2.0. Realization of a Boolean Function

Objective:

- To simplify the Boolean expression and to build the logic circuit.
- To explore the notion of combinational circuits and basic combinational design.

2.1. Introduction

Combinational circuits are more frequently constructed with NAND or NOR gates rather than AND and OR gates. *NAND and NOR gates are more common from the hardware point of view because they are readily available in integrated-circuit form.*

The NAND gate is said to be a universal gate because any digital system can be implement with it. Combinational circuits and sequential circuits as well can be constructed with this gate because the flip-flop circuit (the memory element most frequently used in sequential circuits) can be constructed from two NAND gates.

When a Boolean expression is implemented with logic gates, each term requires a gate, and each variable within the term designates an input to the gate. We define a literal as a single variable within the term that may or may not be complemented.

By reducing the number of terms, the number of literals, or both in a Boolean expression, it is often possible to obtain a simpler circuit. Boolean algebra or K-map is applied to reduce an expression for the purpose of obtaining a simpler circuit.

In the sum of minterms canonical form, every product term includes a literal of every variable of the function.

Product terms of the SOP form which do not include a literal of a variable, say variable B, should be augmented by,

- ✓ AND-ing the product term which misses a literal of B with $(B+\bar{B})$,
- ✓ Subsequently applying the distributive property to eliminate the parenthesis.

In the product of maxterms canonical form, every sum term includes a literal of every variable of the function. Sum terms of the POS form which do not include a literal of a variable, say variable B, ought to be augmented by

- ✓ OR-ing the sum term with $B \cdot \bar{B}$,
- ✓ Subsequently applying postulate $a + b \cdot c = (a + b) \cdot (a + c)$ to distribute the product $B \cdot \bar{B}$

2.2. Realization of Boolean Expression

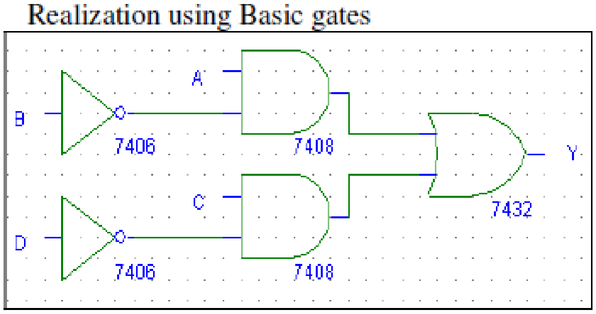
$$Y = A'B'CD' + A'BCD' + ABCD' + AB'CD' + AB'C'D' + AB'C'D + AB'CD$$

K-Map method we get;

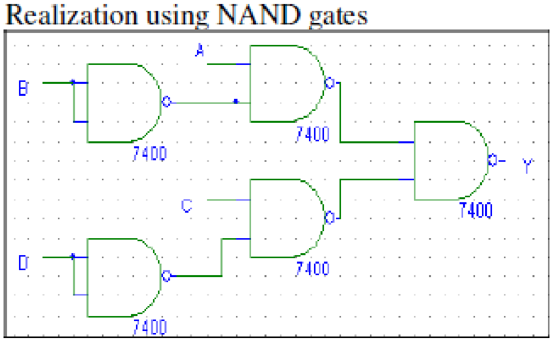
			1
			1
			1
1	1	1	1

$$Y = AB' + CD'$$

Realization using Basic gates

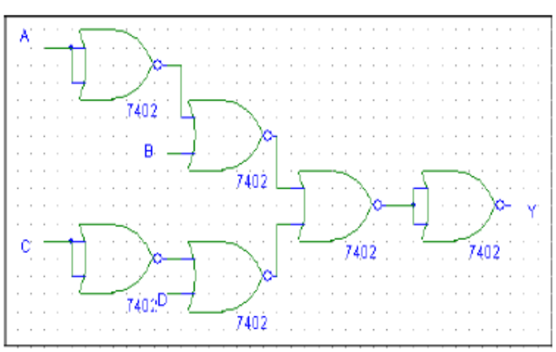


Realization using NAND gates



TRUTH TABLE

INPUTS				OUTPUT
A	B	C	D	Y
0	0	0	0	0
0	0	0	1	0
0	0	1	0	1
0	0	1	1	0
0	1	0	0	0
0	1	0	1	0
0	1	1	0	1
0	1	1	1	0
1	0	0	0	1
1	0	0	1	1
1	0	1	0	1
1	0	1	1	1
1	1	0	0	0
1	1	0	1	0
1	1	1	0	1
1	1	1	1	0



Example

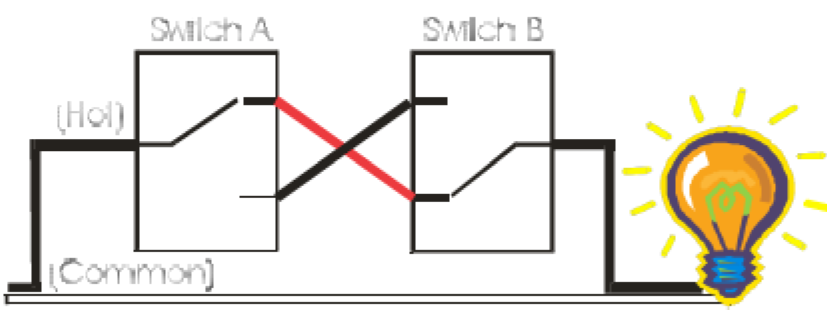
1. **Designing a Digital Light Control**

Consider the problem of constructing a light controller for a certain room in a house. It is desirable for the light to be switched on if:

i. A Burglar Alarm detects an intruder
ii. A Master Light Switch is on, **or**
iii. An Auxiliary Switching system is active, **and** a person(s) is/are present in the room.

Item 3 requires further consideration. First of all, how is the system going to know if a person is in the room? A motion and/or sound detector could be used to produce a "logic 1" (Boolean True) if a person is detected.

The auxiliary switches mentioned above could be the wall switches already found in the room. Assuming that the room has two doors, then a (*three-way*) switch at each door would be convenient. In this configuration, the light is off if both switches are up or both are down, and it is on if one switch is up and the other is down. This allows for the light to be switched no matter what the state of the switches is



For the lights to come on in our design, not only must one switch be up and one down, but a person must also be detected (unless one of the other conditions [Burglar Alarm or Master Switch] turns them on). Note that the "person detector" would probably have a timer which keeps the output high for a designated time after a person is detected.

Basically, what we need is a circuit which will switch the lights on if (and only if)

The Burglar Alarm is On
OR
The Master Switch is On
OR
A person is detected AND one but not both auxiliary switches is up.

Now let's assign some variable names to the various switches (inputs) so that we can write an equation to describe the desired binary function.

Let

B = Burglar Alarm

M = Master Switch

P = Person Detector

A_1 = Auxiliary Switch 1, and

A_2 = Auxiliary Switch 2

Note that the necessary condition of A_1 and A_2 to activate the lights is an *Exclusive OR* function (one, but not both). Using the XOR symbol ⊕, we can write the desired lighting function, F_L as

$$F_L = B + M + P \cdot (A_1 \oplus A_2)$$

A circuit for this function can be drawn as;