

Design of a Combinational Circuit (BCD to 7-Segment Decoder) – Reference Material

Instructor:-

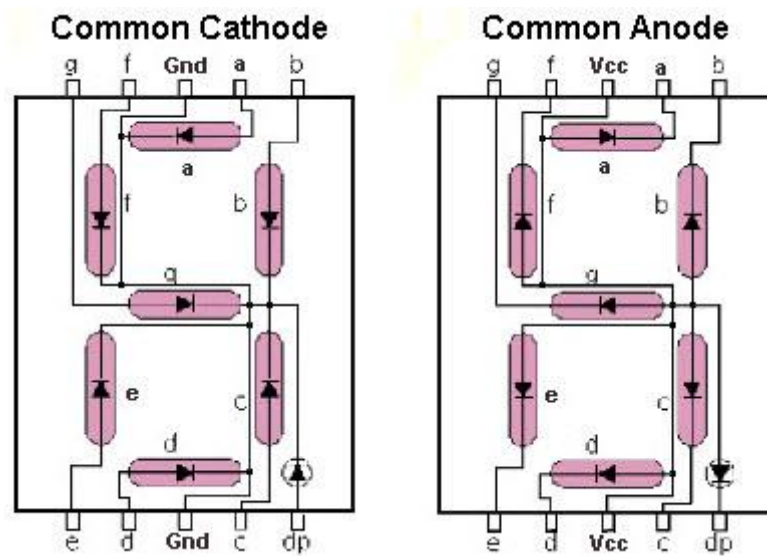
Muhammad Asif Hasan

E: sasifhasan@hotmail.com











School of Electrical Engineering and Computer Sciences
(SEECS)

H-12 Campus, National University of Sciences and
Technology (NUST)

Seven Segment Display (SSD)

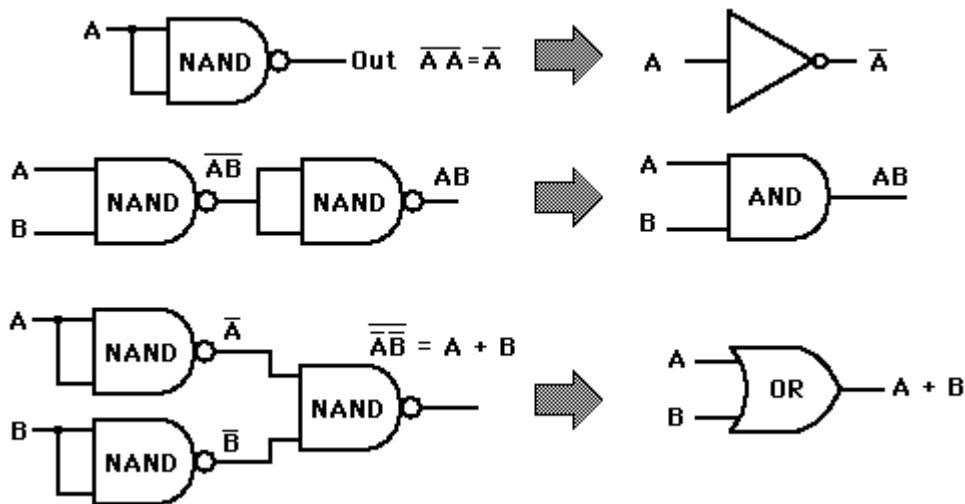


Decoder Table for SSD

Decimal Digit	Input lines				Output lines							Display pattern
	A	B	C	D	a	b	c	d	e	f	g	
0	0	0	0	0	1	1	1	1	1	1	0	
1	0	0	0	1	0	1	1	0	0	0	0	
2	0	0	1	0	1	1	0	1	1	0	1	
3	0	0	1	1	1	1	1	1	0	0	1	
4	0	1	0	0	0	1	1	0	0	1	1	
5	0	1	0	1	1	0	1	1	0	1	1	
6	0	1	1	0	1	0	1	1	1	1	1	
7	0	1	1	1	1	1	1	0	0	0	0	
8	1	0	0	0	1	1	1	1	1	1	1	
9	1	0	0	1	1	1	1	1	0	1	1	

Implementation of Boolean functions using Universal logic gates

'Universal logic gates' are NAND gate and NOR gates. The reason behind this is, NAND gate and NOR gate can perform (or can function like) all the 3 basic gates, such as AND gate, OR gate and NOT gate. We can design any basic logic gate by using NAND gate or NOR gate. This is why they are called as "Universal gates".



Let's see the implementation of the Boolean functions using universal logic gates.

Implementation of Boolean functions using NAND gates

NAND gate is a logical combination of AND gate and NOT gate and this can function like AND gate, OR gate and NOT gate. So we use NAND gates to implement the Boolean function.

The important thing to remember about NAND gate is this is the inverse of basic AND gate. This means the output of the NAND gate is equal to the complement of the output of the AND gate.

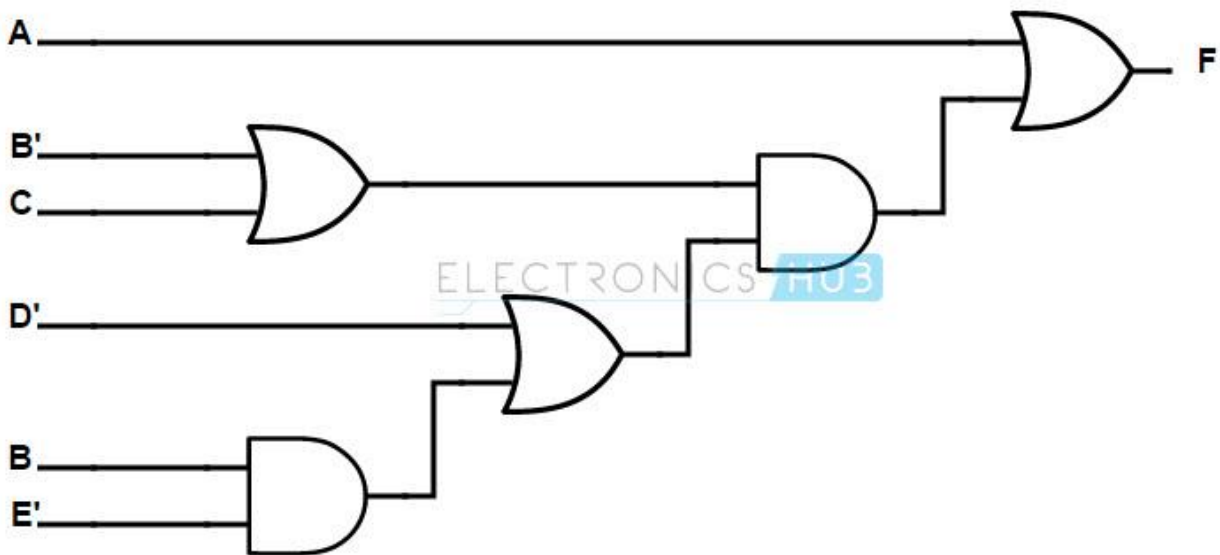
Let's see an example to understand the implementation.

Implement the Boolean function by using a NAND logic gate.

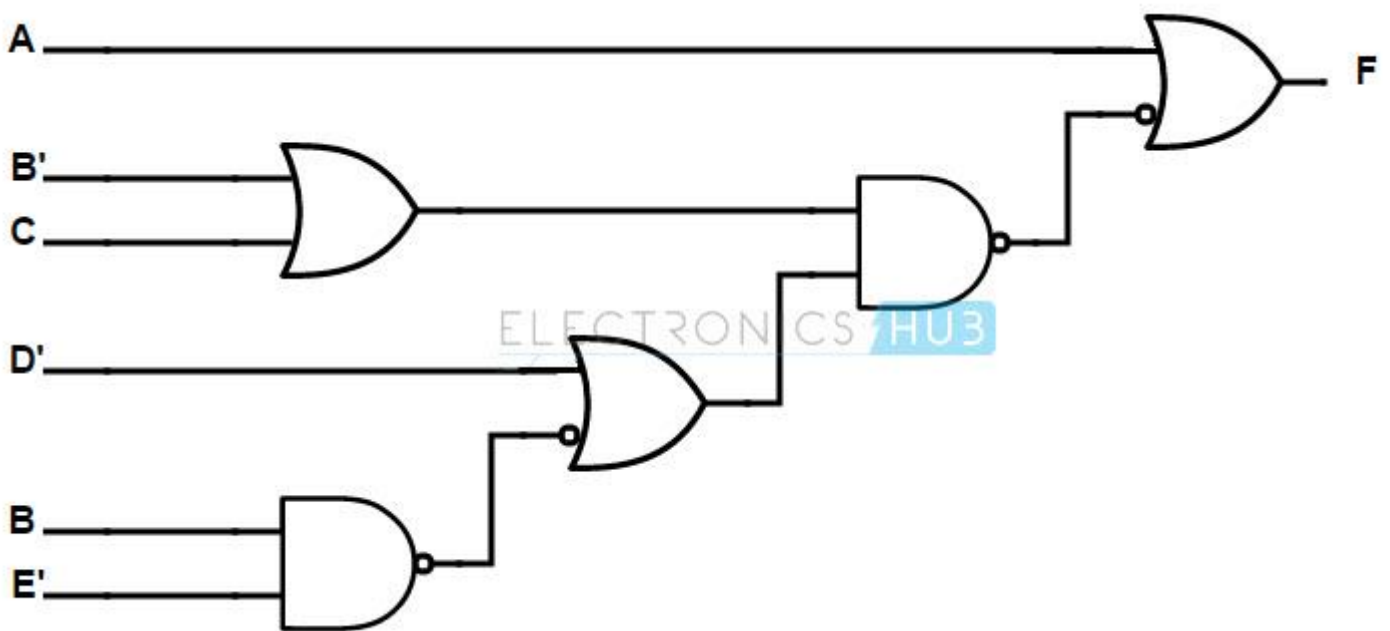
$$F(A, B, C, D, E) = A + (B' + C)(D' + BE')$$

In NAND gate implementation, we use NAND gates at both input and output side. Observe the designed logic diagram below. The step by step procedure to implement the given Boolean function using NAND gates is shown below.

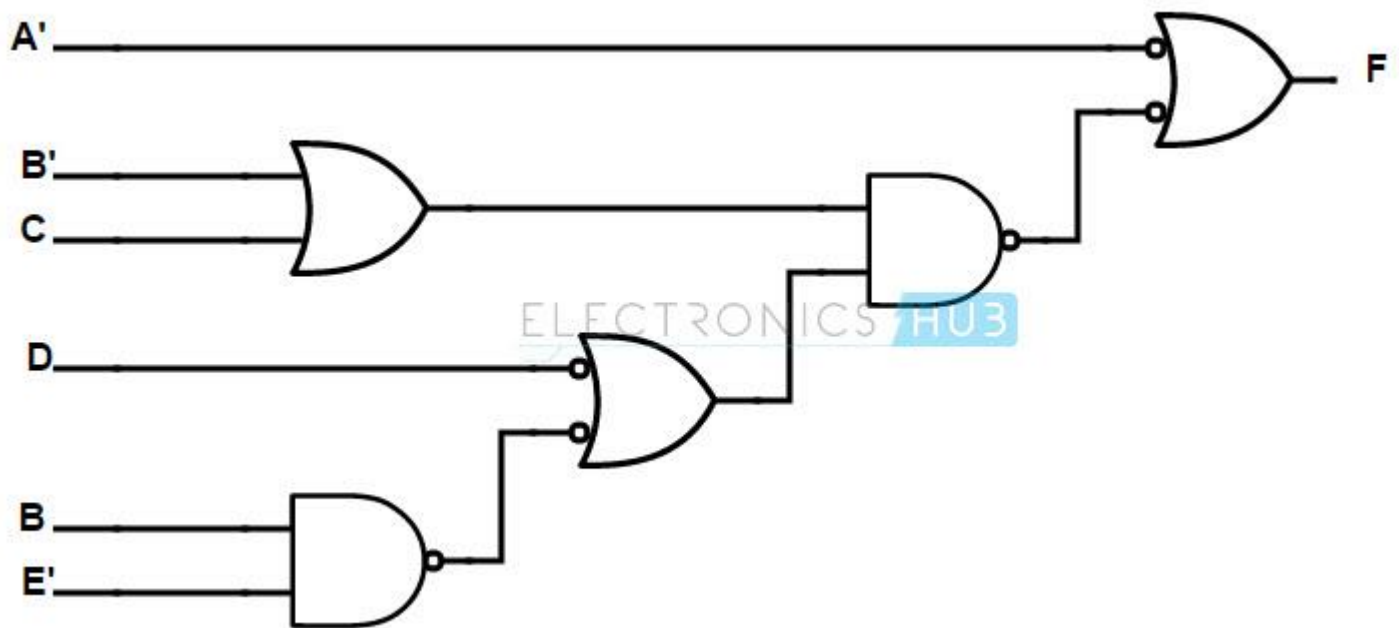
First, the given Boolean function or equation should be represented using AND-OR gates. The AND-OR implementation is shown below.



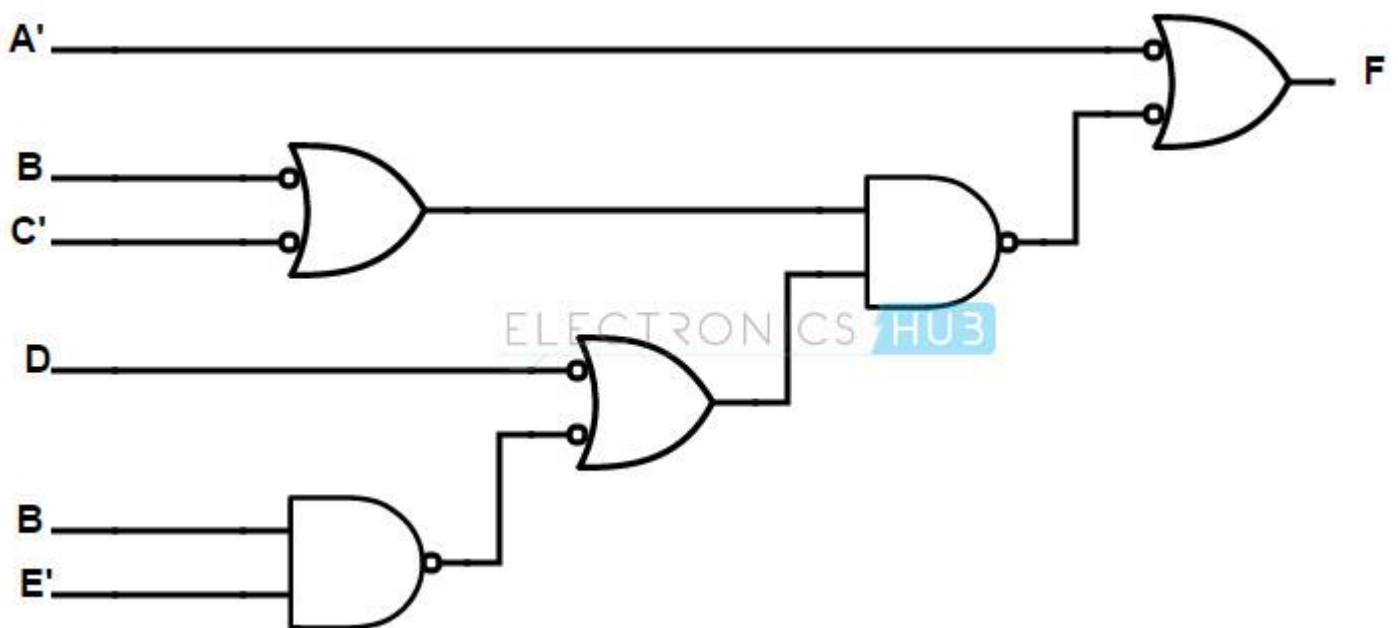
In order to convert the AND gates into NAND gates, a bubble (complement) is introduced at the output of the AND gate. To compensate the bubble, the input of the next gate is also introduced with a bubble. The implementation is shown below.



To impose uniformity at the input, if a gate has one input with a bubble, the other input is also introduced with a bubble. Again, in order to compensate the bubble, the output of the preceding gate is introduced with bubble or complement the literal. The same is shown in the following figure.



If an OR gate is not having any bubble at either of the inputs, bubbles are introduced and are appropriately compensated as shown in the figure below.



An OR gate with two complemented inputs is equivalent to a NAND gate (according to DeMorgan's Law $A' + B' = (AB)'$). Hence, replacing the OR gate, which is having two complemented inputs, with NAND gate, we get the final structure of the implementation of the Boolean function using NAND gates. The final implementation is shown below.

