**Dedicated to the legend**



**[Prof. S Shrinivasan](http://cp-ravikumar-english.blogspot.com/2016/04/remembering-prof-srinivasan.html)**

**For theory :** [refer here](https://youtube.com/playlist?list=PL803563859BF7ED8C)

**For practices :** [**refer here**](https://github.com/Sarabhian/Digital_Electronics)

Experiment No. 1

Aim: To study, design and implement various arithmetic circuits.

# Objective:

1. To understand working of various arithmetic circuits.
2. To understand various Boolean laws, K- map for simplification of digital circuits
3. To design various arithmetic circuits and its realization /implementation.
4. To design and implementation of various adders, subtractors, comparators, ALUs. etc.
5. To analyze various arithmetic circuits.

# Equipment:

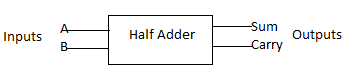
IC’s, gates (Ex-OR, AND, OR), regulated power supply, bread board, connecting wires, LED, DMM.

# Theory:

Arithmetic circuits are simple combination logical circuits which perform digital arithmetic operations like addition, subtraction, multiplication, division and comparison.

## Half-adder:

Block schematic of half-adder operation:

The half-adder needs two binary inputs: augend and addend bits; and gives two binary outputs: sum and carry.

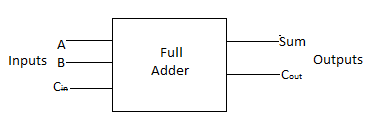
Limitations: In multidigit addition, we have to add two or more bits along with the carry of previous digit addition. Effectively such addition requires addition of three bits. This is not possible with half adder. Cascading of half adders is required to use the in practical life.

Truth Table for half-adder operation:

|  |  |  |  |
| --- | --- | --- | --- |
| Inputs | | Outputs | |
| A | B | Sum | Carry |
| 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 0 |
| 1 | 0 | 1 | 0 |
| 1 | 1 | 0 | 1 |

## Full-adder:

Block schematic of full-adder:

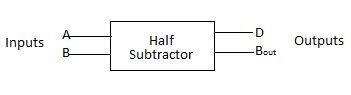
A full adder is a combinational circuit that performs the arithmetic sum of three input bits. It consists of three inputs and two outputs. Two of the input variables, denoted by A and B, represent the two significant bits to be added. The third input, Cin, represents the carry from the previous lower significant position.

Truth table for full-adder:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Inputs | | | Outputs | |
| A | B | Cin | 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 |

## Half-subtractor:

Block schematic of half-subtractor:

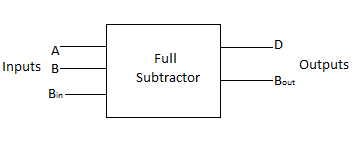
A logic circuit for the subtraction of B (subtrahend) from A (minuend) where A and B are 1-bit numbers are referred to as half-subtractor. The subtraction process has two inputs: A and B; and two outputs: D (difference) and Bout (borrow).

Truth Table for half-subtractor:

|  |  |  |  |
| --- | --- | --- | --- |
| Inputs | | Outputs | |
| A | B | D | Bin |
| 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 |
| 1 | 0 | 1 | 0 |
| 1 | 1 | 0 | 0 |

## Full-subtractor:

Block schematic of full-subtractor:

A full-subtractor is needed to perform multibit subtraction where a borrow from the previous bit may also be there. It has three inputs, A (minuend), B (subtrahend) and Bin (borrow from the previous stage) and two outputs, D (difference) and Bout (borrow).

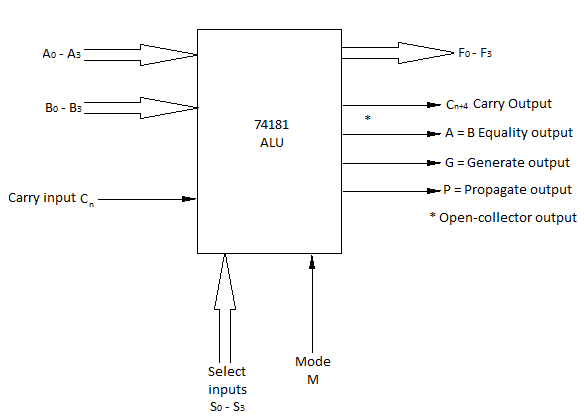
Truth table for full-subtractor:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Inputs | | | Outputs | |
| A | B | Bin | D | Bout |
| 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 1 | 1 |
| 0 | 1 | 0 | 1 | 1 |
| 0 | 1 | 1 | 0 | 1 |
| 1 | 0 | 0 | 1 | 0 |
| 1 | 0 | 1 | 0 | 0 |
| 1 | 1 | 0 | 0 | 1 |
| 1 | 1 | 1 | 1 | 1 |

**NOTE:** Adders and subtractors can be cascaded to give higher order bit addition and subtraction.

## Arithmetic Logic Unit (ALU):

Block diagram of 74181 ALU:



The functions of various input, output and control lines are as follows:

A and B: 4-bit binary data inputs

Cn: Carry input (active-low)

F: 4-bit binary data output

Cn+4: Carry output (active-low)

A = B: Logic 1 on this line indicates A = B

G: Carry generate output

P: Carry propagate output

Select input (S): Used to select any operation

Mode Contol (M): M=0 Arithmetic operations

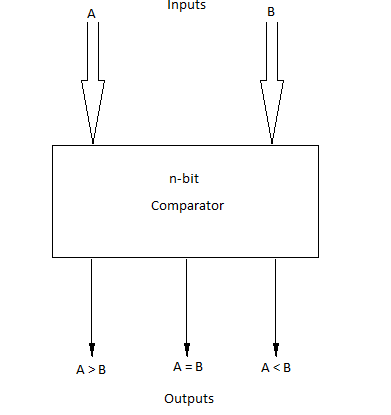
M=1 Logic operations

Truth Table for ALU 74181:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Line |  |  |  |  |  | Active high data | |
| M=1  Logic  Functions | M=0; Arithmetic operations | |
| Selection | | | |
| S3 | S2 | S1 | S0 | Cn=1 (no carry) | Cn=0 (with carry) |
| 0 | 0 | 0 | 0 | 0 | F=AC | F=A | F=A PLUS 1 |
| 1 | 0 | 0 | 0 | 1 | F=(A+B)C | F=A+B | F=(A+B) PLUS 1 |
| 2 | 0 | 0 | 1 | 0 | F=AC.B | F=A+B | F=(A+BC) PLUS 1 |
| 3 | 0 | 0 | 1 | 1 | F=0 | F=MINUS 1  (2’s COMPL) | F=ZERO |
| 4 | 0 | 1 | 0 | 0 | F=(AB)C | F=A PLUS ABC | F=A PLUS ABC PLUS 1 |
| 5 | 0 | 1 | 0 | 1 | F=BC | F=(A+B) PLUS ABC | F=(A+B) PLUS ABC PLUS 1 |
| 6 | 0 | 1 | 1 | 0 | F=A EXOR B | F=A MINUS B MINUS 1 | F=A MINUS B |
| 7 | 0 | 1 | 1 | 1 | F=ABC | F=ABC MINUS 1 | F=ABC |
| 8 | 1 | 0 | 0 | 0 | F=AC +B | F=A PLUS AB | F=A PLUS AB PLUS 1 |
| 9 | 1 | 0 | 0 | 1 | F=(A EXOR B)C | F=A PLUS B | F=A PLUS B PLUS 1 |
| 10 | 1 | 0 | 1 | 0 | F=B | F=(A+BC) PLUS AB | F=(A+BC) PLUS AB PLUS 1 |
| 11 | 1 | 0 | 1 | 1 | F=AB | F=AB MINUS 1 | F=AB |
| 12 | 1 | 1 | 0 | 0 | F=1 | F=A PLUS A\* | F=A PLUS A PLUS 1 |
| 13 | 1 | 1 | 0 | 1 | F=A+BC | F=(A+B) PLUS A | F=(A+B) PLUS A PLUS 1 |
| 14 | 1 | 1 | 1 | 0 | F=A+B | F=(A+BC) PLUS A | F=(A+B) PLUS A PLUS 1 |
| 15 | 1 | 1 | 1 | 1 | F=A | F=A MINUS 1 | F=A |

## Comparators:

Block schematic diagram of n-bit comparator:

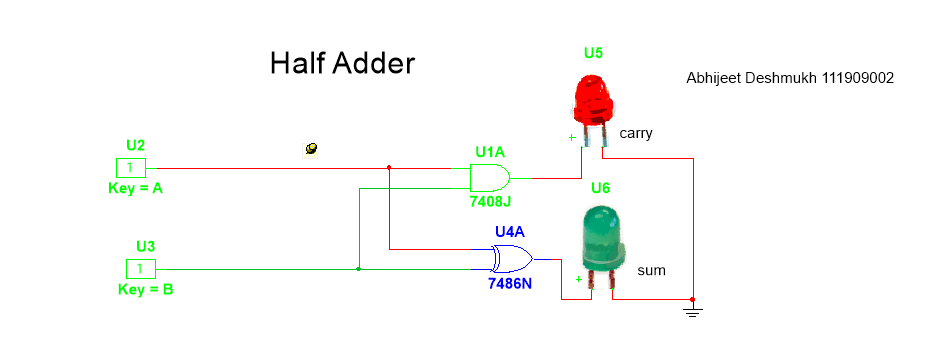
 It receives 2 n-bit inputs A and B and the outputs are A > B, A = B, A < B. The IC’s can be cascaded to compare words of greater lengths without using external gates. The A > B, A = B, A < B outputs of a stage handling less significant bits are connected to corresponding A > B, A = B, A < B cascading inputs of the next stage handling more significant bits. The stage handling the least significant bits must have A = B input connected to logic 1 level and A > B and A < B inputs connected to logic 0 or 1 level.

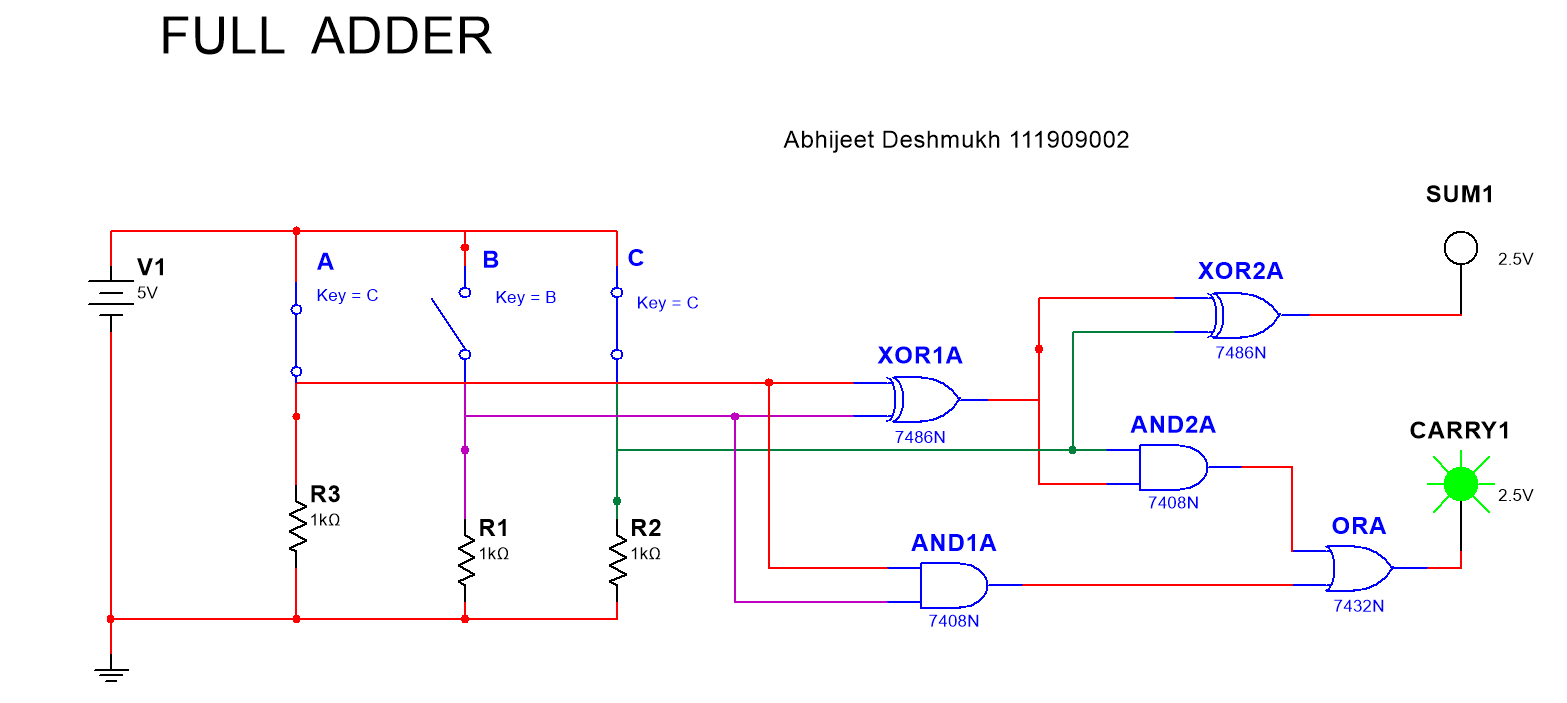
Truth table of 7485 4-bit comparator:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Comparing inputs  A,B | Cascading inputs | | | Outputs | | |
| A > B | A = B | A < B | A > B | A = B | A < B |
| A > B | X | X | X | 1 | 0 | 0 |
| 1 | 0 | 0 | 1 | 0 | 0 |
| X | 1 | X | 0 | 1 | 0 |
| A = B | 0 | 0 | 1 | 0 | 0 | 1 |
| 0 | 0 | 0 | 1 | 0 | 1 |
| 1 | 0 | 1 | 0 | 0 | 0 |
| A < B | X | X | X | 0 | 0 | 1 |

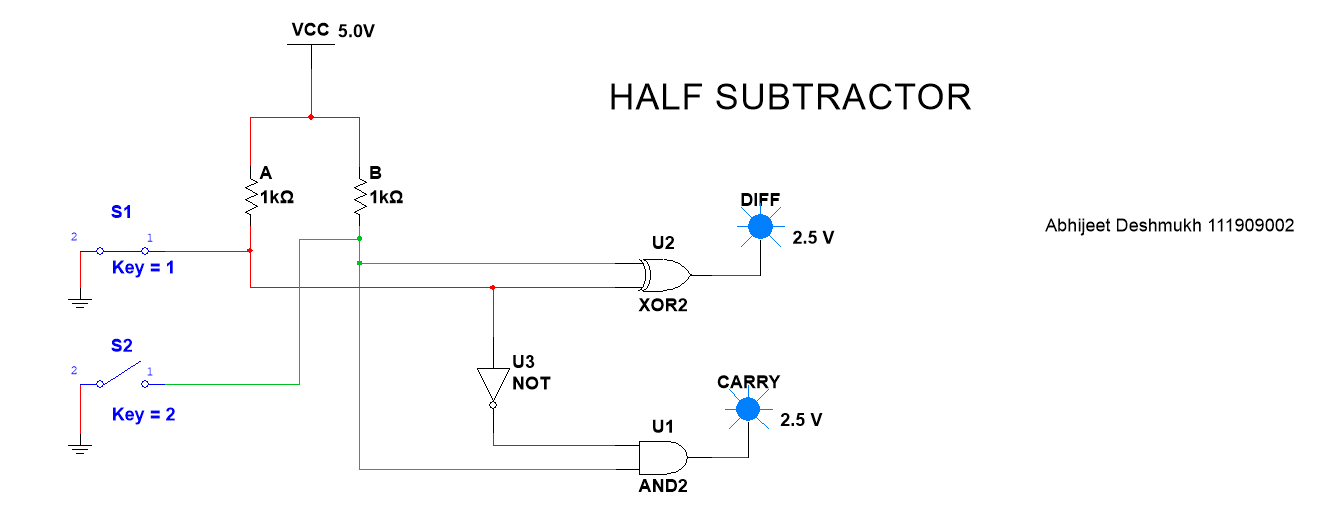
# Design:

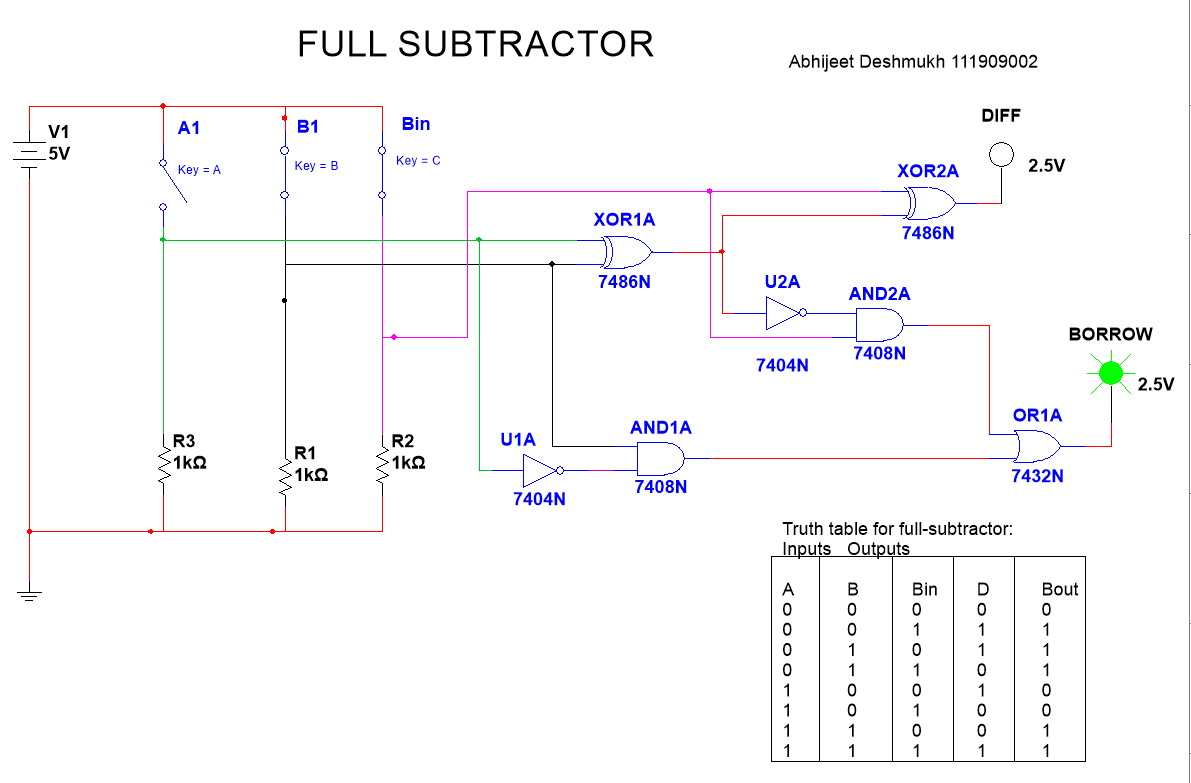
## ADDER



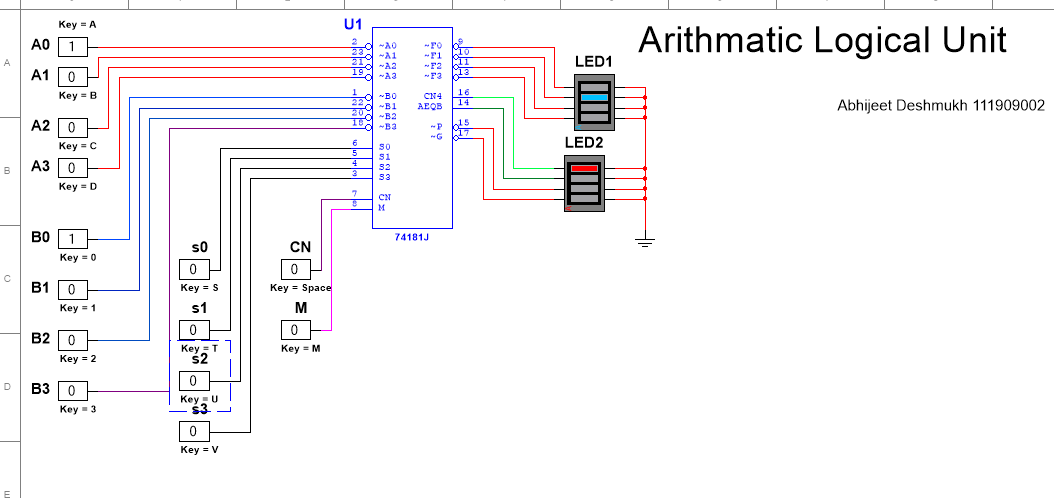


## SUBTRACTOR

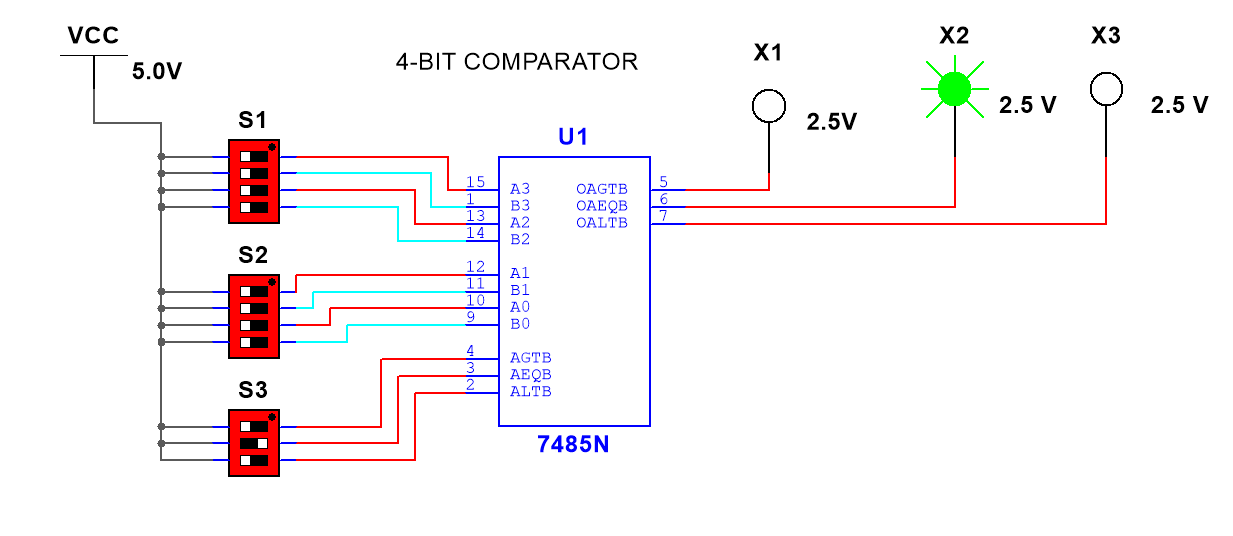




## ALU:



## comparator:



# Procedure:

1. Connect the components as shown in the circuit diagram.
2. Give +5V supply to the IC’s.
3. From the LED observe the outputs.
4. Make a note of the truth tables in the observations.
5. Verify the truth tables accordingly.

# Observations:

In observations I have performed simulation and got below Truth tables.

## Half adder

|  |  |  |  |
| --- | --- | --- | --- |
| Inputs | | Outputs | |
| A | B | Sum | Carry |
| 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 0 |
| 1 | 0 | 1 | 0 |
| 1 | 1 | 0 | 1 |

## Half subtractor

|  |  |  |  |
| --- | --- | --- | --- |
| Inputs | | Outputs | |
| A | B | D | Bin |
| 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 |
| 1 | 0 | 1 | 0 |
| 1 | 1 | 0 | 0 |

## Full adder

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Inputs | | | Outputs | |
| A | B | Cin | 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 |

## Full subtractor

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Inputs | | | Outputs | |
| A | B | Bin | D | Bout |
| 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 1 | 1 |
| 0 | 1 | 0 | 1 | 1 |
| 0 | 1 | 1 | 0 | 1 |
| 1 | 0 | 0 | 1 | 0 |
| 1 | 0 | 1 | 0 | 0 |
| 1 | 1 | 0 | 0 | 1 |
| 1 | 1 | 1 | 1 | 1 |

## ALU

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Line |  |  |  |  |  | Active high data | |
| M=1  Logic  Functions | M=0; Arithmetic operations | |
| Selection | | | |
| S3 | S2 | S1 | S0 | Cn=1 (no carry) | Cn=0 (with carry) |
| 0 | 0 | 0 | 0 | 0 | F=AC | F=A | F=A PLUS 1 |
| 1 | 0 | 0 | 0 | 1 | F=(A+B)C | F=A+B | F=(A+B) PLUS 1 |
| 2 | 0 | 0 | 1 | 0 | F=AC.B | F=A+B | F=(A+BC) PLUS 1 |
| 3 | 0 | 0 | 1 | 1 | F=0 | F=MINUS 1  (2’s COMPL) | F=ZERO |
| 4 | 0 | 1 | 0 | 0 | F=(AB)C | F=A PLUS ABC | F=A PLUS ABC PLUS 1 |
| 5 | 0 | 1 | 0 | 1 | F=BC | F=(A+B) PLUS ABC | F=(A+B) PLUS ABC PLUS 1 |
| 6 | 0 | 1 | 1 | 0 | F=A EXOR B | F=A MINUS B MINUS 1 | F=A MINUS B |
| 7 | 0 | 1 | 1 | 1 | F=ABC | F=ABC MINUS 1 | F=ABC |
| 8 | 1 | 0 | 0 | 0 | F=AC +B | F=A PLUS AB | F=A PLUS AB PLUS 1 |
| 9 | 1 | 0 | 0 | 1 | F=(A EXOR B)C | F=A PLUS B | F=A PLUS B PLUS 1 |
| 10 | 1 | 0 | 1 | 0 | F=B | F=(A+BC) PLUS AB | F=(A+BC) PLUS AB PLUS 1 |
| 11 | 1 | 0 | 1 | 1 | F=AB | F=AB MINUS 1 | F=AB |
| 12 | 1 | 1 | 0 | 0 | F=1 | F=A PLUS A\* | F=A PLUS A PLUS 1 |
| 13 | 1 | 1 | 0 | 1 | F=A+BC | F=(A+B) PLUS A | F=(A+B) PLUS A PLUS 1 |
| 14 | 1 | 1 | 1 | 0 | F=A+B | F=(A+BC) PLUS A | F=(A+B) PLUS A PLUS 1 |
| 15 | 1 | 1 | 1 | 1 | F=A | F=A MINUS 1 | F=A |

## Comparator

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **A1** | **A0** | **B1** | **B0** | **A<B** | **A=B** | **A>B** |
| 0 | 0 | 0 | 0 | 0 | 1 | 0 |
| 0 | 0 | 0 | 1 | 1 | 0 | 0 |
| 0 | 0 | 1 | 0 | 1 | 0 | 0 |
| 0 | 0 | 1 | 1 | 1 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 | 1 |
| 0 | 1 | 0 | 1 | 0 | 1 | 0 |
| 0 | 1 | 1 | 0 | 1 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 0 | 0 |
| 1 | 0 | 0 | 0 | 0 | 0 | 1 |
| 1 | 0 | 0 | 1 | 0 | 0 | 1 |
| 1 | 0 | 1 | 0 | 0 | 1 | 0 |
| 1 | 0 | 1 | 1 | 1 | 0 | 0 |
| 1 | 1 | 0 | 0 | 0 | 0 | 1 |
| 1 | 1 | 0 | 1 | 0 | 0 | 1 |
| 1 | 1 | 1 | 0 | 0 | 0 | 1 |
| 1 | 1 | 1 | 1 | 0 | 1 | 0 |

# Result:

We have performed the simulations of adders, subtractors, ALU and comparators and truth tables are verified successfully.

# Conclusion:

Using basic gates, we can perform arithmetic operations like addition, subtraction, multiplication and logic operations using ALU and comparator ICs.

# What did you learn?

While designing circuit I learnt that TTL ICs namely

7404N – NOT Gate

7408N- AND Gate

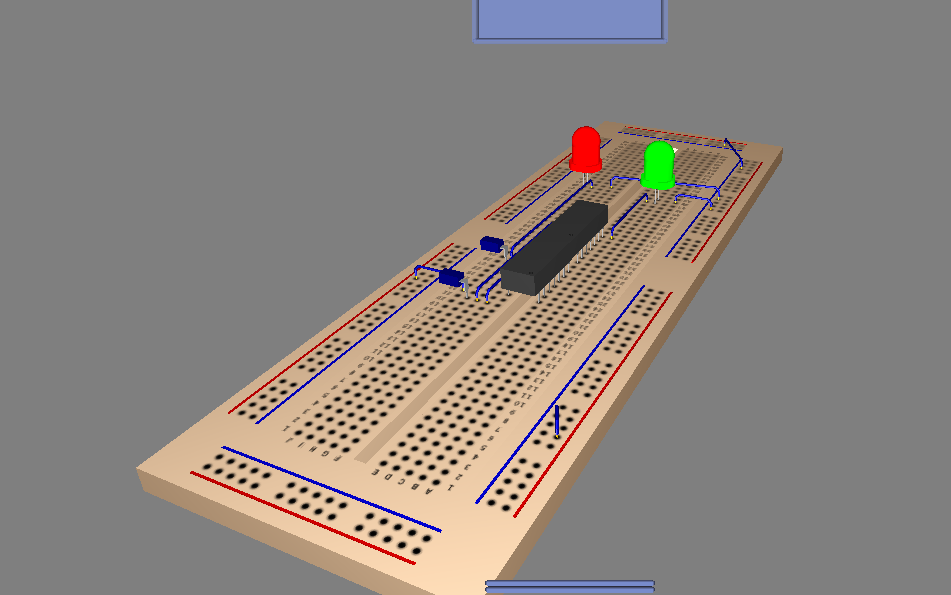
7432N -OR Gate

7486N- XOR Gate

74181J-ALU UNIT

These can be used for creating logic circuits.

# Assignment:



As an assignment I have tried to create 3D model or whiteboard view using multisim but making such 3d models is time consuming so not continued in next experiments.

THANK YOU!