Syllabus of the course SM2:

Chapter 1: Combinational circuits

- Introduction
- Definition of combinational circuits
- Steps of Construction of Combinational Circuits
- Half Adder
- Full Adder
- Subtractor
- Encoder / decoder /Priority encoder
- Multiplexer / demultiplexer
- Converters/ Comparator

Chapter 2: Sequential circuits

- Introduction
- Definition of sequential circuits
- Flip-flops (RS, JK, D, T)
- The registers
- The Counters
- Automata of Moore and Mearly

Chapter 1: Combinational circuits

1. Introduction:

The basic data handled by the physical machine is the Bit (Binary Digit) which can only take two values (0 or 1). These 0 and 1 correspond to the two voltage levels (0-1 and 2-5). So, logical circuits are the basis of all electronic material, especially computers. There are two types of logic circuits:

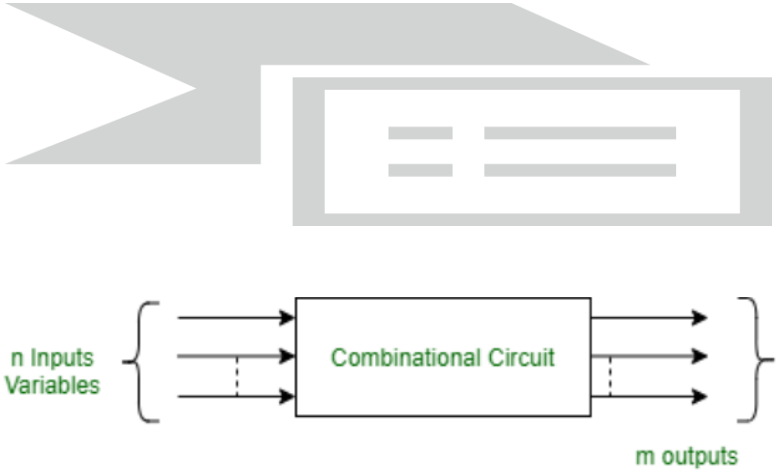
a) Combinational Logic Circuits
b) Sequential Logic Circuits

2. Definition of combinational circuit

It is a type of circuit that generates an output by relying on the input it receives at that instant, and it stays independent of time.

Combinational circuits are defined as the time independent circuits which do not depends upon previous inputs to generate any output are termed as combinational circuits.

The output of a Combinational Circuit depends entirely on the present input and does not depend on any of its previous inputs. No feedback is present between the input and output. Logic gates form the building blocks of such circuits.

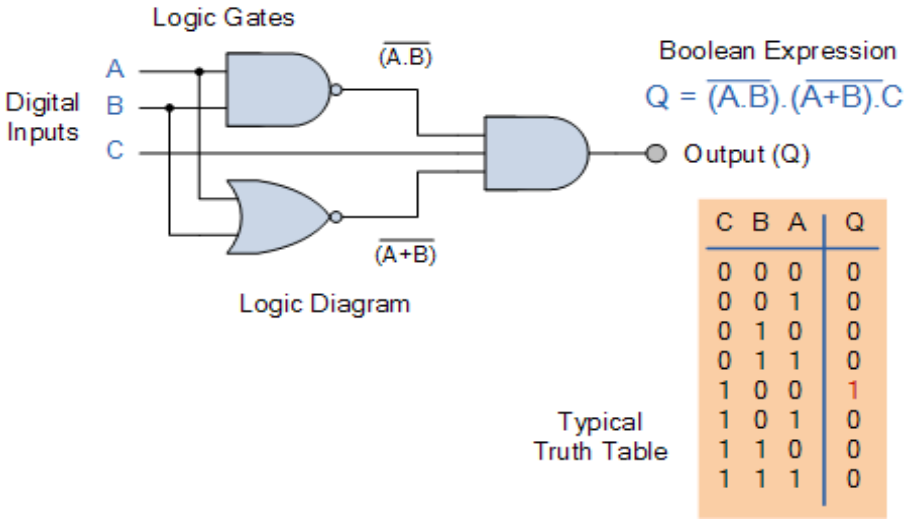


Examples:

Adder, Encoder, Decoder, Multiplexer, Demultiplexer, …etc.

The three main ways of specifying the function of a combinational logic circuit are:

1. **Boolean Algebra** – This forms the algebraic expression showing the operation of the logic circuit for each input variable either True or False that results in a logic "1" output.
2. **Truth Table** – A truth table defines the function of a logic gate by providing a concise list that shows all the output states in tabular form for each possible combination of input variable that the gate could encounter.
3. **Logic Diagram** – This is a graphical representation of a logic circuit that shows the wiring and connections of each individual logic gate, represented by a specific graphical symbol, that implements the logic circuit. All three of these logic circuit representations are shown below.



3. Steps of construction of Combinational Circuits

A Combinational Circuit consist of logic gates whose outputs at any instant of time are determined directly from the present combination of inputs without regard to previous input. Here we are going to learn how to construct and analyze any type of combinational circuit using four general steps:

1) Study logically the problem and identify the number of inputs and outputs of the circuit: First of all, we have to think about the inputs and outputs of the circuit by considering which type of logical operation we want to perform with the circuit. For example, we have to create a circuit that can add two bits. For this, we require two inputs (one for the first bit (A) another for the second bit (B)) and two outputs one for sum (S) of two bits and another for carry (C).

2) Create the Truth Table: In this step we have to create truth table for our circuit so for this first we will create input columns and list all the possible combinations of inputs. In our case 2 bits can have maximum 4 combinations (00 01 10 11). Now in output, we have two columns (Sum and Carry) as discussed earlier. Now we have to fill output columns in such a way that for which logical operation we are constructing circuit. In our circuit, we want addition so we will add those input bits and write the sum of those bits in (Sum) column and if carry is generated we will write 1, else write **0** in (Carry) column.
3) Simplify the Boolean function for each output: In this step, we have to just create a simplified Boolean function according to inputs and outputs of the truth table obtained in the previous step.
4) Construct circuit using Boolean function obtained from third step.

3. Design procedure of a Combinational Circuit

The design procedure of a combinational circuit involves the following steps:

1) The problem is stated and studied logically.
2) The total number of available input variables and required output variables is determined.
3) The input and output variables are allocated with letter symbols.
4) The exact truth table that defines the required relationships between inputs and outputs is derived.
5) The simplified Boolean function is obtained from each output.
6) The logic diagram is drawn.

Example: We want to design a combinational circuit which calculates the result (Quotient) of division by 3 of a pure binary number represented on 3 bits ($a_2a_1a_0$). Such as the Quotient is ($Q_0Q_1Q_2$ ….).

Solution of the example:

- The possible input states are: {000,001,010,011,100,101,110,111}
- The number of outputs:

(000) : 0/3 = 0 $(00)_2$ (001) : 1/3 = 0 $(00)_2$

(010) : 2/3 = 0 $(00)_2$ (011) : 3/3 = 1 $(01)_2$

(100) : 4/3 = 1 $(01)_2$ (101) : 5/3 = 1 $(01)_2$

(110) : 6/3 = 2 $(10)_2$ (111) : 7/3 = 2 $(10)_2$

So the output of 2 Bits (Q_0Q_1).

- **Truth table**

a_2	a_1	a_0		Q_0	Q_1
0	0	0		0	0
0	0	1		0	0
0	1	0		0	0
0	1	1		0	0
1	0	0		0	1
1	0	1		0	1
1	1	0		1	1
1	1	1		1	0

- Simplification of Q_0

a_2a_1 / a_0	00	01	11	10
0	0	0	1	0
1	0	0	1	0

$$Q_0 = a_2 a_1$$

- Simplification of Q_1

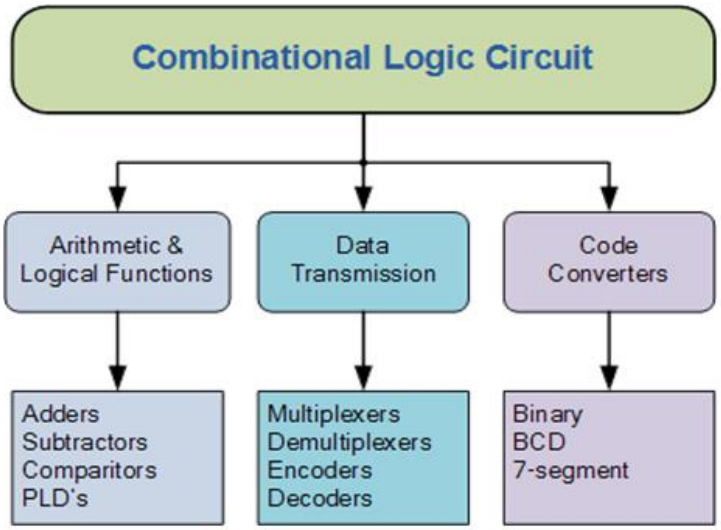
a_2a_1 / a_0	00	01	11	10
0	0	0	0	1
1	0	1	0	1

$$Q_1 = a_2\overline{a_1} + \overline{a_2}a_1 a_0$$

- Circuit diagram

As combinational logic circuits are made up from individual logic gates only, they can also be considered as "decision making circuits" and combinational logic is about combining logic gates together to process two or more signals in order to produce at least one output signal according to the logical function of each logic gate. Common combinational circuits made up from individual logic gates that carry out a desired application include *Multiplexers*, *De-multiplexers*, *Encoders*, *Decoders*, *Full* and *Half Adders* etc.

4. Classification of Combinational Logic



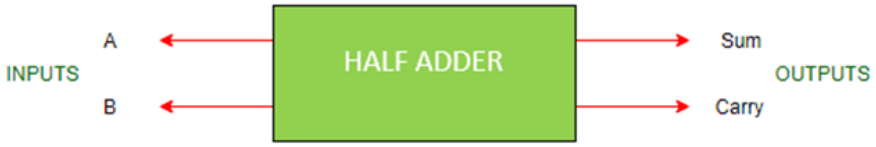
5. Adder and subtractor

We will see how to build a circuit for the addition of two numbers:

- Half Adder: a combinational logic circuit that performs the addition of two single bits
- Full Adder: a combinational logic circuit that performs the addition of three single.
- Adder of n Bits.

5.1. Half Adder

It is an arithmetic combinational logic circuit designed to perform addition of two single bits. It contains two inputs and produces two outputs. Inputs are called Augend and Added bits and Outputs are called Sum and Carry. A half adder ignores a prior carry.



Let us observe the addition of single bits : 0+0=0 , 0+1=1, 1+0=1 ,1+1=10

Sine 1+1=10, the result must be two bits output. So, Above can be rewritten as : 0+0=00 , 0+1=01 , 1+0=01 , 1+1=10

The result of 1+1 is 10, where '1' is carry-output (C_i) and '0' is Sum-output (Normal Output).

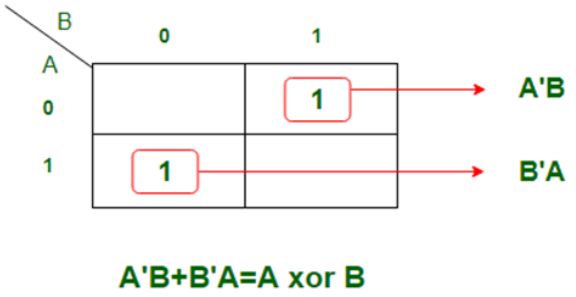
Truth Table of Half Adder:

Inputs		Output	
A_i	B_i	S_i	C_i
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

Next Step is to draw the Logic Diagram. To draw Logic Diagram, We need Boolean Expression, which can be obtained using K-map (Karnaugh map). Since there are two output variables 'S' and 'C', we need to define K-map for each output variable.

K-map for output variable Sum 'S_i':

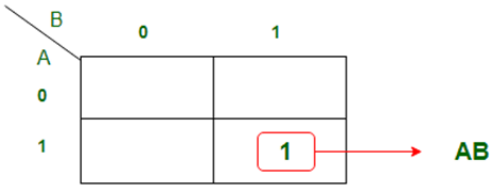
B_i \ A_i	0	1
0	0	1
1	1	0



K-map is of **Sum of products** form. The equation obtained is $S_i = \bar{A_i}B_i + A_i\bar{B_i}$, which can be logically written as: $S_i = A_i \oplus B_i$

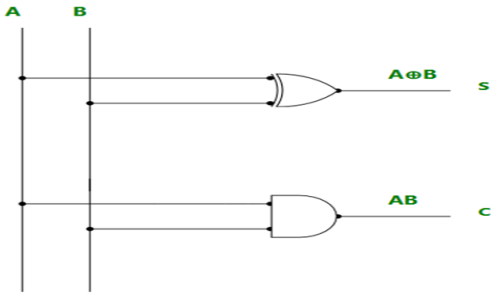
K-map for output variable Carry 'C_i':

B_i \ A_i	0	1
0	0	0
1	0	1



The equation obtained from K-map is $C_i = A_iB_i$

Using the Boolean Expression, we can draw logic diagram as follows:



Limitations: Adding of Carry is not possible in Half adder.

5.2. Full Adder

To overcome the above limitation faced with Half adders, Full Adders are implemented. It is an arithmetic combinational logic circuit that performs addition of three single bits. It contains three inputs (A, B, C_{in}) and produces two outputs (Sum and C_{out}). Where, C_{in} is Carry In and C_{out} is Carry Out.

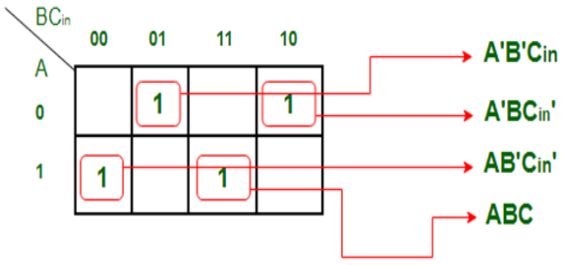
Truth table of Full Adder:

Inputs			Output	
A_i	B_i	C_{i-1}	S_i	C_i
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

Inputs			Outputs	
A	B	C_{in}	Sum	C_{out}
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

K-map Simplification for output variable Sum 'S_i':

A_i B_i / C_{i-1}	00	01	11	10
0	0	1	0	1
1	1	0	1	0



The equation obtained is : $S_i = \bar{A}_i\bar{B}_iC_{i-1} + A_i\bar{B}_i\overline{C_{i-1}} + A_iB_iC_{i-1} + \bar{A}_iB_i\overline{C_{i-1}}$, The equation can be simplified as:

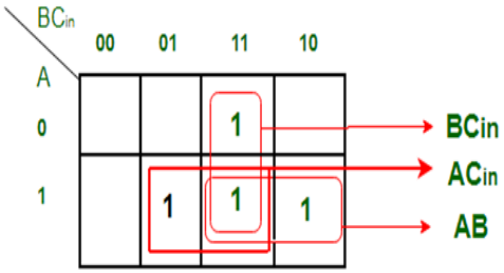
$$S_i = C_{i-1}(\bar{A}_i\bar{B}_i + A_iB_i) + \overline{C_{i-1}}(A_i\bar{B}_i + \bar{A}_iB_i)$$

$$S_i = C_{i-1}(\overline{A_i \oplus B_i}) + \overline{C_{i-1}}(A_i \oplus B_i)$$

$$S_i = (A_i \oplus B_i \oplus C_{i-1})$$

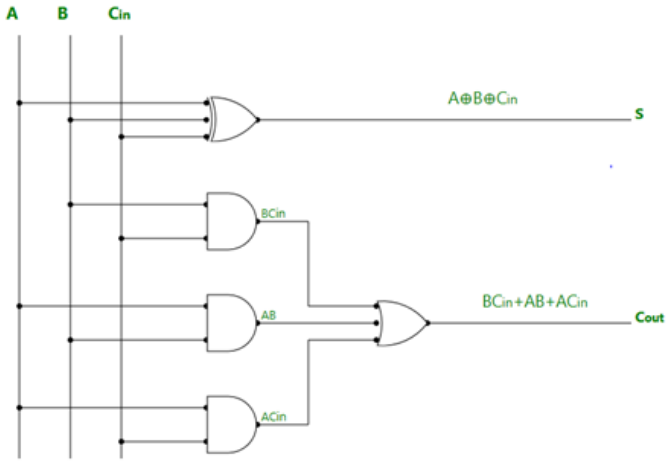
K-map Simplification for output variable 'C_i'

C_{i-1} \ $A_i B_i$	00	01	11	10
0	0	0	1	0
1	0	1	1	1



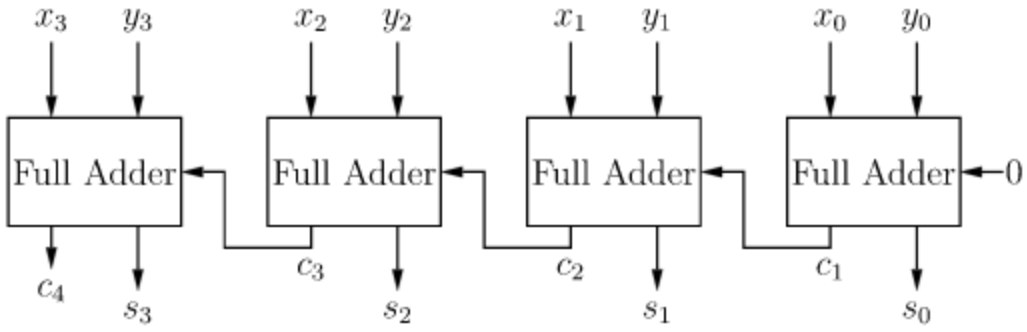
The equation obtained is: $C_i = B_iC_{i-1} + A_iB_i + A_iC_{i-1}$

Logic Diagram of Full Adder:



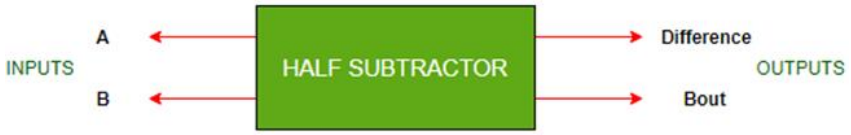
Addition of n Bits:

An n-bit adder can be implemented with full adders. The following figure shows a 4-bit adder:



5.3. Half Subtractor:

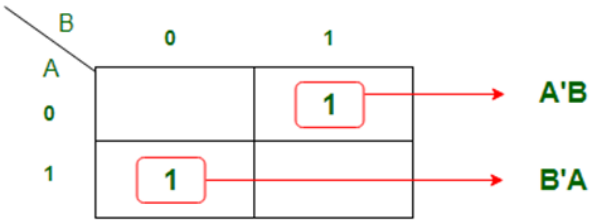
It is a combinational logic circuit designed to perform the subtraction of two single bits. It contains two inputs (A and B) and produces two outputs (Difference and Borrow-output).



Truth Table of Half Subtractor:

Inputs		Outputs	
A	B	D	B_{out}
0	0	0	0
0	1	1	1
1	0	1	0
1	1	0	0

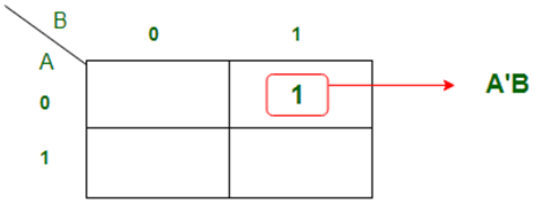
K-map Simplification for output variable 'D':



A'B+B'A=A xor B

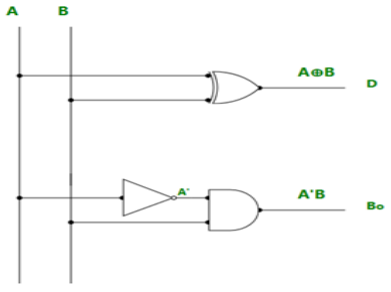
The equation obtained is: $D = A\bar{B} + \bar{A}B$, which can be logically written as: D = A xor B

K-map Simplification for output variable 'B_{out}' :



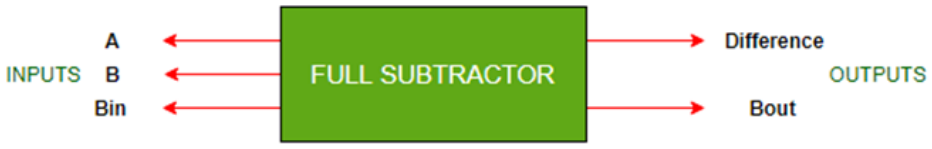
The equation obtained from above K-map is: $B_{out} = \bar{A}B$

Logic Diagram of Half Subtractor:



5.4. Full Subtractor

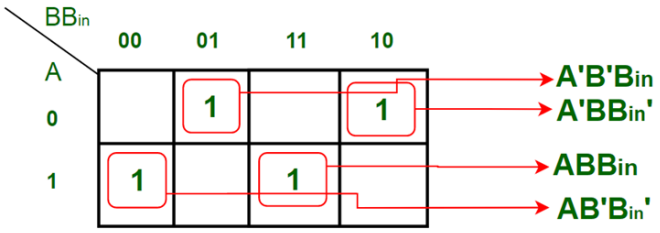
It is a Combinational logic circuit that takes into account an incoming borrow and performs the subtraction of three single bits. It contains three inputs (A, B, B_{in}) and produces two outputs (D, B_{out}). Where, A and B are called **Minuend** and **Subtrahend** bits. And, B_{in} -> Borrow-In and B_{out} -> Borrow-Out



Truth Table of Full Subtractor:

Inputs			Outputs	
A	B	B_{in}	D	B_{out}
0	0	0	0	0
0	0	1	1	1
0	1	0	1	1
0	1	1	0	1
1	0	0	1	0
1	0	1	0	0
1	1	0	0	0
1	1	1	1	1

K-map Simplification for output variable 'D' :



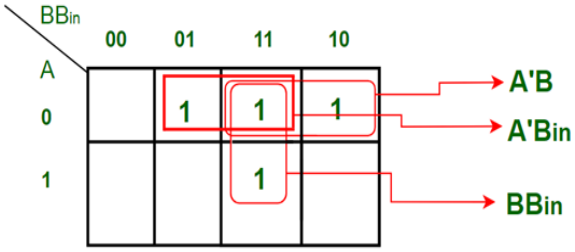
The equation obtained from above K-map is: $D = \bar{A}\bar{B}B_{in} + A\bar{B}\overline{B_{in}} + \bar{A}B\overline{B_{in}} + ABB_{in}$ which can be simplified as:

$$D = B_{in}(\bar{A}\bar{B} + AB) + \overline{B_{in}}(A\bar{B} + \bar{A}B)$$

$$D = B_{in}\overline{(A \oplus B)} + \overline{B_{in}}(A \oplus B)$$

$$D = A \oplus B \oplus B_{in}$$

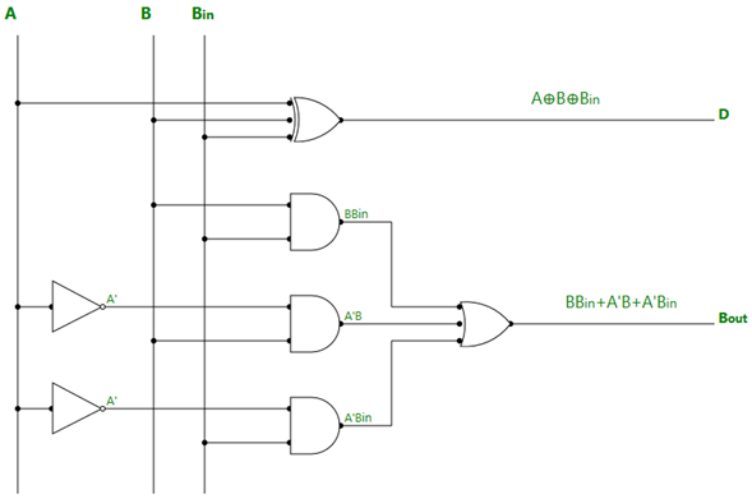
K-map Simplification for output variable 'B_{out}':



The equation obtained is: $B_{out} = BB_{in} + \bar{A}B + \bar{A}B_{in}$

$$B_{out} = \bar{A}B + (\overline{A \oplus B})B_{in}$$

Logic Diagram of Full Subtractor:



Applications:

1. For performing arithmetic calculations in electronic calculators and other digital devices.
2. In Timers and Program Counters.
3. Useful in Digital Signal Processing