

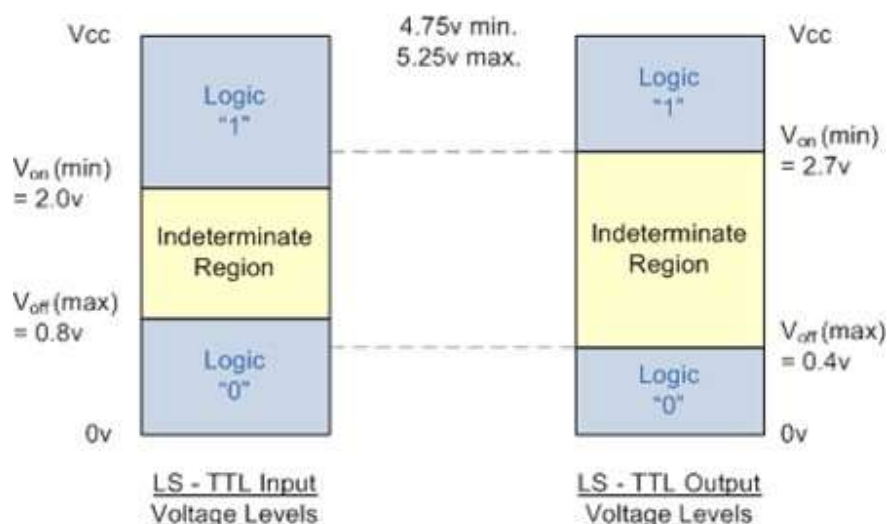
Lab#4: Study of Boolean Logic Operations using Digital ICs

Objective: To study and verify various Boolean logic operations and the De Morgan's laws using digital ICs.

Overview:

Standard commercially available **Digital Logic Gates** are available in two basic forms, **TTL** which stands for *Transistor-Transistor Logic* such as the 7400 series, and **CMOS** which stands for *Complementary Metal-Oxide-Silicon* which is the 4000 series of Integrated Circuits, (IC) or "chips" as it is commonly called. Generally speaking, **TTL** IC's use NPN type *Bipolar Junction Transistors* while **CMOS** IC's use *Field Effect Transistors* or FET's for both their input and output circuitry.

There are a large variety of logic gate types in both the Bipolar and CMOS families of digital logic gates such as 74L, 74LS, 74ALS, 74HC, 74HCT, 74ACT etc, with each one having its own distinct advantages and disadvantages and the exact voltages required to produce a logic "0" or logic "1" depends upon the specific logic group or family. However, when using a standard +5 volt supply any TTL voltage input between 2.0V and 5V is considered to be a logic "1" or "HIGH" while any voltage input below 0.8v is recognized as a logic "0" or "LOW". TTL outputs are typically restricted to narrower limits of between 0 V and 0.4 V for a "low" and between 2.7 V and 5 V. The voltage region between the maximum voltage of logic "0" and minimum voltage of logic "1" of either input or output is called the *Indeterminate Region*. CMOS logic uses a different level of voltages with a logic "1" level operating between 3 and 15 volts.



TTL Input & Output Voltage Levels

There are several simple gates that you need to learn about. With these simple gates you can build combinations that will implement any digital component you can imagine.

- The simplest possible gate is called an "inverter," or a **NOT gate**. It takes one bit as input and produces output as its opposite. The logic table for NOT gate and its symbol are shown below.

NOT Gate

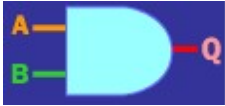
A	Q
0	1
1	0



- The **AND gate** performs a logical "and" operation on two inputs, A and B:

AND Gate


A	B	Q
0	0	0
0	1	0
1	0	0
1	1	1



- The **OR gate** performs a logical "or" operation on two inputs, A and B:

OR Gate


A	B	Q
0	0	0
0	1	1
1	0	1
1	1	1



- It is quite common to recognize two others as well: the **NAND** and the **NOR** gate. These two gates are simply combinations of an AND or an OR gate with a NOT gate.

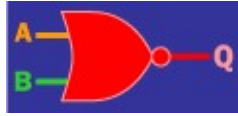
NAND Gate

A	B	Q
0	0	1
0	1	1
1	0	1
1	1	0



NOR Gate

A	B	Q
0	0	1
0	1	0
1	0	0
1	1	0



- The final two gates that are sometimes added to the list are the **XOR** and **XNOR** gates, also known as "exclusive or" and "exclusive nor" gates, respectively. Here are their tables:

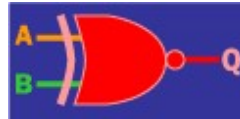
XOR Gate

A	B	Q
0	0	0
0	1	1
1	0	1
1	1	0



XNOR Gate

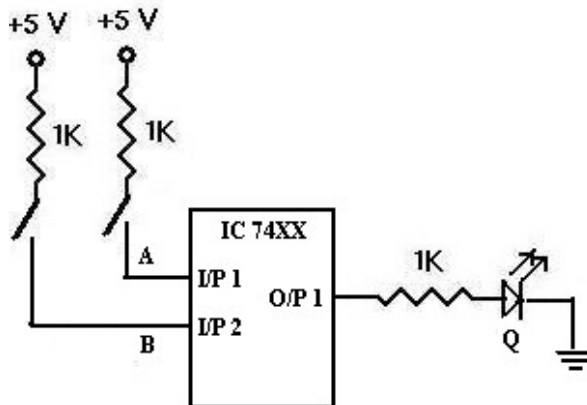
A	B	Q
0	0	1
0	1	0
1	0	0
1	1	1



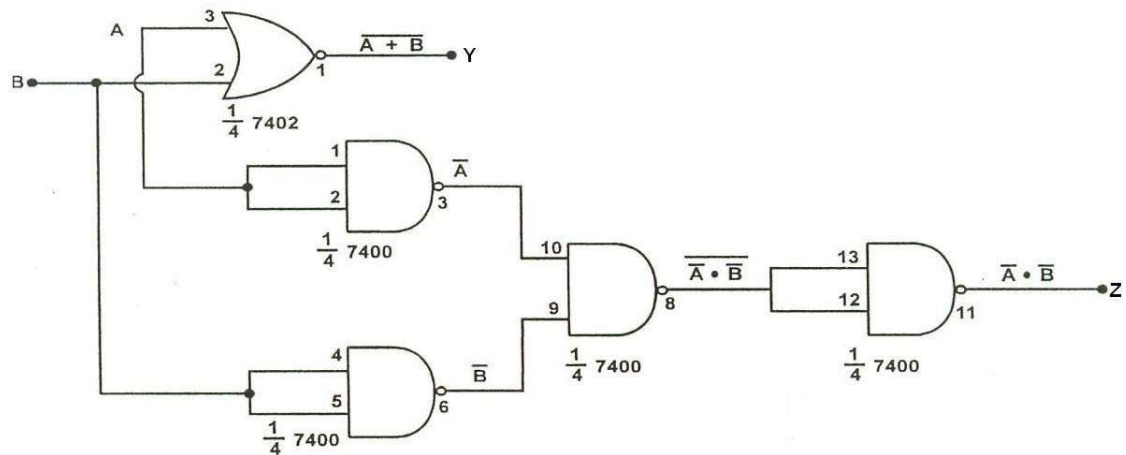
Circuit Components/Equipment:

- (i) Digital ICs, (ii) Resistors, (iii) DIP switch, (iv) D.C. Power supply (5V), (v) LEDs, (vi) Breadboard, (vii) Connecting wires.

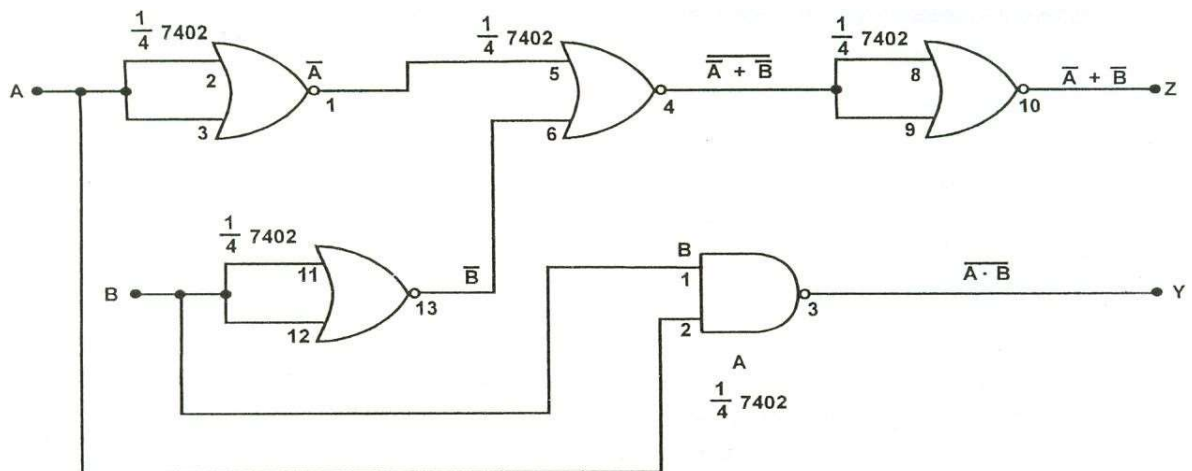
Circuit Diagrams: (in general, for all ICs)



De Morgan's law: $\overline{(\overline{A} + \overline{B})} = \overline{A} \cdot \overline{B}$



De Morgan's law: $\overline{(\overline{A} \cdot \overline{B})} = \overline{A} + \overline{B}$



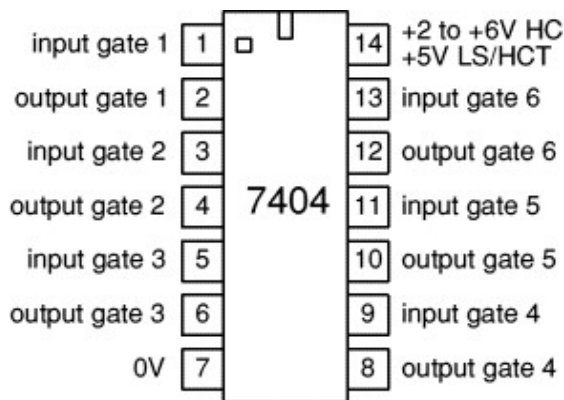
Procedure:

1. Place the IC on the breadboard.
2. Connect pin 14 (V_{CC}) to 5V and pin 7 (ground) to 0V terminal of the power supply.
3. Following the general circuit diagram facilitate all possible combinations of inputs from the power supply, using dip switch and resistors. Connect the output pin to ground through a resistor and LED.
4. Turn on power to your experimental circuit.
5. For each input combination, note the logic state of the outputs as indicated by the LEDs (ON = 1; OFF = 0), and record the result in the table.
6. Compare your results with the truth table for operation.

7. For verification of De Morgan's laws, follow the respective circuit diagrams using appropriate ICs. Follow the general circuit diagram for connections for input and output using dip switch and LEDs.
8. Monitor the outputs Y and Z using LEDs and confirm that Y and Z are the same for any states of A and B.
9. When you are done, turn off the power to your experimental circuit.

Observations: Verification of Boolean Logic Operations

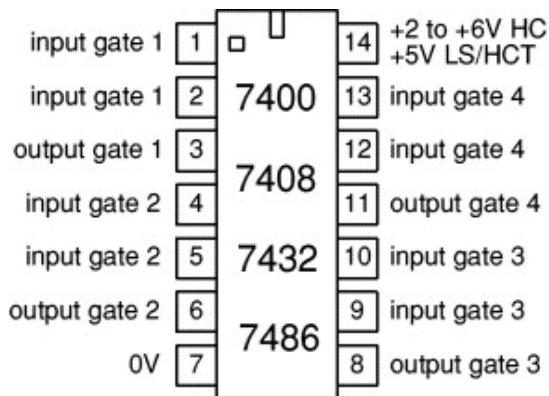
(I) Inverter gate (NOT): IC 7404LS



Gate # 1/2/./6

I/P (A)	O/P (Q)
0	
0	
1	
1	

(II) 2-input AND gate: IC 7408LS



Gate # 1/2/./4

I/P (A)	I/P (B)	O/P (Q)
0	0	
0	1	
1	0	
1	1	

(III) 2-input OR gate: IC 7432LS

Gate # 1/2/./4

I/P (A)	I/P (B)	O/P (Q)
0	0	
0	1	
1	0	
1	1	

(IV) 2-input EX-OR gate: IC 7486LS

Gate # 1/2/./4

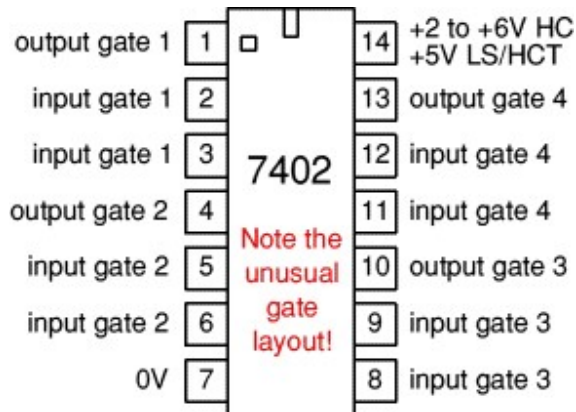
I/P (A)	I/P (B)	O/P (Q)
0	0	
0	1	
1	0	
1	1	

(V) 2-input NAND gate: IC 7400LS

Gate # 1/2/./4

I/P (A)	I/P (B)	O/P (Q)
0	0	
0	1	
1	0	
1	1	

(VI) 2-input NOR gate: IC 7402LS



Gate # 1/2/./4

I/P (A)	I/P (B)	O/P (Q)
0	0	
0	1	
1	0	
1	1	

I/P (A)	I/P (B)	$\overline{(\overline{A} + \overline{B})}$	Y(Observed)	\overline{A}	\overline{B}	$\overline{A} \cdot \overline{B}$	Z(Observed)
0	0						
0	1						
1	0						
1	1						

$$(\overline{A \cdot B}) = \overline{A} + \overline{B}$$

I/P (A)	I/P (B)	$(\overline{A \cdot B})$	Y(Observed)	\overline{A}	\overline{B}	$\overline{A} + \overline{B}$	Z(Observed)
0	0						
0	1						
1	0						
1	1						

Discussions:

Precautions:

Lab#5: Study of Adder and Subtractor Circuits

Objective:

- (i) To construct half and full adder circuit and verify its working
- (ii) To construct half and full subtractor circuit and verify its working
- (iii) To construct a full adder-subtractor circuit

Overview:

Half adder:

Let's start with a **half (single-bit) adder** where you need to add single bits together and get the answer. The way you would start designing a circuit for that is to first look at all of the logical combinations. You might do that by looking at the following four sums:

$$\begin{array}{r} 0 \quad 0 \quad 1 \quad 1 \\ +0 \quad +1 \quad +0 \quad +1 \\ \hline 0 \quad 1 \quad 1 \quad 10 \end{array}$$

That looks fine until you get to $1 + 1$. In that case, you have a **carry bit** to worry about. If you don't care about carrying (because this is, after all, a 1-bit addition problem), then you can see that you can solve this problem with an XOR gate. But if you do care, then you might rewrite your equations to always include **2 bits of output**, like this:

$$\begin{array}{r} 0 \quad 0 \quad 1 \quad 1 \\ +0 \quad +1 \quad +0 \quad +1 \\ \hline 00 \quad 01 \quad 01 \quad 10 \end{array}$$

Now you can form the logic table:

1-bit Adder with Carry-Out

A	B	Q	C
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

By looking at this table you can see that you can implement the sum Q with an XOR gate and C (carry-out) with an AND gate.

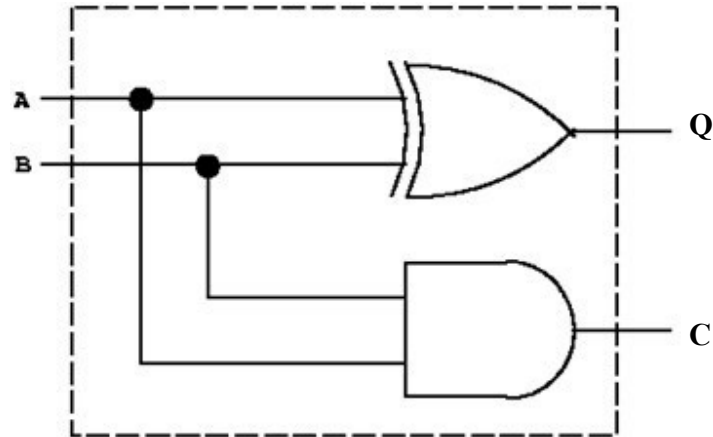


Fig. 1: Schematics for half adder circuit

Full adder:

If you want to add two or more bits together it becomes slightly harder. In this case, we need to create a full adder circuits. The difference between a full adder and a half adder we looked at is that a full adder accepts inputs A and B plus a **carry-in** (C_{N-1}) giving outputs Q and C_N . Once we have a full adder, then we can string eight of them together to create a byte-wide adder and cascade the carry bit from one adder to the next. The logic table for a full adder is slightly more complicated than the tables we have used before, because now we have **3 input bits**. The truth table and the circuit diagram for a full-adder is shown in Fig. 2. If you look at the Q bit, it is 1 if an odd number of the three inputs is one, i.e., Q is the XOR of the three inputs. The full adder can be realized as shown below. Notice that the full adder can be constructed from two half adders and an **OR** gate.

One-bit Full Adder with Carry-In & Carry-Out

C_{N-1}	A	B	Q	C_N
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

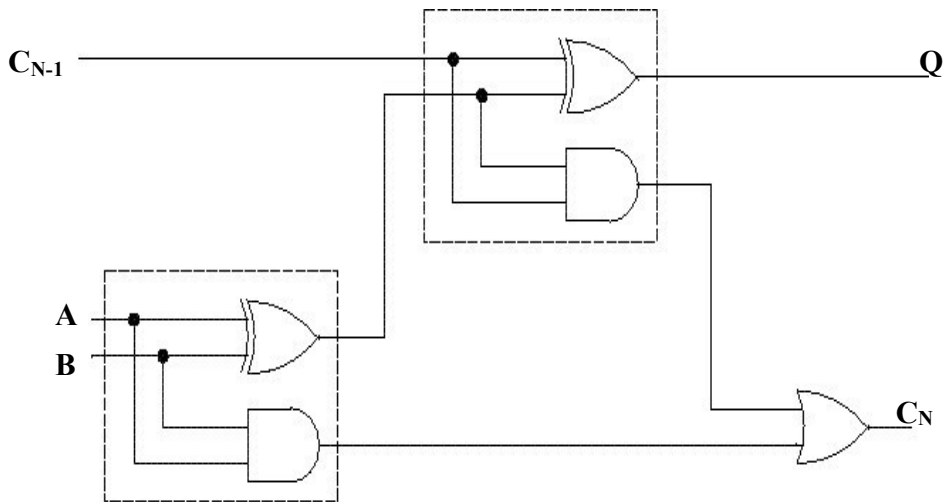


Fig. 2: Truth table and schematics for full adder circuit

Now we have a piece of functionality called a "full adder", which can be combined with a half adder to construct a 2-bit adder. Adders for arbitrarily large (say N-bit) binary numbers can be constructed by cascading full adders. These are called a **ripple-carry** adder, since the carry bit "ripples" from one stage to the next. The schematics for a 4-bit full adder circuit is shown below. This implementation has the advantage of simplicity but the disadvantage of speed problems. In a real circuit, gates take time to switch states (the time is of the order of nanoseconds, but in high-speed computers nanoseconds matter). So 32-bit or 64-bit ripple-carry adders might take 100 to 200 nanoseconds to settle into their final sum because of carry ripple.

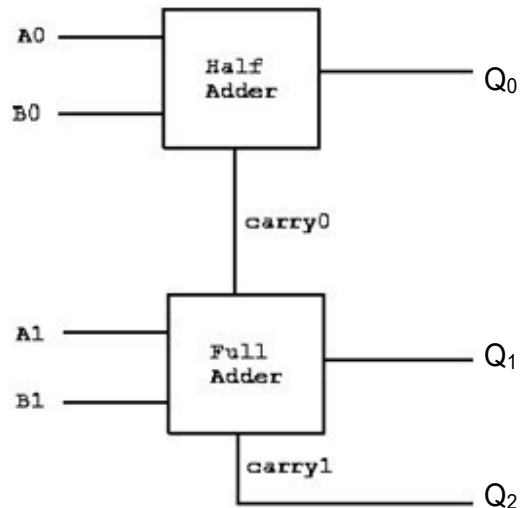


Fig. 3: Schematics of 2-bit adder

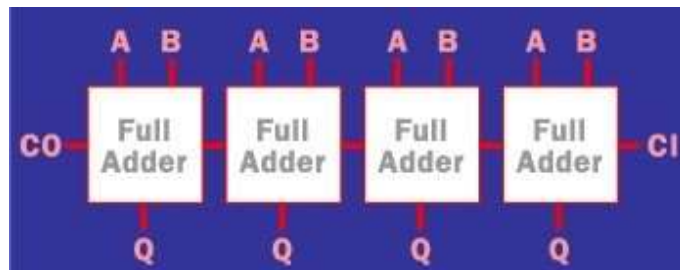


Fig.4: Schematics of 4-bit adder

Subtraction:

In a similar fashion subtraction can be performed using binary numbers. The truth table for a single bit or half-subtractor with inputs A and B is given below along with its circuit diagram (Fig.5). A full subtractor circuit accepts a minuend (A) and the subtrahend (B) and a borrow (B_{IN}) as inputs from a previous circuit. A full subtractor circuit can be realized by combining two half subtractor circuits and an OR gate as shown in Fig. 6.

1-bit Subtractor with Borrow

A	B	Q	B_{IN}
0	0	0	0
0	1	1	1
1	0	1	0
1	1	0	0

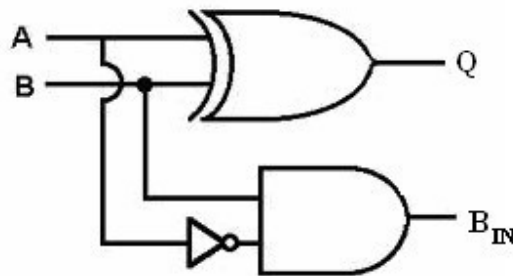


Fig. 5: Truth table and schematics for half subtractor circuit

1-bit Full Subtractor with B_{N-1} & B_N

B_{N-1}	A	B	Q	B_N
0	0	0	0	0
0	0	1	1	1
0	1	0	1	0
0	1	1	0	0
1	0	0	1	1
1	0	1	0	1
1	1	0	0	0
1	1	1	1	1

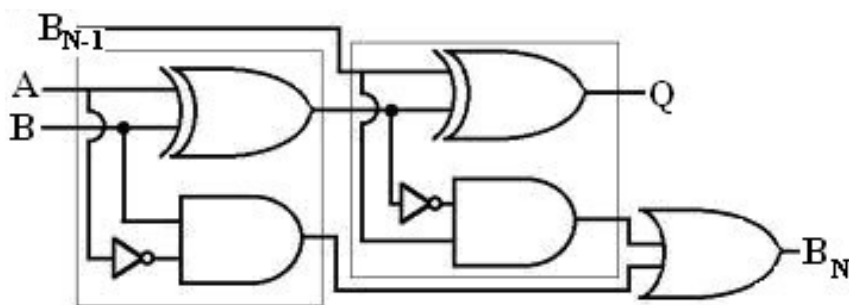


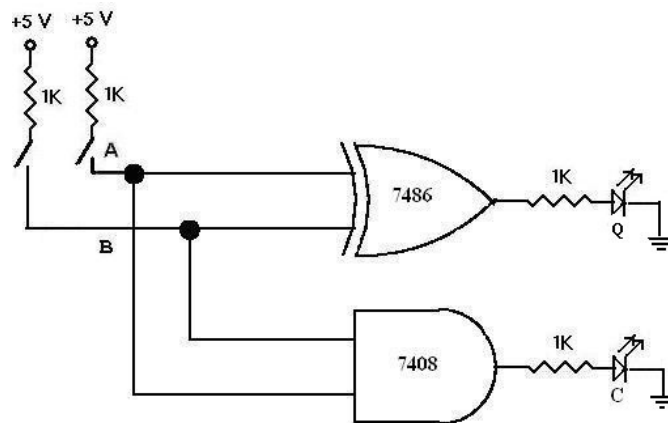
Fig. 6: Truth table and schematics for full subtractor circuit

However, it is possible to use the same circuit to perform addition and subtraction by replacing the 'NOT' gate of the subtractor circuit by an 'XOR' as shown in the circuit diagram below. Here, the second input (first input is from supply) for XOR gate decides the function of the circuit, i.e. addition or subtraction. This means if the second input for XOR is 0, the circuit will do addition and if 1, it will do subtraction.

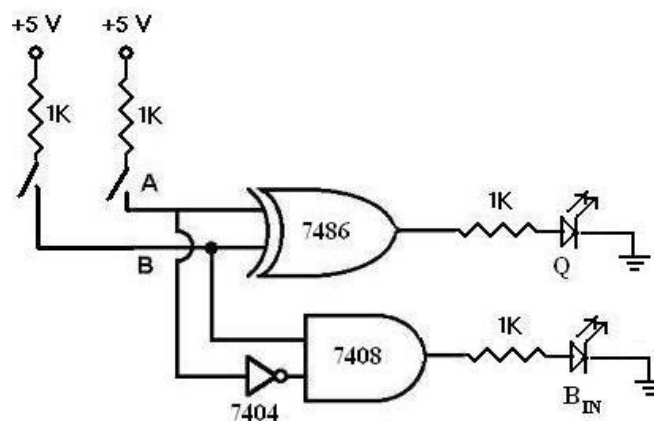
Circuit components/Equipment:

1. Resistors (1K Ω , 5 Nos)
2. ICs (XOR-7486, AND-7408, OR-7432, NOT-7404)
3. A Surface mount dip switch
4. D.C. Power supply (5V)
5. Red/Green LEDs (2 Nos)
6. Connecting wires
7. Breadboard

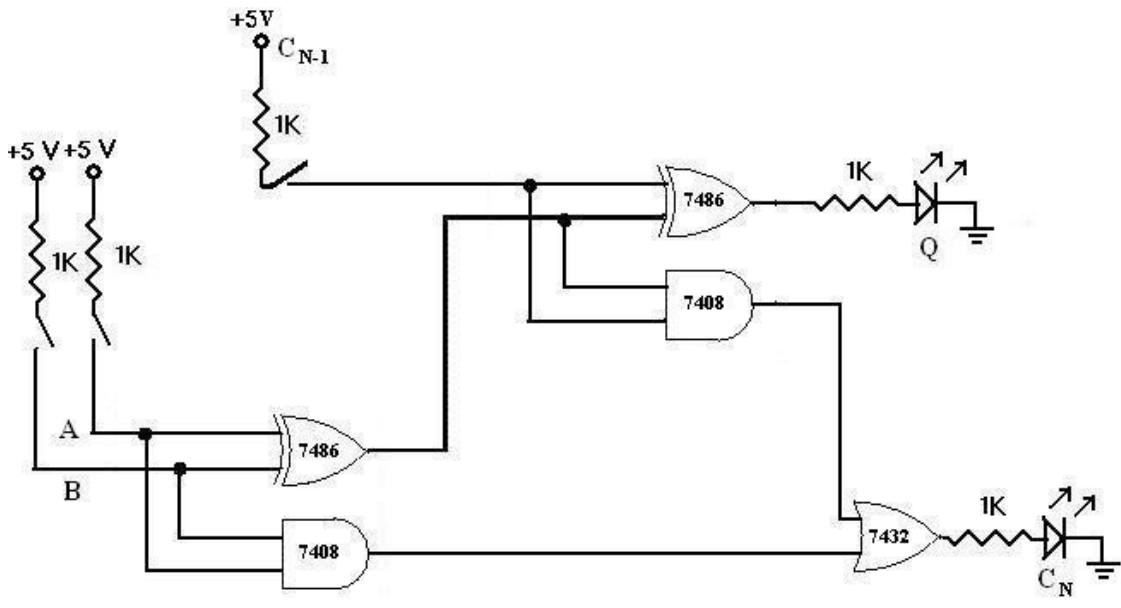
Circuit Diagrams:



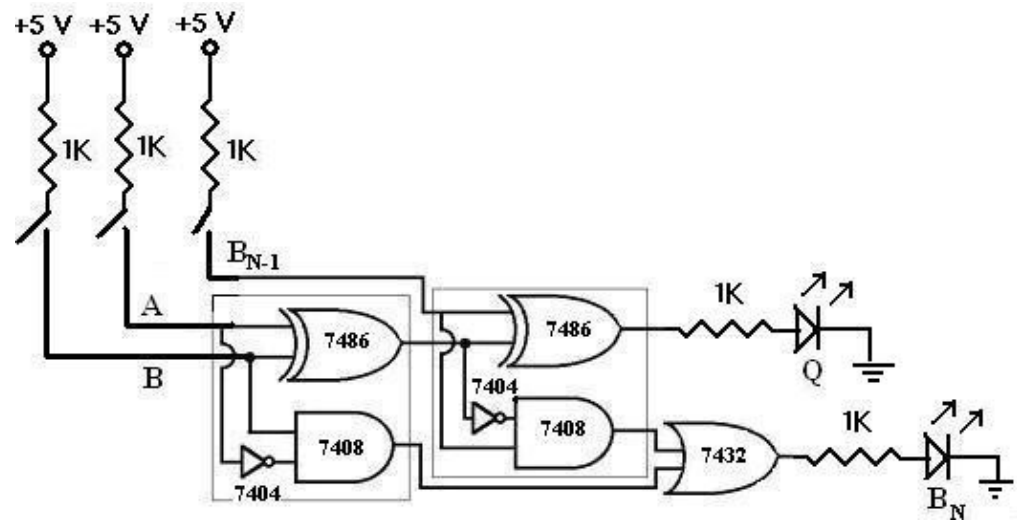
Half Adder



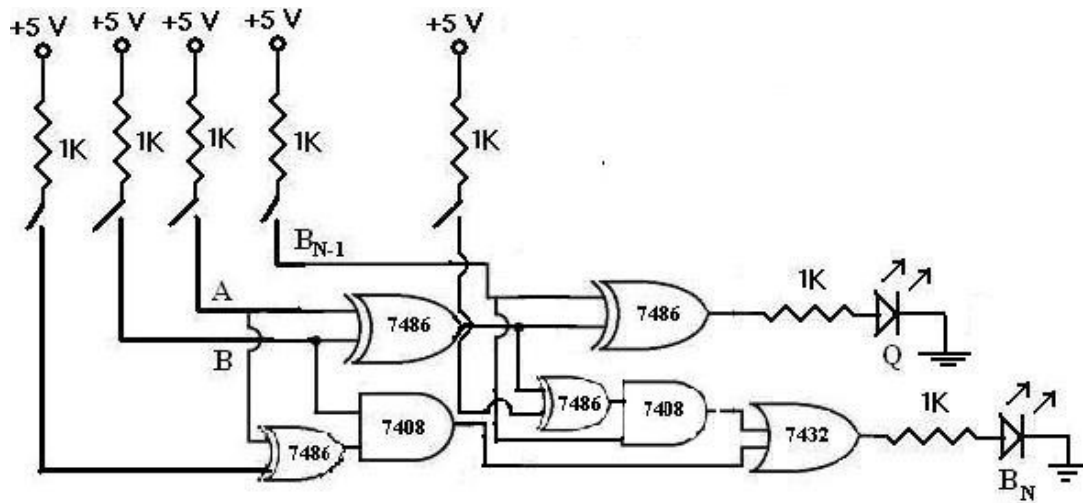
Half Subtractor



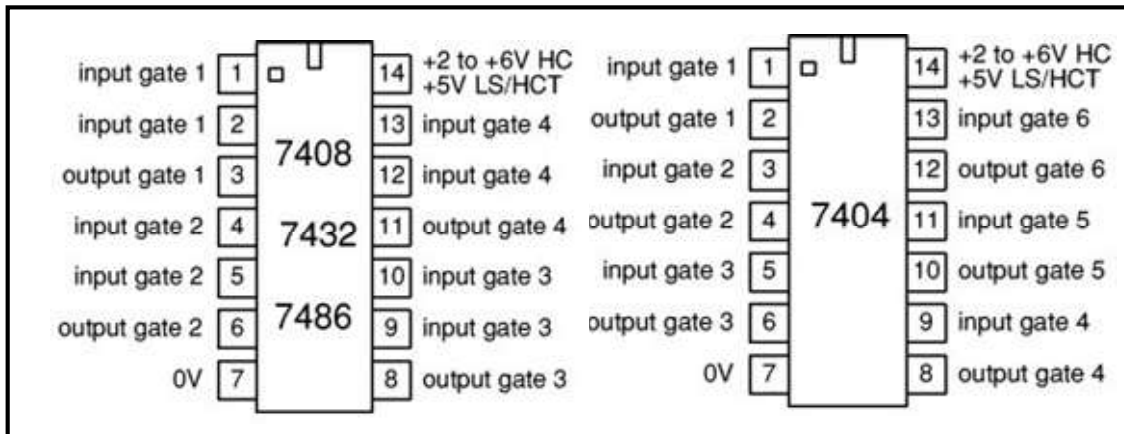
Full Adder



Full Subtractor



Full Adder-Subtractor



Schematics for detailed pin connections for different ICs

Procedure:

1. Assemble the circuits one after another on your breadboard as per the circuit diagrams.
2. Connect the ICs properly to power supply (pin 14) and ground (pin 7) following the schematics for different ICs shown above.
3. Using dip switch and resistors, facilitate all possible combinations of inputs from the power supply.
4. Turn on power to your experimental circuit.
5. For each input combination, note the logic state of the outputs as indicated by the LEDs (ON = 1; OFF = 0), and record the result in the table.
6. Compare your results with the truth table for operation.
7. When you are done, turn off the power to your experimental circuit.