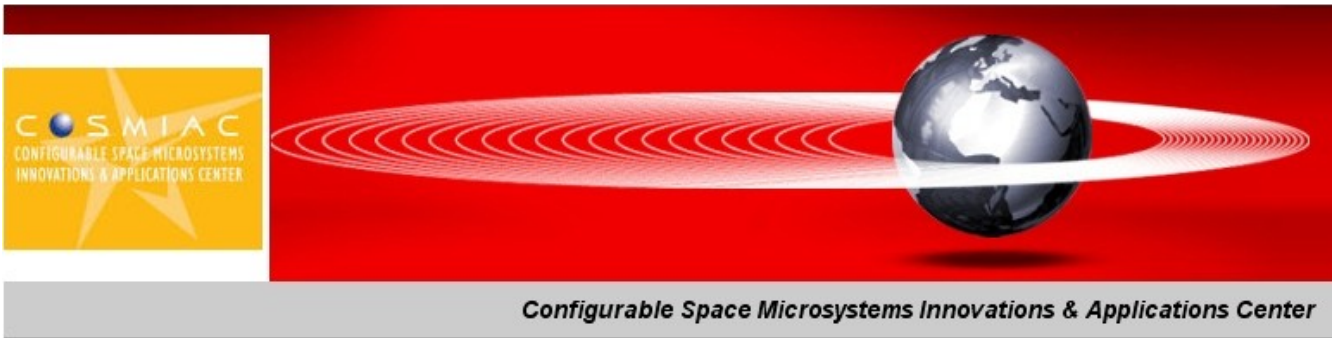


Tutorial 11
ChipscopePro, ISE 10.1 and Xilinx Simulator on the Digilent Spartan-3E board

Introduction

This lab will be an introduction on how to use ChipScope for the verification of the designs done on FPGAs. ChipScope Pro 10.1 is the tool provided by Xilinx for this purpose. The board will be a Digilent Spartan-3E starter kit.

Objective

The objective is to verify the functioning of a simple counter implementation using ChipScope. The counter implemented here is a 4-bit counter operated at five different frequencies. The aim is not to implement complex digital designs but to show the user a method to integrate ChipScope into an existing design in order to verify its operation in a simple and efficient manner. ChipScope is a virtual logic analyzer.

Prerequisites

- Basic knowledge about digital design and FPGAs.
- Acquaintance with Xilinx ISE 10.1 and Xilinx Simulator tools.
- ChipScope is not included in ISE but is a program available through the Xilinx university program.

Application

This document is used by students, who are learning FPGA design and verification.

Design Overview

4-bit counter implemented at five different frequencies using the system clock running at 50 Mhz. The five frequencies are $\frac{50\,Mhz}{2^3}$, $\frac{50\,Mhz}{2^{23}}$, $\frac{50\,Mhz}{2^{24}}$, $\frac{50\,Mhz}{2^{25}}$, $\frac{50\,Mhz}{2^{26}}$, (the frequencies are chosen to make the counter action visible through the LEDs, except for the highest frequency which is

solely to verify the design in the ISE Simulator).
The frequencies are controlled using input signal FREQUENCY connected to three of the available four sliding switches (SW1, SW2 & SW3).
The output of the 4-bit counter is connected to four of the available seven LEDs (LD0, LD1, LD2, LD3).
An asynchronous reset is also provided to the design which is connected to one of the sliding switches (SW0).

Process

1. Complete the counter design to implement functionality as explained (provided).
2. Verify inputs and outputs with test bench waveform (provided).
3. Integrate the ChipScope into the counter design (explained in detail in the Implementation section).
4. Analyze the design using ChipScope (also explained in detailed in the next section).

Implementation

1. Right click on the top module of the design intended for verification or debugging, and select new source. Then select **ChipScope Definition and Connection File** as shown below.

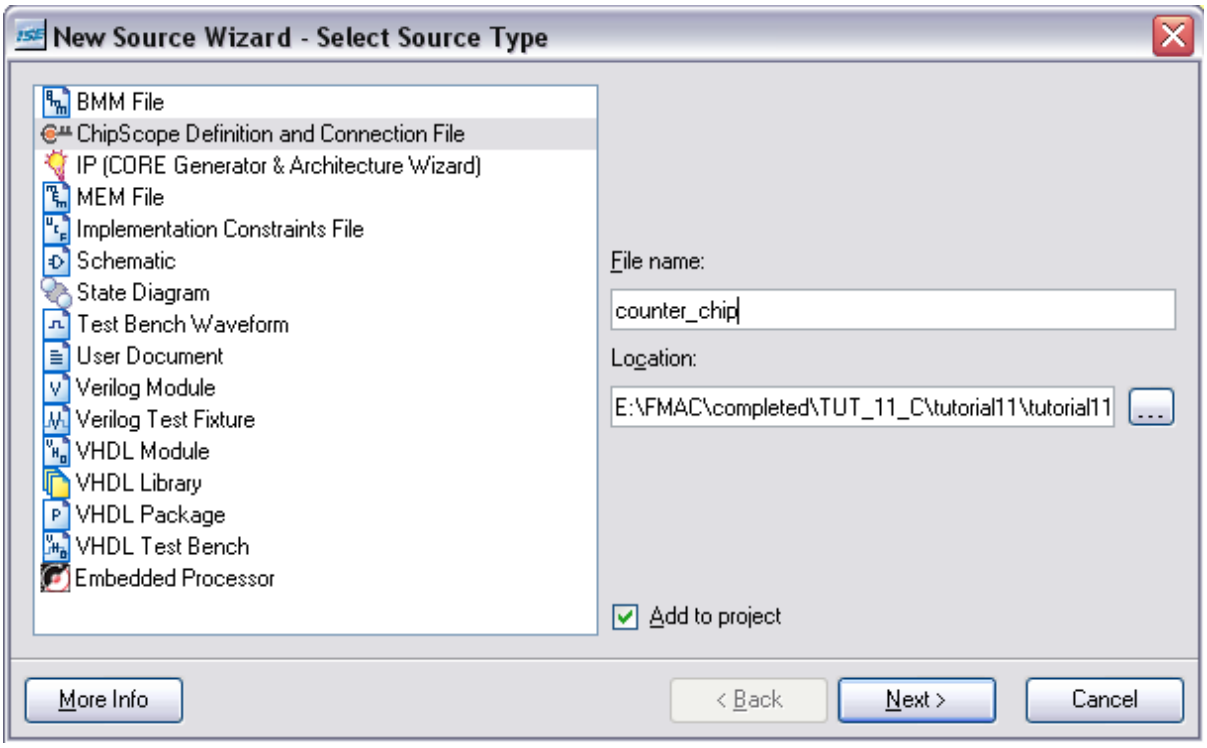


Figure 1: ChipScope Module Selection

2. Then give an appropriate name for the **file name**. Press **next** and select the **hierarchy level** at which the analysis is intended to be performed and then press **next** and then do **finish**.

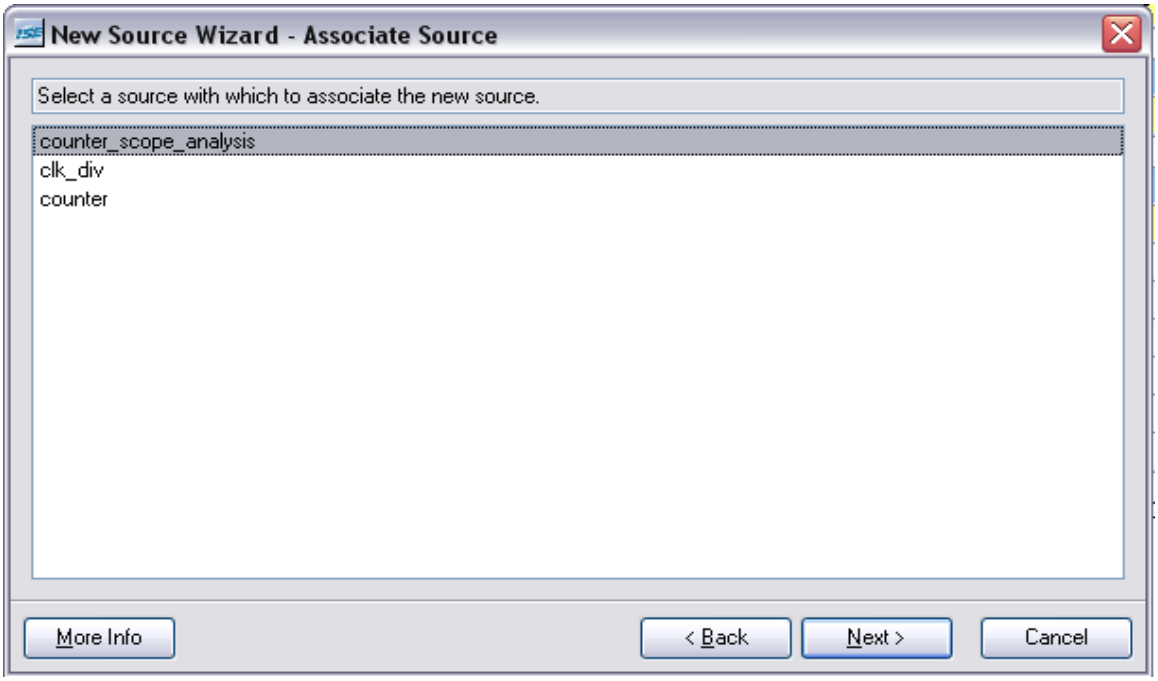


Figure 2: Hierarchy Selection

3. The first two steps will cause a new file (with file name as given) to be created in the source window under the project hierarchy as shown below.

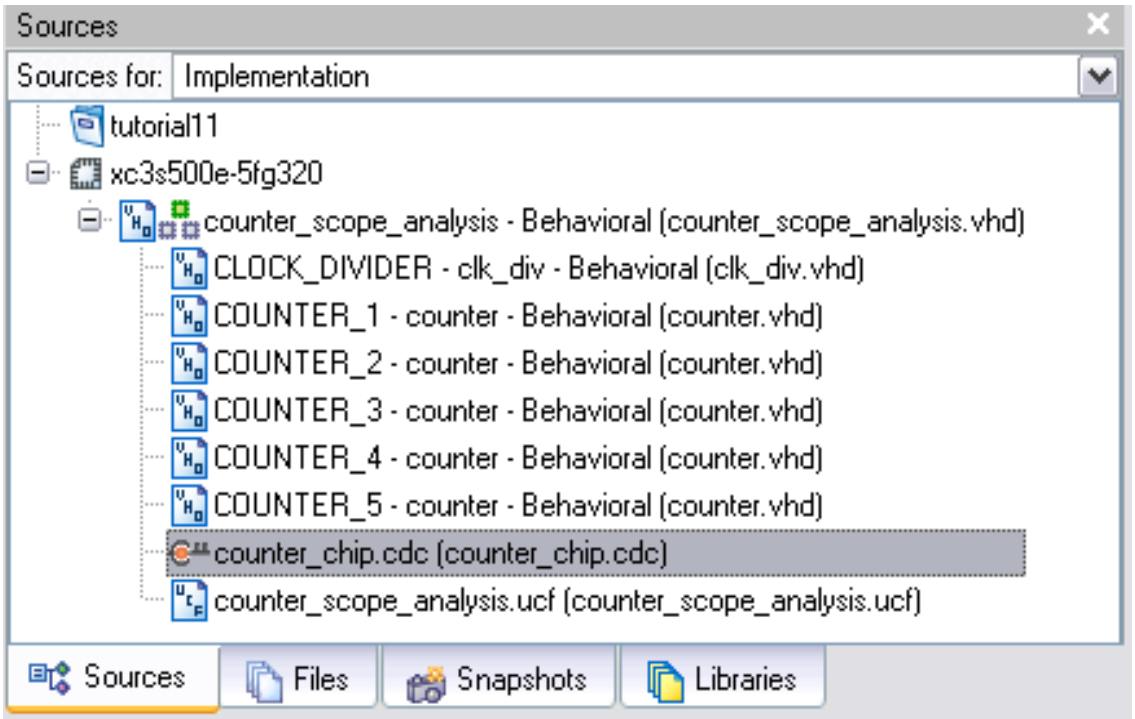


Figure 3: ChipScope Module Location

4. Double click on this new source file which cause the below window to pop up. Keep the default settings with **Use SRLs** and **Use RPMs** as checked.
   This will enable the tool to use Shift Register LUTs instead of flip flops and multiplexers thereby effectively reducing the size and improving the performance of the core generator. RPMs contain RLOC constraints which define the order and structure of the underlying design primitives. Use of RPMs will enable the tool to use relationally placed macros (like FMAP, HMAP, ROM, RAM, etc) allowing logic blocks to be placed relative to increase speed and use die resources efficiently substituting hard macros with an equivalent that can be simulated directly which again increases the core performance.

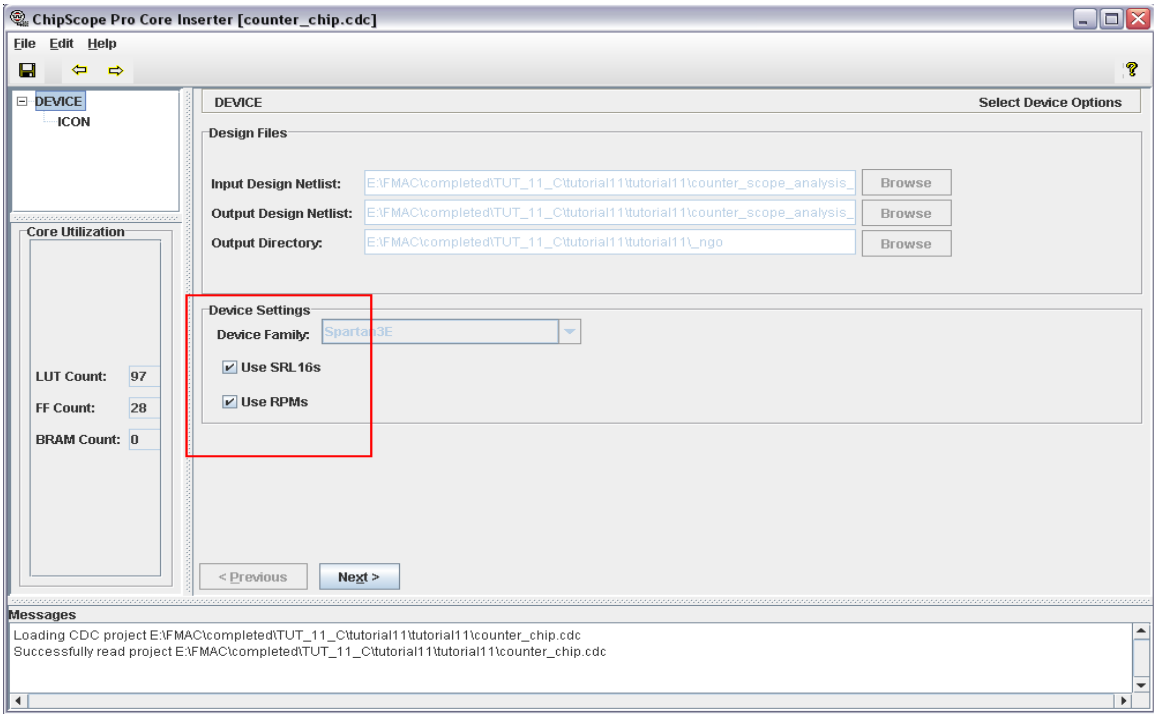


Figure 4: SRL and RPMs

5. Press next and again leave the default conditions i.e., keep the **Disable JTAG Clock BUFG Insertion** box unchecked.
   Disabling JTAG clock will cause the implementation tool to route the JTAG clock using normal routing resources instead of global clock routing resources. This might affect the high speed clock signals. So, unless the global resources are very scarce, it should not be disabled. But, disabling might introduce skew.

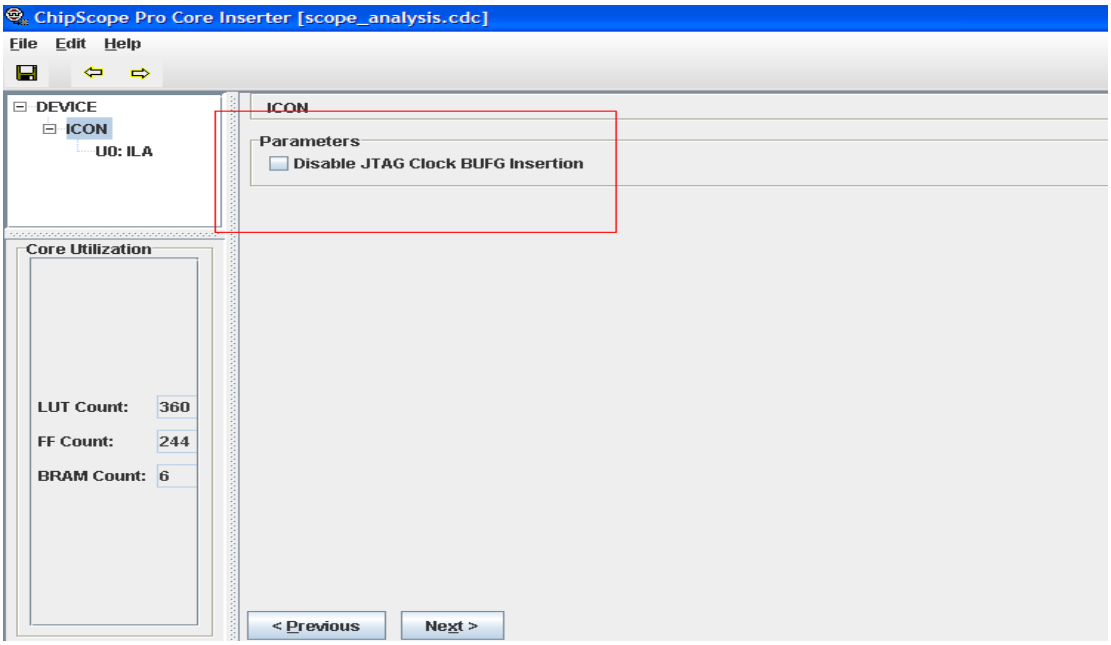


Figure 5: Global Clocks

6. Press **next** and then select the number of **trigger ports** and their respective **widths** depending on the design requirement.
   Triggers are those signals which initiate or trigger a certain sequence of actions influencing certain signals under consideration.
   Here, signal FREQUENCY is the only trigger taken into consideration (which is three bits wide) to control the counter output. Therefore, number trigger ports is set to 1 and the width is set to 3.

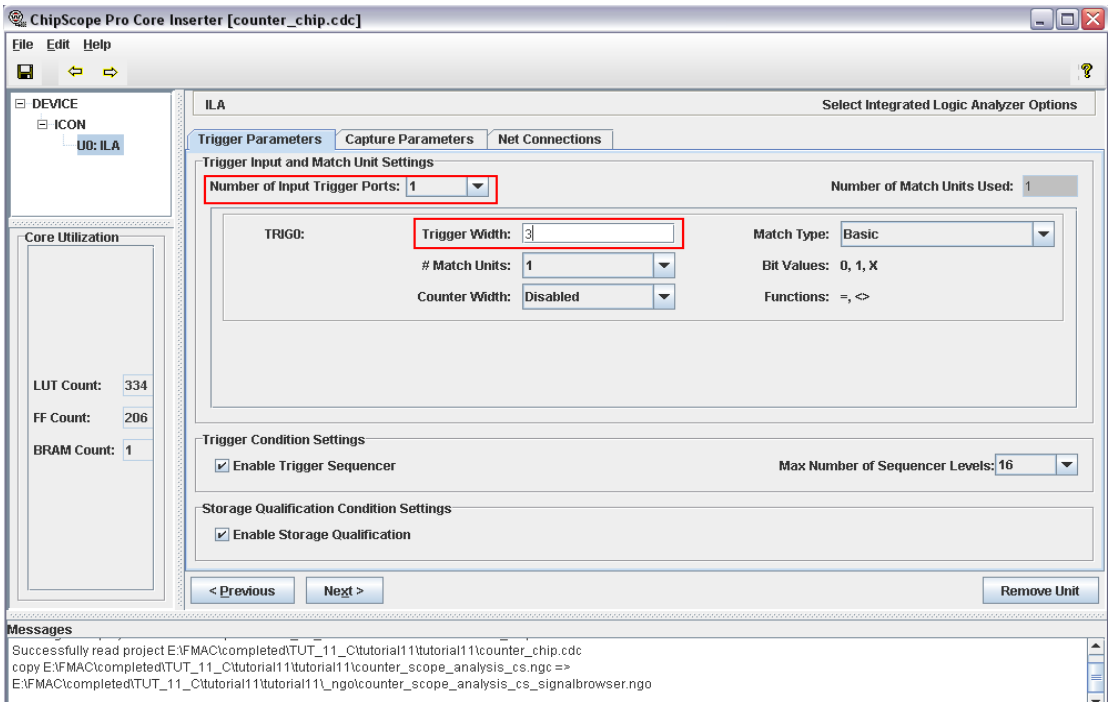


Figure 6: Trigger Options

7. Now **Match Type** should be selected. This defines the type of trigger one wants.
   For example: **Basic** mode, triggers depending on the specific value to which trigger is set.
   Range mode, triggers depending on the range of values in which the trigger is defined.
   Extended mode triggers depending on one or more occurrences of exact or range of trigger values to which trigger is set. A combinatorial logic (like AND/OR) or conditional logic (IF/THEN) between 2 or more signals can also be implemented into a trigger signal.
   Since FREQUENCY signal has definite values, **Basic** mode can be chosen for the Match Type in the present case.

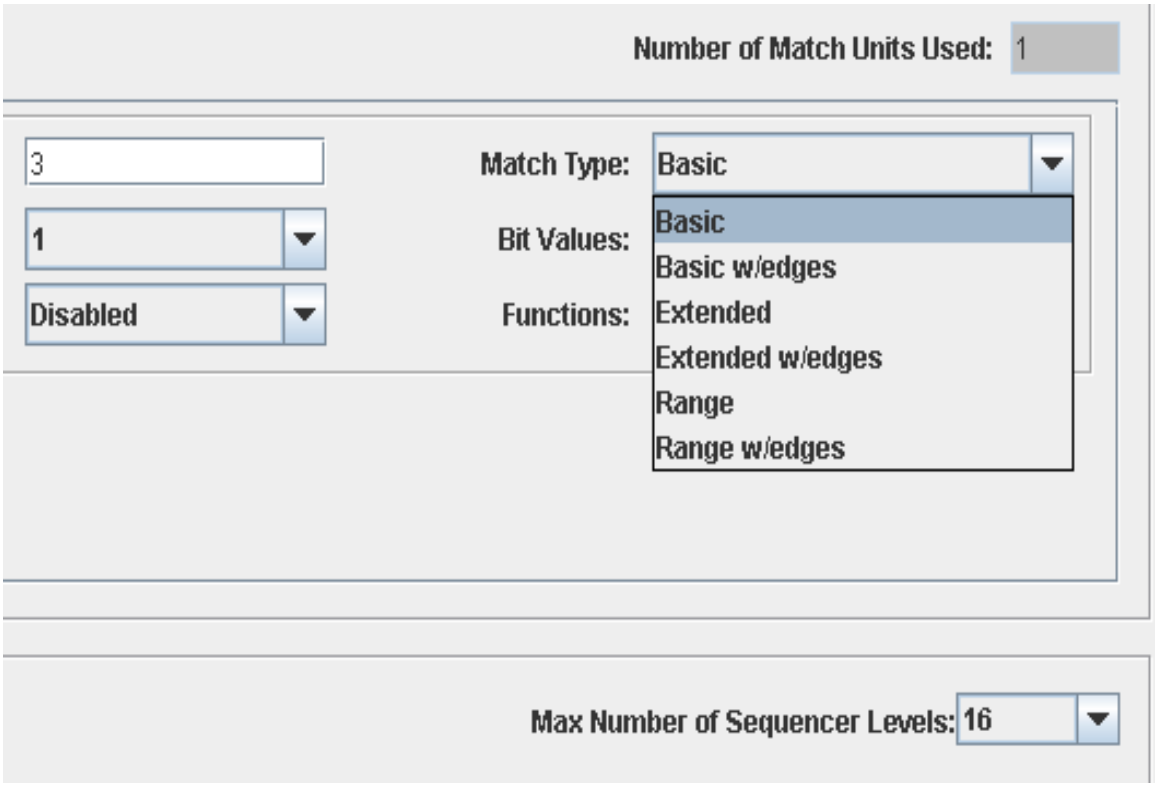


Figure 7: Match Type

8. Now uncheck or check the **Trigger Conditions Settings** i.e., **Enable Trigger Sequencer** and **Enable Storage Qualification** depending on the design requirements.
   Enable Trigger Sequencer can be used to enable a 16 level trigger sequencer which aids in configuring a multi level state machine to trigger upon a user defined traversal scheme of match units.
   Enable Storage Qualification can be used to filter data that is captured based on the user defined conditions that can be combined with trigger events.
   As the present trigger (FREQUENCY signal) is a simple and straight forward trigger, so both the boxes can be unchecked which saves little amount of logic space on the FPGA as shown below (**LUT and FF count**).

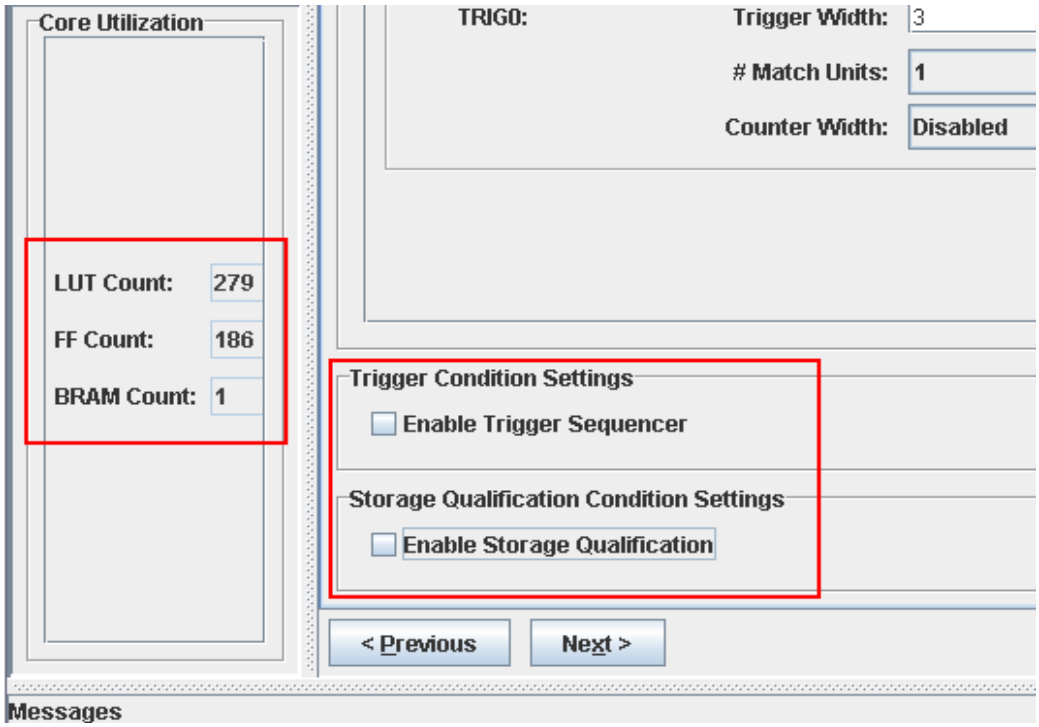


Figure 8: Trigger and Storage Settings

9. Press **next** and depending on the design requirement uncheck or check the **Data Same As Trigger** option.
   If data is not same as trigger then define the **Data Width**.
   The **Data Depth** is defined depending again on the requirements. It is recommended to put maximum limit as it can be adjusted during the analysis phase.
   Select the **Rising** or **Falling** edge of the clock signal depending on whichever edge desired to sample the data.
   As in the present design, the output data (Q) and trigger (FREQUENCY) are different the **Data Same As Trigger** icon is unchecked. Since the width of counter is 4-bit, data width is selected as 4. Rising edge is selected for clock edge for sampling data.

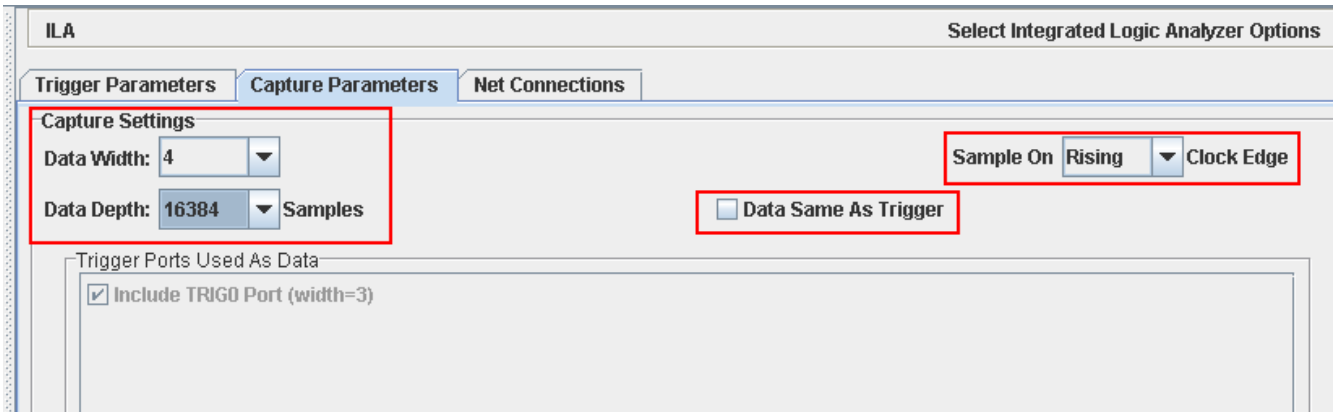


Figure 9: Data Options

10. Press **next** and then press **Modify Connections**.

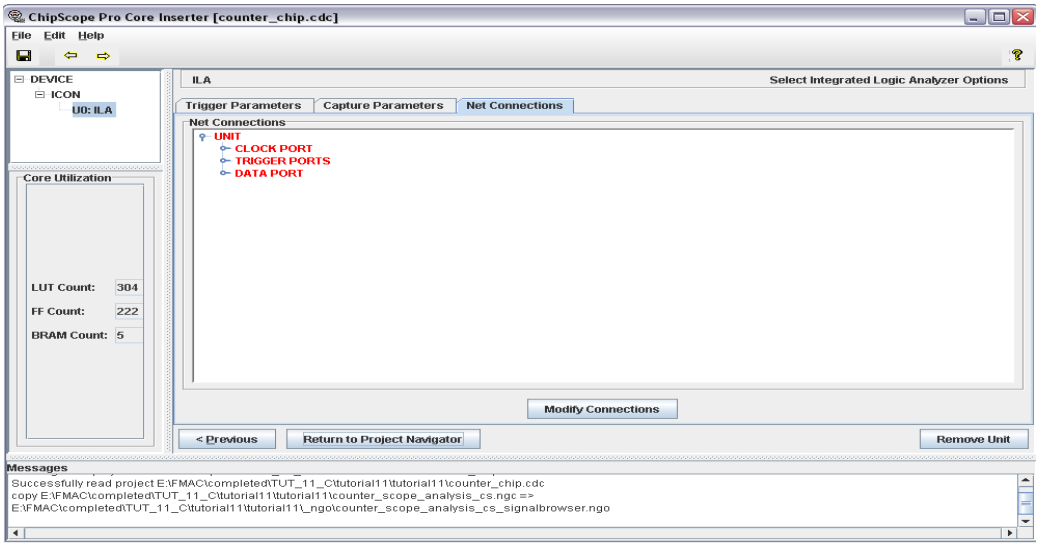


Figure 10: Net Connections

11. Once **Modify Connections** is clicked, the below shown window will pop up. Select the appropriate signals from the list of nets and make connections to the respective clock, trigger and data signals.

    Note: Sometimes certain nets do not show up in the list, which means that during optimizations the tool has found that there is more than one net with same logic. As a result it optimizes it to a single net thereby resulting in absence of few wanted nets. This would require more detailed analysis of the design or modification of the same to make the necessary connections for debugging.

    Once all the connections are made press **OK**.
    Then press **Return to Project Navigator** and **Save Project** changes.

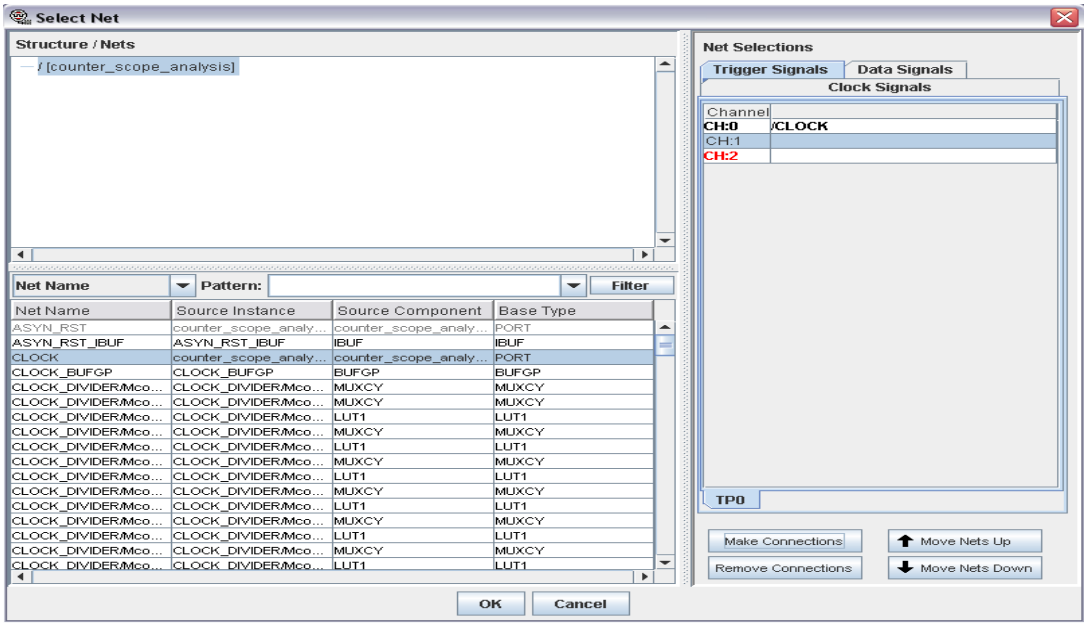


Figure 11: Net Selections

12. Now re-implement the design using the **Implement Design** icon and then do the configuration using the **Configure Target Device** icon.
    Once these steps are done successfully one is ready to analyze the design using the **Analyze Design Using ChipScope** icon (present along with the implement design and configure design icons in the processes window of the ISE tool). By double clicking this icon the following window will pop up.

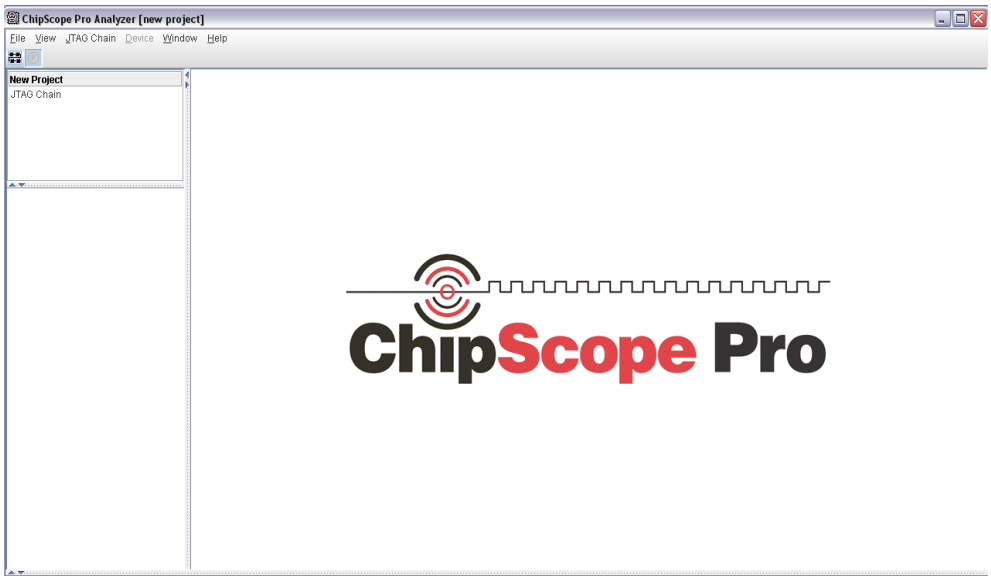


Figure 12: ChipScope Analyzer

In the top left hand corner there is an icon which is used to open the JTAG chain, click on this icon and the following window pops up. It shows all the devices it has found in the JTAG chain, press **OK**.

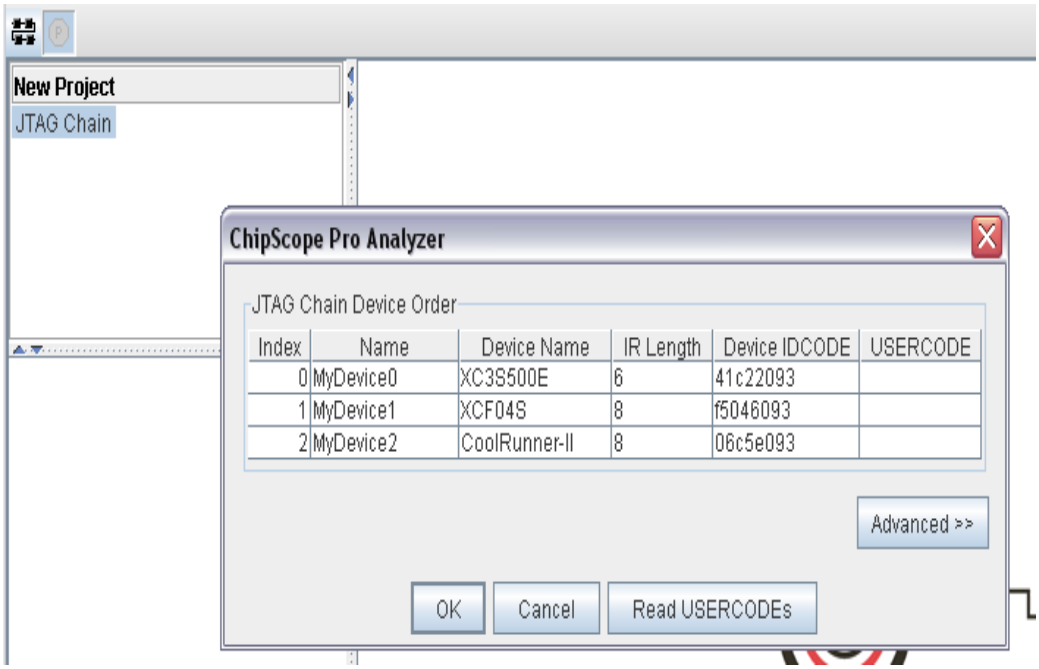


Figure 13: JTAG Chain

13. Once the list is accepted by pressing **OK** the following window shows up. Set the trigger to design criteria, adjust the data depth to the amount needed and then hit the play button (all of them are marked in red boxes). One can observe all the output signal variations and do the verification as needed.

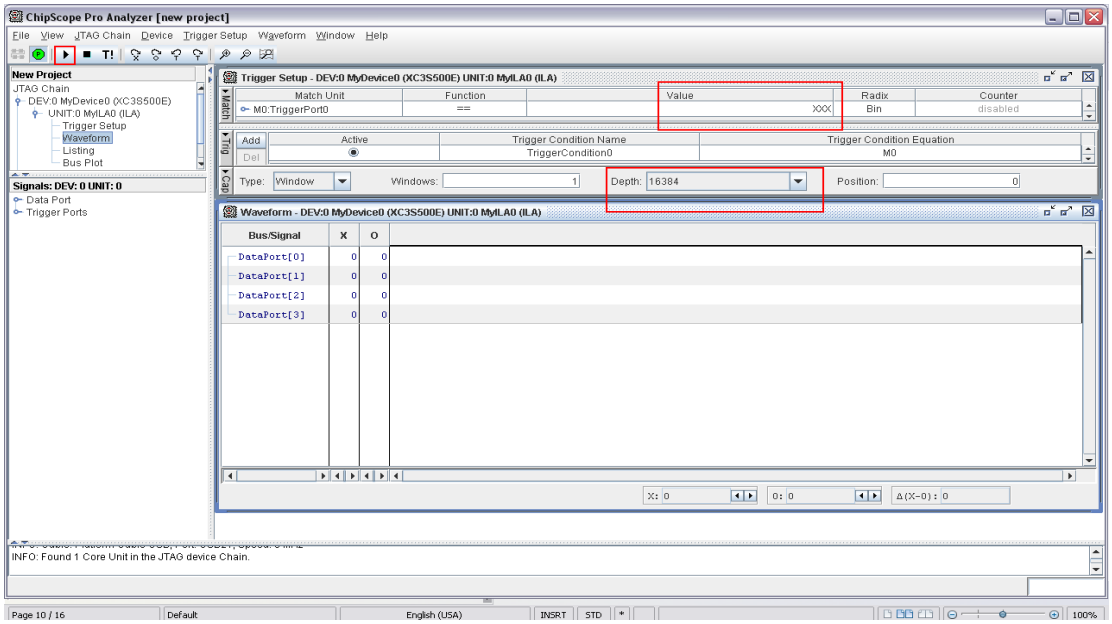


Figure 14: Trigger and data setup

14. The data waveform for the sample counter when trigger is set to “000” is as shown below.
Note: To observe more changes the sample clock of the ChipScope has been set to the LSB of the 4-bit counter.

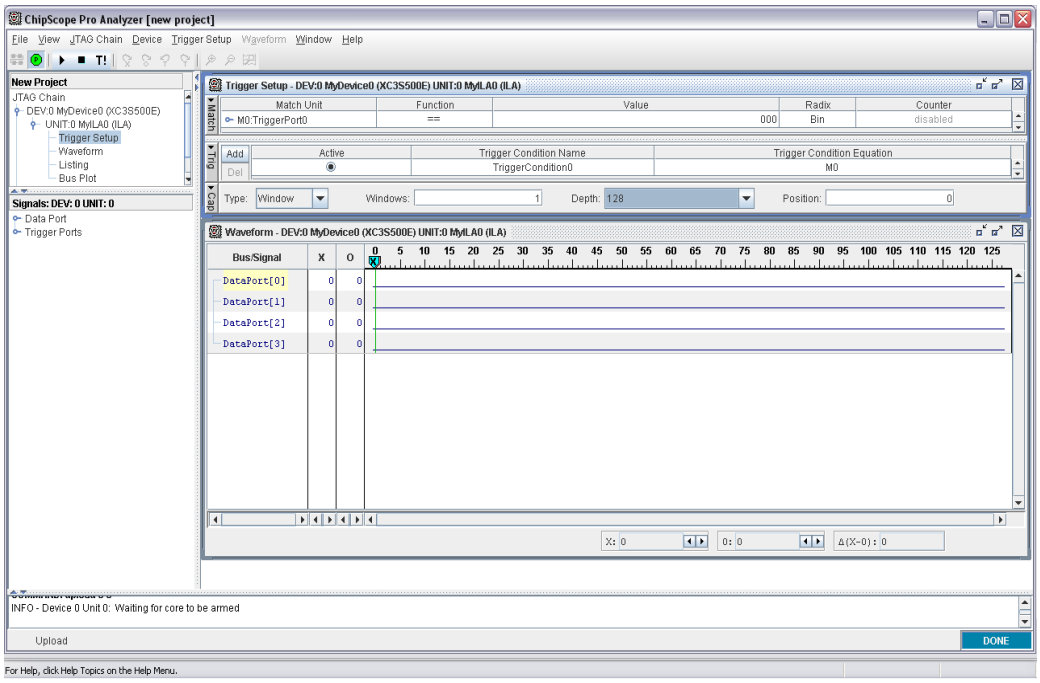


Figure 15: Waveforms

15. Sometimes using the bus plot will be very beneficial to observe certain output signals. This can be done by selecting all the data ports and tying them into a single bus as shown below.

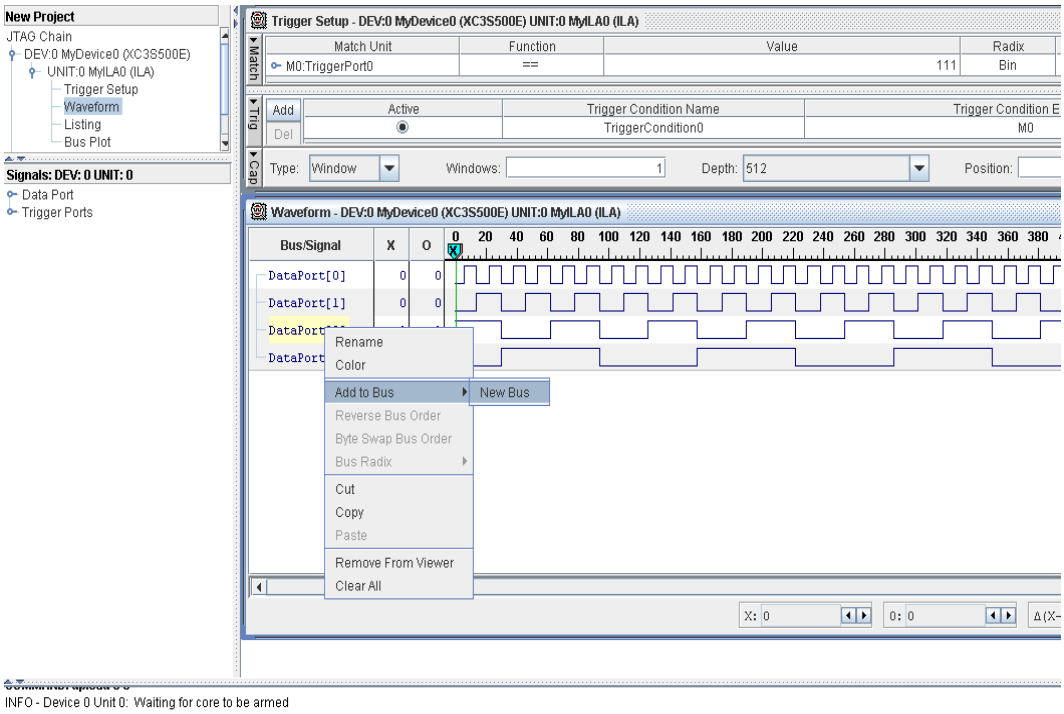


Figure 16: Bus Port Creation

16. The bus plot can be viewed by clicking on the bus plot icon as shown below and selecting the appropriate bus signal intended for viewing.

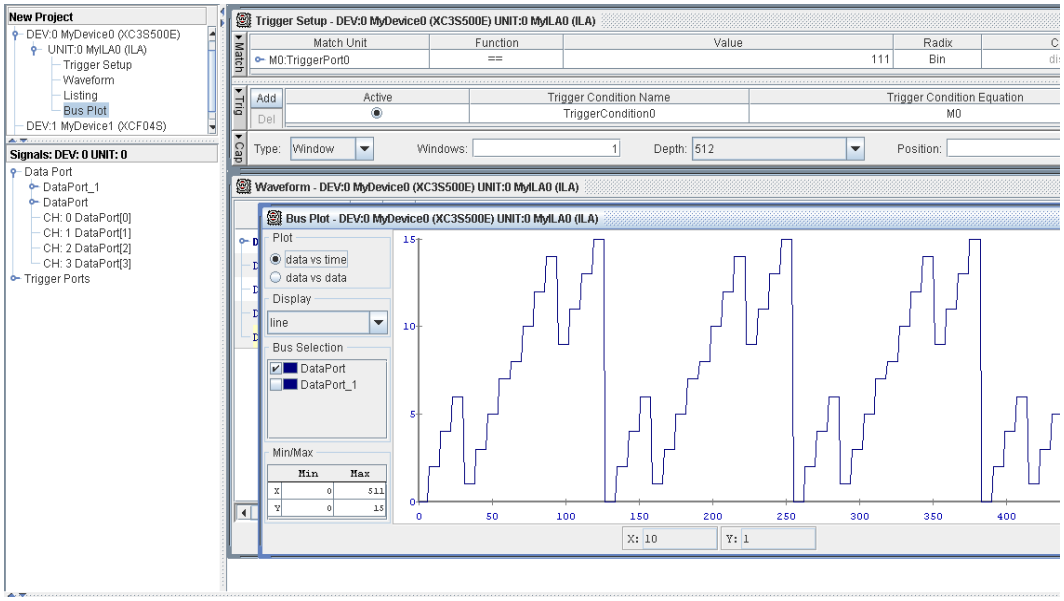


Figure 17: Bus Plot

17. This is the signal that is directly being tapped from the FPGA on the board unlike the ISE Simulator signals which are behavioral simulated (can be observed using the test bench provided along with the sample project). So one gets to observe how the actual signal is behaving on the board which is very essential to resolve timing issues in high speed designs.

-Merits and Demerits

1. The major advantages of ChipScope compared to external logic analyzers are:

   - Reduces the probe delays in analyzing the signals.
   - Reduces the circuit performance degradation caused due to probing.
   - Portable and convenient to analyze circuitry on FPGA.
   - Cost – logic analyzers can cost over $50,000.

2. There are few limitations of ChipScope as compared to external logic analyzers which are:

   - Availability of resources on the FPGA (which comes into picture for large complex designs).

   A simple example is the amount of space occupied by the present counter design with and without ChipScope logic integrated respectively is:

Device Utilization Summary				
Logic Utilization	**Used**	**Available**	**Utilization**	**Note(s)**
Number of Slice Flip Flops	265	9,312	2%	
Number of 4 input LUTs	143	9,312	1%	
Logic Distribution				
Number of occupied Slices	245	4,656	5%	
Number of Slices containing only related logic	245	245	100%	
Number of Slices containing unrelated logic	0	245	0%	
Total Number of 4 input LUTs	270	9,312	2%	
Number used as logic	143			
Number used as a route-thru	88			
Number used as Shift registers	39			
Number of bonded IOBs	9	232	3%	
IOB Flip Flops	3			
Number of Block RAMs	5	20	25%	
Number of GCLKs	2	24	8%	
Number of BSCANs	1	1	100%	
Number of RPM macros	9			
Total equivalent gate count for design	333,964			
Additional JTAG gate count for IOBs	432			

Figure 18: Device Utilization with ChipScope

Device Utilization Summary				
Logic Utilization	**Used**	**Available**	**Utilization**	**Note(s)**
Number of Slice Flip Flops	47	9,312	1%	
Number of 4 input LUTs	32	9,312	1%	
Logic Distribution				
Number of occupied Slices	32	4,656	1%	
Number of Slices containing only related logic	32	32	100%	
Number of Slices containing unrelated logic	0	32	0%	
Total Number of 4 input LUTs	58	9,312	1%	
Number used as logic	32			
Number used as a route-thru	26			
Number of bonded IOBs	9	232	3%	
Number of GCLKs	1	24	4%	
Total equivalent gate count for design	727			
Additional JTAG gate count for IOBs	432			

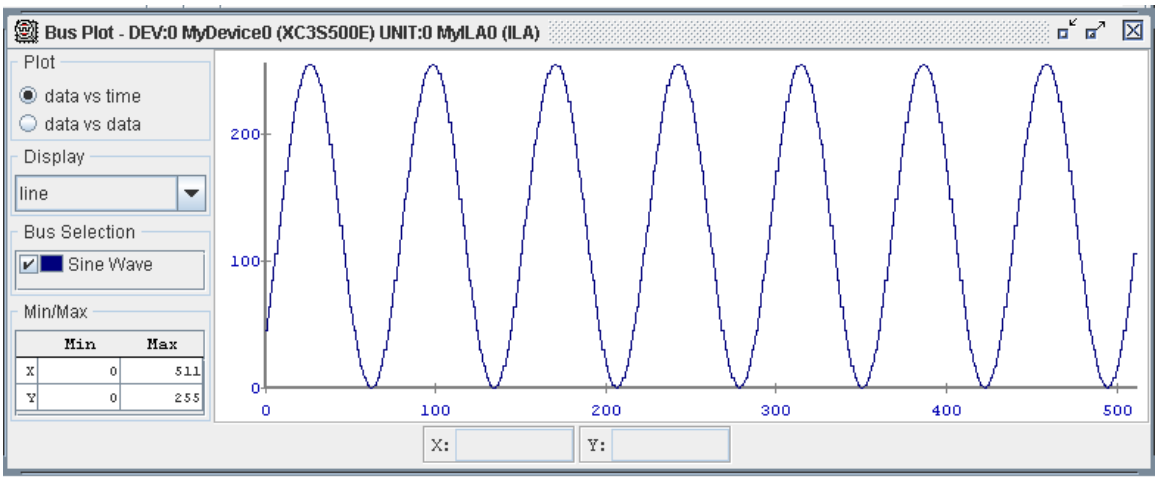
Figure 19: Device Utilization without ChipScope

- Sampling rate cannot be faster than the design clock frequency (making glitch detection not possible).

Exercise

Try to develop a sine wave generator using LUT, CORDIC or any other technique and verify the same using ChipScope Pro tool.

Hint: Using bus plot as shown below:



-Author Bio

Name: Vallabh Srikanth Devarapalli
Graduate student at UNM in ECE department.
E-mail: vsdevara@unm.edu.

Updated By:
Brian Zufelt
Undergraduate student at UNM in ECE department.