ELEC 3004/7312: Signals Systems & Controls

EXPERIMENT 1: INTRODUCTION TO THE NEXYS 2

Aims

In this laboratory session you will:

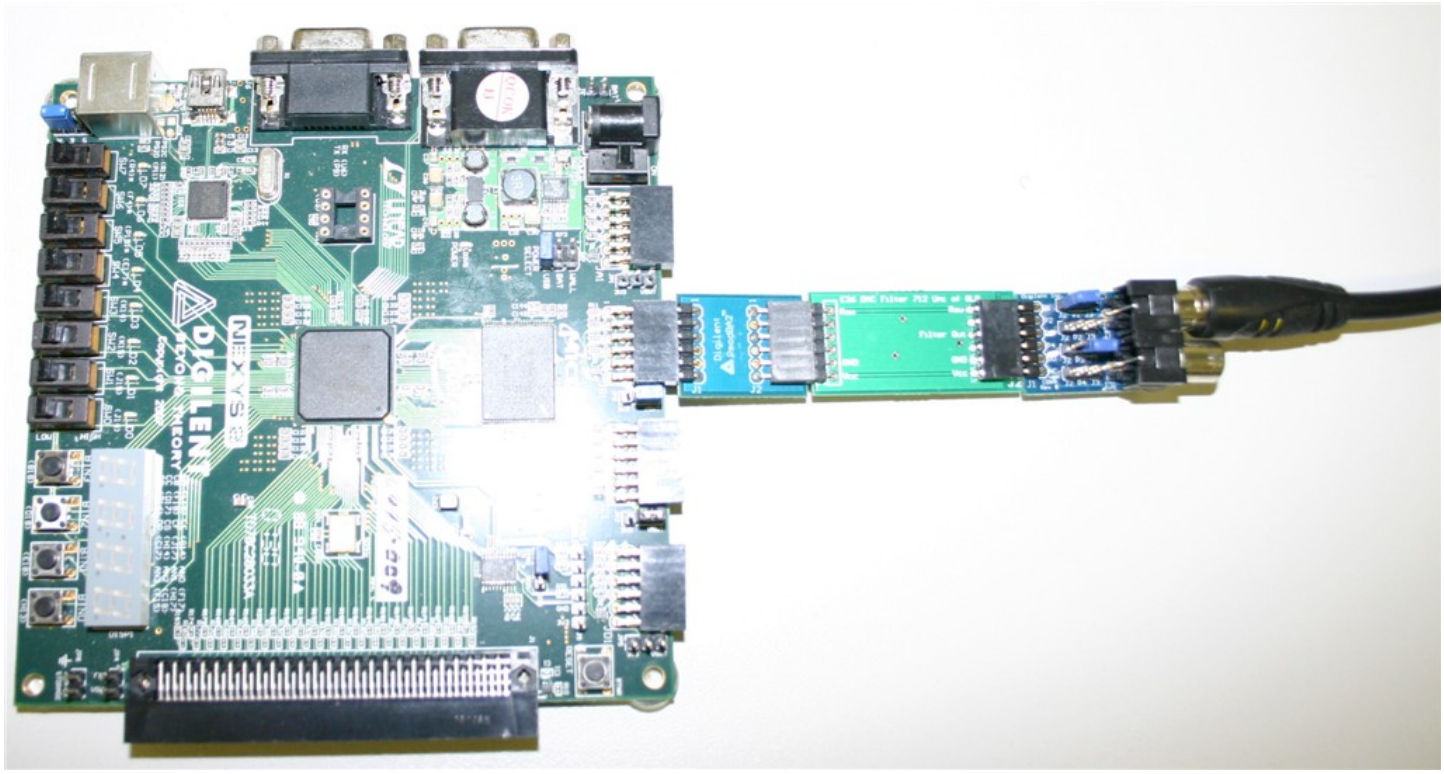
1. Gain familiarity with the workings of the **Digilent Nexys 2** for DSP applications;
2. Have a first look at the **Xilinx ISE** software for FPGA programming, using both VHDL and schematics, as well as **Digilent Adept** download software;
3. Appreciate the use of a look-up-table (**LUT**) method for generating periodic signals, such as sine waves.
4. See the effects of reconstruction filters.

Introduction

Digilent's Nexys 2 is a **field programmable gate array** (FPGA) development board that allows for rapid and interactive implementation and debugging of FPGA designs. The Nexys2 provides communications to and from the FPGA, plus access to a wide range of peripherals such as LCD, RAM, serial flash memory, and analogue to digital/**digital to analogue converters** (ADC/**DAC**) via IO ports. This experiment focuses on the **DAC** interfaces, which are available in 8-bit as well as 12-bit versions.

Equipment

1. PC with Xilinx ISE 13.4, Digilent Adept & MATLAB;
2. Nexys 2 + USB to JTAG interface cable/s
3. Digilent PMOD-DA2 DAC board and PMOD-CON4 RCA board
4. Oscilloscope
5. 2 x cable: RCA male to BNC male approx 0.5 - 1 m
6. 1 x E36 DAC Filter board (UQ designed)



Preparation

Note: (Individual) Preparation will be checked and marked at the start of each laboratory class. Hand in your preparation answers on a sheet to your tutor before the lab session.

Keep a copy to work with in the lab. Keep your marked sheet once it has been returned.

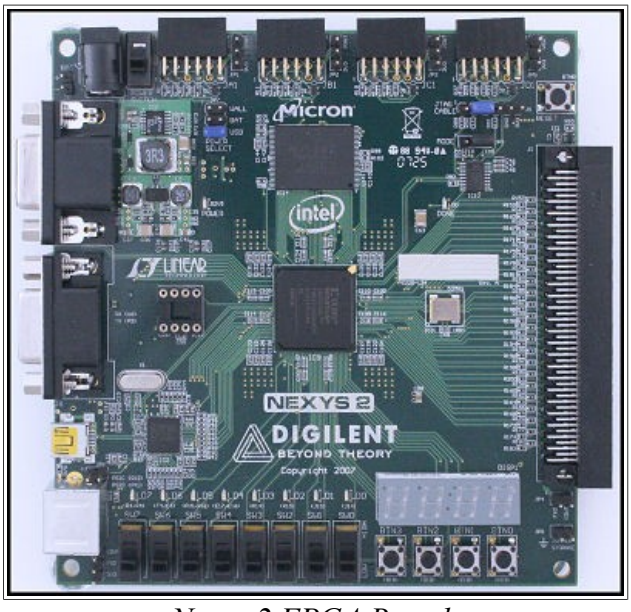
No preparation = no labs (no assessment). It will be best if you can maintain a log book for the labs.

Read and familiarise yourself with the following documents available from the course (or manufacturer's) website:

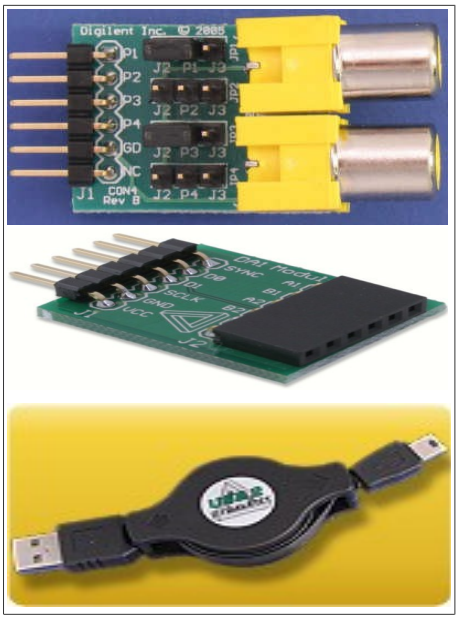
- The Nexys 2 Reference Manual Pages 1-5, 15, 17; http://www.digilentinc.com/Products/Detail.cfm?NavPath=2,400,789&Prod=NEXYS2
- Digilent PMOD interface boards: http://www.digilentinc.com/Products/Catalog.cfm?NavPath=2,401&Cat=9
- PMOD DA2 Dual 12-bit DAC http://www.digilentinc.com/Products/Detail.cfm?NavPath=2,401,487&Prod=PMOD-DA2
- PMOD AD1 Dual 12-bit ADC http://www.digilentinc.com/Products/Detail.cfm?NavPath=2,401,499&Prod=PMOD-AD1

Answer the following questions:

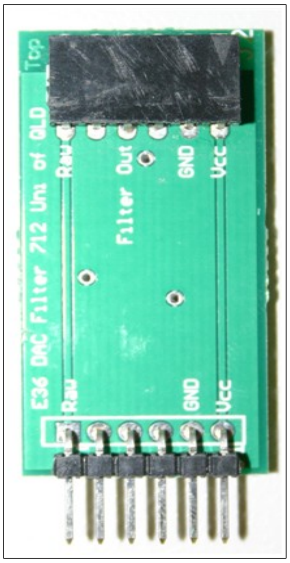
1. Tabulate the two's complement binary representation of the decimal numbers -8 to 7. How many bits are required to do this? Identify the "sign bit" and the most significant bit.
2. Tabulate the unsigned binary representation of the decimal numbers 0 to 15. If this binary representation is then "shifted" (i.e., AC coupled) so that 0 represents $-V_{DD}/2$ and 15 represents $+V_{DD}/2$, again, identify the "sign bit" and the most significant bit.
3. Given the PMOD-AD1 can handle a range of input levels from GND to V_{DD} (where V_{DD} = 5v) and it has a 12-bit, unipolar binary analogue to digital (ADC), what is the quantisation step-size of the ADC?
4. The PMOD-DA2 is a 12-bit unipolar DAC with an output range from GND to VDD (where V_{DD} = 5v). What is the quantisation step-size of its output?
5. Can you explain how the code in SINEWAVE.VHD generates a sine wave? Can you predict the effect of increasing the number of samples in the LUT?
6. For 12-bit unsigned binary, what are the maximum and minimum decimal values?
7. Convert the minimum and maximum values, as well as 2048, 2831 and 3495 to hexadecimal, then into 12-bit binary. What are the values of I (index) for these values in the Look-up-table in the appendix?



Nexys 2 FPGA Board



PMOD-CON4, PMOD-DA2, USB-JTAG

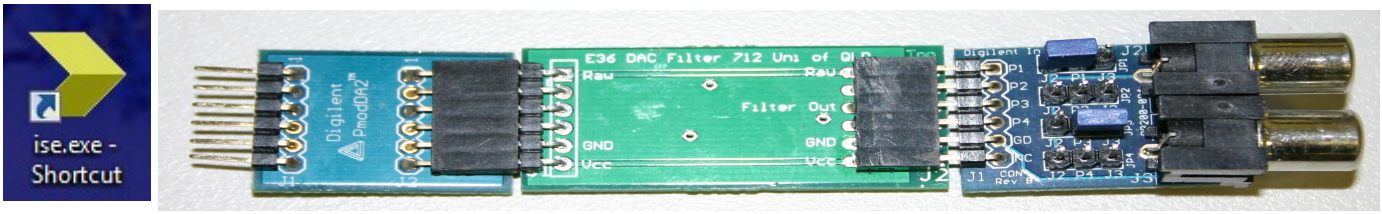


DAC Filter board

Procedure

Part 1: Getting familiar with the Nexys 2 by generating a sine wave

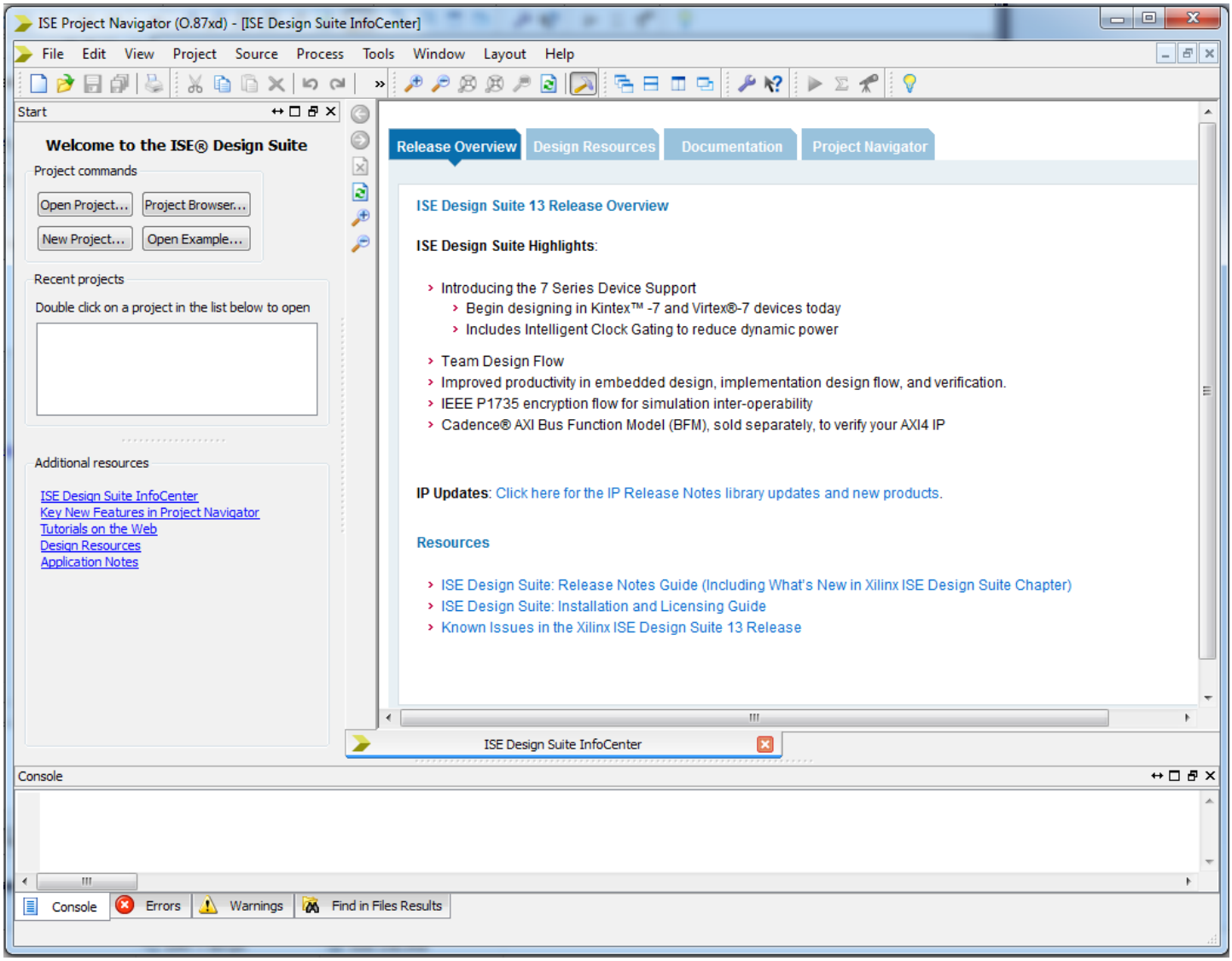
1. Start Xilinx ISE using the desktop icon or run C:\Xilinx\13.4\ISE_DS\ISE\bin\nt64\ise.exe



Assemble the Nexys2 Board with a PMOD-CON4 (RCA), DAC Filter Board and PMOD-DA2 attached to JB1, as shown on page 1.

Now connect an RCA/BNC cable between the PMOD-CON4 upper socket and CH 1 on the oscilloscope**, then connect a second cable from the lower PMOD-CON4 socket to CH 2 on the oscilloscope.**

2. Create an FPGA Project

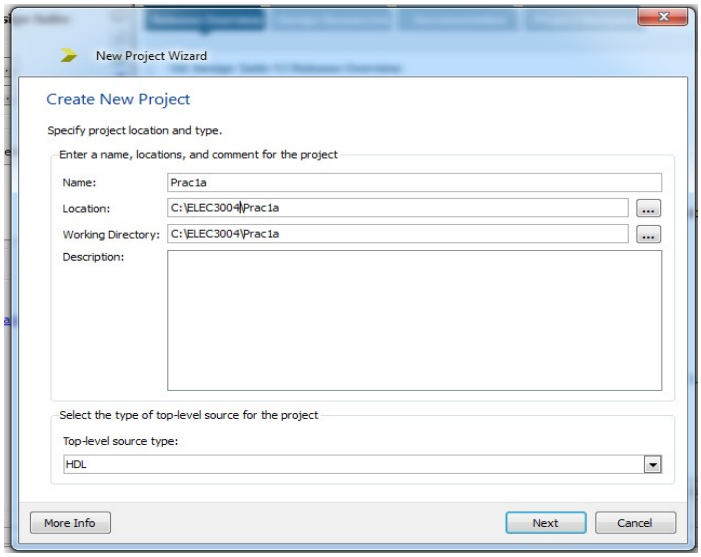


3. Click on "**New Project**".

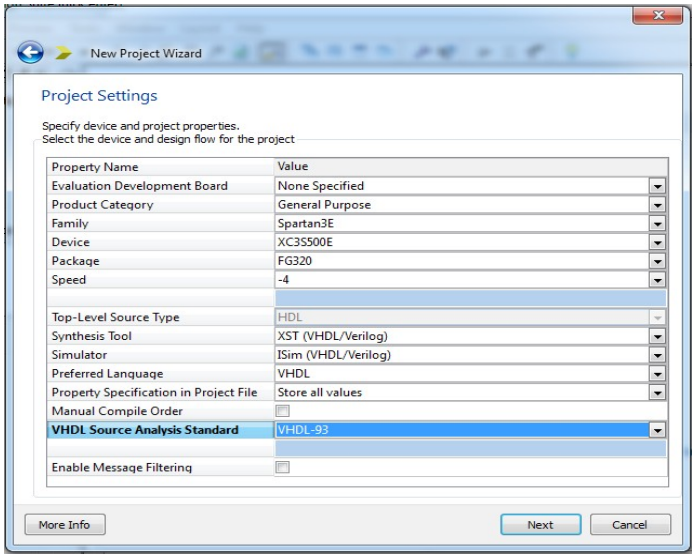
4. Type in the Location or Working Directory, then the Name of your Project.

If you use a network drive, this will significantly slow things down, so use a local drive or USB drive!

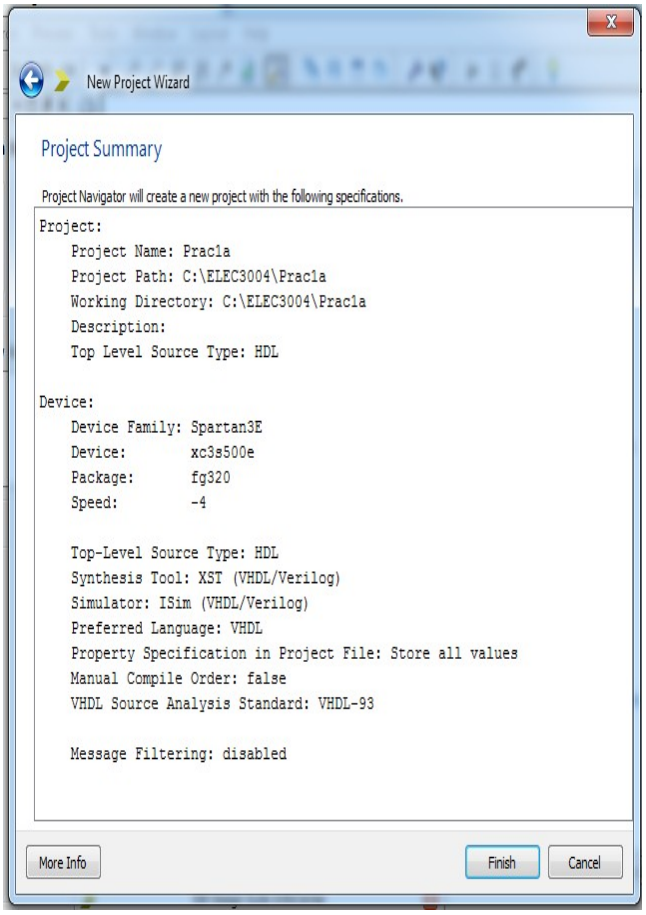
You may have to use the Browse Directory button [...]

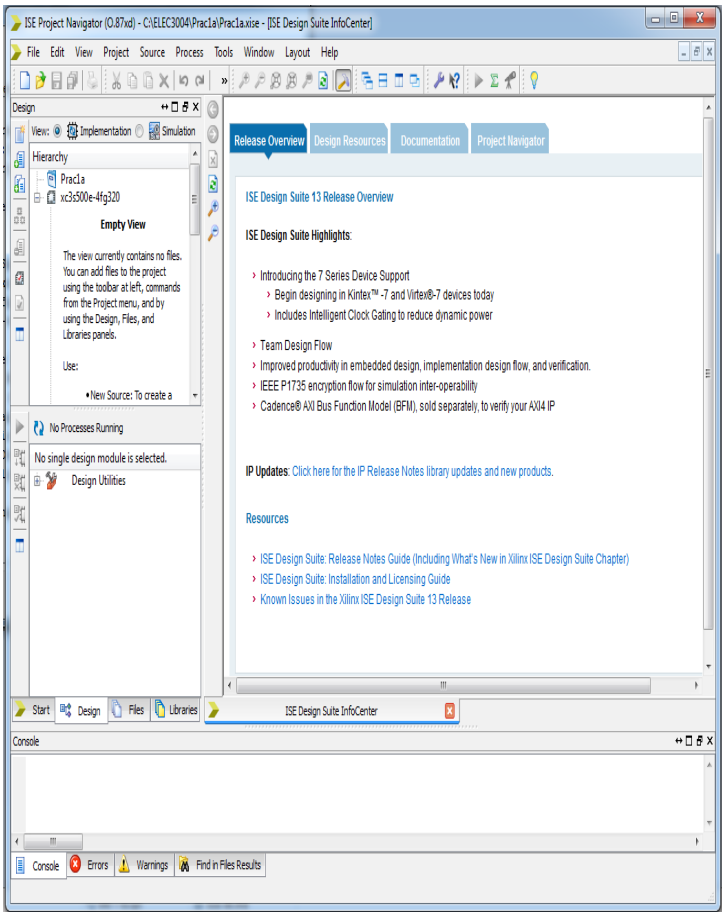


Enter the Device Properties as shown: General Purpose, Spartan 3E, XC3S500E, FG320, -4, HDL, XST, ISIM, VHDL



5. You will now see a Project Summary page. Click “Finish” The files for your project can now be added



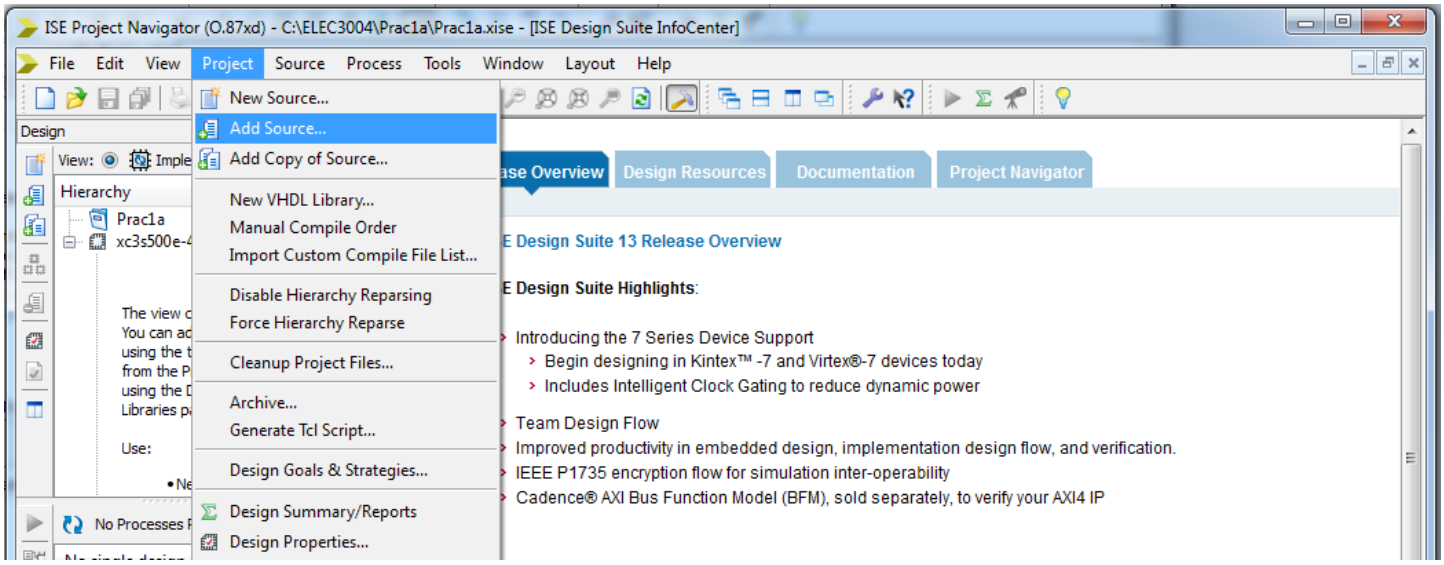


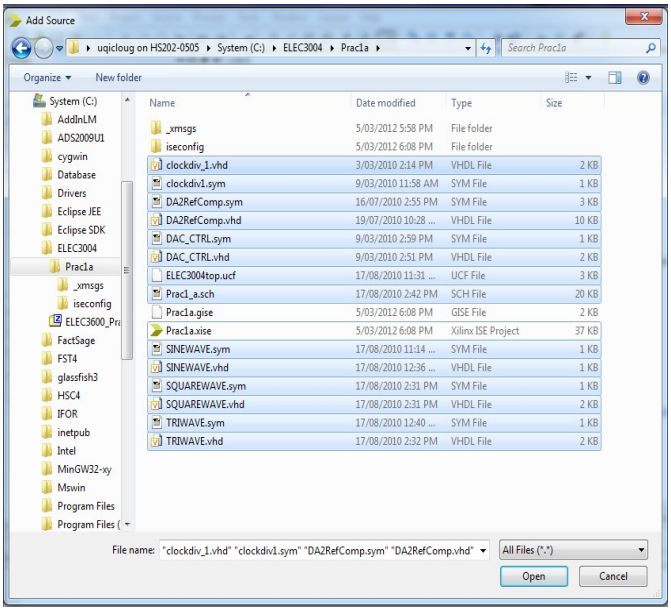
6. Download the following files from the ELEC3004 website and place them in your Working Directory:

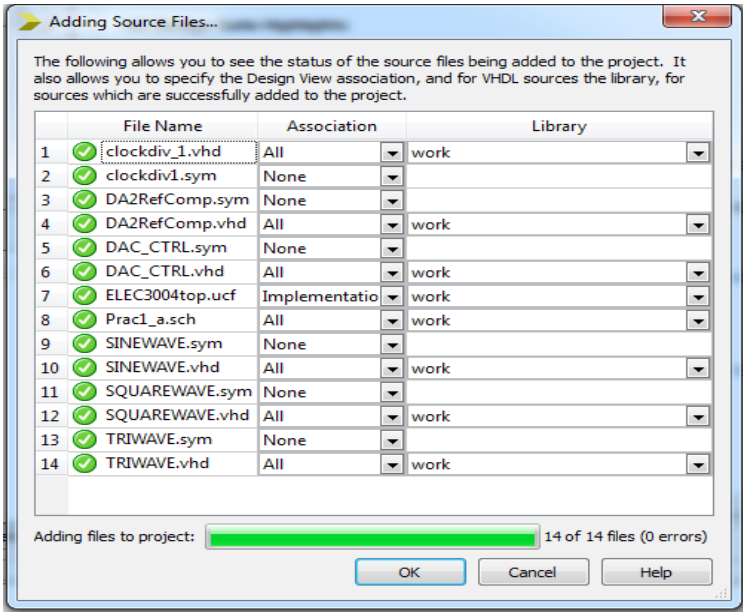
sinewave.vhd	clockdiv1.sym
clockdiv_1.vhd	SINEWAVE.sym
DAC_CTRL.vhd	**DA2RefComp.sym**
DA2RefComp.vhd	DAC_CTRL.sym
squarewave.vhd	squarewave.sym
triwave.vhd	triwave.sym
Prac1_a.sch	ELEC3004top.ucf

These may also be available in a zip file

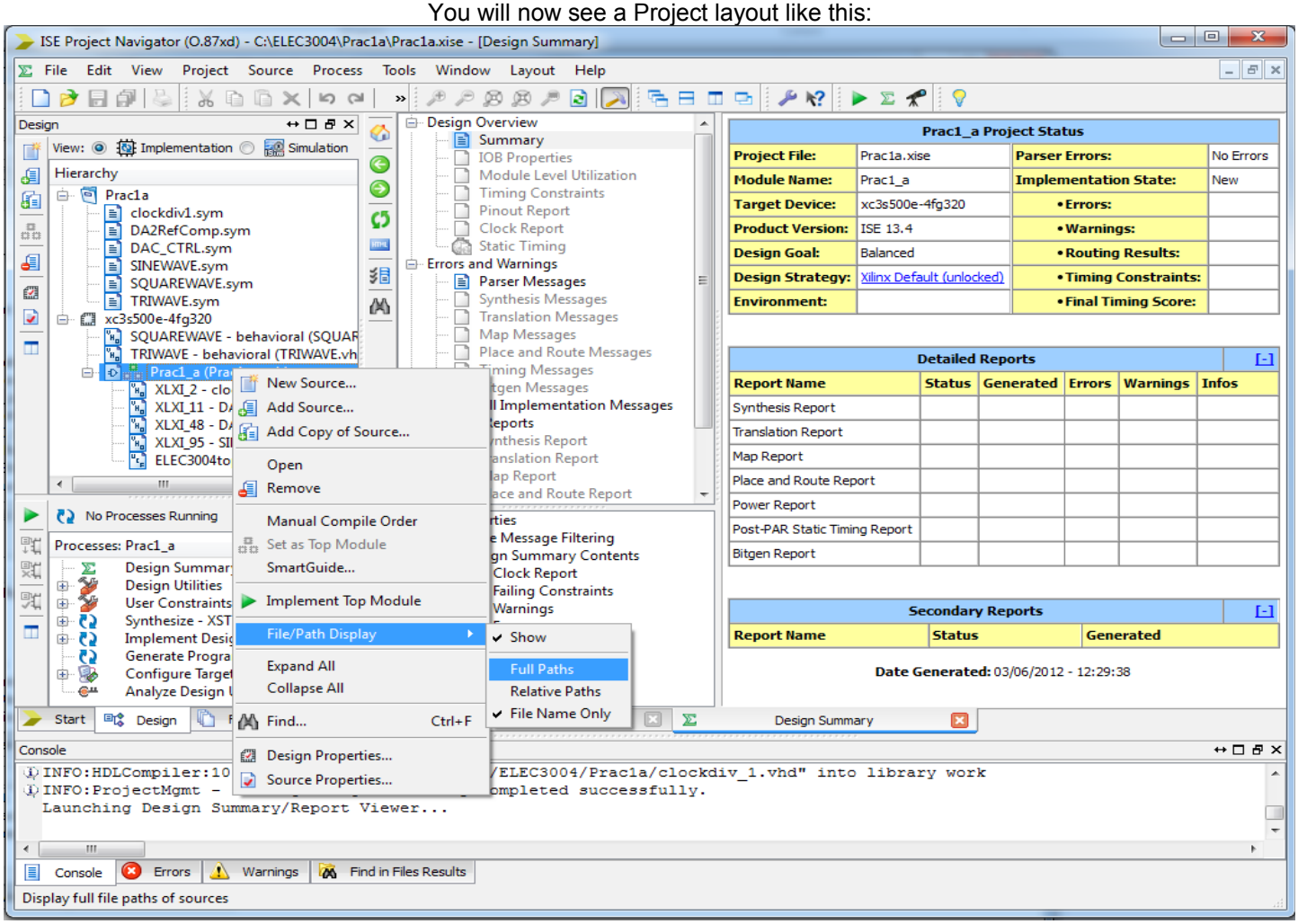
7. Now use **Project\Add Source** and add all the files.





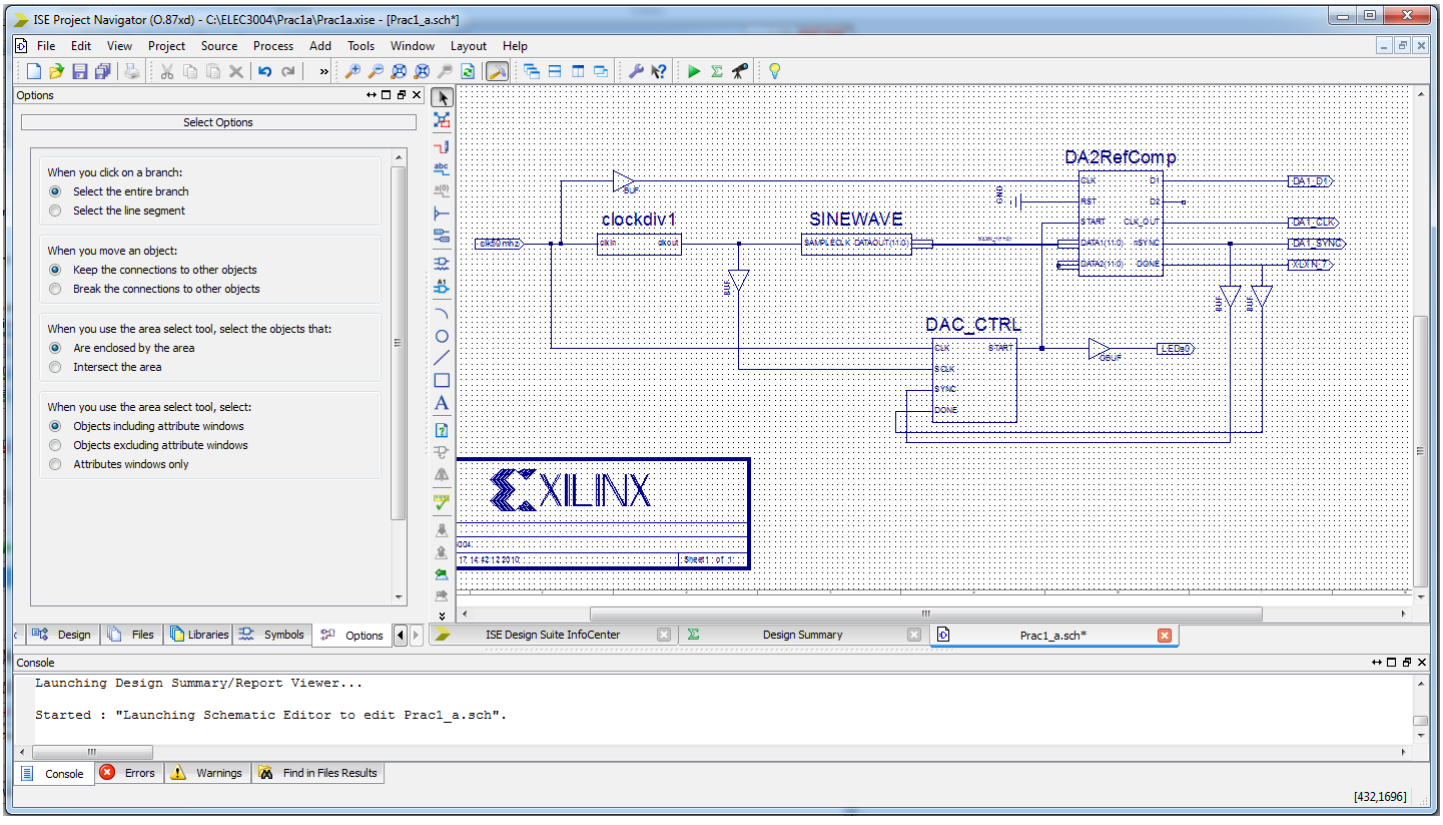


You will now see a Project layout like this:

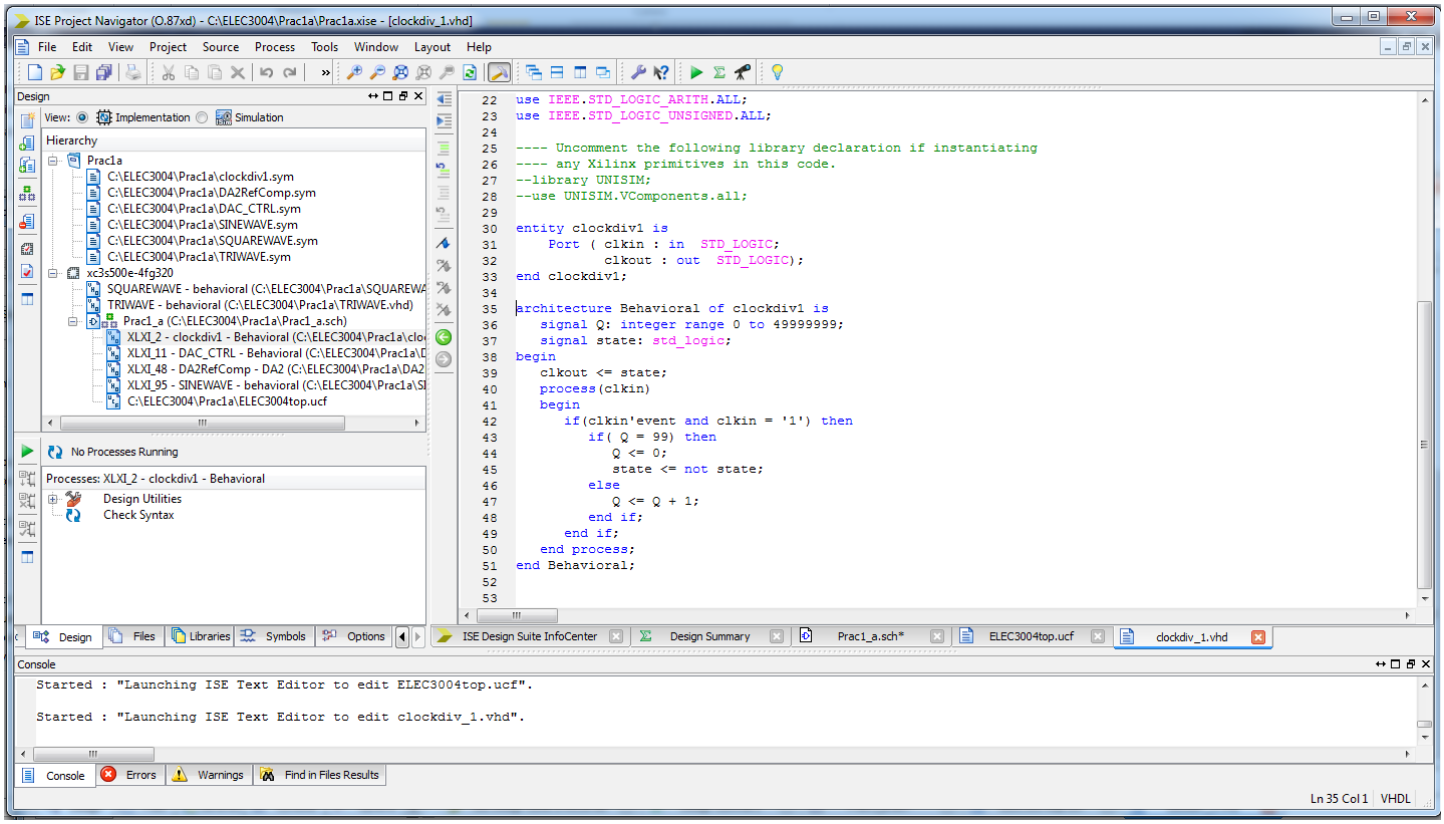


8. **Set the Full Paths option as shown above**

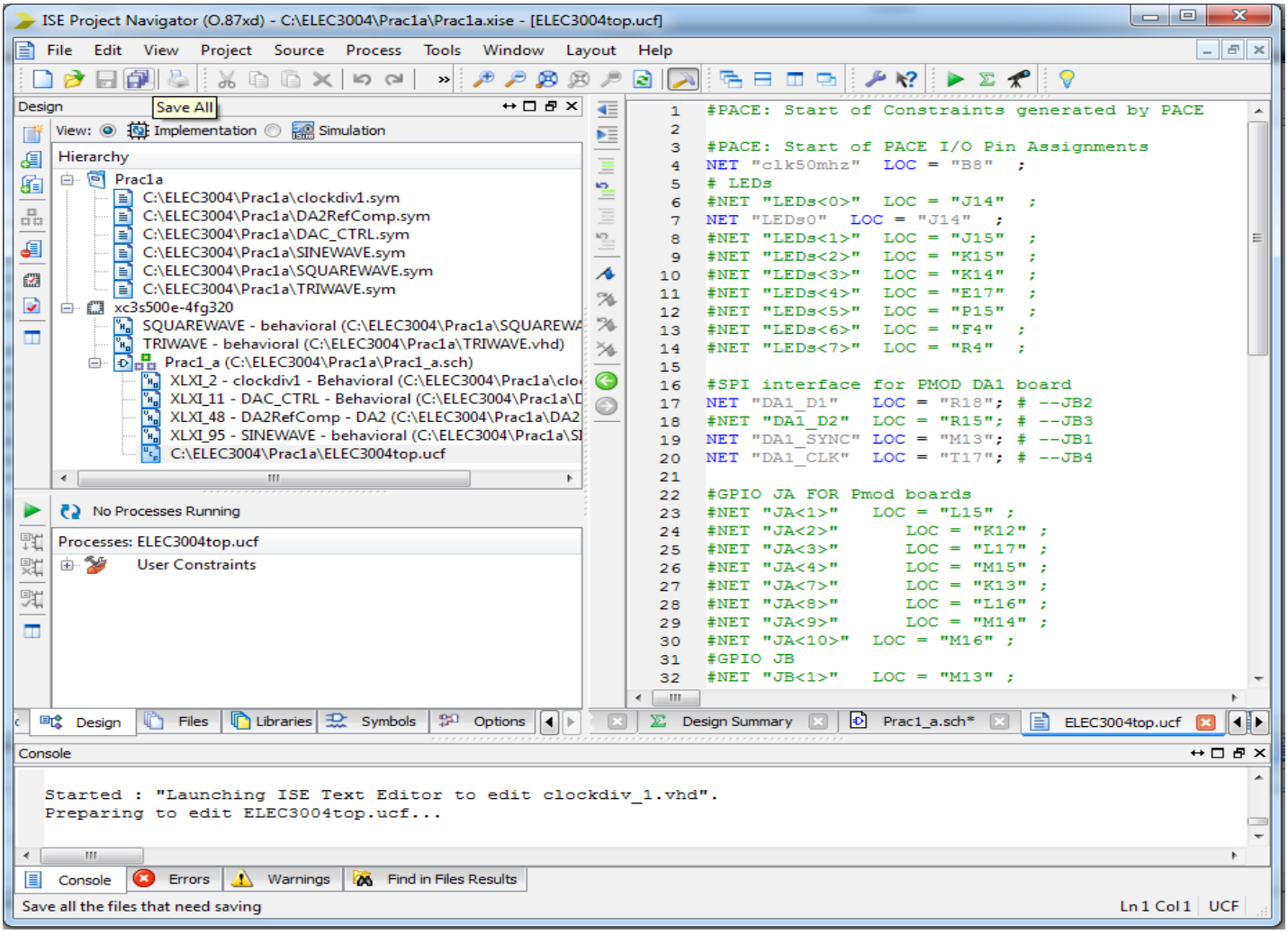
9. If you double-click on Prac1_a.sch, it will open in a window in the Project View Window:



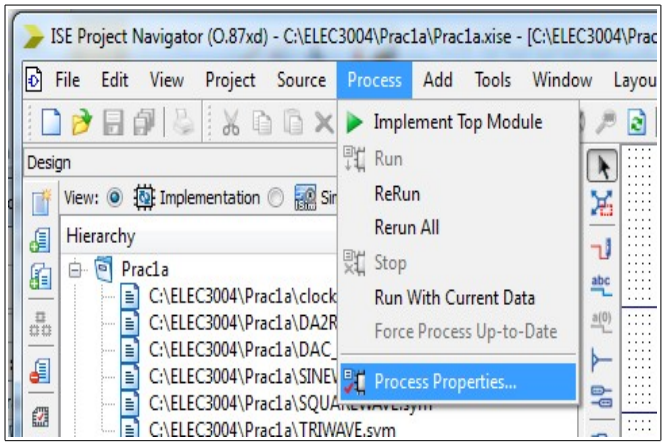
10. If you double-click on any of the ".**vhd**" files, you will see the contents of that file in the Project View window

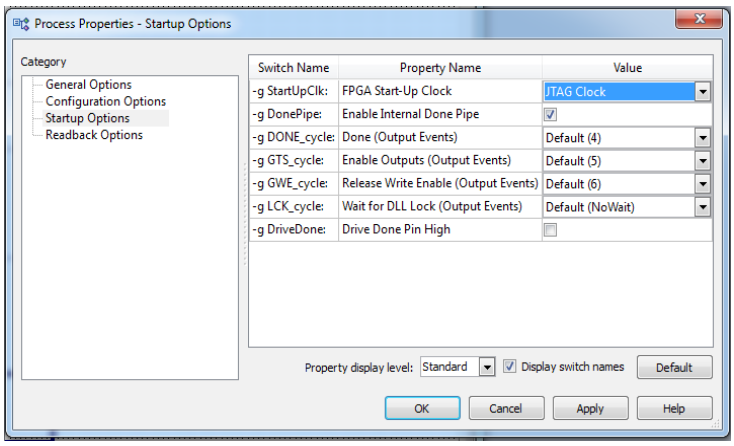


11. To see the contents of ELEC3004top.ucf, double-click on the filename



12. Next select **Process Properties**, then in Startup Options, set **FPGA Start-Up Clock** to JTAG Clock, and then click OK. ***If this is the first time you have used this software, you may have to try instruction 13 below, first, then repeat instruction 12 to set the FPGA Start-Up Clock.***



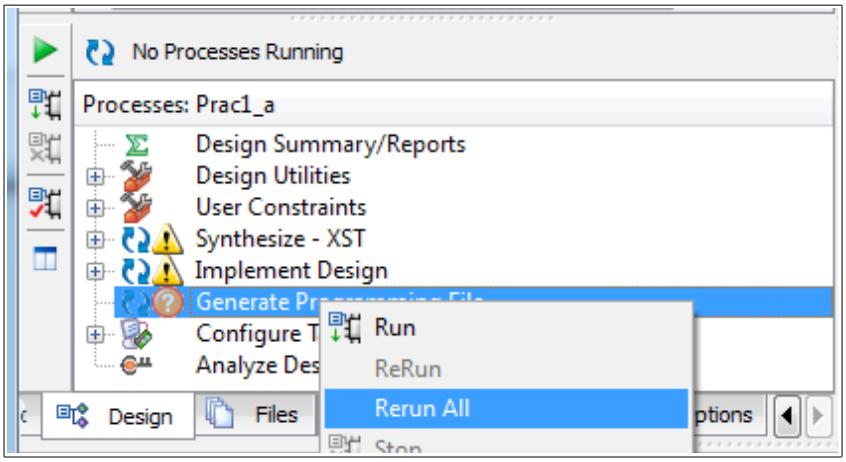


Now you can generate the file which will be downloaded to the Nexys 2 board using Digilent Adept software.

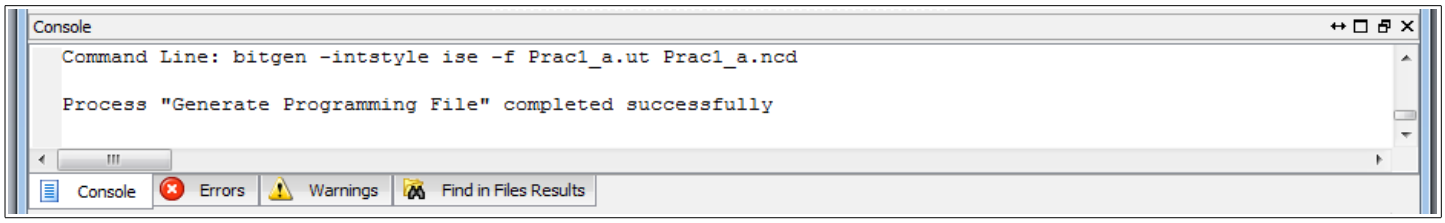
13. **Click once** on the Prac1_a filename in Hierarchy, and then double-click **Generate Programming File**.

If necessary, repeat instruction 12 to set the FPGA Start-Up Clock.

After 1 - 2 minutes, a file called “prac1_a.bit” will be saved to your Working Directory



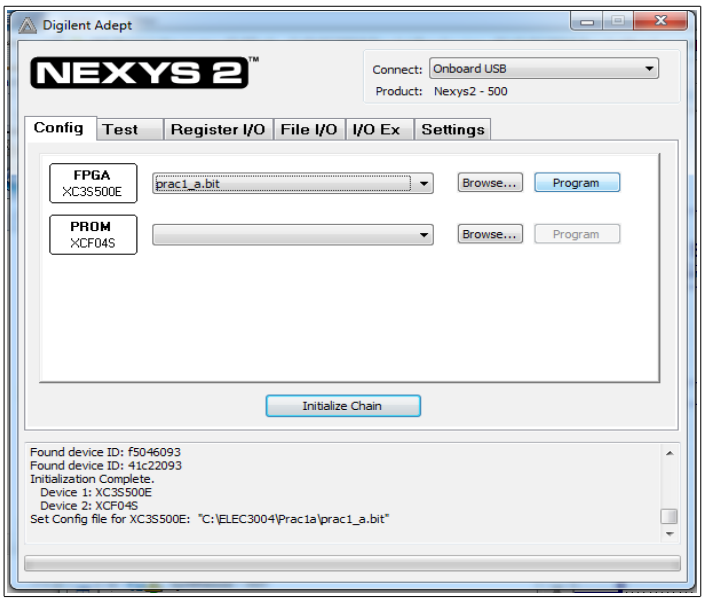
If all has gone well, you will see a message, “**Process "Generate Programming File" completed successfully**” in the Console window at the bottom of the screen.

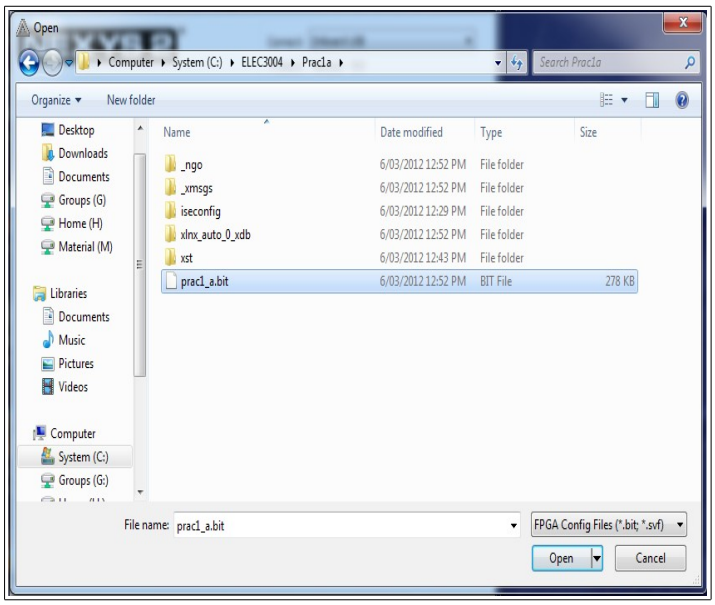


Using the Digilent Adept software

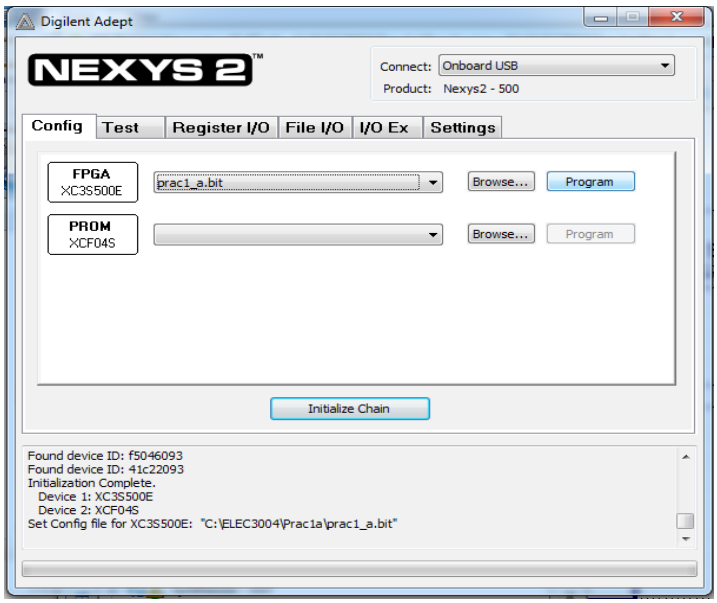
1. To download the file to your board, connect the USB to JTAG cable that comes with the Nexys 2 board, then start up the Digilent Adept software.

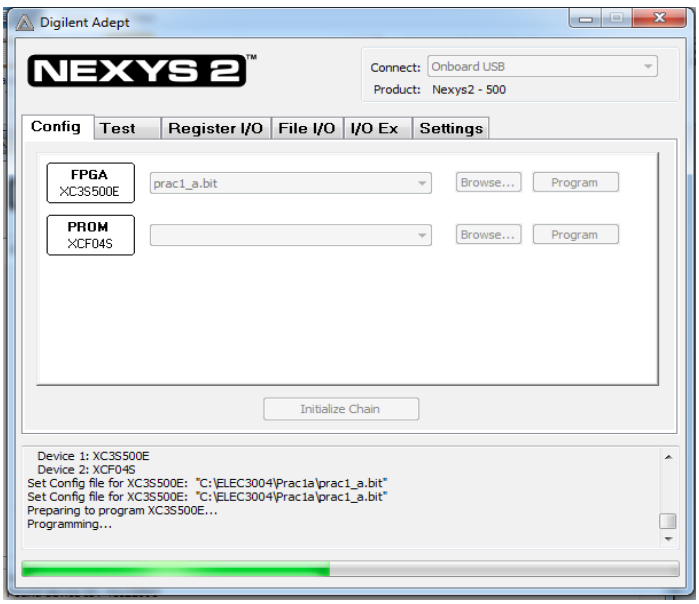
2. From the startup page, you may have to select Onboard USB, then Browse to your file and select Open





3. Now press the **Program** button. A green progress bar will appear





4. You should now have a sinewave being sent to the PMOD-DA2 on port JB1 of the Nexys 2. You may look at this with your oscilloscope to verify that it is working correctly. To see the VHDL code used to generate the sinewave, double-click on the SINEWAVE.VHD file.

Questions

If you have the DAC Filter board available, the Raw signal is connected to the upper PMOD-CON4 connector and the Filtered signal is connected to the lower PMOD-CON4 connector.

1. Plot the output waveform/s on the oscilloscope. What is the amplitude and frequency of the sine wave generated on each available channel?
2. The Master Clock is 50MHz, which is then divided by 200, in clockdiv1.vhd, to produce **Fr**. This frequency, **Fr**, controls the readout rate of the LUT which is n=16 elements long. If it takes 16 clock cycles of **Fr** to produce a complete LUT cycle, what is the theoretical audio output frequency? Is it the same as your measured frequency?
3. Can you derive the simple equation for the audio output frequency, based on the value of clockdiv1, and n = the length of the LUT?
4. Try to confirm that the DAC step size is approximately the same as that calculated in the preparation. If it is not, can you explain why? **Hint** 1: is the DAC connected directly to the output? **Hint** 2: can you see a relationship between the steps in the output waveform, and the changes in the values in the LUT?
5. Suggest how you might modify the VHDL code so that a LUT with only 8 points is required. What are the advantages and disadvantages of this approach?

Part 2: Direct Digital Synthesis

Modify the original HDL code so that:

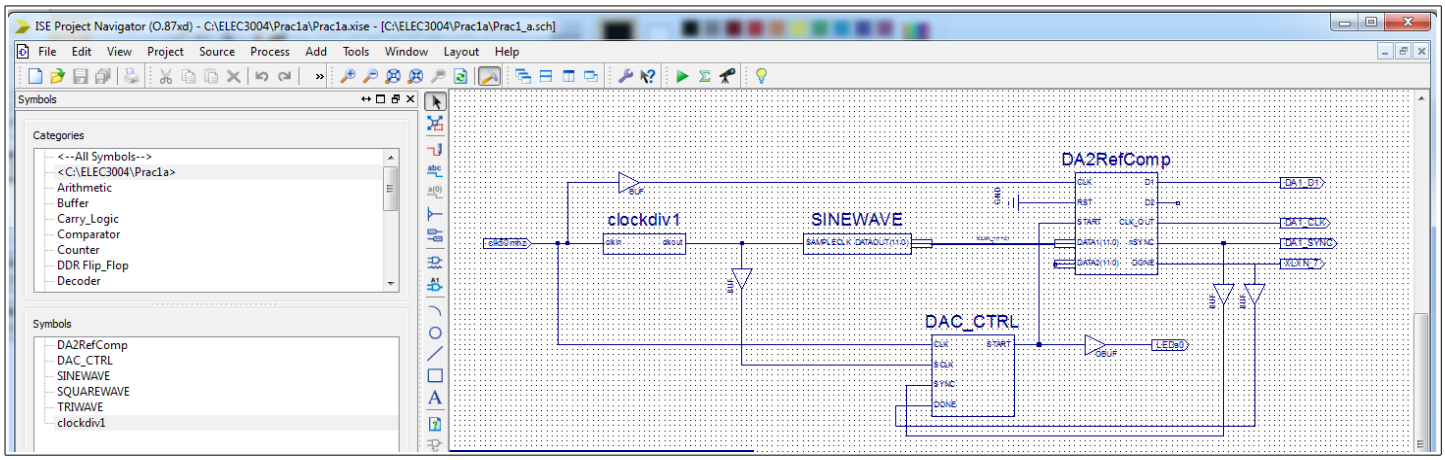
1. A sine wave of **half** the original **amplitude** is generated. Look at the resultant trace of the oscilloscope. **Hint**: how does the "shifted" binary representation change if you divide by two?
2. A sine wave of **half** the original **frequency** is generated. Note: in principle there are two ways of doing this; changing the clock divide or changing the size of the LUT. Try both. Which is best?
3. What would you need to do to generate a sine wave of an arbitrary frequency? Can you write some Matlab code that generates the values required for the LUT? **Hint**: A sinewave ranges from -1 to +1. Now convert this to a 0 to +1 range, then from 0 to the 12-bit maximum value.

Next, try using the **dec2bin(value, no. of bits)** function.

Part 3: Attempt the following for a challenge

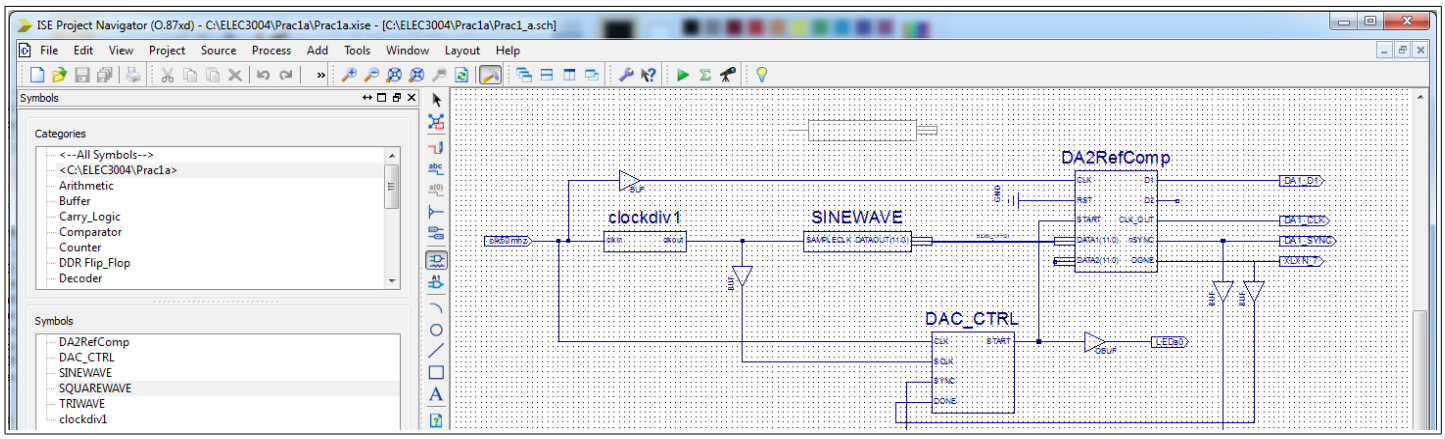
Sinewave.vhd can be replaced with either triwave.vhd, or squarewave.vhd, thus changing the output waveform. In fact, an arbitrary waveform (shape and frequency) could be generated using a suitable LUT.

The following instructions show how to replace sinewave.vhd with one of the others.

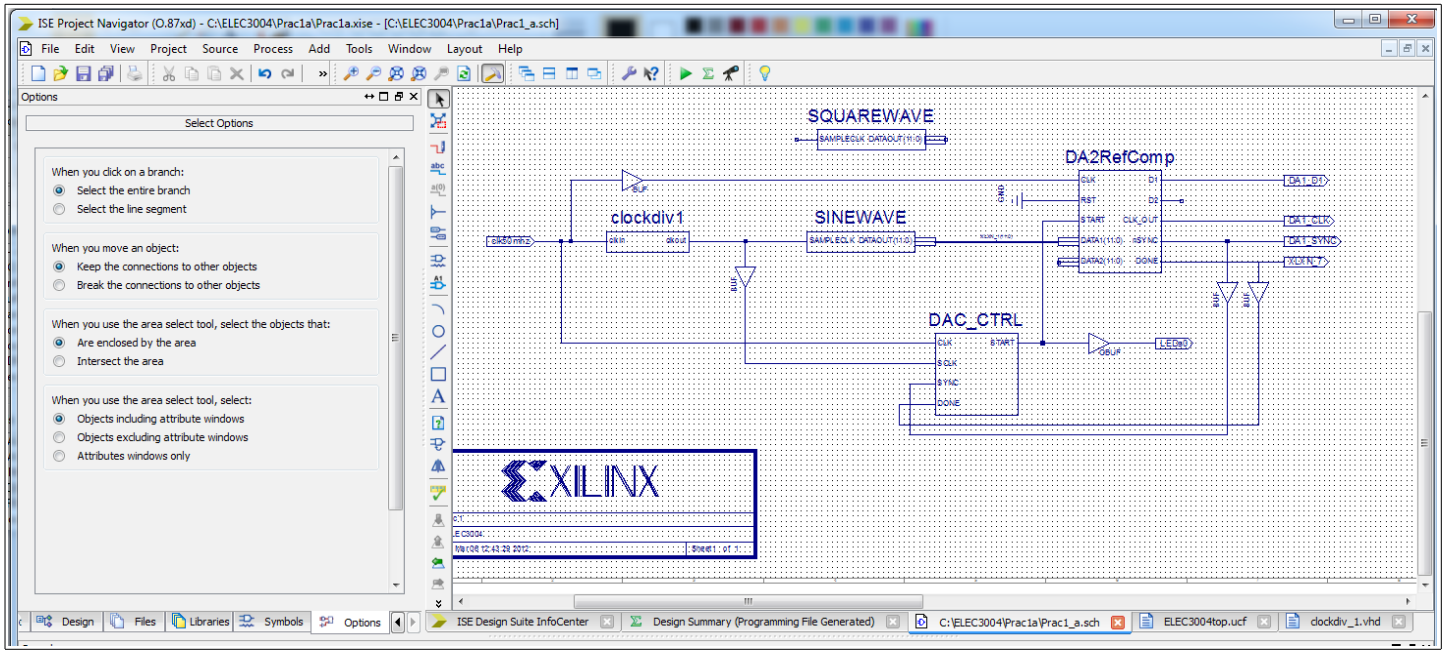


1. Click on the **Symbols** tab, then **click once** on the name of your project in the **Categories** window.

2. You will see a list of symbols related to your project. If you click on SQUAREWAVE in the **Symbols** window, then move the mouse over the schematic, you will see a grey symbol on the schematic.

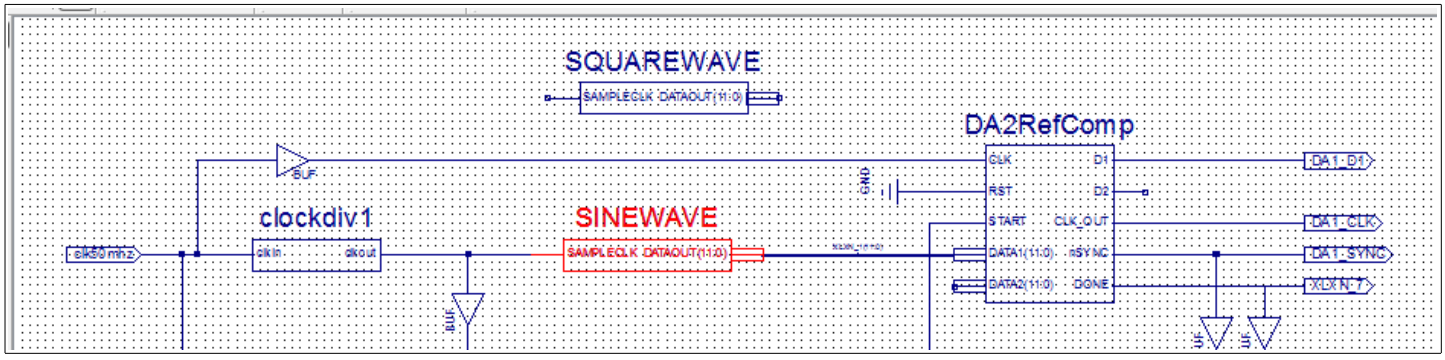


3. If you **click once** at a suitable location on the schematic, then press the ESC key, you will see the SQUAREWAVE symbol at that location. It can now be moved with the mouse. To place multiple copies of the symbol, click at multiple locations until you have enough, then press ESC

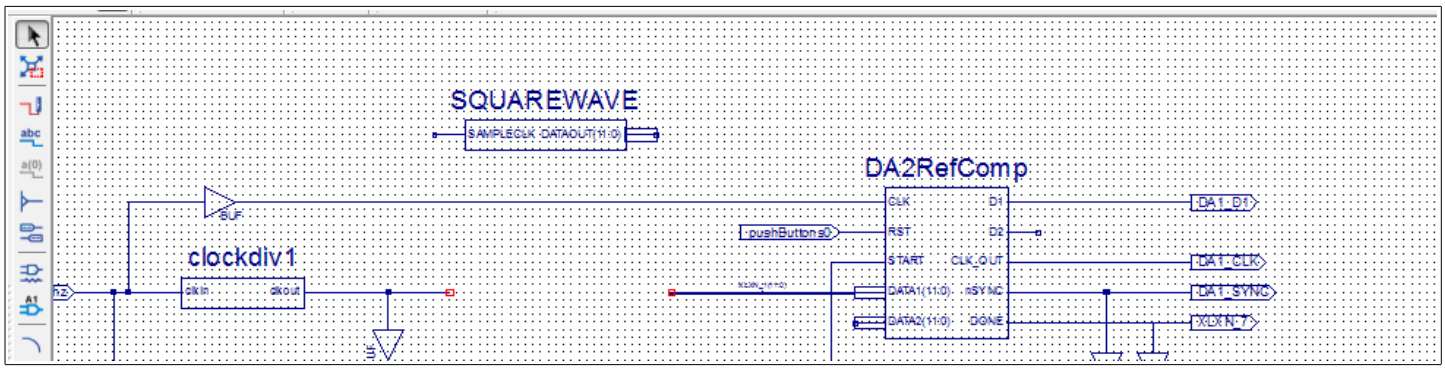


There are four window functions available: F5 = Refresh, F6 = Show all, F7 = Zoom out, F8 = Zoom in.

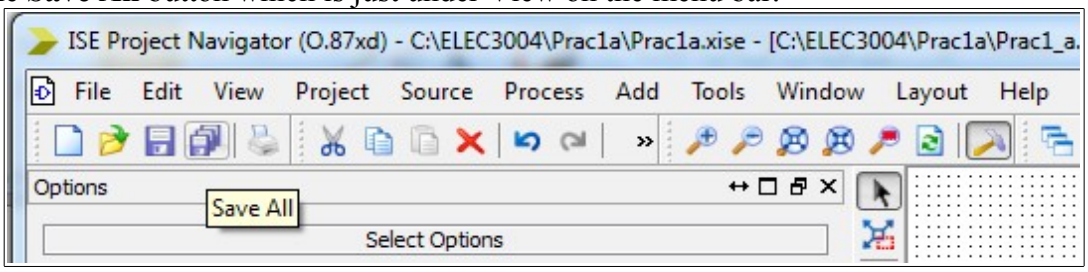
4. If you **click once** on a placed symbol, it will change from blue to red. If you then press the Delete key, it will be removed from the schematic, thus:



5. The SQUAREWAVE module can now be dragged to the location where SINEWAVE was and the connections will be automatically joined.

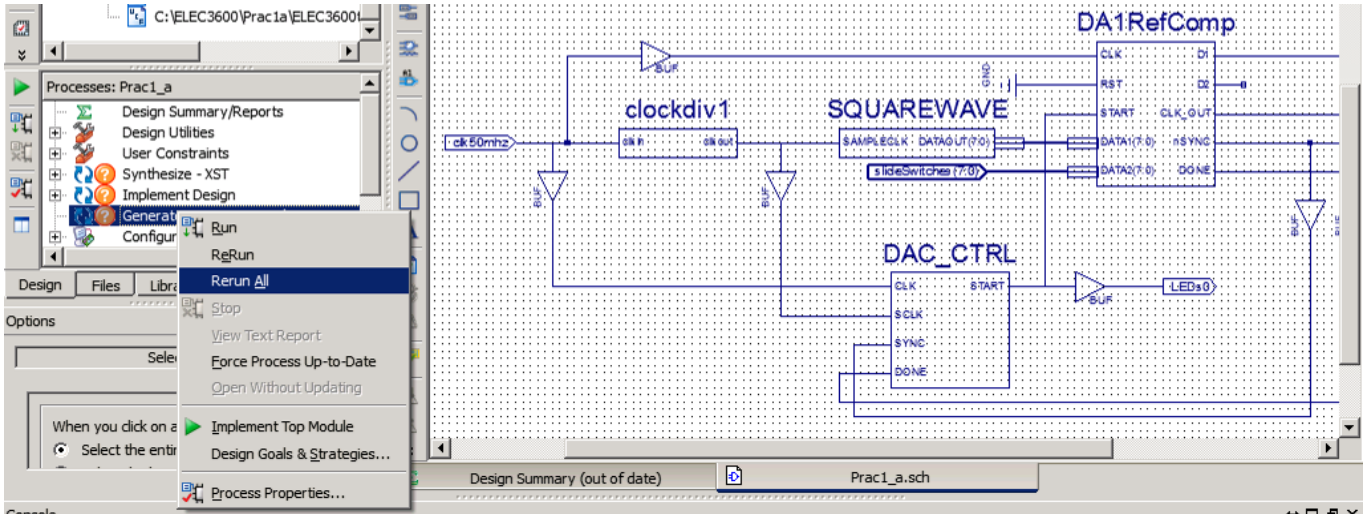


6. Use the **Save All** button which is just under View on the menu bar.



Before generating the new ".bit" file, you may wish to rename the previous "Prac1a.bit" to "Prac1_sinewave.bit", because the process will overwrite the current "Prac1a.bit" with the bitfile relating to your new schematic, and it is easier to compare different project outputs by using Digilent Adept to download a bitfile in a few seconds, than to create/rebuild a project with different functionality.

7. Click once on the **Design** tab, then right-click **Generate Programming File**, then click on **Rerun All** Assuming there are no errors, a new bitfile will be generated.



8. Follow the steps from "Using the Digilent Adept software" to download this new file to the Nexys 2 board.

Q.1 Look at the waveform/s on the oscilloscope. What amplitude and frequency do you observe on each channel?

Q.2 How could you halve the amplitude?

Q.3 How could you double or halve the frequency?

Q.4 If you have the filter board, is there a difference between the waveforms on each channel?

Q.5 Repeat Steps 1 to 8 immediately above, using TRIWAVE.VHD

Appendix:

Sinewave.vhd

```
library IEEE;
use IEEE.std_logic_1164.all;
use IEEE.std_logic_unsigned.all;

entity SINEWAVE is port
   (
        DATAOUT      : out  std_logic_vector(11 downto 0);
        SAMPLECLK    : in   std_logic
   );
end SINEWAVE;


architecture behavioral of SINEWAVE is

signal I : integer range 0 to 15;

begin
            with I select
          DATAOUT <= "100000000000" when 0,     --800
                     "101100001111" when 1,     --B0F
                     "110110100111" when 2,
                     "111101100011" when 3,
                     "111111111111" when 4,
                     "111101100011" when 5,
                     "110110100111" when 6,
                     "101100001111" when 7,
                     "100000000000" when 8,
                     "010011110000" when 9,
                     "001001011000" when 10,
                     "000010011100" when 11,
                     "000000000000" when 12,
                     "000010011100" when 13,
                     "001001011000" when 14,
                     "010011110000" when 15;

            I <= I + 1 when rising_edge(SAMPLECLK);
   end behavioral;
```