Experiment 3
Simple Logic Processor
ECE 385

Jeffrey Huang JHUANG61, Jordan Tan JTAN45

February 7, 2017

Date Performed: February 7, 2017

1 Introduction

In this lab we designed and built a 4 bit logic processor. The processor is capable of 8 different functions. We implemented this using TTL chips with a variety of functionalities. The processing unit uses a finite state machine that is implemented in the control unit of the circuit.

2 Prelab

2.1 Question 1

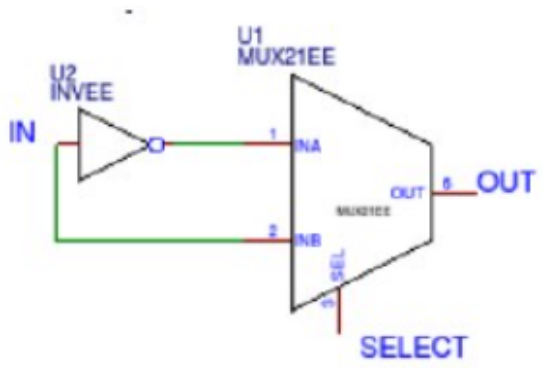


Figure 1: Prelab 1.

The simplest circuit that can optionally invert a signal is a multiplexer with the regular signal and inverted signals as inputs. The control bit determines whether the original signal or inverted signal is outputted.

2.2 Question 2

A modular design allows us to test our circuit easier. Since we know the inputs and outputs for each module, we can reduce the amount of variables that we need to test. For example, if we were to test the routing unit since we know that there are 2 inputs as the select bits. We can test each of the four unique inputs. Since we know the appropriate outputs to those inputs, we can verify that this module is working correctly. When there is an issue after hooking up the different modules, by testing it individually, we can come to the conclusion that the routing unit isn't the problem and we will be able to save time by testing other parts of the circuit first.

3 Operation of the Logic Processor

4 Written Description

4.1 Written Description

Register Unit The register unit consists of two 4 bit shift registers. It's connected to the external switches D_0, D_1, D_2, D_3. Since it's connected directly to the switches, it allows the register unit to be loaded directly from the switches. This unit stores data that the logic processor is operating on. This unit outputs to the computation unit and takes in the data from the routing unit.

Computation Unit The computation unit performs the actual operation on the data passed in from the register unit. The computation unit is capable of performing 8 operations on the data. Since it operates on 1 bit at a time there is a state machine implemented in hardware. This state machine maintains a state, Q, which maintains the state for 4 cycles of the clock. The state holding is implemented with a JK flip flop. The function command is controlled by external switches, F_0, F_1, F_2. The output is the input to the routing unit.

Routing Unit The routing unit controls what data is stored into which register. The output is directly connected to the register unit and depending on the configuration of the external switches, R_0, R_1, it will input the data into register A or B. Internally, it is implemented with 4 to 1 multiplexers with the external switches as the control bits.

Control Unit The control unit starts the operations. There are 3 inputs, $Load_A, Load_B, and Execute$. Based on logic inside the control unit, it either starts the computation with the register unit or allows the register unit to load

data from the external switches. When the execute switch is turned high then the state in the computation unit goes to the execute state. It maintains this state for 4 clock cycles.

4.2 Block Diagram

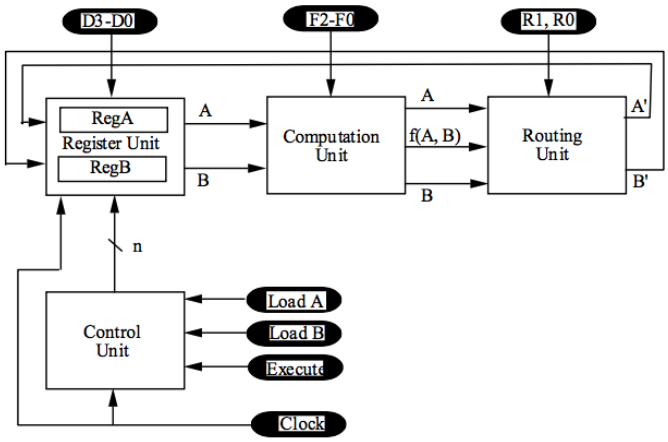


Figure 2: Block Diagram.

The high level block diagram for the circuit.

4.3 State Machine Diagram

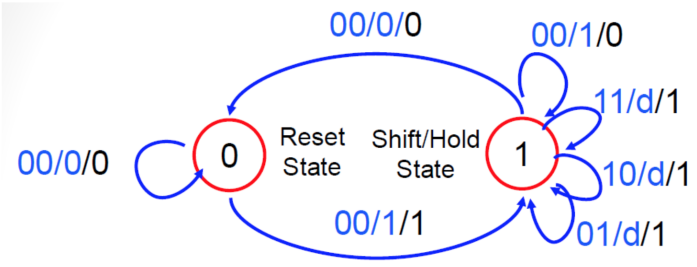


Figure 3: State Machine

The state machine for the computation unit. We implemented this using a Meely machine. The outputs are labeled as $[Counter_1\ Counter_0]/[Execute]/[Q]$

5 Design steps taken

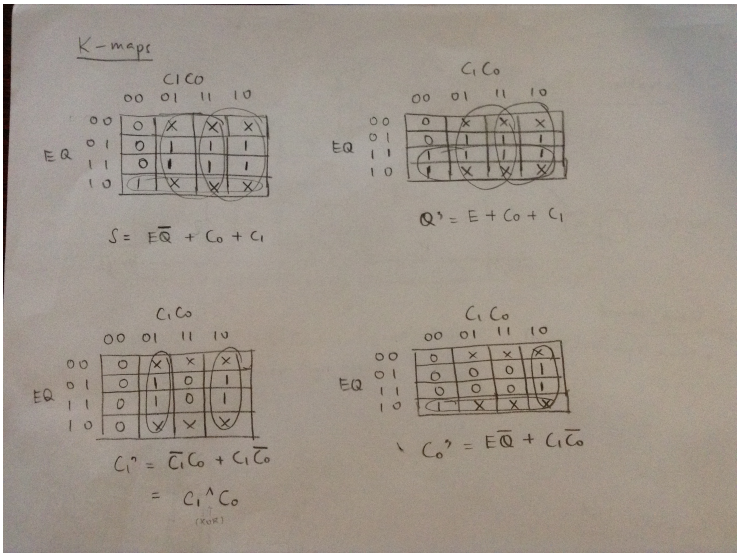


Figure 4: KMap for Control Unit

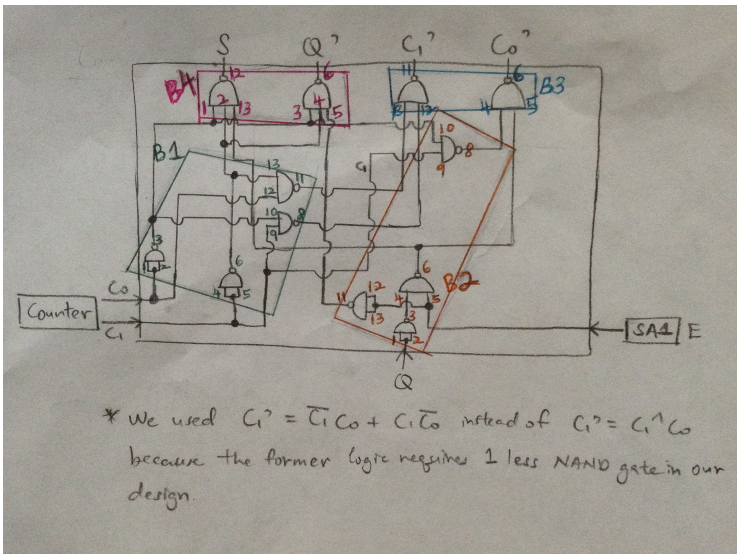


Figure 5: Control Unit Circuit Design

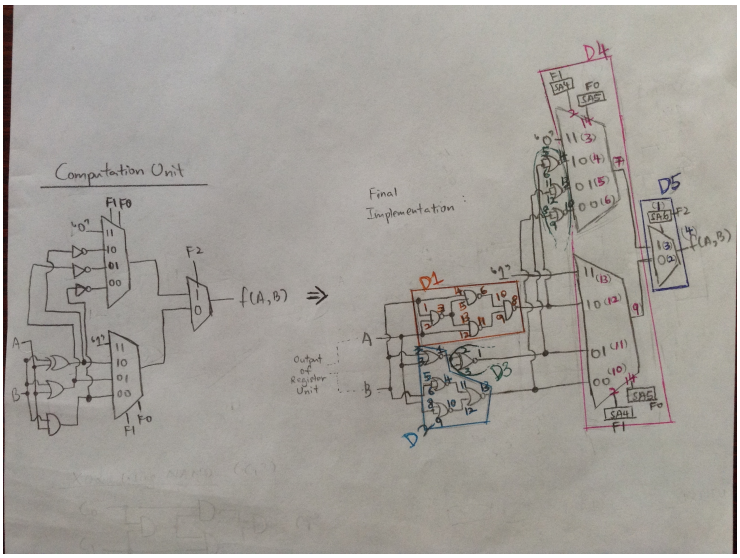


Figure 6: Computation Unit Diagram

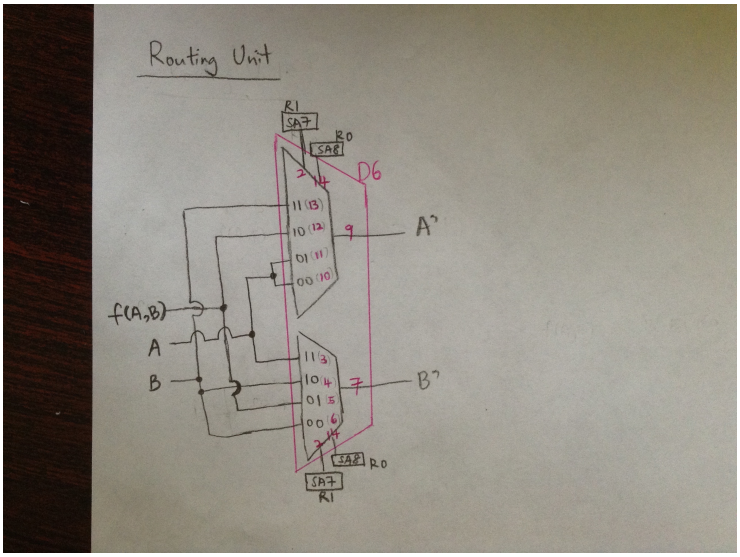


Figure 7: Routing Unit Diagram

Control Unit KMap

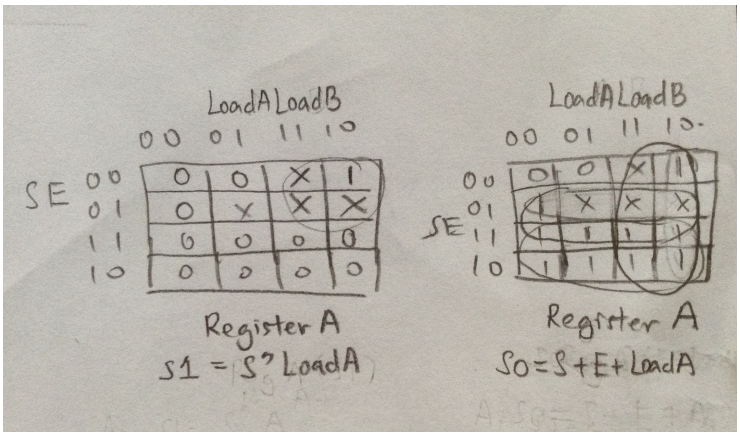


Figure 8: Register Unit Register A KMap

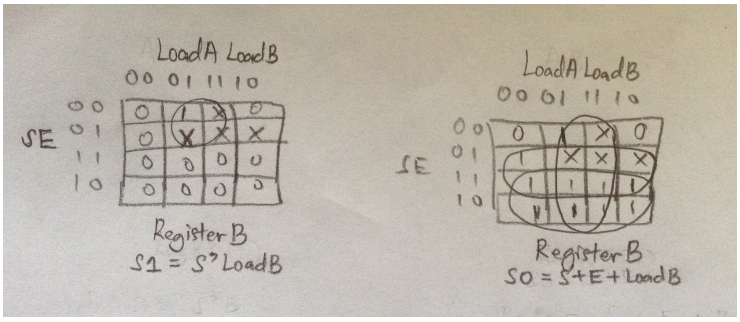


Figure 9: Register Unit Register B KMap

6 Layout sheet

7 Bugs

We ran into alot of bugs during this lab. Most of them involved not writing the TTL chip properly. This led to alot of confusion when we tried to wire everything together. There were parts of the circuit that worked individually, but after hooking all up together it no longer worked.
We also had issues with the switches in our lab box. The voltage when they were switched off was 1.1V. Because of this we weren't able to use it to a simulate a clock signal and forced us to wait until lab to debug our circuit.

8 Conclusion

We didn't have enough time to finish this lab. If we were to redo this we would have came into open lab sections to debug our circuit. The lab was straight forward, we just didn't have enough time to finish the lab.

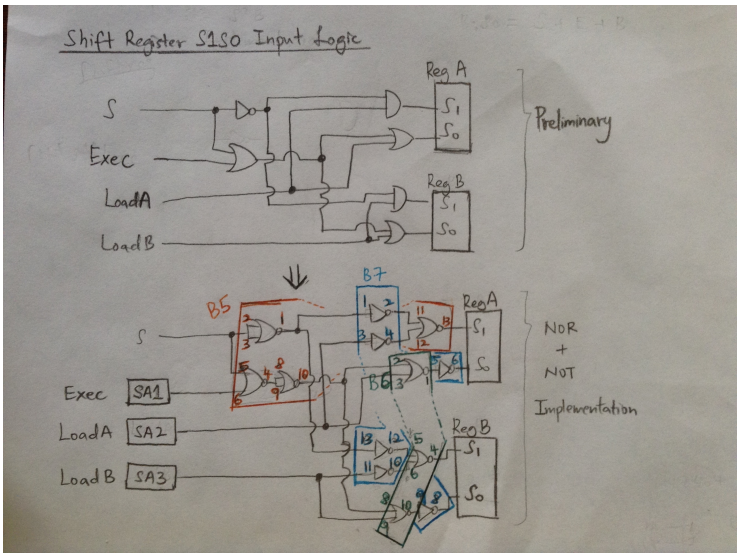
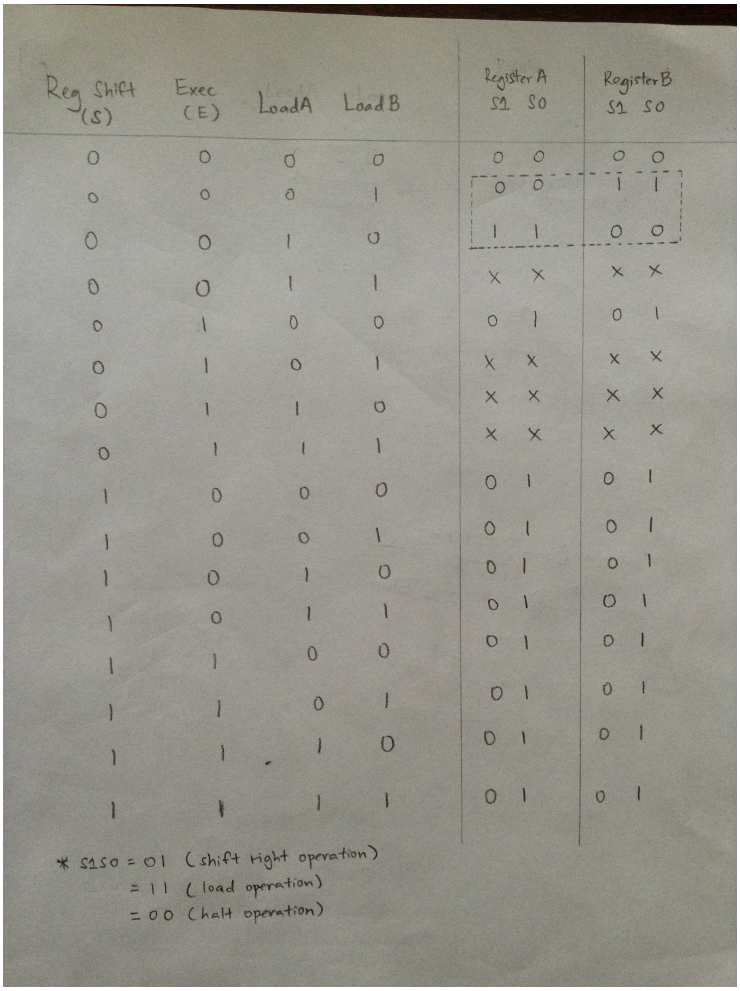


Figure 10: Register Unit Logic



Reg Shift (S)	Exec (E)	LoadA	LoadB	Register A S1	Register A S0	Register B S1	Register B S0
0	0	0	0	0	0	0	0
0	0	0	1	0	0	1	1
0	0	1	0	1	1	0	0
0	0	1	1	X	X	X	X
0	1	0	0	0	1	0	1
0	1	0	1	X	X	X	X
0	1	1	0	X	X	X	X
0	1	1	1	X	X	X	X
1	0	0	0	0	1	0	1
1	0	0	1	0	1	0	1
1	0	1	0	0	1	0	1
1	0	1	1	0	1	0	1
1	1	0	0	0	1	0	1
1	1	0	1	0	1	0	1
1	1	1	0	0	1	0	1
1	1	1	1	0	1	0	1

* S1S0 = 01 (shift right operation)
= 11 (load operation)
= 00 (halt operation)

Figure 11: Register Unit Truth Table

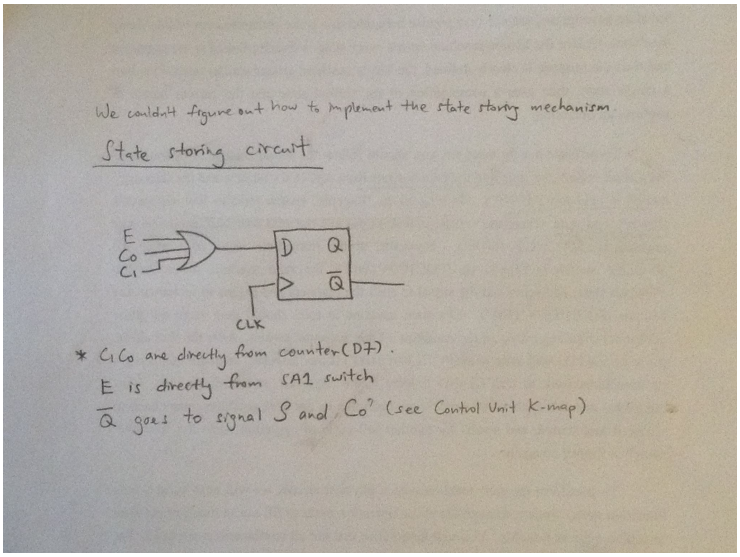


Figure 12: State Machine

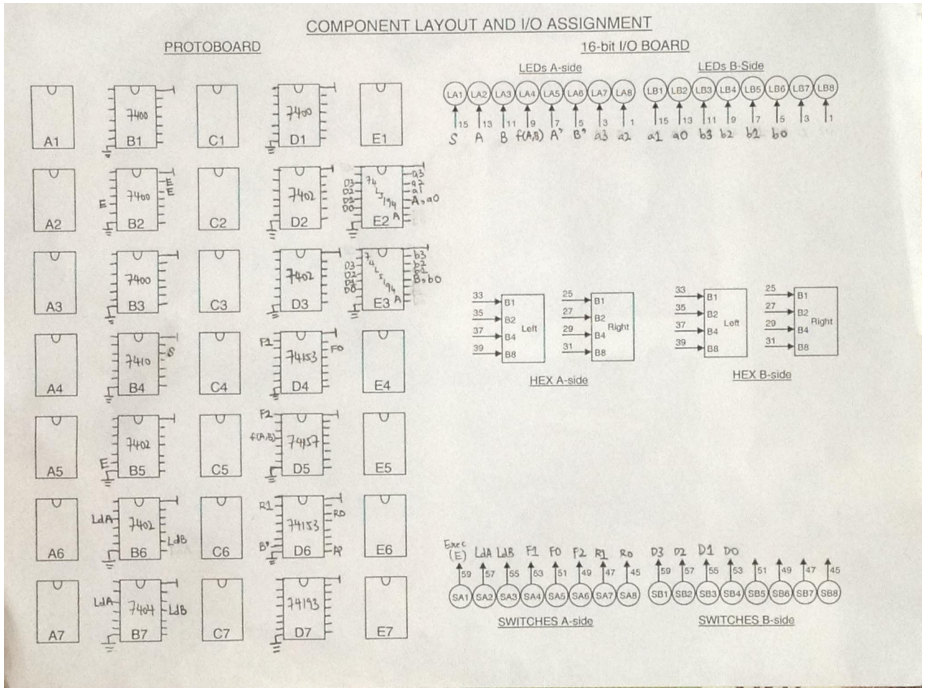


Figure 13: Component Sheet