ECE 385

Spring 2020

Experiment #2

Data Storage

Freddy Zhang and Mehul Dugar

Section: ABO (Thu. 3PM)

Gene Shiue

Introduction

The circuit that was created was a storage device that stores bits to memory. The circuit can also retrieve the data stored in memory and load certain bits to the memory. The memory can hold 4 words with 2 bits in each word.

Operation of the memory circuit

Addressing is implemented using the SAR switches, SAR1 and SAR0. This allows the control unit to know which registers the control unit will be interacting with. From there, the circuit will conduct an operation based on which operation switches (LOAD, FETCH, STORE) are switched high. Once the operation is finished, the output will be stored in the SBR flip-flops.

A write operation first starts with determining which registers will be written on using the SAR switches. Then, the user will use the D-INPUT switches to determine the two bits that will be stored in the registers. After that, the user will need to flip the LOAD switch to high to load the corresponding D-INPUT switches to the SBR. Once the user flips the STORE switch to high, the values in the SBR will transfer into the shift registers. Thus, successfully writing a word into the shift registers. This works because having LOAD set to high will hold the values of the D-INPUT switches in the SBR, and while the clock is running, a counter will be constantly counting up from zero to three and the shift registers will be constantly shifting which means each shift register has its own unique address based on the counter. When the STORE switch is high and the counter and SAR values are the same, the comparator will set the select bit of the 2:1 MUX to one which makes the shift registers store the bits that were in the SBR. In this case, the bits in the SBR correspond to the D-INPUT switches since LOAD was set to high.

To perform a read operation, the user must make sure that the load switch is flipped to low for the operation to be done correctly. The user will first flip the SAR switches to their desired address they want to read. Then, flip the FETCH switch to high. This will notify the control unit to send the bits from the corresponding SAR to the SBR and LEDs. The clock helps the data flow in the circuit similarly to the write operation where the counter uses the clock cycle to count from zero to three to define the four addresses that the shift registers have.

Memory Circuit Implementation

The shift registers were used to as memory. The bits would be stored in the shift registers and the bits would shift right each rising edge of the clock. The bits would circulate back to the beginning of the shift register unless the STORE switch was set to high. Determining whether or not new bits would be stored is based on the 2:1 MUX with the select bit. The select bit of the 2:1 MUX is based on the control unit. A more in-depth explanation of the control unit is written in the next section of the report. The bits in the D flip flops depend on the 3:1 MUX. The select

bit is determined by the control unit, the LOAD switch, and the FETCH switch. Based on the switches, the 3:1 MUX will either output the bits from the respective SAR address, the D input switches, or circulate the bits from the flip flop into the flip flop.

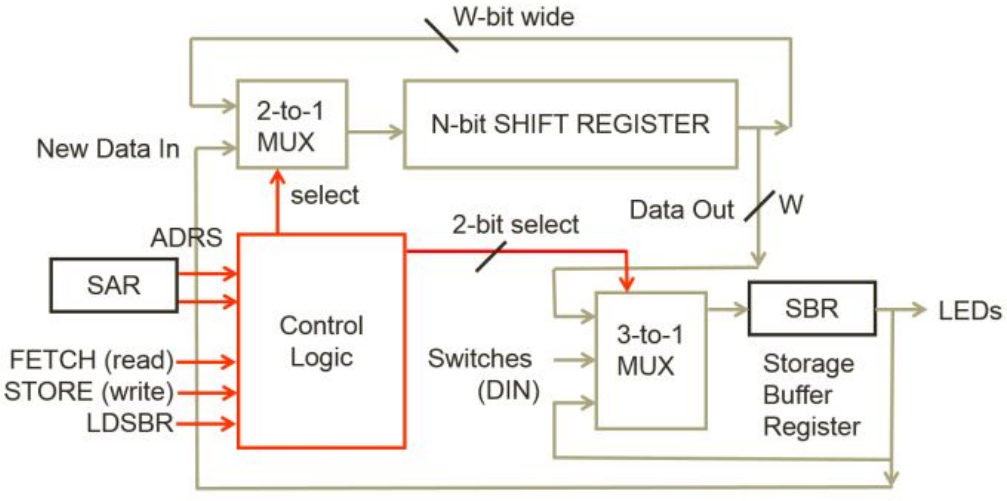
Control Unit

The control unit consists of a counter, a comparator, and two AND gates. The counter counts from 00 to 11 and gets incremented each rising edge of the clock. The comparator has the clock in the A pins and the SAR switches in the B pins. The comparator only has the A=B output connected to make sure that the counter and the SAR switches have the same value. The output of the comparator is connected to two AND gates that are ANDed with FETCH and STORE. The output of those AND gates are connected to the 3:1 MUX and 2:1 MUX respectively. This makes sure that the STORE and FETCH operation will be performed to the correct register.

Design Steps

The lab as a whole was straight forward enough that the chips used in this lab were the obvious choice or the professor told us to use the certain chip. The main variation came from the counter. The two choices were the ripple counter and the synchronous counter. We chose the synchronous counter to prevent glitches from occurring to reduce the probability of the circuit being faulty. Other variations in the circuit were based on personal preferences. For example, the select bits for the MUXs could have been different if we wanted to define the MUXs a different way.

Detailed Circuit Schematic

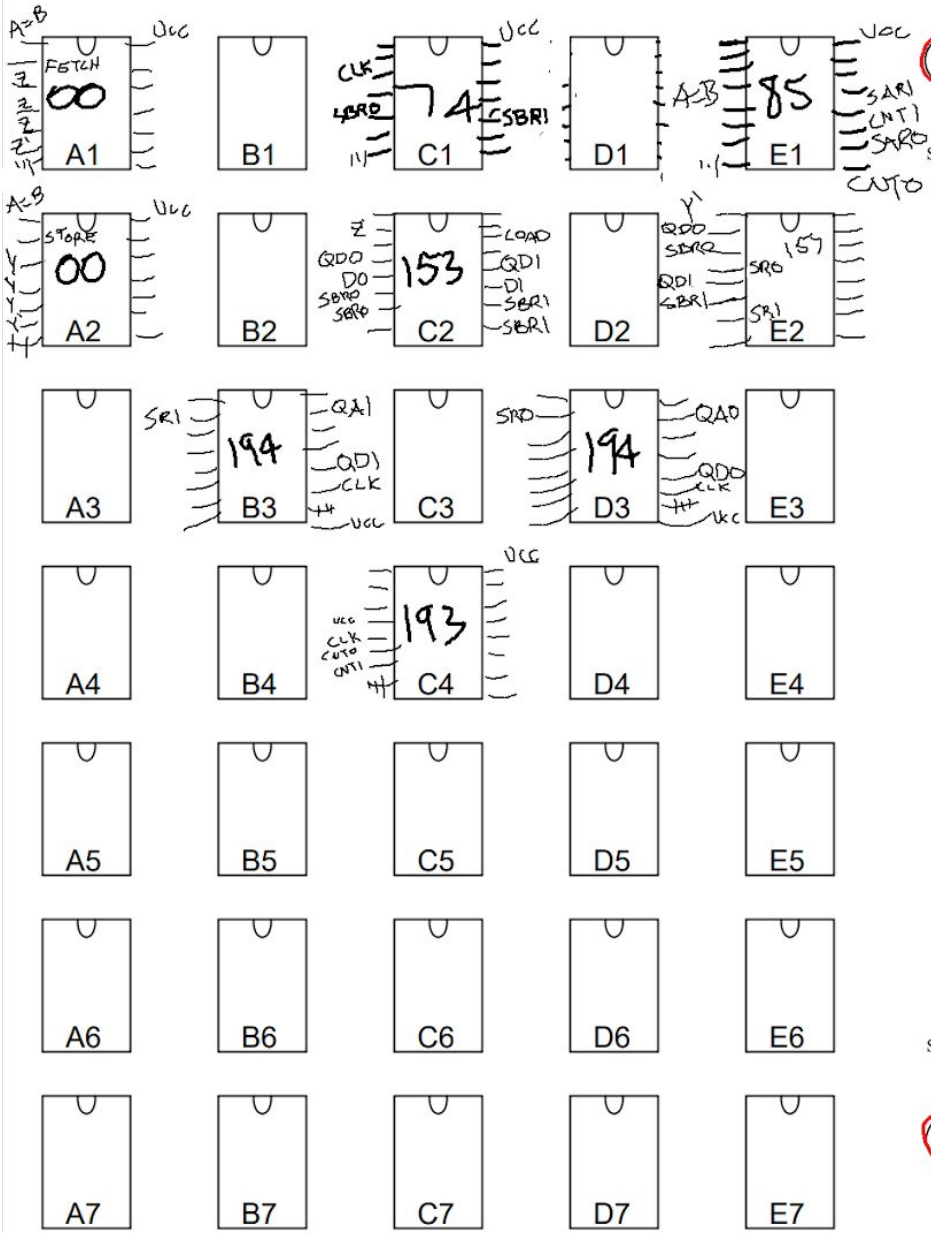


IC1
74LS192N
IC2
74LS85N
V3
74153N
MUX
DSBR
DIN1
DIN0
GND
Vcc
CLK
SAR0
SAR1
FETCH
STORE
AND
X2
V1
EN
G1
74157N
V2
V4
74194N
SRG4
V5/2
V5/1
74174N

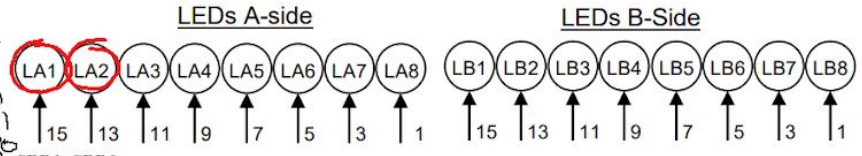
Component Layout Sheet

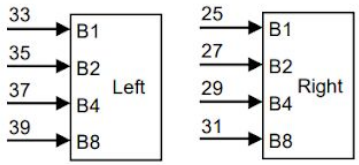
COMPONENT LAYOUT AND I/O ASSIGNMENT

PROTOBOARD

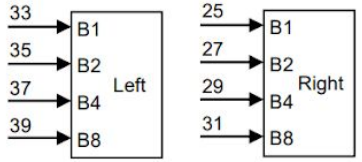


16-bit I/O BOARD

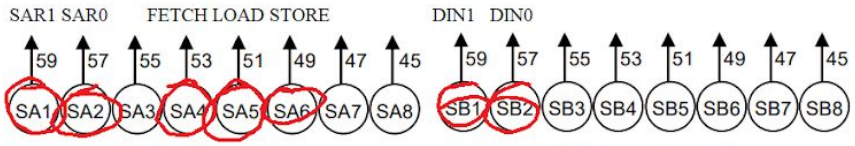




HEX A-side



HEX B-side



SWITCHES A-side

SWITCHES B-side

Bugs Encountered

Most bugs that were encountered were from using the incorrect pins. For example, at the beginning of building the circuit, we did not have a full understanding of how to use the counter chip so the counter did not count up from zero to three correctly. Furthermore, forgetting to ground the strobe was another error we made, it made the MUX unfunctional. The majority of the bugs came from accidentally connecting between wrong pins of the chips in use. We fixed these issues by retracing through our circuit diagram and finding our mismatched connections. We also went over our logic to find points of failures and thus revised it to overcome these points of failures.

Answers to Post-Lab Questions

a. The registers must be shifted on each clock pulse. The clock must run continuously – do not gate the clock (this is bad practice in digital design, why?)

   Clock gating is bad practice because gating the clock can cause gate delays. This leads to synchronization issues which can in-turn lead to unexpected behaviour of the circuit.

b. Only the clock input needs to be debounced to strep through your circuit (why?).

   We need a debounced switch for the clock because the regular switches we use are not perfect, as in, when we place the switch into the new position the contacts “bounce” between the nodes and we need to get rid of it hence the name “debounced”. We need to get rid of it because the bouncing causes unwanted triggering of the circuit and causes unexpected things to occur in the circuit each time it is shifted to a new position. Fortunately for us, the switches used in the switch boxes are all debounced and hence we can use them without any modifications. Additionally, we are also provided with a circuit (in the general guide) for making our own debounced switches to debug our circuit at home. We only need one for the clock because it is used to run other clock dependent components such as the flip-flops or the shift registers and they depend on the rising edge of the clock which if not debounced could cause multiple triggers of these components.

c. What are the performance implications of your shift register memory as compared to a standard SRAM of the same size?

   The shift register helps us break down the complexity by allowing bitwise operations rather than operating on the entire words. But this has implications on the run-time for each operation as we will need to run multiple clock cycles to get the desired memory

with this approach. In this scenario, the use of SRAM of the same size would allow the fetch to be run much faster (in a single clock cycle to be exact) as it fetches memory in a single clock cycle.

d. What are the implications of the different counters and shift register chips, what was your reasoning in choosing the parts you did?

   We had the option of using one of two different kinds of counters, synchronous and ripple. We chose to use the synchronous counter because it was more reliable and we would not have to worry about glitches in that portion of the circuit. The 74194 shift register chips were the chips that were recommended to us in the lecture.

Conclusion

This lab was intended to take us into depth of circuit design by tasking us to build a 2-bit four word register. We also got familiar with usage of MUX's, shift registers and flip-flops in memory storage. We felt the need for this class arises from the fact that RAM is an essential part of computer architecture and the goal of this lab was to build a primitive version of the same. Besides this, the freedom of choice when it came to designing the actual circuit gave us a better understanding of alternative approaches and come up with one that fit our needs.