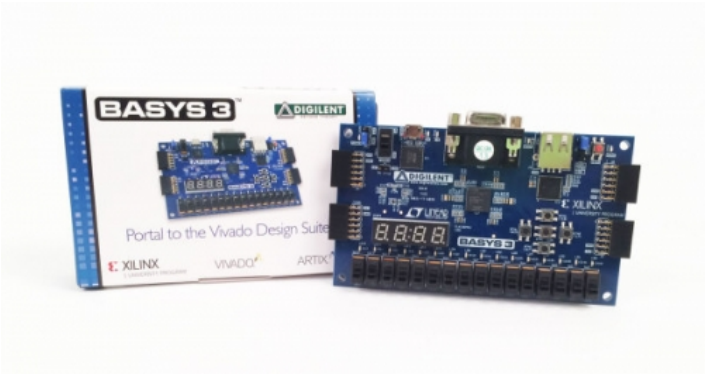
Getting Started with the Basys3



Powering it On

To power on the Basys3, you will need a micro-USB cable. Before you begin, ensure that the jumper on JP1 is in the QSPI position. Plug this cable into the JTAG slot on the Basys3, plug the other end into your computer, and flip the power switch to the **On** position. This will start the out of box demo for the Basys3.

More information on this demo can be found here.

Getting Started with Vivado

This video will give you step by step instructions to get started with your Basys3.

The files needed for this demo can be downloaded by clicking here. You'll also need the constraints file which can be downloaded here.

Getting started with Vivado and Basys3

More information on the Abacus project can be found here.

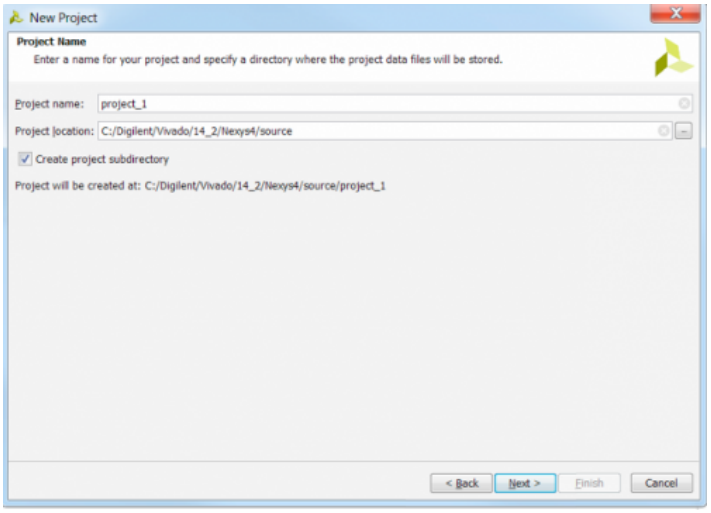
1. Download and Install Vivado

> 1.1) First, install the latest version of Vivado by following our guide here.

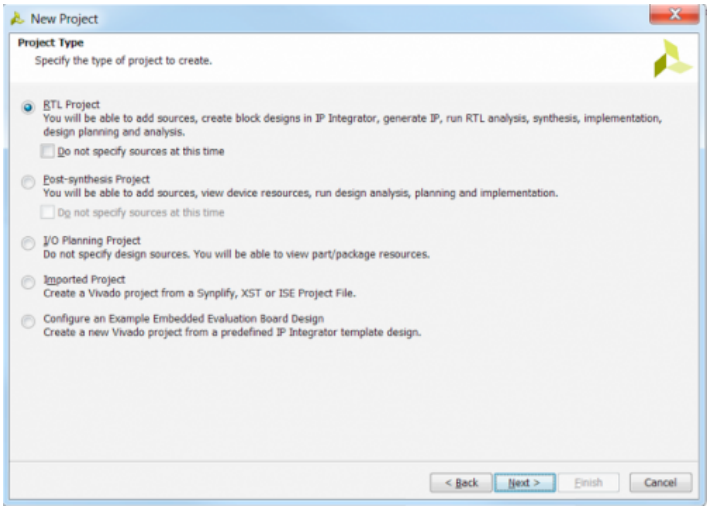
2. Creating a Project

> 2.1) Now that we have Vivado installed, we're going to create a project.

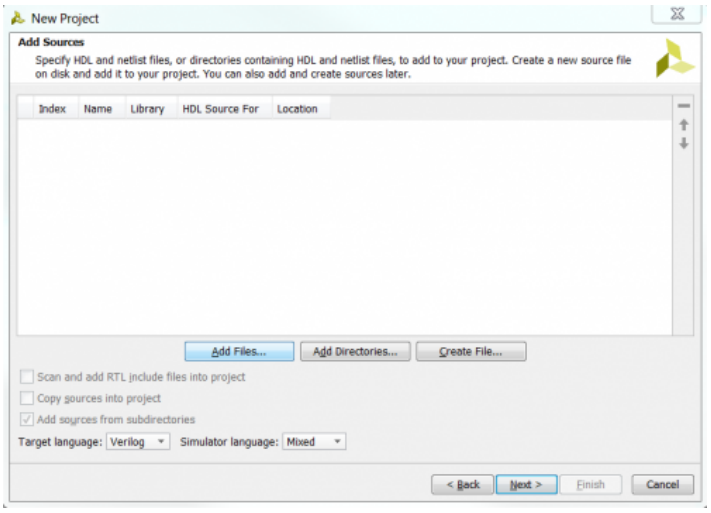
2.2) This opens Vivado's New Project wizard. Click **Next** and you'll see this screen.



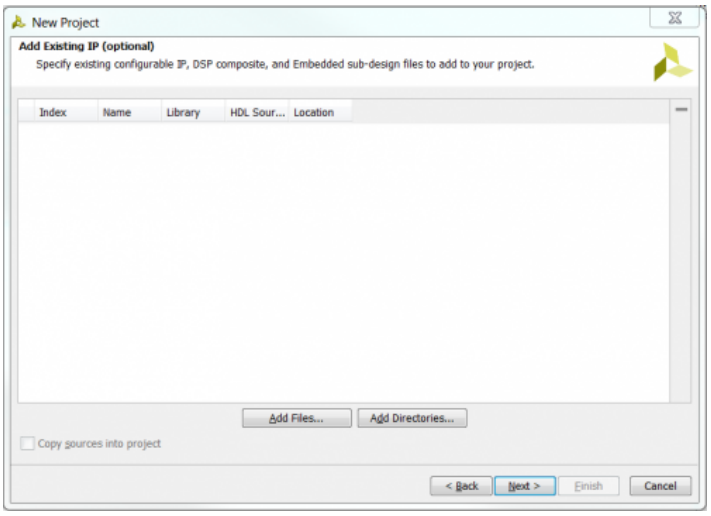
2.3) Name your project (no spaces!) and choose your project saving directory before clicking **Next**. You will now see this screen.



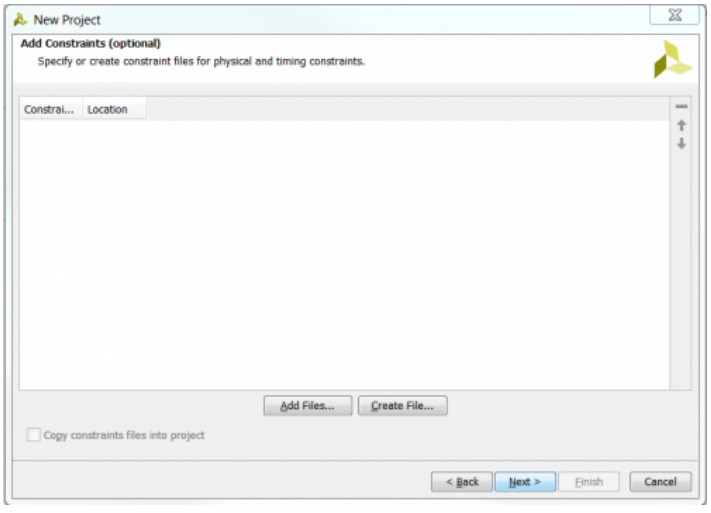
2.4) Select RTL Project and click **Next**. In this window, you can select any other source files or directories that you'll want to use in your projects. We can also select which language we'll be programming in.



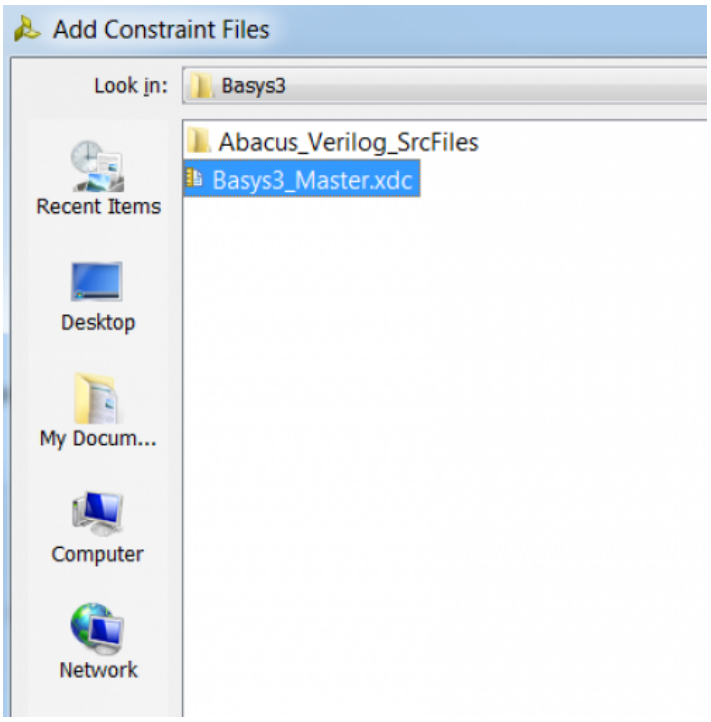
2.5) We'll be importing the pre-built Verilog files so click **Add Files**, navigate to where you saved the project files from before, and select them all. It should be noted that if you check the **Copy sources into project** box, Vivado will create separate copies of these sources and place them within your project directory. Click **Next**. This window lets you choose existing IPs (Intellectual Property) cores if you have them.



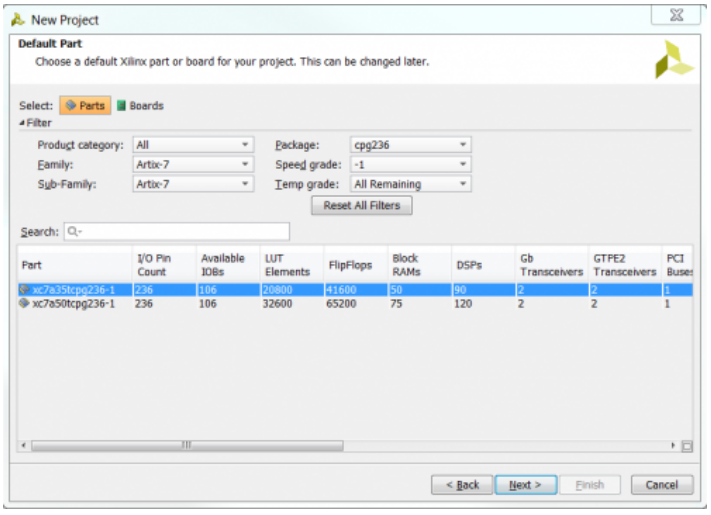
2.6) Click **Next** and you'll see this window.



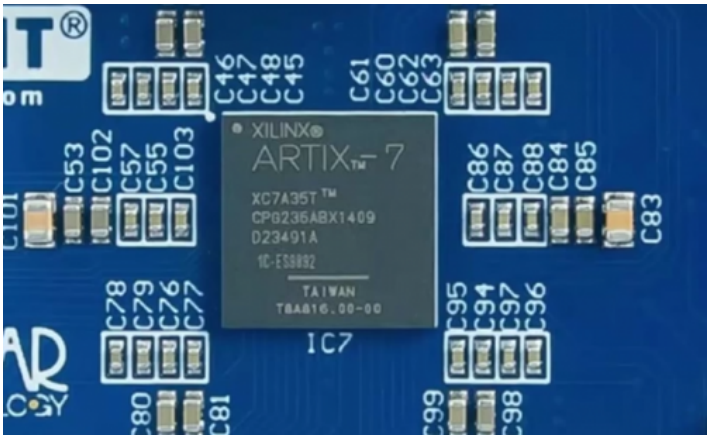
2.7) This is where we'll import our Xilinx Design Constraints file (XDC) to map the HDL signals to the Artix-7 pins. Click on **Add Files**, navigate to where you saved your Basys3_Master.xdc file, select it, and click **Next**.



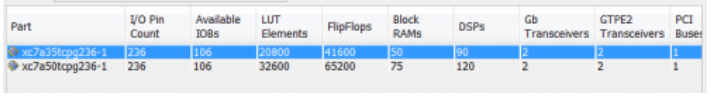
2.8) At this point you'll see the part selection screen.



2.9) You'll find all of the information you need on the Artix-7 chip on your board.



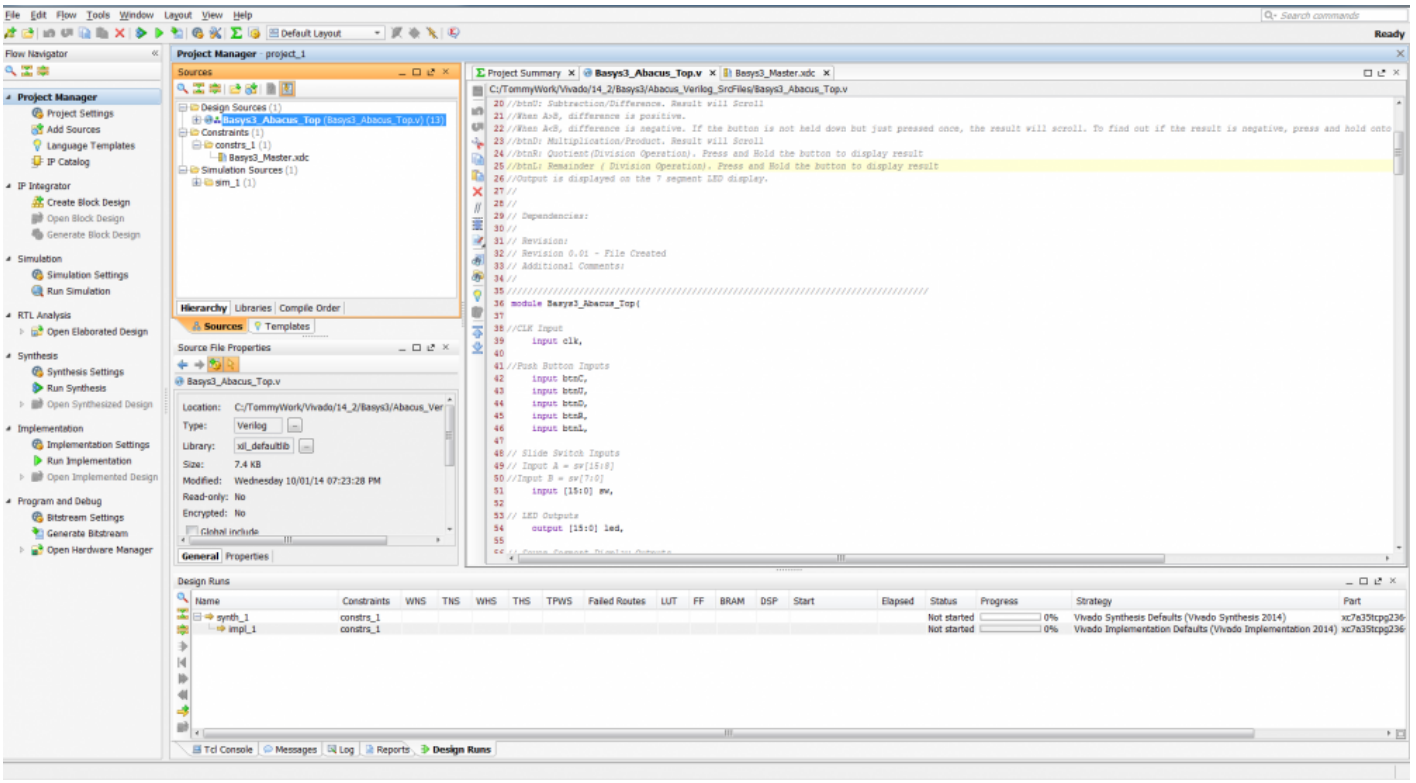
2.10) To find our board set the following filters
– Family: Artix-7
– Sub-Family: Artix-7
– Package: cpg236
– Speed grade: -1
Then choose the top part labeled **xc7a35tcpg236-1** and click **Next** and then **Finish**.



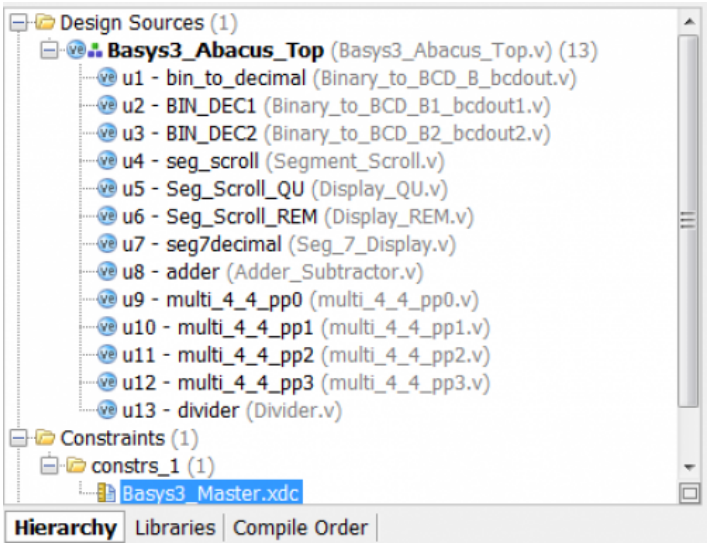
This will create your project and bring you to the Vivado project home page.

3. Working with Vivado

3.1) On the home page, you will see the files that you imported earlier in the *Sources* box.



3.2) Double clicking a file will open it in the window to the right. *Basys3_Abacus_Top* is the top module for the abacus demo we will be running. Clicking the [+] button will reveal the lower level modules used in it.

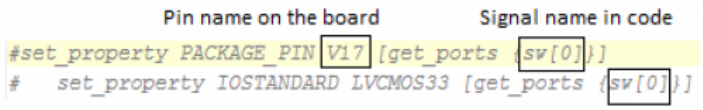


3.3) Before we run our program, we must first map the signals to pins using the Basys3_Master.xdc file we imported. To do this, we will **open** Basys3_Master.xdc. Inside this file, we will see how Vivado maps signals to pins. Each line should be commented out at this point (with the # character), so it should look something like this.

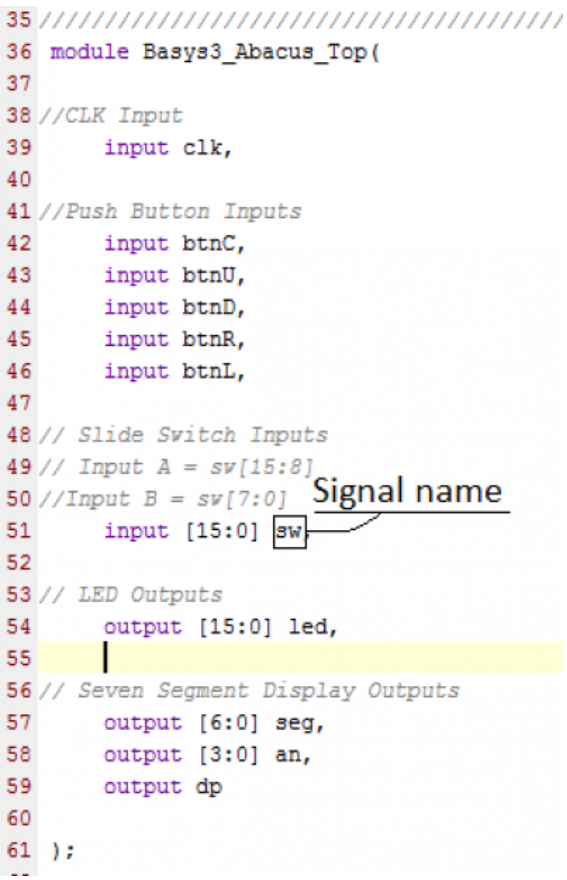
```
#   create_clock -add -name sys_clk_pin -period 10.00 -waveform {0 5} [get_ports

## Switches
#set_property PACKAGE_PIN V17 [get_ports {sw[0]}]
#   set_property IOSTANDARD LVCMOS33 [get_ports {sw[0]}]
#set_property PACKAGE_PIN V16 [get_ports {sw[1]}]
#   set_property IOSTANDARD LVCMOS33 [get_ports {sw[1]}]
#set_property PACKAGE_PIN W16 [get_ports {sw[2]}]
#   set_property IOSTANDARD LVCMOS33 [get_ports {sw[2]}]
#set_property PACKAGE_PIN W17 [get_ports {sw[3]}]
#   set_property IOSTANDARD LVCMOS33 [get_ports {sw[3]}]
#set_property PACKAGE_PIN W15 [get_ports {sw[4]}]
#   set_property IOSTANDARD LVCMOS33 [get_ports {sw[4]}]
#set_property PACKAGE_PIN V15 [get_ports {sw[5]}]
#   set_property IOSTANDARD LVCMOS33 [get_ports {sw[5]}]
#set_property PACKAGE_PIN W14 [get_ports {sw[6]}]
#   set_property IOSTANDARD LVCMOS33 [get_ports {sw[6]}]
#set_property PACKAGE_PIN W13 [get_ports {sw[7]}]
#   set_property IOSTANDARD LVCMOS33 [get_ports {sw[7]}]
#set_property PACKAGE_PIN V2 [get_ports {sw[8]}]
#   set_property IOSTANDARD LVCMOS33 [get_ports {sw[8]}]
#set_property PACKAGE_PIN T3 [get_ports {sw[9]}]
#   set_property IOSTANDARD LVCMOS33 [get_ports {sw[9]}]
#set_property PACKAGE_PIN T2 [get_ports {sw[10]}]
#   set_property IOSTANDARD LVCMOS33 [get_ports {sw[10]}]
#set_property PACKAGE_PIN R3 [get_ports {sw[11]}]
#   set_property IOSTANDARD LVCMOS33 [get_ports {sw[11]}]
#set_property PACKAGE_PIN W2 [get_ports {sw[12]}]
#   set_property IOSTANDARD LVCMOS33 [get_ports {sw[12]}]
#set_property PACKAGE_PIN U1 [get_ports {sw[13]}]
#   set_property IOSTANDARD LVCMOS33 [get_ports {sw[13]}]
#set_property PACKAGE_PIN T1 [get_ports {sw[14]}]
#   set_property IOSTANDARD LVCMOS33 [get_ports {sw[14]}]
#set_property PACKAGE_PIN R2 [get_ports {sw[15]}]
```

3.4) First, we want to make sure our signal names match the ones in the .xdc file. This can be confirmed by comparing the signal name in the XDC file with the signal name in the top module. *These are case sensitive!*

Showing a signal in the XDC file:



Showing the same signal in the top module file:



3.5) Once these are confirmed, we will uncomment whichever constraints we are using in the .xdc file.

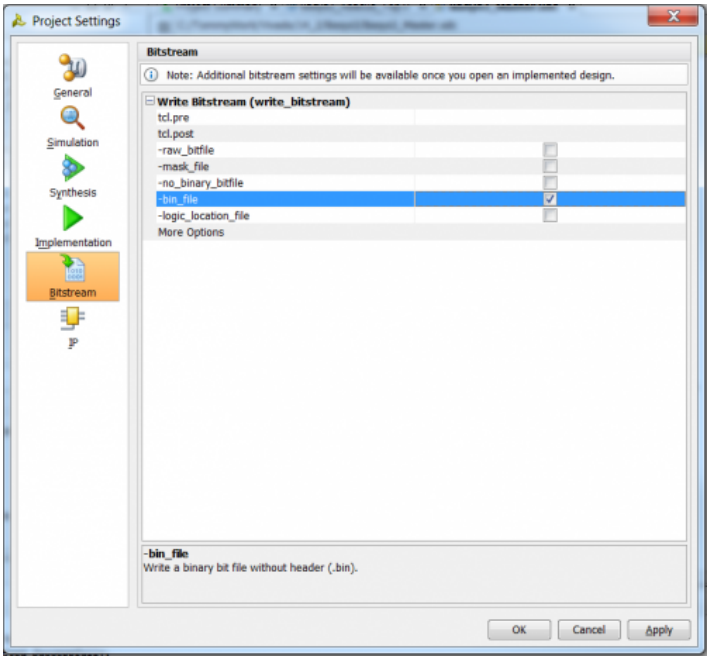
These constraints can be uncommented by selecting the lines of the signals we are using and un-toggling the comments (Ctrl+/). In this case, we are using clk, btnC, btnU, btnD, btnR, btnL, sw[0] through sw[15], led[0] through led[15], seg[0] through seg[6], an[0] through an[3], and dp. Go through the xdc file and **uncomment** the lines corresponding to these signals.

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

# Switches
set_property PACKAGE_PIN V17 [get_ports {sw[0]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {sw[0]}]
set_property PACKAGE_PIN V16 [get_ports {sw[1]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {sw[1]}]
set_property PACKAGE_PIN W16 [get_ports {sw[2]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {sw[2]}]
set_property PACKAGE_PIN W17 [get_ports {sw[3]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {sw[3]}]
set_property PACKAGE_PIN W15 [get_ports {sw[4]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {sw[4]}]
set_property PACKAGE_PIN V15 [get_ports {sw[5]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {sw[5]}]
set_property PACKAGE_PIN W14 [get_ports {sw[6]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {sw[6]}]
set_property PACKAGE_PIN W13 [get_ports {sw[7]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {sw[7]}]
set_property PACKAGE_PIN V2 [get_ports {sw[8]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {sw[8]}]
set_property PACKAGE_PIN T3 [get_ports {sw[9]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {sw[9]}]
set_property PACKAGE_PIN T2 [get_ports {sw[10]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {sw[10]}]
set_property PACKAGE_PIN R3 [get_ports {sw[11]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {sw[11]}]
set_property PACKAGE_PIN W2 [get_ports {sw[12]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {sw[12]}]
set_property PACKAGE_PIN U1 [get_ports {sw[13]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {sw[13]}]
```

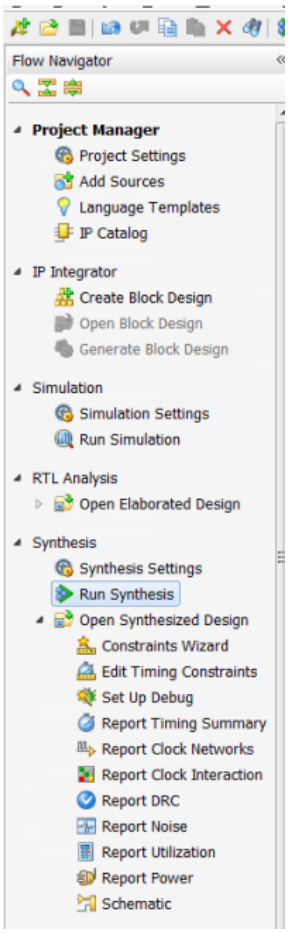
After uncommenting the xdc file, save it and we can start programming your Basys3.

4. Programming the Basys3

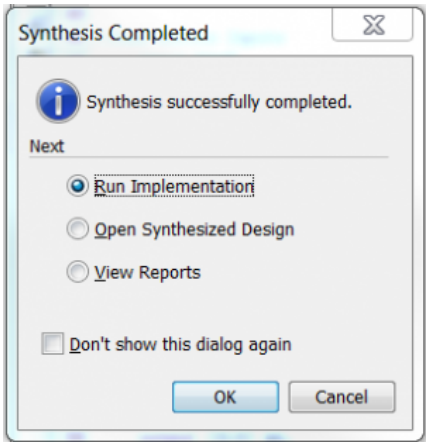
4.1) There are two ways to program your Basys3 FPGA: using a .bit file and using a .bin file. Using a .bit file, we use the JTAG programming cable to load the bit file into the FPGA. Programming with a .bin file will use the QSPI to program the FPGA each time it is powered on. This means you will not have to reprogram it using the JTAG connector each time. We will specify that we want to generate a .bin file by clicking **Tools→Project Settings→Bitstream**. In this window we will check the box next to **bin_file**.



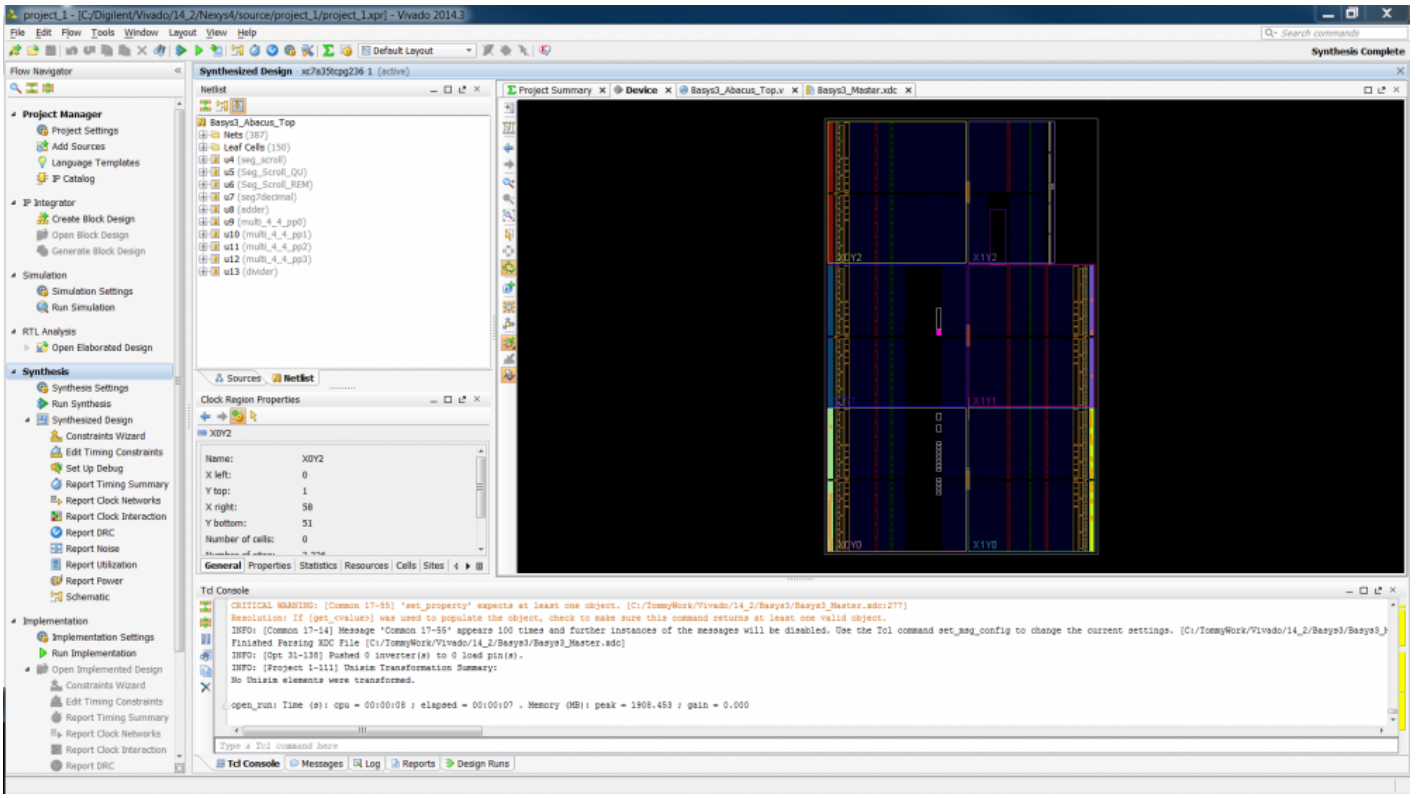
4.2) To begin, we will run the synthesis by clicking **Run Synthesis** beneath Synthesis in the Flow Navigator on the left side of Vivado.



4.3) When Vivado is done synthesizing your project, you will see the Synthesis Completed window.

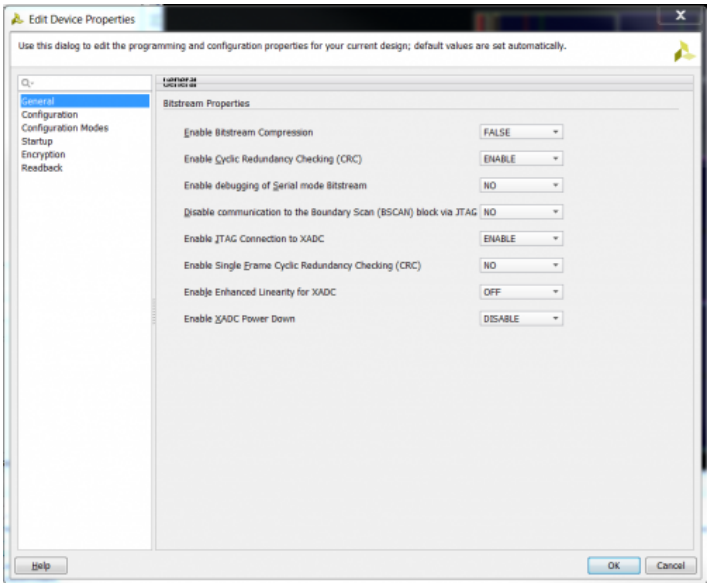


4.4) Click **Open Synthesized Design** and then press **Ok**. You should now see this window

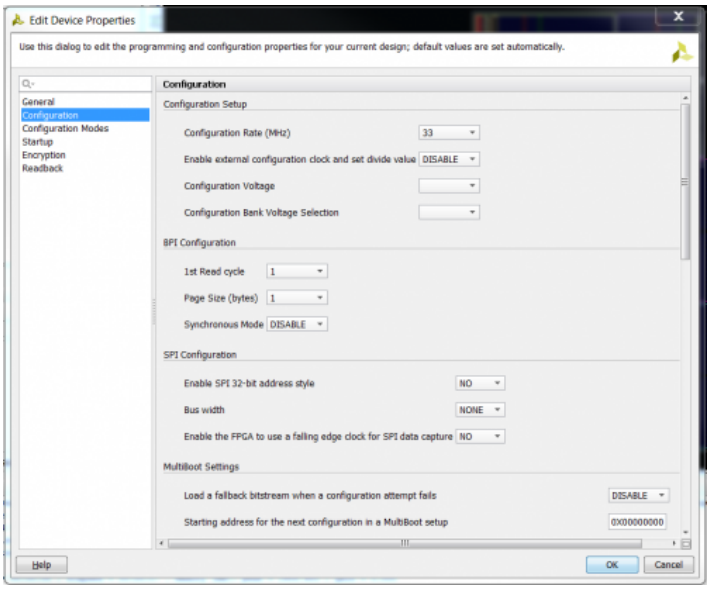


5. Setting the device properties (Recommended)

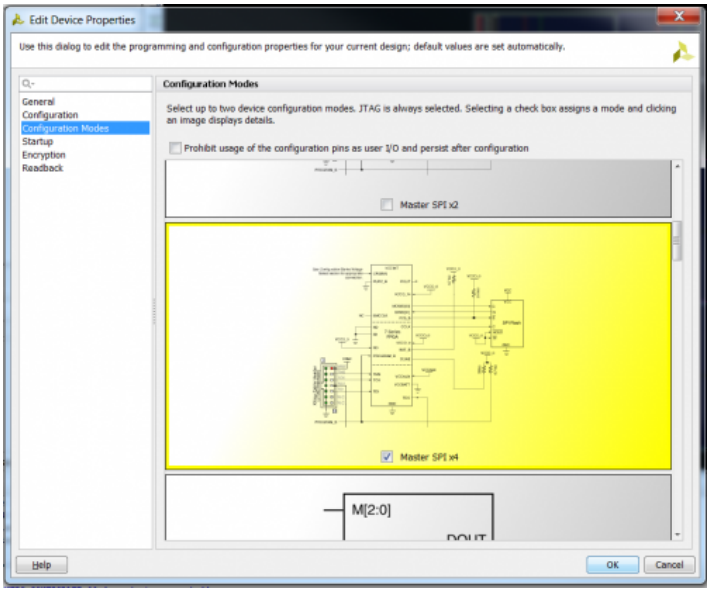
5.1) To improve programming speed, in the main toolbar select **Tools→Edit Device Properties…** Under General, set **Enable Bitstream Compression** to "TRUE".



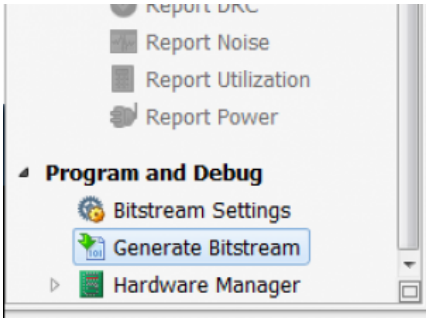
5.2) Under Configuration, set **Configuration Rate (Mhz)** to "33".



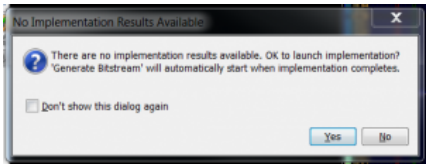
5.3) Under Configuration Modes, select **Master SPI x4**



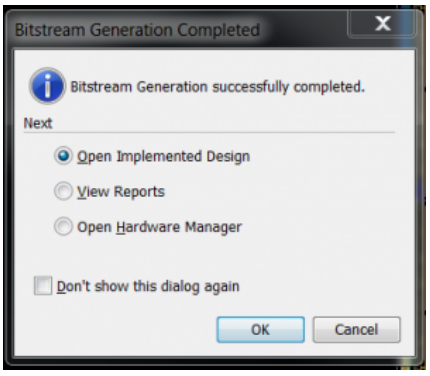
5.4) Press **Ok**, **save** your synthesized design (Ctrl+S) and then click **Generate Bitstream** in the Flow Navigator on the left side.



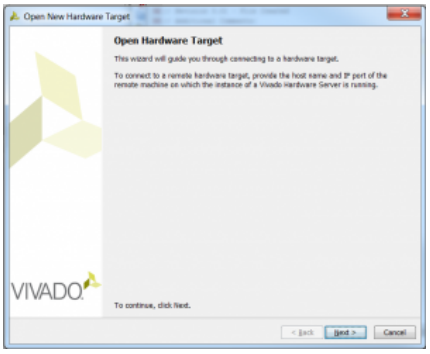
5.5) This will open a box stating that you have not implemented your design.



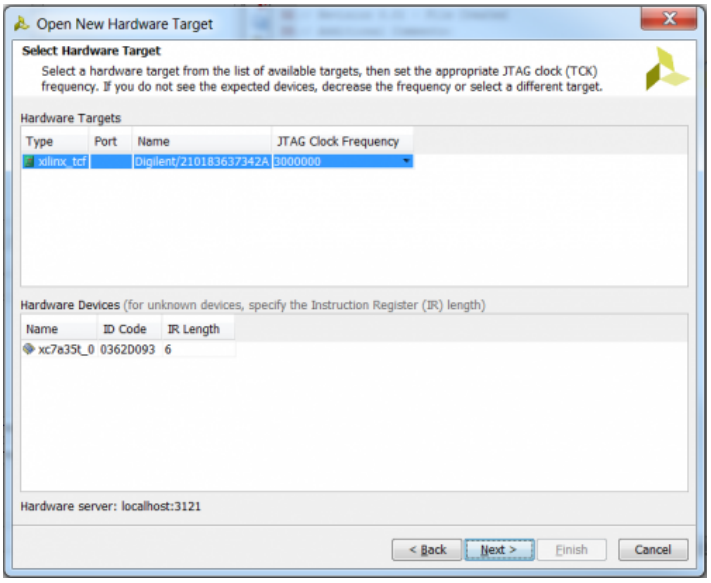
5.6) Click **Ok** and Vivado will generate your .bit file and will show you this box.



5.7) Click **Open Hardware Manager** and click **OK**. You will see this screen.



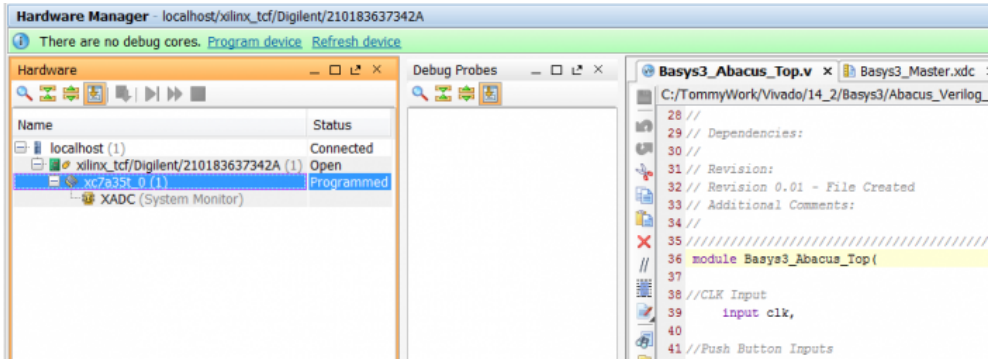
5.8) At this point, make sure your Basys 3 is plugged in via USB and turned on. Now click **Next** twice and you will see this screen.



5.9) Set the **JTAG Clock Frequency** to "30000000", select the device, and click **Next** followed by **Finish**.

6. Programming the Basys3 using a .bit file

6.1) First, make sure that the jumper JP1 is in the JTAG position. You should see something like this:

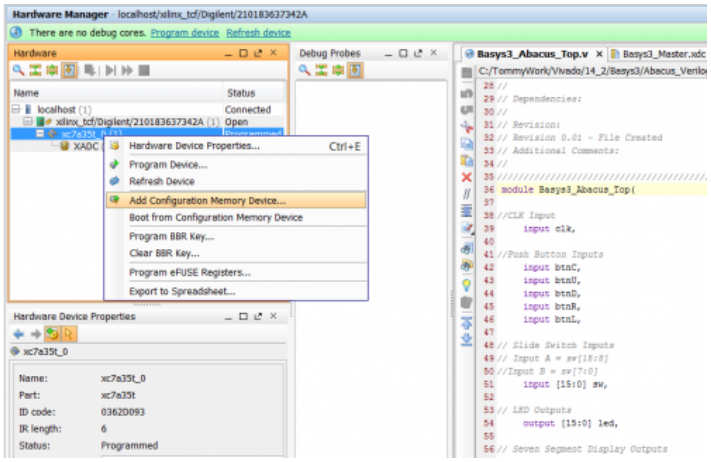


6.2) Click **Program device** (in the green bar) then xc7a35t_0, select your .bit file in the bitstream file box, and click **Program**.

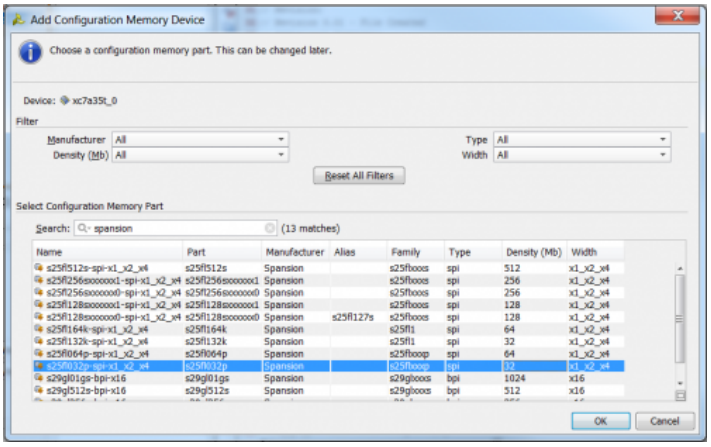
6.3) This will program your Basys3 through the JTAG connector.

7. Programming the Basys3 using .bin file

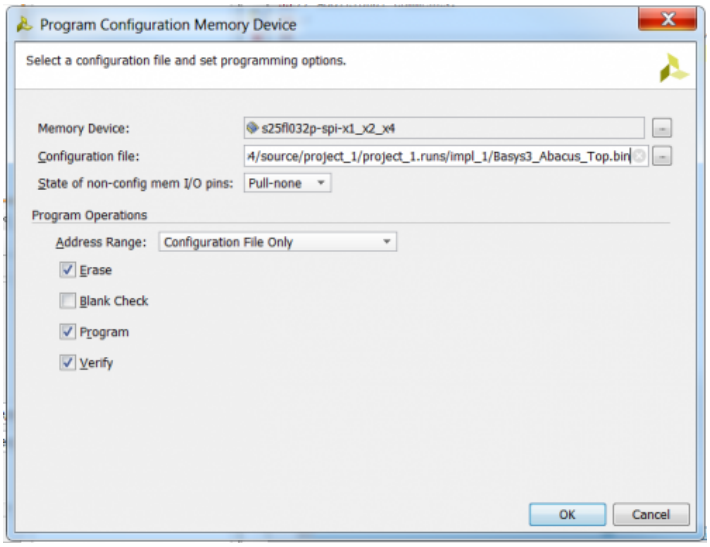
7.1) First, make sure the jumper on JP1 is in the QSPI position. In the Hardware Manager window, under hardware right click your device and click **Add Configuration Memory Device…**



7.2) This window will pop up. Search for "Spansion" and select the 32 bit device (highlighted below). Click **OK** on the next window asking if you want to program the configuration memory device.



7.3) Select the .bin file where it asks for a configuration file and finally click **OK**.



Vivado will now erase the old configuration file, and reprogram your Basys3 with the Abacus demo file. From now on, when you power cycle the Basys3, the abacus demo will load at startup. For information on operating the abacus demo, go here.

---

basys3/gsg.txt · Last modified: 2016/03/16 21:21 by Arthur Brown