Sairam INSTITUTIONS

Sri SAI RAM ENGINEERING COLLEGE
INSTITUTE OF TECHNOLOGY
West Tambaram, Chennai - 44

YEAR II | SEM III

CS 8351

DIGITAL PRINCIPLES AND SYSTEM DESIGN
(Common to CSE & IT)

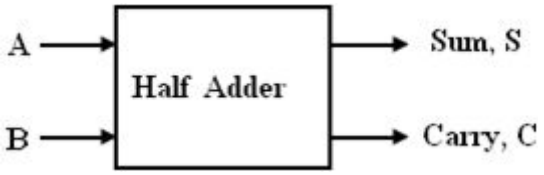
UNIT NO. 2

2.3 BINARY ADDER

Version: 1.0

Half-Adder:

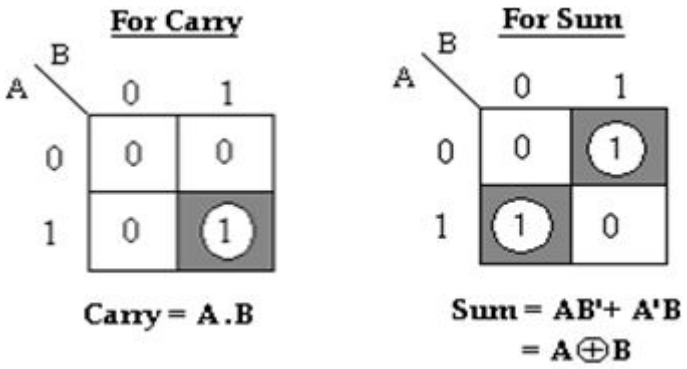
A half-adder is a combinational circuit that can be used to add two binary bits.It has two inputs that represent the two bits to be added and two outputs, with one producing the SUM output and the other producing the CARRY.



Truth table of a half-adder

INPUTS		OUTPUTS	
A	**B**	**CARRY(C)**	**SUM(S)**
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

K-map simplification for carry and sum:

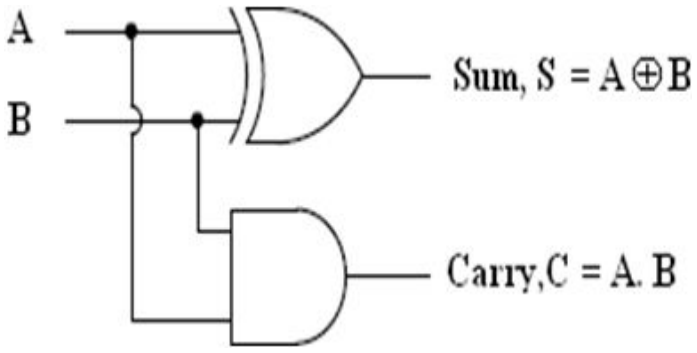


Boolean expression for SUM and CARRY

Sum, S = A'B+ AB'= AÅB

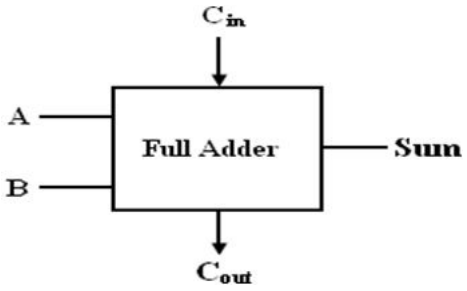
Carry, C = A . B

Logic diagram of the half adder



Full-Adder:

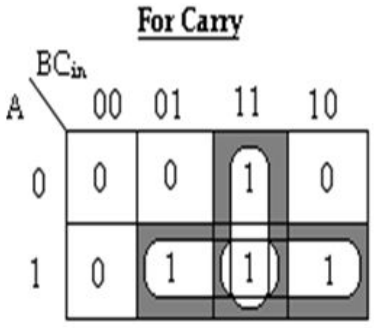
A full adder is a combinational circuit that forms the arithmetic sum of three input bits. It consists of 3 inputs and 2 outputs.Two of the in put variables, represent the significant bits to be added.The third input represents the carry from previous lower significant position. The block diagram of full adder is given by

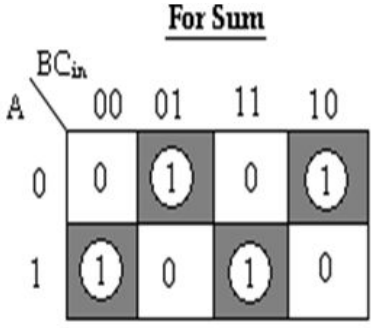


Truth Table:

INPUTS			OUTPUTS	
A	**B**	**C**	**SUM(S)**	**CARRY(Cout)**
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

To derive the simplified Boolean expression from the truth table, the Karnaugh map method is adopted as,





Carry, $C_{out} = AB + AC_{in} + BC_{in}$

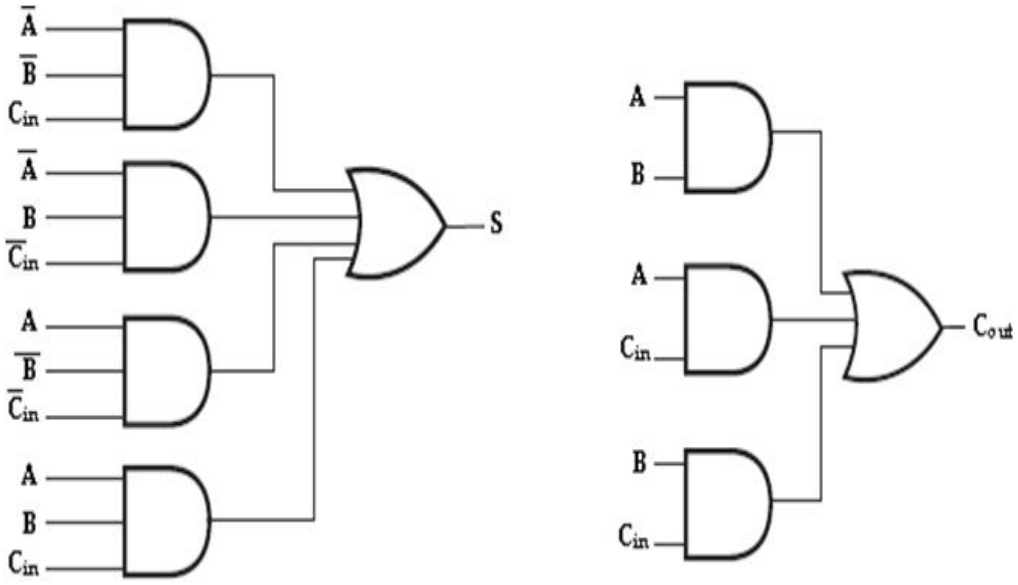
Sum, $S = A'B'C_{in} + A'BC'_{in} + AB'C'_{in} + ABC_{in}$

The Boolean expressions for the SUM and CARRY outputs are given by the equations,

Sum, S = A'B'Cin+ A'BC'in + AB'C'in + ABCin

Carry, Cout = AB+ ACin + BCin .

Logic diagram for the above functions



The logic diagram of the full adder can also be implemented with two half- adders and one OR gate.The S output from the second half adder is the exclusive-OR of Cin and the output of the first half-adder, giving

Sum = Cin Å (A Å B)

[xÅy = x‘y+ xy‘]

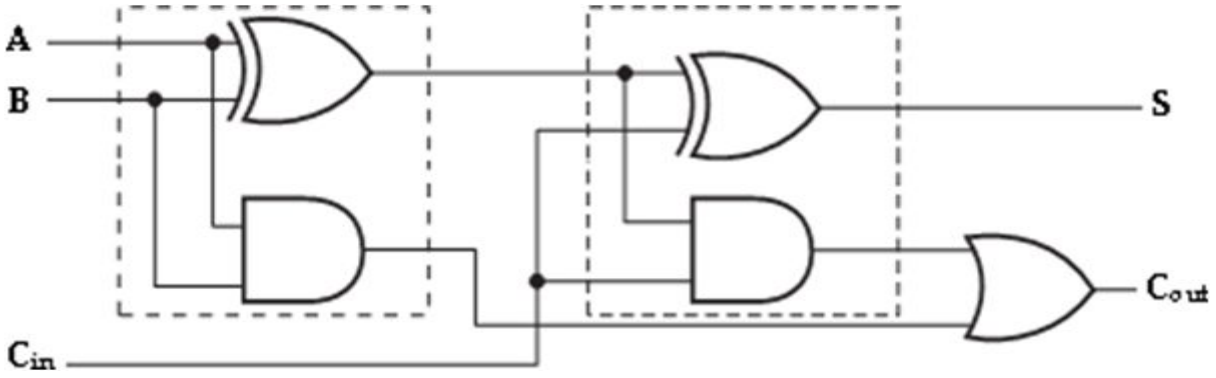
= Cin Å (A‘B+AB‘)
= C‘in (A‘B+AB‘) + Cin (A‘B+AB‘)‘ [(x‘y+xy‘)‘= (xy+x‘y‘)]
= C‘in (A‘B+AB‘) + Cin (AB+A‘B‘)

and the carry output is,

Carry, C_{out} = AB+ C_{in} (A’B+AB’)

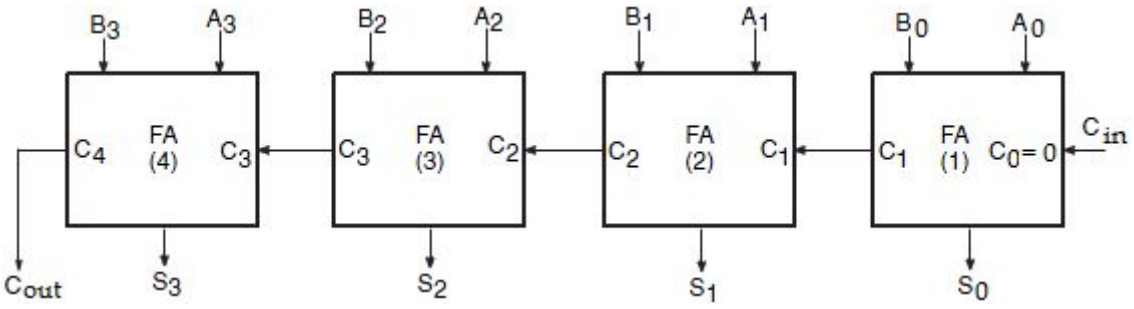
= AB+ A‘BC_{in}+ AB‘C_{in}
= AB (C_{in}+1) + A‘BC_{in}+ AB‘C_{in} [C_{in}+1= 1]
= ABC_{in}+ AB+ A‘BC_{in}+ AB‘C_{in}
= AB+ AC_{in} (B+B‘) + A‘BC_{in}
= AB+ AC_{in}+ A‘BC_{in}
= AB (C_{in}+1) + AC_{in}+ A‘BC_{in} [C_{in}+1= 1]
= ABC_{in}+ AB+ AC_{in}+ A‘BC_{in}
= AB+ AC_{in}+ BC_{in} (A +A‘)
= AB+ AC_{in}+ BC_{in}.

Implementation of full adder with two half-adders and an OR gate



Binary Adder (Parallel Adder):

The 4-bit binary adder using full adder circuits is capable of adding two 4-bit numbers resulting in a 4-bit sum and a carry output as shown in figure below.



4-bit binary parallel Adder

Since all the bits of augend and addend are fed into the adder circuits simultaneously and the additions in each position are taking place at the same time, this circuit is known as parallel adder.

Let the 4-bit words to be added be represented by, A3A2A1A0= 1111 and B3B2B1B0= 0011.

Significant place	4	3	2	1
Input carry	1	1	1	0
Augend word A :	1	1	1	1
Addend word B :	0	0	1	1
1	0	0	1	0 ← Sum

↑ Output Carry

The bits are added with full adders, starting from the least significant position, to form the sum it and carry bit. The input carry C0 in the least significant position must be 0.The carry output of the lower order stage is connected to the carry input of the next higher order stage. Hence this type of adder is called ripple-carry adder.

In the least significant stage, A_0, B_0 and C_0 (which is 0) are added resulting in sum S_0 and carry C_1. This carry C_1 becomes the carry input to the second stage. Similarly in the second stage, A_1, B_1 and C_1 are added resulting in sum S_1 and carry C_2, in the third stage, A_2, B_2 and C_2 are added resulting in sum S_2 and carry C_3, in the third stage, A_3, B_3 and C_3 are added resulting in sum S_3 and C_4, which is the output carry. Thus the circuit results in a sum ($S_3S_2S_1S_0$) and a carry output (C_{out}).

Though the parallel binary adder is said to generate its output immediately after the inputs are applied, its speed of operation is limited by the carry propagation delay through all stages. However, there are several methods to reduce this delay.

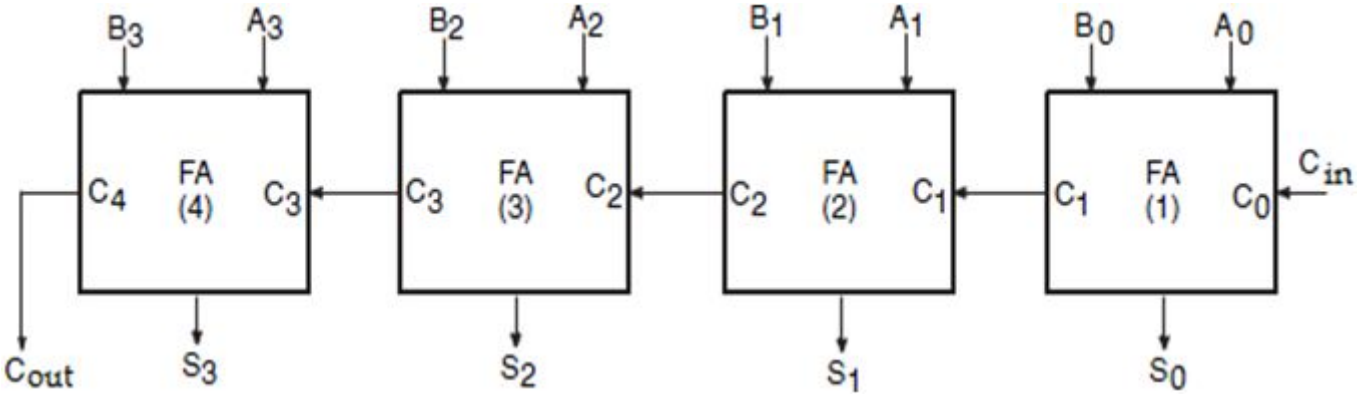
One of the methods of speeding up this process is look-ahead carry addition which eliminates the ripple-carry delay.

Carry Propagation–Look-Ahead Carry Generator:

In Parallel adder, all the bits of the augend and the addend are available for computation at the same time. The carry output of each full-adder stage is connected to the carry input of the next high-order stage. Since each bit of the sum output depends on the value of the input carry, time delay occurs in the addition process. This time delay is called as **carry propagation delay.**

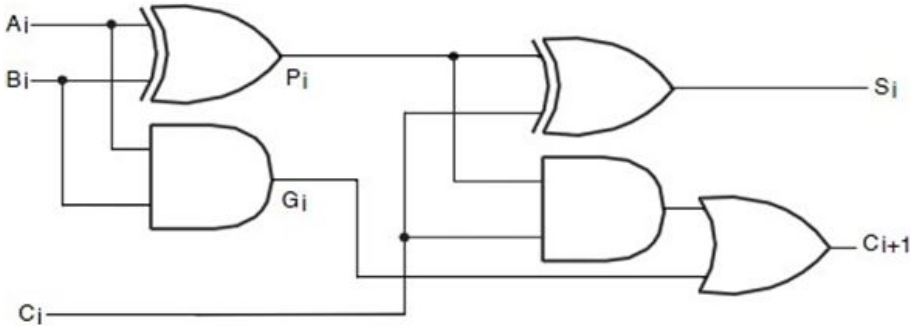
For example, addition of two numbers (0011+ 0101) gives the result as 1000. Addition of the LSB position produces a carry into the second position. This carry when added to the bits of the second position, produces a carry into the third position. This carry when added to bits of the third position, produces a carry into the last position. The sum bit generated in the last position (MSB) depends on the carry that was generated by the addition in the previous position. i.e., the adder will not produce correct result until LSB carry has propagated through the intermediate full-adders. This represents a time delay that depends on the

propagation delay produced in an each full-adder. For example, if each full adder is considered to have a propagation delay of 30nsec, then S_3 will not react its correct value until 90 nsec after LSB is generated. Therefore total time required to perform addition is 90+ 30 = 120nsec.



4-bit Parallel Adder

The method of speeding up this process by eliminating inter stage carry delay is called look ahead-carry addition. This method utilizes logic gates to look at the lower order bits of the augend and addend to see if a higher-order carry is to be generated. It uses two functions: carry generate and carry propagate.



Full-Adder circuit

Consider the circuit of the full-adder shown above. Here we define two functions: carry generate (G_i) and carry propagate (P_i) as,

Carry generate, **Gi = Ai □Bi**

Carry propagate, **Pi = Ai □ Bi**

the output sum and carry can be expressed as,

Si = Pi □

Ci Ci+1 = Gi

□PiCi

G_i (carry generate), it produces a carry 1 when both Ai and Bi are 1, regardless of the input carry C_i.Pi (carry propagate) because it is the term associated with the propagation of the carry from C_i to C_{i+1}.

The Boolean functions for the carry outputs of each stage and substitute for each C_i its value from the previous equation:

C_0= input carry

$$C1= G0 + P0C0$$

$$C2= G1 + P1C1 = G1 + P1 (G0 + P0C0)$$

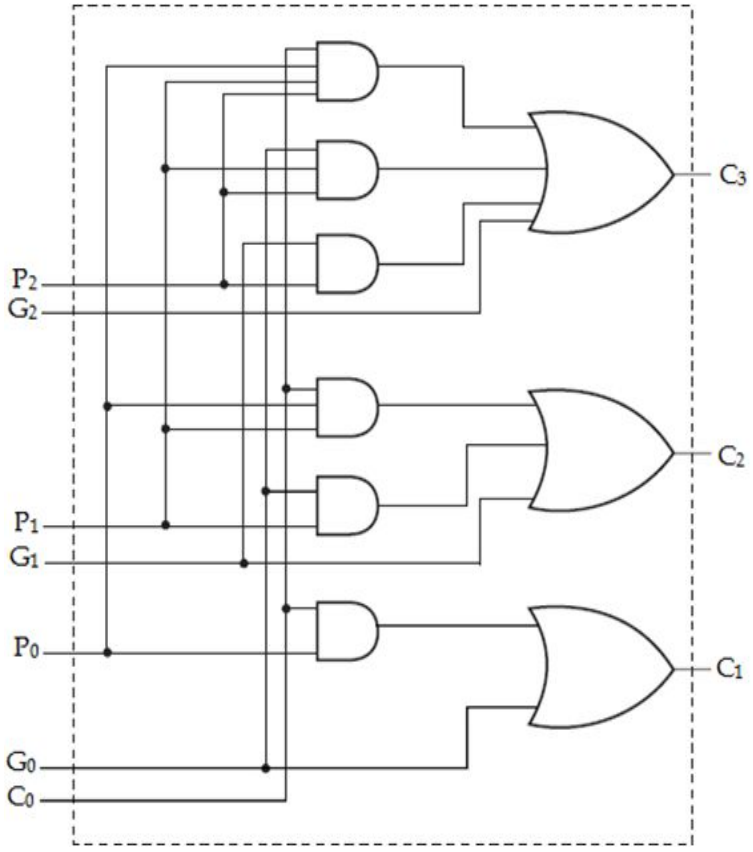
$$= G1 + P1G0 + P1P0C0$$

$$C3= G2 + P2C2 = G2 + P2 (G1 + P1G0 + P1P0C0)$$

$$= G2 + P2G1 + P2P1G0 + P2P1P0C0$$

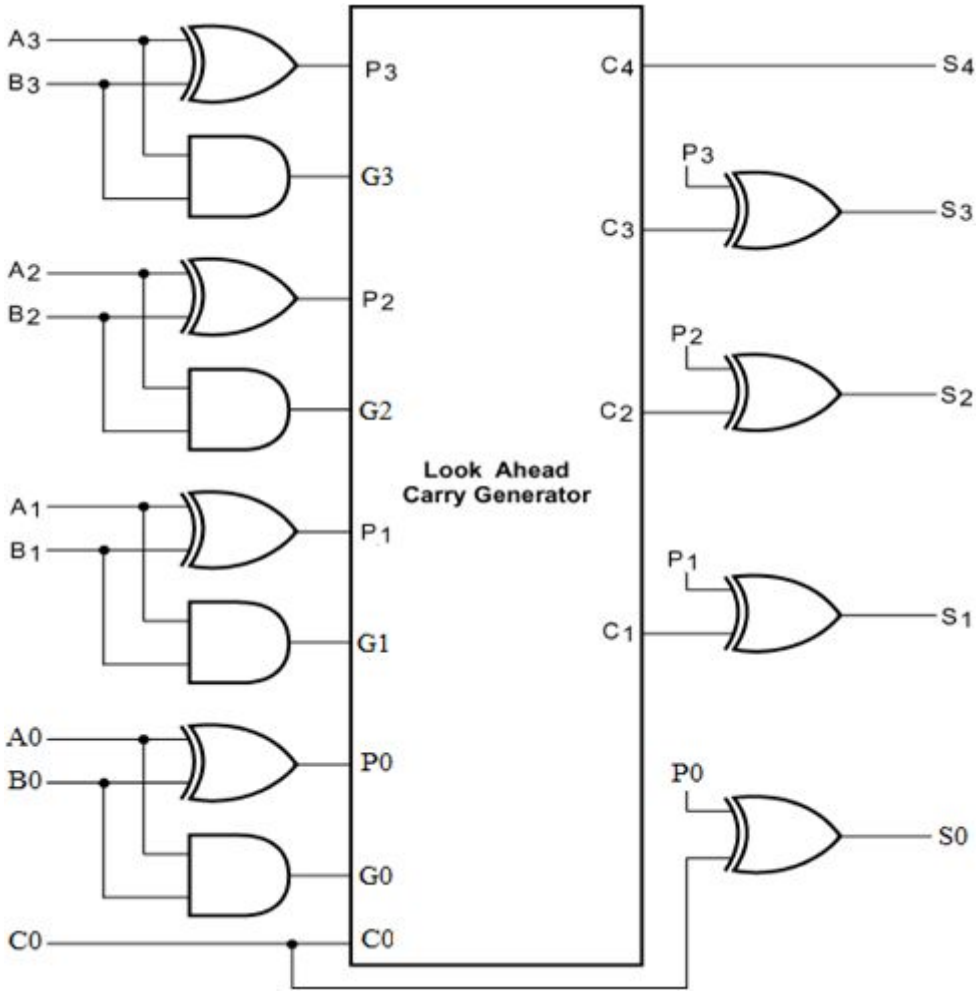
Since the Boolean function for each output carry is expressed in sum of products, each function can be implemented with one level of AND gates followed by an OR gate. The three Boolean functions for C1, C2 and C3 are implemented in the carry look-ahead generator as shown below. Note that C3 does not have to wait for C2 and C1 to propagate; in fact C3 is propagated at the same time as C1 and C2.

Logic diagram of Carry Look-ahead Generator



Using a Look-ahead Generator we can easily construct a 4-bit parallel adder with a Look-ahead carry scheme. Each sum output requires two exclusive-OR gates. The output of the first exclusive-OR gate generates the P_i variable, and the AND gate generates the G_i variable. The carries are propagated through the carry look-ahead generator and applied as inputs to the second exclusive-OR gate. All output carries are generated after a delay through two levels of gates. Thus, outputs S_1 through S_3 have equal propagation delay times.

4-Bit Adder with Carry Look-ahead



SAIRAM
DIGITAL RESOURCES

INFORMATION TECHNOLOGY

CS8351

DIGITAL PRINCIPLES AND SYSTEM DESIGN
(Common to CSE & IT)

SAIRAM
DIGITAL RESOURCES

INFORMATION TECHNOLOGY

CS8351

DIGITAL PRINCIPLES AND SYSTEM DESIGN
(Common to CSE & IT)

SAIRAM
DIGITAL RESOURCES

INFORMATION TECHNOLOGY

CS8351

DIGITAL PRINCIPLES AND SYSTEM DESIGN
(Common to CSE & IT)

SAIRAM
DIGITAL RESOURCES

INFORMATION TECHNOLOGY

CS8351

DIGITAL PRINCIPLES AND SYSTEM DESIGN
(Common to CSE & IT)