ECE 385
Lab 2
Data Storage

Li, Ziying
Kang, Xingjian

January 31, 2024

1 Introduction

This is a memory circuit based on the 74194 shift register. It can contain 4 words of 2-bit using two parallel shift registers. It allows us to control its read and write so that we can write the value into SBR then into shift registers and also read from them. It can store the data for a long time as long as power is supplied.

2 Operation of the memory circuit

2.1 how the addressing is implemented

The whole system is controlled by one unified clock so that the actions in all chips are synchronized. The address is outputted by the last 2 bits of a 4-bit counter. When the request of read or write is inputted by the switch, the control logic will wait the address of register to cycle until it is the same as SAR. So read or write is committed.

2.2 how a write operation is performed on the memory

First, input the data using DIN. Turn on LDSBR to write DIN into SBR, then turn off LDSBR. Then change the SAR to the address we want to store the data in and turn on STORE. After the address in the address counter cycles to be the same as SAR, the data in SBR will be written into shift registers. Turn off STORE.

2.3 how a read operation is performed on the memory

First, change the SAR to the address we want to fetch the data and turn on FETCH. After the address in the address counter cycles to be the same as SAR, the data in shift registers will be written into SBR. Turn off FETCH.

3 Written description and block diagram of memory circuit implementation

3.1 High-level description

First, we need switches or inputs signals for DIN0, DIN1, SAR0, SAR1, LDSBR, STORE, FETCH, a stable and uniform clock signal. Then we use 7400 NAND gates to build control logic. 74153 multiplexer. We use 7474 D flip-flop to store SBR, 74194 shift register to build the main memory. We use 74193 to count addresses and 7485 comparator to match the address in SAR with counter.

3.2 High-level block diagram

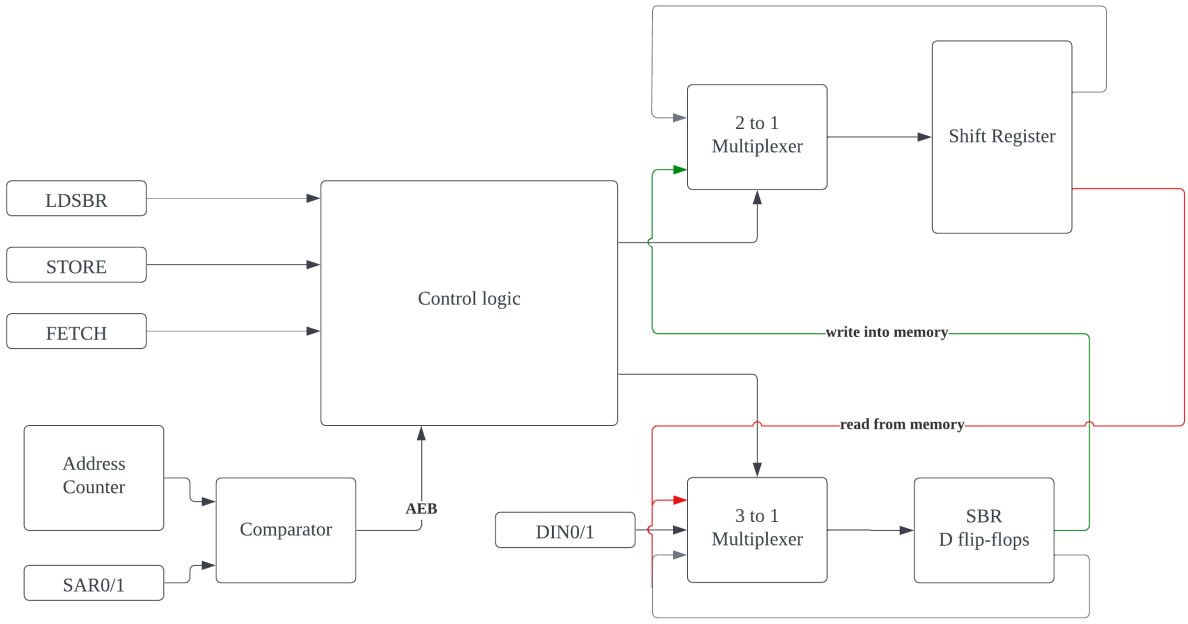

Figure 1: High-level block diagram

4 Control Unit

4.1 An intuitive written description of control unit

The control unit has three main logics, a comparator module, an SBR storage related module and a memory storage related module.

The inputs accepted by the comparator module are the two bits of SAR and the current value of COUNTER. When SAR = COUNTER, the AEB0 port of the COMPARATOR outputs a signal and this signal is used as one of the logics for each of the SBR storage section and the memory storage section.

For the SBR storage related module (the instructions involved are FETCH and LOAD), the other two inputs are the FETCH signal and the LDSBR signal and cannot both be 1. When FETCH = 1 and AEB0 = 1, the control unit will pass a signal specified to be FETCH to 3-to-1 MUX, and the last bit of each of two shift registers will be read into SBR. Similarly, when LDSBR = 1, the control unit will pass a signal specified to be LOAD to 3-to-1 MUX, and the 2-bit data in DIN will be loaded into SBR.

For the memory storage related module (the instructions involved is STORE), another input is STORE signal. When STORE = 1 and AEB0 = 1, the control unit will pass a signal specified to be WRITE to 2-to-1 MUX, and the data in SBR will be loaded into shift registers.

The logic expressions for selection bits are as follows:

$$X0 = \overline{\overline{AEB * STORE}}$$

$$Y0 = \overline{\overline{\overline{FETCF} * LDSBR}}$$

$$Y1 = \overline{\overline{\overline{\overline{FETCH} * AEB} * \overline{LDSBR}}}$$

4.2 A block diagram of control unit

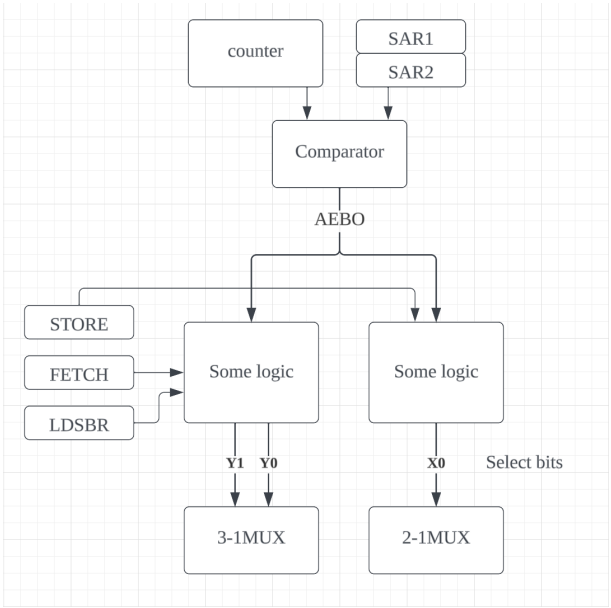

Figure 2: Block diagram for the control unit

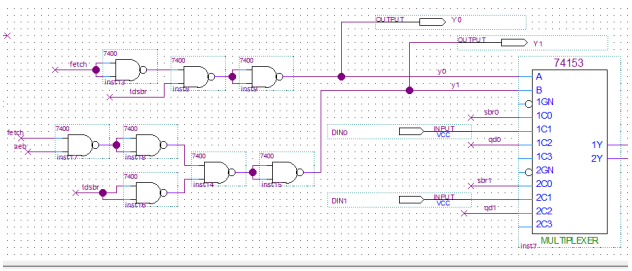

Figure 3: Enter Caption

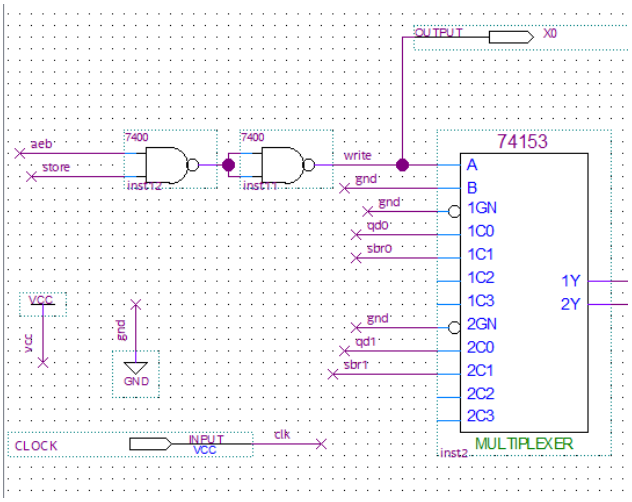

Figure 4: Enter Caption

5 Design steps taken and detailed circuit schematic

5.1 Truth table

The truth tables for 3-to-1 MUX and 2-to-1 MUX are as below:

2-1 MUX

AEBO	STORE	Selection signal X0	Function
0	0	0	NOP
1	0	0	NOP
0	1	0	NOP
1	1	1	STORE

Figure 5: Truth table for 2-to-1 MUX

3-1 MUX

FETCH	LDSBR	AEBO	Selection signal Y1	Selection signal Y0	Function
0	0	0	0	0	NOP
0	1	0	0	1	LOAD
1	0	0	0	0	NOP
0	0	1	0	0	NOP
1	0	1	1	0	FETCH
0	1	1	0	1	LOAD

Figure 6: Truth table for 3-to-1 MUX

5.2 Written description of the design considerations taken

While designing the control circuit, we took into account the decoupling of the various functional modules to minimize dependencies between different modules of the whole system, and this separation helps create a more flexible, maintainable circuit design.

In total, we need to implement the logical functions of three instructions, two of which are related to the storage of SBR and one of which is related to the memory storage, so we have divided these three instructions into two modules, of which STORE is in one module and FETCH and LOAD are in another module.

5.3 Detailed Circuit Schematic

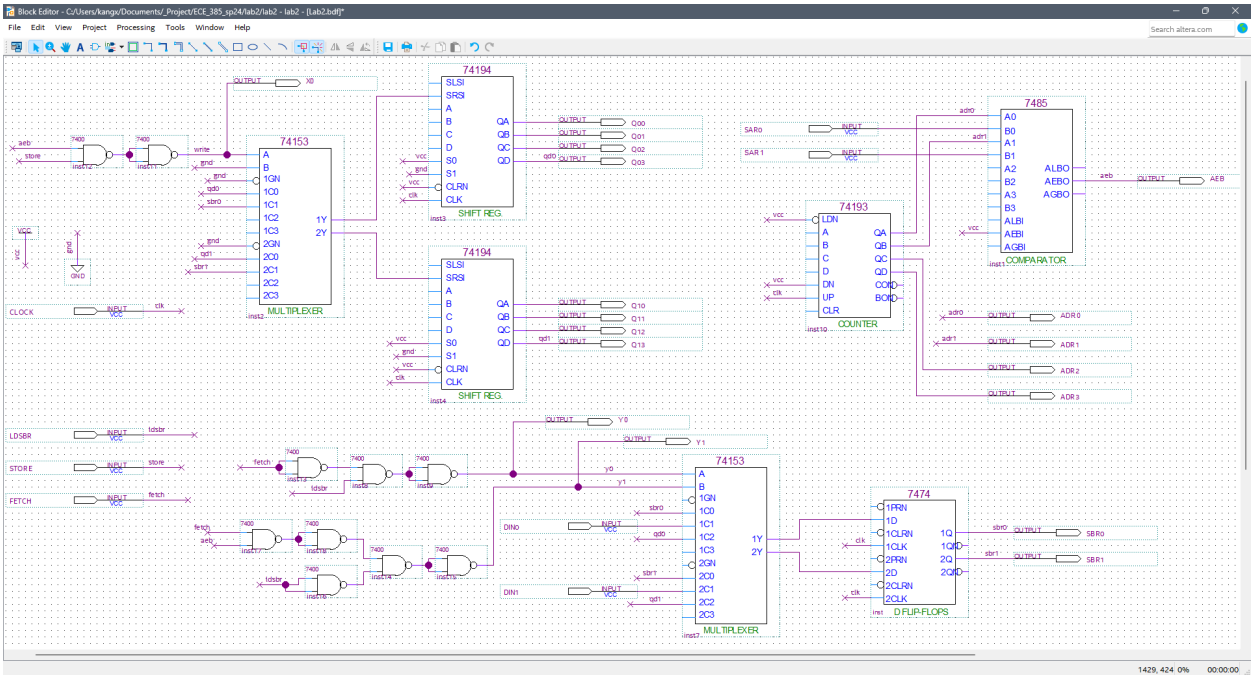

Figure 7: Circuit diagram in Quartus

6 Description of all bugs encountered, and corrective measures taken

1. address output behave wired. Wrong order in connecting cables.
2. The value in SBR D flip-flop is wrong. Connect wrong cables on the multiplexer before the D flip-flop.

We added many outputs to monitor important nodes so that we can have accurate judgements on what is wrong.

7 Answers to questions in pre-lab and post-lab

7.1 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?)

To preserve the synchronization and integrity of the data being shifted, we should avoid gating the clock when shifting registers. The risk of the clock signal being interrupted or delayed at specific times might be introduced when we gate the clock, otherwise we may read dirty/incorrect data.

7.2 Only the clock input needs to be de-bounced in order toto strep through your circuit (why?)

Since all other components are under the clock's control, just the clock needs to be de-bounced. The de-bounced clock input may be able to stabilize the output signal if their signals are not steady.

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

SRAM has faster access times compared to shift registers, since SRAM is designed to be able to access a specified memory location directly. Shift register is slower because it may have to wait for a number of cycles to shift the required data to right position.

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

For counters: We chose 74193 (4-bit Synchronous Up-Down Counter) for counter. We didn't use 7493 (4-bit Ripple Counter) since each D flip-flop in it does not have synchronous clock (i.e., rising edge) and not all bits are updated at the same time, which may lead to wrong outcomes.

For shift register chips: We chose 74194 (4-bit Bidirectional Universal Shift Register) for shift resgisters. We didn't choose 7495 (4-bit Parallel-Access Shift Register) since it does not have a dedicated parallel load input, so synchronous parallel loading cannot be guaranteed.

8 Conclusion

We build a 2-bit four-word shift-register storage unit without using the parallel load or parallel output capability. It has 4 functions: STORE, LOAD, FETCH, and NOP. We build this with Quartus Prime, and components we use are 74194 (shift register), 7474 (D flip-flop), 74193 (counter), 74153 (multiplexor), and 7485 (comparator).

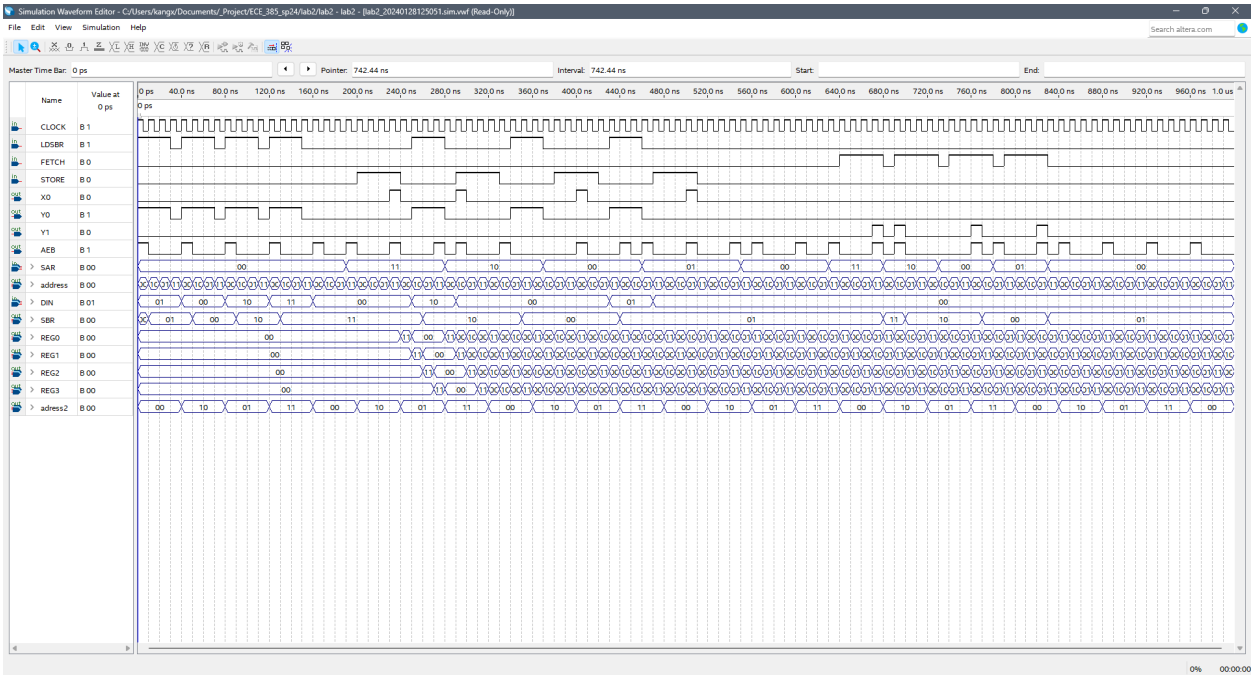

Figure 8: Waveform simulation results