EELE 367 – Logic Design
Lab #2 – Ripple Counter, 7-Segment Decoder, and Logic Analysis

Overview

In this lab you will design a 32-bit ripple counter using a D-flip-flop component. You will also create four 7-segment decoders to drive the displays on the I/O shield using the counter as the input to the decoders. You will also be introduced to taking measurements on the counter using the logic channels of an oscilloscope.

Outcomes

After completing this lab you should be able to:

- Use the STD_LOGIC data types in your VHDL model.
- Design a 32-bit ripple counter out of D-flip-flop components.
- Design a 7-segment decoder using the conditional modeling constructs of VHDL (if/then or case).
- Determine the frequency that each bit within a counter is running.
- Take a digital measurement using the logic channels of an oscilloscope.

Deliverables

The deliverable(s) for this lab are as follows:

- Pre-lab: A spreadsheet calculating the rate each bit changes in a 32-bit counter clocked at 50MHz. This is to be uploaded to the lab DropBox prior to starting the lab (10%).
- Demonstration of the successful operation of your design (40%).
- Uploading your top.vhd files for this lab to the course DropBox (10%).
- Demonstration of a successful measurement using the logic channels of an oscilloscope (30%).
- Uploading a screenshot of your logic waveform to the course DropBox (10%).

Pre-Lab

Calculate how fast each bit within a 32-bit counter is running if it is clocked with a 50 MHz clock signal. This calculation can be done quickly using a spreadsheet. Consider the fact that the LSB of the counter is running at ½ of the incoming clock. This means that the *frequency* of the LSB is 25MHz, however, since each logic value of the LSB is used as a value of the count (i.e., when it is high the LSB=1, when it is LOW the LSB=0), then the actual *data rate,* or the rate that the counter counts, is 50Mb/s (b/s = bits per second). Create a spreadsheet that tabulates the frequency, period, data rate, and unit interval (i.e., the period of the data rate) for each of the 32-bits of a counter when clocked at 50MHz. If we wanted to use four consecutive bits of the counter to drive a 7-segment display and have the values change at a rate that could be visible by the human eye, what four bits would we use? Highlight these bits in the spreadsheet. Upload your spreadsheet to the Lab #2 DropBox prior to starting the lab.

Lab Work & Demonstration

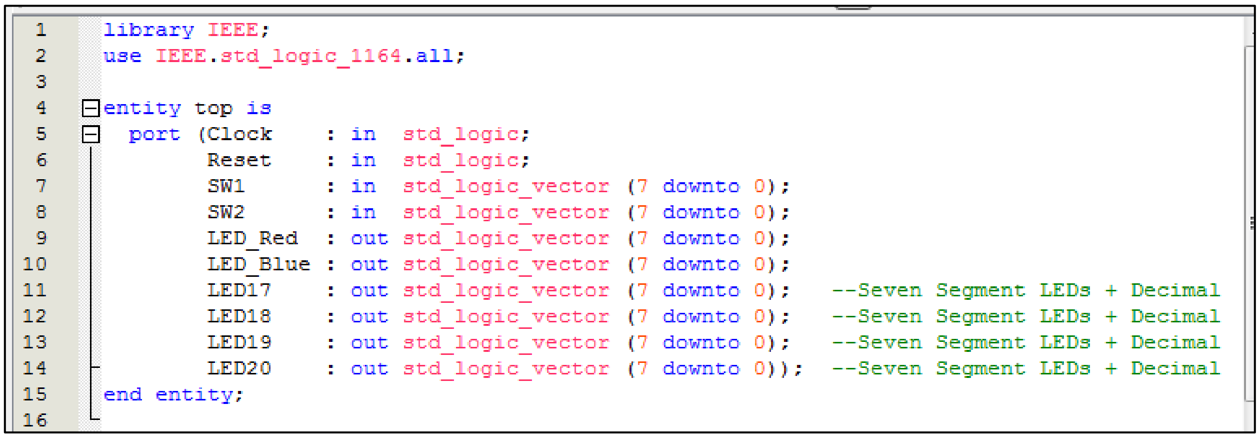
Part 1 – System Design & Demonstration (40%)

A. Create a New Quartus Project

   You are going to create a new Quartus II project for this lab. Instead of creating this from scratch like in Lab #1, you can copy your lab from last week and preserve many time-consuming aspects of creating a project (e.g., selecting the device, assigning pins, etc.). In order to copy a project, you must first open an existing project and then perform *Project – Copy Project*. You can then specify the new location and name of the new project. Before you do this, make sure to create a new folder called "Lab02_Riple_Counter_n_7segment_Decoder" that you will put your new project into. Open your Lab #1 project and copy it over to a new project using *Project – Copy Project*. Open the new project.

B. Use the IEEE.STD_LOGIC_1164 Package

Now that you have a new project for Lab #2, you can begin designing your system. The first thing you need to do is change your data types in your top.vhd to use STD_LOGIC. These data types are most commonly used in industry and allow you to model additional behavior of modern digital hardware. You will need to include the IEEE.std_logic_1164 library/package information at the top of your top.vhd file and then change each of the data types of the ports in your entity. Your entity should look like this after you make the changes:

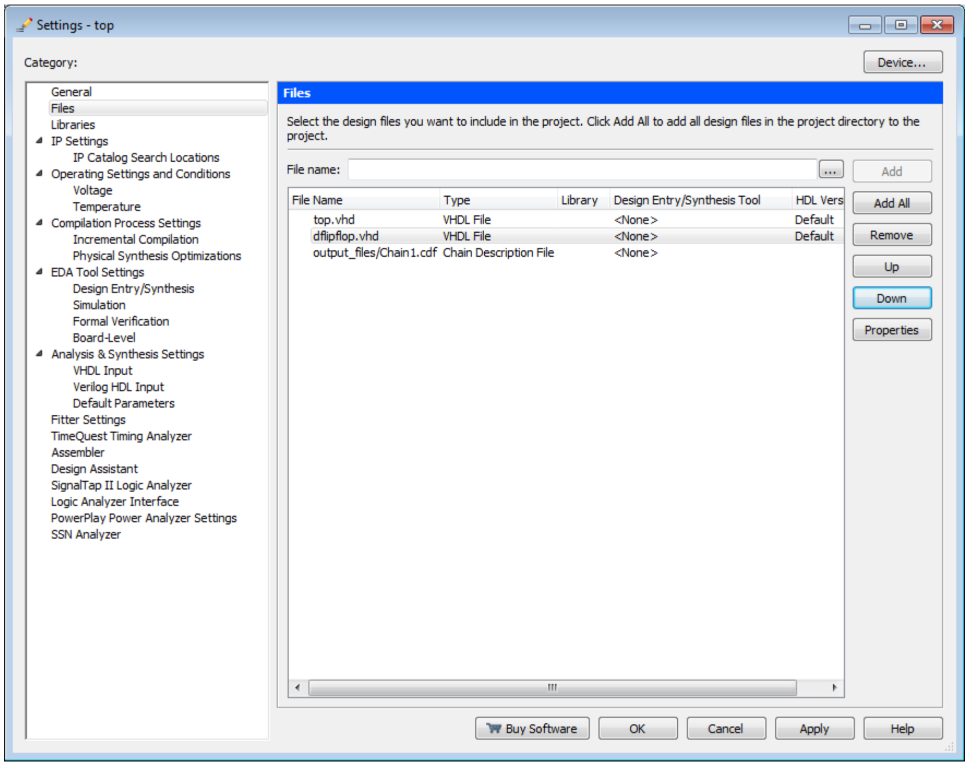


```
library IEEE;
use IEEE.std_logic_1164.all;

entity top is
  port (Clock     : in  std_logic;
        Reset     : in  std_logic;
        SW1       : in  std_logic_vector (7 downto 0);
        SW2       : in  std_logic_vector (7 downto 0);
        LED_Red   : out std_logic_vector (7 downto 0);
        LED_Blue  : out std_logic_vector (7 downto 0);
        LED17     : out std_logic_vector (7 downto 0);   --Seven Segment LEDs + Decimal
        LED18     : out std_logic_vector (7 downto 0);   --Seven Segment LEDs + Decimal
        LED19     : out std_logic_vector (7 downto 0);   --Seven Segment LEDs + Decimal
        LED20     : out std_logic_vector (7 downto 0));  --Seven Segment LEDs + Decimal
end entity;
```

C. Add a D-flip-flop Model to your Project

Now you are going to add a D-flip-flop model to your project that can be instantiated in your top.vhd in order to create the 32-bit ripple counter. Download the Dflipflop.vhd file from the course site and put it in your project directory. In the Project Navigator window of Quartus (upper left), click on the “Files” tab. Highlight “Files”, right-click, and select “Add/Remove Files in Project”. In the Settings window, browse to the Dflipflop.vhd file (the browse button is […]), and then add to your project. Once it appears in the list, highlight it and move it down so that it is below top.vhd. This tells Quartus that it is not the top level system to synthesis but rather a sub-system called by top.vhd.


Settings - top

Category:

- General
- Files
- Libraries
- IP Settings
  - IP Catalog Search Locations
- Operating Settings and Conditions
  - Voltage
  - Temperature
- Compilation Process Settings
  - Incremental Compilation
  - Physical Synthesis Optimizations
- EDA Tool Settings
  - Design Entry/Synthesis
  - Simulation
  - Formal Verification
  - Board-Level
- Analysis & Synthesis Settings
  - VHDL Input
  - Verilog HDL Input
  - Default Parameters
- Fitter Settings
- TimeQuest Timing Analyzer
- Assembler
- Design Assistant
- SignalTap II Logic Analyzer
- Logic Analyzer Interface
- PowerPlay Power Analyzer Settings
- SSN Analyzer

Files

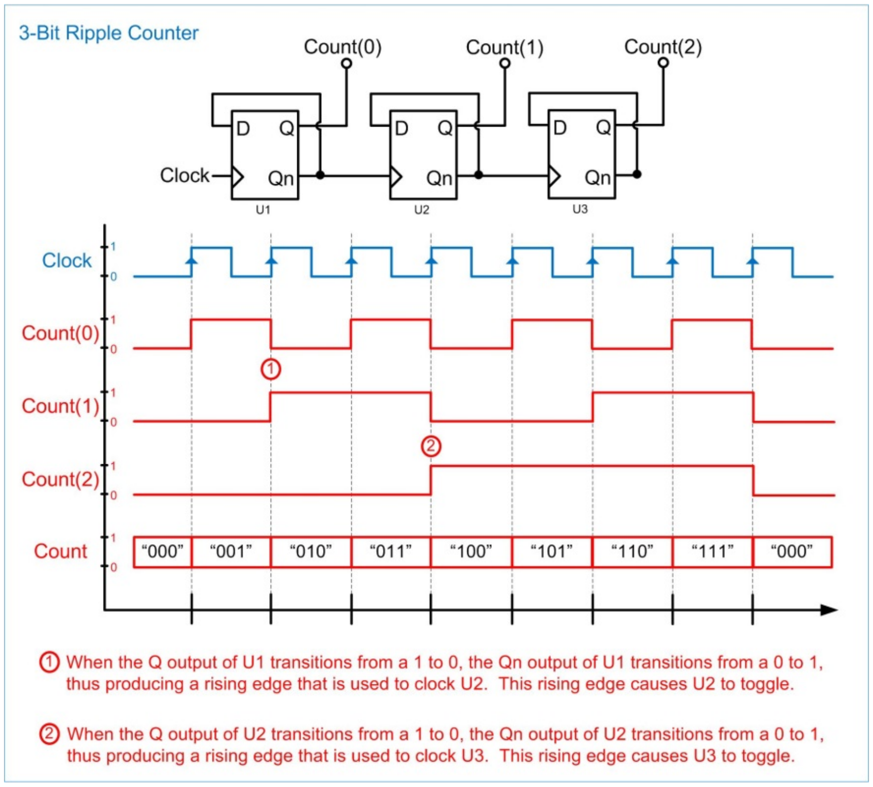
Select the design files you want to include in the project. Click Add All to add all design files in the project directory to the project.

File name:

File Name	Type	Library	Design Entry/Synthesis Tool	HDL Vers
top.vhd	VHDL File		<None>	Default
dflipflop.vhd	VHDL File		<None>	Default
output_files/Chain1.cdf	Chain Description File		<None>	

D. Design the 32-Bit Ripple Counter

Now you can design the 32-bit ripple counter in your top.vhd by instantiating 32 D-flip-flops. You will need to create internal nodes for the Q and Qn outputs of the D-flips-flops. Call the internal nodes “Count” and “CountN” and use the type std_logic_vector. Recall that the topology of a ripple counter is as follows:

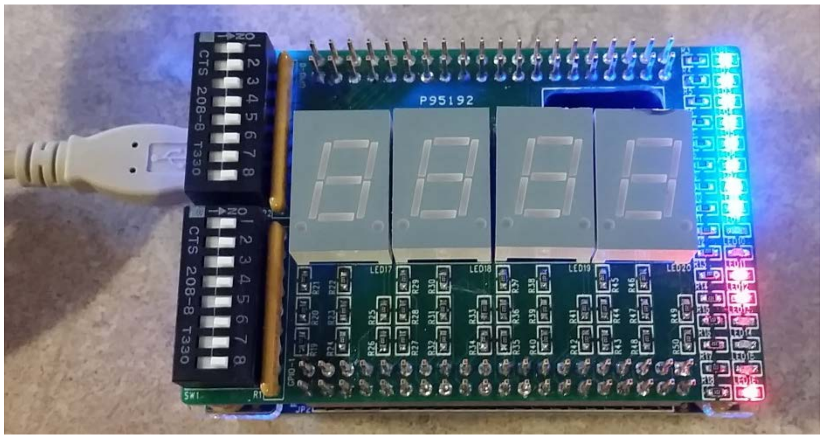


① When the Q output of U1 transitions from a 1 to 0, the Qn output of U1 transitions from a 0 to 1, thus producing a rising edge that is used to clock U2. This rising edge causes U2 to toggle.

② When the Q output of U2 transitions from a 1 to 0, the Qn output of U2 transitions from a 0 to 1, thus producing a rising edge that is used to clock U3. This rising edge causes U3 to toggle.

E. Connect the Upper 16-bits of the Counter to the Red and Blue LEDs

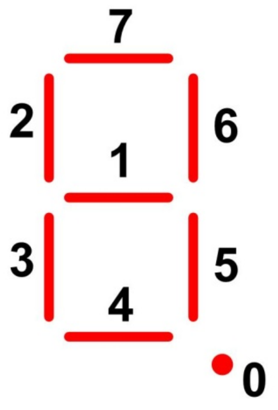
Route the upper 16-bits of the counter to the LEDs on the I/O shield (LED_Red(7:0) <= Count(31:24) and LED_Blue(7:0) <= Count(23:16)). This can be done with simple signal assignments. At this point it is a good idea to compile and test your design. When you run your design, you'll see the upper 16-bits of your counter on the blue and red LEDs of the I/O shield. Notice that the lower bits of the counter (blue) are moving too fast to see with the human eye.



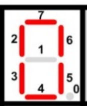
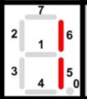
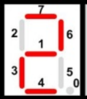
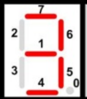
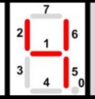
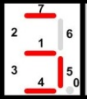
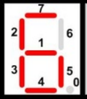
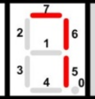
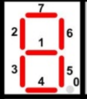
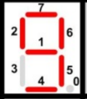
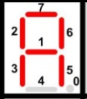
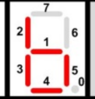
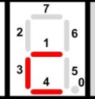
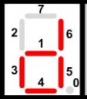
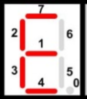
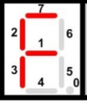
F. Design the 4x 7-Segment Decoders

Now create 4 separate 7-segment decoders to drive the displays on the I/O shield (LED17, LED18, LED19, LED20). Create your 7-segment decoders using processes. You can use whichever modeling approach you'd like (if/then or case). The inputs to your decoders will be groups of 4-bits coming from your 32-bit counter. Use the following mapping: LED17 <= "decode of Count(31:28)", LED18 <= "decode of Count(27:24)", LED19 <= "decode of Count(23:20)", and LED20 <= "decode of Count(19:16)". The following table is helpful when designing your decoder. Consider getting one decoder working and then using copy/paste to implement the next three.

Character Display

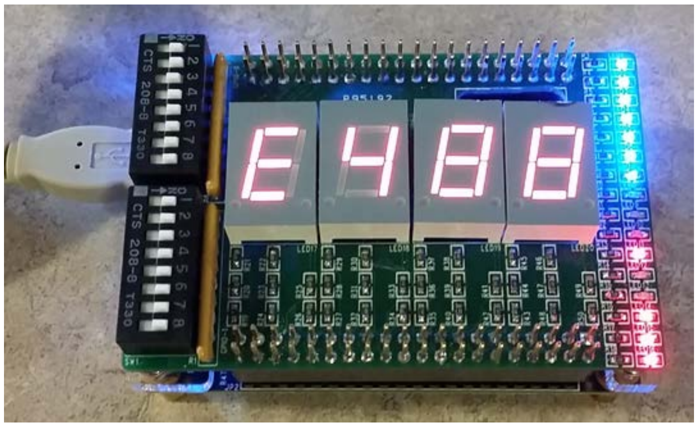


Common Cathode
0=OFF, 1=ON

Input		F_7	F_6	F_5	F_4	F_3	F_2	F_1	F_0
0 0 0 0		1	1	1	1	1	1	0	0
0 0 0 1									
0 0 1 0									
0 0 1 1									
0 1 0 0									
0 1 0 1									
0 1 1 0									
0 1 1 1									
1 0 0 0									
1 0 0 1									
1 0 1 0									
1 0 1 1									
1 1 0 0									
1 1 0 1									
1 1 1 0									
1 1 1 1									

G. Test your Design

Compile and download your design. You should now see the 7-segment displays counting. Notice that the least significant position (LED20) is moving too fast to be observed with the human eye.



H. Demonstrate your Design

DEMO

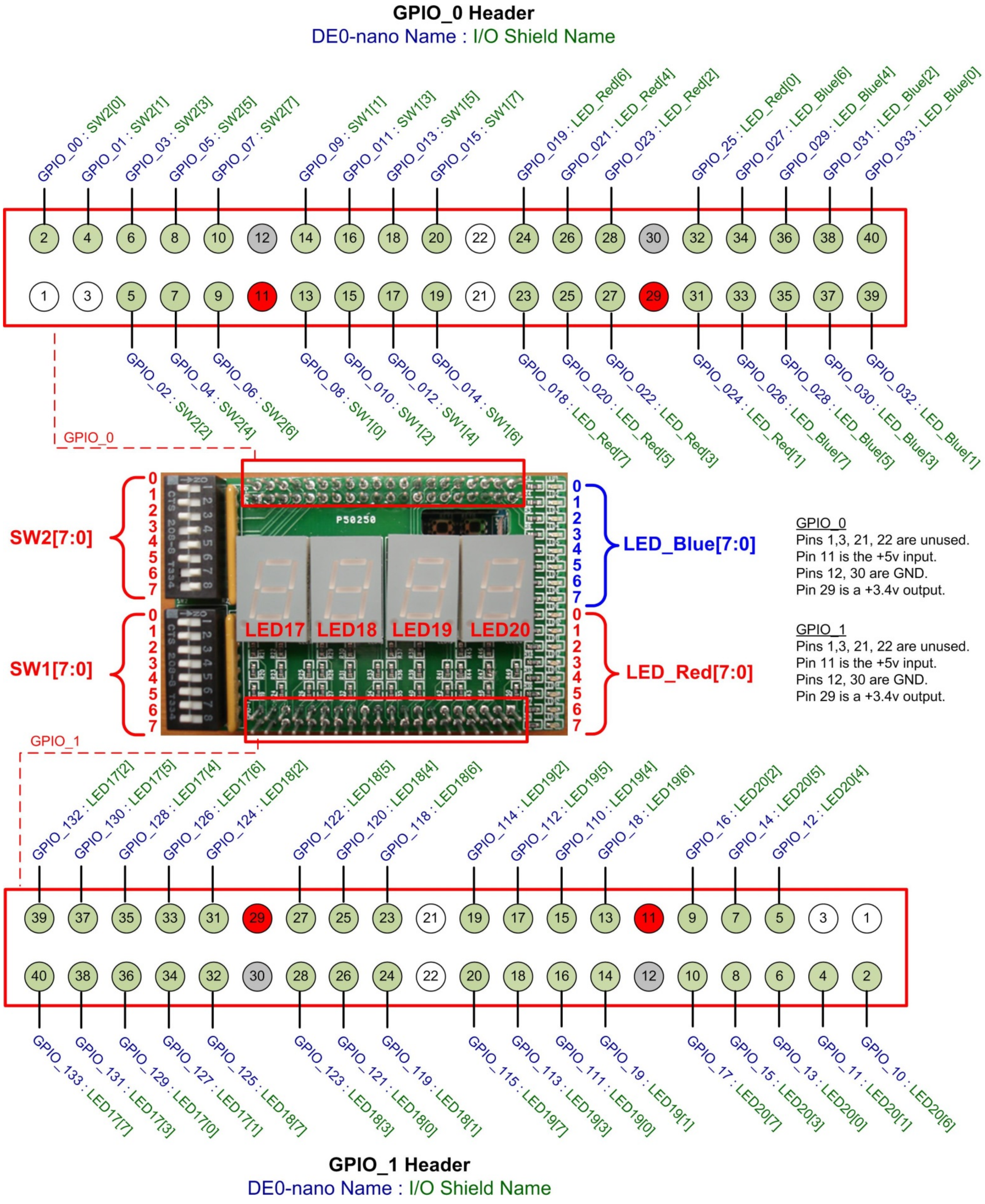
Demonstrate the proper operation of your design (40% of your grade). Upload your top.vhd file to the Lab #2 DropBox (10% of your grade).

Part 2 – Taking Measurements with the Logic Channels of an Oscilloscope (40%)

In this part you are going to take a measurement on the 8-bits of your counter driving the blue LEDs on the I/O shield. These are the Count(23:16) bits of your counter. These bits can be observed on the GPIO_0 pin header of the I/O shield. You will be taking a logic analyzer measurement on these signals. A logic measurement is different from an oscilloscope measurement in that instead of seeing the exact analog voltage, you only observe either a HIGH or LOW. By only recording the logic state of the signal being observed, this allows the complexity of the measurement circuitry to be reduced and more channels can be added to the instrument. This allows large sets of digital signals to be observed simultaneously. The logic analyzer will allow you to view the individual bits in addition to grouping them into a bus to see the overall decimal or hex value of the group. Logic Analyzers can have anywhere from 8 to thousands of channels. Most modern oscilloscopes now have the ability to measure digital channels in addition to their traditional analog channels. These oscilloscopes are referred to as *Mixed Signal Oscilloscopes* (or MSOs). The oscilloscopes in the digital lab (Tektronix MSO 4054) have the ability to measure up to 16 digital channels. The Analog Discovery instrument being used by the online lab section also has the ability measure 16 digital signals. We are going to connect eight logic channels to the appropriate pins of the GPIO_0 pin header on the I/O shield in order to measure the count value and frequency of the 8-bits driving the blue LEDs. We will then compare the measured frequency with our calculated frequency from the pre-lab.

Note: This lab has tutorials on how to setup this measurement on both the lab MSO's and the Analog Discovery. You only need to read the appropriate section.

The following figure shows the signal mapping of the pin headers on the I/O shield. Note that the pins on the I/O shield are straight pass-through connections to the DE0-nano pins.



GPIO_0
Pins 1,3, 21, 22 are unused.
Pin 11 is the +5v input.
Pins 12, 30 are GND.
Pin 29 is a +3.4v output.

GPIO_1
Pins 1,3, 21, 22 are unused.
Pin 11 is the +5v input.
Pins 12, 30 are GND.
Pin 29 is a +3.4v output.

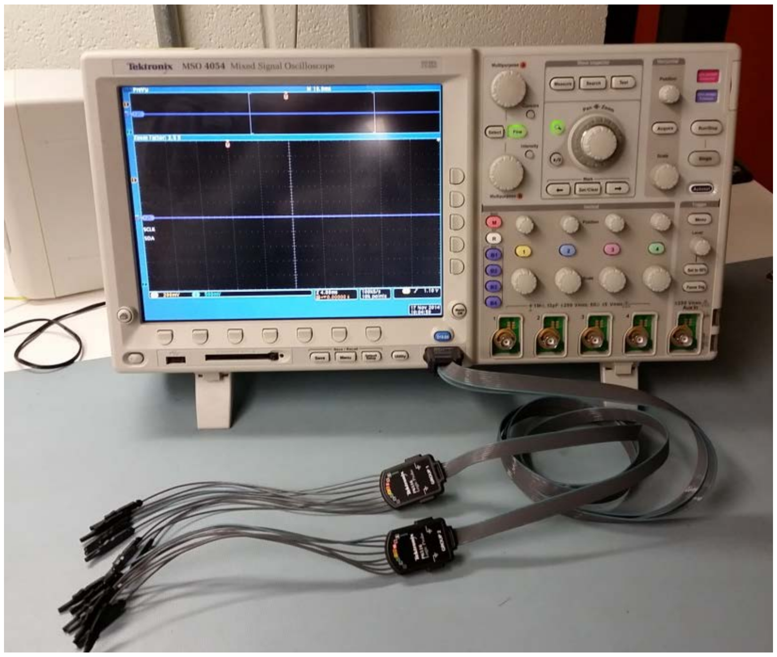
Using the Tektronix MSO 4054

A. Starting up the Tektronix MSO 4054

In the digital lab there is a Tektronix MSO 4054 mixed signal oscilloscope at each station. In addition to having four analog input channels, these instruments support up to 16 digital input signals. In order to use the digital channels, you must plug in a separate logic probe (P6516).

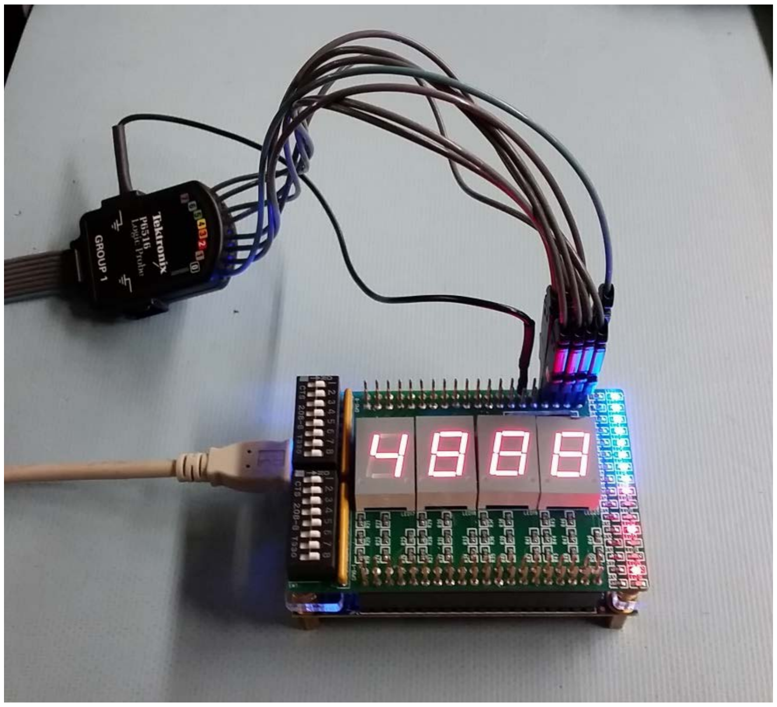
Turn on the Tektronix SO 4054 oscilloscope.

Plug in the Tektronix P6516 Logic Probe into the front panel of the Tektronix MSO 4054 Oscilloscope. The input port for the probe is right below the blue “D15-D0” button. Once plugged in the connection should look like this:



B. Setting up the Probing Connection

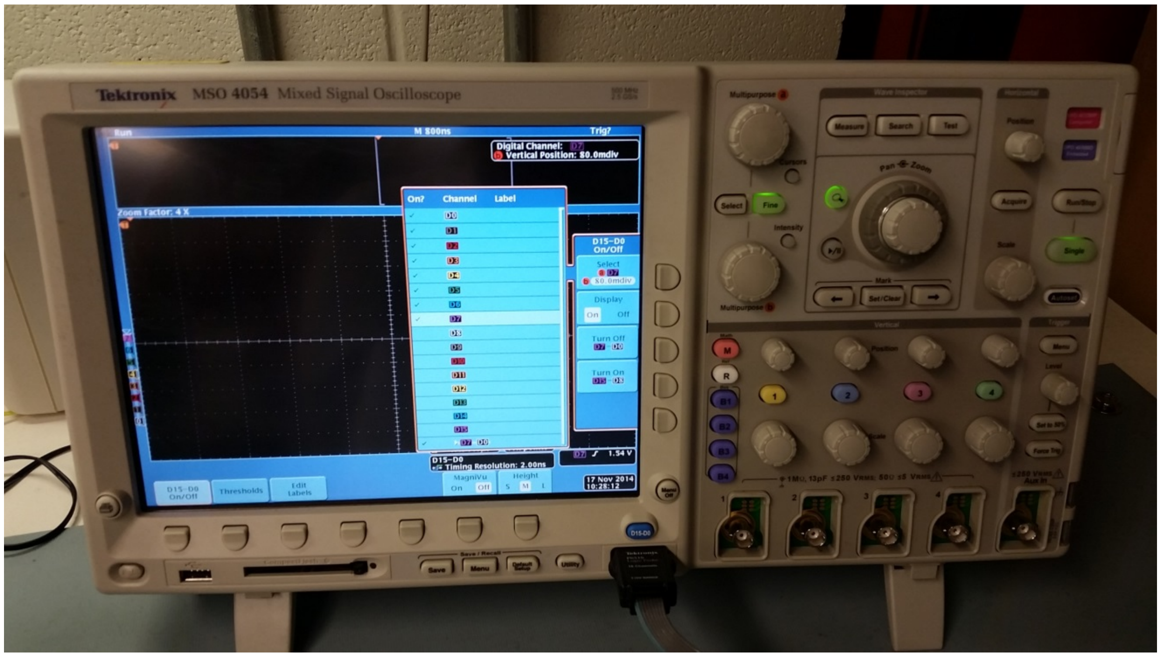
Connect the digital channels [7:0] of the P6516 logic probe to the header pins on the I/O shield corresponding to LED_Blue[7:0]. Notice that the P6516 probe has two groups. Bits [7:0] are on the group 1 section of the probe. See the pinout figure provided above for the header pinout. Also connect the ground on the probe to one of the ground signals on the pin header. Your connection should look like this.



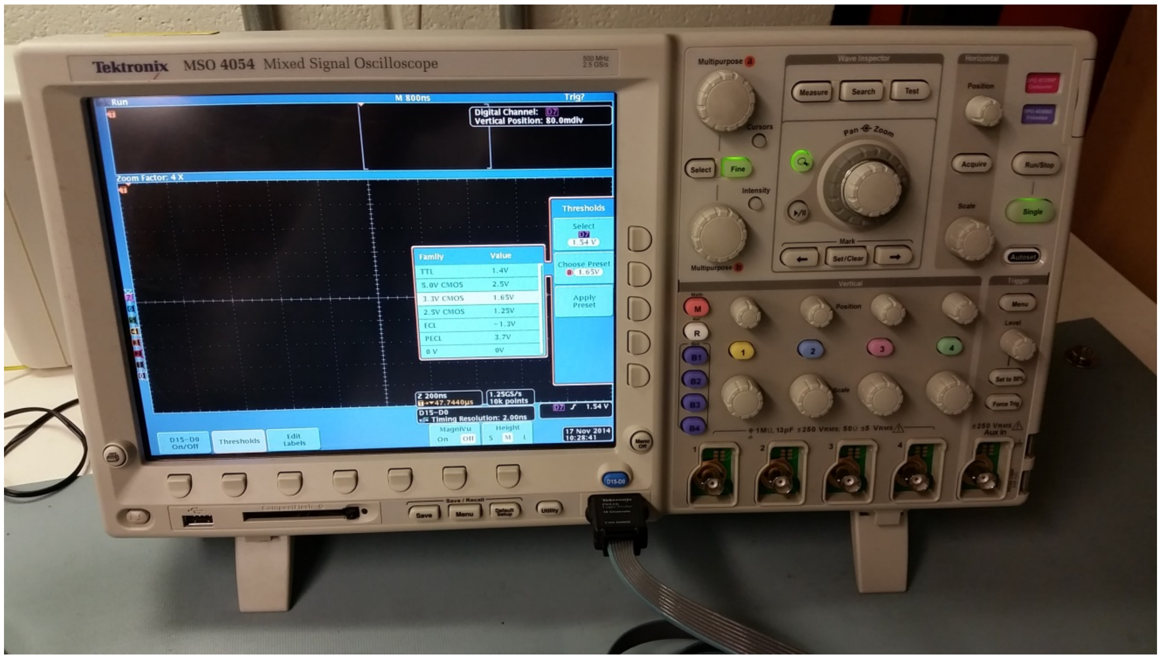
C. Configure the Digital Channels of the MSO

Turn on the digital channels by pressing the blue "D15-D0" button on the oscilloscope. This will turn on the digital channels menu on the bottom of the screen.

Select the "D15-D0 On/Off" button (lower left of the scope). This will allow you to turn on/off the individual channels of the digital inputs. On the right of the screen, use the side buttons to turn on only D7-D0. When you are done, you can click "Menu Off" to go back.



Select the “Thresholds” button (lower left of the scope). This will allow you to set the logic threshold for the digital inputs. On the right side of the screen, select the “Choose Preset”, then use the Multipurpose (a) scroll knob to select “3.3V CMOS, 1.65V”. You want to assign this preset to all 8 channels (i.e., D7-D0), so make sure to scroll down to that section on the menu. Click on the “Apply Preset” button. When you are done, you can click “Menu Off” to go back.



Family	Value
TTL	1.4V
5.0V CMOS	2.5V
3.3V CMOS	1.65V
2.5V CMOS	1.25V
ECL	-1.3V
PECL	3.7V
0 V	0V

Digital Channel: D7
Vertical Position: 80.0mdiv

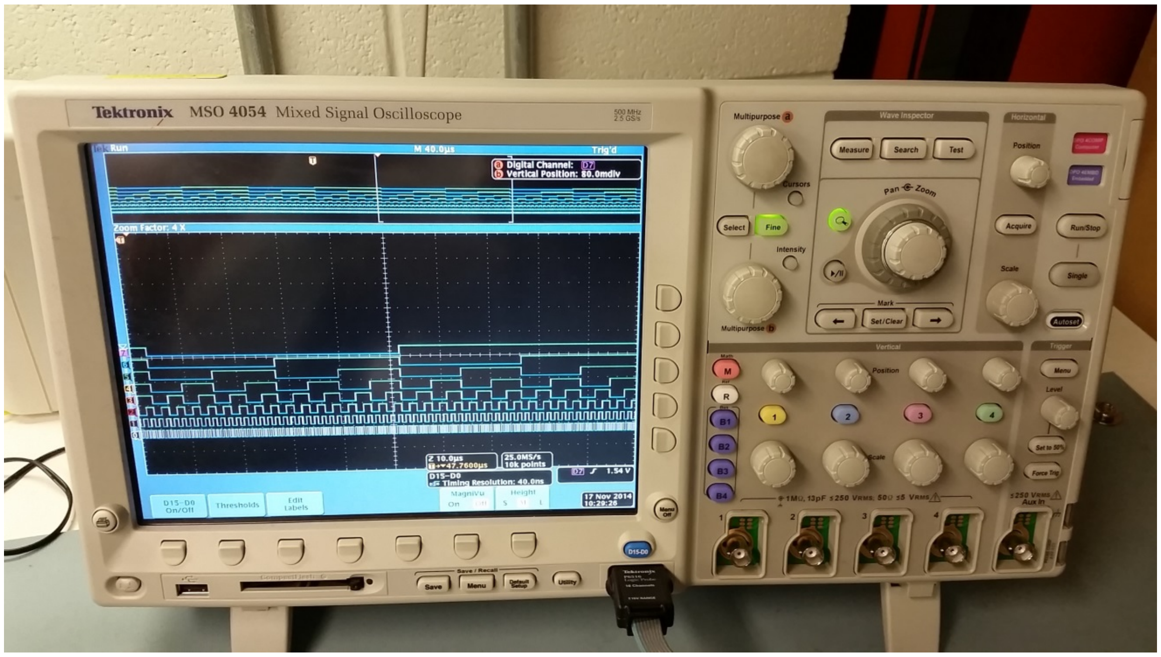
D. Setup the Trigger of the MSO

Set the trigger to D7. We can’t trigger the MSO on a group value, so we will trigger when D7 goes to a 0. The only time this will occur is when the 8-bit counter rolls over from x”FF” to x”00”. This will give the same trigger as if we triggered on Count=x”00”. To do this, we want to setup the MSO to trigger when D7 has a falling edge. Click on the Trigger-Menu button (right side of scope). Click on source, and then use the Multipurpose (a) scroll knob to set:

Type = Edge
Source = D7
Coupling = DC
Slope = falling
Level 1.65v
Auto

E. Run the Measurement

If not already running, press the “Run” button the oscilloscope. You should see the 8 digital channels on the bottom of the oscilloscope. You may need to zoom in/out to see them.



F. Setup a Bus View of your Counter

Turn on a new Bus by pressing the “B1” button.

Use the bus menus (bottom of the scope screen, just like the prior settings) to set the following:

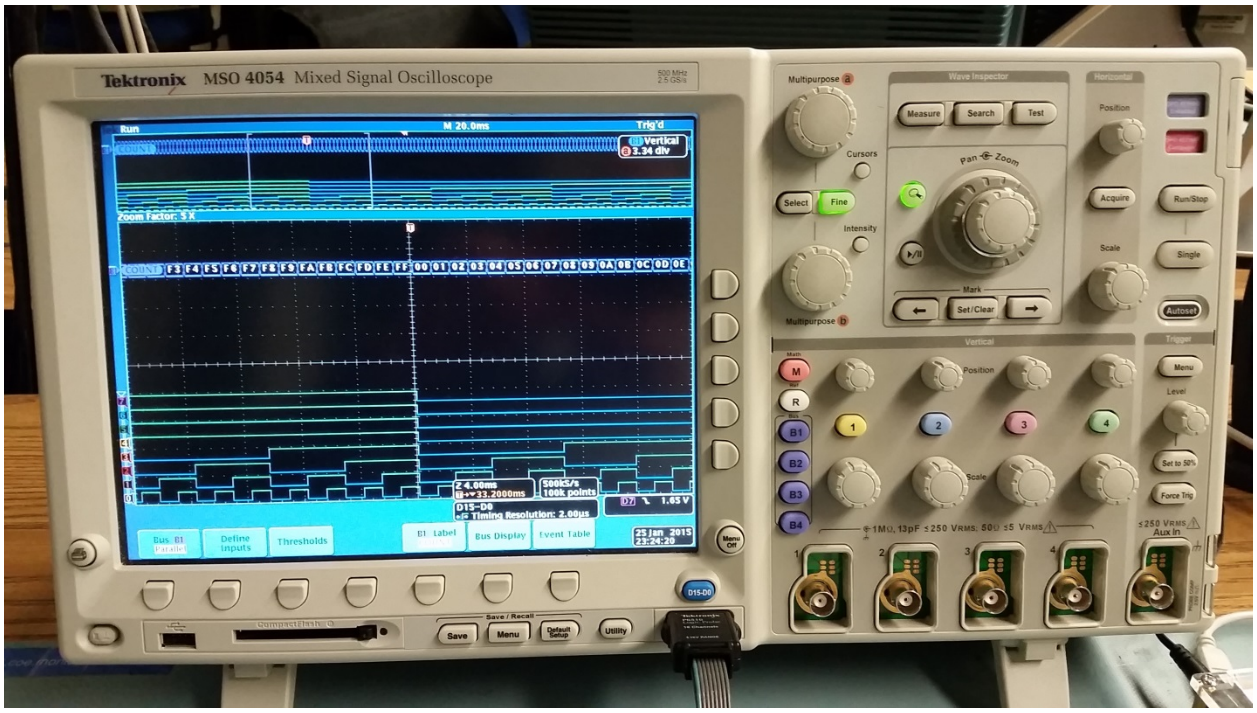
Type of bus: Parallel
Define inputs: Clocked Data = No, 8-bits
Bit 7 = D7
B1 Label = “Count”, you can select this from a list of common names or type in manually.
Bus Display = Bus, Hex

G. View the Counter as a Bus on the MSO

Scroll horizontally until your trigger is in the center of the screen. The trigger is denoted as a T.

Zoom to have a horizontal scale of 4ms/div.

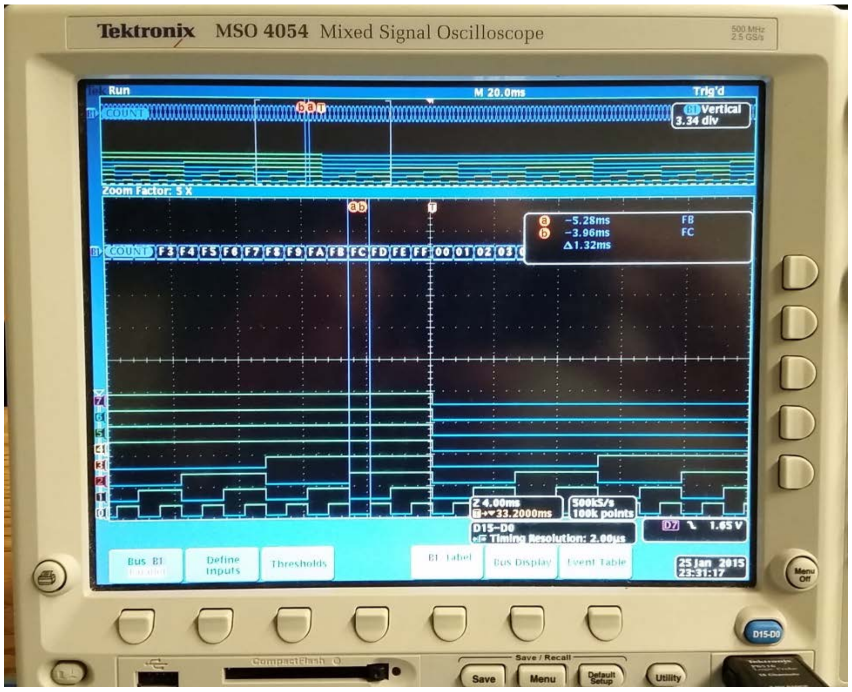
You should see the 8-bit counter roll from 0x”FF” to 0x”00”.



H. Measure the Period of the Signals and the Counter rate

Press the “Cursor” button on the oscilloscope to turn on the cursors (if they aren’t already on).

Use the multipurpose scroll knobs to move the cursors horizontally. The (a) wheel will control the position of one cursor while the (b) wheel will control the other. Position the cursors at the transitions between adjacent counts and record the period of the counter. Verify that the counter is running at the speed that you predicted in your pre-lab.



I. Demonstrate your Measurement

DEMO

Demonstrate your successful measurement verifying that the counter values are correct and the count frequency is as predicted (30%). Take a screenshot of your waveform and upload to the course website (10%).

To take a screenshot on the MSO, press the “Menu” button at the bottom, then press “Save Screen Image”. You can then choose to save the image to a thumb drive.

It is also acceptable to take a clear photo of the MSO screen using your phone.

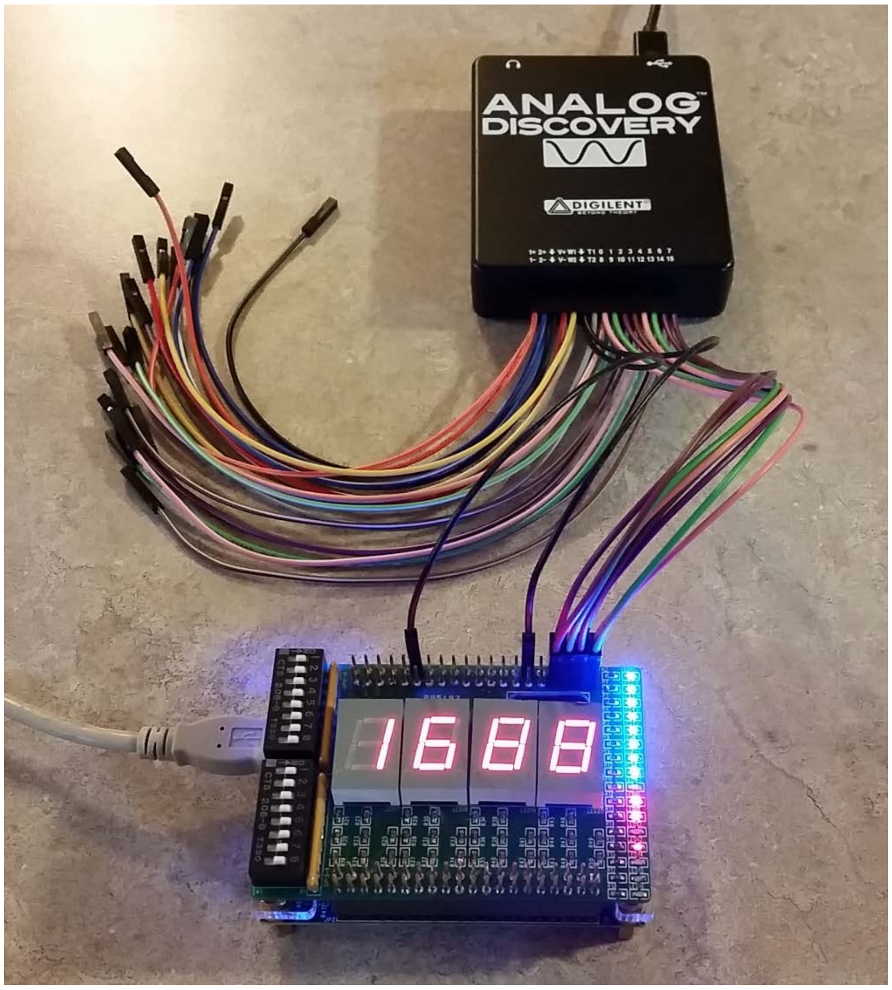
Using the Analog Discovery

A. Installing the Waveform Software

The Analog Discovery instrument uses an application called *Waveforms* that can be downloaded from the Digilent website (http://www.digilentinc.com/). Type in "Waveforms" in the search box and the first item returned will lead you to the download page. Once you download the install executable, launch it and install on your computer.

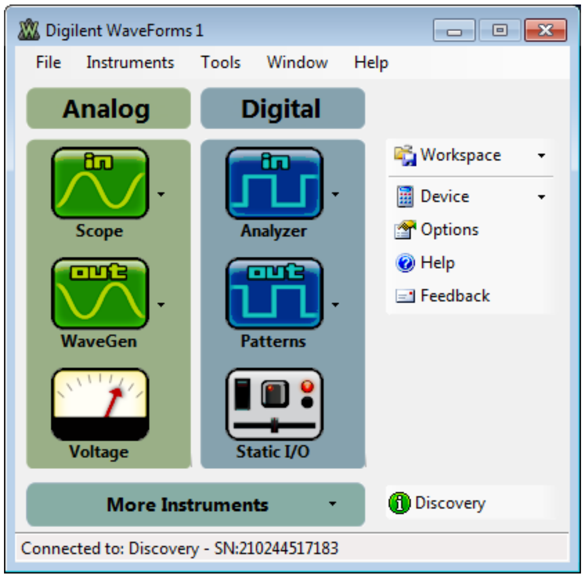
B. Setting up the Probing Connection

Connect the digital channels [7:0] on the Analog Discovery to the header pins on the I/O shield corresponding to LED_Blue[7:0]. See the above figure for the header pinout. Also connect the two grounds on the Analog Discovery to the two ground signals on the pin header. Connect the USB cable between the Analog Discovery and your computer. Your connection should look like this.



C. Launch the Waveforms Program

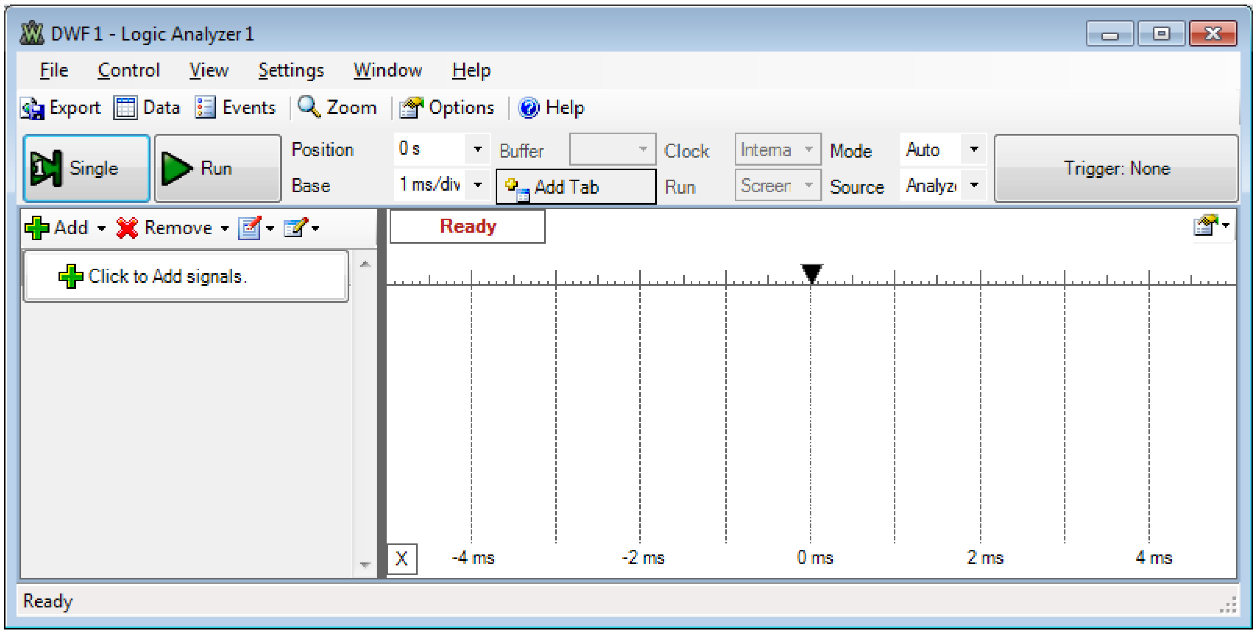
Start the Waveforms application using State – All Programs – Digilent – Waveforms – Waveforms



Connected to: Discovery - SN:210244517183

D. Setup the Logic Analyzer Measurement

Click on the “Analyzer” tool to bring up the logic analyzer instrument.



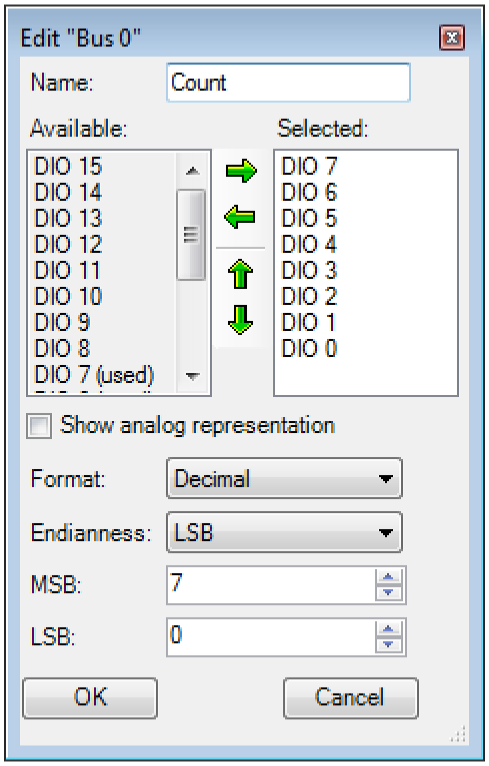
Position: 0 s
Base: 1 ms/div
Buffer
Clock: Interna
Run: Screen
Mode: Auto
Source: Analyz
Ready

Now click on the button labeled “Click to Add Signals” and then choose “Bus”.

Highlight DIO7:0 (this stands for digital I/O) in the “Available” column and add them to the “Selected” column.

Label the new bus “Clock”.

Click “OK”.



Edit "Bus 0"

Name: Count

Available:

DIO 15
DIO 14
DIO 13
DIO 12
DIO 11
DIO 10
DIO 9
DIO 8
DIO 7 (used)

Selected:

DIO 7
DIO 6
DIO 5
DIO 4
DIO 3
DIO 2
DIO 1
DIO 0

Show analog representation

Format: Decimal

Endianness: LSB

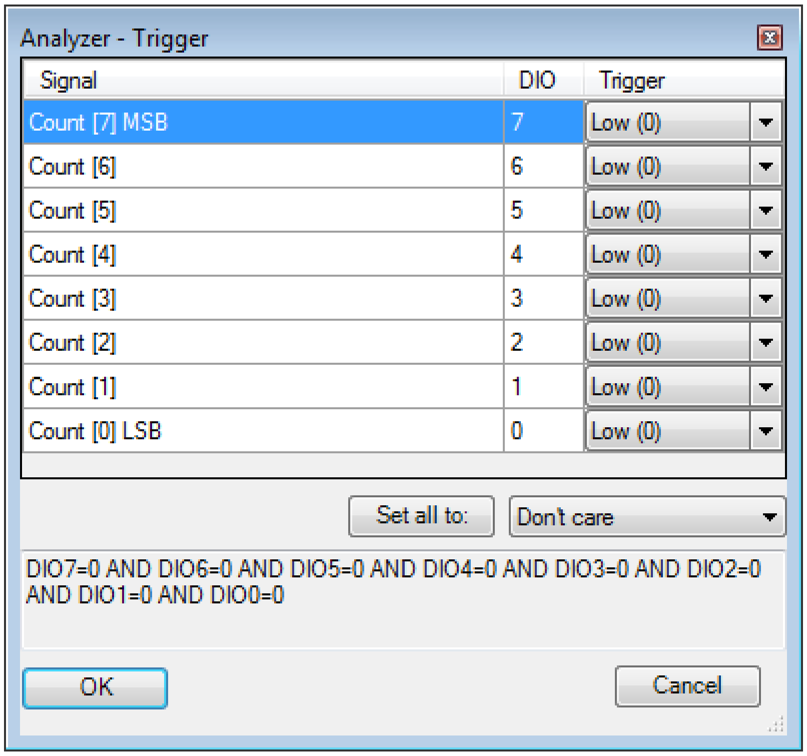
MSB: 7

LSB: 0

Now setup the trigger to look for a counter value of x”00”. This will center the value x”00” on the center of the screen when running. Click on the “Trigger” button.

Set the value for each bit to Low (0).

Click “OK”.



Analyzer - Trigger

Signal	DIO	Trigger
Count [7] MSB	7	Low (0)
Count [6]	6	Low (0)
Count [5]	5	Low (0)
Count [4]	4	Low (0)
Count [3]	3	Low (0)
Count [2]	2	Low (0)
Count [1]	1	Low (0)
Count [0] LSB	0	Low (0)

DIO7=0 AND DIO6=0 AND DIO5=0 AND DIO4=0 AND DIO3=0 AND DIO2=0 AND DIO1=0 AND DIO0=0

E. Take a Logic Analyzer Measurement.

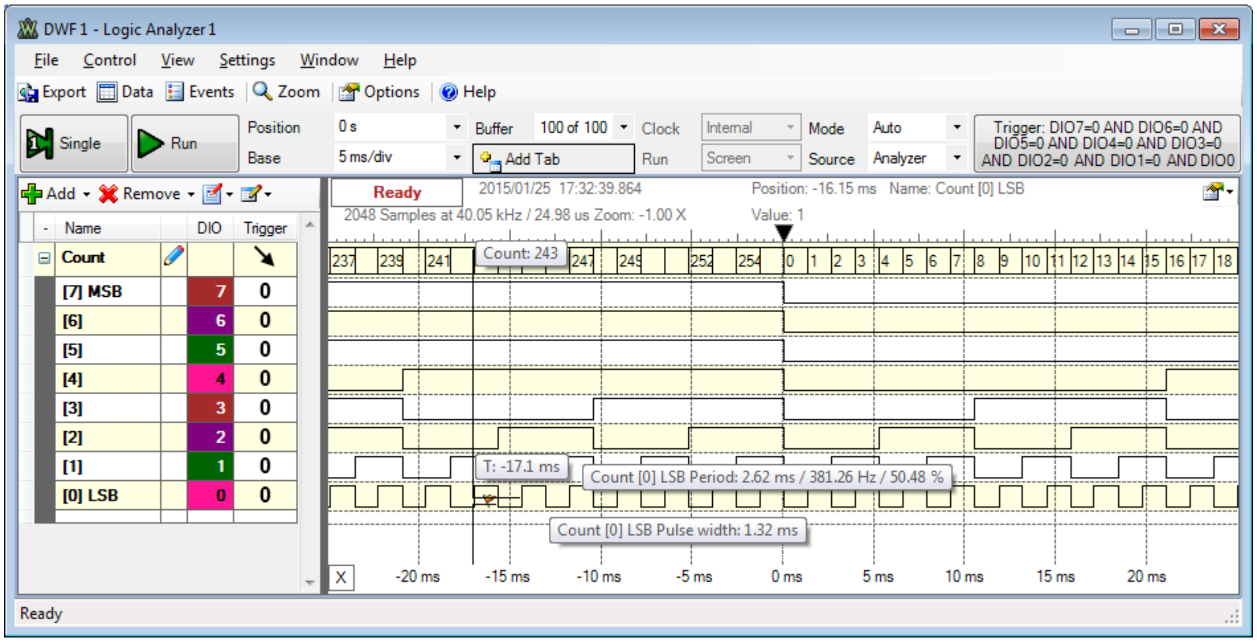
At this point you can press the "Run" button and the measurement will start. The measurement will run continuously and you will see the logic waveforms on the screen.

You can zoom in and out using the "Base" setting. For this counter speed, you may want to set this to 5ms.

Right Click in the waveform screen and select "Hot Track". This will allow you to click on or mouse-over a waveform and get measurement information such as period, frequency, high time, and duty cycle.

Notice that you can see the individual bits of the 8-bit counter and also the decimal value of the bus.

Your waveforms should look like this:



Position: 0 s
Base: 5 ms/div
Buffer: 100 of 100
Clock: Internal
Run: Screen
Mode: Auto
Source: Analyzer

Trigger: DIO7=0 AND DIO6=0 AND DIO5=0 AND DIO4=0 AND DIO3=0 AND DIO2=0 AND DIO1=0 AND DIO0

Ready 2015/01/25 17:32:39.864
2048 Samples at 40.05 kHz / 24.98 us Zoom: -1.00 X
Position: -16.15 ms Name: Count [0] LSB
Value: 1

Name	DIO	Trigger
Count		
[7] MSB	7	0
[6]	6	0
[5]	5	0
[4]	4	0
[3]	3	0
[2]	2	0
[1]	1	0
[0] LSB	0	0

Count: 243
T: -17.1 ms
Count [0] LSB Period: 2.62 ms / 381.26 Hz / 50.48 %
Count [0] LSB Pulse width: 1.32 ms

Ready

F. Measure the Period of the Signals and the Counter rate.

Using the Hot Track feature, verify that the counter is running at the speed that you predicted in your pre-lab.

G. Demonstrate your Measurement

DEMO

Demonstrate your successful measurement verifying that the counter values are correct and the count frequency is as predicted (30%). Take a screenshot of your waveform and upload to the course website (10%).