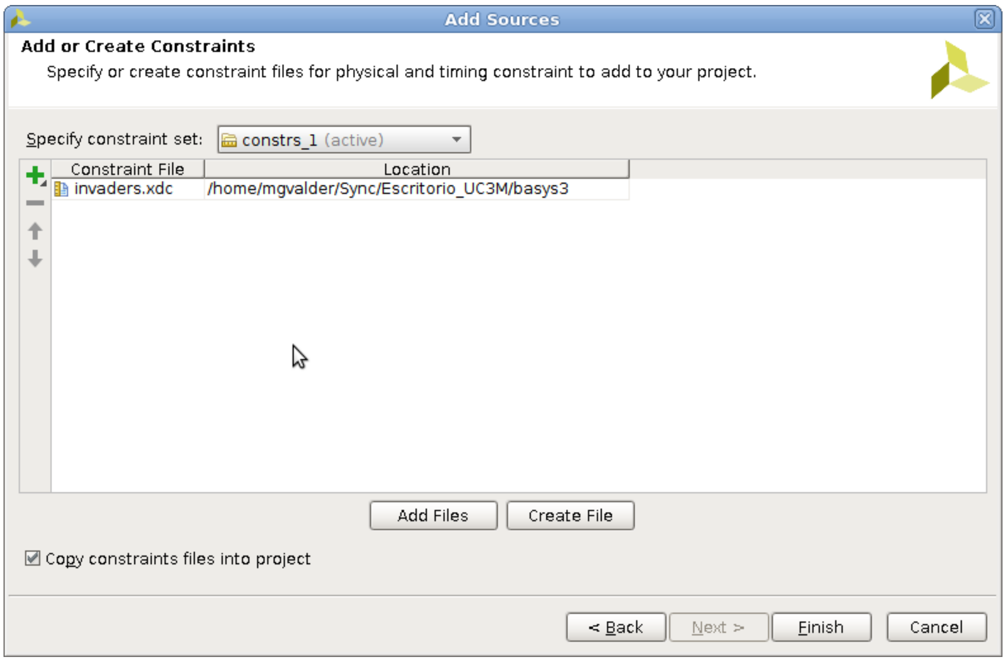
Specify HDL and netlist files, or directories containing HDL and netlist files, to add to your project. Create a new source file on disk and add it to your project.

Index	Name	Library	Location
1	bala.vhd	xil_defaultlib	/home/mgvalder/Sync/Escritori...
2	contador.vhd	xil_defaultlib	/home/mgvalder/Sync/Escritori...
3	display.vhd	xil_defaultlib	/home/mgvalder/Sync/Escritori...
4	marcianos.vhd	xil_defaultlib	/home/mgvalder/Sync/Escritori...
5	nave.vhd	xil_defaultlib	/home/mgvalder/Sync/Escritori...
6	sistema.vhd	xil_defaultlib	/home/mgvalder/Sync/Escritori...
7	spaceinv.vhd	xil_defaultlib	/home/mgvalder/Sync/Escritori...
8	temporizador.vhd	xil_defaultlib	/home/mgvalder/Sync/Escritori...
9	vga.vhd	xil_defaultlib	/home/mgvalder/Sync/Escritori...

- ☐ Scan and add RTL include files into project
- ☑ Copy sources into project
- ☑ Add sources from subdirectories

Figure 8. Add or Create Design Sources dialog

Follow the same process to add the XDC constraints file (.xdc file extension). At the "Add sources dialog", select "Add or create constraints" and then select the .xdc file. XDC file is required for synthesis, but not for simulation.


Add Sources

Add or Create Constraints

Specify or create constraint files for physical and timing constraint to add to your project.

Specify constraint set: constrs_1 (active)

Constraint File	Location
invaders.xdc	/home/mgvalder/Sync/Escritorio_UC3M/basys3

☑ Copy constraints files into project

Figure 9. Add or Create Contraints dialog

Constraints file (XDC file) contains information about the pin location in the FPGA of design inputs and outputs, and information about the clock configuration.

To configure a pin location, the following format is used (indentation is cosmetic):

```
##Buttons
set_property PACKAGE_PIN U18 [get_ports btnC]
  set_property IOSTANDARD LVCMOS33 [get_ports btnC]
```

This entry specify "U18" as the pin location for design port "btnC". In addition, it states the I/O standard for this port as LVCMOS33 (FPGA pin powered to 3.3V, CMOS levels). LVCMOS33 is required for all Basys3 pins.

To specify the clock named "Clk" in the design, the following code is used:

```
## Clock signal
set_property PACKAGE_PIN W5 [get_ports Clk]
  set_property IOSTANDARD LVCMOS33 [get_ports Clk]
  create_clock -add -name Clk100 -period 10.00 -waveform {0 5} [get_ports Clk]
```

"Clk" is defined as located in W5 pin, LVCMOS33 IO standard and 100MHz frequency with a square waveform. "Clk100" is a clock name, just in case there are several clocks. Other tools (timing reports, for example) will use this name to refer to this clock.

Be careful with syntax. Errors in XDC file throw error messages quite difficult to debug.

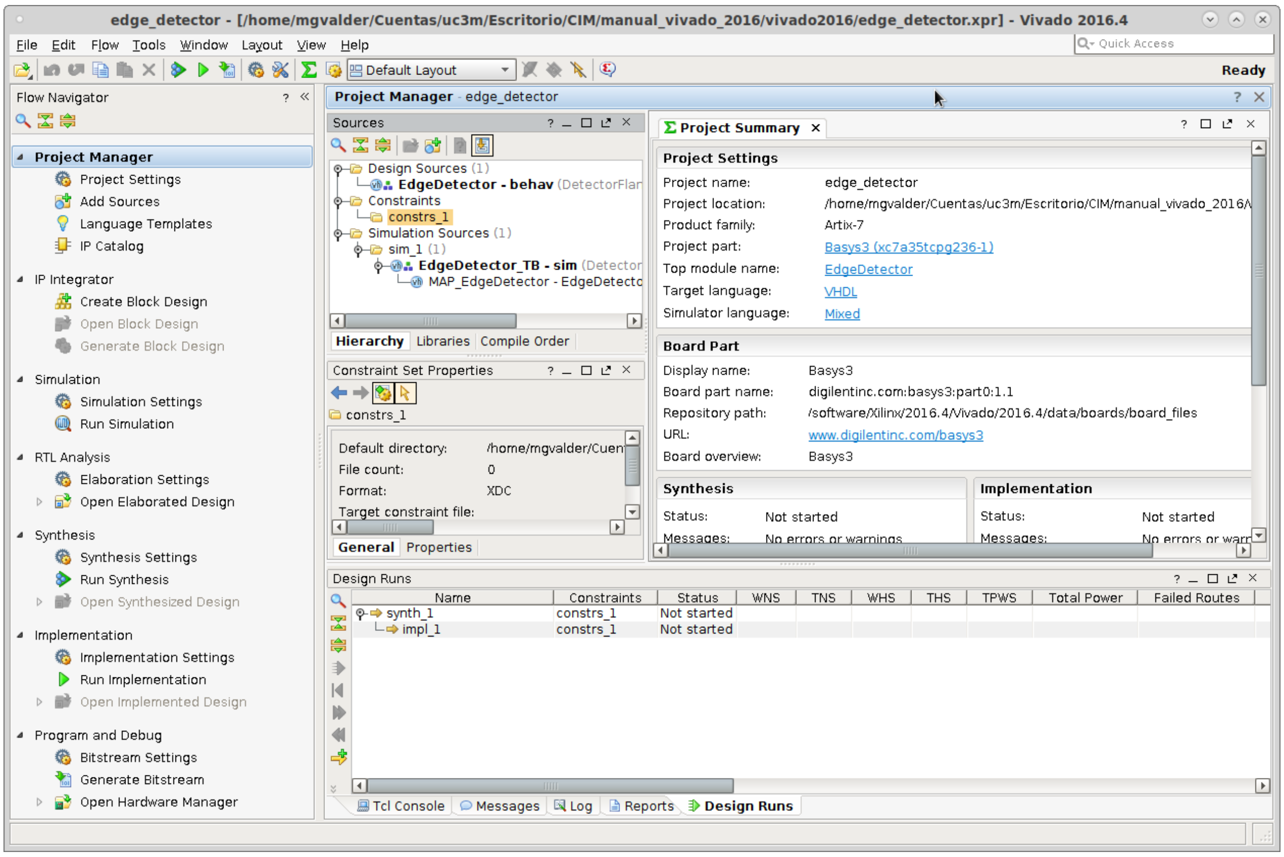
You can find a XDC template for Basys3 board at:

https://github.com/Digilent/Basys3/tree/master/Resources/XDC

Simulation

In this stage, different simulations can be performed. We will run a behavioral simulation, using VHDL files containing circuit and testbench descriptions.

Once VHDL files have been added to the project, Project Manager should show design VHDL files (under Design Sources) and simulation VHDL files (under Simulation Sources). Design files should appear also in the simulation sources section. Files can be moved to simulation or design sources by right clicking them and selection the appropriate "Move to..." option.



edge_detector - [/home/mgvalder/Cuentas/uc3m/Escritorio/CIM/manual_vivado_2016/vivado2016/edge_detector.xpr] - Vivado 2016.4

Project Manager - edge_detector

Sources

```
Design Sources (1)
  └ EdgeDetector - behav (DetectorFlan
Constraints
  └ constrs_1
Simulation Sources (1)
  sim_1 (1)
    EdgeDetector_TB - sim (Detector
      └ MAP_EdgeDetector - EdgeDetecto
```

Constraint Set Properties

constrs_1

Default directory:	/home/mgvalder/Cuen
File count:	0
Format:	XDC
Target constraint file:	

Project Summary

Project Settings

Project name:	edge_detector
Project location:	/home/mgvalder/Cuentas/uc3m/Escritorio/CIM/manual_vivado_2016/
Product family:	Artix-7
Project part:	Basys3 (xc7a35tcpg236-1)
Top module name:	EdgeDetector
Target language:	VHDL
Simulator language:	Mixed

Board Part

Display name:	Basys3
Board part name:	digilentinc.com:basys3:part0:1.1
Repository path:	/software/Xilinx/2016.4/Vivado/2016.4/data/boards/board_files
URL:	www.digilentinc.com/basys3
Board overview:	Basys3

Synthesis

Status:	Not started
Messages:	No errors or warnings

Implementation

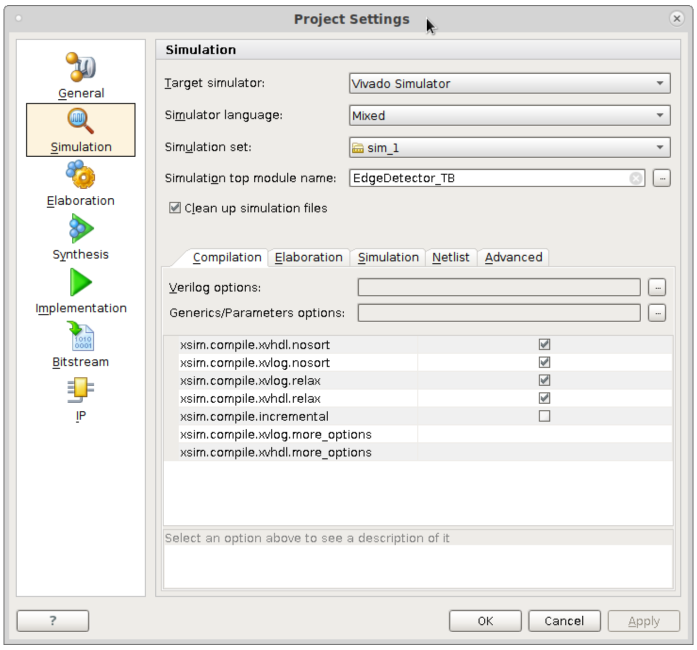
Status:	Not started
Messages:	No errors or warr

Design Runs

Name	Constraints	Status	WNS	TNS	WHS	THS	TPWS	Total Power	Failed Routes
synth_1	constrs_1	Not started							
impl_1	constrs_1	Not started							

Figure 10. Project Manager view with simulation files

Before starting simulation, it must be configured. Select Simulation → Simulation Settings.



Project Settings

Simulation

Target simulator:	Vivado Simulator
Simulator language:	Mixed
Simulation set:	sim_1
Simulation top module name:	EdgeDetector_TB

☑ Clean up simulation files

Compilation | Elaboration | Simulation | Netlist | Advanced

Verilog options:	
Generics/Parameters options:	

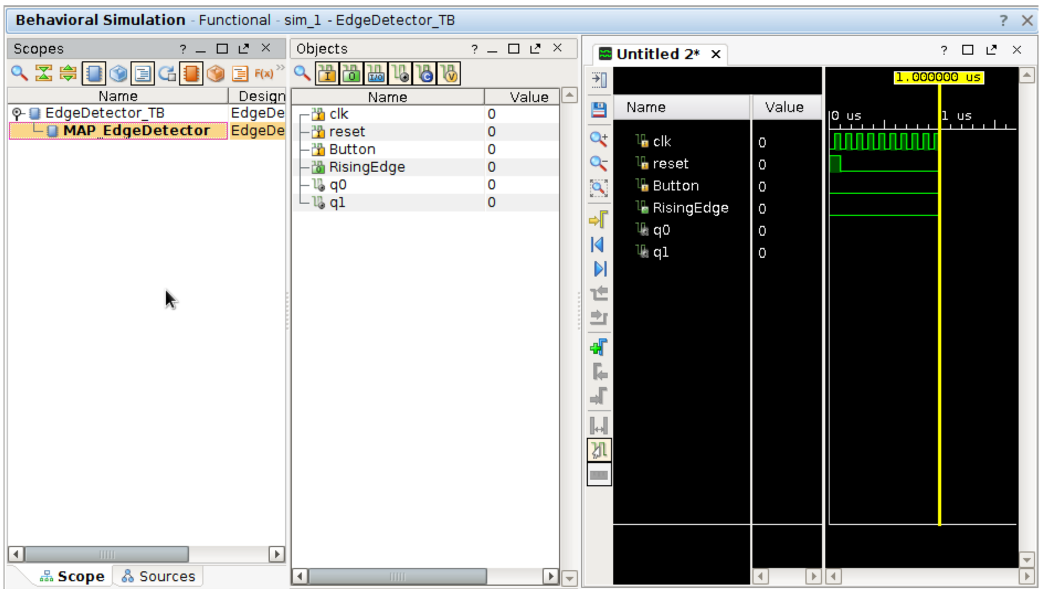
xsim.compile.xvhdl.nosort	☑
xsim.compile.xvlog.nosort	☑
xsim.compile.xvlog.relax	☑
xsim.compile.xvhdl.relax	☑
xsim.compile.incremental	☐
xsim.compile.xvlog.more_options	
xsim.compile.xvhdl.more_options	

Select an option above to see a description of it

Figure 11. Project setting: simulations

Check "Target simulator" is Vivado Simulator, "Simulator language" is VHDL and "Simulation top module name " is the entity name of your testbench. Push OK button.

Then, you can start the simulator. Select Simulation → Run Simulation → Run Behavioral Simulation. Simulation panel will open, including Scopes and Objects tabs and waveform windows.



Behavioral Simulation - Functional - sim_1 - EdgeDetector_TB

Scopes

Name	Design
EdgeDetector_TB	EdgeDe
MAP_EdgeDetector	EdgeDe

Objects

Name	Value
clk	0
reset	0
Button	0
RisingEdge	0
q0	0
q1	0

Untitled 2*

Name	Value
clk	0
reset	0
Button	0
RisingEdge	0
q0	0
q1	0

Figure 12. Behavioral simulation window with docked waveform window

By default, simulation windows will show signals in the testbech. Although we can perform the simulation with this signals, they are not the real circuit inputs and outputs. To selects the circuit real signals, right click the circuit name in *Scopes* and select “Add to wave window”. You can also right click any item in Scopes or Objects windows and add it to wave window.

Usually, is is more comfortable to un-dock the waveform window.

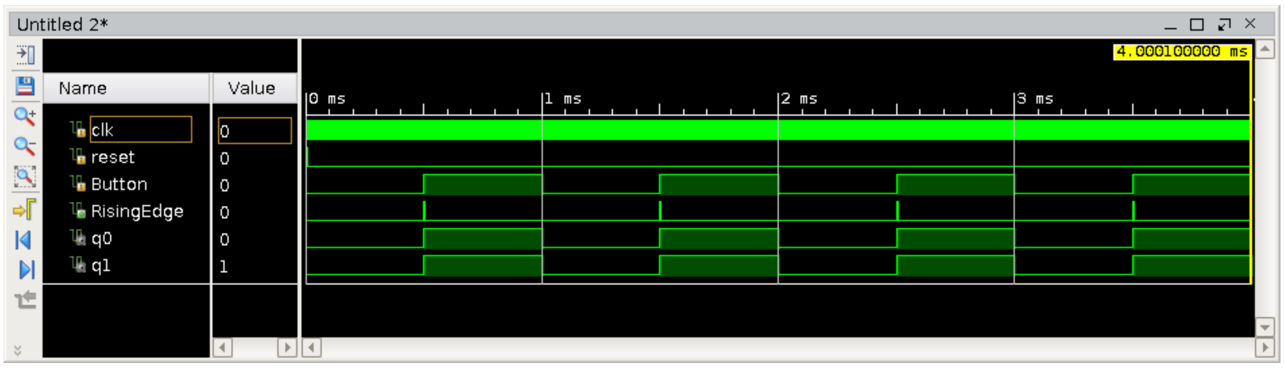


Figure 13. Waveform window (undocked)

Most useful actions to perform in simulation are:

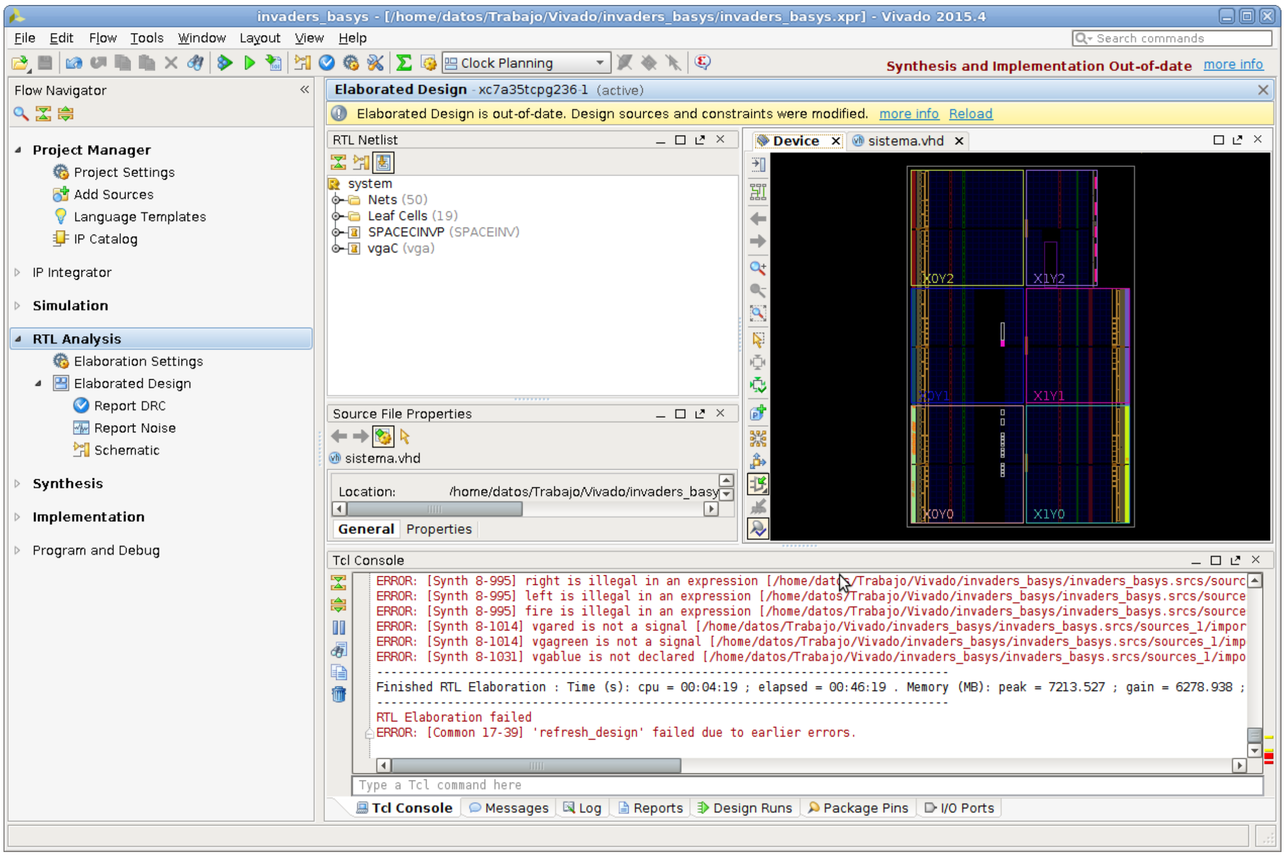
- Restart: go back to time 0 ns. Useful when adding new signals to the waveform window. It does not reload source files.
- Run All: runs the simulation indefinitely. The simulation will stop when a failing assertion is found or no more events are generated. Use with caution, as if one of this conditions is not met, the simulation will run forever.
- Run for X ns. Run simulation for the specified time.
- Relaunch simulation. Restart simulation, compiling sources again. This command must be used if source files have changed.

RTL Analysis

In this stage, inserted code files (VHDL and XDC) are analysed for syntax errors.

Click on *Elaborated Design* under *RTL Analysis* to analyse the source code. Any error related to the design will be reported in the log window (bottom). When correcting errors, be sure to scroll up the log window and start with the first error. Subsequent errors might be caused by previous ones.

If RTL Analysis has already been run and source code is modified afterwards a “Reload” option will appear at the window top, below the toolbar.



Elaborated Design is out-of-date. Design sources and constraints were modified. more info Reload

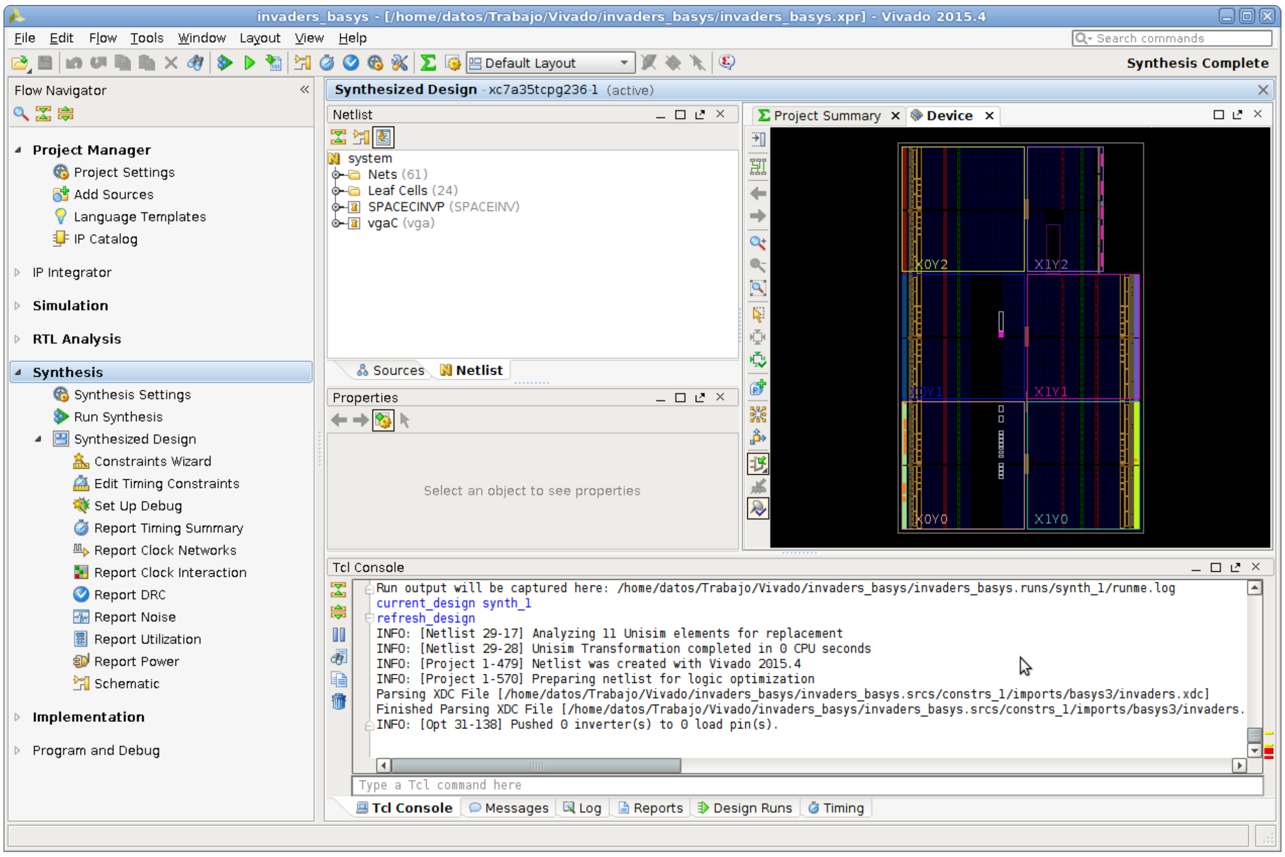
```
ERROR: [Synth 8-995] right is illegal in an expression [/home/datos/Trabajo/Vivado/invaders_basys/invaders_basys.srcs/sourc
ERROR: [Synth 8-995] left is illegal in an expression [/home/datos/Trabajo/Vivado/invaders_basys/invaders_basys.srcs/source
ERROR: [Synth 8-995] fire is illegal in an expression [/home/datos/Trabajo/Vivado/invaders_basys/invaders_basys.srcs/source
ERROR: [Synth 8-1014] vgared is not a signal [/home/datos/Trabajo/Vivado/invaders_basys/invaders_basys.srcs/sources_1/impor
ERROR: [Synth 8-1014] vgagreen is not a signal [/home/datos/Trabajo/Vivado/invaders_basys/invaders_basys.srcs/sources_1/imp
ERROR: [Synth 8-1031] vgablue is not declared [/home/datos/Trabajo/Vivado/invaders_basys/invaders_basys.srcs/sources_1/impo
---------------------------------------------------------------------------------
Finished RTL Elaboration : Time (s): cpu = 00:04:19 ; elapsed = 00:46:19 . Memory (MB): peak = 7213.527 ; gain = 6278.938 ;
---------------------------------------------------------------------------------
RTL Elaboration failed
ERROR: [Common 17-39] 'refresh_design' failed due to earlier errors.
```

Figure 14. RTL Analysis view

Synthesis and Implementation

Once RTL Analysis has been run without errors, synthesis can take place. Synthesis is the process of transforming HDL code into a netlist. A netlist consists in cells (mainly gates and flip-flops) and their interconnections.

Synthesis is performed by selecting "Run Synthesis". After synthesis, some reports can be reviewed under the "Synthesized design" entry. Synthesis reports are approximate because Implementation is not yet performed. Post-implementation reports are more accurate, in general.



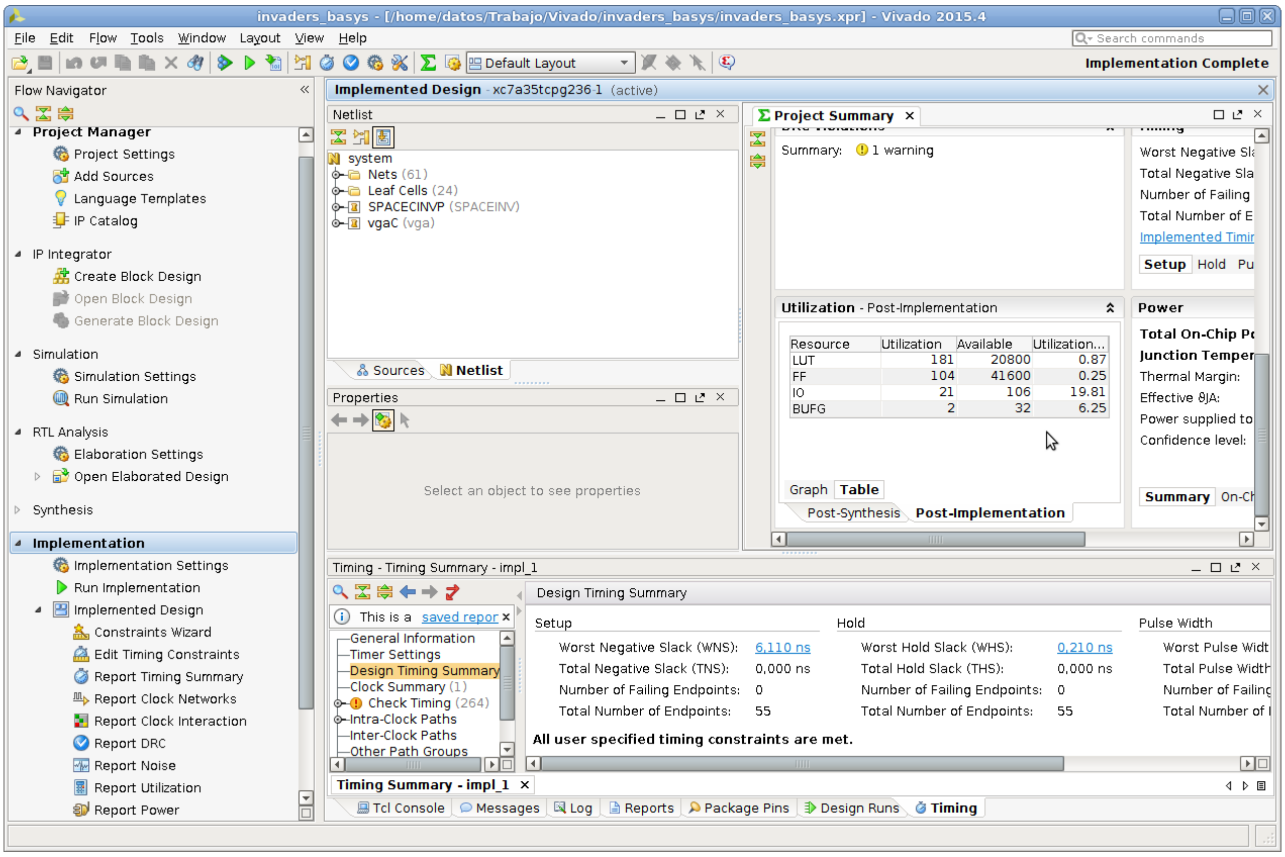
```
Run output will be captured here: /home/datos/Trabajo/Vivado/invaders_basys/invaders_basys.runs/synth_1/runme.log
current_design synth_1
refresh_design
INFO: [Netlist 29-17] Analyzing 11 Unisim elements for replacement
INFO: [Netlist 29-28] Unisim Transformation completed in 0 CPU seconds
INFO: [Project 1-479] Netlist was created with Vivado 2015.4
INFO: [Project 1-570] Preparing netlist for logic optimization
Parsing XDC File [/home/datos/Trabajo/Vivado/invaders_basys/invaders_basys.srcs/constrs_1/imports/basys3/invaders.xdc]
Finished Parsing XDC File [/home/datos/Trabajo/Vivado/invaders_basys/invaders_basys.srcs/constrs_1/imports/basys3/invaders.
INFO: [Opt 31-138] Pushed 0 inverter(s) to 0 load pin(s).
```

Figure 15. Synthesis view

Implementation is the process of allocating every cell in the netlist to real resources in the selected FPGA. While the synthesis process is common to a whole FPGA family, *Implementation* is specific to every device.

Implementation is performed by selecting "*Run Implementation*". After implementation, area, timing and power reports can be reviewed.

A resource usage briefing is shown in the project summary. A post-synthesis and post-implementation summary are shown under the Project Summary window. Graphical or table representation can be selected. Post-implementation information is more accurate, if available.


invaders_basys - [/home/datos/Trabajo/Vivado/invaders_basys/invaders_basys.xpr] - Vivado 2015.4

Implementation Complete

Implemented Design - xc7a35tcpg236-1 (active)

Netlist: system; Nets (61); Leaf Cells (24); SPACECINVP (SPACEINV); vgaC (vga)

Properties: Select an object to see properties

Project Summary — Summary: 1 warning

Utilization - Post-Implementation

Resource	Utilization	Available	Utilization...
LUT	181	20800	0.87
FF	104	41600	0.25
IO	21	106	19.81
BUFG	2	32	6.25

Timing - Timing Summary - impl_1

Design Timing Summary

Setup		Hold		Pulse Width
Worst Negative Slack (WNS):	6,110 ns	Worst Hold Slack (WHS):	0,210 ns	Worst Pulse Widt
Total Negative Slack (TNS):	0,000 ns	Total Hold Slack (THS):	0,000 ns	Total Pulse Width
Number of Failing Endpoints:	0	Number of Failing Endpoints:	0	Number of Failing
Total Number of Endpoints:	55	Total Number of Endpoints:	55	Total Number of I

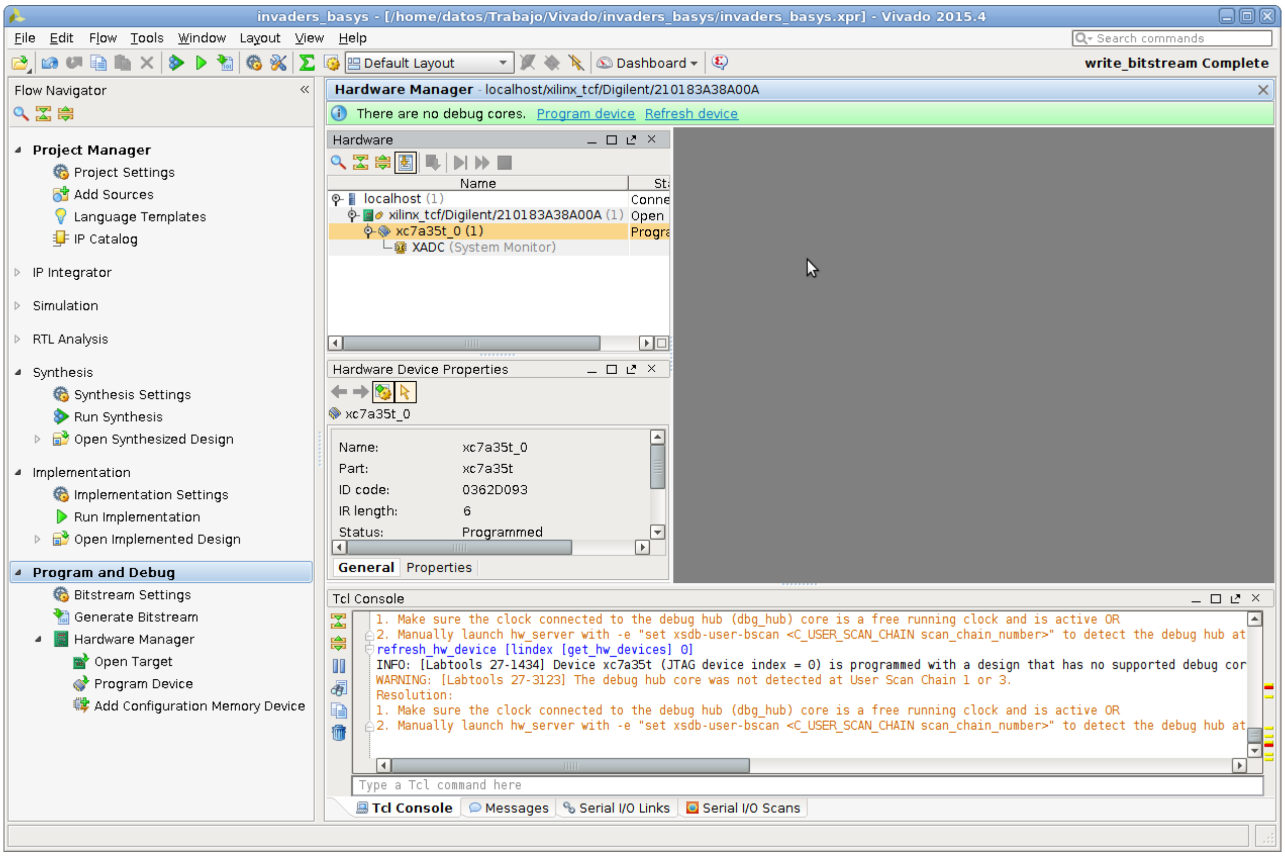
All user specified timing constraints are met.

Figure 16. Implementation view with resource utilization information

Device programming and Debug

The last step is to build the file used to configure the FPGA, called Bitstream, and download it to the Basys3 board. The bitstream is generated by selecting "Generate Bitstream" under "Program and Debug flow".

Once the bistream has been generated, connect the Basys3 board to a PC USB port and switch it on. Then, select "Open Target", "Auto Connect". The tool should connect to the board, showing the available devices.

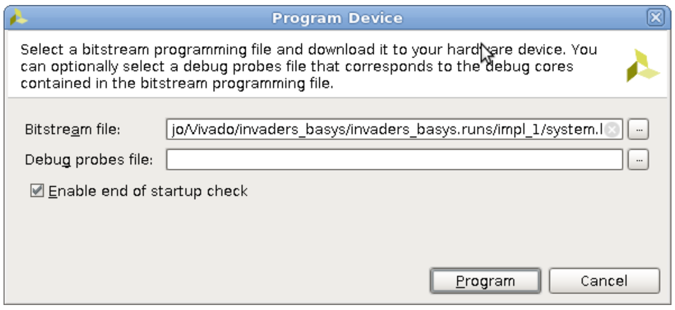


Name:	xc7a35t_0
Part:	xc7a35t
ID code:	0362D093
IR length:	6
Status:	Programmed

```
1. Make sure the clock connected to the debug hub (dbg_hub) core is a free running clock and is active OR
2. Manually launch hw_server with -e "set xsdb-user-bscan <C_USER_SCAN_CHAIN scan_chain_number>" to detect the debug hub at
refresh_hw_device [lindex [get_hw_devices] 0]
INFO: [Labtools 27-1434] Device xc7a35t (JTAG device index = 0) is programmed with a design that has no supported debug cor
WARNING: [Labtools 27-3123] The debug hub core was not detected at User Scan Chain 1 or 3.
Resolution:
1. Make sure the clock connected to the debug hub (dbg_hub) core is a free running clock and is active OR
2. Manually launch hw_server with -e "set xsdb-user-bscan <C_USER_SCAN_CHAIN scan_chain_number>" to detect the debug hub at
```

Figure 17. Program and Debug view

In this case, the device in the board should be a "xc7a35t". Select "Program Device" to download the bitstream to the device. A dialog will show the .bit file to download.



Select a bitstream programming file and download it to your hardware device. You can optionally select a debug probes file that corresponds to the debug cores contained in the bitstream programming file.

Bitstream file:	jo/Vivado/invaders_basys/invaders_basys.runs/impl_1/system.l
Debug probes file:	

☑ Enable end of startup check

Figure 18. Program device dialog

After the downloading process, the “Done” green LED in the board (top right) should turn on. The FPGA is configured and running.

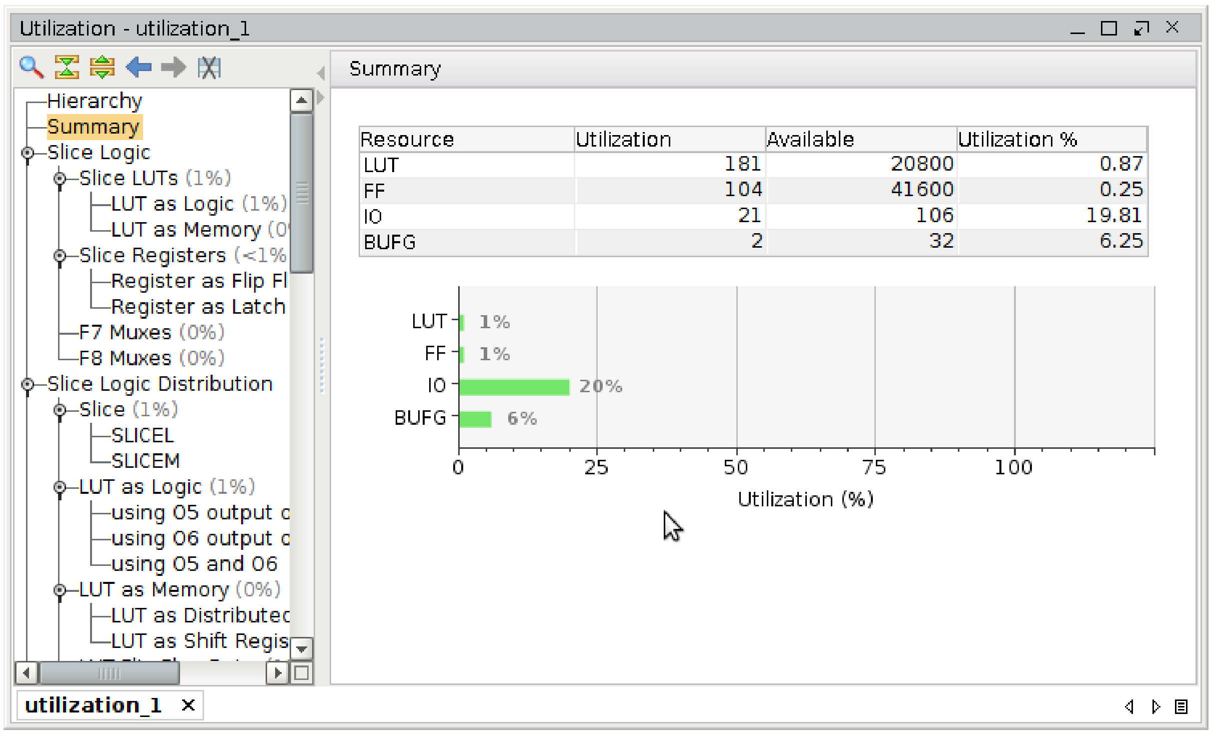
Area and timing reports

It is important to obtain feedback information about the synthesis and implementation processes. In particular, we are interested in area and timing related information.

Regarding area, we want to know the total amount of resources used by our design: flip-flops, LUTS (combinational functions), memories, IO cells, etc.

Regarding timing, we want to make sure our design can run with the 100MHz clock provided by the board.

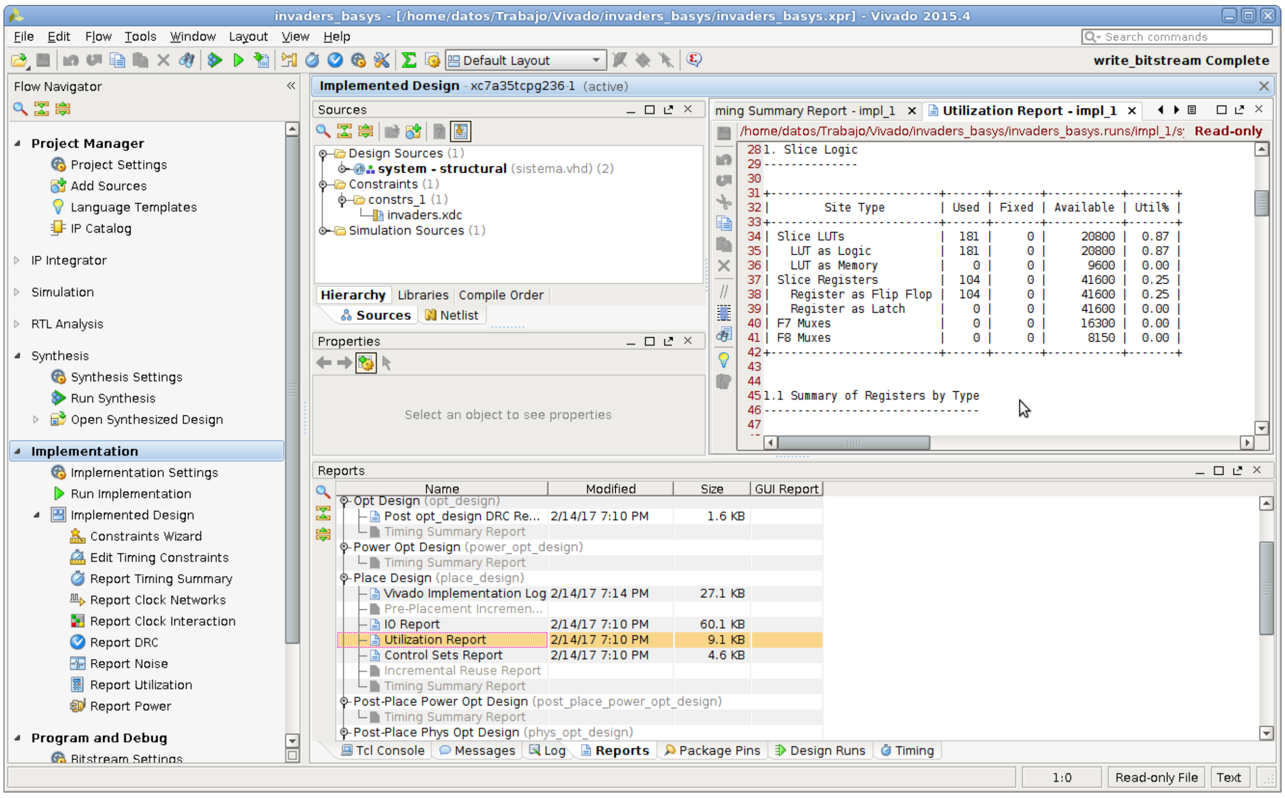
Used area is best reported by selecting “Report Utilization” under Implementation flow. Results will be shown in a window with detailed information of all used resources.



Resource	Utilization	Available	Utilization %
LUT	181	20800	0.87
FF	104	41600	0.25
IO	21	106	19.81
BUFG	2	32	6.25

Figure 19. Utilization report window

A text based report can be accessed through the *Reports* Window →*Placed Design* → *Utilization Report*.



```
1. Slice Logic
--------------

+-------------------------+------+-------+-----------+-------+
|        Site Type        | Used | Fixed | Available | Util% |
+-------------------------+------+-------+-----------+-------+
| Slice LUTs              |  181 |     0 |     20800 |  0.87 |
|   LUT as Logic          |  181 |     0 |     20800 |  0.87 |
|   LUT as Memory         |    0 |     0 |      9600 |  0.00 |
| Slice Registers         |  104 |     0 |     41600 |  0.25 |
|   Register as Flip Flop |  104 |     0 |     41600 |  0.25 |
|   Register as Latch     |    0 |     0 |     41600 |  0.00 |
| F7 Muxes                |    0 |     0 |     16300 |  0.00 |
| F8 Muxes                |    0 |     0 |      8150 |  0.00 |
+-------------------------+------+-------+-----------+-------+


1.1 Summary of Registers by Type
--------------------------------
```

Name	Modified	Size	GUI Report
Opt Design (opt_design)			
Post opt_design DRC Re...	2/14/17 7:10 PM	1.6 KB	
Timing Summary Report			
Power Opt Design (power_opt_design)			
Timing Summary Report			
Place Design (place_design)			
Vivado Implementation Log	2/14/17 7:14 PM	27.1 KB	
Pre-Placement Incremen...			
IO Report	2/14/17 7:10 PM	60.1 KB	
Utilization Report	2/14/17 7:10 PM	9.1 KB	
Control Sets Report	2/14/17 7:10 PM	4.6 KB	
Incremental Reuse Report			
Timing Summary Report			
Post-Place Power Opt Design (post_place_power_opt_design)			
Timing Summary Report			
Post-Place Phys Opt Design (phys_opt_design)			

Figure 20. Text Utilization report