

# Adding the ILA Core to an Existing Design Lab

## Introduction

This lab consists of adding a ChipScope™ Pro software ILA core with the Core Inserter tool and debugging a non-functioning design.

The files for this lab are for the Spartan™-3E 1600 device but can also be used for other Xilinx FPGA devices with minor modification.

## Objectives

After completing this lab, you will be able to:

- Use the Core Inserter to add ILA cores to an existing design
- Define and use the ILA core within a design
- Use the ChipScope Pro Analyzer tool to configure an FPGA, set basic trigger conditions, analyze, and debug a design

## Procedure

Software requirements:

- ChipScope Pro 11.3 software
- ISE™ 11.3 software

Hardware requirements:

- Spartan-3E 1600 FPGA MicroBlaze Embedded Development Kit
- USB configuration cable (included in the kit)
- Power supply for the Spartan-3E 1600 FPGA board (included the kit)

The design used for this lab is a digital clock design targeting the Spartan-3E 1600 FPGA MicroBlaze Development Kit board. The design generates a digital clock and displays the output using the LCD display available on the board.

This lab is separated into steps, followed by general instructions and supplementary detailed steps allowing you to make choices based on your skill level as you progress through the lab.

If you need help completing a general instruction, go to the detailed steps below it, or if you are ready, simply skip the step-by-step directions and move on to the next general instruction.

This lab comprises six primary steps: You will configure the design using the ChipScope Pro software, insert an ILA core into the design; set up the ChipScope Pro Analyzer tool interface, capture data, debug the design and, finally, observe the amount of resources used.

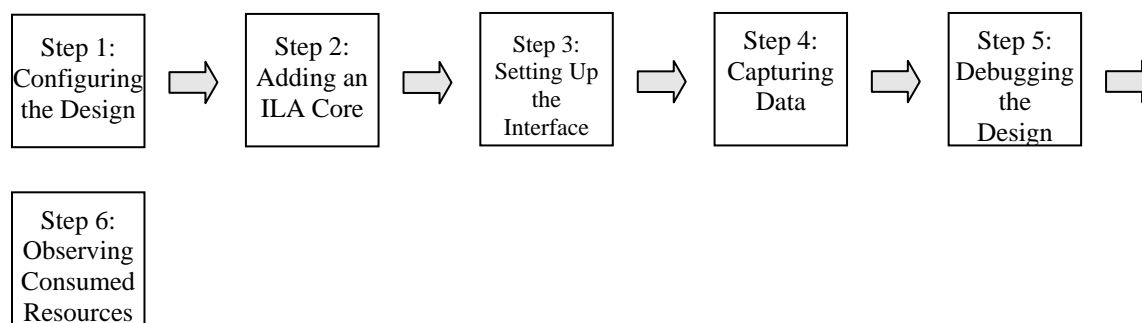
Before beginning, verify that your software is properly installed.

- ChipScope Pro software: Select **Start** → **Programs** → **Xilinx ISE Design Suite 11** → **ChipScope Pro** → **Analyzer**. Select **Help** → **About** to verify version 11.3
- Project Navigator in the ISE software: Select **Start** → **Programs** → **Xilinx ISE Design Suite 11** → **ISE** → **Project Navigator**. Select **Help** → **About** to verify version 11.3

Verify that the hardware is set up properly by checking the following settings.

- Connect the USB programming cable (not the Platform Cable USB, but just the USB cable) from your machine to the board
- Power the board on—the PC should detect the new Xilinx programming device (install the drivers if prompted)

## General Flow for this Lab



## Configuring the Design

## Step 1

**1-1. Implement the design (*clock\_part1.ise*) located in the *C:\training\chipscope\_pro\labs\InserterFlow-VHDL* (VHDL users) or *C:\training\chipscope\_pro\labs\InserterFlow-Verilog* (Verilog users) folder. Use the ChipScope Pro Analyzer tool to configure the device.**

**1-1-1.** Select **Start** → **Programs** → **Xilinx ISE Design Suite 11** → **ISE** → **Project Navigator** to open the Project Navigator in the ISE software.

**1-1-2.** From the Project Navigator, select **File** → **Open Project**.

- **VHDL users:** Browse to *C:\training\chipscope\_pro\labs\InserterFlow-VHDL*
  - **Verilog users:** Browse to *C:\training\chipscope\_pro\labs\InserterFlow-Verilog*
- Click **clock\_part1.xise** and click **Open**.

**1-1-3.** Right-click *exampleInserter.vhd|.v* and select **Set As Top Module** to ensure that the design is the top module by. Right-click *exampleInserter.vhd|.v* and select **Open**.

**1-1-4.** In the Processes window, double-click **Generate Programming File**. If a webtalk window opens, just close it.

**1-1-5.** When the bitstream has been generated, double-click **Analyze Design Using ChipScope** in the Processes window.

**1-1-6.** Select **JTAG Chain** → **Xilinx Platform USB Cable**.

**1-1-7.** In the dialog box that opens, note the USB-JTAG speed and port number and click **OK**.

This is one way to initialize the JTAG chain.

**1-1-8.** Right-click the **XC3S1600E** device in the Project pane and select **Configure**.

**1-1-9.** Click **Select New File** from the JTAG Configuration area of the dialog box that opens.

**1-1-10** Select *exampleInserter.bit*. Click **Open** and **OK**.

Notice the blue status bar in the bottom-right corner of the screen as it configures.

**1-2. This is a simple clock design. If the design were working properly, the LEDs would increment at one Hertz.**

**However, the design is not working properly; confirm this by observing that the LEDs on the Spartan-3E 16000 FPGA starter board are not incrementing.**

**1-2-1.** Before you add a ChipScope Pro ILA core to this design, familiarize yourself with the resources for this clock design. From within the Project Navigator, in the Sources window, double-click *exampleInserter.vhd*.v to open the source file.

**1-2-2.** In the Processes window, double-click **Design Summary/Reports** and answer Question 1.

### Question 1

Record the resource utilization for this design from the Design Summary within the Project Navigator in the Prior to Adding ChipScope Pro Cores column. Leave the remaining two columns blank for now.

Type of Resource	Prior to Adding ChipScope Pro Cores	ChipScope Pro Core Resources	After Insertion
Number of Slice Flip-Flops			
Number of 4-input LUTs			
Number of RAMB16s			

**1-2-3.** Exit the ChipScope Pro software. Do not save any changes.

## Adding an ILA Core

### Step 2

**2-1. Use the Core Inserter tool in the ChipScope Pro software to add an ILA core to this design via the Create New Source process. Create three match units of 11, 11, and 7 signals respectively. Assign hertz\_count and hertz\_en to the first match unit, khertz\_count and khertz\_en to the second, and, finally, mhertz\_count and mhertz\_en to the last match unit. Each match unit can be set with basic triggering.**

**2-1-1.** From the Project Navigator Sources window, select *exampleInserter.vhd*.v.

**2-1-2.** Select **Project** → **New Source**.

**2-1-3.** In the New Source dialog box, select **ChipScope Definition and Connection** and enter *Clock\_ILA* in the file name field.

**2-1-4.** Click **Next**, click **Next**, and click **Finish**.

**2-1-5.** Double-click *Clock\_ILA.cdc* in the Sources window to launch the ChipScope Pro software.

**2-1-6.** In the ChipScope Pro Core Inserter window, observe the files that will be created. Confirm that the Spartan-3E device family has been specified. Confirm that SRL16s and RPMs have been checked.

**2-1-7.** Click **Next**.

**2-1-8.** Confirm that U0:ILA was created by looking in the upper-left pane. If it was not, click **New ILA Unit** at the bottom of the dialog box. Confirm that an ILA core has been added to the device tree in the left-hand window (U0: ILA). Click **Next**.

2-2. In the Trigger Parameters tab, choose the number of trigger inputs and specify the parameters for each of these trigger ports as outlined in Figures 1 and 2.

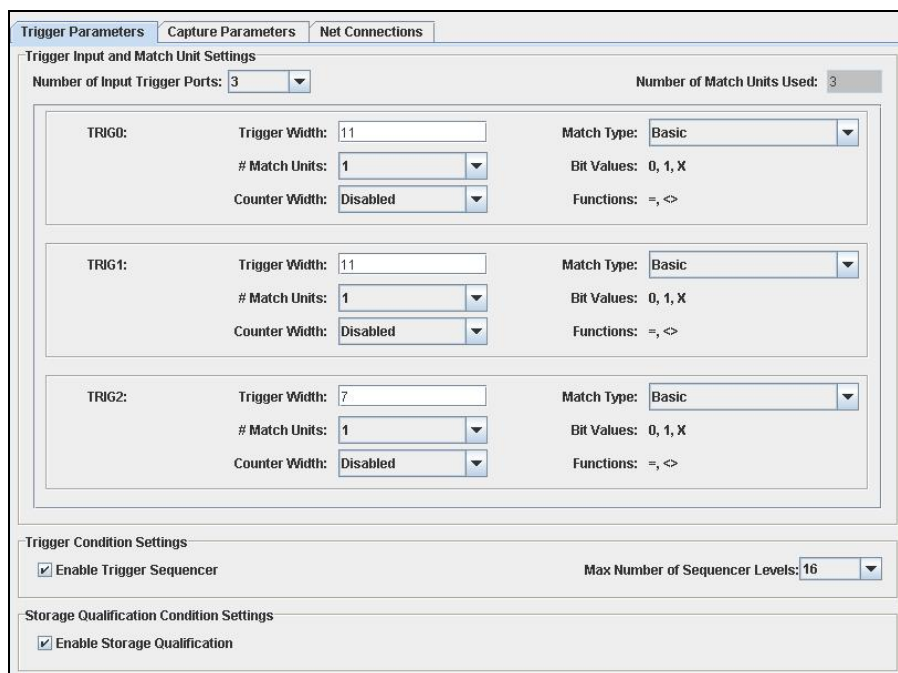


Figure 1. Trigger Parameters Tab

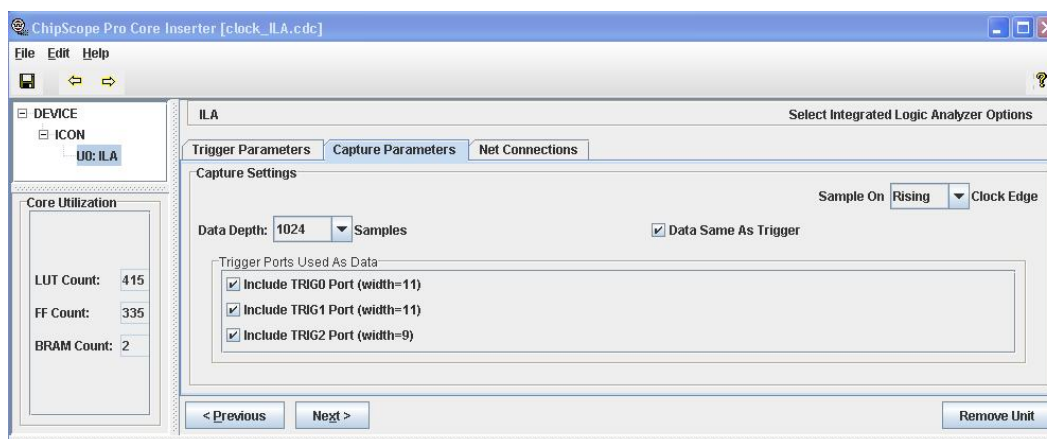


Figure 2. Capture Parameters Tab

2-2-1. In the Trigger Parameters tab, select 3 from the Number of Input Trigger Ports drop-down list.

2-2-2. Define the three trigger ports as follows.

- TRIG0 with a trigger width of 11, 1 match unit, counter width disabled, and with a basic match type
- TRIG1 with a trigger width of 11, 1 match unit, counter width disabled, and with a basic match type
- TRIG2 with a trigger width of 7, 1 match unit, counter width disabled, and with a basic match type

2-2-3. Leave the **Enable Trigger Sequence** and **Enable Storage Qualification** options selected.

The Trigger Parameters tab should match **Figure 3**.

ILA Select Integrated Logic Analyzer Options

Trigger Parameters Capture Parameters Net Connections

Trigger Input and Match Unit Settings

Number of Input Trigger Ports: 3 Number of Match Units Used: 3

TRIG0: Trigger Width: 11 Match Type: Basic  
 # Match Units: 1 Bit Values: 0, 1, X  
 Counter Width: Disabled Functions: =, <>

TRIG1: Trigger Width: 11 Match Type: Basic  
 # Match Units: 1 Bit Values: 0, 1, X  
 Counter Width: Disabled Functions: =, <>

TRIG2: Trigger Width: 7 Match Type: Basic  
 # Match Units: 1 Bit Values: 0, 1, X  
 Counter Width: Disabled Functions: =, <>

Trigger Condition Settings

Enable Trigger Sequencer Max Number of Sequencer Levels: 16

Storage Qualification Condition Settings

Enable Storage Qualification

< Previous Next > Remove Unit

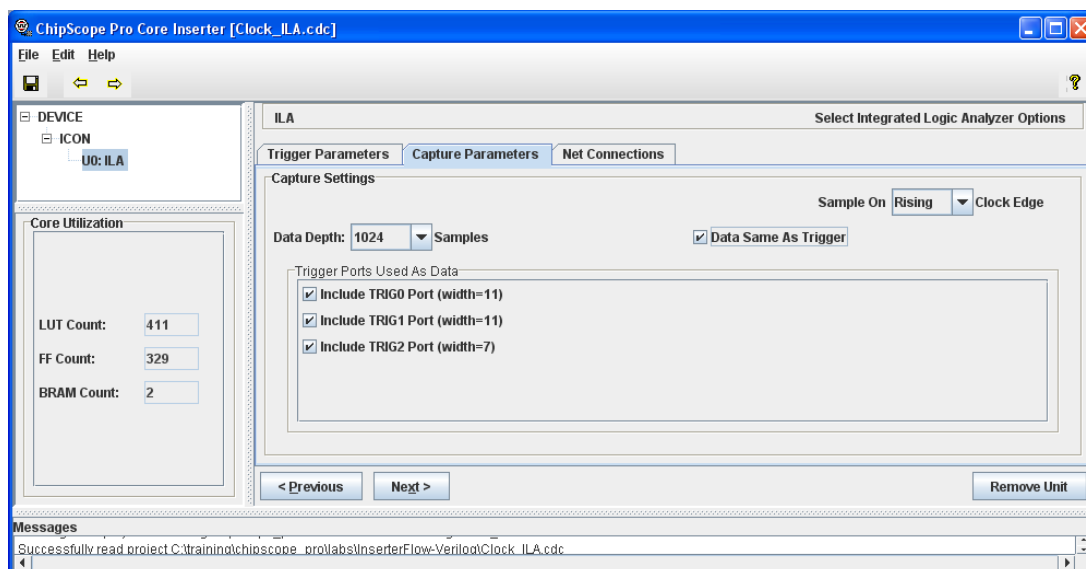
**Figure 3. Trigger Parameters Tab**

**2-2-4.** Click **Next**.

**2-2-5.** In the Capture Parameters tab, select the following (NOTE...there is a bug in the gui. If you click in the Capture Settings pane you will de-assert the settings. If this happens click again in the pane to reverse it.)

- o Data depth of 1024
- o Sample on rising clock edge and data the same as trigger
- o Leave TRIG0, TRIG1, and TRIG2 selected

The Capture Parameters tab should match **Figure 4**.



**Figure 4. Capture Parameters Tab**

- 2-2-6. Observe the block RAM and other resources used in the Core Utilization pane in the lower-left hand pane.

## Question 2

In the table provided for Question 1, enter the resource utilization in the ChipScope Pro Core Resources column for the following: LUT count, flip-flop count, and block RAM count.

- 2-2-7. Click Next.

## 2-3. In the Net Connections tab, connect the clock signal clk50 to CP0 CH:0.

- 2-3-1. In the Net Connections tab, double-click **Clock Port** to open the Select Net dialog box.

- 2-3-2. Enter *clk\** in the Patter filter box. Click **Filter**.

- 2-3-3. Select **clk50** and click **Make Connections** in the lower-right hand corner. You will notice clk50 in is grayed-out. This is because it is a pad and you can only assign signals attached to a buffer.

- 2-3-4. Observe that the design clock net has now been associated with the ILA core.

## 2-4. In the Trigger/Data Signals tab of the Net Selections area, make the following connections.

### TP0 (trigger port 0) tab:

- Assign **hertz\_count<0:9>** to **CH:0** through **9**
- Assign **hertz\_en** to **CH:10**

### TP1 (trigger port 1) tab:

- Assign **khertz\_count<0:9>** to **CH:0** through **9**
- Assign **khertz\_en** to **CH:10**

### TP2 (trigger port 2) tab:

- Assign **mhertz\_count<0:5>** to **CH: 0** through **5**

- Assign `mhertz_en` to CH:6

The Net Connections tab should match Figure 5.

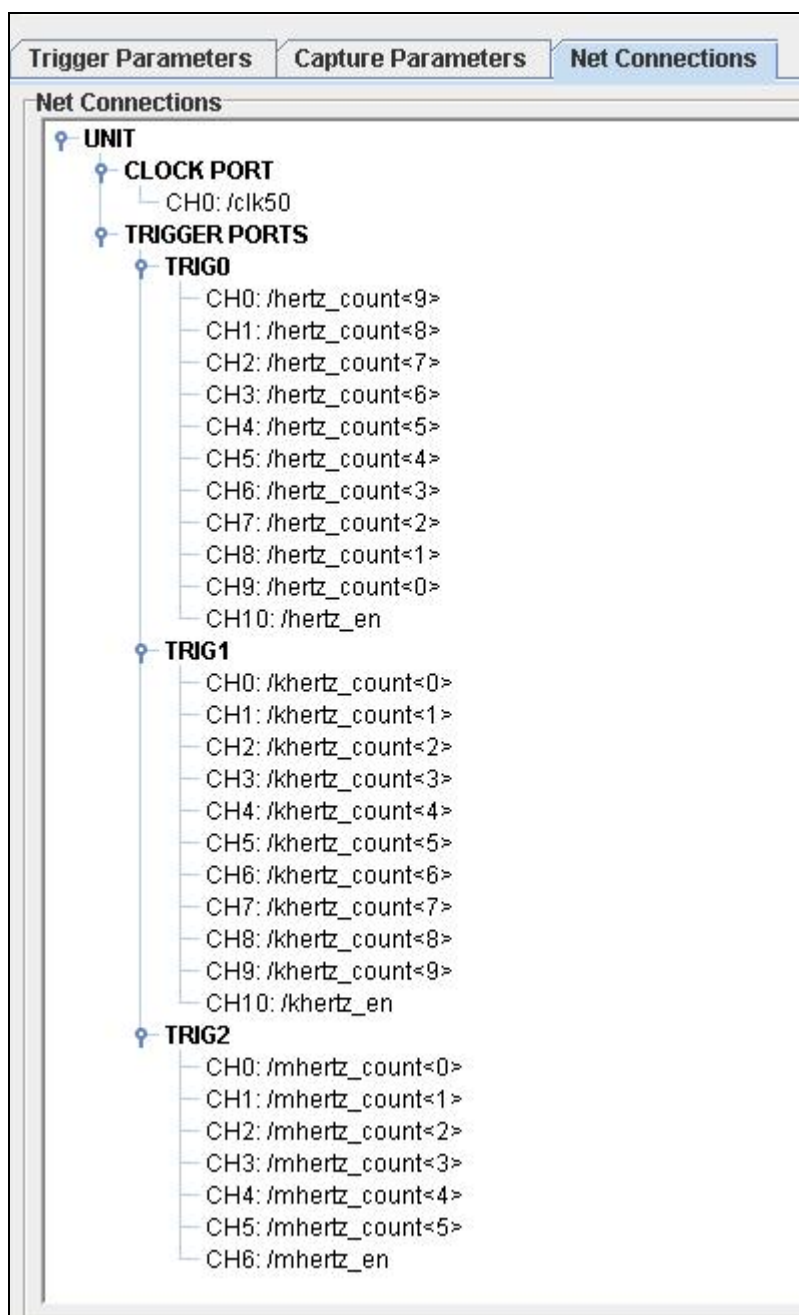


Figure 5. Net Connections Tab

- 2-4-1. Delete `clk*` in the Patter filter box. Click **Filter**

Click the **Net Name** column to filter the bus signal incrementally (for example, 0 through 9 versus 9 down to 0).

- 2-4-2. From the Net Selections pane (right hand side of gui), click the **Trigger/Data Signals** tab and click the **TP0** tab at the bottom.

Notice that TP0 (representing Trigger Port 0) is now available.

- 2-4-3. In the pattern box, enter *hertz\_count\** and click **Filter**. Using the **Shift** key, select *hertz\_count<0>* through *hertz\_count<9>*.
- 2-4-4. Click **Make Connections** in the lower-right corner of the window. If the connections are out of order simply select one of the connections and use the **Move Nets Up** or **Move Nets Down** buttons to connect the net to appropriate channel. The connections should match Figure 5.
- 2-4-5. In the Pattern filter box, enter *hertz\_en* and click **Filter**.
- 2-4-6. Select *hertz\_en* in the Filter nets window.
- 2-4-7. Click **Make Connections**.
- 2-4-8. Repeat detailed steps 2-4-2 through 2-4-7 to make the following connections.

TP1 (trigger port 1) tab:

- Assign *khertz\_count<0:9>* to CH:0 through 9
- Assign *khertz\_en* to CH:10

TP2 (trigger port 2) tab:

- Assign *mhertz\_count<0:5>* to CH: 0 through 5
- Assign *mhertz\_en* to CH:6

## 2-5. Return to the Project Navigator in the ISE software and create a new bitstream.

- 2-5-1. Click **OK** in the Net Connections window.
- 2-5-2. In the ChipScope Pro Core Inserter window, click **Return to Project Navigator** at the bottom of the window.
- 2-5-3. Click **Yes** to save the project and return to the Project Navigator.
- 2-5-4. In the Sources window, select *exampleInserter.vhd\,v*. In the Processes window, double-click **Generate Programming File** to generate a new bit file.

## 2-6. Configure the device by using the ChipScope Pro Analyzer tool.

- 2-6-1. In the Processes window, double-click **Analyze Design Using Chipscope**.
- 2-6-2. Select **JTAG Chain** → **Xilinx Platform USB Cable**.
- 2-6-3. In the dialog box that opens, click **OK**.

A new dialog box opens, confirming the JTAG devices for the Spartan-3 FPGA starter board.

- 2-6-4. Click **OK**. Right-click the **XC3S1600E device** in the Project pane and select **Configure**.
- 2-6-5. Click **Select New File** from in the JTAG Configuration region of the dialog box that opens.
  - **VHDL users:** Browse to *C:\training\chipscope\_pro\Labs\InserterFlow-VHDL*
  - **Verilog users:** Browse to *C:\training\chipscope\_pro\labs\InserterFlow-Verilog*
- 2-6-6. Select *exampleInserter.bit* and click **Open**. Click **OK**. This will program the device with the new bitstream that includes the ChipScope cores.
- 2-6-7. Observe the configuration status bar in the lower-right corner of the ChipScope Pro Analyzer tool display.

After the device has been configured, notice that the design now contains an ILA core named MyILA0 and a Trigger Setup and Waveform display are available.

## Setting Up the Interface

## Step 3

**3-1. Note that within the Waveform display, the channels have the names DataPort[0]...DataPort[28]. To set up the ChipScope Pro Analyzer tool interface, import the CDC file that you generated when you defined the ILA core.**

**3-1-1.** Select **File** → **Import**.

**3-1-2.** In the Signal Import dialog box, click **Select New File**. Select *Clock\_ILA.cdc* and click **Open**. Click **OK** in the Signal Import dialog box.

**3-1-3.** Observe that each channel now has the signal net name you associated.

**3-1-4.** In the Waveform window, using the **Shift** key, select *hertz\_count<0>* through *hertz\_count<9>*.

**3-1-5.** Right-click *hertz\_count<0-9>* and select **Move to Bus** → **New Bus**.

**3-1-6.** Observe that a new bus has been created named *hertz\_count*.

**3-1-7.** Select *hertz\_count<0-9>* in the Waveform window and press **Delete**.

**3-1-8.** Group *khertz\_count<0-9>* and *mhertz\_count<0-5>* using detailed steps 3-1-4 through 3-1-5 above.

**3-1-9.** Right-click the *hertz\_count* bus in the Waveform window and select **Bus Radix** → **Unsigned Decimal**. Click **OK** in the Decimal Values window.

**3-1-10.** Repeat detailed step 3-1-9 for *khertz\_count* and *mhertz\_count*.

You should be left with what is shown in **Figure 6**.

The screenshot shows a window titled "Waveform - DEV:0 MyDevice0 (XC3S200) UNIT:0 MyILA0 (IL)". It contains a table with columns for "Bus/Signal", "X", and "O". The signals listed are /mhertz\_count, /hertz\_count, /khertz\_count, /hertz\_en, /khertz\_en, and /mhertz\_en. The values for the 'en' signals are 0, while the 'count' signals are empty.


Bus/Signal	X	O
/mhertz_count		
/hertz_count		
/khertz_count		
/hertz_en	0	0
/khertz_en	0	0
/mhertz_en	0	0

**Figure 6. Waveform Display**

## Capturing Data

## Step 4

**4-1. To capture data, regardless of trigger condition, click the T! button or select Trigger Setup → Trigger Immediate. Observe that the waveform display now contains data.**

**4-1-1.** Click the  button in the horizontal toolbar or select **Trigger** → **Trigger Immediate** to observe an immediate trigger.

**4-2. To set a trigger condition, expand the Trigger Setup window to full size. In the Match Unit column, note and expand the three trigger ports that were defined. In the Function column, you can select = or <>. The value for each trigger port is set to a default of don't care (X). The Radix column reflects the bus radix binary that you selected for each bus. The counter is disabled for each trigger port.**

**4-2-1.** Maximize the Trigger Setup window.


**4-2-2.** Expand **M0: Trigger Port0**.

**4-2-3.** Change the value of hertz\_en from *X* to *0* and press **Enter**.

**4-2-4.** Collapse **M0: Trigger Port0** and notice the value now set in TriggerPort0 (0XX\_XXXX\_XXXX versus the previous value of XXX\_XXXX\_XXXX).

**4-2-5.** In the Capture section of the Trigger Setup window, set **Position** to 512. This sets the trigger sample to the center of the 1024-sample-deep Waveform window.

**4-2-6.** Restore the Trigger Setup window to its original size.

**4-2-7.** Click the  button in the horizontal toolbar or select **Trigger Setup** → **Run** from the drop-down list to arm the trigger based on the defined trigger condition.

**Note:** Why is a value of '0' selected as a trigger? Try triggering on a '1'. You will notice that there is no unique situation where this occurs. The original designer of this lab knew this to be true (as this is where the bug exists) so he had you save a little time and look at something a little more interesting initially.

## Debugging the Design

## Step 5

**5-1. Determine the problem with the design.**

**5-1-1.** In the Trigger Setup window, set the following trigger point on M0.

- M0: Trigger Port 0 == X11\_1110\_1000

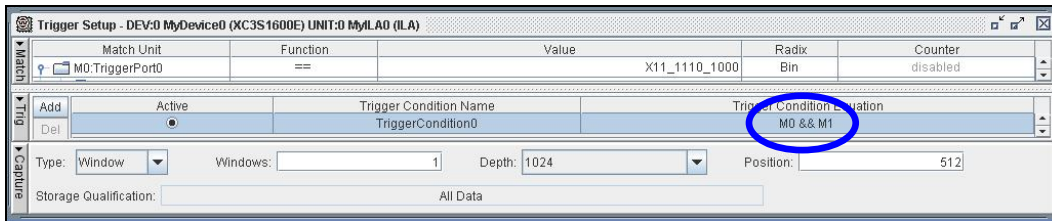
This represents the hertz\_count value at which hertz\_en becomes enabled.

**5-1-2.** Set the following trigger point on M1.

- M1: Trigger Port 1 == 1XX\_XXXX\_XXXX

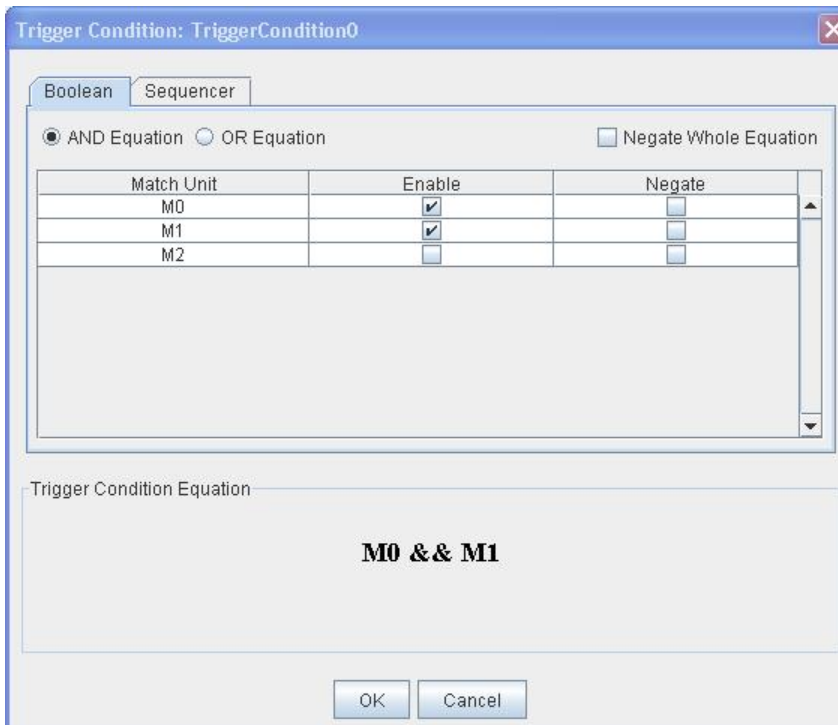
This represents the khertz\_en signal. When the above two trigger points are true, hertz\_en should be enabled.

**5-1-3.** In the Trigger Condition page of the Trigger Setup window, click **Trigger Condition Equation (M0)** (**Figure 7**).



**Figure 7. Click Trigger Condition Equation**

**5-1-4.** In the Trigger Condition window, select **Enable for M1** (leave M0 enabled also) (**Figure 8**).



**Figure 8. Enable M1**

**5-1-5.** Click **OK**.

**5-1-6.** Click the Trigger Run  button in the horizontal toolbar.

Notice that some of the counts may not be incrementing properly. This may be a function of the displaying of the data, not the data itself.

**5-1-7.** Try reversing the bus order (right-click the signal name and select **Reverse bus order**).

**5-1-8.** Observe the Waveform window data.

### Question 3

Do you see any problems or discrepancies with the enable signals?

Hint: Compare hertz\_en with khertz\_en and mhertz\_en. Note that this is the value when hertz\_en should be enabled.

---



---

**5-2.** Go back to the Project Navigator and analyze the code for *exampleInserter.vhd*.\.v.

**VHDL users:** Look at the process on lines 198 through 218 to identify the problem with `hertz_en` being enabled too frequently.

**Verilog users:** Look at the always blocks on lines 164 through 184.

**Correct the design source and recompile the design. Confirm that the problem has been fixed by the correct operation of the board and by using the ILA core within the design.**

**5-2-1.** Go back to Project Navigator, double-click *exampleInserter.vhd*/.v in the Sources window.

- **VHDL users:** Scroll down to line 198 and examine the process
- **Verilog users:** Scroll down to line 164 and examine the process

#### Question 4

What problem is caused by `hertz_en` always being enabled?

---

- **VHDL users:** Scroll to line 215 and change `hertz_en <= '1'`; to `hertz_en <= '0'`;
- **Verilog users:** Scroll to line 184 and change `hertz_en <= 1'b1`; to `hertz_en <= 1'b0`;

**5-2-2.** Select **File** → **Save**.

**5-2-3.** In the Processes window, double-click **Generate Programming File**.

**5-3. Confirm that the problem has been fixed by the correct operation of the board and by using the ILA core within the design.**

**5-3-1.** When the Generate Programming File process is complete, go back to the ChipScope Pro Analyzer tool window.

**5-3-2.** Right-click the **XC3S1600E** device in the New Project pane and select **Configure**.

**5-3-3.** Click **OK**.

**5-3-4.** Verify that the counter and LEDs now work correctly.

## Observing the Amount of Consumed Resources

## Step 6

---

**6-1. Within the ISE software, compare the amount of resources consumed by the ILA core.**

**6-1-1.** Click **View Design Summary**.

## Question 5

In the table provided for Question 2, using the Design Summary within the Project Navigator, enter the resource utilization in the After Insertion column for the following.

- Total Number of Slice Registers
- Total Number of 4-input LUTs
- Number of RAMB16s

## Question 6

Compare the utilization to what you recorded in Question 2 before the ILA core was added. Had it been necessary, what would have been some of the ways you could have saved resources?

---



---

## Question 7

Comparing the utilization to what you recorded in Questions 2 and 3, before the ILA core was added, is this about what you expected?

---



---

## Question 8

If the original design had used 90 percent of the available resources and you created the exact same ILA core, would the size of the core have been different?

---



---

## Additional Exercises (Optional)

## Step 7

**7-1. Open the CDC file and change the settings to reduce the amount of resources consumed by the ILA core. Reimplement and analyze in the ChipScope Pro Analyzer tool.**

**7-2. The ChipScope Pro software is more than just blinking LEDs and a waveform. There are more ways to look at data.**

**Double-click the *Listing* entry under the ILA core (in the Project window). Notice the list of sample numbers and the indicators of where the markers are. There is currently no data visible.**

**Click and drag *khertz\_count* into the Listing window. Notice that a column of data appears.**

**Repeat the above with *mhertz\_count* (Figure 9).**

<->	Sample	/mhertz...	/khertz...
<input checked="" type="radio"/>	0	02	000
<input checked="" type="checkbox"/>	1	03	000
	2	04	000
	3	05	000
	4	06	000
	5	07	000
	6	08	000
	7	09	000
	8	0A	000
	9	0B	000
	10	0C	000
	11	0D	000
	12	0E	000
	13	0F	000
	14	10	000
	15	11	000
	16	12	000
	17	13	000
	18	14	000
	19	15	000
	20	16	000
	21	17	000
	22	18	000
	23	19	000
	24	1A	000
	25	1B	000
	26	1C	000
	27	1D	000
	28	1E	000
	29	1F	000
	30	20	000
	31	21	000
	32	22	000
	33	23	000
	34	24	000
	35	25	000
	36	26	000

X: 1      0: 0      Δ(X-0): 1

Figure 9. mhertz and khertz Columns

### Question 9

When do you think this view might be helpful?

---



---

Double-click the *Bus-Plot* view (from the Project window).

Notice that the buses are available for your selection. Simply click a bus and select how you want to display the plot (Figure 10).

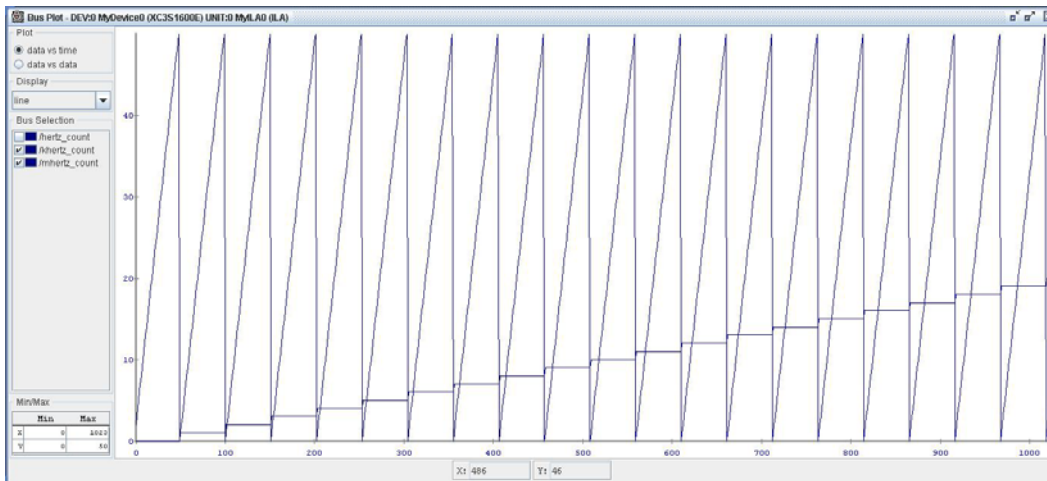


Figure 10. Bus-Plot View

## Question 10

When do you think this view might be helpful?

---



---

## Conclusion

In this lab, you completed the major stages of setting up for debug with ChipScope using the Inserter flow.

## Answers

- Record the resource utilization for this design from the Design Summary within the Project Navigator in the Prior to Adding ChipScope Pro Cores column. Leave the remaining two columns blank for now.

VHDL:

Number of Slice Flip-Flops:	37
Number of 4-input LUTs:	41
Number of RAMB16s:	N/A (none were used)

Verilog:

Number of Slice Flip-Flops:	37
Number of 4-input LUTs:	41
Number of RAMB16s:	NA (none were used)

- In the table provided for Question 1, enter the resource utilization in the ChipScope Pro Core Resources column for the following: LUT count, flip-flop count, and block RAM count.

LUT count:	411
FF count:	329
BRAM count:	2

- Do you see any problems or discrepancies with the enable signals?

Yes, the hertz\_en signal transitions from 1 to 0 only briefly, rather than the other way around.

- What problem is caused by hertz\_en always being enabled?

The LEDs are enabled at 50 MHz, which is too fast for the eye to see.

- In the table provided for Question 1, using the Design Summary within the Project Navigator, enter the resource utilization in the After Insertion column for the following.

VHDL:

Total Number of Slice Registers:	390
Total Number of 4-input LUTs:	302
Number of RAMB16s:	2

Verilog:

Total Number of Slice Registers:	366
Total Number of 4-input LUTs:	420
Number of RAMB16s:	2

- Compare the utilization to what you recorded in Question 2 before the ILA core was added. Had it been necessary, what would have been some of the ways you could have saved resources?

When creating the ILA core, you could have reduced the depth of data samples and reduced the maximum number of sequencer levels.

- Comparing the utilization to what you recorded in Questions 1 and 2, before the ILA core was added, is this about what you expected?

Yes. Although it appears that some of the LUTs were optimized out, because adding the LUTs from Questions 1 and 2 is more than the total that are actually used.

- If the original design had used 90 percent of the available resources and you created the exact same ILA core, would the size of the core have been different?

No, the core size would have remained the same.

9. When do you think this view might be helpful?

This view is good for bus listings, offering a compressed textual view, and for any time that you are looking for sequences over a wide range. (The waveform view often requires you to be zoomed in so far that in order to see the textual name that the “big picture” is lost.)

10. Where might the bus-plot view be useful?

In this lab, it is a quick way to check if the lower order (faster) counter is resetting to zero. In other applications, it is a convenient way to see if data is being generated or processed in the way that is expected. For example, if you were building a DDS (direct digital synthesizer – waveform synthesis), you could use this view to determine very quickly if your design was working.