

Submitted by : Spogmai Jan

Roll no : 2430 -0081

Submitted to : Sir bilal dar

Subject : digital logic design lab

Assignment no 05

Q1:**What is half subtractor?**

A **Half Subtractor** is a basic digital circuit used to subtract two binary numbers (**A** and **B**). It has two outputs:

1. **Difference (D)** → Shows the subtraction result.
2. **Borrow (B)** → Indicates if borrowing is needed.

Q2**: For implementing half subtractor how many EX-OR, AND gates and Not gates are required?**

To implement a **Half Subtractor**, we need:

1. **1 XOR Gate** → For the **Difference (D = A ⊕ B)**
2. **1 NOT Gate** → To invert A (for Borrow calculation)
3. **1 AND Gate** → For the **Borrow (B = A̅ • B)**

Q3 : **What are the logical equations for difference & borrow?**

The logical equations for a **Half Subtractor** are:

Difference : A xor B (it gives 1 when A and B are different .

Borrow : A⋅B (**NOT and AND Gates**)

Borrow occurs when **A is 0** and **B is 1** (i.e., subtraction needs borrowing).

Q4 : **How full subtractor is different from half subtractor.**

A **Half Subtractor** subtracts two binary numbers (**A and B**) and gives a **Difference** and **Borrow** but does not consider borrow from a previous stage. A **Full Subtractor**, on the other hand, subtracts three inputs (**A, B, and Borrow-in**) and gives a **Difference** and **Borrow-out**, making it useful for multi-bit subtraction. The Full Subtractor is an improved version of the Half Subtractor as it handles borrow from the previous stage.

Q5 : **How many bits we use in half subtractor for subtraction?**

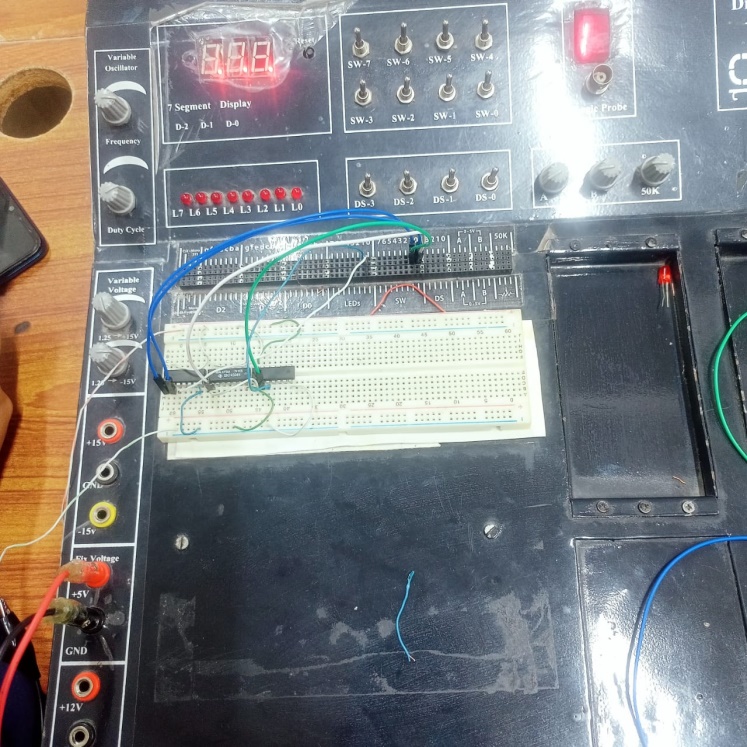
A **Half Subtractor** subtracts **only 1-bit** at a time. It takes two **single-bit** binary inputs (**A and B**) and produces a **1-bit Difference** and a **1-bit Borrow**.

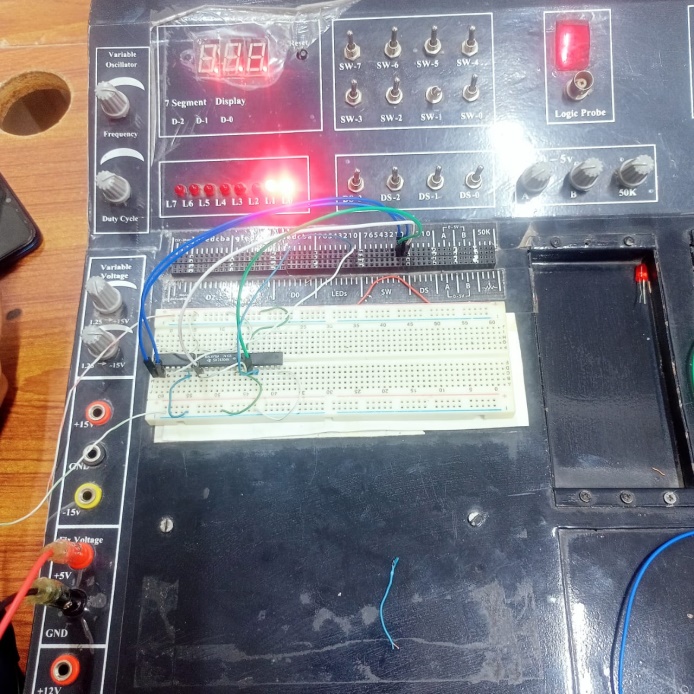
**-------------------------------------------------------**

**Question No. 6:**

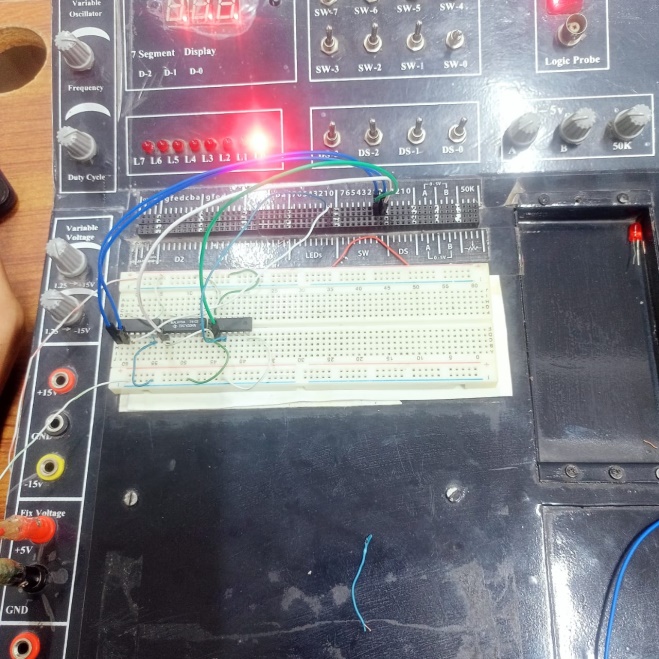
**OBSERVATION [L=logic 0, H=logic 1]**

|  |  |  |  |
| --- | --- | --- | --- |
| **INPUT A** | **INPUT B** | **𝐷 = 𝐴⨁𝐵** | **𝐵0 = 𝐴′𝐵** |
| **L** | **L** | **L** | **L** |
| **L** | **H** | **H** | **H** |
| **H** | **L** | **H** | **L** |
| **H** | **H** | **L** | **L** |

**Switch 0 is off Switch 1 is off both Led is 0** **:**

**Switch 0 is off Switch 1 is on Both Led is 1**

**Switch 0 is on switch 1 is off led 0 is on led 1 is off**



**Question No 7:**

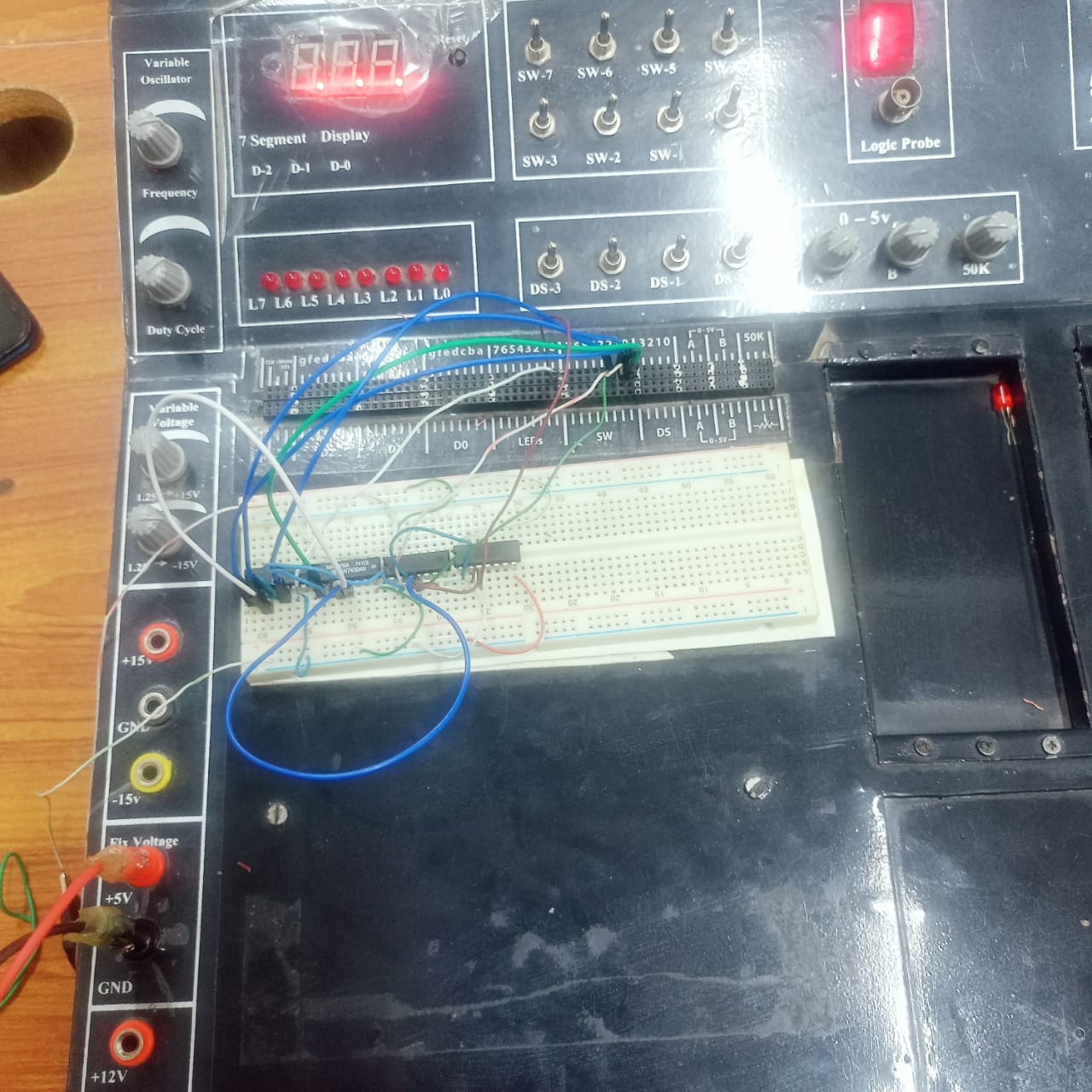
**OBSERVATION [L=logic 0, H=logic 1]**

**Full Subtractor:**

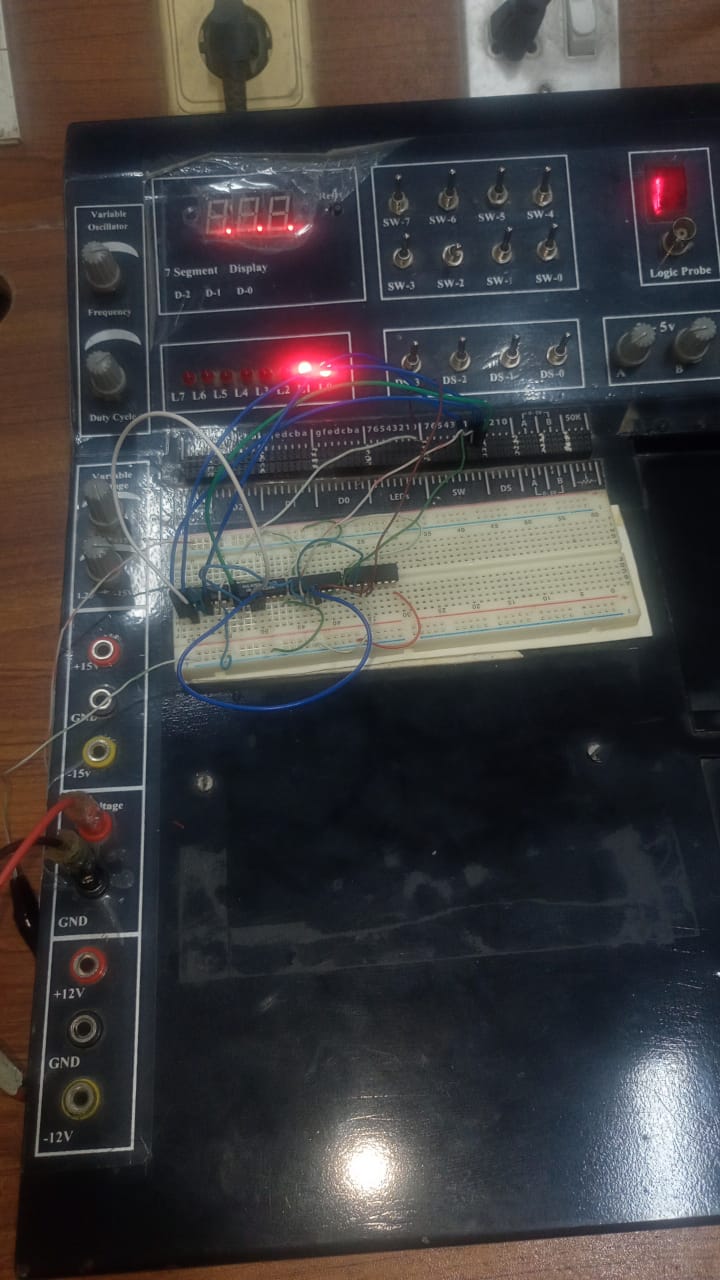
Truth Table

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Input**  **A** | **Input**  **B** | **Input**  **C** | | **D=A XOR B XOR C** | | **B0= A B+ A C + B** |
| 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 | | 1 |
| 1 | 1 | 0 | | 0 | | 0 |
| 1 | 1 | | 1 | 1 | 1 | |

**Switch 0 is off switch 1 is off switch 2 is off Leds are off**



**Switch 0 and Switch 1 is off switch 2 is on Led 0 and Led 1 is on**

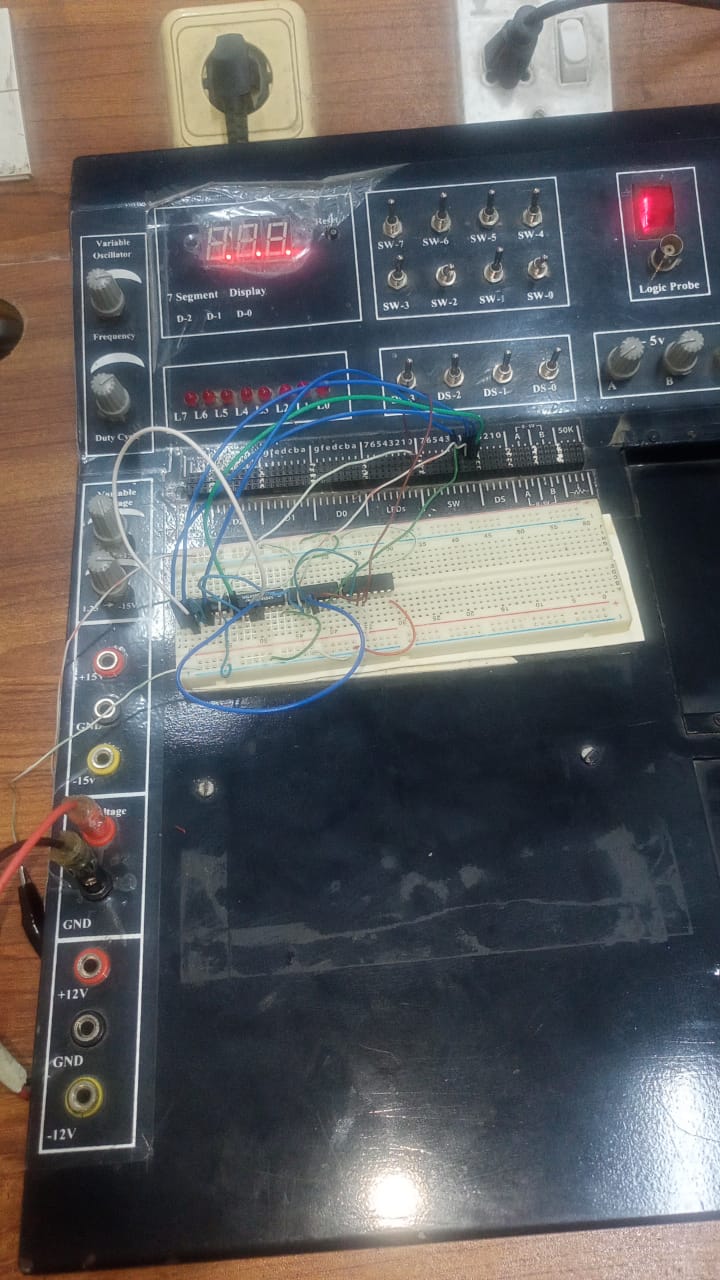


**Switch 0 is off Switch 1 is on switch 2 is off Led 0 and Led 1 is on**

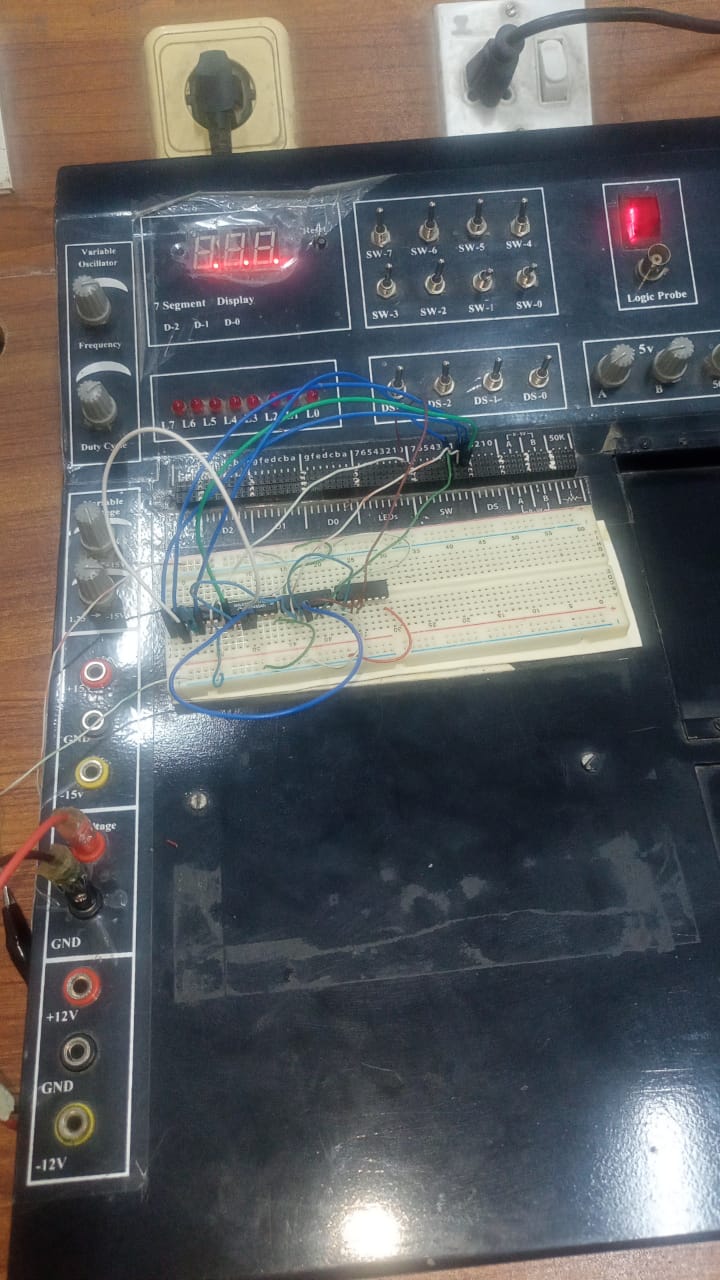


**Switch 0 is on a Switch 1 and switch 2 is of Led 0 is on and Led 1 is off**



**Switch 0 on Switch 1 is off switch 2 is on Led 0 and Led 1 is off**

**Switch 0 on Switch 1 is on switch 2 is of Led 0 and Led 1 is off**



**Switch 0 and Switch 1 and switch 2 is on Led 0 and Led 1 is on**

--------------------------------------------------