Experiment No. 3

Title: To Design and implement different arithmetic circuits using Vlabs.

Batch:B3 **Roll No.: 16010422185** **Experiment No.: 3**

Aim: To Design and implement different arithmetic circuits using Vlabs.

Resources needed: Simulation Platform(VLabs)

Theory:

Pre-Test:

Design and Implementation of Different Arithmetic Circuits

2s complement of 11011000
- a: 00101000
- b: 01101000
- c: 11001000
- d: 10101010

Binary number representation by decimal number 37
- a: 101010
- b: 111000
- c: 100101
- d: 101101

Decimal number representation by the binary number 1111 1111
- a: 247
- b: 249
- c: 251
- d: 255

Ones complement of 10100111
- a: 01100110
- b: 01011000
- c: 11100111
- d: 11100101

Submit Quiz

4 out of 4

Post-test:

Add binary numbers 1010 and 0001
- a: 1100
- b: 1111
- c: 1011
- d: 1110

Add binary numbers 1011 and 1101
- a: 11011
- b: 11000
- c: 11111
- d: 11010

Subtract decimal numbers 73 and 46 using 2s complement arithmetic
- a: 10101111
- b: 01011111
- c: 01101111
- d: 00110011

Subtract 1010 from 1111 using 2s complement
- a: 0101
- b: 1010
- c: 1101
- d: 1011

Use 2s complement arithmetic to solve expression given in decimal numbers : 58-33+4
- a: 01000100
- b: 01110101
- c: 01011111
- d: 01011101

Submit Quiz

5 out of 5

Designing Half Adder:

BooleanLogicTable

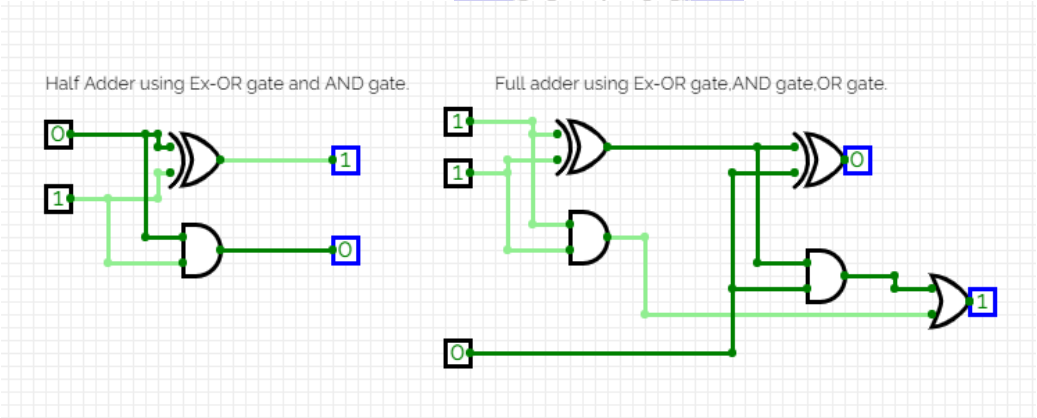
A	B	C	S
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

Generate Circuit Print Truth Table

BooleanLogicTable

x	y	z	C	S
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

Generate Circuit Print Truth Table



1. Binary Adder - Subtractor

Digital computers perform variety of information tasks. Among the functions encountered are the various arithmetic operations. The most basic arithmetic operation is the addition of two binary digits. This simple addition consists of four possible elementary operations. 0 + 0 = 0, 0 + 1 = 1, 1 + 0 = 1, 1 + 1 = 10. The first three operations produce sum of one digit, but when the both augends and addend bits are equal 1, the binary sum consists of two digits. The

higher significant bit of the result is called carry .When the augend and addend number contains more significant digits, the carry obtained from the addition of the two bits is called half adder. One that performs the addition of three bits (two significant bits and a previous carry) is called half adder. The name of circuits from the fact that two half adders can be employed to implement a full adder.

A binary adder-subtractor is a combinational circuit that performs the arithmetic operations of addition and subtraction with binary numbers. We will develop this circuit by means of a hierarchical design. The half adder design is carried out first, from which we develop the adder. Connecting n full adders in cascade produces a binary adder for two n-bit numbers.

The subtraction circuit is included by providing a complementing circuit.

1.1 Half Adder

From the verbal explanation of a half adder, we find that this circuit needs two binary inputs and two binary outputs. The input variables designated the augend and addend bits; the output variables produce the sum and carry. We assign symbols x and y to the inputs and S (for sum) and C (for carry) to the outputs. The truth table for the half adder is listed in Table 1. The C output is 1 only when both inputs are 1. The S output represents the least significant bit of the sum.

The simplified Boolean functions for the two outputs can be obtained directly from the truth table. The simplified sum of products expressions are:

$$S = x'y + xy'$$
$$C = xy$$

The logic diagram of the half adder implemented in sum of product is shown in fig. 1(a). It can be also implemented with an exclusive-OR and AND gate as shown in fig. 1(b). This form is used to show that two half adders can be used to construct a full adder.

1.2 Full-adder

A full adder is a combinational circuit that forms the arithmetic sum of three bits. It consists of three inputs and two outputs. Two of the inputs variables, denoted by x and y, represent the two significant bits to be added. The third input, z, represents the carry from the previous lower significant position. Two outputs are necessary because the arithmetic sum of three binary digits ranges in value from 0 to 3, and binary 2 or 3 needs two digits. The two outputs are designated by the symbols S for sum and C for carry. The binary variable S gives the value of the least significant bit of the sum. The binary variable C gives the output carry. The truth table of the full adder is listed in table 2. The eight row under the input variables designate all possible combinations of the three variables. The output variables are determined from the arithmetic sum of the input bits. When all input bits are 0, the output is 0. The S output is equal to 1 when only one input is equal to 1 or when all three inputs are equal to 1. The C output has a carry of 1 if two or three inputs are equal to 1.

A input and output bits of the combinational circuit have different interpretations at various stages of the problem. Physically, the binary signals of the inputs are considered binary digits

to be added arithmetically to form a two-digit sum at output. On the other hand, the binary values are considered as variables of Boolean functions when expressed in the truth table or when the circuit is implemented with logic gates.

4-Bit Adder
The bits are added with full adders, starting from the position to form the sum bit and carry. The input carry C0 in the least significant position must be 0. The value of Ci+1 in a given significant position is the output carry of the full adder. This value is transferred into the input carry of the full adder that adds the bits one higher significant position to the left. The sum bits are thus generated starting from the rightmost position and are available for the correct sum bits to appears at the outputs.

The 4bit adder is a typical example of a standard component. It can be used in many applications involving arithmetic operations. Observe that the design of this circuit by the classical method would require a truth table with 29 = 512 entries, since there are nine inputs to the circuit. By using an iterative method of cascading a standard function, it is possible to obtain a simple and straightforward implementation.

1.3 Binary Subtractor

The subtraction of unsigned binary numbers can be done most conveniently by means of complement. Subtraction A–B can be done by tacking the 2's complement of B and adding it to A. The 2's complement can be obtained by taking the 1's complement and adding one to the least significant pair of bits. The 1's complement can be implemented with the inverters and a one can be added to the sum through the input carry. The circuit for subtracting A–B consists of an adder with inverter placed between each data input B and the corresponding input of the full adder. The input carry C0 must be equal to 1 when performing subtraction. The operation thus performed becomes A, plus the 1's complement of B, plus 1.This is equal to A plus 2's complement of B. For unsigned numbers this gives A–B if A = B or the 2's complement of (B–A) if A < B. for signed numbers, the result is A – B, provided that there is no overflow.

The addition and subtraction operations can be combined into one circuit with one common binary adder. This is done by including an EX-OR gate with each full adder. A 4-bit adder-subtractor circuit is shown in fig 4. The mode input M controls the operation. When M = 0, the circuit is an adder, and when M = 1, the circuit becomes a subtractor. Each EX-OR gate receives input M and one of the inputs of B. when M = 0, we have B (Ex-OR) 0 = B. the full adder receive the value of B, the input carry is 0, and the circuit performs A plus B. when M = 1, we have B (Ex-OR) 1= B' and C0 = 1. The B inputs are complemented and a 1 is added through the input carry. The circuit performs the operation A plus the 2's complement of B. (The EX-OR with output is for detecting an overflow.)

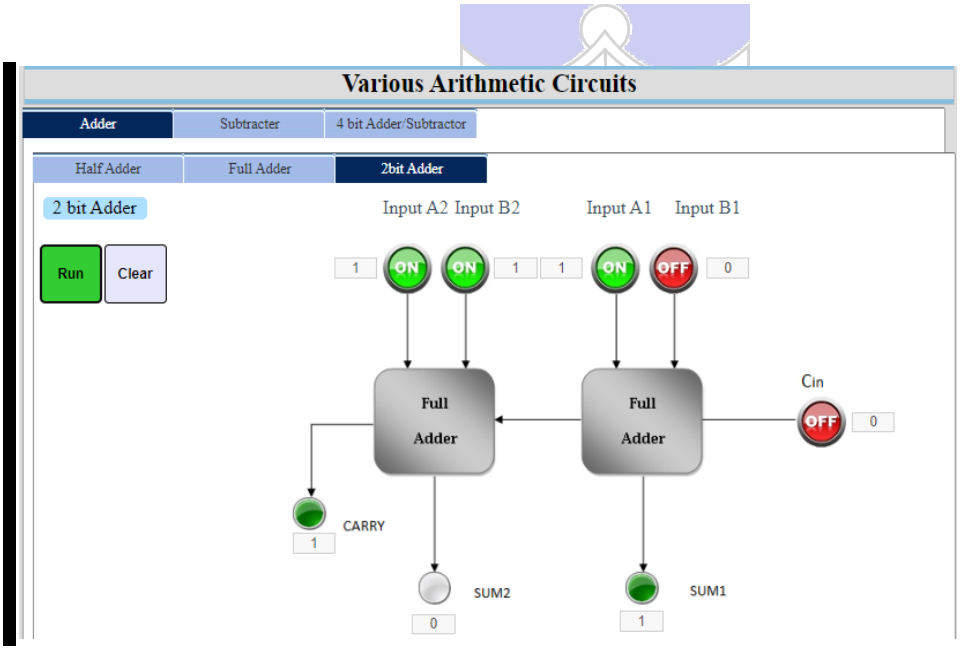
It is worth noting that binary numbers in the signed-complemented system are added and subtracted by the same basic addition and subtraction rules as unsigned numbers. Therefore, computers need only one common hardware circuit to handle both type of arithmetic. The user or programmer must interpret the results of such addition or subtraction differently. Depending on whether it is assumed that the numbers are signed or unsigned.

Procedure:

1. Select appropriate arithmetic operation.
2. Enter input A and / or B.
3. Select run button in the top to execute the operation.
4. Interpret the result.

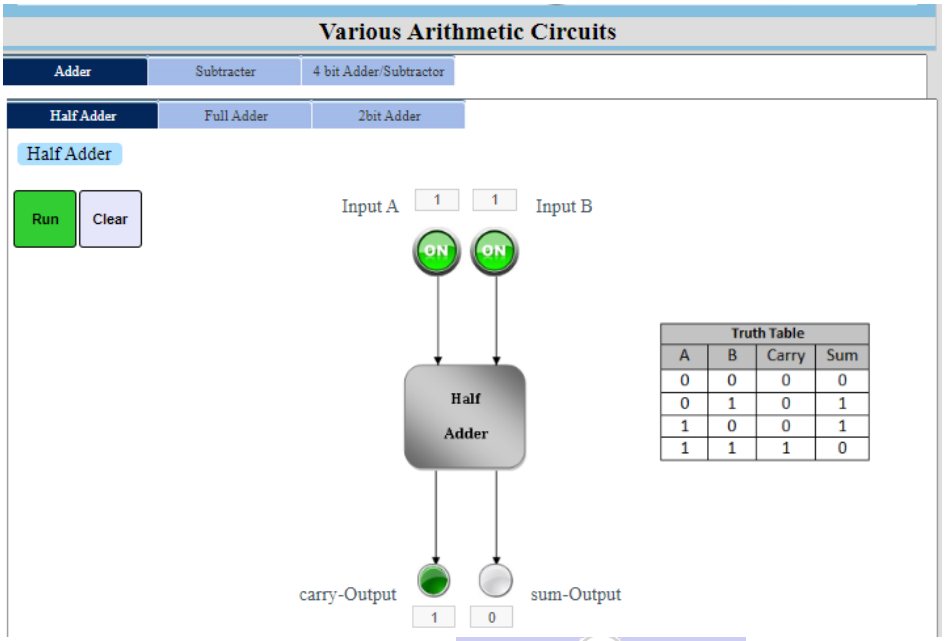
Observations and Results:

1. Binary Adder - Subtractor

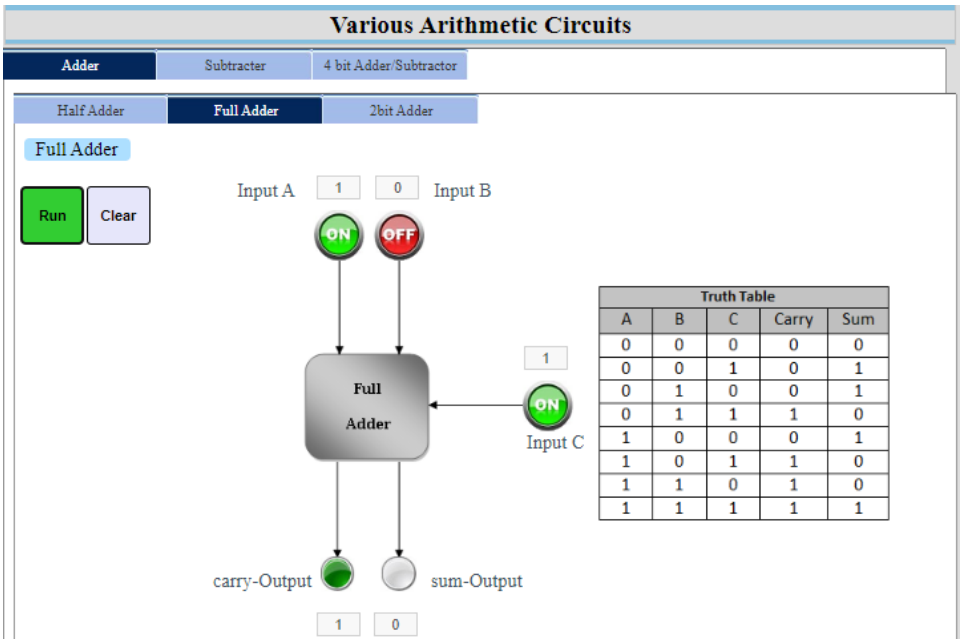


---

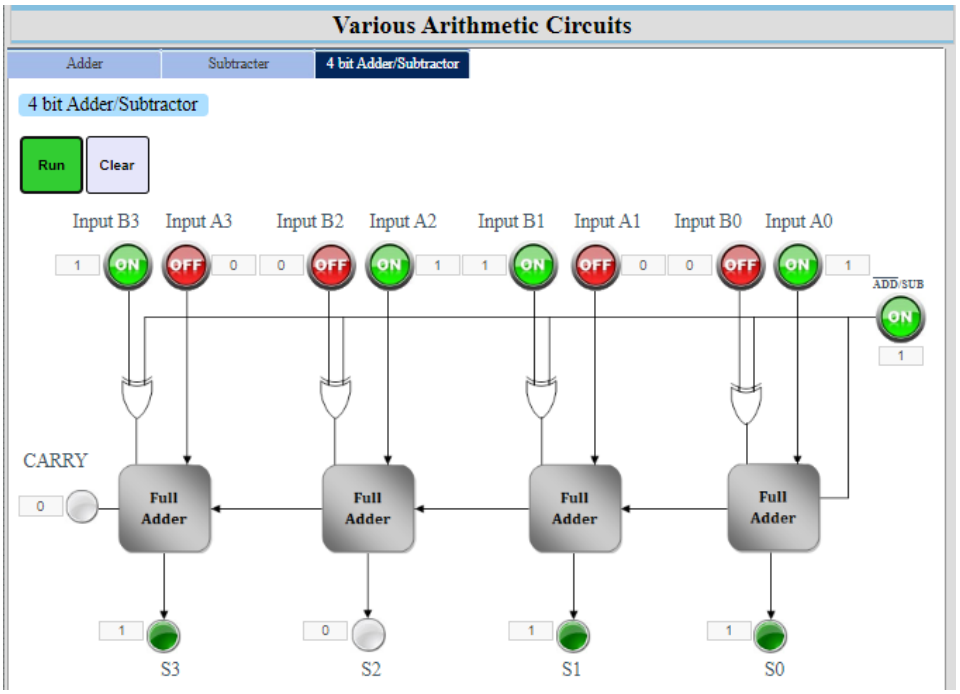
2)Half-adder:



3) Full Adder:



4) Binary subtractor



:

:

Conclusion:

In this experiment, we learned how to create a half-adder/subtractor and a full-adder/subtractor using various logic gates to demonstrate how to add and subtract binary values.

Grade: AA / AB / BB / BC / CC / CD /DD

Signature of faculty in-charge with date

1. References:Virtual lab

2. Books/ Journals/ Websites:
3. 1. R. P. Jain, “Modern Digital Electronics”, Tata McGraw Hill.
4. 2. http://www.allaboutcircuits.com/worksheets/basic-logic-gates/