

**Experiment # 4**  
Counters and Logic Analyzer

1. Synopsis:

In this lab we will build an up-counter and a down-counter using 74LS76A J-K Flip Flops. The counter output will be observed on the Oscilloscope and the Logic Analyzer. The objective of this lab is to understand the operation of the counter, the differences between an Oscilloscope and a Logic Analyzer, and how to setup the Logic Analyzer (Agilent Logic Analyzer model: 1664A) to measure the desired signals.

2. Logic Analyzer Theory and Description:

2.1 What is a Logic Analyzer?

A Logic Analyzer (shown in figure 1) is a tool very similar to an Oscilloscope. It measures multiple signals and displays the output vs. time on the screen. One main difference between an oscilloscope and a logic analyzer is that an oscilloscope can precisely display the voltages of the different signals (example: 3.85V, 4.25V). A logic analyzer can not do this. It displays the signals in digital form (either a 1 or a 0). So whether the signal is at 3.85V or 4.25V, it will be recorded as simply ‘1’ (HIGH). This may seem less useful, but is actually an advantage. Since the logic analyzer displays 1’s and 0’s, it is much easier to display more signals at once and debug a problem in a digital system.

A logic analyzer can usually display many more signals than an oscilloscope can. Most scopes display 2 to 4 signals while logic analyzers can usually display or record at least 32 signals at once. The logic analyzer we have in the lab has two pods (group of connections) each containing 16 pins to connect to your circuit. You can see that this would be very helpful for debugging a system with many digital signals.

Another difference which was hinted at above is the ability to store signals. A logic analyzer can usually store all of the signals it monitors in its signal storage memory. A traditional oscilloscope can not typically store what happened in the past. (A digitizing oscilloscope can store signals). So *a traditional oscilloscope needs the signals to be repetitive for it to display the signal in a stable manner*. Unlike a traditional oscilloscope, a logic analyzer stores signals and does not require signals to be repetitive. This can be an advantage in many situations where signals are non-repetitive. Many digital systems will wait for a start signal, do the needed computation and go to the DONE state and may not behave like a simple repetitive counter. A logic analyzer is suitable in debugging such systems.
A very important concept to learn is when to use the correct tool. The following sections will give a few guidelines for choosing the right one.

2.2 When to use an Oscilloscope?

An oscilloscope is normally used when you need to see small voltage changes in a signal. A logic analyzer cannot do this because it only measures a signal digitally (either 1 or 0). When you wish to see the actual (analog) behavior of a signal such as raise time, fall time, overshoot, ringing, noise, signal deterioration, you use an oscilloscope.

Another situation where a scope is needed is for very fast signals. A scope can precisely measure the time between two signals and usually at a higher resolution than a logic analyzer.

2.3 When to use a Logic Analyzer?

A logic analyzer is normally used when you need to see many signals at the same time. A scope just cannot do this well. In debugging mother-board of a computer, you may want to see all address signals, all data signals, and several control signals needing 80 to 128 channels!

Another advantage of using a logic analyzer is that it sees signals as other digital components in the system would. If a voltage appears to be a 1 signal on a scope it might not be interpreted as a 1 by the digital hardware if the voltage is not high enough. A logic analyzer will see the signal just like the digital hardware would and show these kinds of problems.
A third advantage of a logic analyzer is the ability to do complicated triggering. A logic analyzer has the ability to look for a certain pattern of signals and then watch what happens after the pattern is encountered. A scope can only trigger on a raising edge or falling edge of one signal so this is not possible. In digital systems, you may frequently need to watch for a certain pattern of signals to debug a specific problem. Maybe you want to capture the signal activity when a printer is accessed just after a page-feed-escape-control-sequence was sent to the printer. A logic analyzer works best for this as you will see in the second part of the lab.

2.4 Analysis modes of the Logic Analyzer

There are normally two main analysis modes in a logic analyzer. They are Timing Analysis and State Analysis.

**Timing Analysis** is similar to a digital oscilloscope. The vertical axis is the signal (1 or 0) and the horizontal axis is time. For a given time step, which can be set up by us, it will measure all of the signal values at each step. Each time step will be represented by one position on the x-axis. A logic analyzer can usually store many time measurements. After it is done measuring, the signals will be displayed on the screen. The trigger point is shown in the middle of the screen. The logic analyzer keeps filling up the storage memory with signal information on a continuous basis (treating the memory as a circular buffer) while waiting for the trigger point. After the trigger point, it (the logic analyzer) fills one half of the memory so that we can see signal activity before and after the trigger.

One part of timing analysis that is different from a digital scope is the triggering. On a scope the signal is displayed immediately after the trigger occurs. A logic analyzer continuously captures the data and stops after the trigger signal is contained in the data. The data on both sides of this signal (before and after) is available for review as stated before.

**State Analysis** is a little bit different. In the Timing Analysis mode, the analyzer records the value of each signal at regular time intervals. The frequency at which this “scanning” takes place is determined by the logic analyzer’s internal clock frequency (in our analyzer this could be 125 or 250 Mhz). In case of State Analysis, signal values are measured and recorded based on an external clock. This external clock is usually the system clock used in the design that is being debugged. So, basically, in State Analysis mode, the logic analyzer samples data on the clock but records only those samples which satisfy some specified criterion. Of course, this gives us lower timing resolution of events, but in many situations this summarized view of what’s happening in the system is very useful. An example of where state analysis could be useful is microprocessor-based system design. Suppose, you want to know the sequence of memory accesses the system went through. Typically, a microprocessor does not request data from memory on every clock. So, essentially, you want to take measurements every time the *address strobe* signal is activated. So, you may specify the capture criterion as: *Collect one sample on the falling edge of microprocessor clock when the address strobe signal is active*. The address strobe signal goes high only once during one memory access and hence you collect only one sample.
2.5 Key Descriptions and Functions of the logic analyzer

There is a group of six buttons at top left corner of the button area on the logic analyzer. These six buttons take you to different menus from which you can setup the logic analyzer to record the desired data. The six buttons are System, Config, Format, Trigger, List, and Waveform (see figure below)

![Figure 2: Menu buttons](image)

**System:** This button takes you to a menu where you can access files on the disk. The logic analyzer has the capability of storing and loading all of its settings. This is very helpful because it can take a while to get the logic analyzer setup for a specific project.

**Config:** The config button is the first step in setting up the logic analyzer. Here you can choose timing or state analysis. Note: we will normally just use the 1st analyzer.

**Format:** This menu is very important because it sets up the signals that will be measured with the logic analyzer. These signals are named by using the arrow keys to highlight a label. Press SELECT and then name the signal. Press DONE when complete. The next important step is to map the label to a specific pin or group of pins connected to your circuit from the logic analyzer. To do this highlight the area under the correct POD and press SELECT. Press SELECT again to pick a specific pin. Press DONE when finished.

**Trigger:** The trigger menu is used to tell the logic analyzer when to start taking the measurements. The type of analysis you are doing (Timing or State) will determine the type of triggering and how it is setup. There can be many states to go through before the trigger happens and these are setup here.

**List:** The list menu shows a list of all the labels you had setup in the FORMAT menu and the type of signals (binary or hex) expected out of each label.
Waveform: This shows all of the signals you labeled and the results (before and after) after the trigger. This is normally the menu you will be on when you press the run button to see the results.

The following buttons help you edit the menu’s above or start the measurements.

Arrow Keys: These should be self-explanatory if you have used a computer before.

Select: This button selects a highlighted item for changing.

Done: Use this button to confirm when you are done changing something in a menu.

Rotary Knob: This knob can control multiple items and is usually used when there is a list to scroll through.

Run: This button puts the logic analyzer in measurement mode where it waits for the appropriate trigger before taking the measurements. If the signals are not changing fast enough, it may take some time for the analyzer to fill-up its memory.

Stop: This stops the logic analyzer from taking further measurements. You may want to use this if the analyzer is taking too much time to fill-up the buffer memory and you wish to abort and see what it has gathered so far.
3. Prelab:

Q 3.1: Using your outstanding memory (!?) of EE101, write the function table of a J-K flip flop and describe what will happen when you tie J and K both to Vcc. (4+1pts).

<table>
<thead>
<tr>
<th>J</th>
<th>K</th>
<th>Q</th>
<th>( \bar{Q} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

When J and K are both tied to VCC, ---------------------------------------------------------------

Q 3.2: Refer to the data sheet of 74LS76A JK flip flop and determine whether it is a positive edge triggered flip flop or negative edge triggered flip flop. (5 pts)

Q 3.3: What are the three main differences between an oscilloscope and a logic analyzer? (12pts)

<table>
<thead>
<tr>
<th>Oscilloscope</th>
<th>Logic Analyzer</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Q 3.4: In the state analysis mode, what should the CLOCK input (from the Logic Analyzer’s pod) be connected to? (5 pts)

Q 3.5: Which menus would we go to for each of the following operations? (3 pts)

<table>
<thead>
<tr>
<th>Operation</th>
<th>Menu</th>
</tr>
</thead>
<tbody>
<tr>
<td>To load a format file from the disk</td>
<td></td>
</tr>
<tr>
<td>To switch from timing analysis to state analysis mode</td>
<td></td>
</tr>
<tr>
<td>To look at the signals that we are investigating (in a graphical way)</td>
<td></td>
</tr>
</tbody>
</table>
4. Procedure:

Part 1: 4-bit ripple counter design using JK flip-flops

Here we are building a 4-bit ripple counter (which counts up only and rolls over) using four JK Flip-flops (two 74LS76A chips). We display the count on 4 singular LEDs.

4.1 Using the 74LS76A JK flip-flops connect the circuit as shown in Figure 3. The pin out of the 74LS76A is given on the side. Notice that the JK flip-flops need to be in toggle mode for the circuit to work properly. The clear (CLR) pins from each flip-flop should be connected together and then to a switch as shown. This CLR signal is active low so connecting opening the switch disables the clear function. to start with keep the switch closed. After you switch-on power, confirm that the counter outputs are all zero. When you open the switch, the counter starts counting.

4.2 We want you to experiment both sourcing mode and sinking mode of connecting LEDs. Connect QA and QB to LEDs in sourcing mode and, QC and QD to LEDs in sinking mode as shown in Figure 4. The current-limiting resistance in series with each LED is 330 ohms.

4.3 We want to clock the above counter using 40KHz clock. Generate a 0 to +5V square wave at 40kHz and verify it on the oscilloscope.

4.4 After verifying the clock signal, connect it (the output of the function generator) to the clock input on your counter circuit. Open the switch controlling the CLR signal All LEDs should appear ON if your circuit is working properly. Note: Actually the LEDs are flickering really fast which makes them appear to be ON continuously.
4.5 Now adjust the input clock frequency from 40kHz to about 1Hz on the function generator. You should now see the four LEDs going through the binary counting sequence 0000 to 1111.

4.6 Connect the MSB (Most Significant Bit) QD to the Oscilloscope on CH1. Set the triggering slope to falling edge. Now connect each of the other four signals QC, QB, QA, and CLOCK to CH2 of the scope, one at a time in that order. You should see that each of the signals is 2x as fast as the one before it. Verify that your signals match the waveform shown in figure 5 below.

![Figure 5: 4-bit counter waveform](image)

Part 2: Logic Analyzer

Next, we are going to capture the output of the counter using the Logic Analyzer.

Logic Analyzer in **Timing Analyzer mode**:

4.7 Connect each of the outputs of the counter (QA, QB, QC and QD) to the logic analyzer via pins 1 through 4 of POD 2, as shown in figure 6. Connect QA to pin 1, QB to pin 2 and so on. Connect the clock CLK also to a pin (pin 0 as shown here) like any other signal. (Here, in Timing Analyzer mode, you do NOT need to connect the clock signal of your circuit to the pin labeled as CLK on the pod.) Finally, connect your circuit’s ground to the GND pin (black wire) of the logic analyzer. This way, your circuit and the logic analyzer shall have a **common ground** as a common reference.

![Figure 6: Logic analyzer pod connection](image)

4.8 Most of the setup of the logic analyzer has been done for you. Load the format file (**lab4_TA_A**) (TA = Timing Analyzer; just a name to remember) by doing the following steps:

- Turn on the logic analyzer with the boot-disk in its 3.5” floppy drive and wait for the logic analyzer to boot up. Now remove the boot disk and insert the disk with configuration files. When it boots up, it will be in the config menu. It will say so at the top. Make sure that the analyzer is in the **Timing Analyzer mode**.
Press the SYSTEM button to switch to the system menu. Use the rotary knob to select the file named lab4_TA_.A.

- Verify that the operation selector shows “load”. If it does not say “load” use the arrow keys to highlight the operation selector. Press the SELECT button and use the arrow keys to select LOAD. Press DONE when you are finished.

- Now, use the arrow keys to highlight the EXECUTE button and press the SELECT key. The settings for the logic analyzer will now be loaded.

4.9 You should look through the other five menus by pressing the different buttons and look at the settings you just loaded. Specifically, you should look at the TRIGGER menu.

4.10 Go to the FORMAT menu. Here you can see that two of the labels, QD and QC, have been setup for you. You need to setup the logic analyzer for QB and QA. Completed FORMAT is shown in figure 7.

Because you want the information after clear, you can configure the Timing Trace specification menu so that the Analyzer shows the trace point after CLEAR = 1 and QA = QB = QC = QD = 0 at the positive edge on QA.
After the format setup is complete, go to the WAVEFORM menu. Close the clear switch to keep the counter cleared. Press the RUN key and the logic analyzer will wait for the trigger. Open the clear switch. Soon, the logic analyzer will display the captured waveform which will look similar the one shown in figure 9 below.

Figure 8: Timing Trace Specification

Figure 9: 4-bit counter waveform
By adjusting the time/division setting and delay setting of the waveform menu, you can change the TIME scale magnification and scroll the display to make it appear as below.

Logic Analyzer in **State Analyzer mode:**

In some state machine analysis, it may be desirable to obtain a listing of states rather than waveforms. For example, in a microprocessor-based system debugging, you may want to know the sequence of memory accesses performed by the processor with out unnecessary details. The Agilent logic analyzers have a sophisticated state specification mechanism to capture just the right information you are looking for.

4.11 The logic analyzer can be configured as a **state analyzer** in the configuration menu.

4.12 Counter connections: The J clock of the POD1 shall now be connected to the clock of the counter.

4.13 State Trace specification: Here, note that on negative transition of the clock the counter changes count. Hence it is appropriate to sample the **counter state** at the positive edge of the clock when the count is **stable**. Note that we specified $J^\uparrow$ in the state format specification, which means that the logic analyzer should sample the data at the positive edge of the J clock when the count is **stable**.
Figure 11: State Format Specification

<table>
<thead>
<tr>
<th>Label</th>
<th>CLEAR</th>
<th>QA</th>
<th>QB</th>
<th>QC</th>
<th>QD</th>
<th>COUNT</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
</tbody>
</table>

```
Clear: K1 15...B7...0
QA:    15...B7...0
QB:    15...B7...0
QC:    15...B7...0
QD:    15...B7...0
COUNT: 15...B7...0
```

Figure 12: State Trigger Specification

<table>
<thead>
<tr>
<th>Label</th>
<th>CLEAR</th>
<th>QA</th>
<th>QB</th>
<th>QC</th>
<th>QD</th>
<th>COUNT</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
</tbody>
</table>

```
a: 1 0 0 0 0 0 0
b: x 1 1 1 1 1 15
```

```
Modify Trigger
Arming Control
Acquisition Control
Count Off
```
4.14 The above trace specification captures one complete count sequence (no more, no less).

![Figure 13: State Listing](image)

<table>
<thead>
<tr>
<th>Label</th>
<th>CLEAR</th>
<th>QA</th>
<th>QB</th>
<th>QC</th>
<th>QD</th>
<th>COUNT</th>
</tr>
</thead>
<tbody>
<tr>
<td>-15</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>00</td>
</tr>
<tr>
<td>-14</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>01</td>
</tr>
<tr>
<td>-13</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>02</td>
</tr>
<tr>
<td>-12</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>03</td>
</tr>
<tr>
<td>-11</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>04</td>
</tr>
<tr>
<td>-10</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>05</td>
</tr>
<tr>
<td>-9</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>06</td>
</tr>
<tr>
<td>-8</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>07</td>
</tr>
<tr>
<td>-7</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>08</td>
</tr>
<tr>
<td>-6</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>09</td>
</tr>
<tr>
<td>-5</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>10</td>
</tr>
<tr>
<td>-4</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>11</td>
</tr>
<tr>
<td>-3</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>12</td>
</tr>
<tr>
<td>-2</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>13</td>
</tr>
<tr>
<td>-1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>14</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>15</td>
</tr>
</tbody>
</table>

Figure 13: State Listing
5. Lab Report:

| Name: ______________________ | Date: _______________________ |
| Lab Session: ________________ | TA’s Signature: ______________ |

For TAs: Prelab (out of 30): ______ Hardware (out of 20): ______
Waveform on Oscilloscope (out of 20): ____________
Waveform on Logic Analyzer (out of 20): ____________
State Listing on Logic Analyzer (out of 20): ____________
Report (out of 40): ____________

Comments:

Q 5.1: Draw the complete circuit (including LEDs, resistors, etc.) as you wired for the counter. Label properly and write the pin number for each input/output. (10 pts)

Labeling convention:
U1.4 = Component “U1” pin 4
First 74LS76A = U1
Second 74LS76A = U2

Q 5.2: What will you need to change in the circuit to make the counter a down counter instead of an up counter? Note: Please use Q outputs only. Do not use Q outputs. (5 pts)
Q 5. 3: Given below is an incomplete design for an up/down 3-bit counter. When the control line (up/down) is HIGH, the counter should count up (000, 001,..., 111), else it should count down (111, 110,..., 000). The desired behavior of the up/down counter is shown in figure 8. Complete the design by adding necessary circuitry and explain how the counter will work.

Note: If you need to make different connections to the clock input of a JK FF, you can use a circuit equivalent to a 2-to-1 mux (two AND gates and an OR gate). (15 pts)

Explanation:

Q 5. 4: Suppose you are building a 16-bit counter (output Q[15:0]). If your TA asks you to display Q9 and Q11 on the dual channel oscilloscope, would you trigger the scope with Q9 or Q11 or any one of them? Explain (10 pts).