**PURPOSE**

The purpose of this lab was to design and build a synchronous digital circuit using finite state machines. The duty cycle of the circuit’s output was to depend on the values of two inputs as shown in Table 1. The Altera Quartus II software was used to design and test initial implementations of the circuit, after which it was designed on a protoboard and tested using the Intronix LogicPort Logic Analyzer. The timing diagrams of the two implementations were compared.

The physical circuit design portion of this lab also required an optimization of the total wire length by using as few gate packages as possible. The amount of gate packages was reduced in the beginning of the lab by rearranging the output sequence and reducing the sequential logic required to create the circuit. The circuit was then designed and wired efficiently so as to use the minimal amount of space on the protoboard.

**PROCEDURE**

1. The output sequence was rearranged to that of Table 2. Initially, it was planned that the output bit Z would be the output of the 74151 eight-to-one multiplexor package, as the counter would cycle through each state in the multiplexor over eight clock cycles. The actual sequence of the output bit Z was not relevant as the inputs of the multiplexor could be switched. This change was made to increase the viewing quality of the output waveform on both the Quartus simulation and the Logic Analyzer software.

2. The state transition diagram of the baseline design was drawn with eight states. Since the output of the finite state machine depends both on the present state as well as the inputs, this is a Mealy finite state machine. This transition diagram is shown in Figure 1.

3. The state transition table was drawn for each combination of inputs. These transition tables are shown in Tables 3a-3d.

4. Assuming the use of 74LS74A D flip-flops, the state transition tables were annotated to show the required inputs for each flip-flop, shown in Table 4. As all four inputs result in the same state transition table, eight states were necessary to create the finite state machine. Three, or log2(8), flip-flops would be needed to represent these eight states.

5. Karnaugh maps were used to generate the next-state logic for D2, D1, and D0. These Karnaugh maps are shown in Figures 2a-2c. These flip-flops would effectively iterate through each position in the 8-bit clock cycle, which would in turn iterate through the 74151.

6. The logic for the inputs of the eight-to-one multiplexor was generated. Shown in Table 5, the logic for each multiplexor input was derived from its corresponding column in Table 5 and is shown in Equations 2a-2h.

7. The design was then optimized by using a synchronous three-bit counter instead of the 7474-based counter (TA Chen Pan confirmed that this was allowed). This three-bit counter would be created by using the four-bit counter from the 74193 package and ignoring the most significant bit.

8. A new project for the design was created in the Altera Quartus software. A new schematic file was added to the design.

9. The baseline design was created using two 74LS74A packages and a 74151 multiplexor. The next-state logic for each state (derived in Step 5) was added using AND and XOR gates as necessary.

10. The preset and clear inputs for each flip-flop were connected to Vcc. Doing this means that the flip-flop will transfer data from the D input to the Q and Q’ outputs at the rising edge of the clock signal.

11. To make the flip-flops synchronous, the external clock signal was connected directly to each individual flip-flop’s clock input.

12. The combinational logic for the inputs of the 74151 multiplexor was generated using one AND and one OR gate. This logic could be (and was eventually) replaced with a 74157 package.

13. A file was created in the MAX+plus II Waveform Editor. Using the Node Finder option, the clock input was added to the waveform. It was given a value of ‘Clock’ and assigned a period of 100 ns. This value is somewhat arbitrary, though it is similar enough to the period of the physical circuit that the results of the two can be compared.

14. The logic inputs A and B were each added to the waveform. The waveform was saved, compiled, and simulated for each input combination of A and B.

15. Another schematic file was created for the optimized design. This design used the 74193 4-bit counter to cycle through the eight-bit output sequence in place of the 74LS74A packages. This design also used the 74157 package to create the value of each input for the 74151 multiplexor, as this package would be used to design the physical circuit. Since the 74157 contains four two-to-one multiplexors, and the multiplexor is a universal logic gate, the functions *AB* and *A + B* could be created using this single package. Both schematics are shown in Figures 3a-3b.

16. Another waveform file was generated with the optimized design. The file was compiled and simulated for each input combination of A and B. The output waveform Y was the same as the output waveform generated with the baseline design. These waveforms are shown in Figures 4a-4d.

17. The circuit was implemented on a protoboard using a 555-timer circuit for the clock input. For the output to correctly display on the LogicPort Logic Analyzer software, a period of 2 ms was chosen. Two 1 kΩ resistors were chosen for the circuit, pictured in Figure 5. Using Equations 2a-2c, the required capacitance was calculated to be 1 μF [1].

18. The circuit was connected in a manner such that the minimal amount of wire was used in connecting the packages.

19. The Intronix Logic Analyzer was connected to the physical circuit at all relevant input and output pins, listed in Table 5.

20. The LogicPort Logic Analyzer software was opened and the trigger pattern for the circuit was selected. The circuit was to trigger on the rising edge and was to depend on the clock signal.

21. Output waveforms for each combination of inputs were viewed. The waveform matched the output sequence in Table 2, so the circuit was verified to be functioning correctly.

**RESULTS**

|  |  |
| --- | --- |
| **AB** | **Z** |
| 00 | 00000001…. |
| 01 | 00000111…. |
| 10 | 00011111…. |
| 11 | 01111111…. |

|  |  |
| --- | --- |
| **AB** | **Z** |
| 00 | 01000000…. |
| 01 | 01010100…. |
| 10 | 01010111…. |
| 11 | 011111111…. |

Table 1: The output sequences for each input combination as given in the laboratory directions

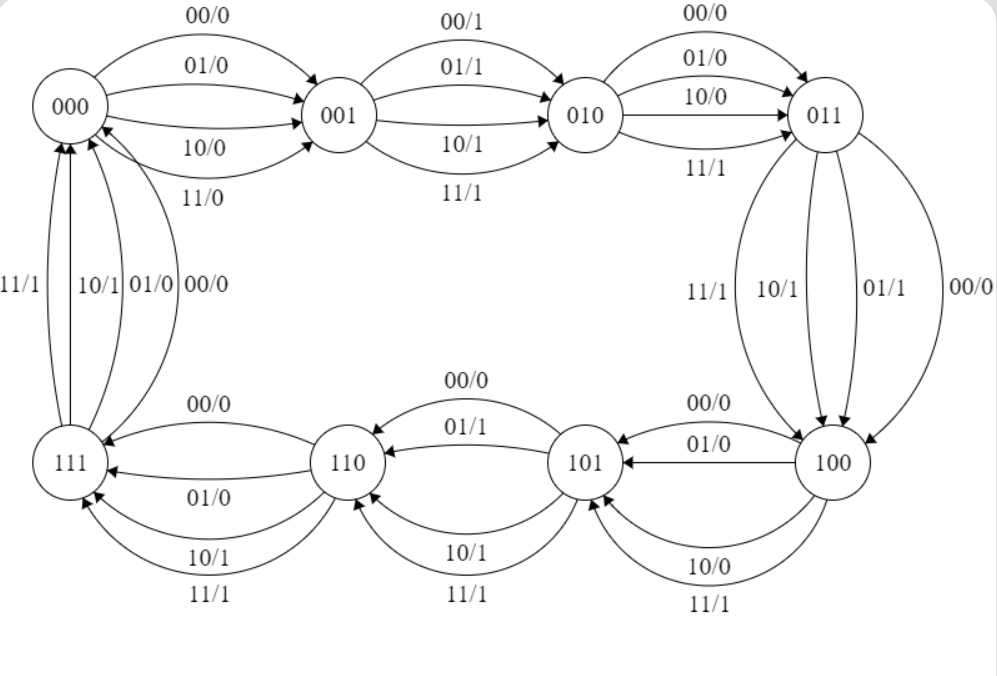


Figure 1: The state transition diagram for the Mealy finite state machine. The number after the slash is the output for that given state and input combination.

Table 2: The output sequences converted to the form used throughout the design process

Tables 1 and 2 above show the change made to the output sequence for each combination of inputs. This change was made primarily for ease of viewing on both the Quartus simulation and the LogicPort software, and it came after Dr. Jones’ encouragement to try different output sequences with the same duty cycle. Figure 1 above is the state transition diagram for the Mealy finite state machine. Tables 3a-3d below show the state transition tables that describe the next-state logic for each 74LS74A flip-flop in the baseline design, while Table 4 annotates this logic to find the data values for the next state of the flip-flop. Figures 2a-2c show these values translated into Karnaugh maps, while Equations 1a-1c show the resulting sum-of-products expression for each data value.

|  |  |  |
| --- | --- | --- |
| **PS** | **NS** | **Y** |
| 000 | 001 | 0 |
| 001 | 010 | 1 |
| 010 | 011 | 1 |
| 011 | 100 | 1 |
| 100 | 101 | 1 |
| 101 | 110 | 1 |
| 110 | 111 | 1 |
| 111 | 000 | 1 |

|  |  |  |
| --- | --- | --- |
| **PS** | **NS** | **Y** |
| 000 | 001 | 0 |
| 001 | 010 | 1 |
| 010 | 011 | 0 |
| 011 | 100 | 1 |
| 100 | 101 | 0 |
| 101 | 110 | 1 |
| 110 | 111 | 1 |
| 111 | 000 | 1 |

|  |  |  |
| --- | --- | --- |
| **PS** | **NS** | **Y** |
| 000 | 001 | 0 |
| 001 | 010 | 1 |
| 010 | 011 | 0 |
| 011 | 100 | 0 |
| 100 | 101 | 0 |
| 101 | 110 | 0 |
| 110 | 111 | 0 |
| 111 | 000 | 0 |

|  |  |  |
| --- | --- | --- |
| **PS** | **NS** | **Y** |
| 000 | 001 | 0 |
| 001 | 010 | 1 |
| 010 | 011 | 0 |
| 011 | 100 | 1 |
| 100 | 101 | 0 |
| 101 | 110 | 1 |
| 110 | 111 | 0 |
| 111 | 000 | 0 |

*Tables 3a-3d: The next-state logic for the input combinations of AB = 00, 01, 10, and 11 respectively.*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **D2** |  | **Q1Q0** |  |  |  |
|  |  | **00** | **01** | **11** | **10** |
| **Q2** | **0** | 0 | 0 | 1 | 0 |
|  | **1** | 1 | 1 | 0 | 1 |

**(a) D2 =** Q2Q1’ + Q1(Q2 ⊕ Q0)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **D1** |  | **Q1Q0** |  |  |  |
|  |  | **00** | **01** | **11** | **10** |
| **Q2** | **0** | 0 | 1 | 0 | 1 |
|  | **1** | 0 | 1 | 0 | 1 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **D0** |  | **Q1Q0** |  |  |  |
|  |  | **00** | **01** | **11** | **10** |
| **Q2** | **0** | 1 | 0 | 0 | 1 |
|  | **1** | 1 | 0 | 0 | 1 |

Table 4: The next-state logic annotated to show the next-state values for each flip-flop.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Q2** | **Q1** | **Q0** | **D2** | **D1** | **D0** |
| 0 | 0 | 0 | 0 | 0 | 1 |
| 0 | 0 | 1 | 0 | 1 | 0 |
| 0 | 1 | 0 | 0 | 1 | 1 |
| 0 | 1 | 1 | 1 | 0 | 0 |
| 1 | 0 | 0 | 1 | 0 | 1 |
| 1 | 0 | 1 | 1 | 1 | 0 |
| 1 | 1 | 0 | 1 | 1 | 1 |
| 1 | 1 | 1 | 0 | 0 | 0 |

Equations 1a-1c formed the basis for the baseline design, shown in schematic form in Figure 3a below. The inputs to the 74151 multiplexor are shown for each combination of inputs in Table 5 below. The logic for each individual input is derived below in Equations 2a-2h. This design was simulated on the Altera Quartus Waveform Editor. For the optimized design, the D flip-flop-based counter was replaced with a 74193 4-bit synchronous counter package. The most significant bit was ignored to effectively result in a 3-bit synchronous counter. Figure 3b shows the schematic for the optimized design. The combinational logic for A and B was encapsulated into a 74157 package containing four 2-to-1 multiplexors.

**(c) D0** = Q0’

**(b) D1 =** Q1 ⊕ Q0

*Equations 1a-1c: Next-state equations for the data values of each flip-flop.*

*Figures 2a-2c: The Karnaugh maps for D2, D1, and D0 respectively. Each is derived from Table 4.*

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **A** | **B** | **Q0** | **Q1** | **Q2** | **Q3** | **Q4** | **Q5** | **Q6** | **Q7** |
| 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 0 |
| 1 | 0 | 0 | 1 | 0 | 1 | 0 | 1 | 1 | 1 |
| 1 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

**(f) Q5** = A+B

**(e) Q4** = AB

**(d) Q3** = A+B

**(c) Q2** = AB

**(b) Q1** = 1

**(a) Q0** = 0

**(g) Q6** = A

**(h) Q7** = A

*Table 5: The required inputs to each input of the 74151 multiplexor based on the sequence in Table 2.*

*Equations 2a-2h: The logic equations for each input to the 74151 eight-to-one multiplexor.*

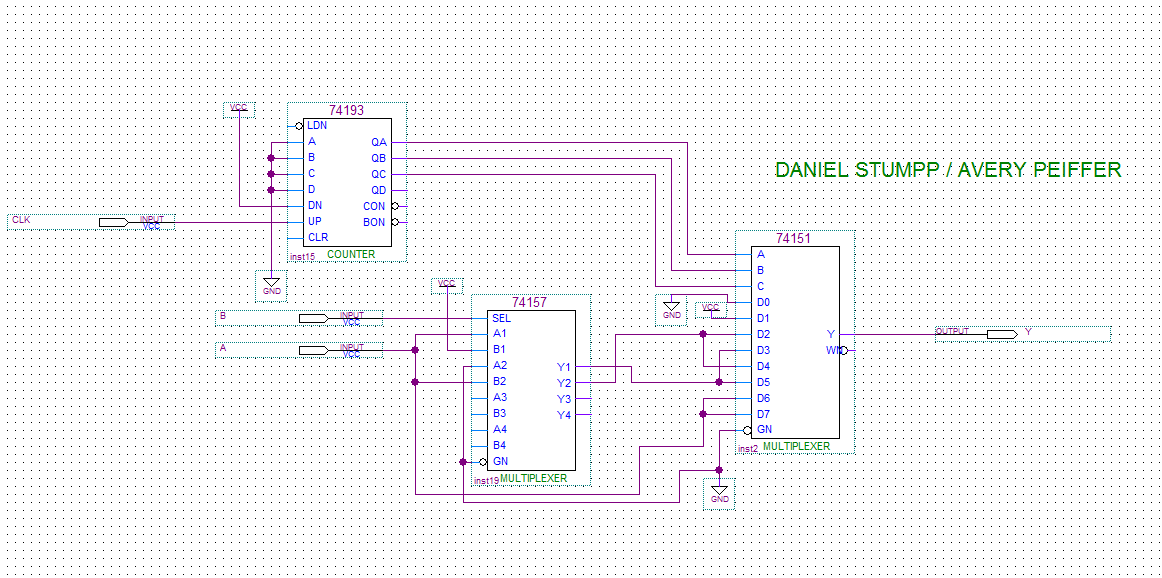


Figure 3a: The baseline design using two 74LS74A flip-flops, one 74151 eight-to-one multiplexor, and external combinational logic as necessary.

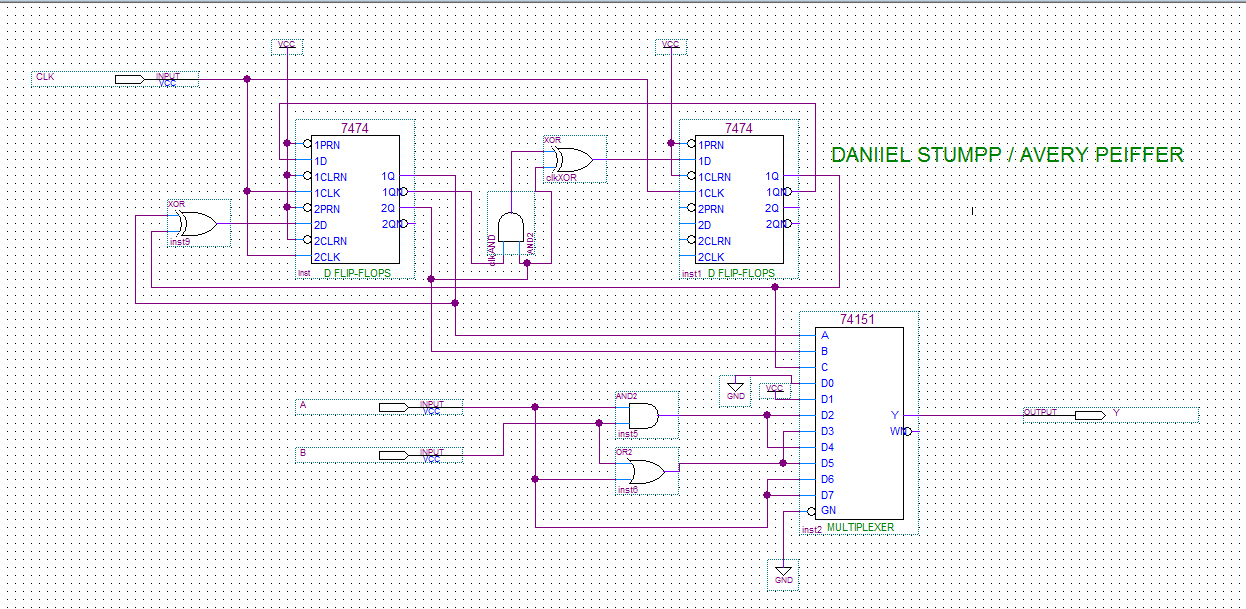
These two schematics were simulated in an Altera Quartus waveform file and generated the output waveforms shown in Figures 4a-4d below. There is a slight timing delay of 5 ns between the rising edge of the clock signal and the change to the output waveform, but the clock period of 100 ns is large enough that the delay does not affect the rest of the circuit.

Figure 3b: The optimized design using a 74193 4-bit counter, a 74157 package containing four 2-to-1 multiplexors, and a 74151 8-to-one multiplexor.

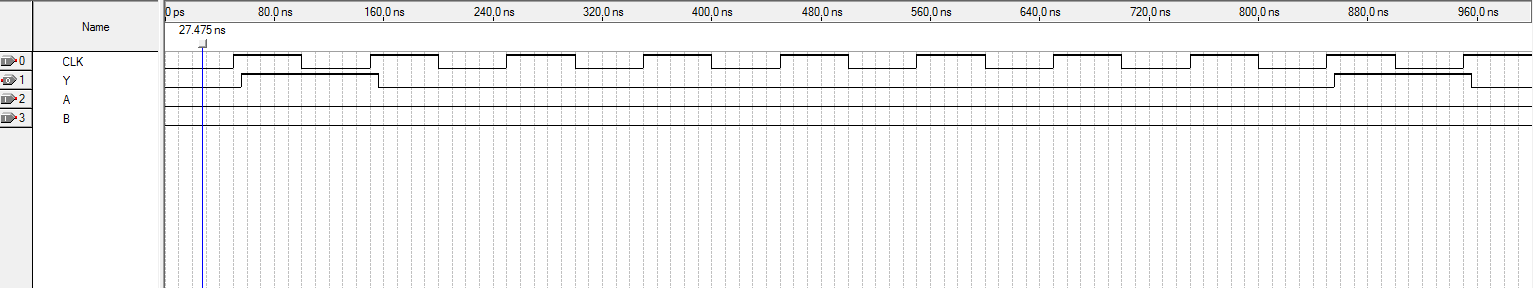


Figure 4a: The output waveform for AB = 00. Note that Y is high once over eight clock cycles.

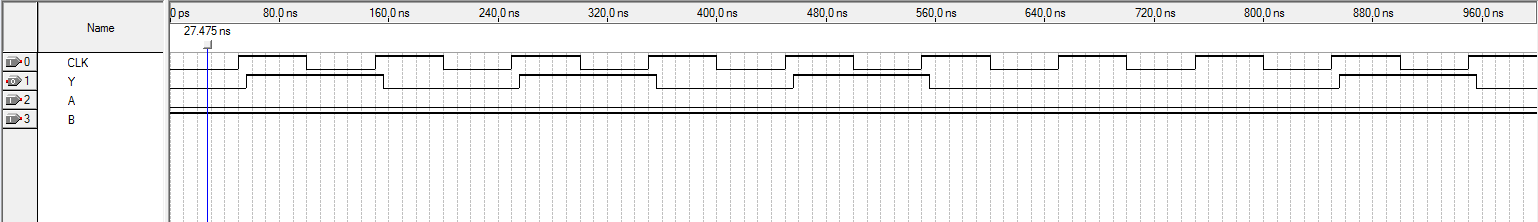


Figure 4b: The output waveform for AB = 01. Note that Y is high for three of the eight clock cycles.

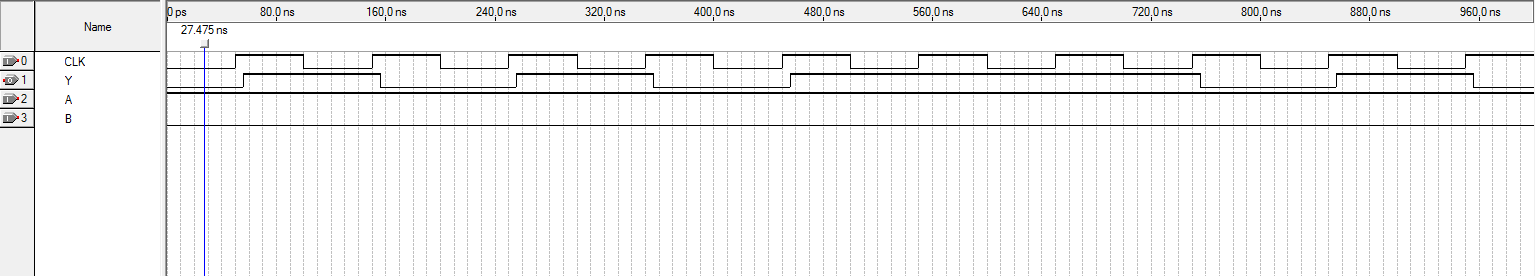


Figure 4c: The output waveform for AB = 10. Note that Y is high for five of the eight clock cycles.

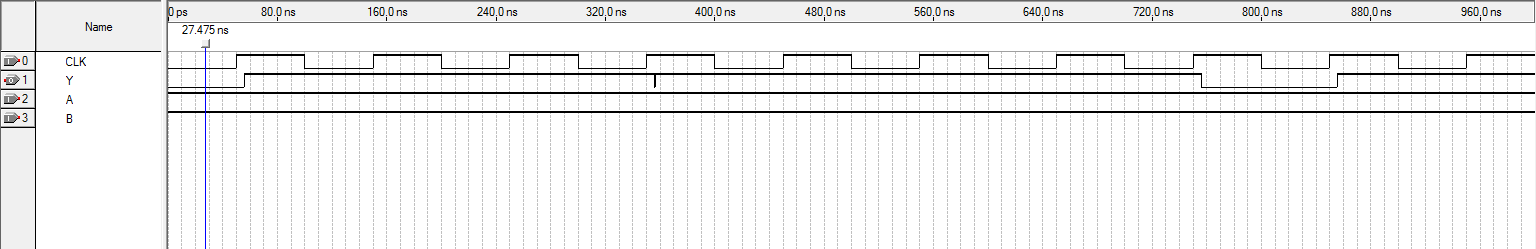
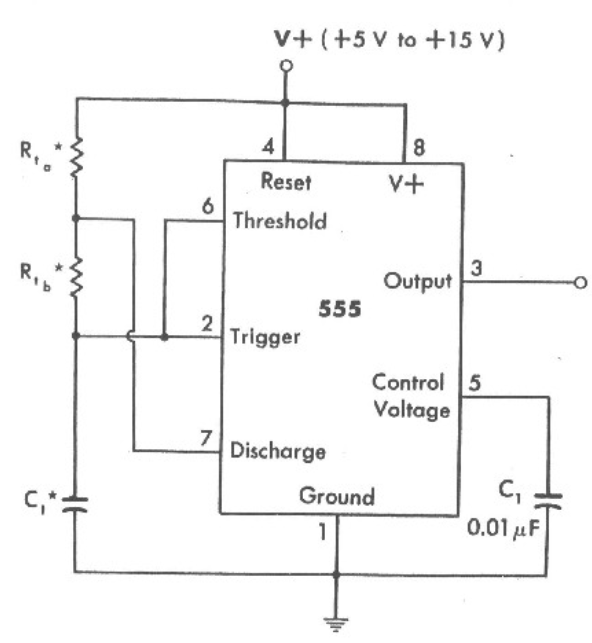


Figure 4d: The output waveform for AB = 11. Note that Y is high for seven of the eight clock cycles.

The physical circuit was designed using a 555-timer circuit to provide the clock signal. The 555-timer circuit is shown below in Figure 5 [1]. Equations 2a-2c were used to calculate the required capacitance C1 when two 1 kΩ resistors were selected to be Ra and Rb in the circuit.



(a)

(b)

(c)

t1 = 0.693(Ra + Rb)C1

t2 = 0.693RbCt

T = 0.693(Ra + 2Rb)C1

Equations 2a-2c: Equations to calculate the period of the 555-timer circuit in Figure 5 [1].

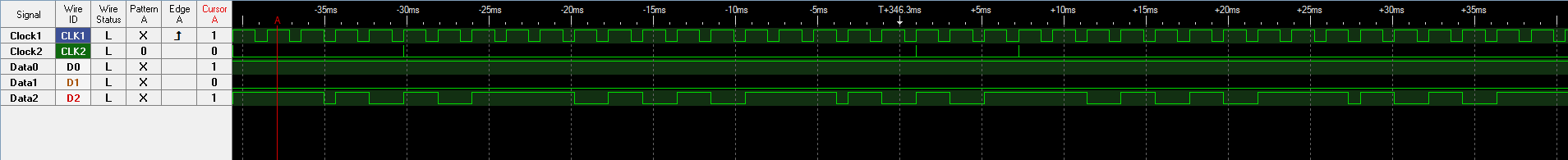
Figure 5: The 555-timer circuit schematic [1]. Ra and Rb were selected to each be 1 kΩ. For the circuit to have a period of 2 ms, C1 was calculated to be 1 μF.

The physical circuit was connected to the LogicPort Logic Analyzer through the connections shown in Table 5.

|  |  |
| --- | --- |
| **Physical Circuit Pin** | **Logic Analyzer Pin** |
| Input A | 1 (Data0) |
| Input B | 2 (Data1) |
| Output Z | 3 (Data2) |
| Clock | 20 |
| Ground | 9 |

Table 5: Connections between the physical circuit and the LogicPort Logic Analyzer.

When the optimized circuit design was simulated on the LogicPort Logic Analyzer software, an interesting error occurred. Figure 6 shows the output waveform for AB = 10 alongside the clock waveform. In this simulation, the output changes on the falling edge of the clock signal once every sixteen clock cycles. However, the output is supposed to be triggered only on the rising edge of the clock signal.



3

2

1

Figure 6: This output waveform for AB = 10 shows the output changing on the falling edge of the clock signal at points 1 and 3. It is supposed to only change on the rising edge of the clock signal, as seen at point 2.

Through testing, the source of this issue was revealed to be interference from the fourth bit of the 74193 counter. Previously, the fourth bit had been simply ignored, as it was assumed that the transitions of 0111-1000 and 1111-0000 would have identical results in the output waveform. To rectify this issue, the reset pin of the 74193 counter was tied to the Q3 output (pin 7). Doing this meant that the counter would reset itself when it transitioned from 0111 to 1000; therefore, the fourth bit was entirely ignored. The schematic for the 74193 package as well as the adjustments made to the physical circuit are shown below in Figures 7 and 8 [2]. A 1 kΩ resistor was connected between the counter’s reset pin and ground so that the connection between Pins 7 and 14 would cause the counter to reset only when Pin 7 went high.

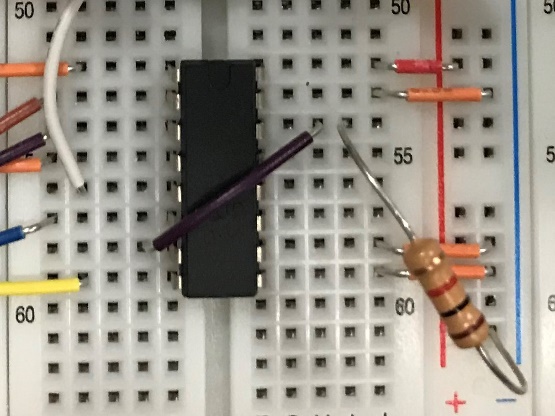
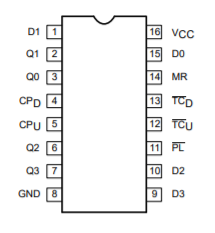


Figure 7: The schematic for the 74193 4-bit counter package.

Figure 8: The 74193 package with Pins 7 and 14 connected and a 1 kΩ resistor connected between Pin 14 and ground.

Figures 9a-9d show the LogicPort software output waveforms for each combination of inputs after this circuit adjustment was made. These waveforms are identical to those generated in the Altera Quartus software, though the period of the clock signal differs.

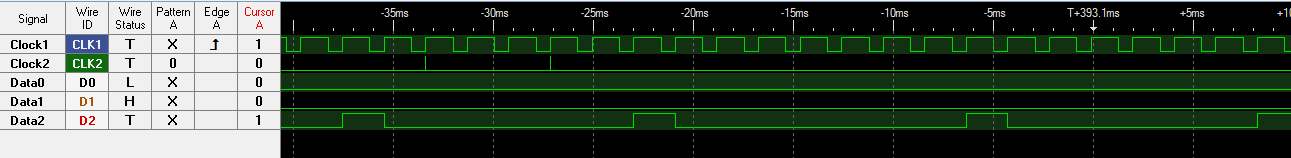


Figure 9a: Output waveform for AB = 00.

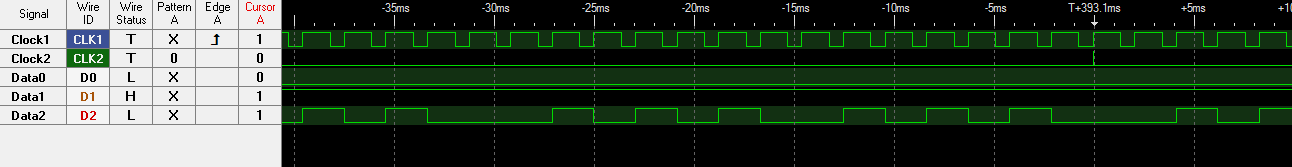


Figure 9b: Output waveform for AB = 01.

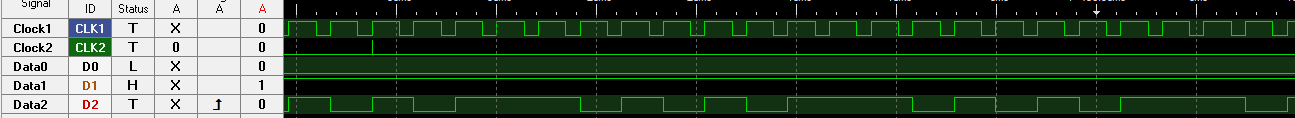


Figure 9c: Output waveform for AB = 10.

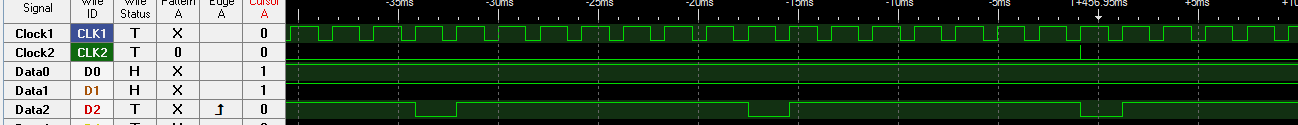


Figure 9d: Output waveform for AB = 11.

**CONCLUSIONS**

In this lab, a synchronous digital circuit was designed, simulated, and built. A finite state machine was designed and utilized as the counter for the baseline circuit design, but ultimately abandoned in favor of a dedicated four-bit synchronous counter for an optimized design. The four-bit counter did not function exactly as intended when initially implemented in the protoboard design; its unreliability likely means that more attention will have to be paid when using the 74193 package in the future.

The LogicPort Logic Analyzer was used to observe the outputs of the protoboard implementation of the circuit. The Logic Analyzer software proved to be valuable in the process of debugging circuits, as the issue with the four-bit counter was relatively easy to identify and test. Otherwise, the output waveforms matched the output sequences described in Table 2. Since the output bit came from an eight-to-one multiplexor, the multiplexor inputs could easily be rearranged so that the output sequences matched those in Table 1.

**REFERENCES**

[1] 555 Timer Circuit Data Sheet

[2] 74193 Data Sheet

[3] Dr. Alex Jones’ lecture notes and laboratory manual readings for Lab 10

[4] 74LS74A Data Sheet

[5] 74151 Data Sheet

[6] 74157 Data Sheet

[7] Partner Daniel Stumpp