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

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:

0	0	1	1
+ 0	+ 1	+ 0	+ 1
0	1	1	10

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:

0	0	1	1
+ 0	+ 1	+ 0	+ 1
00	01	01	10

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.

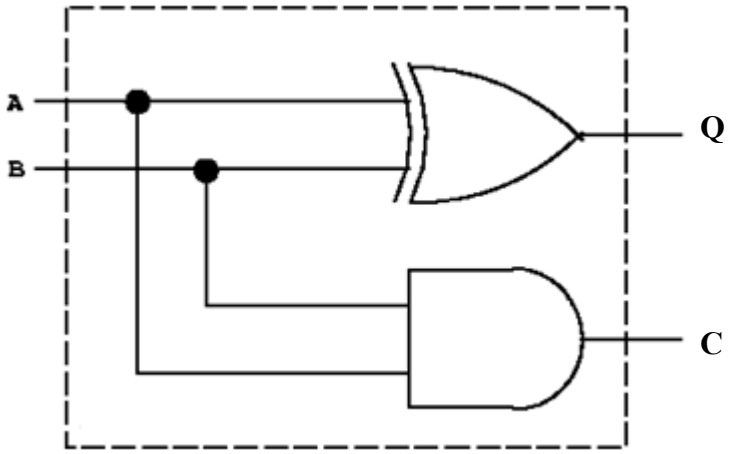


Fig. 1: Schematics for half adder circuit

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

.

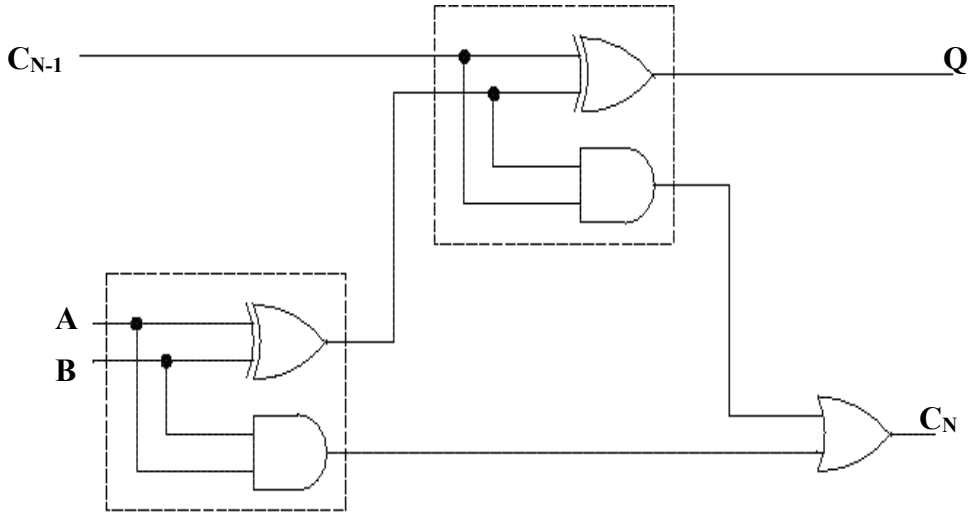


Fig. 2: Truth table and schematics for full adder circuit

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.

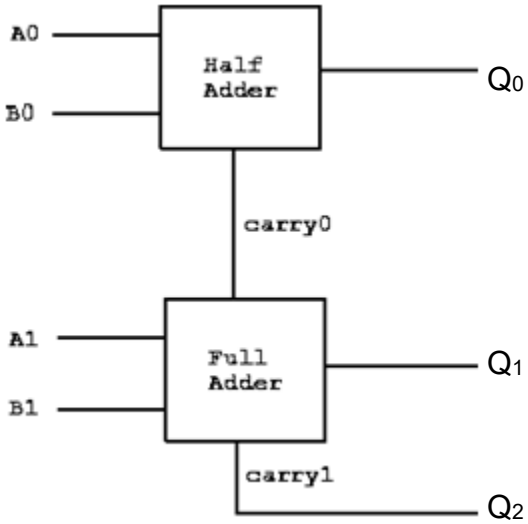


Fig. 3: Schematics of 2-bit adder

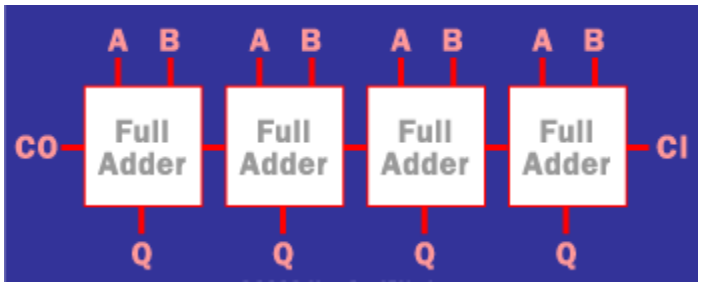


Fig.4: Schematics of 4-bit adder

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

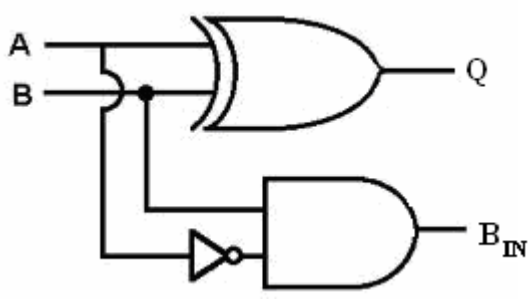


Fig. 5: Truth table and schematics for half subtractor circuit

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

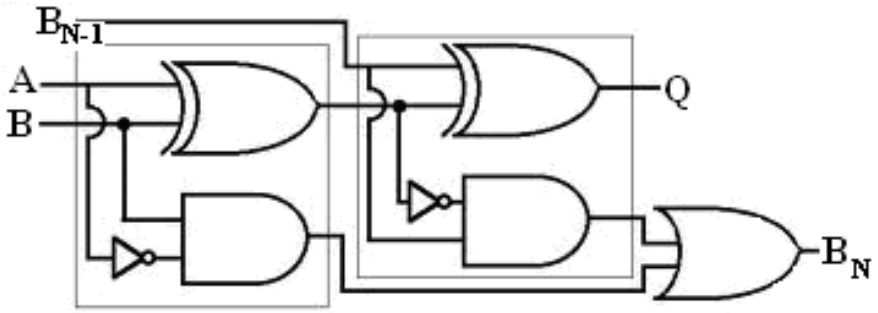


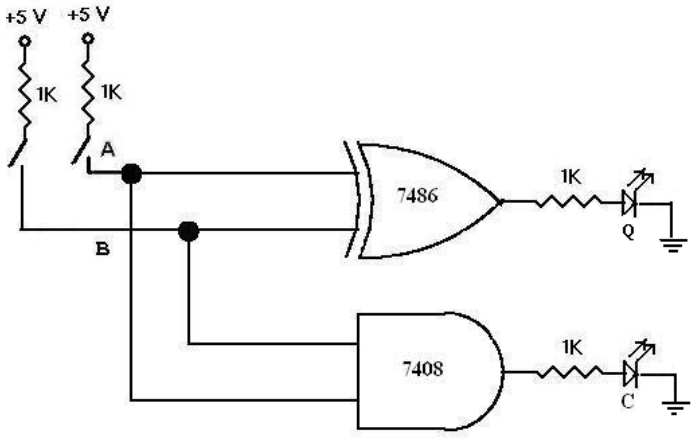
Fig. 6: Truth table and schematics for full subtractor circuit

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.

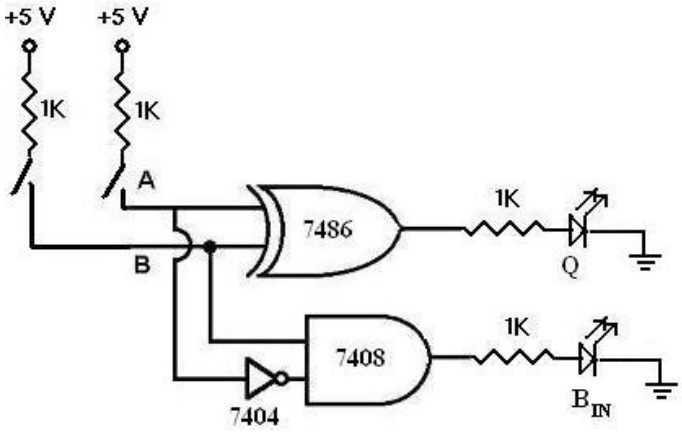
Circuit components/Equipments:

1. Resistors (1KΩ, 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

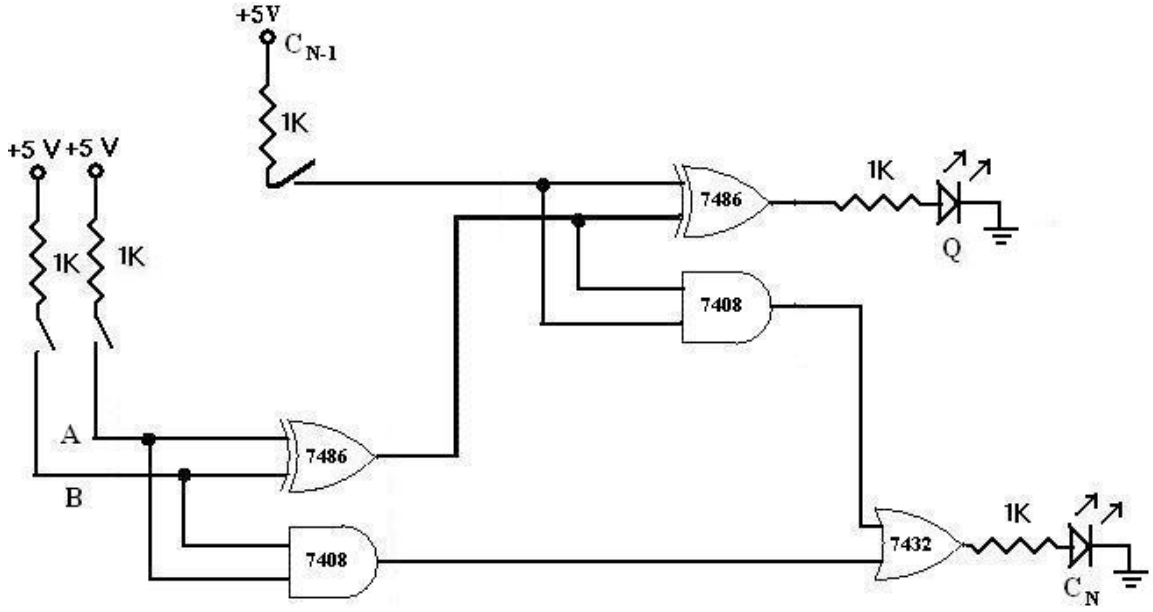
Circuit Diagrams:



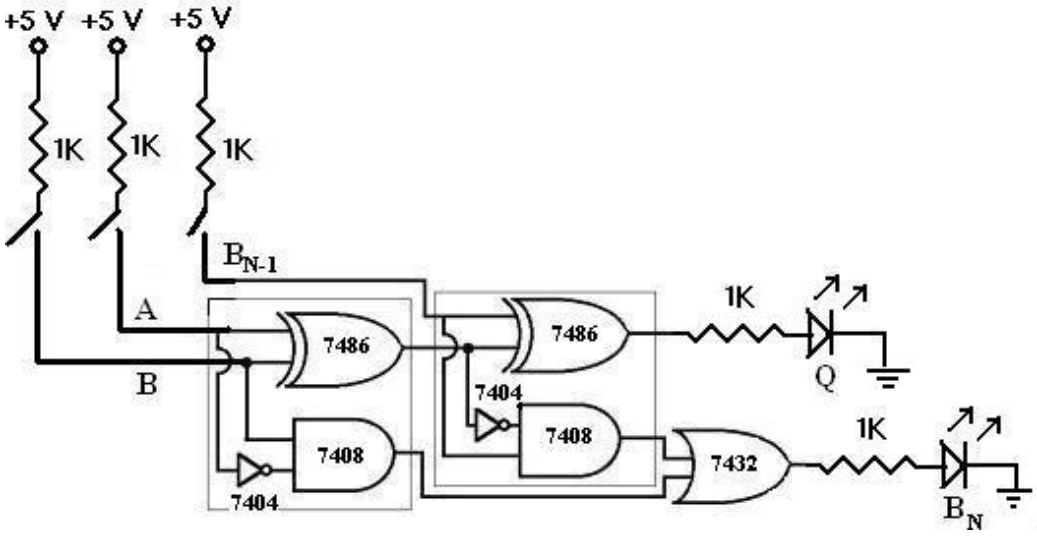
Half Adder



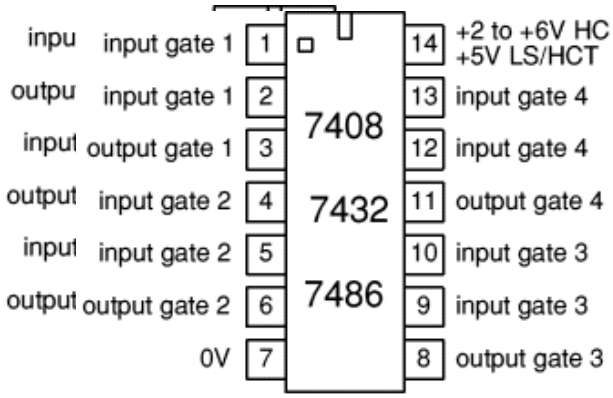
Half Subtractor



Full Adder



Full Subtractor



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.

Observations:

(i) Half Adder:

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

(iii) Half Subtractor:

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

(ii) Full Adder:

C_{N-1}	A	B	Q	C_N
0	0	0		
0	0	1		
0	1	0		
0	1	1		
1	0	0		
1	0	1		
1	1	0		
1	1	1		

(iv) Full Subtractor:

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

Discussions:

Precautions: