DSM LAB REPORT-3

Lab 3: Full Adder and Subtractor

Group 14: Aaryan Nakul Shah (2024113014)

Objective:

To design, assemble, and test binary adders and subtractors.

Part A: Binary Half Adder

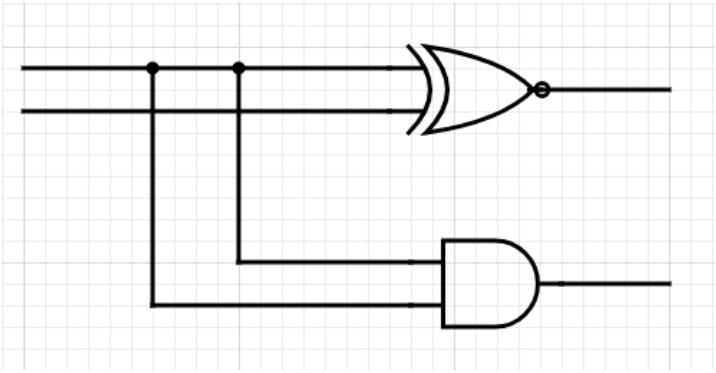
Objective:

To make a Binary Half Adder.

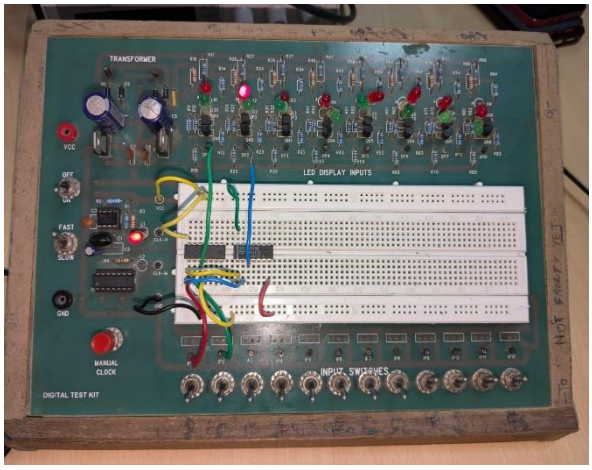
Components Used:

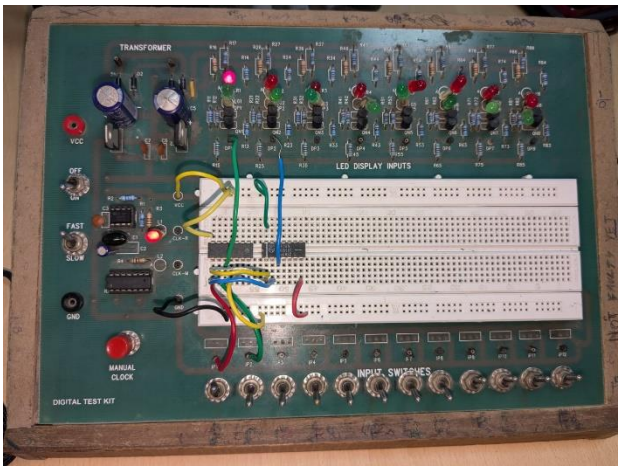
Digital Test Kit, wires, and ICs = MC74AC86N,74F08N

Reference Circuit:



Outputs:



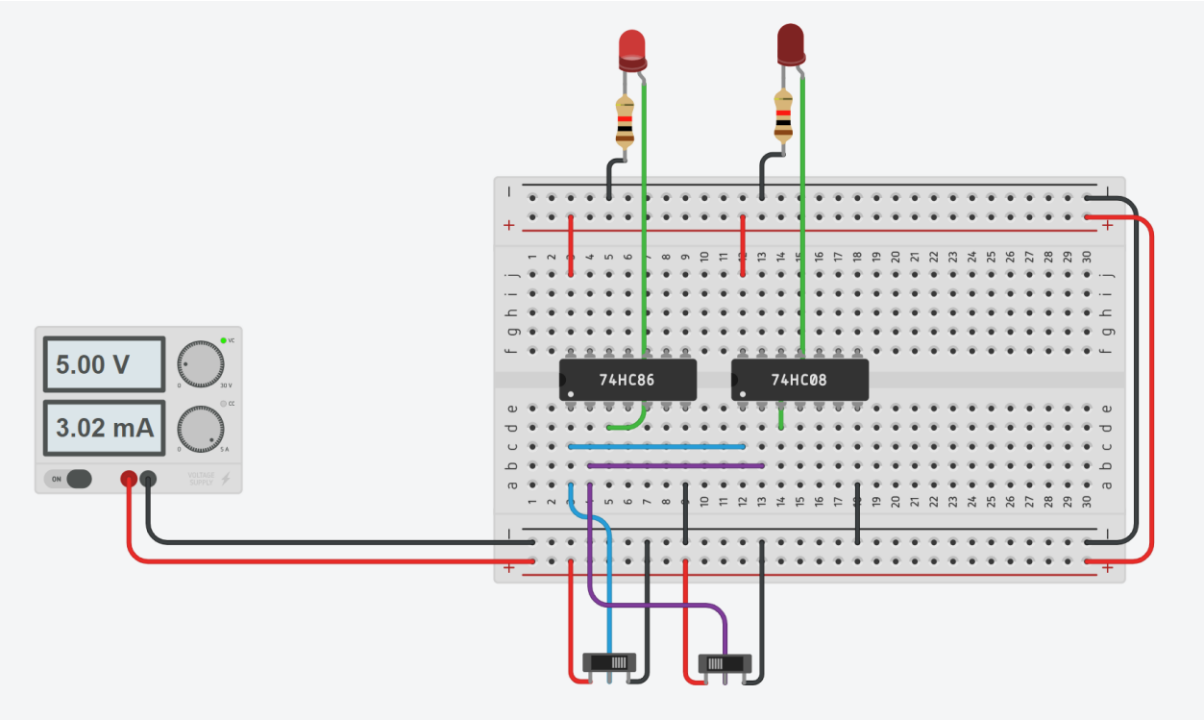


Procedure:

1. Connect the VCC and GND pins of the Digital Test Kit and the ICs.
2. The XOR of the two inputs gives the sum(S1) of a Binary Half Adder while the AND of the two inputs gives the carry(C1). Hence, make necessary connections using the reference circuit.

Tinkercad Simulations:

https://www.tinkercad.com/things/dKln3XVVgin-dsmlab3exp1?sharecode=tx30-XSmnFIHOA687drFDbhVO4GZ1EgGI6_4oFRY8Rw



Conclusion:

Half Binary Adder successfully made; the truth table is as follows:

A	B	SUM (A⊕B)	CARRY (A·B)
0	0	0	0
1	0	1	0
0	1	1	0
1	1	0	1

Part B: Binary Full Adder

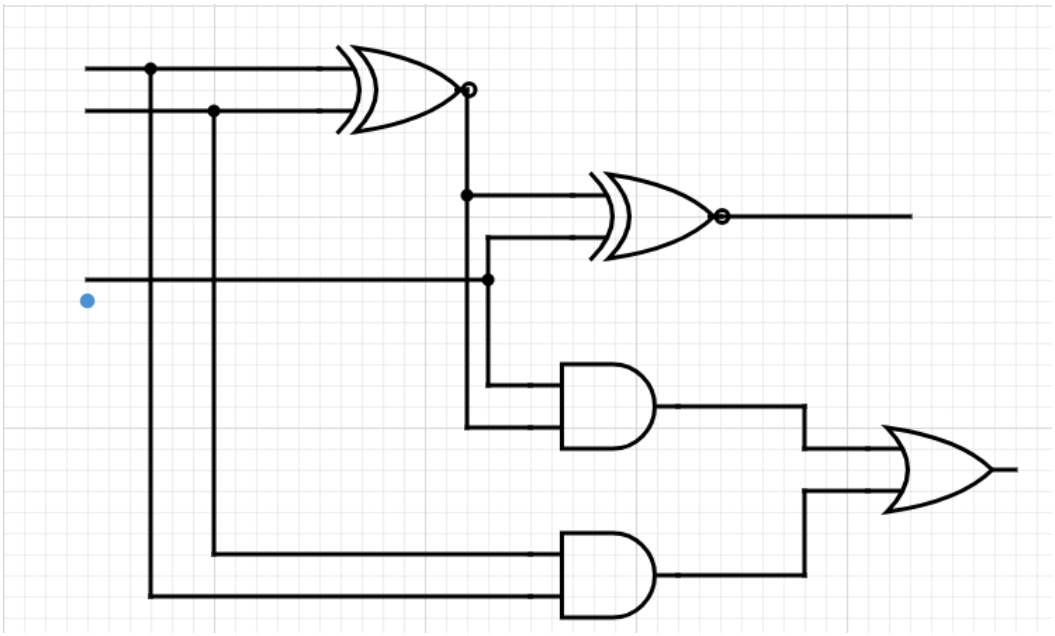
Objective:

To make a Binary Full Adder.

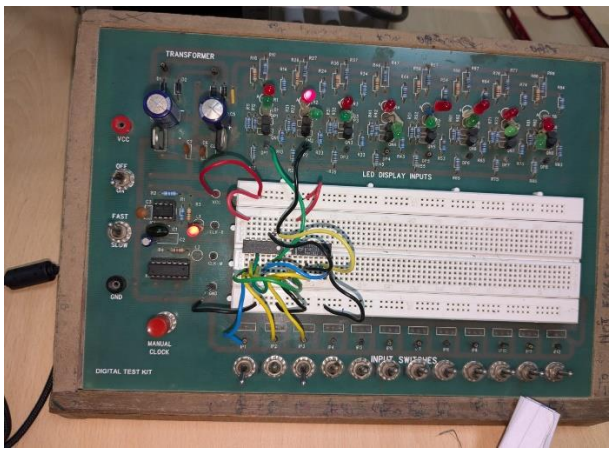
Components Used:

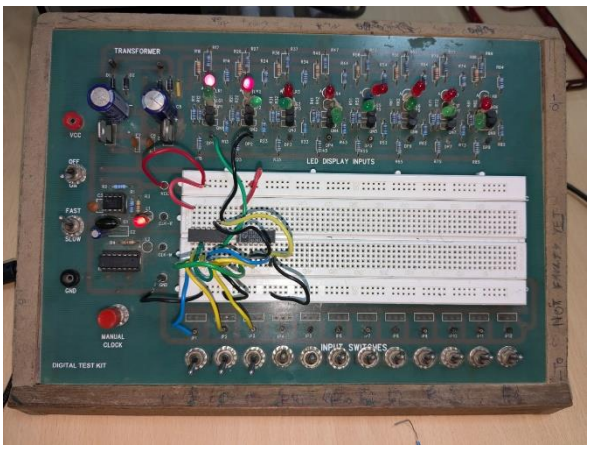
Digital Test Kit, wires, and ICs = MC74AC86N,74F08N

Reference Circuit:



Outputs:



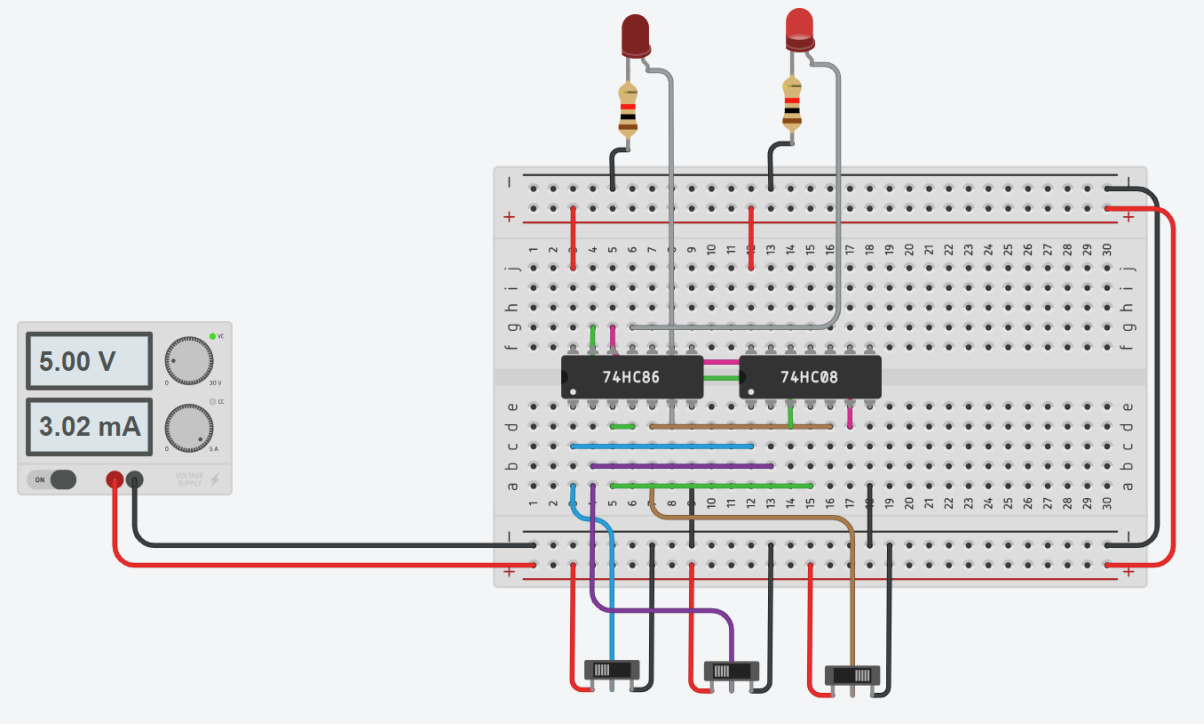


Procedure:

1. Set up the circuit as in Part A.
2. The final sum is obtained by the XOR of the third input and S1, while the final carry is determined by C1 + C2, with C2 representing the result of the AND operation between S1 and the third input.
3. Since we need to carry out the OR operation without using the OR gate, we can compare the truth tables of the XOR and OR gates. They differ only when both inputs are 1. Looking at the truth table of C1 and C2, we can see that they are never 1 simultaneously. Therefore, an XOR operation on C1 and C2 will be logically equivalent to performing the OR operation between C1 and C2.

Tinkercad Simulations:

https://www.tinkercad.com/things/7Qw0pqpfbjX-dsmlab3exp2?sharecode=w0xrNIRsHeHdP_MwNXSeV3icc8QTEMuGB6fX67W5_vc



Conclusion:

Full Binary Adder successfully made; the truth table is as follows:

A	B	C	C1	C2	CARRY	SUM
0	0	0	0	0	0	0
0	0	1	0	0	0	1
0	1	0	0	0	0	1
0	1	1	0	1	1	0
1	0	0	0	0	0	1
1	0	1	0	1	1	0
1	1	0	1	0	1	0
1	1	1	1	0	1	1

Part C: Binary Half Subtractor

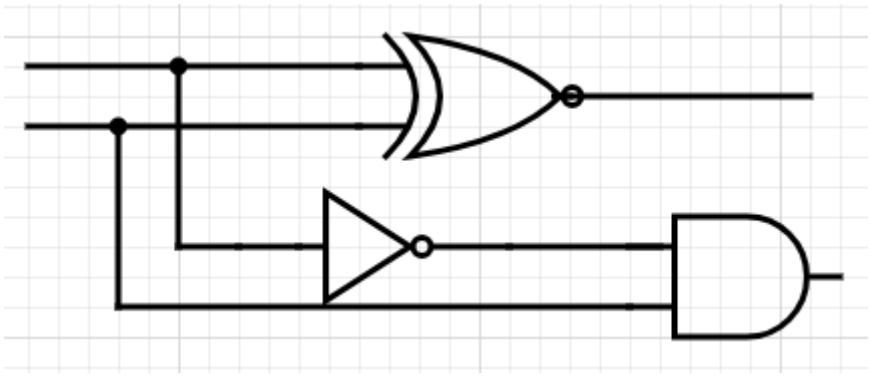
Objective:

To make a Binary Half Subtractor.

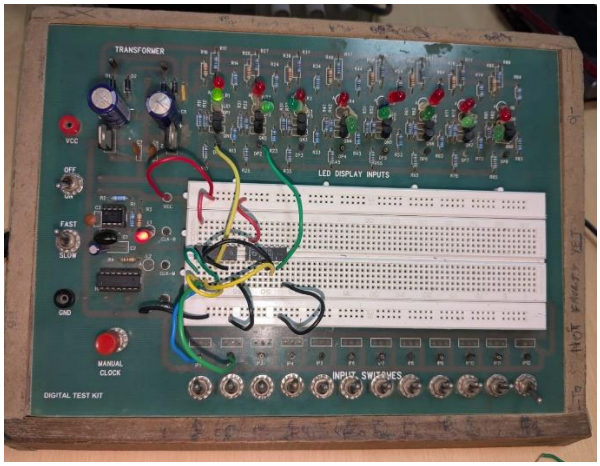
Components Used:

Digital Test Kit, wires, and ICs = MC74AC86N,74F08N

Reference Circuit:



Outputs:



Procedure:

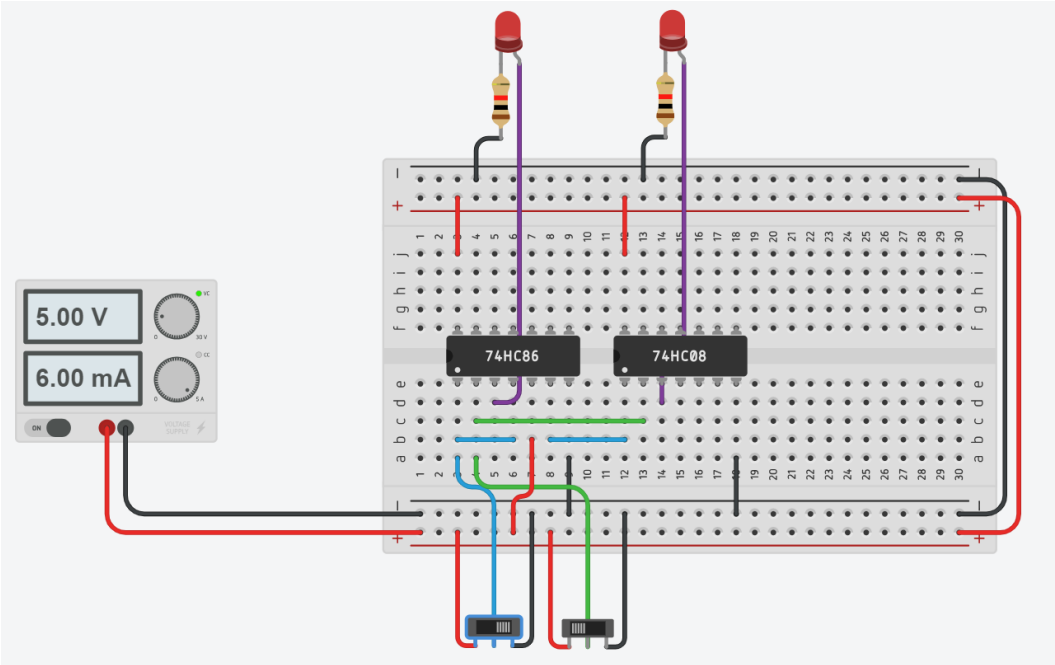
1. Connect the VCC and GND pins of the Digital Test Kit and the ICs.
2. The XOR of the two inputs gives the difference(D1) of a Binary Half Subtractor while the borrow(B1) is given by the AND of the ‘not’ or

complement of one of the inputs with the other. Hence, make necessary connections using the reference circuit.

3. To 'not' one of the inputs, you can use either the NOT gate or take an XOR of that input with the other input as 1 (accomplished by connecting VCC as input).

Tinkercad Simulations:

https://www.tinkercad.com/things/cWzwkH0gl0V-dsmlab3exp3?sharecode=ud8ogkLUoaLVJXuKkJ947YUfWUauljmqHIlBcecSqKY



Conclusion:

Half Binary Adder successfully made; the truth table is as follows:

A	B	DIFFERENCE(A⊕B)	BORROW (A'·B)
0	0	0	0
1	0	1	0
0	1	1	1
1	1	0	0

Part D: Binary Full Subtractor

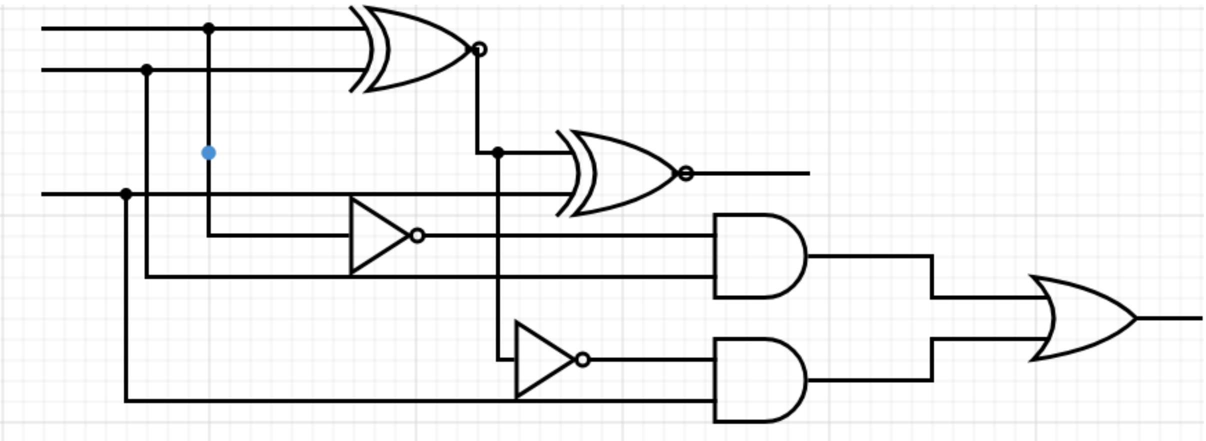
Objective:

To make a Binary Full Subtractor.

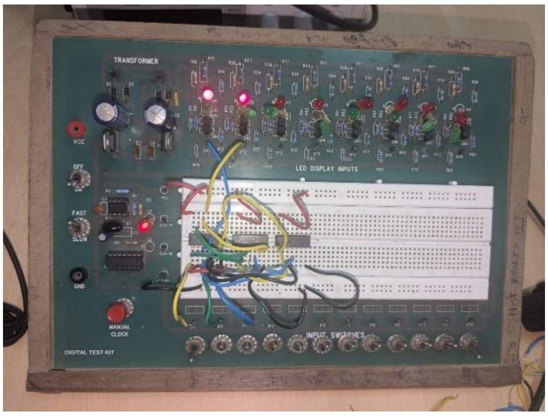
Components Used:

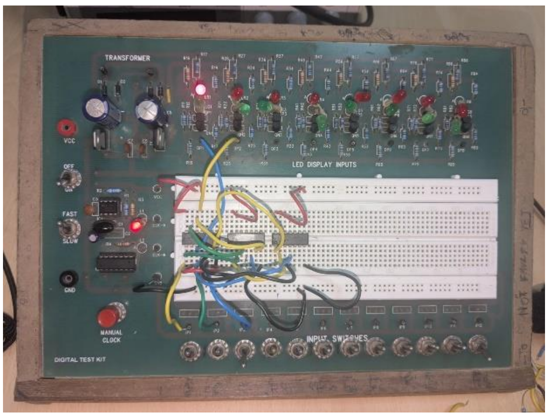
Digital Test Kit, wires, and ICs = MC74AC86N,74F08N,74HC04

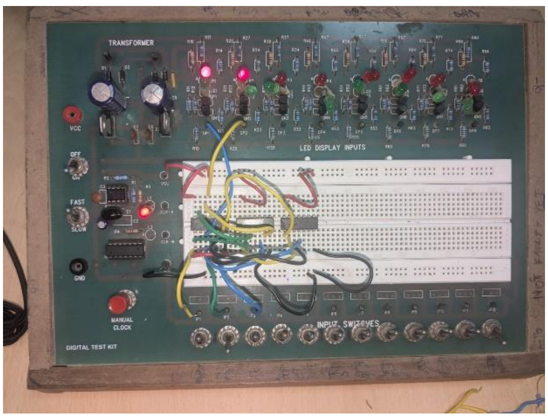
Reference Circuit:



Outputs:





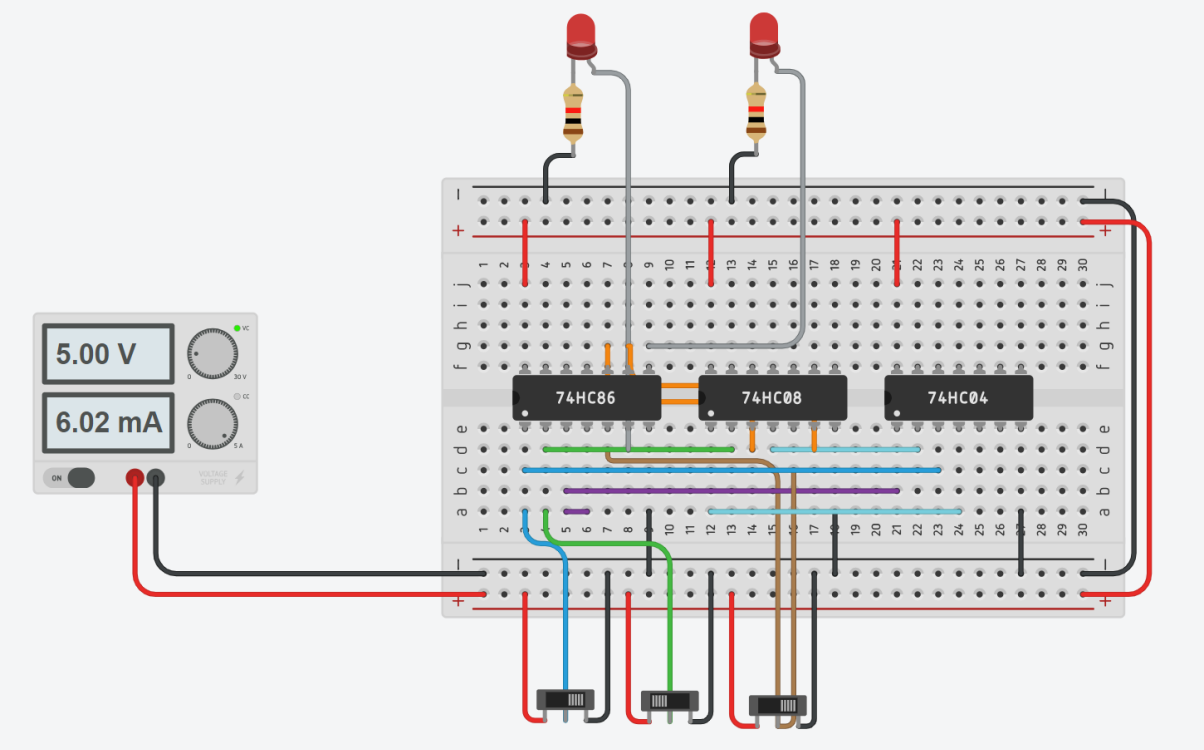


Procedure:

1. Set up the circuit as in Part C.
2. The final difference is obtained by the XOR of the third input and S1, while the final borrow is determined by B1 + B2, with B2 representing the result of the AND operation between D1' ('not' of D1) and the third input. Carry out the implementation of the NOT gate using the XOR gate as done in Part C.
3. Since we need to carry out the OR operation without using the OR gate, we can compare the truth tables of the XOR and OR gates. They differ only when both inputs are 1. Looking at the truth table of B1 and B2, we can see that they are never 1 simultaneously. Therefore, an XOR operation on B1 and B2 will be logically equivalent to performing the OR operation between B1 and B2.

Tinkercad Simulations:

https://www.tinkercad.com/things/gvxLLqhfNDi-dsmlab3exp4?sharecode=m8P3zH9FSs2rKOsjqSDVdH6m5_pQ2LELPVoMBy8Ru1M



Conclusion:

Full Binary Subtractor successfully made; the truth table is as follows:

A	B	C	B1	B2	BORROW	DIFFERENCE
0	0	0	0	0	0	0
0	0	1	0	1	1	1
0	1	0	1	0	1	1
0	1	1	1	0	1	0
1	0	0	0	0	0	1
1	0	1	0	0	0	0
1	1	0	0	0	0	0
1	1	1	0	1	1	1