Tiffany Q. Liu
March 9, 2011
CSC 270
Lab #6

Lab #6: D Flip-flops and Finite State Machines

Introduction

The focus of this lab was working with the 74LS74 D flip-flop and building finite state machines.

Materials

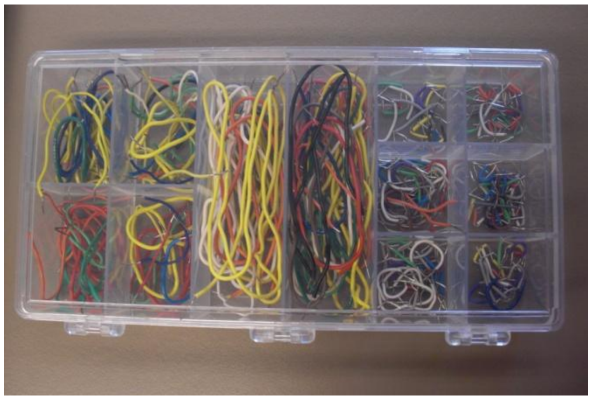


Figure 1. Wiring Kit.

Figure 2. Digital Training Kit.

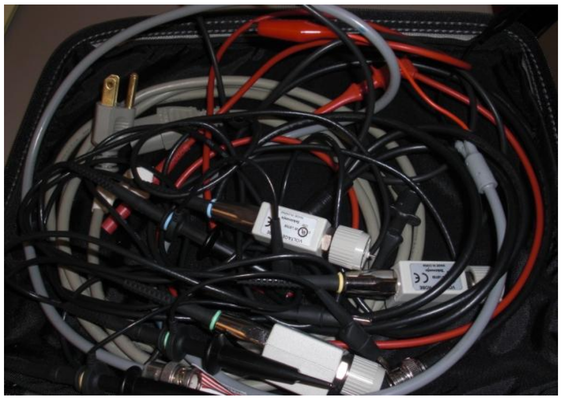


Figure 3. Oscilloscope cables.

Figure 4. Tektronix MSO3000/DPO3000 Oscilloscope

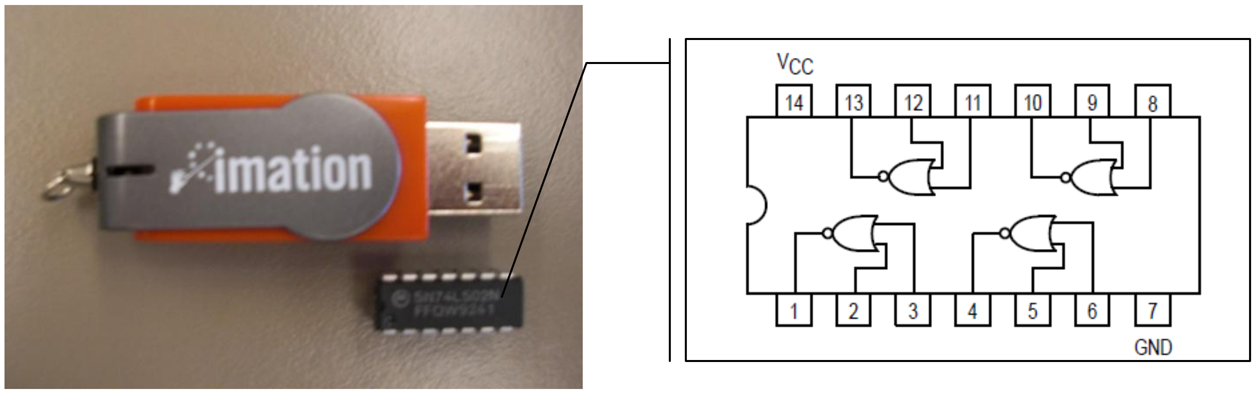


Figure 5. Quad 2-Input NOR Gate 74LS02 Compared to a USB Flash Drive.

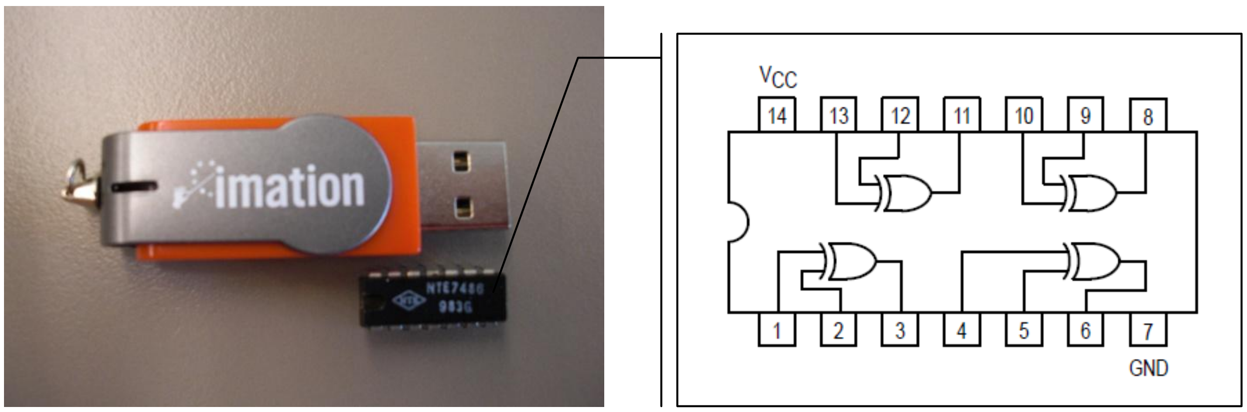


Figure 6. Quad 2-Input XOR Gate 74LS86 Compared to a USB Flash Drive.

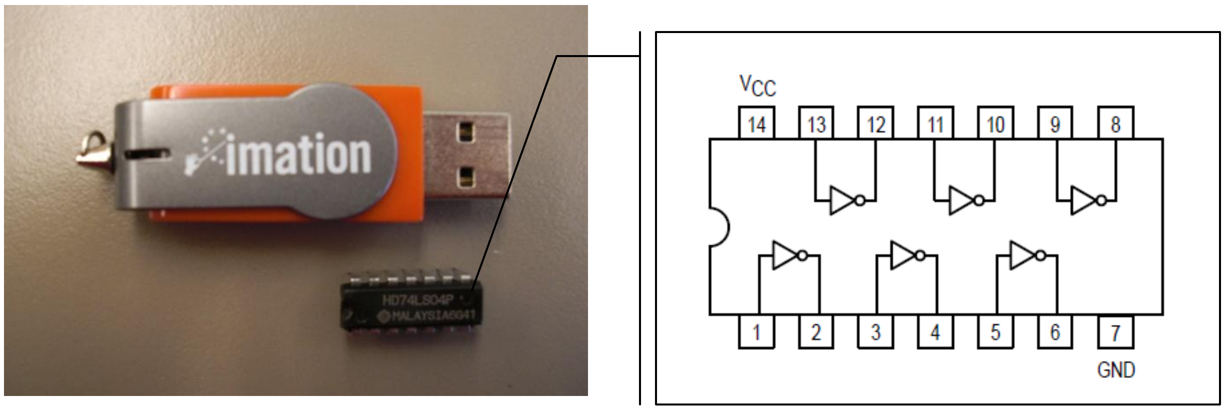


Figure 7. Hex Inverter 74LS04 Compared to a USB Flash Drive.

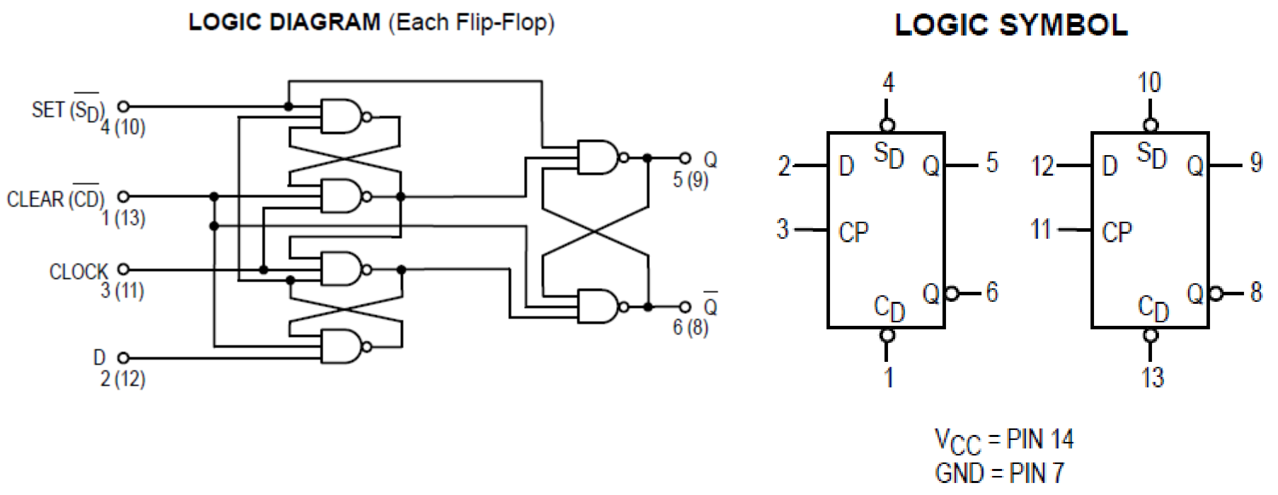


Figure 8. Dual D-Type Positive Edge-Triggered Flip-Flop 74LS74.

The 74LS74 D Flip-Flop

For this part of the lab, we investigated how the 74LS74 D flip-flop worked. First we wired a circuit with the D 74LS74 such that the Preset, Clear, and D were connected to their own switches; Clock was connected to the 1Hz clock signal; and D, Clock, Q, and Q-bar to 4 LEDs (Fig. 9):

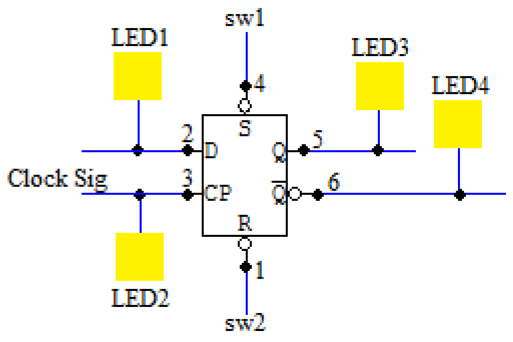


Figure 9. 74LS74 D Flip-Flop Circuit.

Once the circuit was wired up, we first set both Preset and Clear to 1 and observed that Q was capturing what D was with every tick of the clock. On a timing diagram, a tick is represented by a low to high edge:

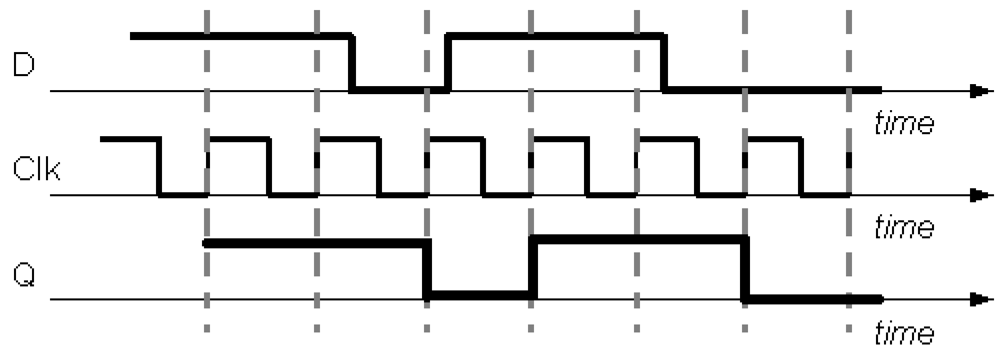


Figure 10. Timing Diagram for Circuit from Figure 9 Describing D, Clock, and Q When Clear and Preset are 1.

Every time the clock ticked, Q captured the state of D. Otherwise, Q was storing what it previously captured. Then we activated the clear input by setting it to 0 and made the following observation:

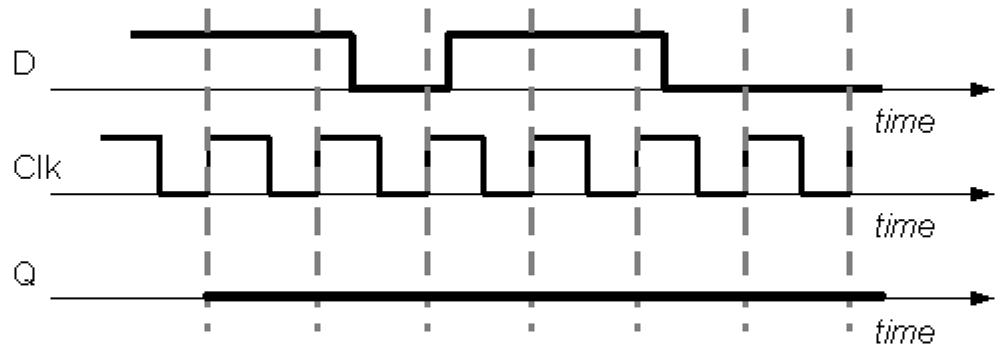


Figure 11. Timing Diagram for Circuit from Figure 9 Describing D, Clock, and Q When Clear is 0 and Preset is 1.

Q remained at 0 no matter what the clock or data was doing. This is because activating the clear input forces Q to 0. Finally, we returned Clear to 1 and activated the preset input by setting it to 0 and made the following observation:

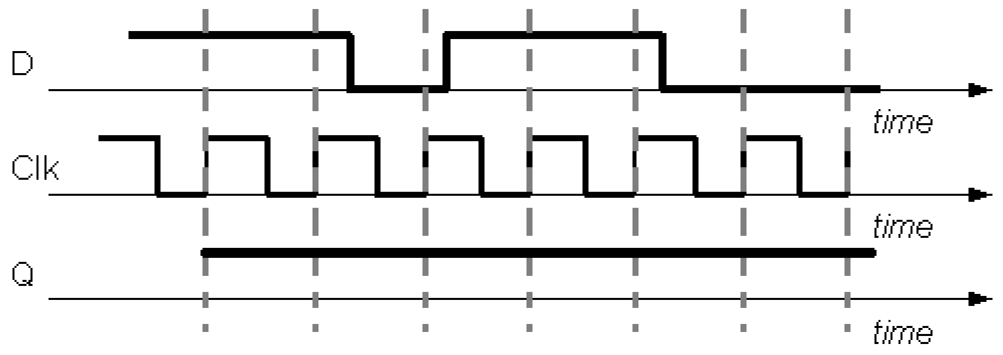


Figure 12. Timing Diagram for Circuit from Figure 9 Describing D, Clock, and Q When Clear is 1 and Preset is 0.

This time, Q remained at 1 no matter what the clock or data was doing since activating the preset input forces Q to 1.

Finite State Machine: 1-Bit Flip-Flop Sequencer/Oscillator

Next we implemented a 1-bit flip-flop sequencer with a 74LS74 and a NOT gate by connecting Clock to the 1Hz clock signal, Clear and Preset to switches set at 1, the output Q to the input of an inverter and the output of the inverter to the input D; Clock and Q were also connected to LEDs:

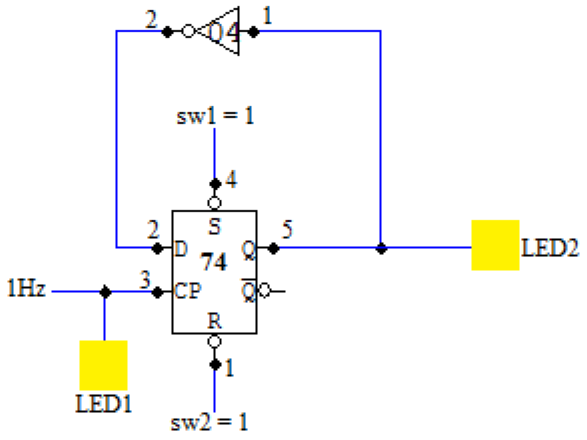
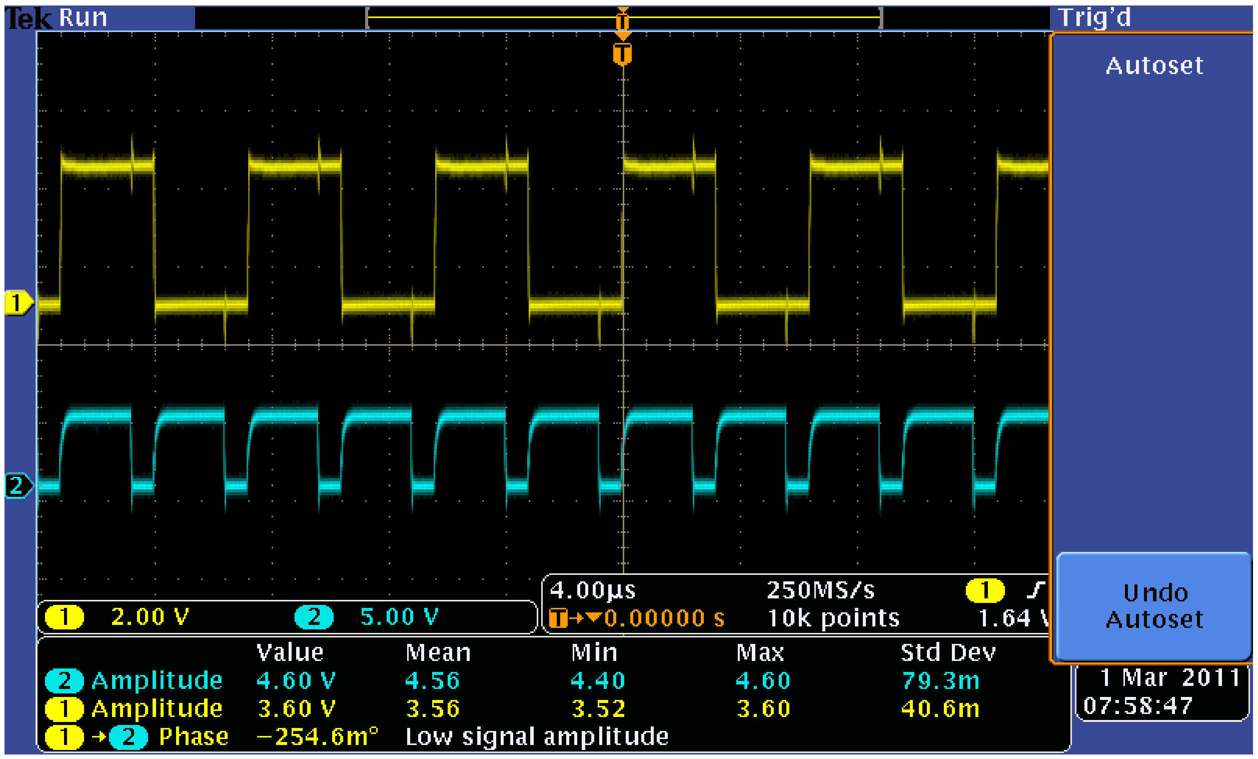


Figure 13. 1-Bit Flip-Flop Sequencer

We observed that with every two ticks of the clock, Q ticked once as indicated by the LEDs. This meant that the period of Q was twice the period of the clock. Since frequency is inversely related to the period and knowing that the frequency of the clock was 1 Hz, we can conclude that the frequency of Q was half the frequency of the clock or 0.5Hz.

We then set the clock frequency to 100kHz and connected Clock and Q to the oscilloscope to observe the waveforms generated:



```
1  2.00 V          2  5.00 V        4.00µs          250MS/s      1  ⨍
                                    T→▼0.00000 s    10k points   1.64 V
                Value       Mean     Min      Max      Std Dev
2 Amplitude     4.60 V      4.56     4.40     4.60     79.3m
1 Amplitude     3.60 V      3.56     3.52     3.60     40.6m
1 → 2 Phase     −254.6m°    Low signal amplitude
                                                                 1 Mar 2011
                                                                 07:58:47
```

Figure 14. Screen Capture from Oscilloscope of Clock (yellow) and Q (blue).

It can be seen from the scope that with one period of the clock, Q had two periods. This meant that Q had half the frequency of the clock, which indicates that this 1-bit flip-flop sequencer is a frequency divider.

This circuit can be viewed as a finite state machine (FSM) with 2 states (0 and 1). The following is a diagram of this FSM:

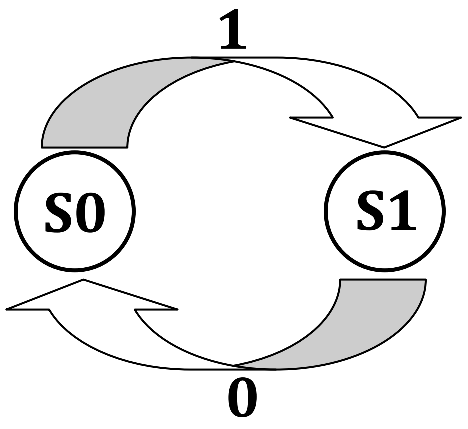


Figure 15. State Diagram for Circuit from Figure 13.

Finally, knowing that the circuit in Figure 13 is a divide-by-2 frequency divider, we can implement a divide-by-4 frequency divider using both of the D flip-flops in the 74LS74. Since one D flip-flop divides the clock signal in half at the output Q, then we can put that output as the clock signal of the second D flip-flop, which will then output a signal with a frequency that is a quarter of the first clock signal's frequency. For this, we built the following circuit:

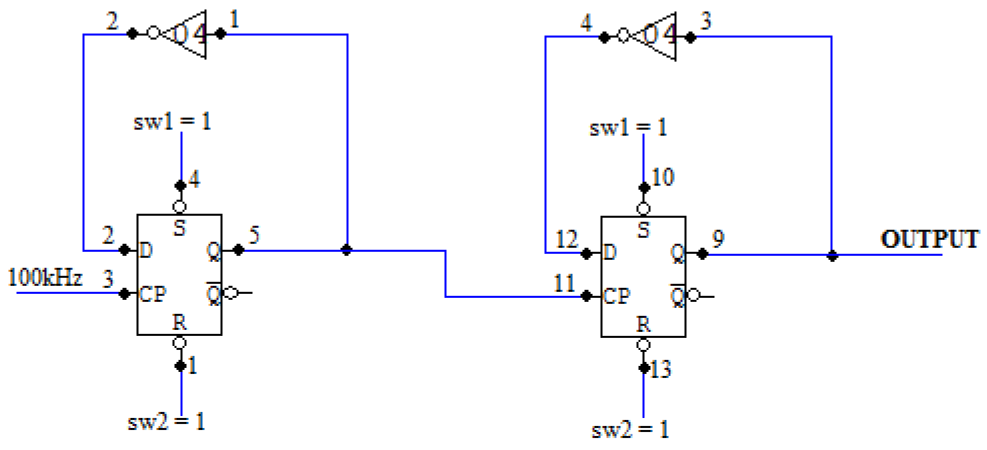
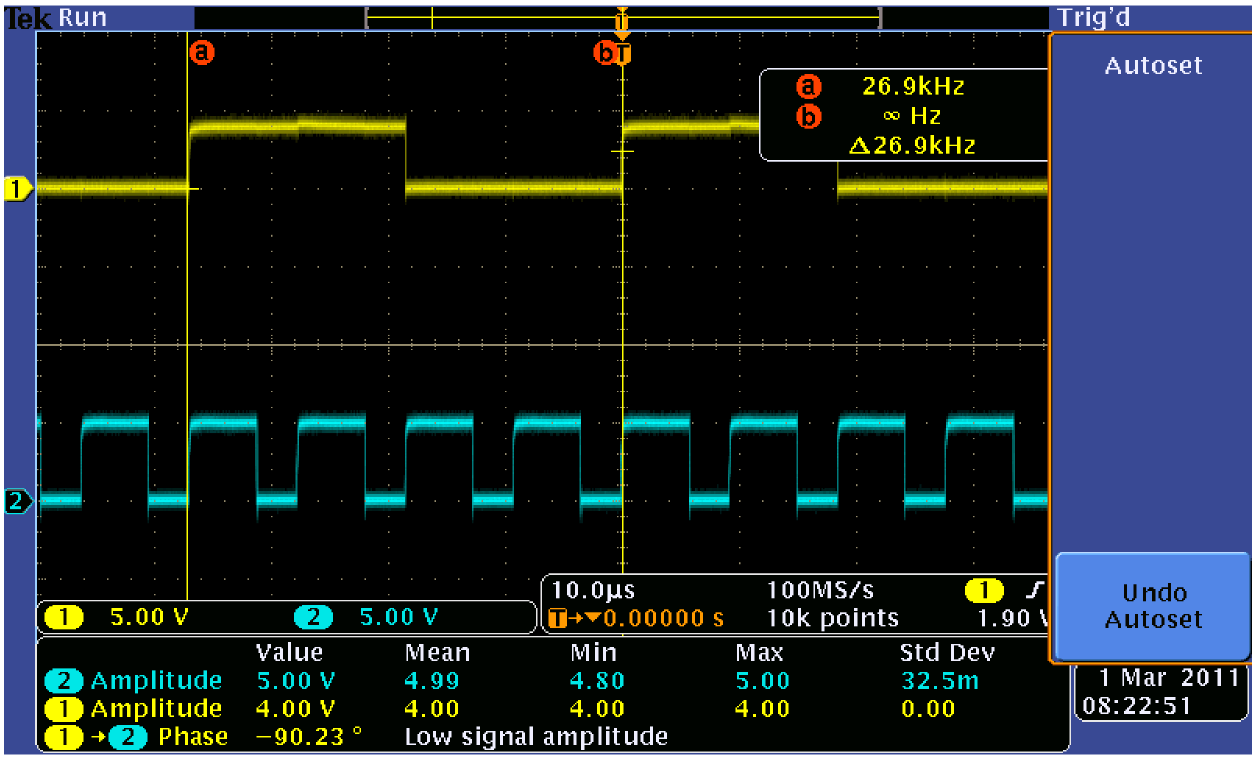


Figure 16. Circuit Diagram for a Divide-By-4 Frequency Divider.

We confirmed that our circuit was in fact a divide-by-4 frequency divider by connecting both the clock and Q to the oscilloscope to observe their waveforms:



```
a   26.9kHz
b   ∞ Hz
  Δ26.9kHz

1  5.00 V        2  5.00 V
10.0µs           100MS/s        1  ∫
T→▾0.00000 s     10k points     1.90 V

                 Value     Mean    Min     Max     Std Dev
2 Amplitude      5.00 V    4.99    4.80    5.00    32.5m
1 Amplitude      4.00 V    4.00    4.00    4.00    0.00
1 → 2 Phase      −90.23 °  Low signal amplitude

1 Mar 2011
08:22:51
```

Figure 17. Screen Capture from Oscilloscope of Clock (yellow) and Q (blue) of Circuit from Figure 16.

As can be seen from the screen capture, within one period of the clock signal, Q had a 4 periods, which indicates that Q had a frequency that was a quarter of the clock's frequency.

Finite State Machine: 2-Bit Flip-Flop Sequencer

For this part of the lab, we built two FSMs, each using 2 D flip-flops. For the first machine, flip-flop 1 had input D1 and outputs Q1 and Q1'. Flip-flop 2 had input D0 and outputs Q0 and Q0'. To build the machine, we connected D1 to Q0, D0 to the output of Q1 NOR Q0, Preset 1 and Preset 0 to a switch set at 1, Clear 1 and Clear 0 to another switch also set at 1, the two clocks to the 1Hz clock signal, and Q0 and Q1 to two LEDs:

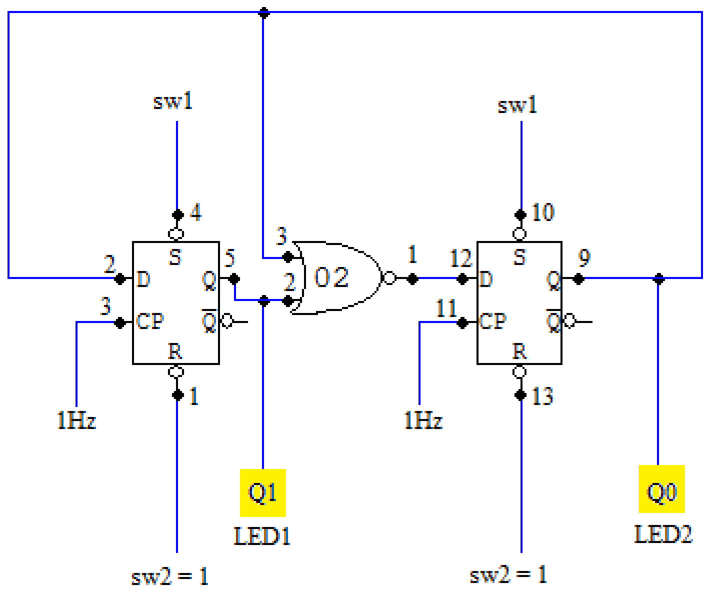


Figure 18. 2-Bit Flip-Flop Sequencer Finite State Machine 1.

After observing the LEDs, we determined that the machine had 3 states where Q1 and Q0 could be 1 and 0, 0 and 1, and 0 and 0 respectively. We then activated the switch that controlled the two presets momentarily and then deactivated it. When we did that, the machine went into a fourth state where Q1 and Q0 were both 1, and as soon as we deactivated the presets, the machine went back to having only the original 3 states. The following is a simplified state diagram of this machine:

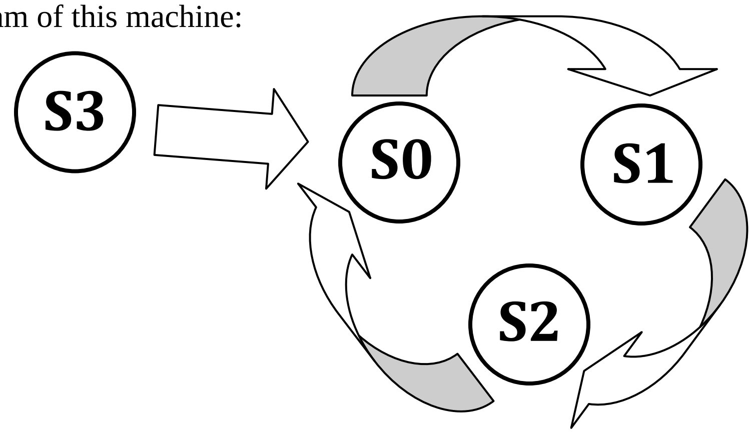


Figure 19. Simplified State Diagram of FSM 1 (Figure 18).

For the second machine, we left everything from the first machine as is except we connected D0 to the output of the inversion of Q1 XOR Q0 instead of the output of Q1 NOR Q0:

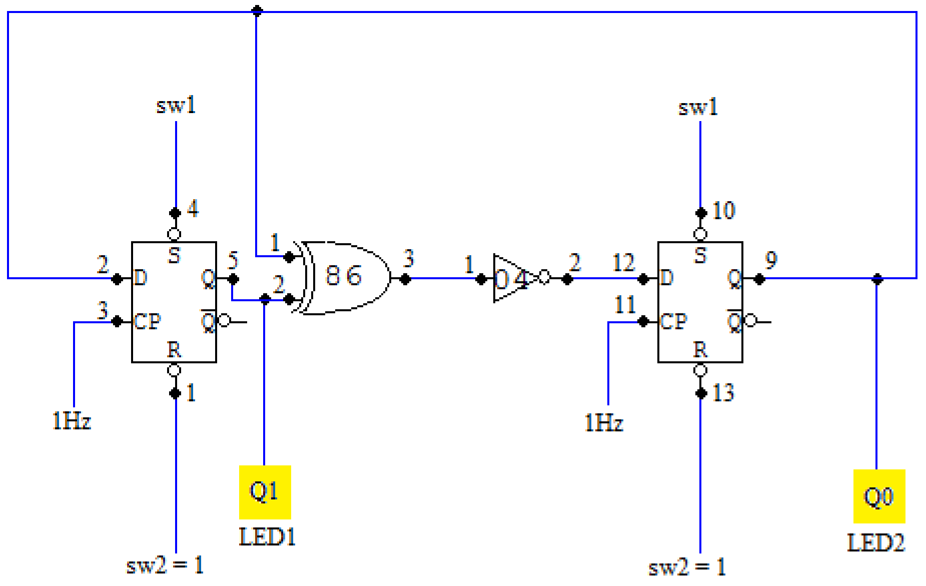


Figure 20. 2-Bit Flip-Flop Sequencer Finite State Machine 2.

When we wired up our second machine, we observed that it had the same 3 states as the first machine. However, when we activated the switch that controlled the two presets momentarily and ten deactivated it, the machine could get stuck in the fourth state where both Q1 and Q0 were 1 (depending on the tick of the clock, the machine could remain in the original three states). The following is a simplified state diagram of this machine:

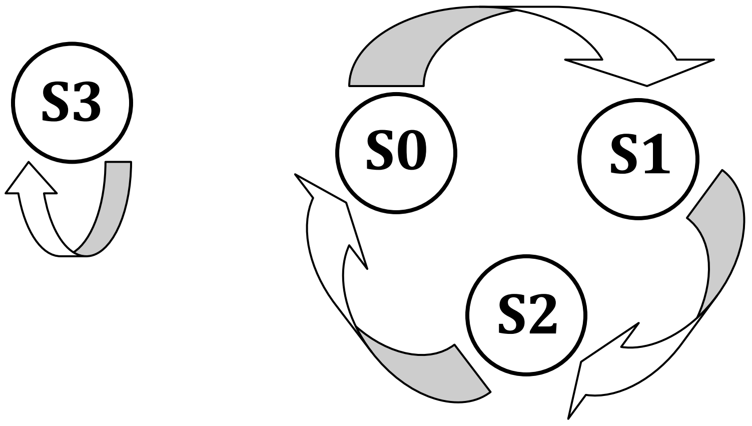


Figure 21. Simplified State Diagram of FSM 2 (Figure 20).

Finite State Machine with Output

Finally we took our second machine and toggled the switch controlling the presets until the machine got back to having 3 states. Then we used outputs Q1 and Q0 to generate a signal that is 1 for 1 period of time and 0 for two period of time. Since either Q1 or Q0 is 1 and 0 for two states and both Q1 and Q0 are 0 for one state, we noticed that if we XORed Q1 and Q0 we would get a signal that is 1 for two states and 0 for one state. Since we wanted the opposite, we then took the output of the XOR and inverted it:

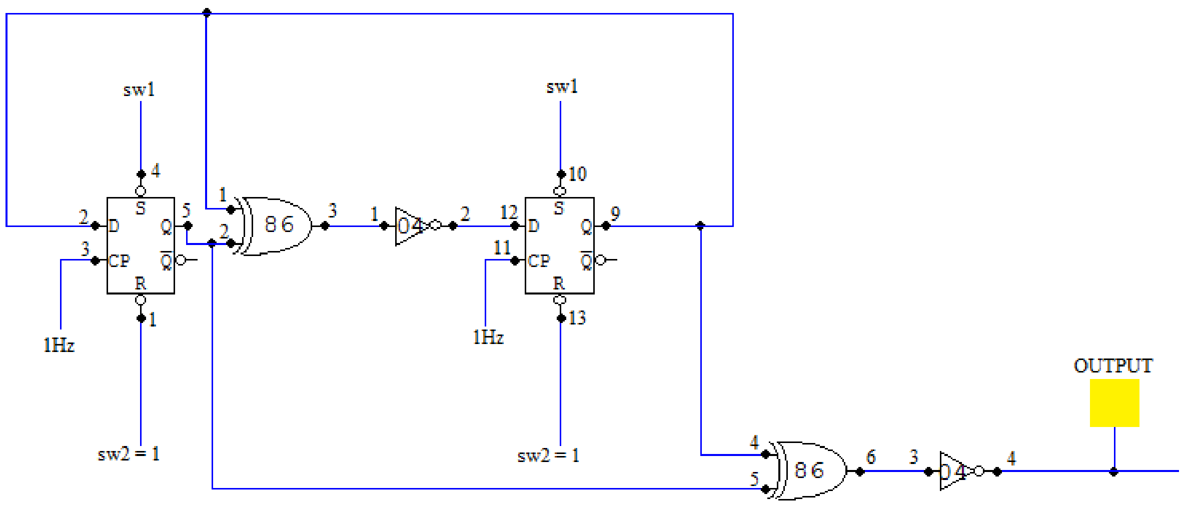


Figure 22. 2-Bit Flip-Flop Sequencer Finite State Machine 2 with Output.

When the circuit was wired, we found that the LED for the signal we were interested in was off (0) for two counts and on (1) for one count. In other words, it had the following timing diagram:

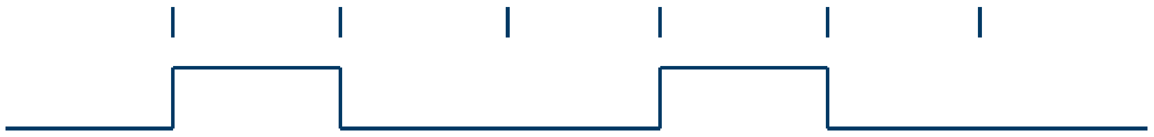


Figure 23. Timing Diagram (Taken from D. Thiebaut) of Output of Circuit from Figure 22.