****

**Faculty of Engineering and Technology**

**Department of Electrical and Computer Engineering**

**ENCS211 – Digital and Computer Organization Lab**

**Exp. #7 Constructing Memory Circuits Using Flip Flops**

**Instructor: Dr. Ahmad AlSadeh**

**TA: ENG. Yousef Anini**

**Submitted by: Laila Daher**

**Student ID# 1170056**

**Partner: Dania Abu-Jazar**

**Summer 2019**

**Abstract**

The main aim of this experiment is to understand the structure and functions of RAM – Random Access Memory, and to understand and test the circuit of 64-bit Random Access Memory.

**Table of Contents**

1. Introduction (4)
2. Theory (5)
   1. Constructing Random Access Memory (RAM) with D Flip Flop (5)
   2. 64-Bit Random Access Memory (RAM) Circuit (6)
3. Procedure (7)
4. Conclusion (10)
5. Feedback (11)
6. References (12)
7. **Introduction**

Random-access memory (RAM) is a form of [computer data storage](https://en.wikipedia.org/wiki/Computer_data_storage) that can be read and changed in any order, typically used to store working [data](https://en.wikipedia.org/wiki/Data) and [machine code](https://en.wikipedia.org/wiki/Machine_code). A [random-access](https://en.wikipedia.org/wiki/Random_access) memory device allows [data](https://en.wikipedia.org/wiki/Data) items to be [read](https://en.wikipedia.org/wiki/Read_(computer)) or written in almost the same amount of time irrespective of the physical location of data inside the memory. (1)

In this experiment we will learn the structure of the RAM using D flip flops and Understand and test the circuit of 64-bit Random Access Memory (RAM).

1. **Theory** 
   1. **Constructing Random Access Memory (RAM) wit D flip flop**

There are two different types of basic structure for RAM:

Figure (1) shows a RAM circuit in which the input and output are on the same port (they are not separated). It has two control ports, one is R/W (R: Read or OUTPUT and W: Write or INPUT). And the other one is the Enable port.

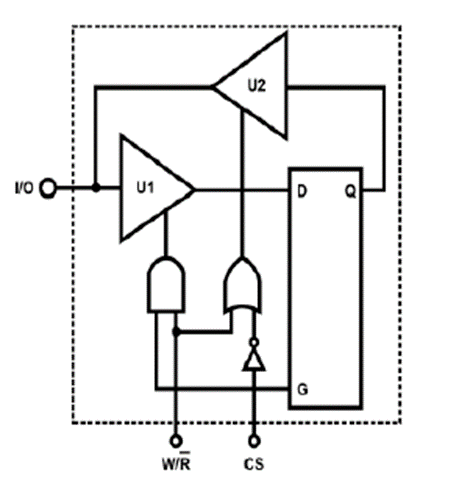


Figure (1)

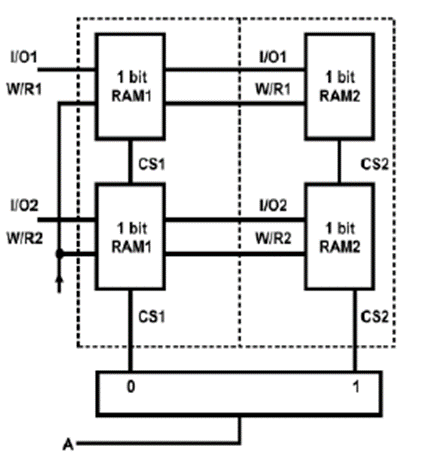
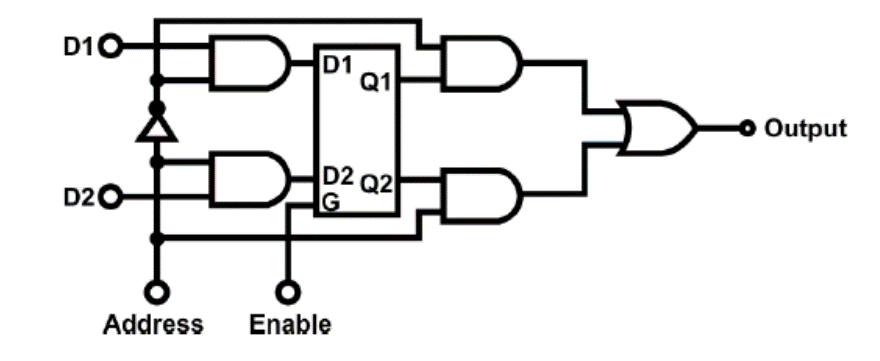
Figure (2) shows another connection of the RAM which will increase its capacity. Address Line A allows us to select between RAM1 and RAM2. As shown below there’s only one address line which means that we only can select one of 2 RAMs to use.

Figure (2)

Figure (3) shows a 2-address (2-bits) RAM circuit which has separated input and output. If Address = 0, then the data on D1 is enabled and made as an output. If Address = 1, then the data on D2 is enabled and made as an output. The Enable port must be 1 to have an output changing due to inputs.

Figure (3)

* 1. **64 bit Random Access Memory (RAM) Circuit**

The selection of data process is controlled by address selectors. Number of locations is determined by the number of address lines. If there are 4 address lines, then there are 16 locations (2^4) exist. Each location can store a 4-bit data since the capacity is 16×4, where 16 is the number of address lines and 4 is the number of data. Figure (4) shows a 16×4 memory with 64 memory capacities, and its function table.

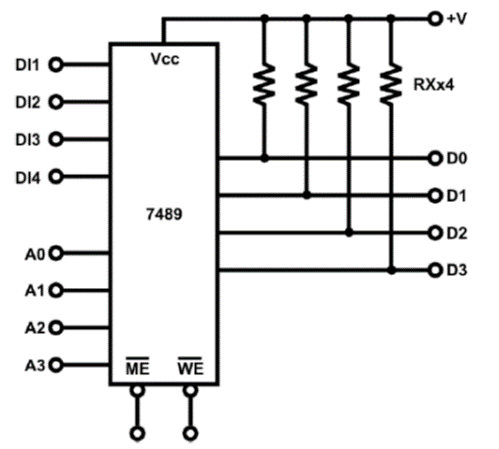


Figure (4)

1. **Procedure**

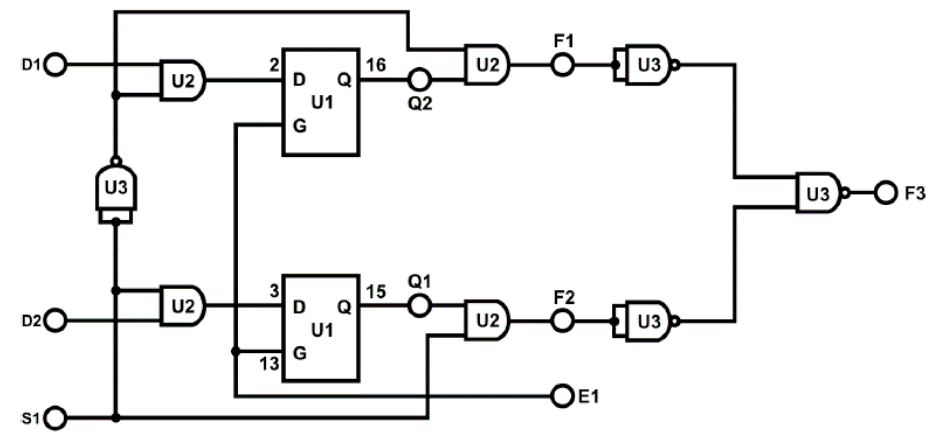
In this experiment, we built a random access memory (RAM) using D FLIP-FLOP. We used a RAM block with D Flip-Flop of module IT-3011.Connected the circuit as shown in Figure (5) below.

Figure (5)

From Figure (5), data of D1, D2 are entering D Flip-Flop. S1 is optional, if it equals "0”, then there’s no change. Otherwise, the value inside the D Flip-Flop of D1, D2 goes to F1, F2. Table (1) below shows the functionality of memory work.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | | | |  | | |
| E1 | S1 | D2 | D1 | F3 | F2 | F1 |
| 0 | 0 | 0 | 0 | 1 | 0 | 1 |
| 0 | 0 | 0 | 1 | 1 | 0 | 1 |
| 0 | 0 | 1 | 0 | 1 | 0 | 1 |
| 0 | 0 | 1 | 1 | 1 | 0 | 1 |
| 0 | 1 | 0 | 0 | 1 | 1 | 0 |
| 0 | 1 | 0 | 1 | 1 | 1 | 0 |
| 0 | 1 | 1 | 0 | 1 | 1 | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 | 0 |
| 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 1 | 1 | 0 | 1 |
| 1 | 0 | 1 | 0 | 0 | 0 | 0 |
| 1 | 0 | 1 | 1 | 1 | 0 | 1 |
| 1 | 1 | 0 | 0 | 0 | 0 | 0 |
| 1 | 1 | 0 | 1 | 0 | 0 | 0 |
| 1 | 1 | 1 | 0 | 1 | 1 | 0 |
| 1 | 1 | 1 | 1 | 1 | 1 | 0 |

Table (1)

We used the RAM block in module IT-3011 as shown in Figure (6). we stored the numbers inside the memory (RAM) then read the number, if (WE) equals '0' then the number will be stored in memory, and if (WE) equals '1' then we will read the number from memory.

To store a number in memory we changed (WE) to '0' and entered the number in (D0,D1,D2,D3) in the (RAM) block, then pressed the clock (ME) to do this operation. Else, to read the number which we stored in an address we changed (WE) to '1' and enter the Address number which we want to read from (A0,A1,A2,A3) in (RAM) block then press the clock (ME) to do this operation. Table 2 shows the functionality of the memory.

****

Figure (6)

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | | | |  | | | | | |  | | | | | |
| A3 | A2 | A1 | A0 | ME | WE | D4 | D3 | D2 | D1 | ME | WE | F4 | F3 | F2 | F1 |
| 0 | 0 | 0 | 0 |  | 0 | 0 | 0 | 0 | 0 |  | 1 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 1 |  | 0 | 0 | 0 | 0 | 1 |  | 1 | 0 | 0 | 0 | 1 |
| 0 | 0 | 1 | 0 |  | 0 | 0 | 0 | 1 | 0 |  | 1 | 0 | 0 | 1 | 0 |
| 0 | 0 | 1 | 1 |  | 0 | 0 | 0 | 1 | 1 |  | 1 | 0 | 0 | 1 | 1 |
| 0 | 1 | 0 | 0 |  | 0 | 0 | 1 | 0 | 0 |  | 1 | 0 | 1 | 0 | 0 |
| 0 | 1 | 0 | 1 |  | 0 | 0 | 1 | 0 | 1 |  | 1 | 0 | 1 | 0 | 1 |
| 0 | 1 | 1 | 0 |  | 0 | 0 | 1 | 1 | 0 |  | 1 | 0 | 1 | 1 | 0 |
| 0 | 1 | 1 | 1 |  | 0 | 0 | 1 | 1 | 1 |  | 1 | 0 | 1 | 1 | 1 |
| 1 | 0 | 0 | 0 |  | 0 | 1 | 0 | 0 | 0 |  | 1 | 1 | 0 | 0 | 0 |
| 1 | 0 | 0 | 1 |  | 0 | 1 | 0 | 0 | 1 |  | 1 | 1 | 0 | 0 | 1 |
| 1 | 0 | 1 | 0 |  | 0 | 1 | 0 | 1 | 0 |  | 1 | 1 | 0 | 1 | 0 |
| 1 | 0 | 1 | 1 |  | 0 | 1 | 0 | 1 | 1 |  | 1 | 1 | 0 | 1 | 1 |
| 1 | 1 | 0 | 0 |  | 0 | 1 | 1 | 0 | 0 |  | 1 | 1 | 1 | 0 | 0 |
| 1 | 1 | 0 | 1 |  | 0 | 1 | 1 | 0 | 1 |  | 1 | 1 | 1 | 0 | 1 |
| 1 | 1 | 1 | 0 |  | 0 | 1 | 1 | 1 | 0 |  | 1 | 1 | 1 | 1 | 0 |
| 1 | 1 | 1 | 1 |  | 0 | 1 | 1 | 1 | 1 |  | 1 | 1 | 1 | 1 | 1 |

Table (2)

1. **Conclusion**

From this experiment, we saw how memory works, how numbers are stored and reused for reading. Also we understood the mechanism of the (RAM), and that numbers in memory are deleted or replaced only if we stored another number in the same location of a previous data stored in that address.

1. **Feedback**

This experiment took approximately and hour and a half to complete

No noticeable errors were recorded

1. **References**
2. <https://en.wikipedia.org/wiki/Random-access_memory>