**ECE 385**

**2021 FA**

**LAB 2**

**Data Storage**

Zhou Qishen, Xie Mu

**Introduction**

In this lab, we designed a 2-bit,4-words memory, by using multiplexer, register, comparator, counter, DFLIP-PLOP and other basic component, which can realize operations of STORE, FETCH and IDOL.

**Operation Description**

**Addressing Space**

Because the register we designed is a 2 bit register, so the address list would be:

|  |  |
| --- | --- |
| Word | Address |
| 0 | 00 |
| 1 | 01 |
| 2 | 10 |
| 3 | 11 |

**Operation: STORE**

When data coming in, it will be stored in 2-to-1 MULTIPLEXER. First, compare the value of COUNTER and SAR. If matched, the COMPARATOR output a 1, to AND with STORE input. Then, load the data in the 2-to-1 MULTIPLEXER into SHIFT REGISTER.

**Operation: FETCH**

First, compare the value of COUNTER and SAR. If matched, the COMPARATOR output a 1, to AND with FETCH input. Then, load the data into the 3-to-1 MULTIPLEXER from SHIFT REGISTER, then into SBR. After that we could read the data or replace the data with new data.

**Description of circuit**

**High Level Description**

For this memory circuit, we designed 6 modules, CONTROL UNIT, 2 MUXs, SAR, SBR, SHIFT REGISTER. We use 74194 chip, which is 4-to-1 Multiplexer, for the 2 multiplexers. And SAR and SBR is both 74169 D FLIP-FLOP, for the temporary storage of data. For SHIFT REGISTER, we decide to use 74194 Shift Register, which is 1-bit each.

Diagram

Description automatically generated

**Control Unit**

Diagram

Description automatically generated

The control logic is quite simple. We drive counter by a CLOCK signal, which will make it loop from 00 to 11. Then send the counter value to compare with SAR value in the COMARATOR. If the value is matched, FETCH and STORE command will become valid.

**Design Steps**

Diagram, schematic

Description automatically generated

Diagram

Description automatically generated

Control Unit

Diagram, schematic

Description automatically generated

SBR

Diagram, schematic

Description automatically generated

SAR

Background pattern

Description automatically generated

MUXs

Diagram, schematic

Description automatically generated

Shift Reg

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | 00 | 01 | 10 | 11 |
| 0 | 0 | 1 | 0 | 1 |
| 1 | 0 | 1 | 0 | 1 |

K map 4-to-1 MUX A (LDSBR)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| FETCH\Cp | 00 | 01 | 10 | 11 |
| 0 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 1 | 1 |

K map 4-to-1 MUX B (FETCH & Cp)

|  |  |  |
| --- | --- | --- |
| STORE\Cp. | 0 | 1 |
| 0 | 0 | 0 |
| 1 | 0 | 1 |

K map SEL: STORE and Cp

**Component Layout Sheet**

Diagram

Description automatically generated

**Post-Lab Question**

**Only the clock input needs to be de-bounced to strep through your circuit (why?).**

Because we should ensure that the circuit is work in the simultaneously.

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

For SRAM, we could access to every address directly. However, we need a lot of work to match the address. But of shift register memory, we only need to wait the correct address pop up. This feature allow us to make shift register memory to a larger scale.

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

In this lab, using which chip does not really matter. We find several available chip and compared the function. Then we choose the simplest chip to design the circuit.

**Conclusion**

In this lab, we learnt how to select chips, pros and cons for shift register, and how to design a memory circuit based on the shift register.