| **EXPERIMENT NO.** | **TITLE** | **LAB OUTCOMES** |
| --- | --- | --- |
| **1** | **Study of PC Motherboard Technology (South Bridge and North Bridge). Internal Components and Connections used in Computer System** | **1** |
| **AIM:** | *To understand PC Assembly by studying different components mounted on motherboard. To identify the different Bridges and their role in computer System. To comprehend connections of diverse components* | |

**Name: Shashwat Tripathi**

**Roll No: 58**

**Div: D10A**

**Date of Performance:**

**Shashwat Tripathi**

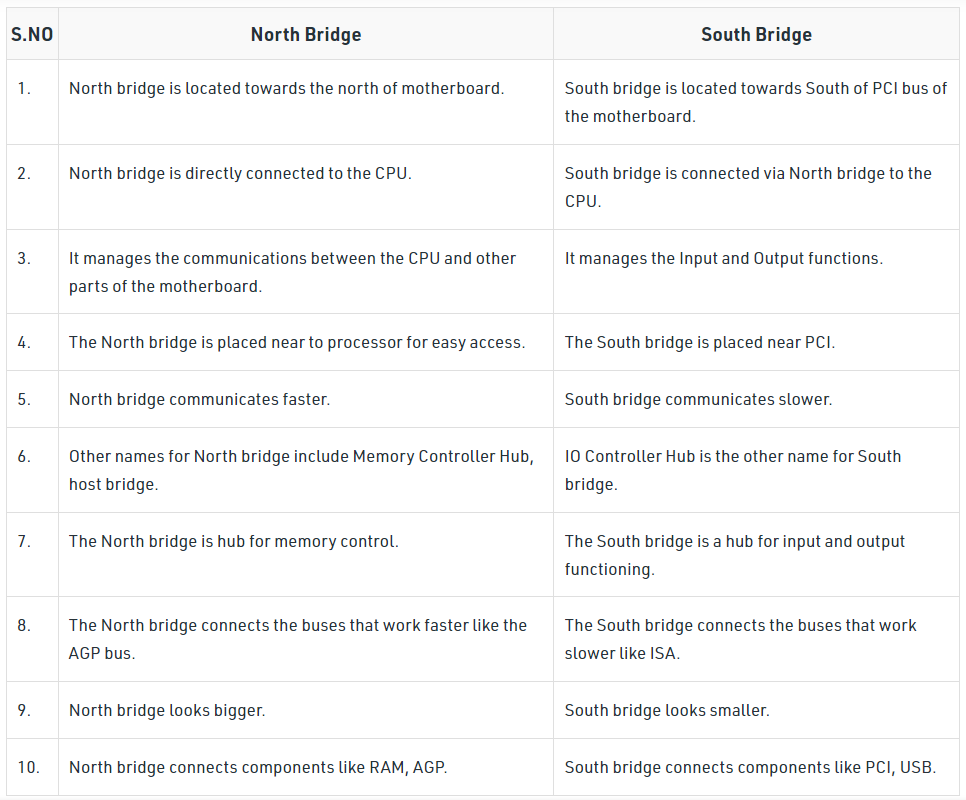
**D10A 58**

**BATCH: C**

**LAB EXPERIMENT NO: 1**

**Q) Differentiate between NorthBridge And SouthBridge.**

Ans:



**Q)** **Components of the motherboard.**

Ans:

1. **CPU (Central Processing Unit) chip**- CPU is the electronic circuitry in a computer that executes instructions that make up a program
2. **RAM (Random Access Memory) slots**- RAM is a kind of computer memory that can be read and written. It is mainly used to save data and machine code
3. **Southbridge/northbridge**- They are the two chips in the core logic chipset on the motherboard. Typically, the southbridge implements the slower capabilities of the motherboard in a northbridge/southbridge chipset computer architecture.
4. **BIOS (Basic Input/Output System)**- BIOS, also called system BIOS, PC BIOS or ROM BIOS, is firmware that is used to perform hardware initialization during the booting process.
5. **I/O port**- Input/output ports are the connections between the CPU and peripheral devices on a motherboard.
6. **USB (Universal Serial Bus)**- USB is an industry standard that creates specifications for connectors, cables and protocols for connection; power supply (interfacing) and communication among computers.
7. **CPU slot**- A CPU slot, also called a CPU socket or Processor socket, contains one or more mechanical components that provide mechanical and electrical connections between the PCB and a microprocessor (CPU).
8. **PCI (Peripheral Component Interconnect) slot**- Peripheral Component Interconnect is a local computer bus for connecting hardware to a computer.
9. **AGP (Accelerated Graphics Port) slot**- AGP was designed as a high-speed point-to-point channel for connecting a video card (graphics card) to a computer system.
10. **ISA (Industry Standard Architecture) slot**- ISA is the 16-bit internal bus of IMB PC/AT and similar computers that are based on Intel 80286 and its immediate successors during the 1980s.
11. **Parallel port**- A parallel port is a kind of interface for attaching peripherals on desktops.
12. **FDC (Floppy-Disk Controller)**- FDC is a special-purpose chip and associated disk controller circuitry. It controls and directs reading from and writing to a computer’s floppy disk drive (FDD).
13. **IDE (Integrated Drive Electronics) controller**- The devices used for connecting IDE, Ethernet, FireWire, USB and other systems can be called host adapter.
14. **CMOS (Complementary Metal-oxide-semiconductor) battery**- CMOS battery, also called memory battery, clock battery or real-time clock (RTC), is generally a CR2032 lithium coin cell.
15. **Power supply connector**- A power supply provides the necessary electrical power to let the computer to work. It takes standard 110-Volt AC (Alternative Current) power to DC (Direct Current) power of 12 Volt, 5 Volt, 3.3 Volt, etc.
16. **Mouse and keyboard ports**- Many PCs use the PS/2-style connectors for both keyboard and mouse; and the connectors are marked clearly for different usage.
17. **DIP (Dual In-line Package) switch**- A DIP switch is a manual electric switch packaged with others in a standard dual in-line package.
18. **Jumper**- A jumper is a short length of conductor that is used to close, open or bypass part of an electronic circuit.
19. **Heat sink/heatsink (cooling system)**- A heat sink is a passive heat exchanger that transfers the heat generated by parts of motherboard into a fluid medium like liquid or air.
20. **Clock generator**- A clock generator is an electronic oscillator (circuit) that produces a clock signal for usage in synchronizing a circuit’s operation.

**Q) Different Types of connections.**

Ans: Computer systems use various types of connections to communicate and transfer data between different components. Some of the most commonly used connections in computer systems are:

**USB:** Universal Serial Bus (USB) is a standard connection used to connect devices like keyboards, mice, printers, external hard drives, and other peripherals to a computer.

**Ethernet:** Ethernet is a wired connection used to connect computers and other devices in a local area network (LAN) or to connect a computer to the internet.

**Wi-Fi:** Wi-Fi is a wireless connection that uses radio waves to connect devices to a network, allowing them to access the internet or communicate with other devices on the network.

**Bluetooth:** Bluetooth is a wireless connection used to connect devices like smartphones, wireless headphones, and other peripherals to a computer or other device.

**HDMI:** High-Definition Multimedia Interface (HDMI) is a connection used to transmit audio and video signals from a computer to an external display, such as a TV or monitor.

**DisplayPort:** DisplayPort is another connection used to transmit audio and video signals from a computer to an external display.

**Thunderbolt:** Thunderbolt is a high-speed connection used to connect external devices to a computer, such as external hard drives, monitors, and other peripherals.

**SATA:** Serial Advanced Technology Attachment (SATA) is a connection used to connect hard drives and other storage devices to a computer.

**PCIe:** Peripheral Component Interconnect Express (PCIe) is a connection used to connect expansion cards, such as graphics cards and network interface cards, to a computer.

| **EXPERIMENT NO.** | **TITLE** | **LAB OUTCOMES** |
| --- | --- | --- |
| **2** | 1**. Verify the truth table of various logic gates**  **(basic and universal gates)**  **2. Realize Half adder and Full adder**  **3. Implementation of MUX and DeMUX** | **3** |
| **AIM:** | *To understand the implementation of combinational circuits* | |

**Name: Shashwat Tripathi**

**Roll No: 58**

**Div: D10A**

**Date of Performance:**

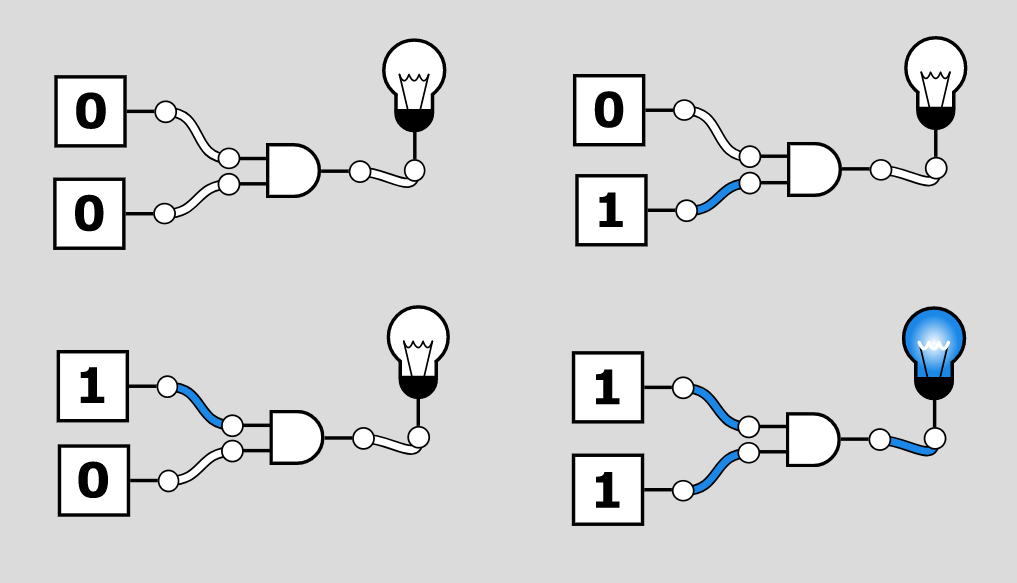
**Shashwat Tripathi**

**D10A 58**

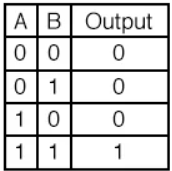
**BATCH: C**

**LAB EXPERIMENT NO: 2**

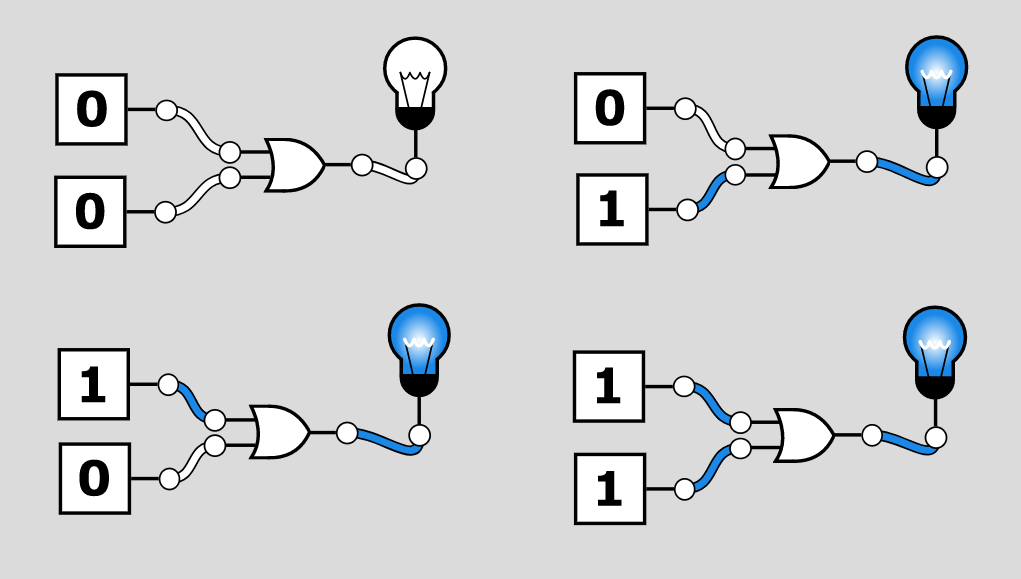
**AND GATE:** The output state of the AND gate will always be low when any of the input states is low. Simply, if any input value in the AND gate is set to 0, then it will always return low output(0). The logic or Boolean expression for the AND gate is the logical multiplication of inputs denoted by a full stop or a single dot as A.B=Y



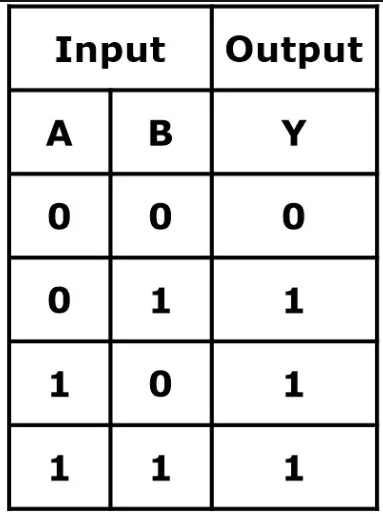
**TRUTH TABLE-**



**OR GATE:** The output state of the OR gate will always be low when both of the inputs states is low. Simply, if any input value in the OR gate is set to 1, then it will always return high-level output(1).The logic or Boolean expression for the OR gate is the logical addition of inputs denoted by plus sign(+) as A+B=Y

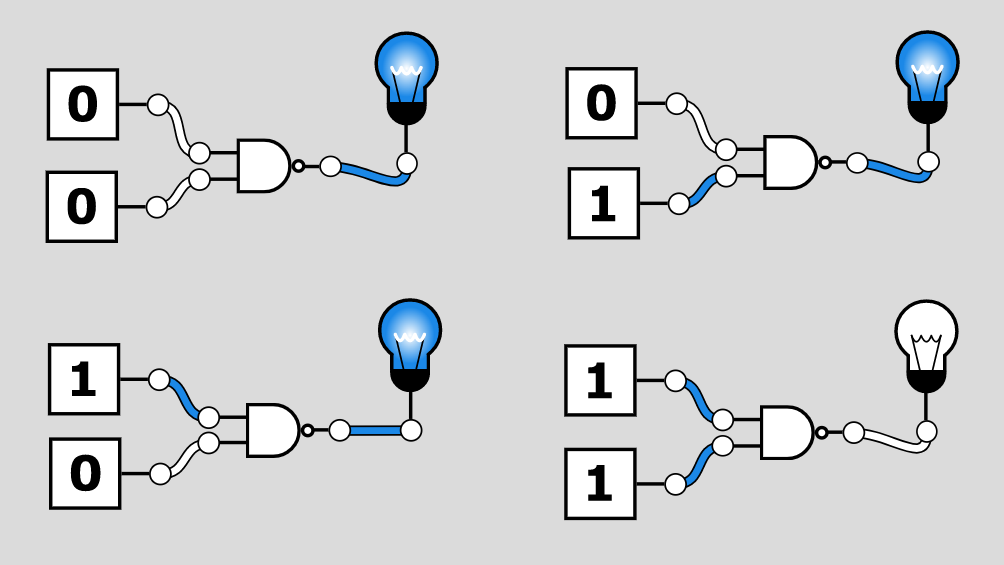


**TRUTH TABLE-**

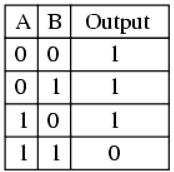
****

**NAND GATE:** The NAND gate is the universal gate. It means all the basic gates such as AND, OR, and NOT gate can be constructed using a NAND gate. The NAND gate is the combination of the NOT-AND gate. The output state of the NAND gate will be low only when all the inputs are high.The logic or Boolean expression for the NAND gate is the complement of logical multiplication of inputs denoted by a full stop or a single dot as

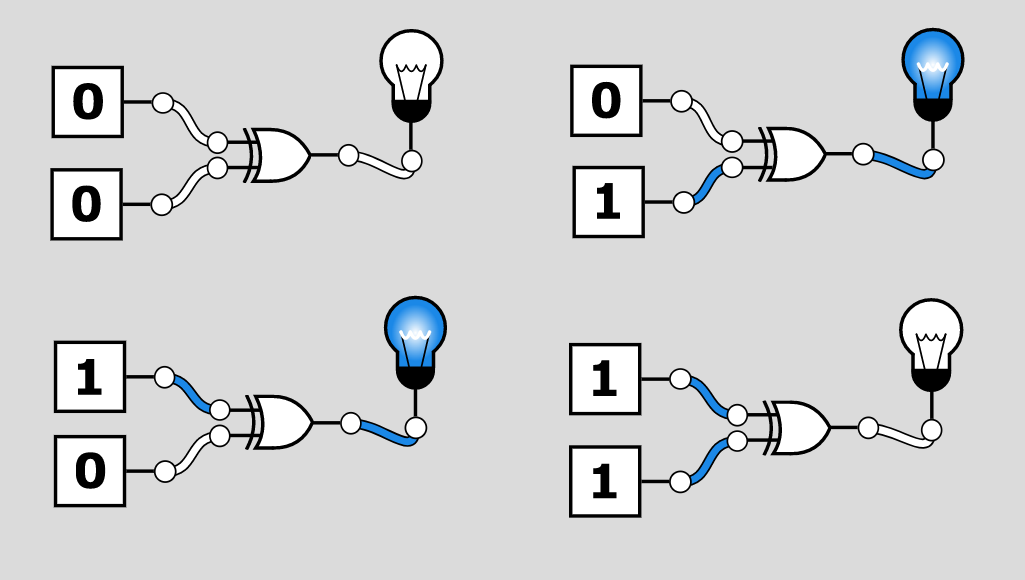
(A.B)'=Y



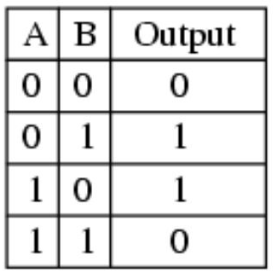
**TRUTH TABLE-**

****

**XOR GATE-** The Exclusive-OR gate or XOR gate is achieved by combining standard logic gates together. XOR gate is used extensively in error detection circuits, computational logic comparators and arithmetic logic circuits. The Exclusive OR gate gives an output only if its two inputs are dissimilar, namely if one of them is high (one) and the other is low (zero).

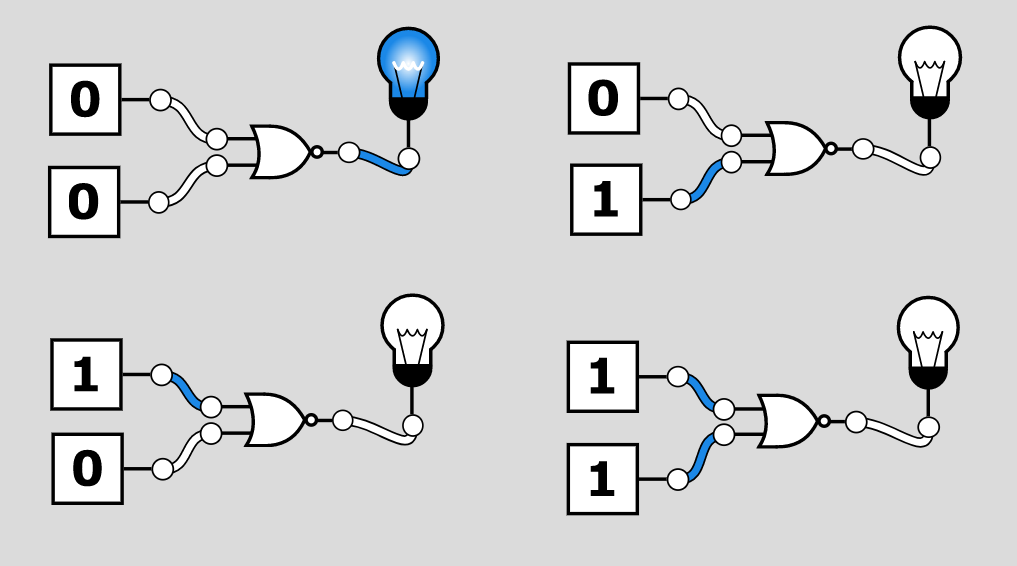


**TRUTH TABLE-**

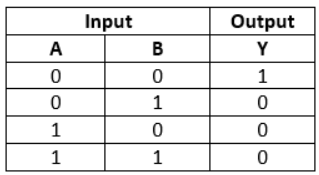


**NOR GATE:** The NOR gate is also a universal gate. So, we can also form all the basic gates using the NOR gate. The NOR gate is the combination of the NOT-OR gate. The output state of the NOR gate will be high only when all of the inputs are low. Simply, this gate returns the complement result of the OR gate.

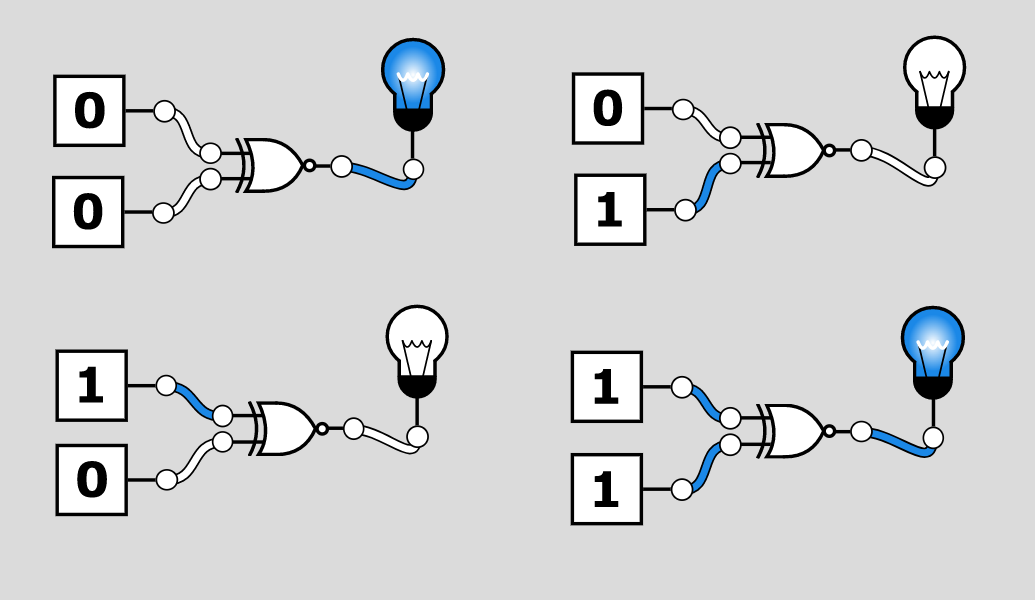
The logical or Boolean expression for the NOR gate is the complement of logical multiplication of inputs denoted by the plus sign as (A+B)'=Y



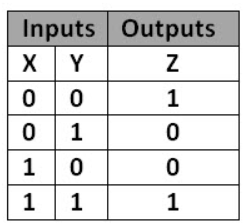
**TRUTH TABLE-**

****

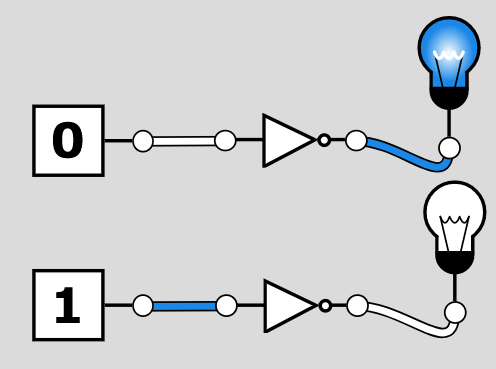
**XNOR GATE:** The XNOR gate is the complement of the XOR gate. It is a hybrid gate. Simply, it is the combination of the XOR gate and NOT gate. The output level of the XNOR gate is high only when both of its inputs are the same, either 0 or 1. The symbol of the XNOR gate is the same as XOR, only complement sign is added.



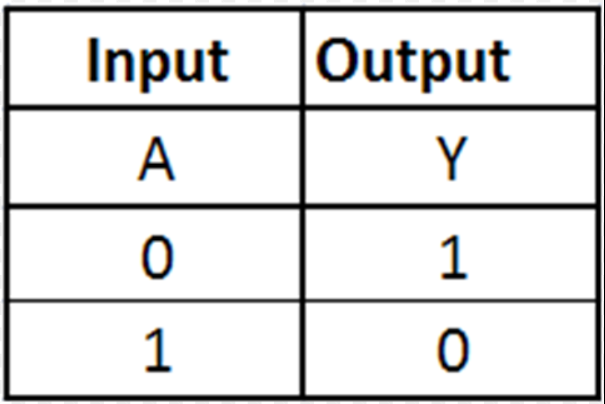
**TRUTH TABLE-**

****

**NOT GATE:** The NOT gate is the most basic logic gate of all other logic gates. NOT gate is also known as an inverter or an inverting Buffer. NOT gate only has one input and one output. When the input signal is "Low", the output signal is "High" and when the input signal is "High", the output is "Low". The Boolean expression for the NOT gate is as follows: A'=Y



**TRUTH TABLE-**

****

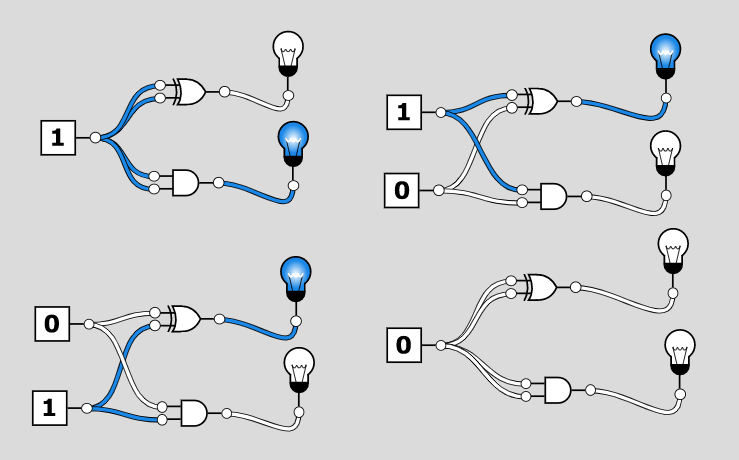
**HALF-ADDER**

The Half-Adder is a basic building block of adding two numbers as two inputs and produce out two outputs. The adder is used to perform OR operation of two single bit binary numbers. The augent and addent bits are two input states, and 'carry' and 'sum 'are two output states of the half adder.

The SOP form of the sum and carry are as follows:

Sum = x'y+xy'

Carry = xy



**TRUTH TABLE**

| **A** | **B** | **SUM** | **CARRY** |
| --- | --- | --- | --- |
| 0 | 0 | 0 | 0 |
| 1 | 0 | 1 | 0 |
| 0 | 1 | 1 | 0 |
| 1 | 1 | 0 | 1 |

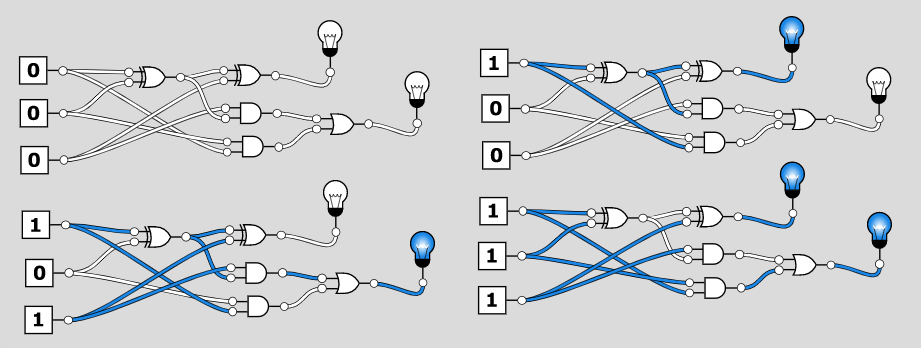
**FULL-ADDER**

The half adder is used to add only two numbers. To overcome this problem, the full adder was developed. The full adder is used to add three 1-bit binary numbers A, B, and carry C. The full adder has three input states and two output states i.e., sum and carry.

The SOP form can be obtained with the help of K-map as:

Sum = x' y' z+x' yz+xy' z'+xyz

Carry = xy+xz+yz



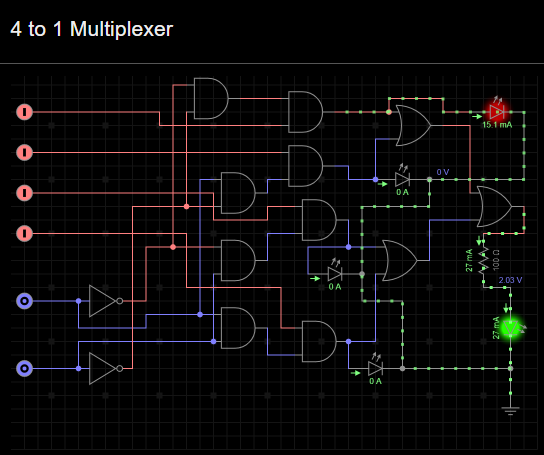
**TRUTH TABLE**

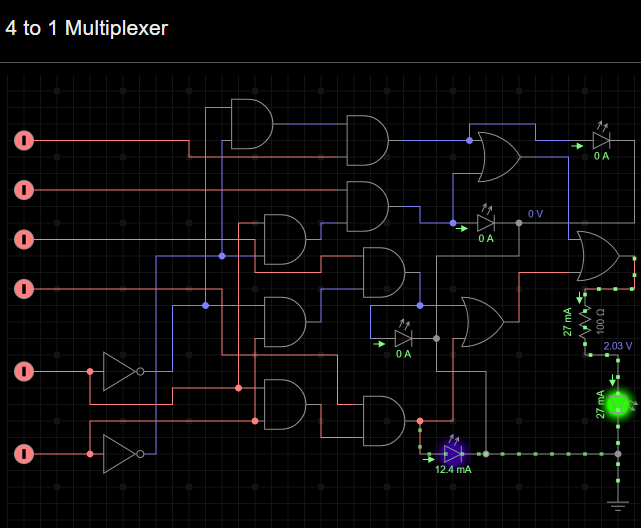
| **A** | **B** | **C** | **SUM** | **CARRY** |
| --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 1 | 0 |
| 0 | 1 | 0 | 1 | 0 |
| 0 | 1 | 1 | 0 | 1 |
| 1 | 0 | 0 | 1 | 0 |
| 1 | 0 | 1 | 0 | 1 |
| 1 | 1 | 0 | 0 | 1 |
| 1 | 1 | 1 | 1 | 1 |

**MULTIPLEXER**

A digital logic circuit which is capable of accepting several inputs and generating a single output is known as multiplexer or MUX. Thus, the multiplexer is a type of data selector which takes many inputs and gives a selected output. In a multiplexer, there are 2n input lines and 1 output line, where n is the number of select lines.

Therefore, a multiplexer is a combinational circuit which is designed to switch one of the many input lines to a single output line by the use of a control signal. For this reason, the multiplexer is also referred to as a many to one circuit.

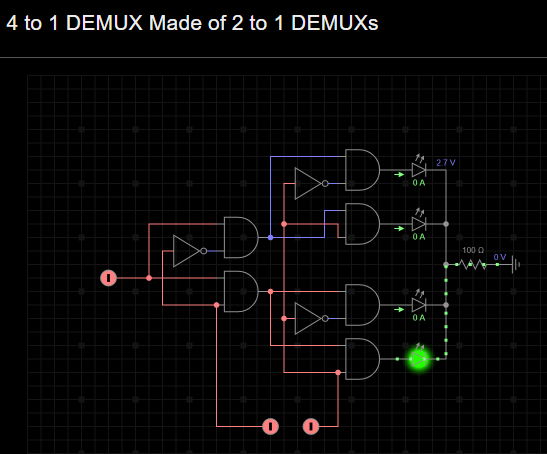


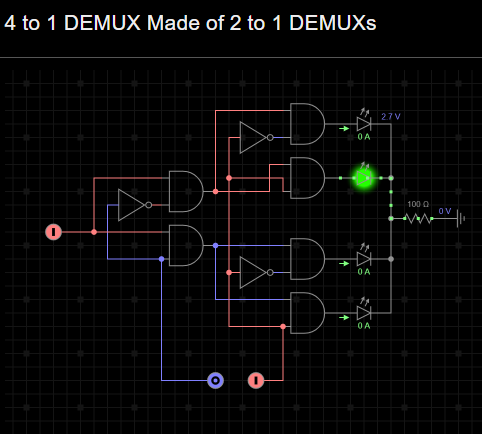
****

**DEMULTIPLEXER**

A digital combinational circuit which takes one input signal and generates multiple output signals is known as demultiplexer or DEMUX. As it distributes a single input signal over many output lines, hence it is also referred to as a type of data distributor.

In a demultiplexer, there is only 1 input line and 2n output lines. Where, n denotes the number of select lines. Therefore, it can be noted that a demultiplexer reverses the operation of a multiplexer.

****

****

| **EXPERIMENT NO.** | **TITLE** | **LAB OUTCOMES** |
| --- | --- | --- |
| **3** | **Program for 16 bit BCD addition** | **3** |
| **AIM:** | *To understand the Arithmetic and logical operations in 8086 Assembly language Programming* | |

**Name: Shashwat Tripathi**

**Roll No: 58**

**Div: D10A**

**Date of Performance:**

**Shashwat Tripathi**

**D10A 58**

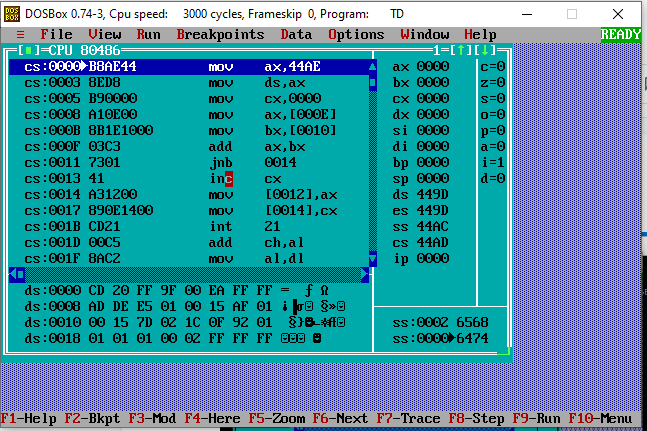
**BATCH: C**

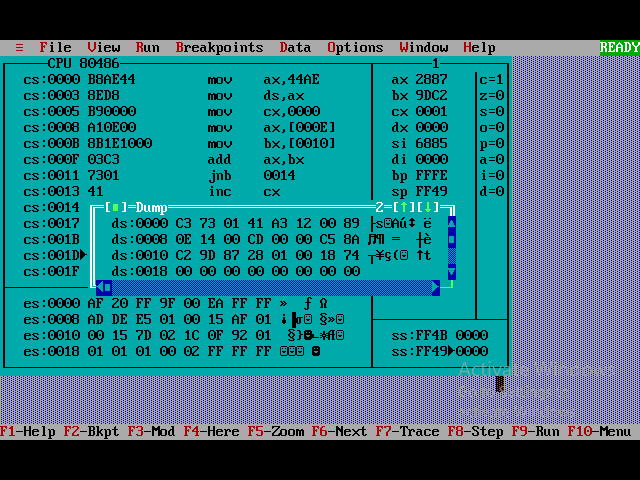
**LAB EXPERIMENT NO: 3**

**CODE-**



**OUTPUT-**





| **EXPERIMENT NO.** | **TITLE** | **LAB OUTCOMES** |
| --- | --- | --- |
| **4** | **1.Program to count number of 1’s and 0’s in**  **a given 8 bit number**  **2.Program to find even and odd numbers**  **from a given list** | **4** |
| **AIM:** | *To understand the Loop Operations in 8086 Assembly language Programming* | |

**Name: Shashwat Tripathi**

**Roll No: 58**

**Div: D10A**

**Date of Performance:**

**Shashwat Tripathi**

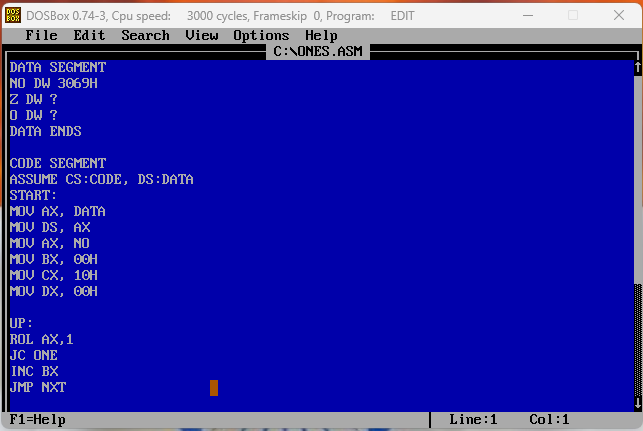
**D10A 58**

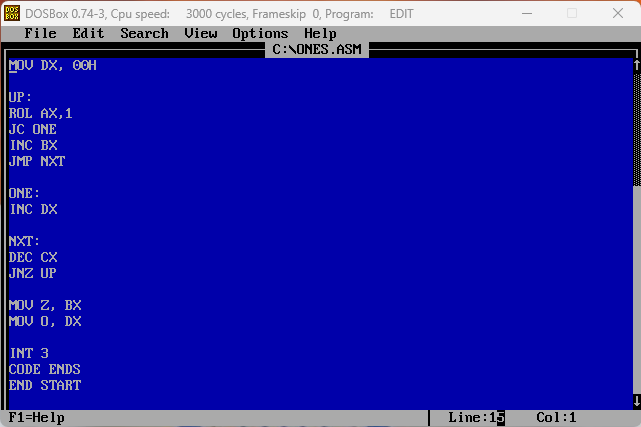
**BATCH: C**

**LAB EXPERIMENT NO: 4**

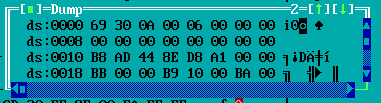
1. **To count no. of zeroes and ones of a hexadecimal number.**

CODE:





OUTPUT:



1. **Find even and odd numbers from a given list**

.model small

.stack 100h

.data

ev db 'Even$'

od db 'Odd$'

.code

main proc

mov ax,@data

mov ds,ax

mov ah,1

int 21h

mov bl,2

div bl

cmp ah,0

je IsEven

mov dx,10

mov ah,2

int 21h

mov dx,13

mov ah,2

int 21h

mov dx,offset od

mov ah,9

int 21h

mov ah,4ch

int 21h

IsEven:

mov dx,10

mov ah,2

int 21h

mov dx,13

mov ah,2

int 21h

mov dx,offset ev

mov ah,9

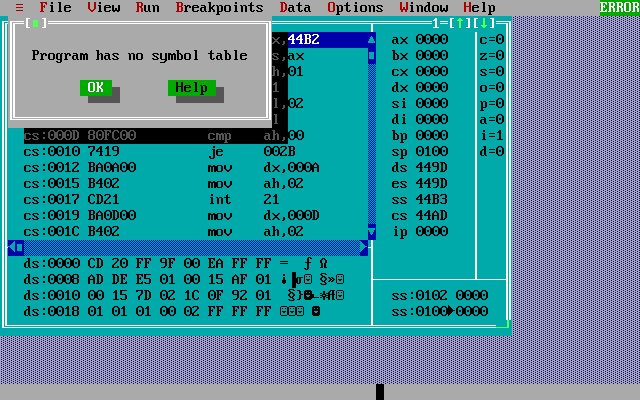
int 21h

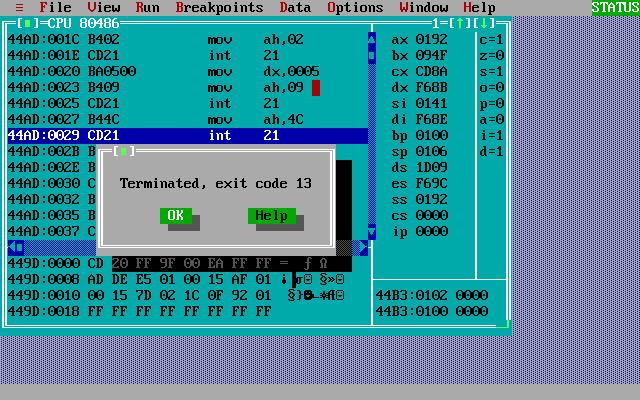
mov ah,4ch

int 21h

main endp

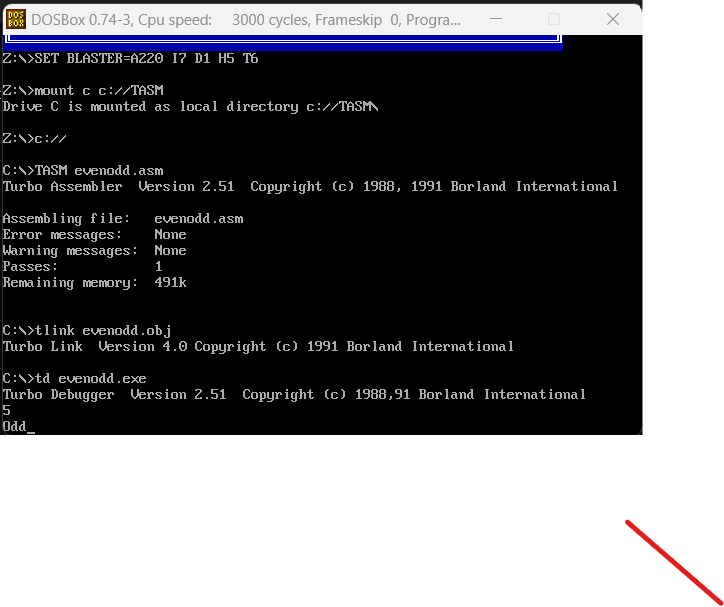
end main



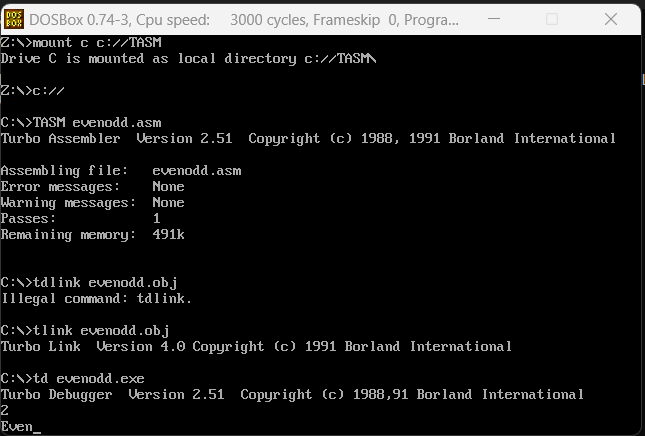


**OUTPUT:**

i) For Odd Number



ii) For Even Number-



| **EXPERIMENT NO.** | **TITLE** | **LAB OUTCOMES** |
| --- | --- | --- |
| **5** | **Check whether a given string is a**  **Palindrome or not.** | **5** |
| **AIM:** | *To understand String and Procedure in 8086 Assembly language Programming* | |

**Name: Shashwat Tripathi**

**Roll No: 58**

**Div: D10A**

**Date of Performance:**

**Shashwat Tripathi**

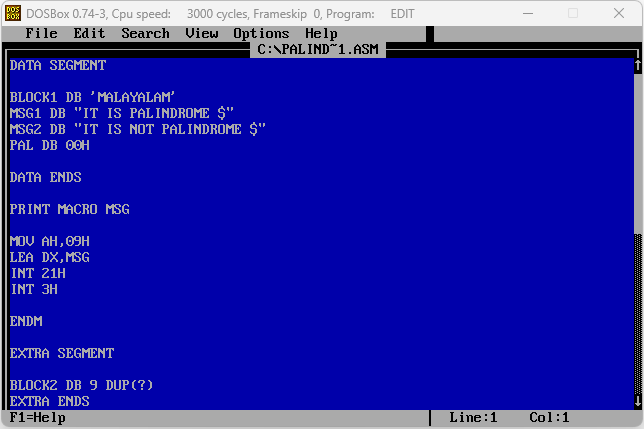
**D10A 58**

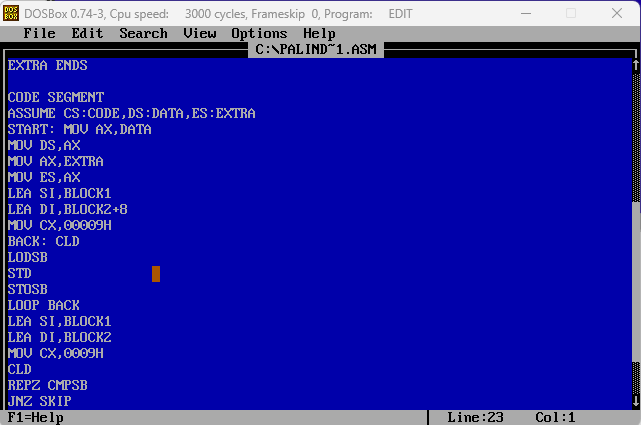
**BATCH: C**

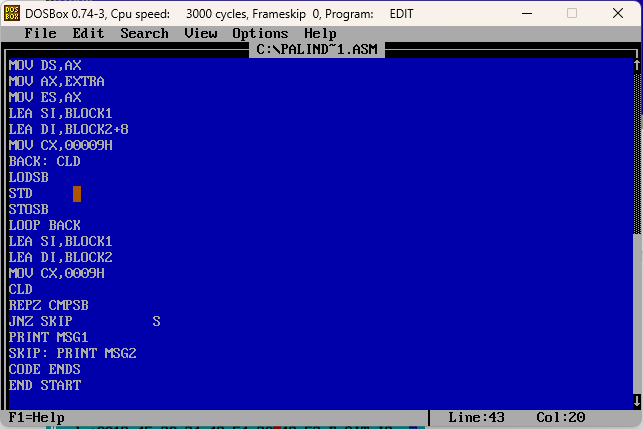
**LAB EXPERIMENT NO: 5**

**To check if a string is palindrome or not**

**Code:**



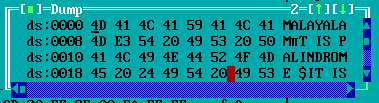




**Input1:**

**”MALAYALAM”**

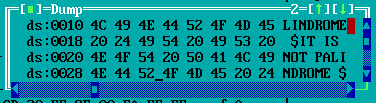
**Output1:**



**Input2:**

**”SHASHWAT”**

**Output2:**

****