CS225: Switching Theory

End Semester Assignment Report

Submitted By:

Name: Tarusi Mittal

Roll No:1901Cs65

A1

Write a critical analysis report on Computer memory Architectures: Past and Present. (Use tables for critical comparison, type Architecture, no of cells, Number of Transistors, Memory Capacity, speed, year of introduction, application, power consumption, technology used etc.)

Solution

Memory:

What is memory? Where is it used?

Memory refers to the processes that are used to acquire, store, retain, and later retrieve information. There are three major processes involved in memory: encoding, storage, and retrieval.
In computing, memory refers to a device that is used to store information for immediate use in a computer or related computer hardware device. Memory forms the essentials of computer architecture and is essential to the functioning of every computer

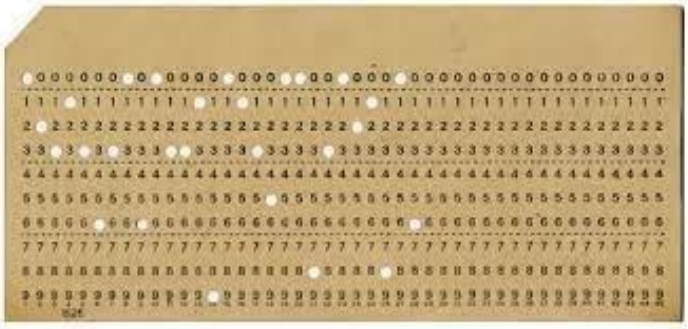
Volatile vs. non-volatile memory.

Memory can be either volatile and non-volatile memory. Volatile memory is memory that loses its contents when the computer or hardware device loses power. Computer RAM is an example of volatile memory. It is why if your computer freezes or reboots when working on a program, you lose anything that hasn't been saved. Non-volatile memory, sometimes abbreviated as NVRAM, is memory that keeps its contents even if the power is lost. EPROM is an example of non-volatile memory. Memory architecture describes the methods used to implement electronic computer data storage in a manner that is a combination of the fastest, most reliable, most durable, and least expensive way to store and retrieve information.

History

Here is the timeline of the early memory:

1. Punch Cards: Punch cards (or "punched cards"), also known as Hollerith cards or IBM cards, are paper cards where holes may be punched by hand or machine to represent computer data and instructions. They were a widely-used means of inputting data into early computers. The cards were fed into a card reader connected to a computer, which converted the sequence of holes to digital information. They have been dated back to as early as 1725. Data is stored by punching holes into the cards which is then interpreted by a machine.The punch cards are read column wise.



2. **Vacuum Tubes**: Introduced in the early 1940s, the Vacuum Tubes were used to store numbers in an octal-base and were capable of operating on around 20 numbers at a time with each number ranging upto 10^{10}.

3. Delay Line Memory: This came into the industry in the early 1950s. This memory was built using Mercury tubes. It analysed the sound waves in the tubes to store, read and write data.

4. Williams Tube: Introduced in the late 1950s, this is said to be the first random accessible memory in computing. It was a major improvement over its predecessor 'Selectron Tube' and was less expensive.

5. Magnetic Memory: The memories developed earlier were volatile and would not retain after power loss. This led to researchers to dive into creating memory which retained data written on it and which could be

read from and written to. Researchers developed forms of magnetic memory which was improved over the years. This memory replaced all tube-based memories since it was highly space efficient and could store more data and also was non-volatile. Different forms of magnetic memory were developed:

a. Magnetic Tape: The magnetic tape works by using a magnetizable material coated on a plastic layer. It was developed in Germany in 1928 and revolutionised the sound and music industry. The cassettes used by us in the late 1990s and early 2000s all run on the principle of this magnetic tape.

b. Floppy Disk: These were developed in the late 1960s and commercially available in 1971. The floppy disk has certain concentric magnetic tracks. As the read/write control head passes over it, it uses magnetic polarisation to write and write data. The data is stored in binary form '0' or '1' and they are represented by magnetic north and south poles. Transfer Speeds go upto 300 kB/second. The access time is around 200 milliseconds which is excessively slow as compared to modern memory. It can store upto around 1.4 MB of data in an average disk. Therefore it consumes a large amount of space.

c. The Hard Disk Drive: First introduced and popularised in the late 1970s, the Hard Disk Drive was a significant improvement over the traditional Magnetic Memory like Floppy Disks. It worked on the same principle, however, it used very highly compact ferromagnetic films to store data. This allowed the disk to store larger amounts of data than the traditional floppy. HDDs were excessively cheap and became so popular that they were brought into use by IBM in almost all their computing devices. From there on the HDDs are used as mass storage inside the modern computers, even in today's times. As of 2020, the HDDs are capable of rotating at 5400-7200 RPM which allows for very high speeds of data writing and reading. The compact size makes it one of the most efficient memories in terms of data stored: size ratio. The HDDs today are capable of storing upto 5-10 TBs of data easily!

Modern Memory

What is Semiconductor Memory?

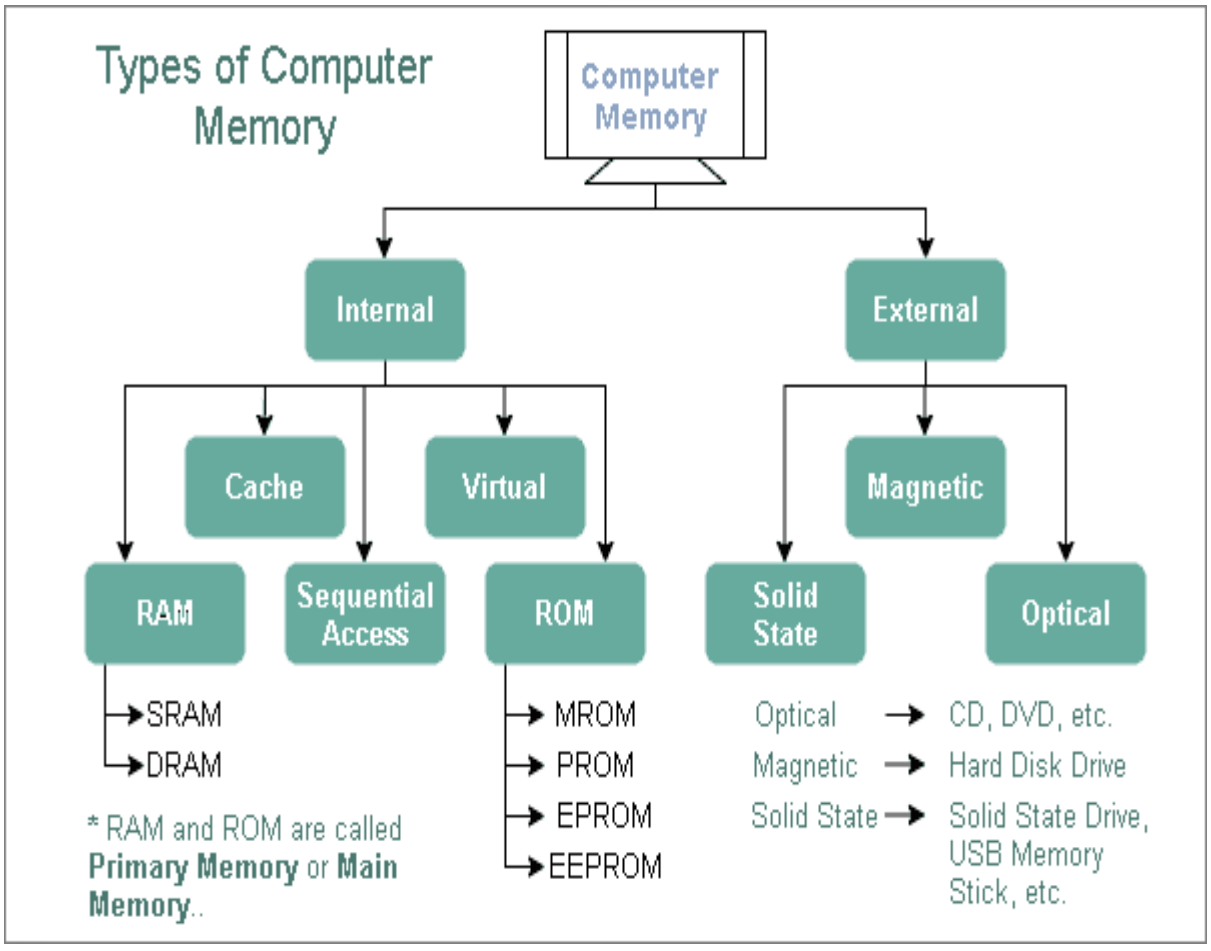
Semiconductor memory is a type of semiconductor device tasked with storing data. There are two electronic data storage mediums that we can utilize, magnetic or optical.

Magnetic storage: Stores data in magnetic form, Affected by magnetic fields, Has high storage capacity, Doesn't use a laser to read/write data. Magnetic storage devices are; Hard disk , Floppy disk, Magnetic tape etc.

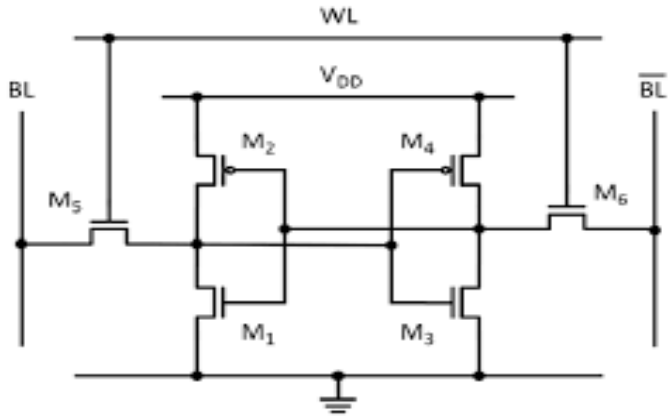
Optical storage: Stores data optically, uses laser to read/write, Not affected by magnetic fields, Has less storage than a hard disk, Data accessing is high, compared to a floppy disc.

Optical storage devices are; CD-ROM, CD-R, CD-RW, DVD etc.

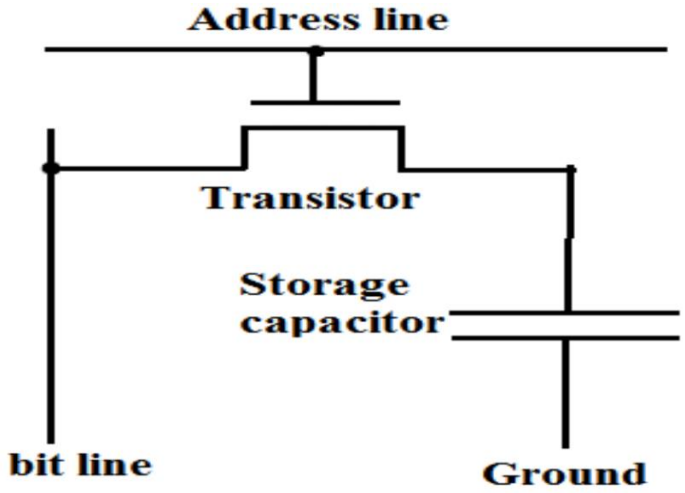
Some modern semiconductor based memory architectures are as follows:



1. Static Random Access Memory (SRAM): Static RAM or SRAM is a volatile memory in which data is lost as soon as power cuts off. It is generally implemented as an array of an array of cells. Each of these cells is basically a latch formed with two not gates and two NMOS transistors. In total each cell contains 6 MOSFET Transistors. The data stored in SRAM is stored in a 'static' manner. No need for refreshing the data is there. SRAM is very highly fast in terms of data access and read/write due to the fact that it does not require any refreshing time for its memory cells. It is expensive and very large in size. It is used as a cache for CPUs.



2. Dynamic Random Access Memory (DRAM): Dynamic RAM or DRAM is again a volatile memory in which data is lost as soon as power cuts off. Like SRAM, it is also implemented as an array of an array of cells. Each of these cells has one NMOS transistor and a small capacitor. The capacitor stores the voltage of the data bit to be stored in the cell. The transistor is used for conduction during writing. The main issue with the design is that the capacitor has very low capacitance due to which it discharges very quickly. Due to this the DRAM must be refreshed within stipulated intervals of time (~64ms for DDR2 RAM). This is why it is called Dynamic RAM.



3. Read Only Memory (ROM): Read Only Memory is a non-volatile memory in which the data is hardcoded on the circuit. The data is thus stored in a more or less permanent fashion on the circuit. The data is not changeable in the basic versions of the ROM. ROM variants like EPROM (Erasable Programmable ROM) can be erased and reprogrammed electrically/UV light several times through quantum tunneling.

4. Flash Memory: Flash Memory is a non-volatile Read Only Memory which is electrically erasable and programmable (EPROM). It uses FN Tunneling for programming and erasing. In terms of speed, it has a read speed comparable to DRAM (~ns) and write speed comparable to a disk (~ms). Each individual cell of the flash memory is implemented using a floating gate MOSFET. Charge inversion towards the floating gate is responsible for programming and charge accumulation away from the floating gate is responsible for erase of data. Data is read by measuring the effect of applying intermediate voltage (between possible threshold voltages according to the number of states) applied on the cell. The cell may even store more than just one bit since it may have more than two states. There are two types of flash memory based on the gates used NOR and NAND.

Feature	NOR Flash		NAND Flash	
	General	S70GL02GT	General	S34ML04G2
Capacity	8MB – 256MB	256MB	256MB – 2GB	256MB
Cost per bit	Higher	6.57×10^{-9} USD/bit for 1ku	Lower	2.533×10^{-9} USD/bit for 1ku
Random Read speed	Faster	120ns	Slower	30µS
Write speed	Slower		Faster	
Erase speed	Slower	520ms	Faster	3.5ms
Power on current	Higher	160mA (max)	Lower	50mA (max)
Standby current	Lower	200µA (max)	Higher	1mA (max)
Bit-flipping	Less common		More common	
Bad blocks while shipping	0%		Up to 2%	
Bad block development	Less frequent		More frequent	
Bad block handling	Not mandatory		Mandatory	
Data Retention	Very high	20 years for 1K program-erase cycles	Lower	10 years (typ)
Program-erase cycles	Lower	100,000	Higher	100,000
Preferred Application	Code storage & execution		Data storage	

5. Solid State Drive (SSD): A solid-state drive (SSD) is a solid-state storage device that uses integrated circuit assemblies to store data persistently, typically using flash memory. Compared with the electromechanical drives, SSDs are typically more resistant to physical shock, run silently, and have quicker access time and lower latency.

Table of Comparison of different Memory types

Property	Past			Present			
	Tube Memory	**Magnetic Tape (Floppy)**	**Optical Disk (CD)**	**Modern HDD**	**DRAM**	**SRAM**	**Flash Memory**
Years Active	1940s	1971 - 2000	1982 - 2010	1956 - present	1970 - present	1987 - present	1987 - present
Architecture	Based on physical properties of matter	Based on magnetic polarisation of materials	Based on laser light reflection	Based on magnetic polarisation of small materials	Based on MOSFET (CMOS) Architecture	Based on MOSFET (CMOS) Architecture	Based on F-N Tunneling
Speed	Very Very Slow	Very Slow ~100 kB/s	Very Slow ~300 kB/s	Moderate ~150 Mb/s	Fast ~15 Gb/s	Very Fast ~80 Gb/s	Moderate ~100 Mb/s
Density (Capacity)	Very Low ~1 kB	Low ~1 MB	Slightly Low ~600 MB	Very high ~1 TB	High ~16 GB	Moderate ~16 MB	Very Very High ~ 256 GB
Power	High	Moderate	Very High	High	Low	High	Very Low
Volatile	Yes	No	No	No	Yes	Yes	No
Refresh	N/A	No	No	No	Yes	No	No
Transistors	Nil	Nil	Nil	Nil	1 CMOS	6 CMOS	1 floating gate MOSFET
Application	Early Computers	Removable storage for early computers	Removable and dense storage for 1990s computers, used heavily in entertainment industry	High Density storage for modern day computers	Used as Random Memory for modern Computers	Used as Cache Memory for Modern Day Computers	Used in the construction of SSDs and as portable small storage devices with very high capacity.

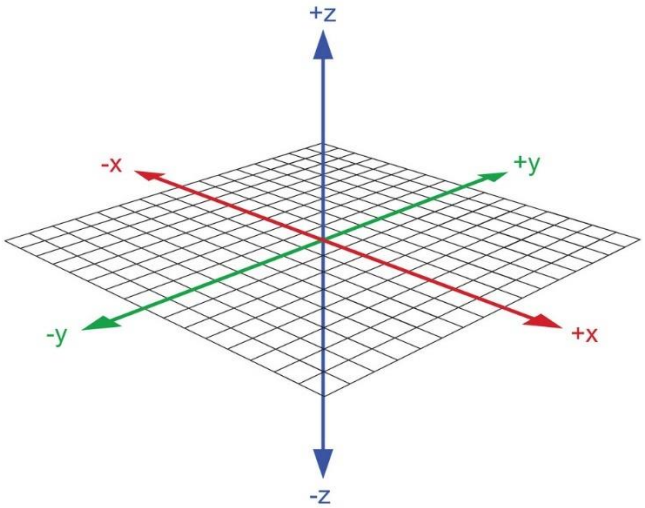
A2

Identify a practical problem that you are familiar which can be formulated as a Finite State Machine (FSM). Clearly state the problem and solve using one of the FSM design approaches. Show all the states, inputs , outputs, and design steps. Each state is represented with a circle, and each transition with an arrow. Simulate the solution using Logicsim and include screen shots in the assignment file (Also submit *.circ file; use filename Roll_no_ A2.circ)

Solution

Problem: An escape drone that has the capability to find the path out of a cave with one or more exits.

Sometimes there is a situation that while making roads in between the mountains or finding things inside caves, caves often collapse trapping the persons inside. This drone once enabled will react to the intensity of the light and move the itself in that particular direction and hence successfully taking people out of the cave. Once the person moves out of the cave the drone can be manually turned off.
The drone will have sensors on all the six sides of it and one by one they will measure the intensity. The direction will be chosen along which the intensity will be increasing. When the intensity starts to remain same or start decreasing along the particular direction then the other side sensor will measure the intensity and the process will be repeated. Hence the drone is able to move along on all the six axis(+x,-x,+y,-y,+z,-z).



We will consider the drone standing at the point 0,0,0 initially.

States of FSM
The states of FSM determine the behaviour of FSM. In our escaping drone its function is to move to the direction where the intensity of light is greater.

Our escaping drone FSM consists of 6 states:

1. **Right** - It is our initial state. Move the drone towards the +x axis
2. **Left** – Moves the drone towards the -x axis.
3. **Up** - Moves the drone towards the +z axis.
4. **Down** - Moves the drone towards the -z axis.
5. **Front** - Moves the drone towards the +y axis.
6. **Back** - Moves the drone towards the -y axis.

External Input to the FSM:

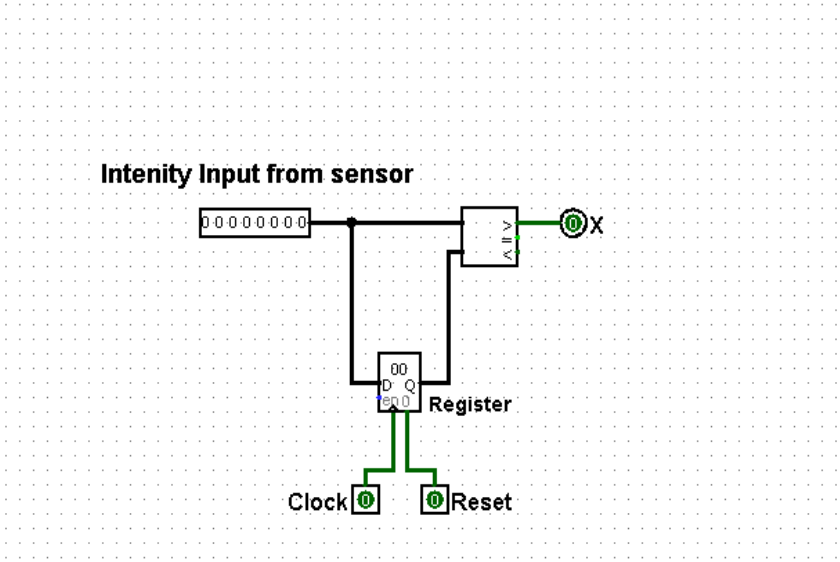
For our drone we are taking the intensity of the light as our input. For now let us consider that we are measuring it by one sensor and it can change direction so as one sensor can measure intensity in every direction.

We will use a Boolean function to take the input of intensity in our problem.

$$x(c,p) = c>p$$

here c is the intensity at the current position and p is the intensity at the previous position.

We will use this x as the input to our FSM. We will first implement the logic for x.
In this we will be storing the value of previous intensity and then comparing it with the current intensity. If the intensity is higher the output will be 1 otherwise 0.



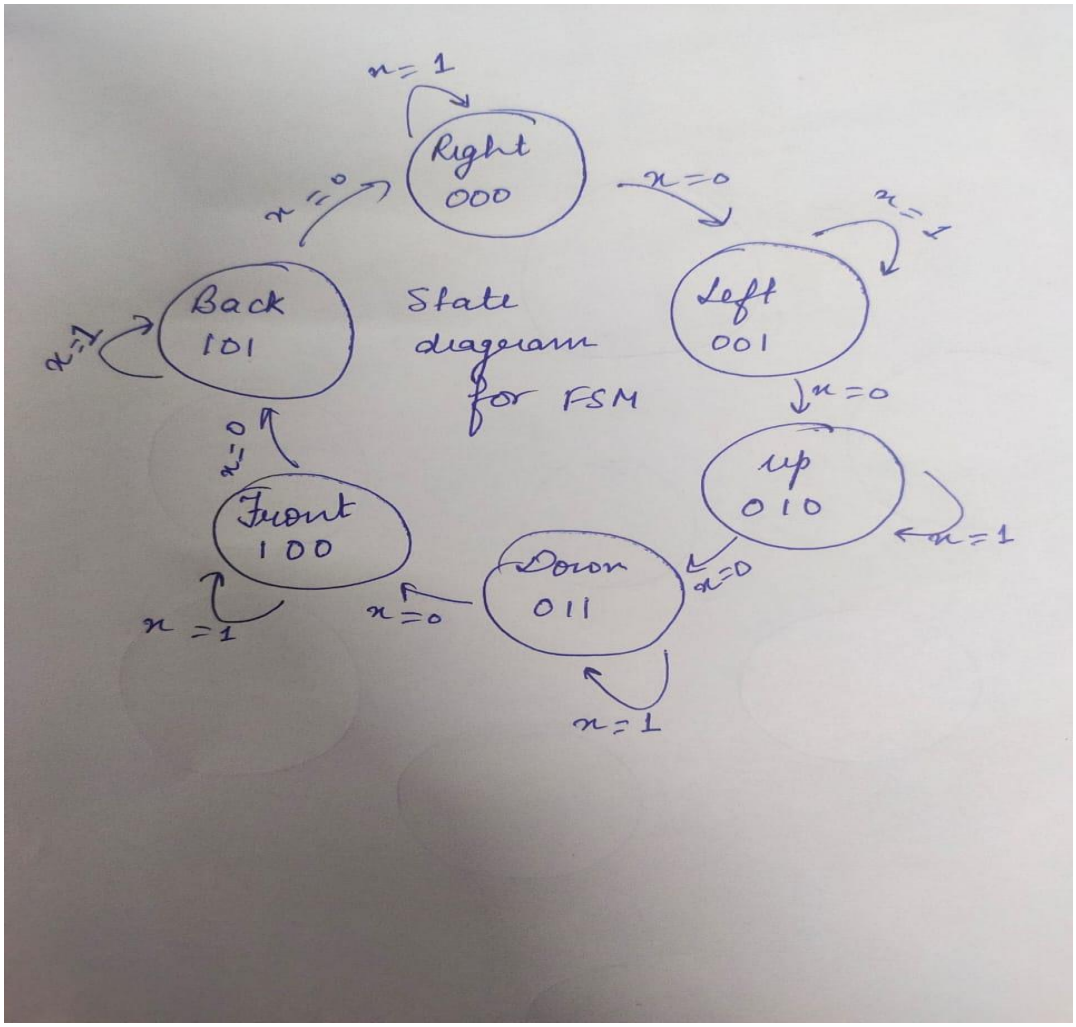
From the circuit before we will be giving the input to our FSM.

Output of FSM
The output of the FSM is as follows:

Direction along which the drone will move:

1. 000: This output will instruct the drone to move in the right direction (+x axis)
2. 001: This output will instruct the drone to move in the left direction (-x axis)
3. 010: This output will instruct the drone to move in the upward direction (+z axis)
4. 011: This output will instruct the drone to move in the downward direction (-z axis)
5. 100: This output will instruct the drone to move in the front direction (+y axis)
6. 101: This output will instruct the drone to move in the back direction (+y axis)

The state diagram:



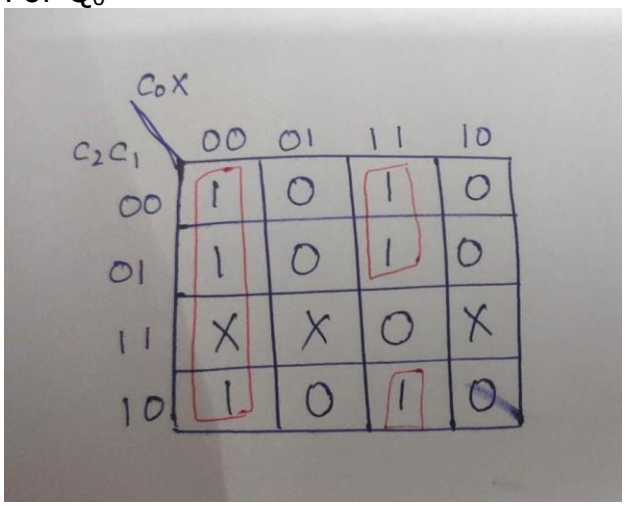
The Transition Table

PRESENT STATE			INPUT (x)	NEXT STATE			DIRECTION OF MOVEMENT		
C_2	C_1	C_0	x	Q_2	Q_1	Q_0	D_2	D_1	D_0
0	0	0	0	0	0	1	0	0	0
0	0	0	1	0	0	0	0	0	0
0	0	1	0	0	1	0	0	0	1
0	0	1	1	0	0	1	0	0	1
0	1	0	0	0	1	1	0	1	0
0	1	0	1	0	1	0	0	1	0
0	1	1	0	1	0	0	0	1	1
0	1	1	1	0	1	1	0	1	1
1	0	0	0	1	0	1	1	0	0
1	0	0	1	1	0	0	1	0	0
1	0	1	0	0	0	0	1	0	1
1	0	1	1	1	0	1	1	0	1
Other States			X	X	X	X	X	X	X

K-Maps

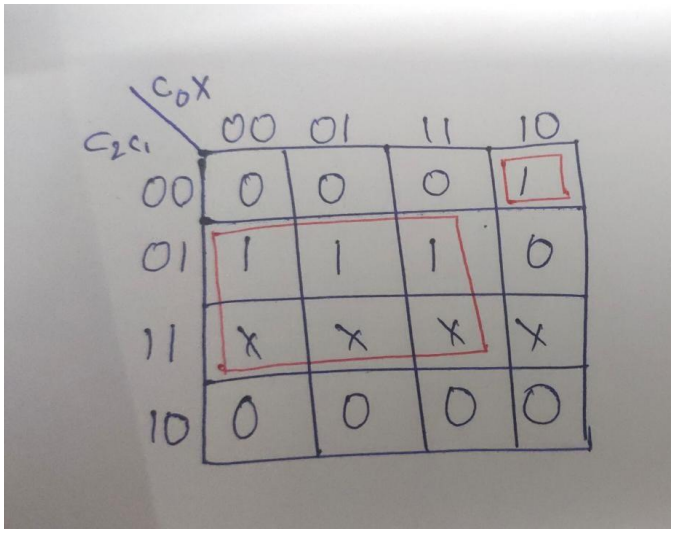
Now from the transition table we will find the outputs for next state and direction of movement.

For Q_0



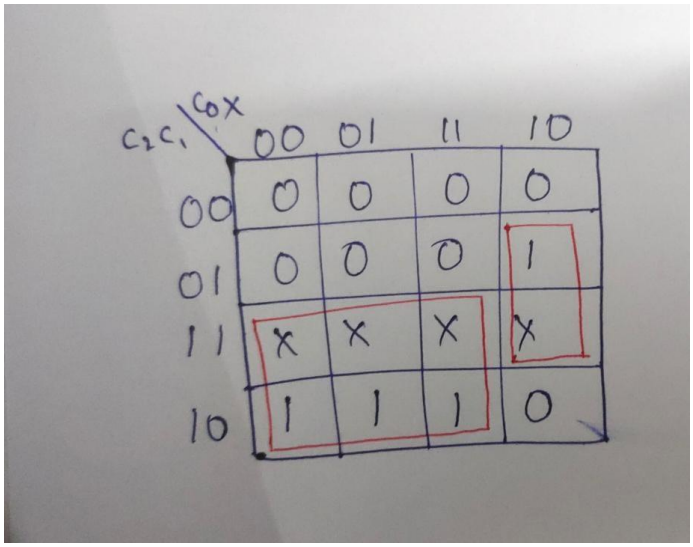
$$Q_0 = C_0'x' + C_2'C0x + C_1'C_0x$$

For Q_1



$$Q_1 = C_2'C_1'C_0x' + C_1C_0' + C_1x$$

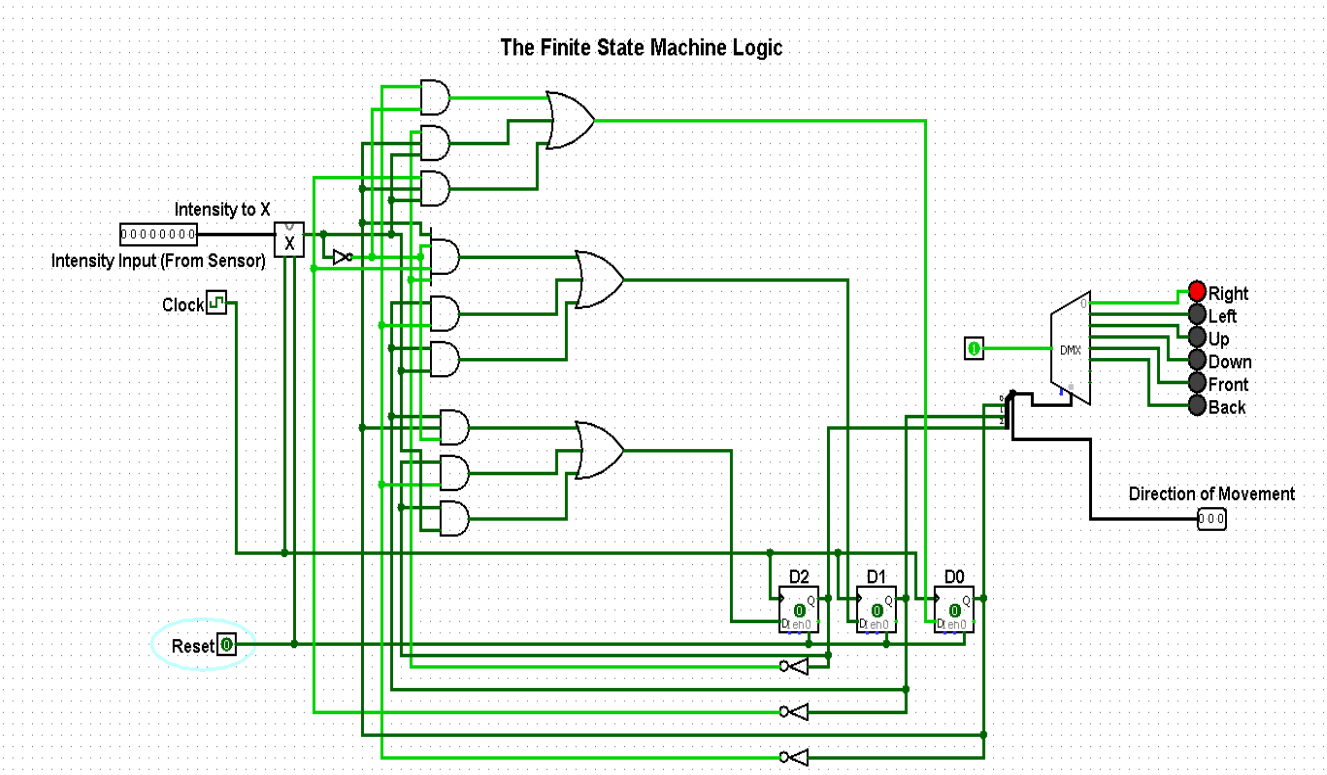
For Q_2

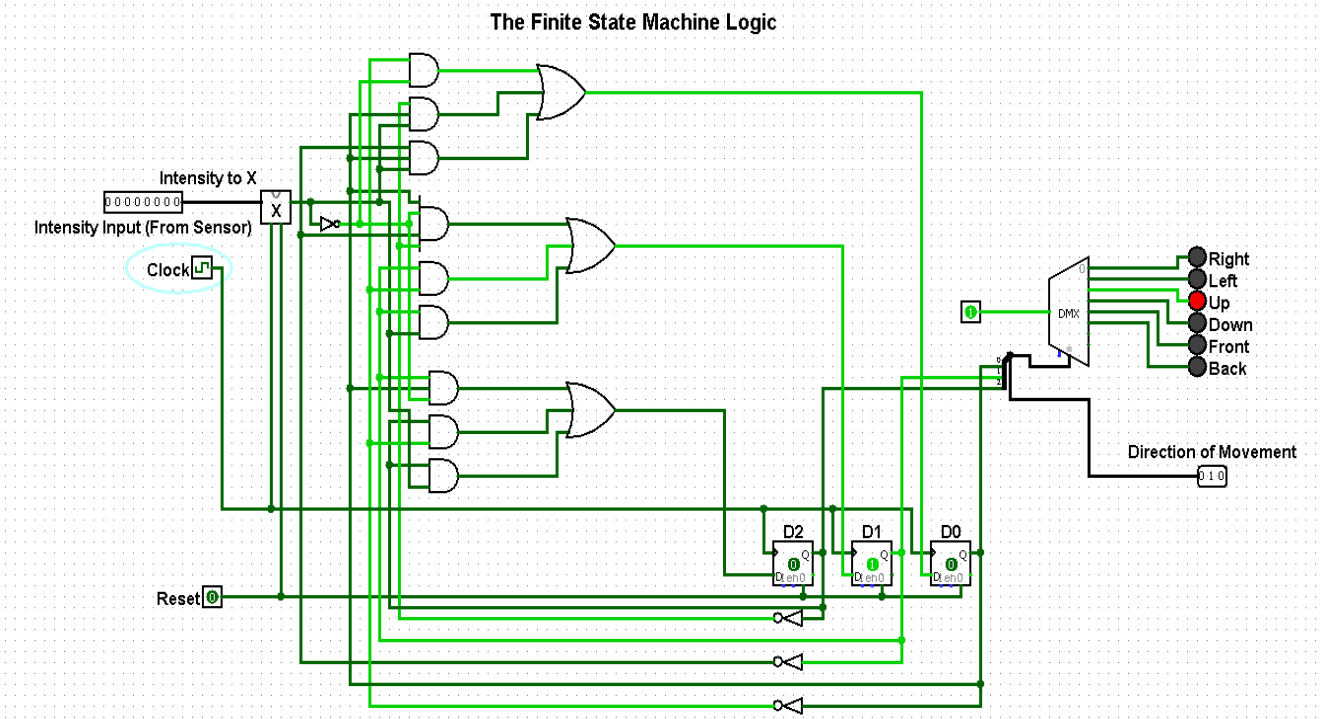


$$Q_2 = C_1C_0x' + C_2C_0' + C_2x$$

For D_2, D_1 and D_0 we can see from the table that they are equal to C_2, C_1 and C_0 respectively

The Circuit Diagram





Working of the Circuit

1. At the initial position our circuit is at the state 000 which is along the +x axis that is the "Right". The drone will continue to move towards the right direction until x=0 i.e the intensity in that direction starts to fall down. As soon as x=0, it will operate the sensor of the left side(001).

2. Now the drone has completed in the right direction(000) and it will start analysing the left direction(001). The drone will continue to move towards the left direction until x=0 i.e the intensity in that direction starts to fall down. As soon as x=0, it will operate the sensor of the upward side(010).

3. Now the drone has completed in the left direction(001) and it will start analysing the upward direction(010). The drone will continue to move towards the upward direction until x=0 i.e the intensity in that direction starts to fall down. As soon as x=0, it will operate the sensor of the downward side(011).

4. Similar steps will happen in the downwards, front and the backward side.

5. After the drone completely analyses the backward side the sensor again shifts to the right side and the process will start again.

This drone will automatically lead the person to the exit of the cave having with the maximum intensity in case of multiple exits. After evacuating the cave the drone can be manually turned off. The reset button of the FSM will help to bring the drone in the initial condition.

A3

Design an experiment which conveys one of the concepts in CS225. Clearly state the aim of the experiment, method of solving, and solution Create a logic-sim set up and simulate the same to convey the concept and include screen shots in the assignment file (Also submit also *.circ file; use file name: Rollno_A3.circ

Solution

Experiment: Designing and implementing a 4x2 SRAM with all functionalities like read and write from a particular address using binary cells.

What is SRAM?
Static random-access memory (**static RAM** or **SRAM**) is a type of random-access memory (RAM) that uses latching circuitry (flip-flop) to store each bit. SRAM is volatile memory; data is lost when power is removed.

SRAM offers a simple data access model and does not require a refresh circuit. Performance and reliability are good and power consumption is low when idle.

Since SRAM requires more transistors to implement, it is less dense and more expensive than DRAM and also has a higher power consumption during read or write access.

Designing the Circuit:

To design the circuit, we will first start with the binary cell.

A binary cell is a basic memory unit to store a binary value. We will implement binary cell using SR latch.

As we know SR Flip Flops can be used to store one bit of data. We will use this property and add some additional circuit to m=change it into a binary cell of RAM.

The components required are

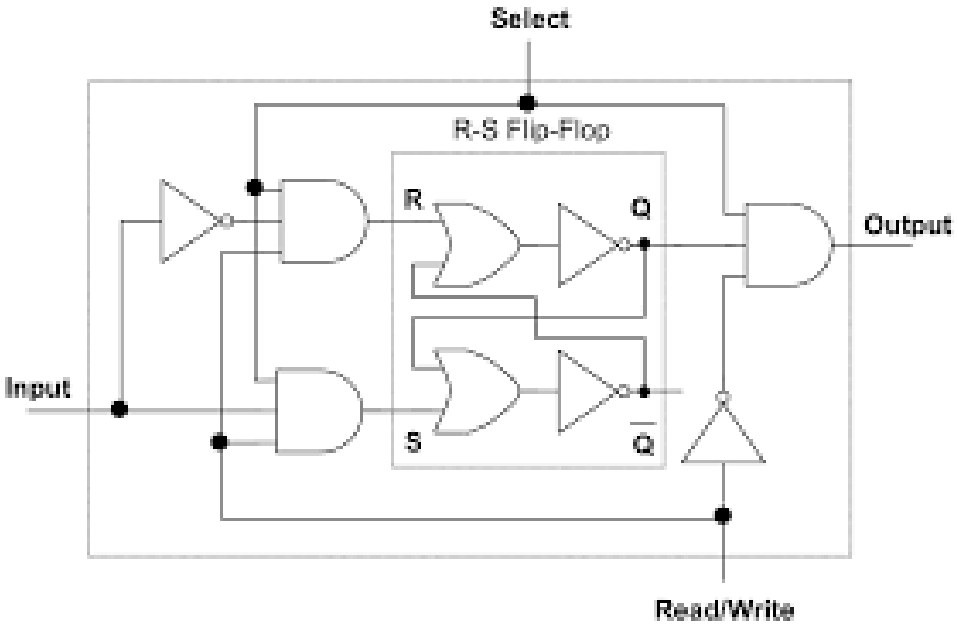
1. Input: The data to be stored

2. Write/Read: It tells whether the data has to be written or read. Here we will use) for reading and 1 for writing.

3. Select: Our binary cell will only be activated when the select is in "High" state.

4. Output: It gives the output when "Read" is enabled.

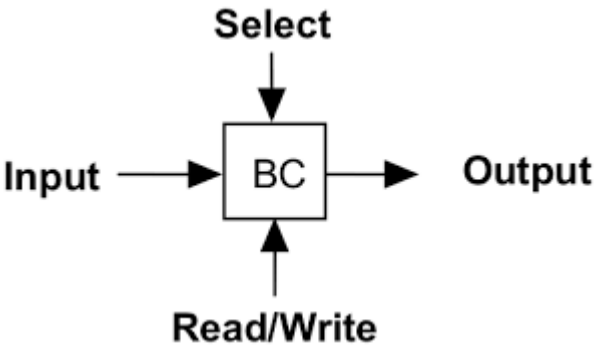
Here is a simple logic showing the binary cell



BINARY CELL(BC)

Reference:

From now we will show the Binary Cell(BC) as this:



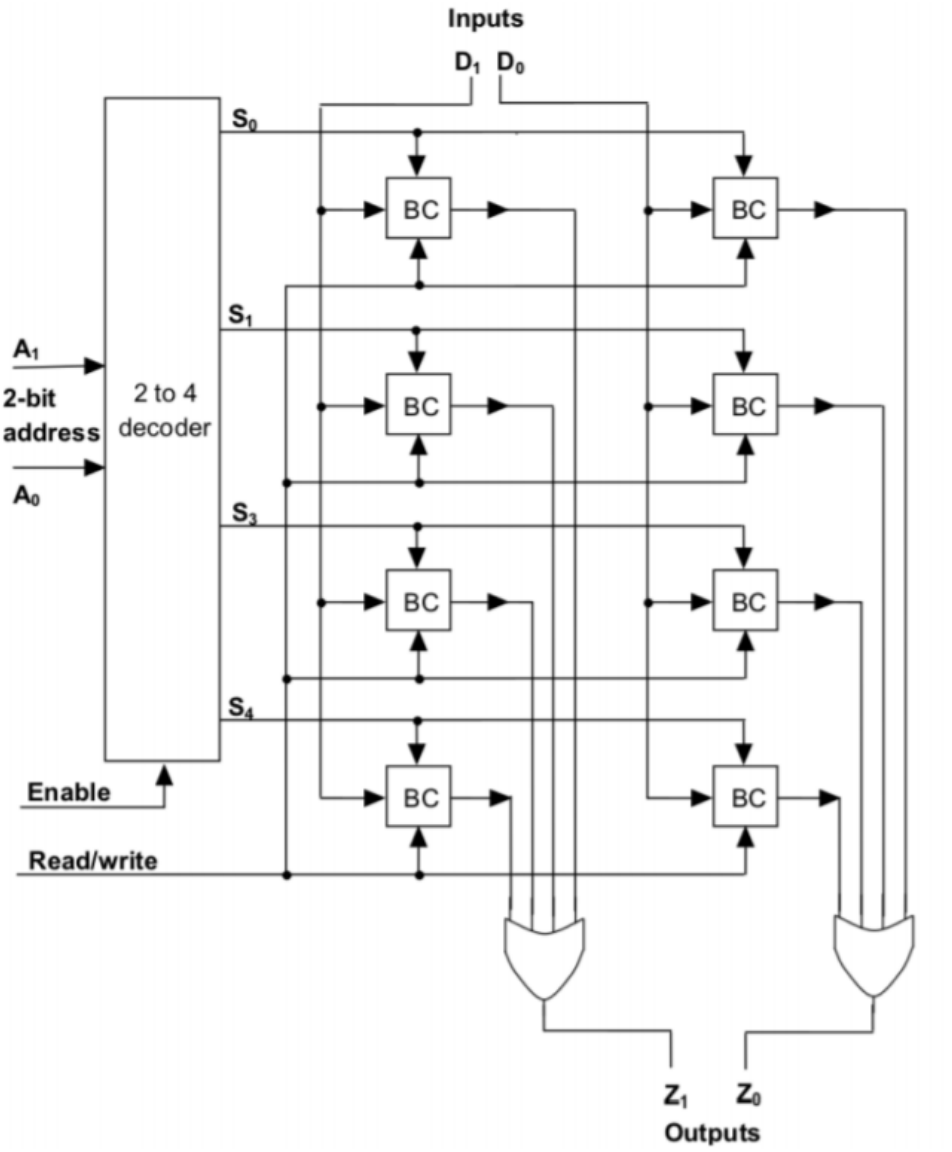
Precautions:

While implementing the BC these things should be taken care of:

1. Read and Write can't be performed at the same time
2. Output is only visible when both "Select" and "read" allow
3. Input can only be saved when both "Select" and "write" allow
4. R and S being compliments prevents the possibility of the forbidden value [R = "high", S = "high" simultaneously]

To build a 4x2 RAM using binary cells we need the following functionalities:

1. 2 to 4 Decoder: To select the right binary cells using given address
2. 2- bit Input
3. 2-bit output
4. Read/Write Address

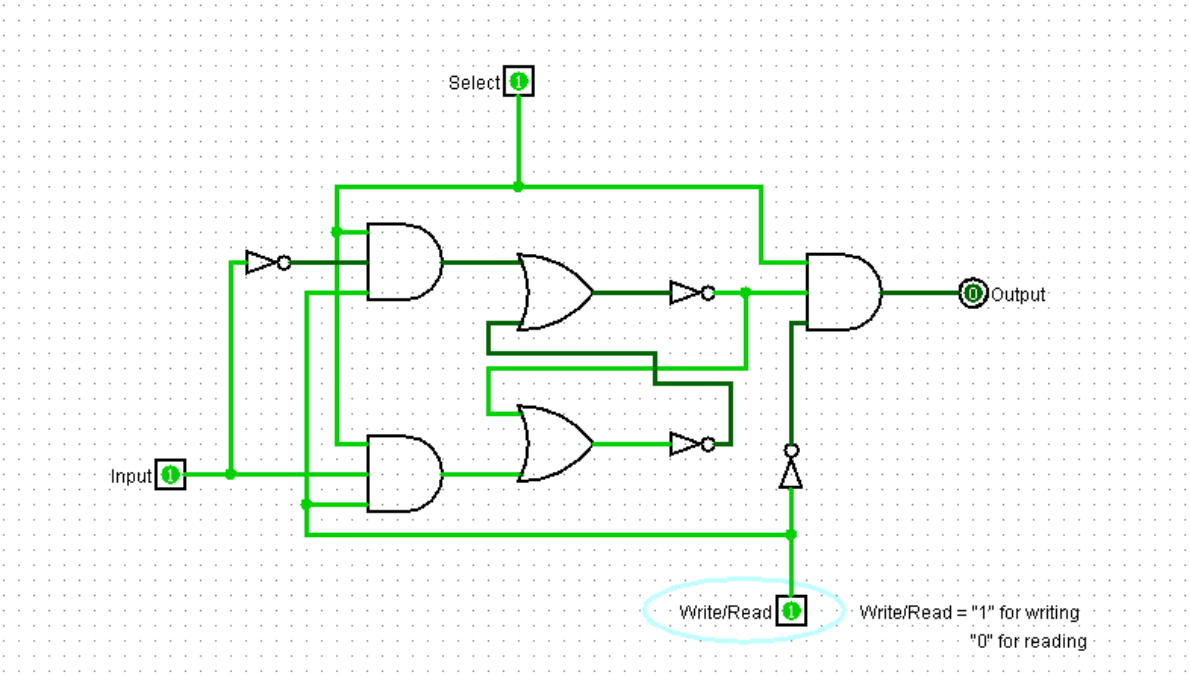


4 x 2 RAM Memory (SRAM)

(Using Binary Cell (BC) Memory Units)

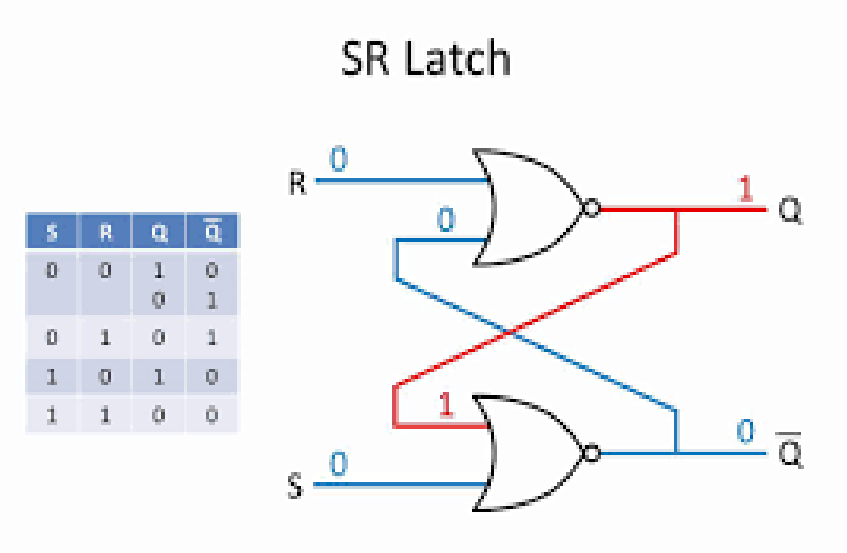
LOGISIM IMPLEMENTATION

Binary Cell:

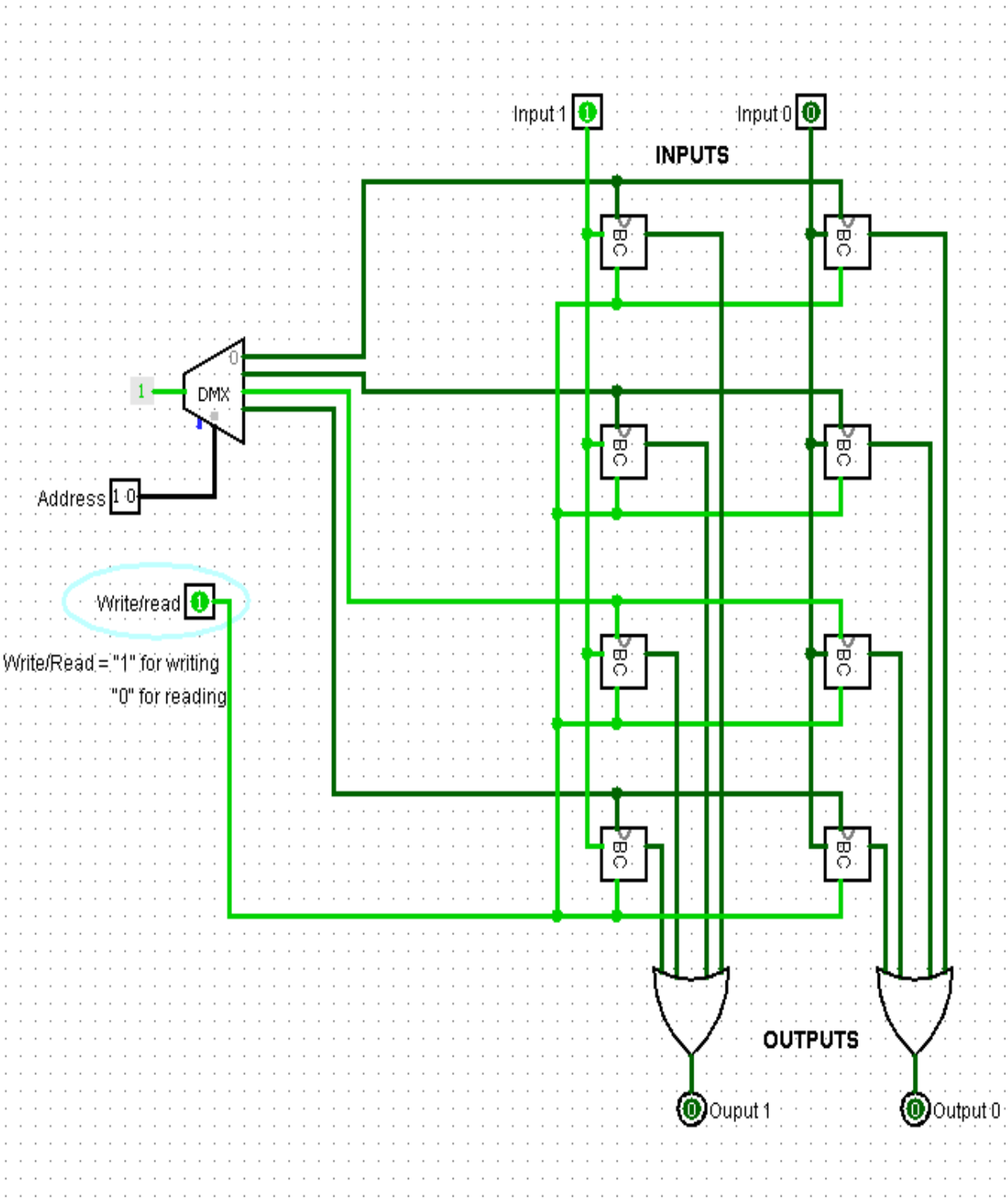


A Binary Cell(BC)
For RAM Memory

Here our binary cell is able to store one bit
The binary cell is implemented using the SR latch.
The SR latch has been implemented using logic gates.



4 x 2 SRAM

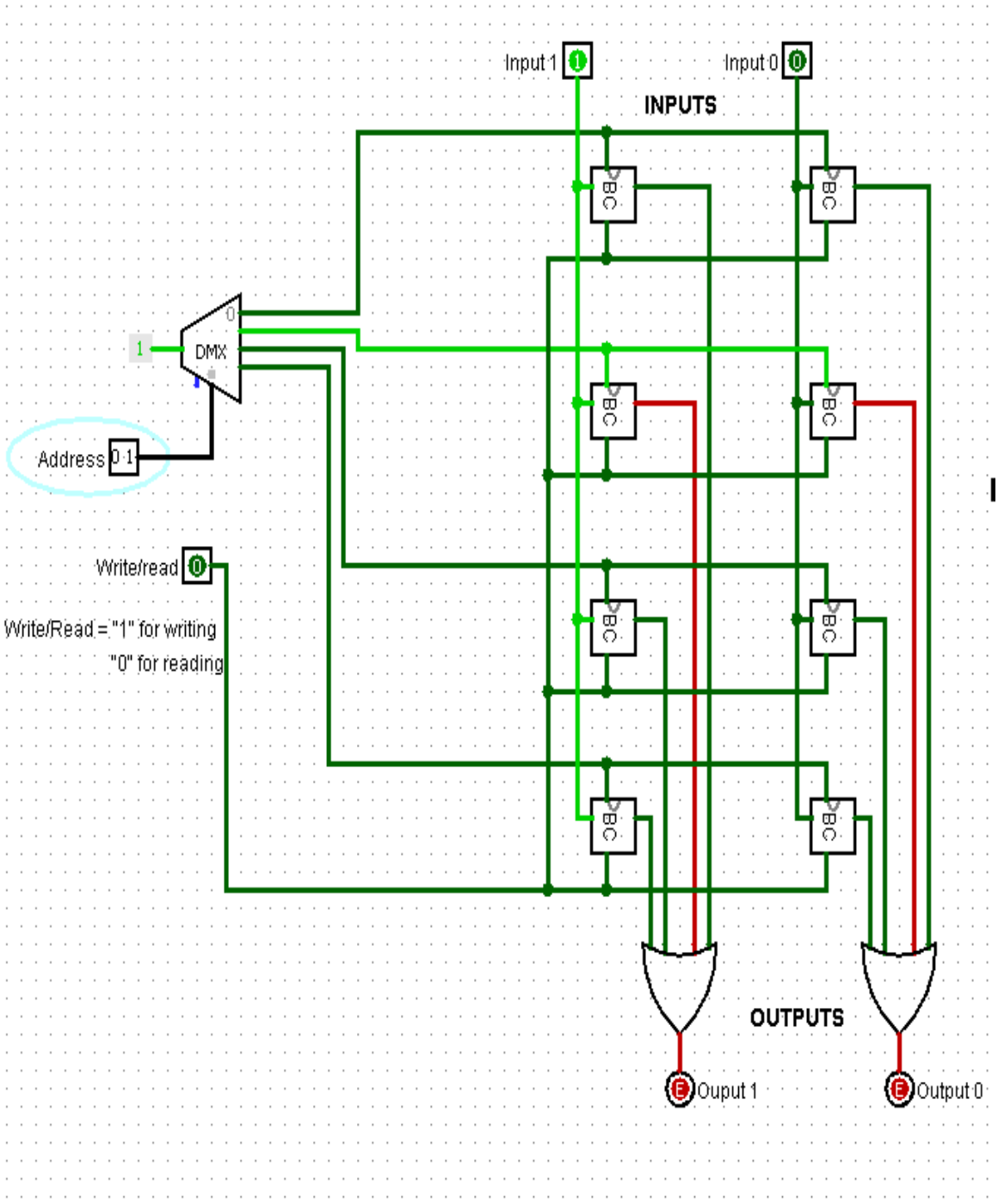


4x2 Memory RAM(SRAM)
Using binary cells(BC)

Example

Step 1:

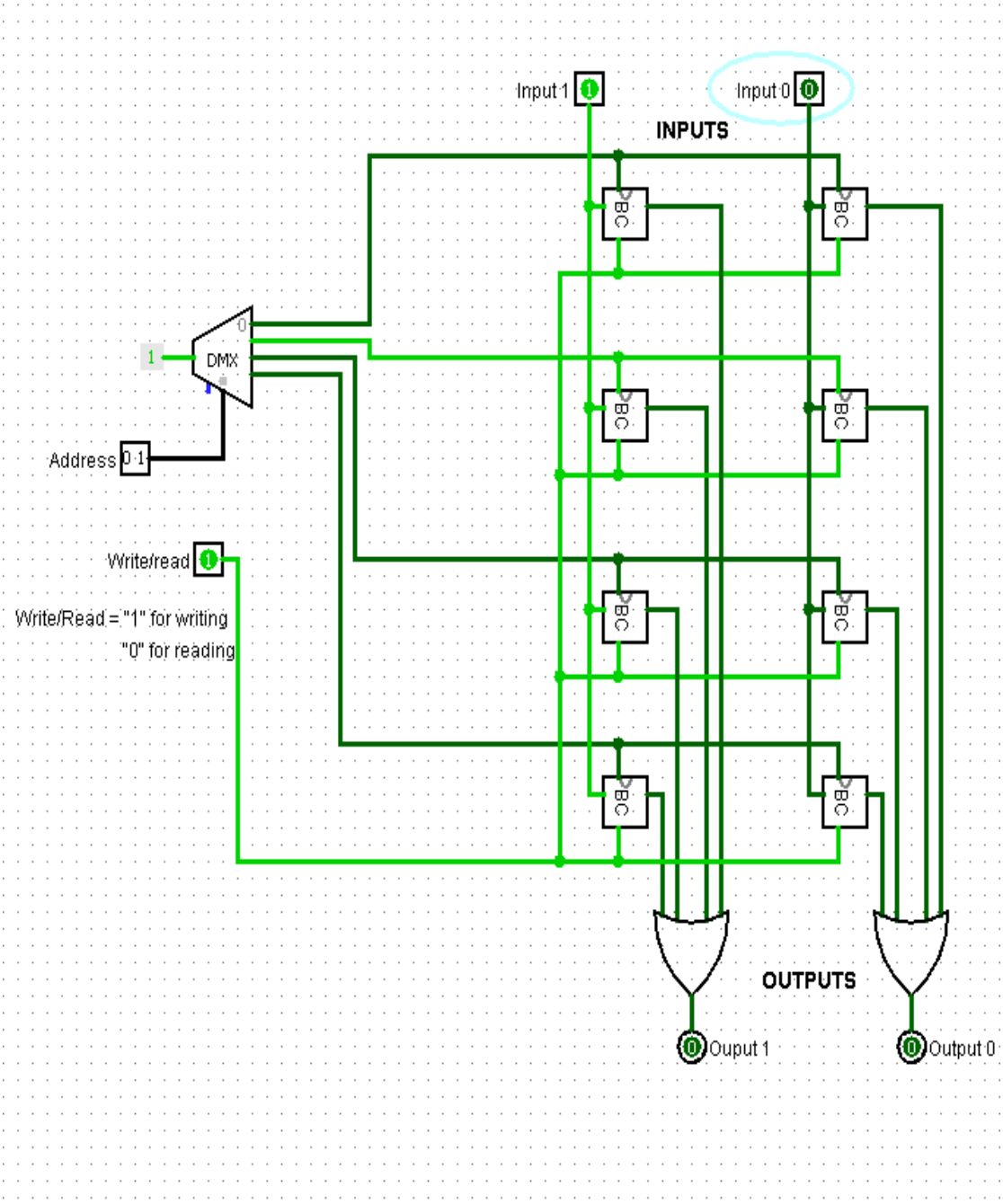
At initial stage there is no value stored at the address [0 1]



We can see as there is no data stored initially so the output shows an error

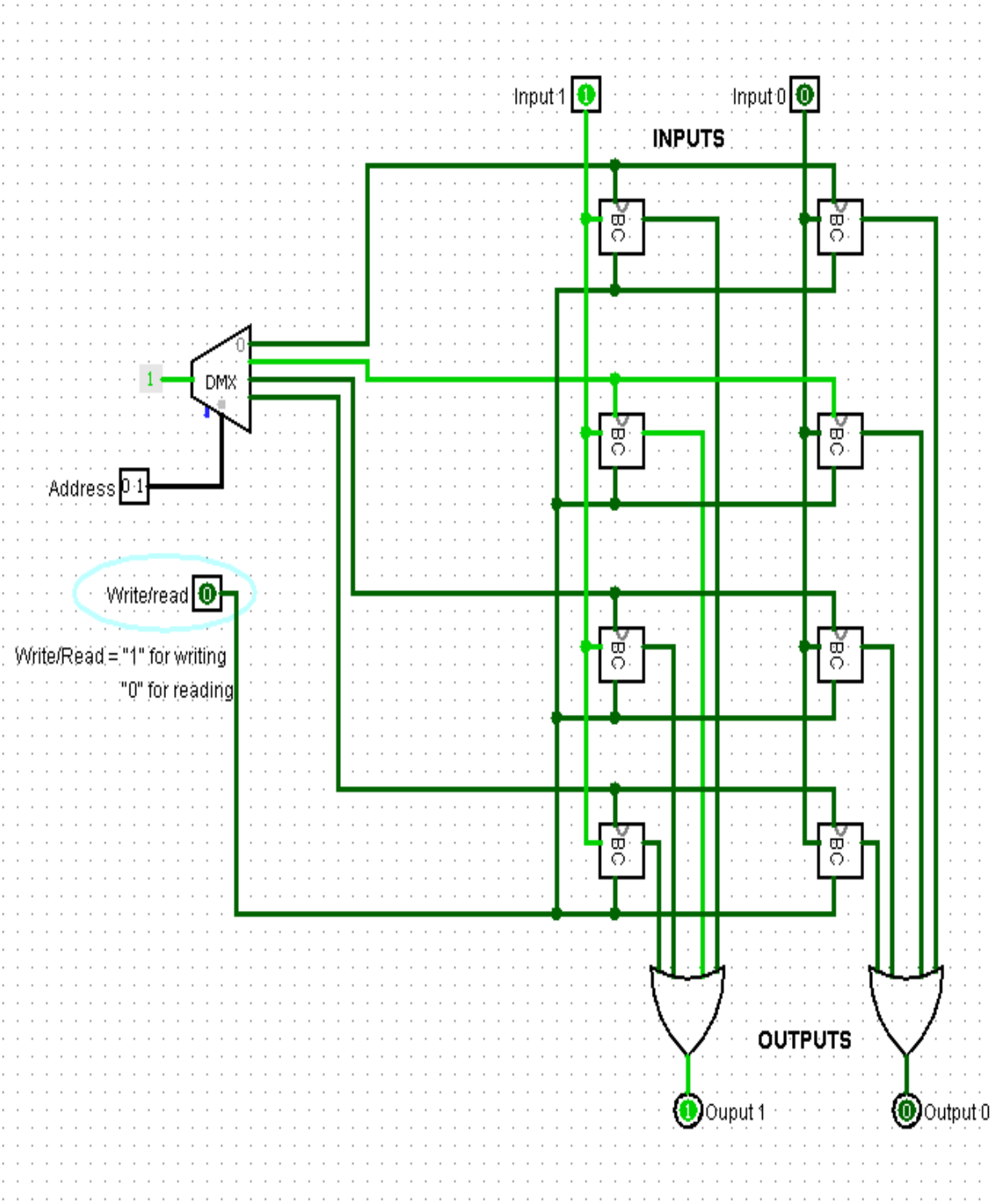
Step 2:

We store the value [1 0] on address [0 1]



Step 3:

Taking the output of the value stored at [0 1]



We can see that the output comes out to be [1 0].

Steps:

1. At first the write port was not enabled so whatever values we try to incorporate in the address they will not be stored.
2. As soon as the write port is enabled and a particular address is selected the bits are stored at the address.
3. Now if we enable the read port for that address, it will output the bits that were stored in it in the second step.

This circuit holds the power to store 4 – 2 bit numbers in it.

The same can be implemented from the logisim file.

_____End Of Assignment_____