**LIST OF EXPERIMENTS**

**Course Name :** Computer Organization and Architecture

**Course Code :** 18CS203J

|  |  |  |
| --- | --- | --- |
| **Experiment No.** | **Title of the Experiment** | **Simulators** |
| 1 | Exploring Components of Virtual Desktop | Virtual Desktop Simulator |
| 2 | Assembling and Disassembling of Virtual Desktop |
| 3 | 8086 Simulator [8-bit] (Immediate, Direct Addressing Mode) | 8086 Simulator |
| 4 | 16-bit Addition and Subtraction |
| 5 | Multiplication of 8-bit numbers and Factorial |
| 6 | Half Adder, Full Adder | Logic Gate Simulator |
| 7 | Ripple Carry Adder |
| 8 | Carry Look Ahead Adder |
| 9 | Design of Array Multiplier(2-bit multiplier) |
| 10 | Study of Booth Algorithm and Design | Virtual Lab |
| 11 | Study of Carry Save Addition-Program to carry out carry save |
| 12 | Design of processing unit |
| 13 | Design of Basic Pipeline |

**8086 MICROPROCESSOR**

1. **ADDITION OF TWO 8 BIT NUMBERS**

**AIM**: To implement assembly language program to find sum of two 8 bit numbers

**PROGRAM:**

DATA SEGMENT

N1 DB 44H

N2 DB 44H

RES DW ?

DATA ENDS

CODE SEGMENT

ASSUME CS:CODE, DS:DATA

START:

MOV AX,DATA

MOV DS,AX

MOV AL,N1

MOV BL,N2

ADD AL,BL

MOV RES,AL

INT 21H

CODE ENDS

END START

**OUTPUT:**

RES: 88H

**IMMEDIATE MODE**

DATA SEGMENT

RES DW ?

DATA ENDS

CODE SEGMENT

ASSUME CS:CODE, DS:DATA

START:

MOV AX,DATA

MOV DS,AX

MOV AL,03H

MOV BL,02H

ADD AL,BL

MOV RES,AL

INT 21H

CODE ENDS

END START

**OUTPUT:**

RES: 05H

**DIRECT MODE**

DATA SEGMENT

RES DW ?

DATA ENDS

CODE SEGMENT

ASSUME CS:CODE, DS:DATA

START:

MOV AX,DATA

MOV DS,AX

MOV AL,[0001]

MOV BL,[0002]

ADD AL,BL

MOV RES,AL

INT 21H

CODE ENDS

END START

**INPUT:**

[0001] : 03H

[0002] : 02H

**OUTPUT:**

RES: 05H

1. **ADDITION AND SUBTRACTION OF TWO 16 BIT NUMBERS**

**AIM**: To implement assembly language program to find ADDITION and SUBTRACTION of two 16 bit numbers

**ADDITION**

**PROGRAM:**

DATA SEGMENT

N1 DW 4444H

N2 DW 4444H

RES DW ?

DATA ENDS

CODE SEGMENT

ASSUME CS:CODE, DS:DATA

START:

MOV AX,DATA

MOV DS,AX

MOV AX,N1

MOV BX,N2

ADD AX,BX

MOV RES,AX

INT 21H

CODE ENDS

END START

**OUTPUT:**

RES: 8888H

**SUBTRACTION**

**PROGRAM:**

DATA SEGMENT

N1 DW 4444H

N2 DW 4444H

RES DW ?

DATA ENDS

CODE SEGMENT

ASSUME CS:CODE, DS:DATA

START:

MOV AX,DATA

MOV DS,AX

MOV AX,N1

MOV BX,N2

SUB AX,BX

MOV RES,AX

INT 21H

CODE ENDS

END START

**OUTPUT:**

RES: 0000H

1. **MULTIPLICATION OF TWO 8 BIT NUMBERS**

**AIM**: To implement assembly language program to find PRODUCT of two 8 bit numbers.

**PROGRAM:**

DATA SEGMENT

N1 DB 02H

N2 DB 03H

RES DW ?

DATA ENDS

CODE SEGMENT

ASSUME CS:CODE, DS:DATA

START:

MOV AX, DATA

MOV DS, AX

MOV AL, N1

MOV BL, N2

MUL BL

MOV RES, AX

INT 21H

CODE ENDS

END START

**OUTPUT:**

RES: 0006H

1. **FACTORIAL OF A GIVEN NUMBER**

**AIM**: To implement assembly language program to find factorial of a given number

**Program:**

DATA SEGMENT

X DW 05H

FACT DW ?

DATA ENDS

CODE SEGMENT

ASSUME CS: CODE, DS: DATA

START:

MOV AX, DATA

MOV DS, AX

MOV AX, 01H

MOV CX, X

UP:

MUL CX

LOOP UP

MOV FACT, AX

MOV AH, 4CH

INT 21H

CODE ENDS

END START

**INPUT:**

06H

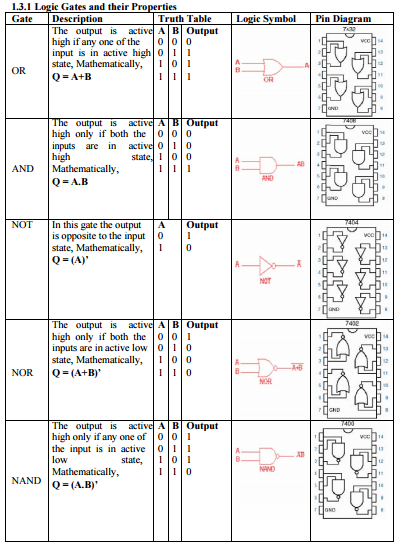
**OUTPUT:**

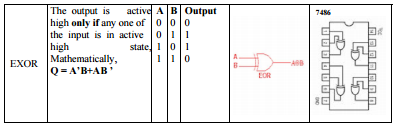
78H

**BASIC LOGIC GATES**

**Aim**

To familiarize with circuit implementations using ICs and test the behavior of different logic gates





**Questions**

1. Implement the basic gates using Universal gates.
2. Implement NOR using NAND gates and NAND gate using NOR gates.
3. What type of logic gate does this logic circuit configuration represent?

****

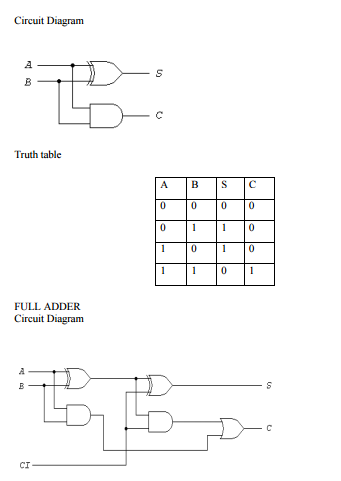
1. NAND Gate
2. EXOR Gate
3. NOR Gate
4. EXNOR Gate
5. **Implement the circuits and truth table for the following expressions**
   1. **A(B+C)**
   2. **AB+BC(B+C)**
   3. **A+A’B**

**HALF ADDER AND FULL ADDER**

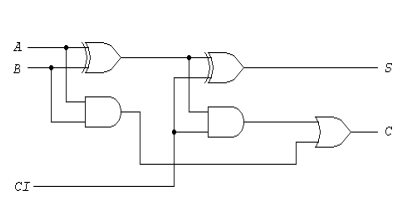
**AIM**

To design and verify the truth table for half adder & full adder.

**HALF ADDER**

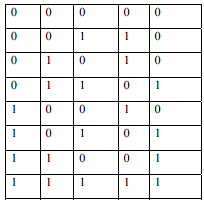


**FULL ADDER**



**TRUTH TABLE**





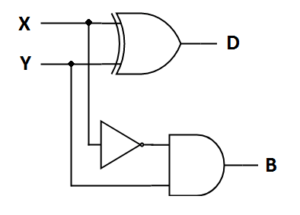
**HALF SUBTRACTOR AND FULL SUBTRACTOR**

**AIM:**

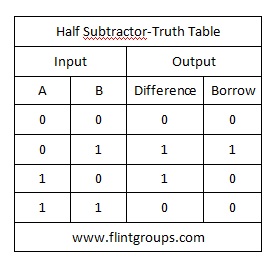
To design and verify the truth table for half subtractor & full subtractor

**HALF SUBTRACTOR**

**Circuit Diagram**

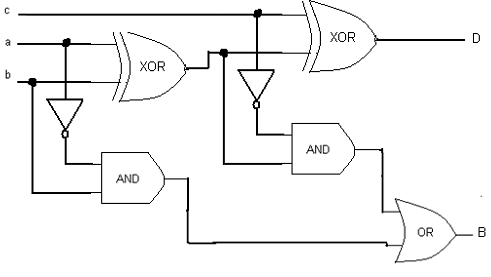


**TRUTH TABLE:**

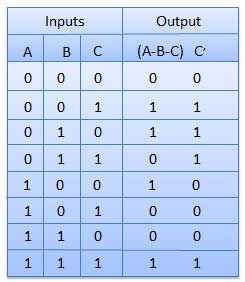


**FULL SUBTRACTOR:**

**Circuit diagram**

****

**TRUTH TABLE**

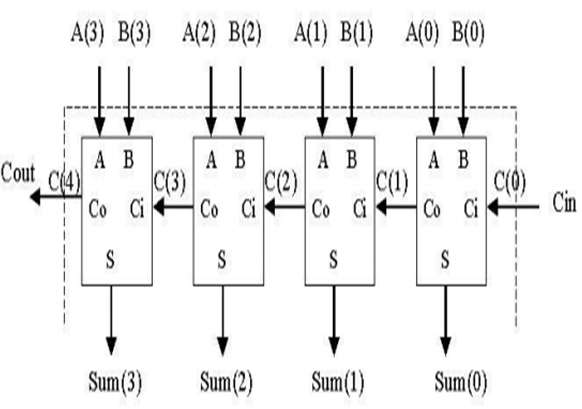


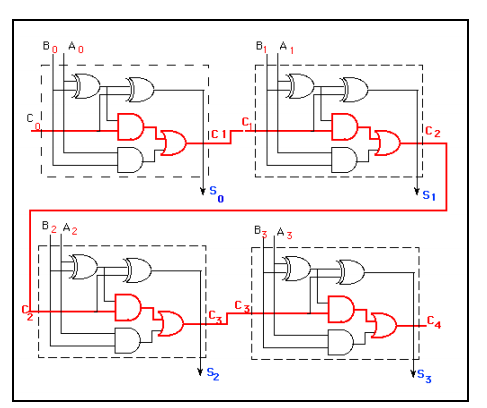
**RIPPLE CARRY ADDER**

**AIM :**

Implement ripple carry adder

**Circuit diagram:**



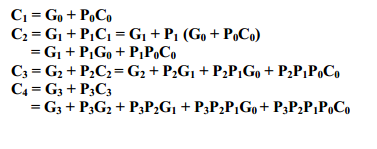


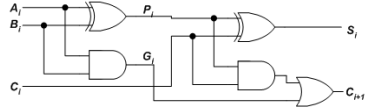
**CARRY LOOKAHEAD ADDER**

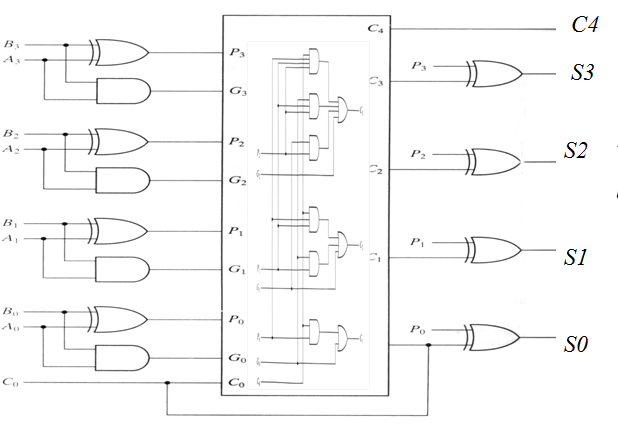
**AIM:**

To implement carry look ahead adder

**CIRCUIT DIAGRAM**

****



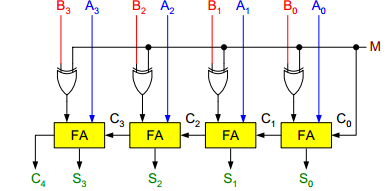


**BINARY PARALLEL ADDER AND SUBTRACTOR**

**AIM:**

**To implement binary adder and subtractor**

**Circuit diagram**

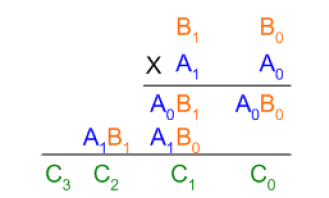


**BINARY MULTIPLIER**

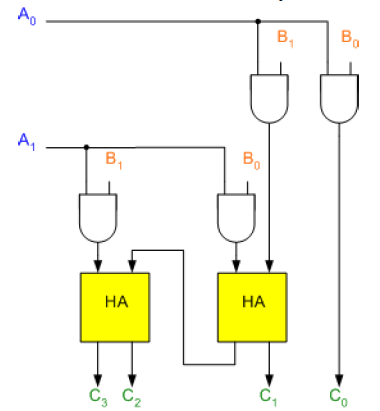
**AIM**

To implement binary multiplier.

**EXAMPLE:**



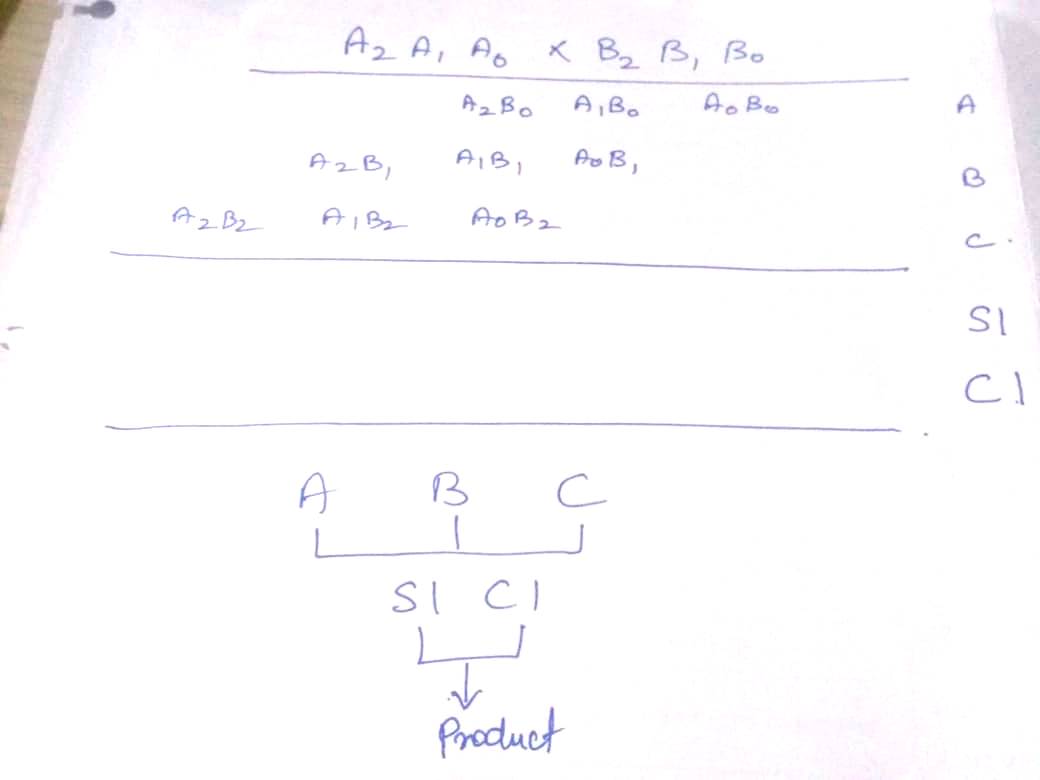
**Circuit Diagram**

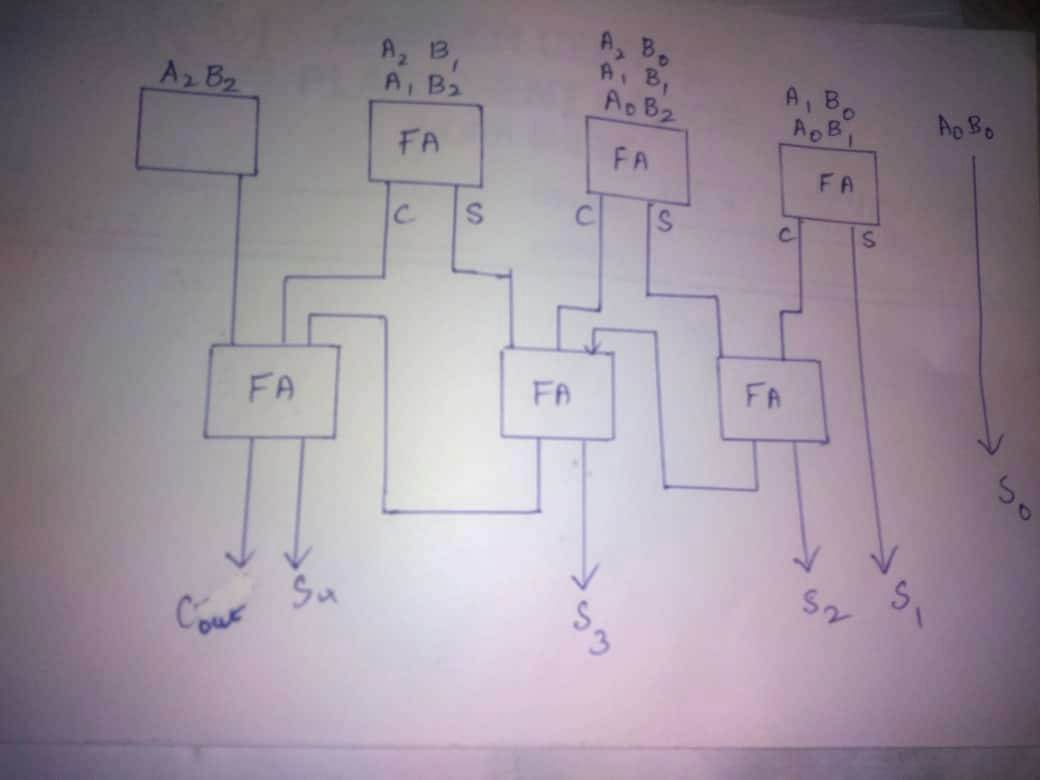


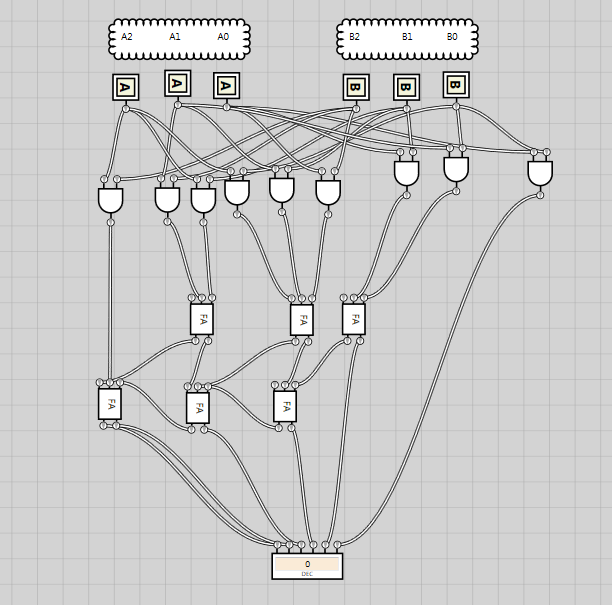
**CARRY SAVE MULTIPLIER**

**AIM:**

**To implement Carry Save Multiplier**

****

****

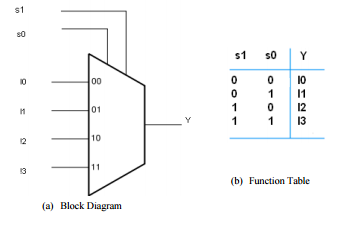
****

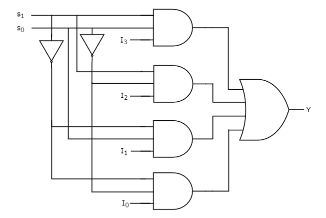
**MULTIPLEXER**

**AIM:**

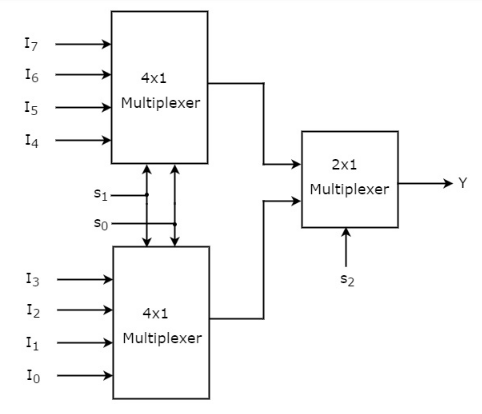
To implement 4:1 multiplexer, 8:1 and 16:1 Multiplexer

**Circuit diagram**

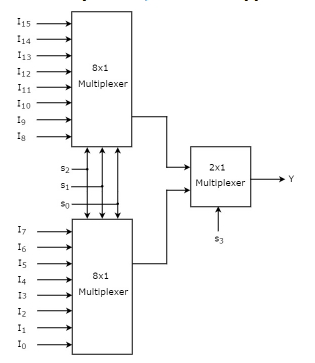




**4:1 Multiplexer**



**8:1 Multiplexer using 4:1 Multiplexer**

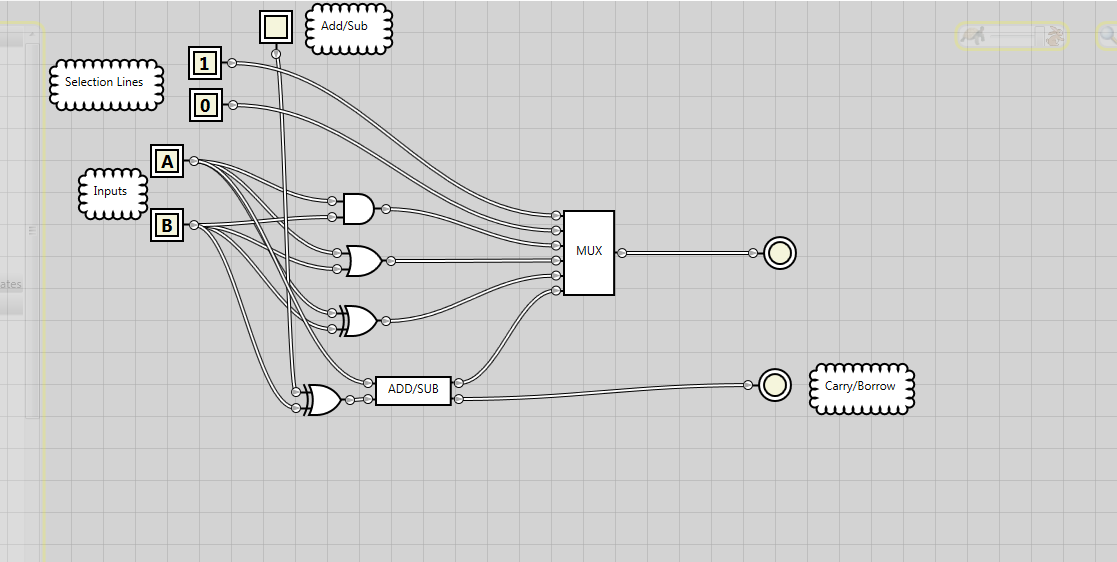


**16:1 Multiplexer using 8:1 Multiplexer**

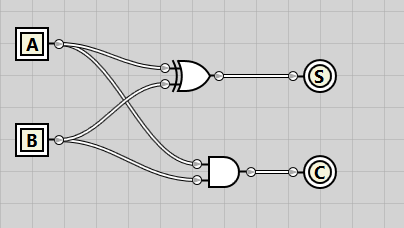
**ARITHMETIC LOGIC UNIT**

**Aim :**

To implement Arithmetic and Logical Unit which perform **Addition and subtraction arithmetic operations** and **AND, OR, XOR logical operations**.



ADD/SUB

****