Chapter 6 | Memory

- Memory Elements
- Introduction
- Memory Chips
- Memory Units
- Layout
- Address Decoder Logic
- Storage Elements
- Big Endian and Little Endian

CS102 Computer Environment

Memory

Memory Elements

Introduction

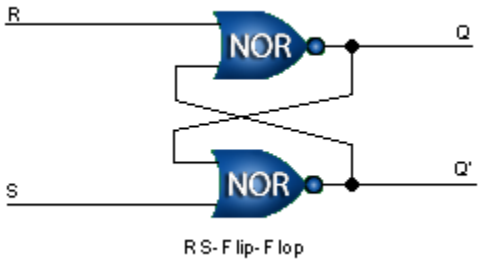
Half and full adders use combinatorial logic. Combinatorial logic circuits output depends only on the inputs. In many situations, the output depends on the current input. Circuits that remember their current output are called sequential logic. Memory is required by sequential logic circuits and memory elements can be constructed by appropriately feeding back the output of gates to the input.

RS-Flip-Flop

Stands for: Reset, Set Flip-Flop.

RS-flip-flop is the simplest possible memory element.

The RS-flip-flop is composed of two NOR gates.



RS-Flip-Flop

RS-flip-flop truth table

R	S	Q	Q'	Description
0	0	Q	Q'	Hold state
0	1	1	0	Set
1	0	0	1	Reset
1	1	?	?	Not Allowed

When S=1 and R=0.

a. The output at the bottom NOR gate is equal to zero (Q' = 0).

b. Hence, both inputs to the top NOR gate are equal to zero,

c. Thus Q =1.

d. Implies that the input combinations S=1 and R=0 leads to the
e. Flip-flop being set to Q=1.

When S=0 and R=1.

f. The output becomes Q=0 and Q'=1.
g. We say that the flip-flop is reset.

When S=0, R=0, Q=0 and Q'=1.

h. The output at the top NOR gate remains at Q=0.
i. The output at the bottom NOR gate stays at Q'=1.

When S=0, R=0, Q=1 and Q'=0.

j. The output at the top NOR gate remains at Q=1.
k. The output at the bottom NOR gate stays at Q'=0.

Therefore, when S=0 and R=0, the flip-flop remains in its state.

S=1 and R=1 is not allowed. Therefore, it should be avoided.

The output Q is the opposite of Q'.

The RS-flip-flop can also be constructed from NAND gates.

D-Flip-Flop

The D-flip-flop has a single data input.

The data input is connected to the S input of an RS-flip-flop.

The inverse of the D is connected to the R input.

The S=1 and R=1 combination will never occur.

To allow a flip-flop to be in a holding state, an additional Enable input is required.

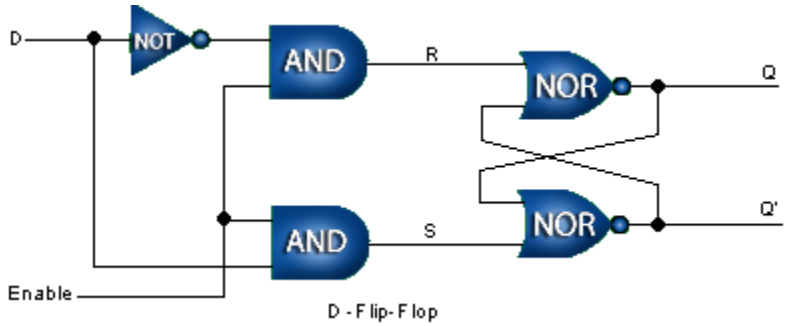
The Enable input is AND-ed with the D input.

When Enable=0, R=0 and S=0. Therefore, the flip-flop state is held.

When Enable=1, S=D and R is the inverse of D.

The value of D determines the value of the output Q when Enable=1.

When Enable returns to 0, the most recent input D is remembered.



D - Flip-Flop

D-flip-flop truth table

E	D	Q	Q'	Description
0	0	Q	Q'	Hold state
0	1	Q	Q'	Hold state
1	0	0	1	Reset
1	1	1	0	Set

Memory Chips

Memory chips are usually divided into two groups:

1. Read/write memory (RAM).
2. Read only memory (ROM).

A ROM is a non-volatile memory with a binary pattern of 0s and 1s permanently programmed by the manufacturer.

ROM memories come in four versions:

1. Standard ROM, programmed by the manufacturer.
2. PROM (Programmable Read-Only Memory). By using special equipment, The PROM can be programmed permanently by the user or distributor. However, it can only be programmed once.
3. EPROM (Erasable Programmable Read-Only Memory). Can be programmed and erased by the user. Stored data in the EPROM can be erased by shining high-intensity ultraviolet light through a special transparent window in the top of the IC.
4. EAROM (Electrically Alterable Read-Only Memory). It can be erased and programmed by the user, using special equipment.

The read write memory is a memory that can be easily programmed, erased and reprogrammed by the user.

The read/write memory is often called Random Access Memory (RAM).

RAM is made up of storage elements.

Storage elements could be similar to the D-flip-flop covered is the previous chapter.

Every storage element has a unique address or number represented in binary.

The size of the storage element is 8-bits.

The address is provided to the RAM chip in order to read from or write to the corresponding memory element.

Memory Units

8 bits = 1 byte.

Memory word = CPU number of bits.

1024 bytes = 1 Kilobyte (KB).

1024 KB = 1 Megabyte (MB).

1024 MB = 1 Gigabyte (GB).

Layout

Storage elements are arranged on a square grid.

Storage elements are made of columns and rows.

One1 Megabit or 128 Kilobyte of RAM consists of 1024 rows and 1024 columns of storage elements.

$1024 = [\sqrt{(128_{kb} \text{ x } 1024_{Byte} \text{ x } 8_{bits})}]$

$\text{Ceil } (\sqrt{\text{bits}})$

Each individual storage element can be identified through its coordinates.

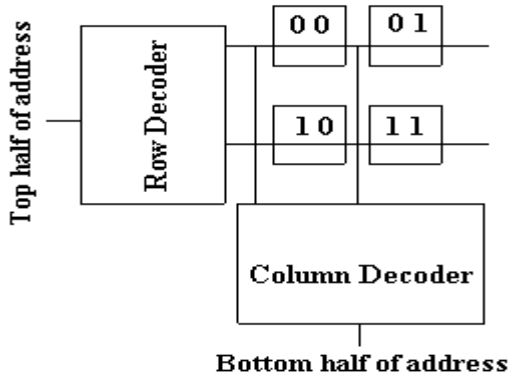
In other words, individual storage elements can be identified by their row and column.

To address a particular storage allocation (memory element), the address information must be translated into row and column specification.

The address information is divided into two halves.

1. The top half is used to select the row.
2. The bottom half is used to select the column.

A 4-bit RAM figure will look like this:



Address decoder logic

The address decoder has a binary number with N bits as its input.

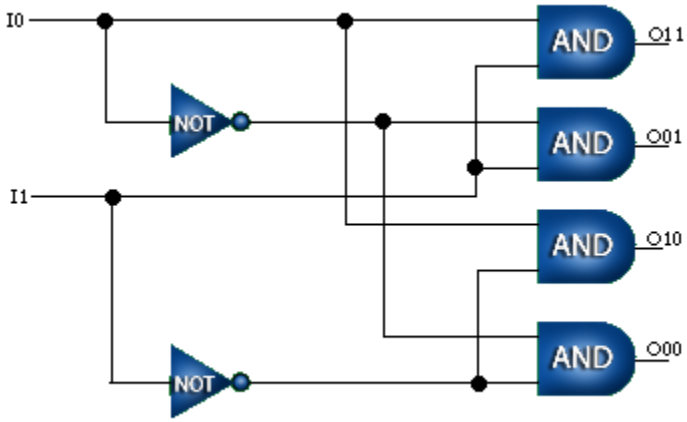
It has 2^N outputs.

At any time, only one output line is '1' and all the others are '0s'.

The line that is '1' specifies the desired column or row.

Decoder with two inputs and 2^N outputs.

$$N = \frac{\ln x}{\ln 2}$$



2-bit Decoder

2-bit decoder truth table

Input		Output			
I0	I1	O11	O10	O01	O00
0	0	0	0	0	1
0	1	0	0	1	0
1	0	0	1	0	0
1	1	1	0	0	0

Storage Elements

Each storage element is connected to one row and one column.

Since for every address, only one row and one column selector line will be '1', exactly one storage element can be selected.

The storage element is selected by AND-ing its row and its column selector line.

If a D-flip-flop is used as a storage element and connect its Enable input to the output of the AND gate mentioned above.

The D-input is connected to the data input line that is common to all storage elements.

The output of the flip-flop is AND-ed with the output of the first AND gate and then connected to a data output line that is common to all storage elements.

Only the selected storage element contributes to the common output line.

The read process should leave the value of the gate unchanged.

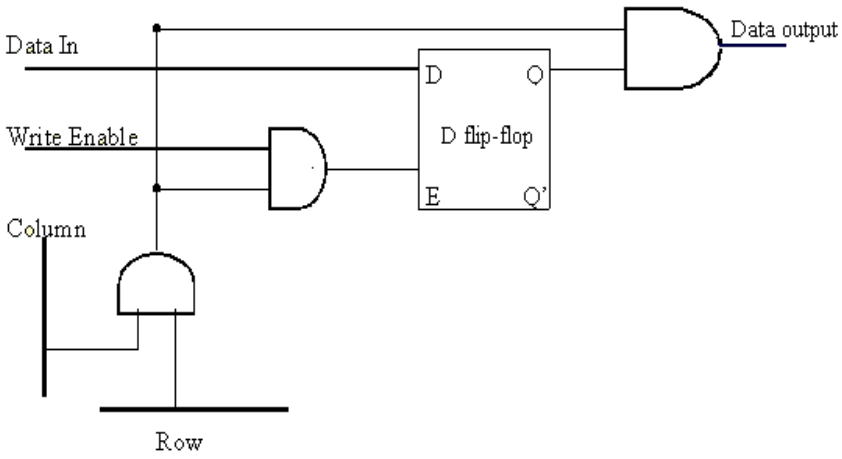
To prevent data destruction during a read operation a 'write enable' signal is introduced.

The signal is '1' only when new data is to be stored.

It is AND-ed with the output of the first AND gate and then applied to the flip-flop's enable input.

During read operations, the 'write enable' is '0', the flip-flop is disabled and does not change its state.

The write enable signal is common to all storage elements.



How to store numbers in memory

The area where a computer machine stores its data is called a memory.

A computer machine's memory is an array of consecutively numbered or addressed memory cells holding a bit value.

Every byte in a machine memory has a unique number or address.

Example: If we have a memory size of 8 bytes, memory address 0 will point to the first byte, memory address 1 will point to the second byte and memory address 7 will point to the last byte, byte number 8. This is what the empty memory will look like:

Address	Content	
0	00000000	(8 bits)
1	00000000	(8 bits)
2	00000000	(8 bits)
3	00000000	(8 bits)
4	00000000	(8 bits)
5	00000000	(8 bits)
6	00000000	(8 bits)
7	00000000	(8 bits)

This is what the memory will look like after we store:

- Value 7 in address 1,
- Value 3 in address 2,
- Value 8 in address 5, and
- Value 255 in address 6.

Address	Content	
0	00000000	(8 bits)
1	00000111	(8 bits)
2	00000011	(8 bits)
3	00000000	(8 bits)
4	00000000	(8 bits)
5	00001000	(8 bits)
6	11111111	(8 bits)
7	00000000	(8 bits)

How to store characters in memory

Since the computer machine can only understand binary numbers, the character set is represented by number.

Example: Upper case character A is represented by the number 65 in base 10 which is 01000001 in binary.

The ASCII character code tables contain the decimal values of the extended ASCII (American Standards Committee for Information Interchange) character set.

The extended character set includes the ASCII character set and 128 other characters for graphics and line drawing, often called the "IBM ® character set."

Example: If we want to write the word "Hello!" in a memory of size 8 bytes.

Address	Content base 2	Content base 10
0	00110000	48
1	01100101	101
2	01101100	108
3	01101100	108
4	01101111	111
5	00100001	33
6	00000000	00
7	00000000	00

Big Endian and Little Endian

Computers are designed around two different architectures for handling memory storage.

Big Endian and little Endian refer to the order in which bytes are stored in memory.

Big Endian is derived from 'Big End In' and little Endian is derived from 'Little End In'.

On an Intel based CPU computer, the little end is stored first.

A 4-byte value like 0x87654321 would be stored as 0x21 0x43 0x65 0x87.

On a Motorola based CPU computer, the big end is stored first.

A 4-byte value like 0x87654321 would be stored as 0x87 0x65 0x43 0x21.

Let's see what the little Endian based memory will look like after we store.

- Value 258_{10} = 00000001 00000010_2 = 0102_{16} in address 4.
- Value 258_{10} requires 2 bytes of storage.

Address	Content base 2	Content base 16	
0	00000000	00	(8 bits)
1	00000000	00	(8 bits)
2	00000000	00	(8 bits)
3	00000000	00	(8 bits)
4	00000010	02	(8 bits)
5	00000001	01	(8 bits)
6	00000000	00	(8 bits)
7	00000000	00	(8 bits)