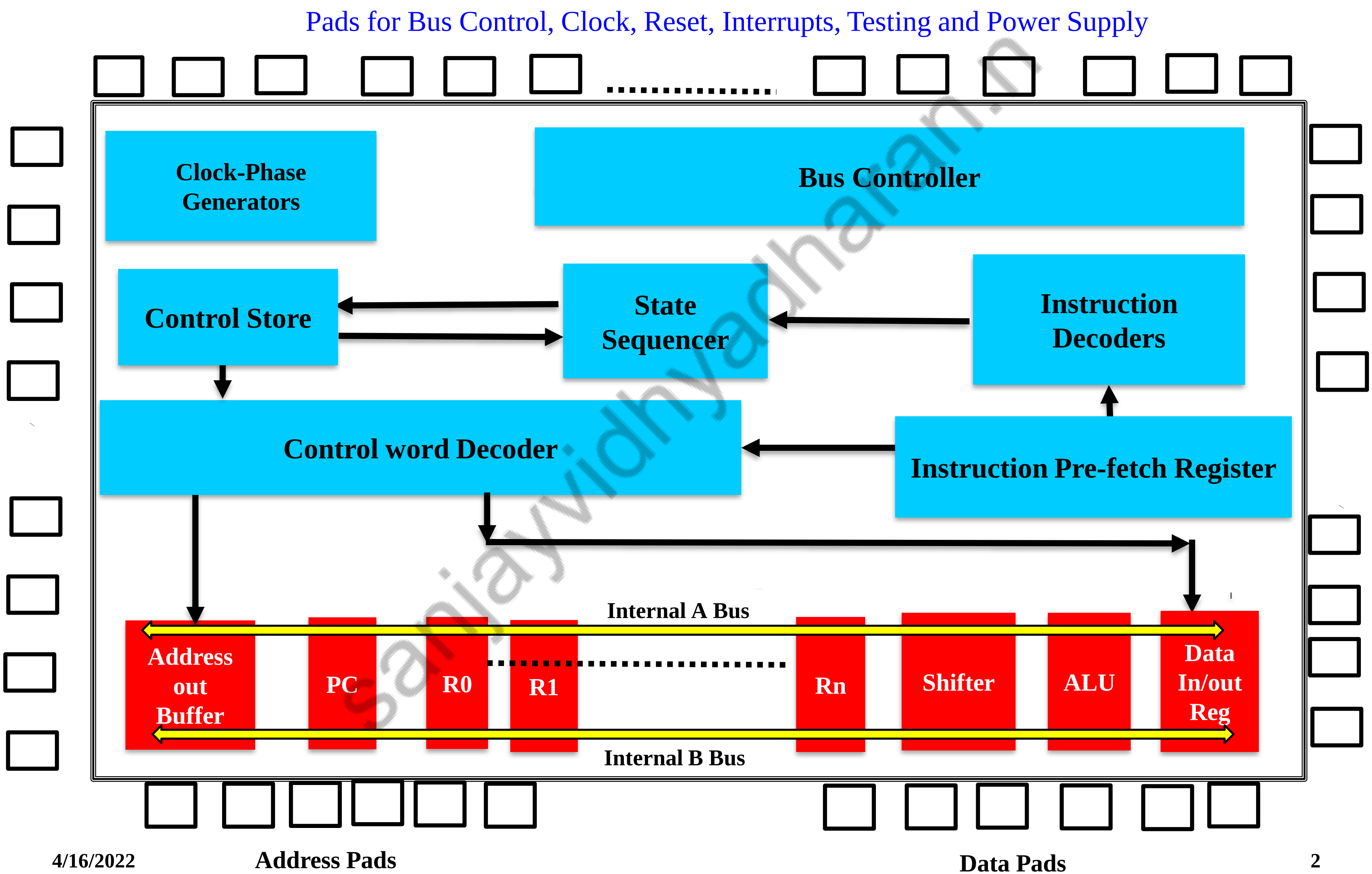
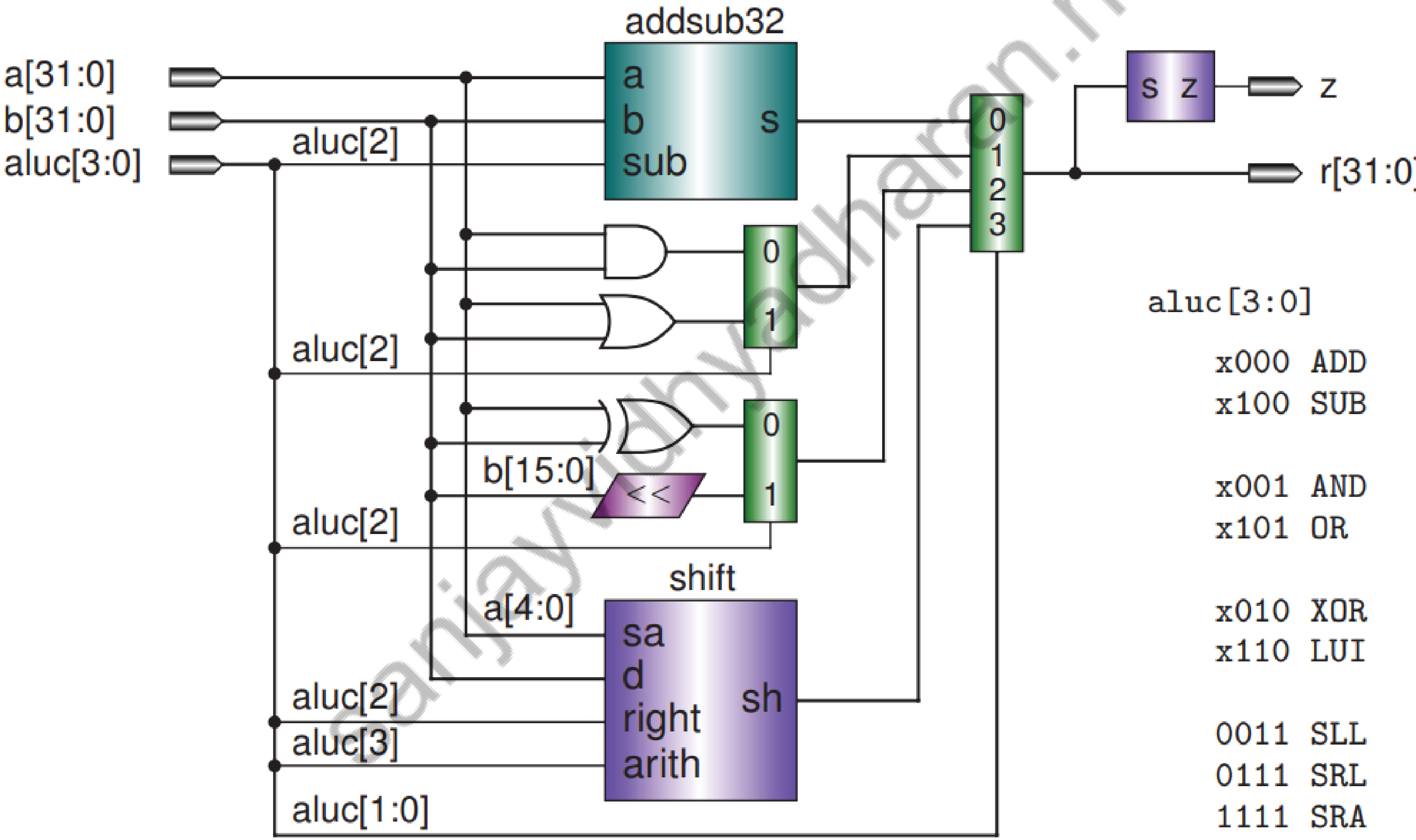
VLSI Design: 2021-22
Lecture 16
Arithmetic Circuits: Part-1

By Dr. Sanjay Vidhyadharan

Microprocessor Design



ALU Design

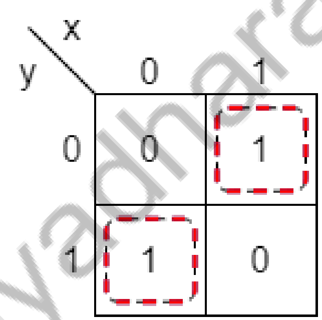


aluc[3:0]	
x000	ADD
x100	SUB
x001	AND
x101	OR
x010	XOR
x110	LUI
0011	SLL
0111	SRL
1111	SRA

Half Adder

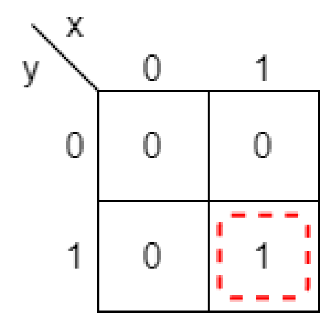
Inputs		Output	
A	B	Carry	Sum
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

K-map for Sum

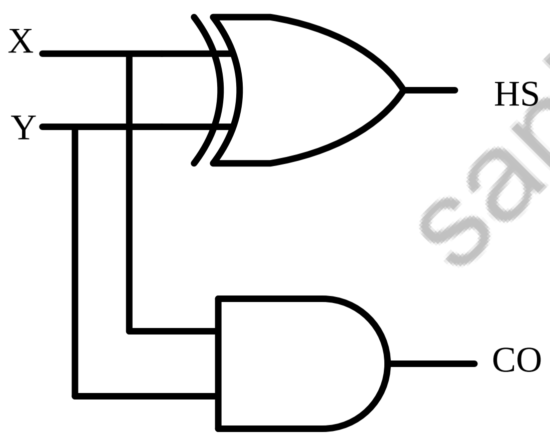


S = xy' + x'y

K-map for Carry



C = xy



Sum = X•Y' + X'•Y = X $\oplus$ Y
Carry = X•Y

Time Delay for the Half Adder?

1 gate delay

A gate delay of an xor for the half sum

A gate delay of an AND gate for the carry out

Full Adder

Full-adder can also implemented with two half adders and one OR gate.

$S = A \oplus B \oplus C_{in}$

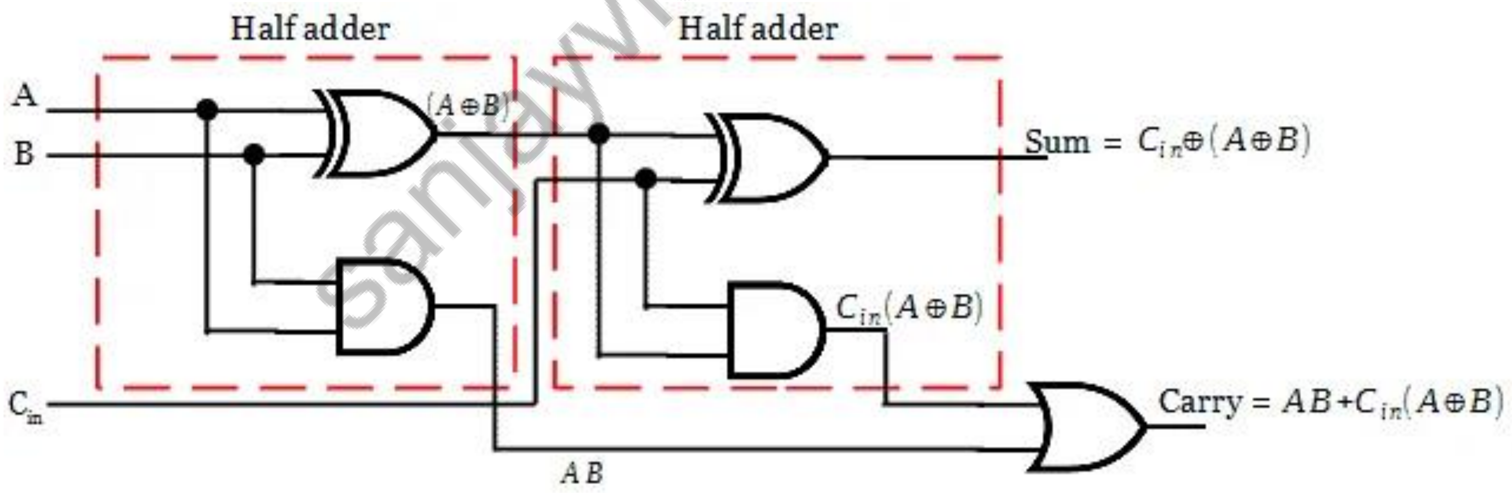
$C_{out} = AB + BC_{in} + AC_{in}$

$= AB + C_{in}(A+B)$

$= AB + C_{in}(A \oplus B + AB)$

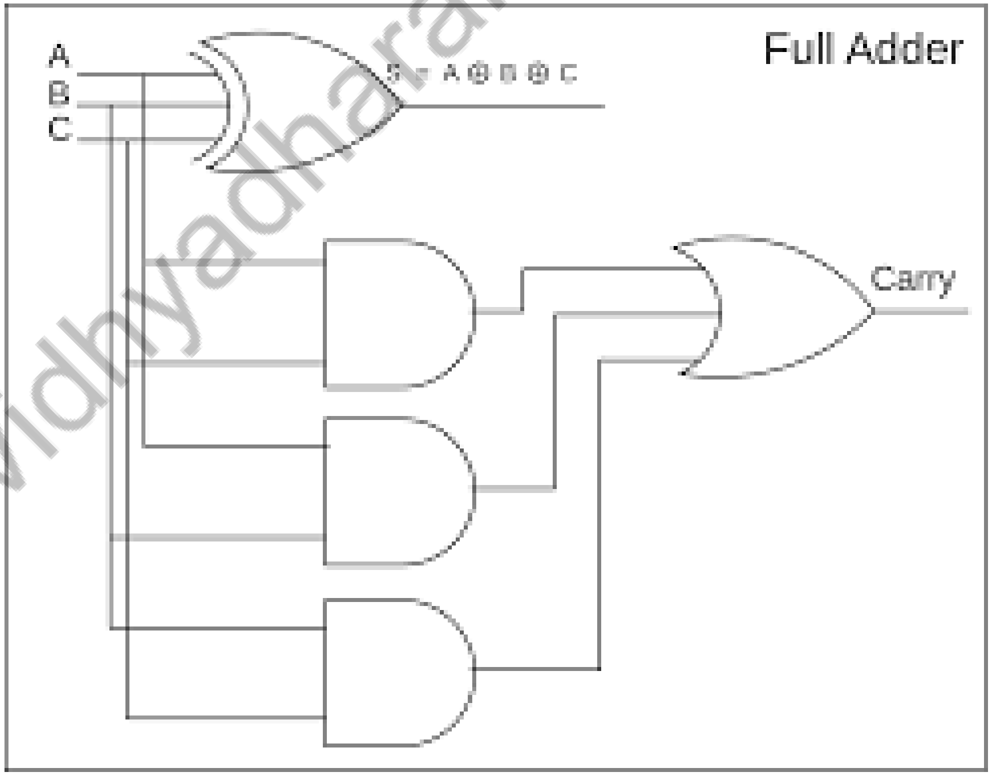
$= AB + C_{in}(A \oplus B) + C_{in}AB = AB(1 + C_{in}) + C_{in}(A \oplus B)$

$= AB + C_{in}(A \oplus B)$



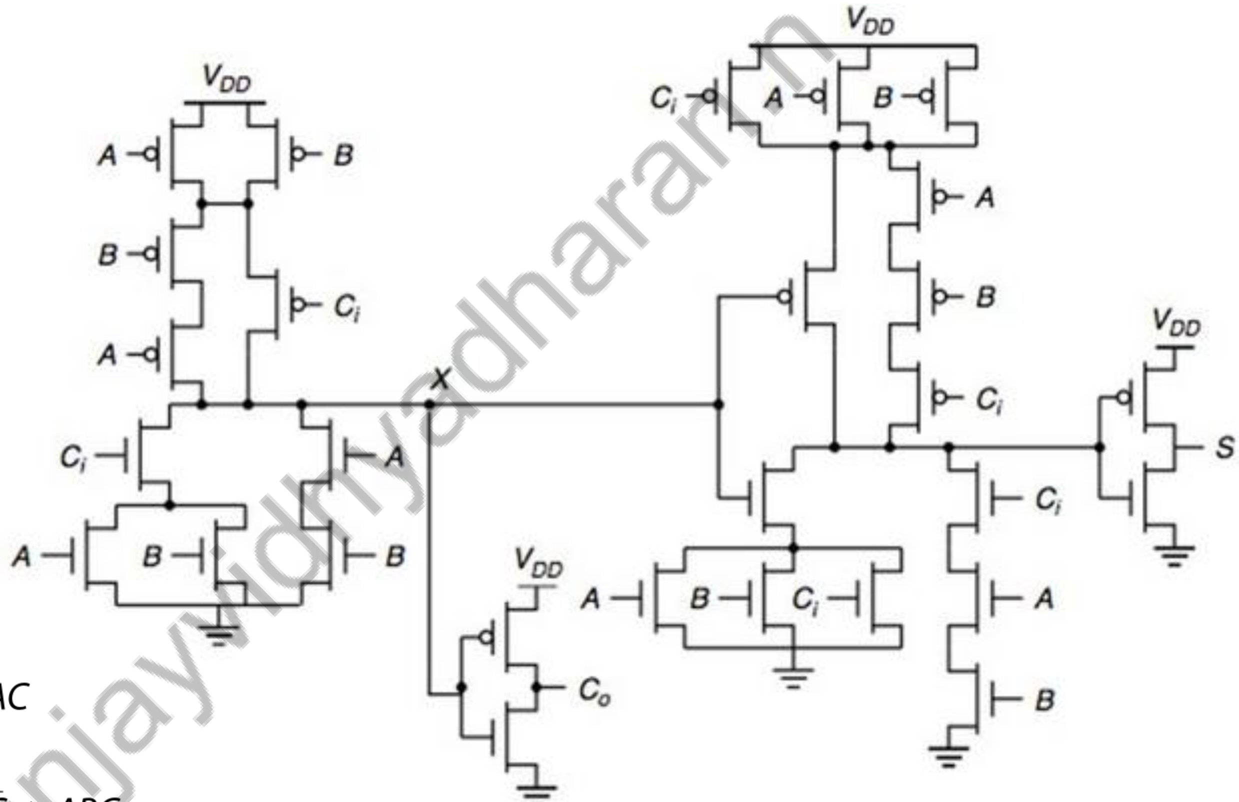
Full Adder

A	B	Carry In	Sum	Carry 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



CMOS 28T Adder

A	B	Carry in	Sum	Carry 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



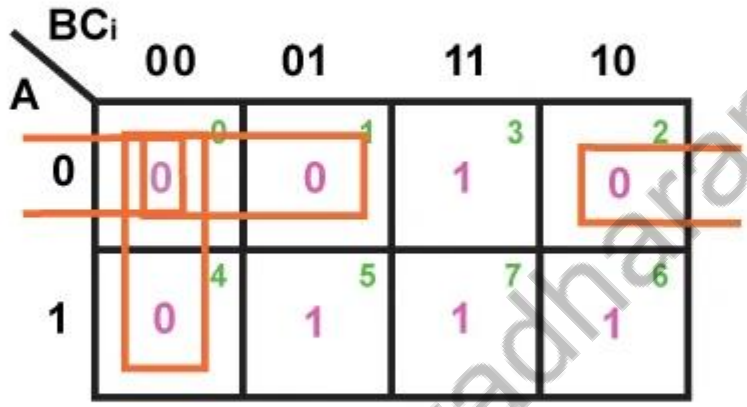
$$Carry(CY) = AB + BC + AC$$

$$Sum(S) = \bar{A}\bar{B}C_i + A\bar{B}\bar{C} + \bar{A}B\bar{C} + ABC$$

$$CY = AB + C(A + B)$$
$$S = \overline{CY}(A + B + C) + ABC$$

Simplified Expressions

CMOS 28T Mirror Adder

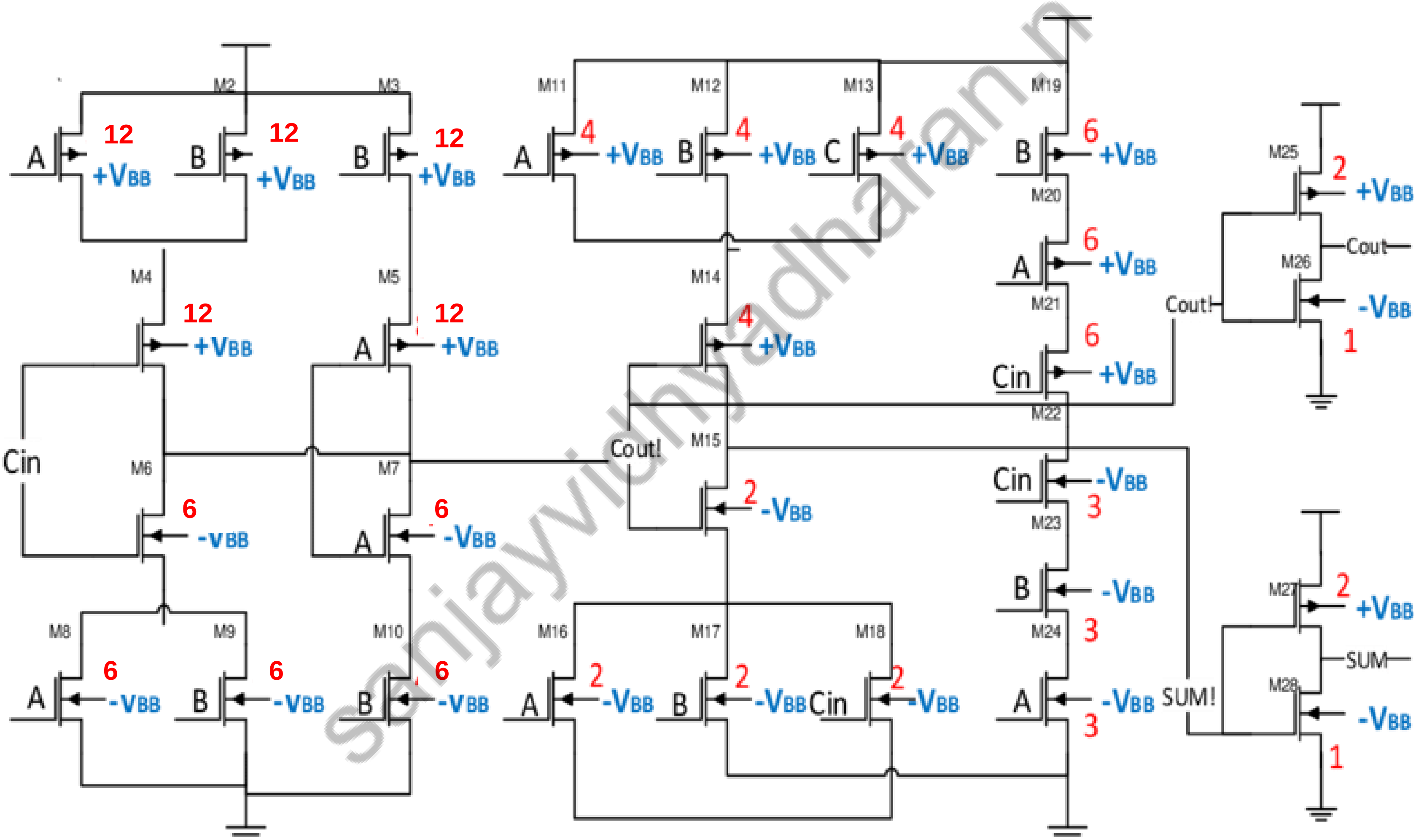


$$Carry = AB + BC + CA = AB + C(A + B)$$

$$Pull - Down\ Network\ for\ Carry\ Bar = AB + C(A + B)$$

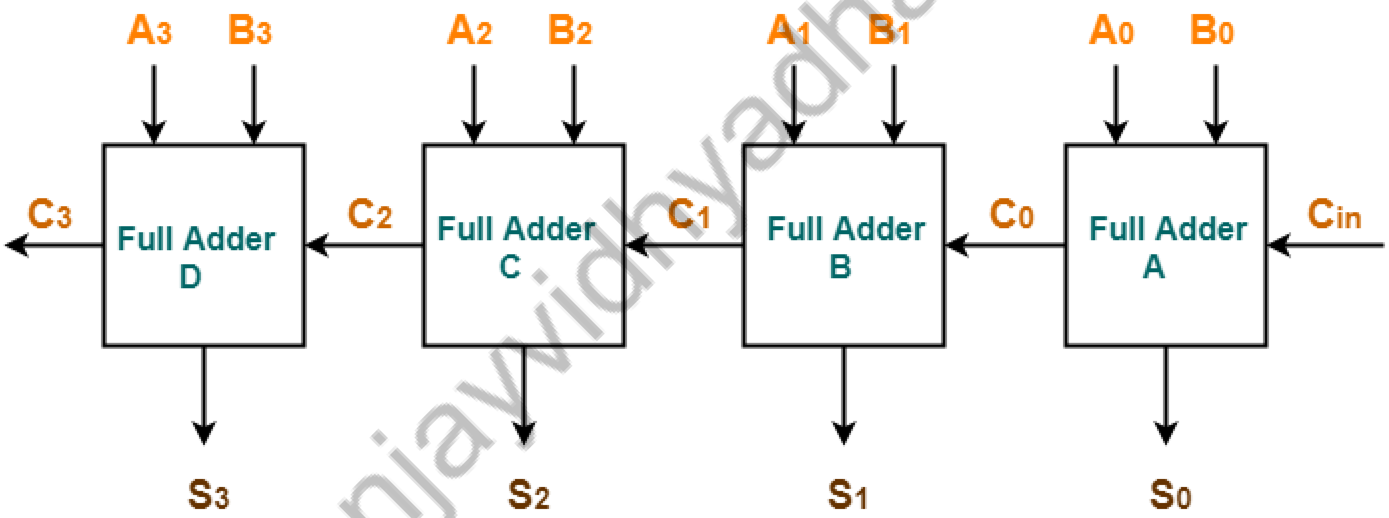
$$Pull - UP\ Network\ for\ Carry\ Bar = A'B' + B'C' + C'A'$$
$$= A'B' + C'(A' + B')$$

CMOS 28T Mirror Adder



Ripple Carry Adder

This is called Ripple Carry Adder, because of the construction with full adders are connected in cascade.



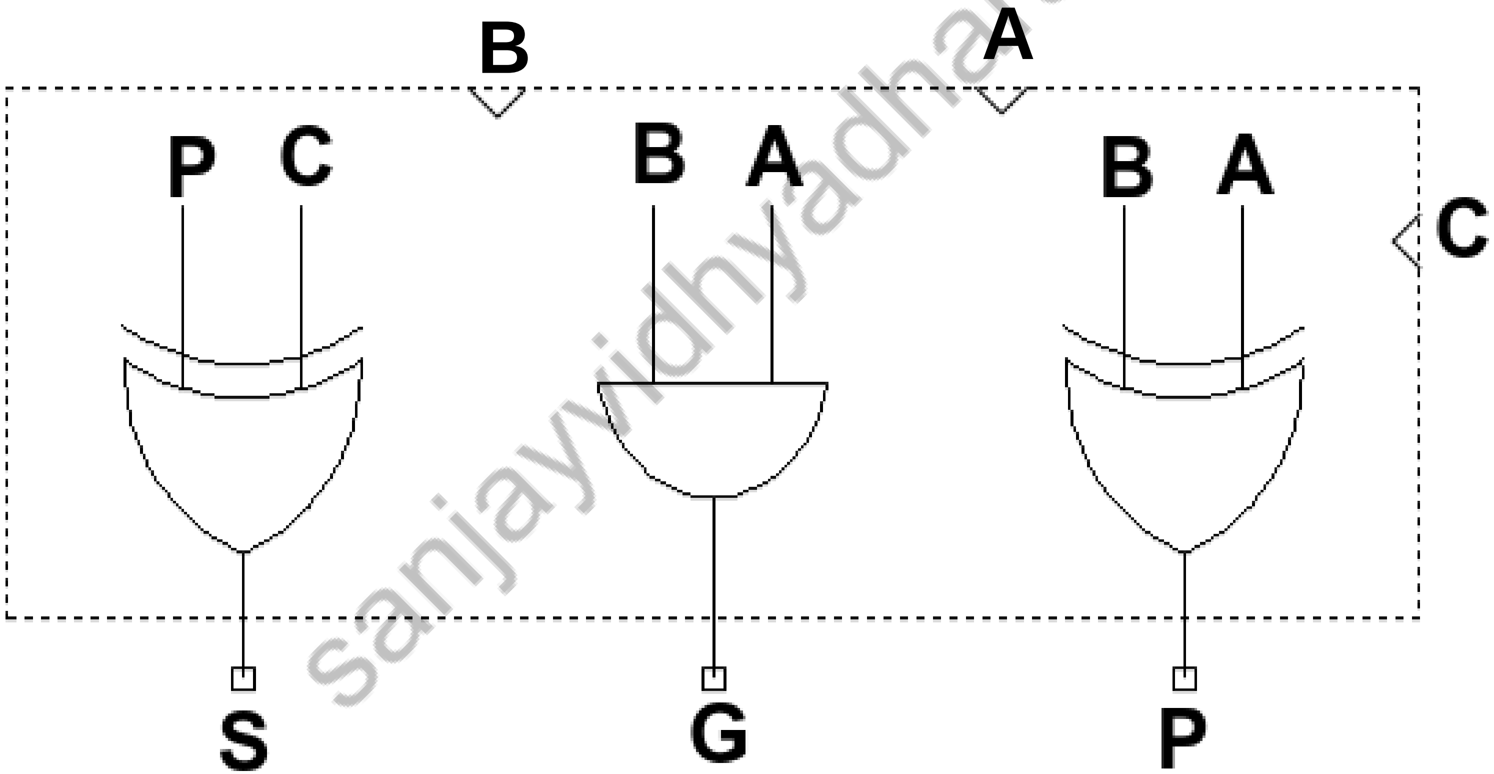
4-bit Ripple Carry Adder

Delay= 4 X Full Adder Delay = 8 Gate Delays

Delay= (N-1) t_{carry} + t_{sum}

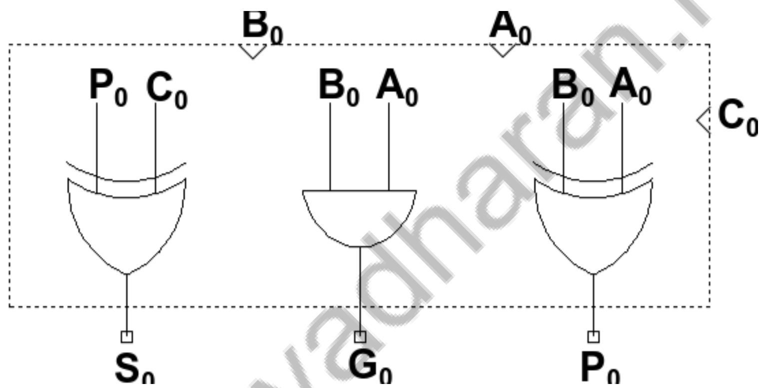
Carry Look-Ahead Adder

1-bit CLA

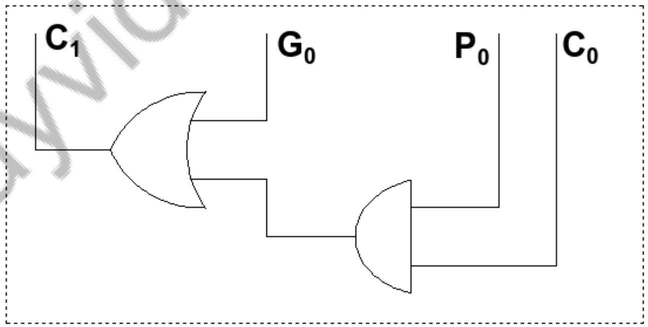


Carry Look-Ahead Adder

CLA

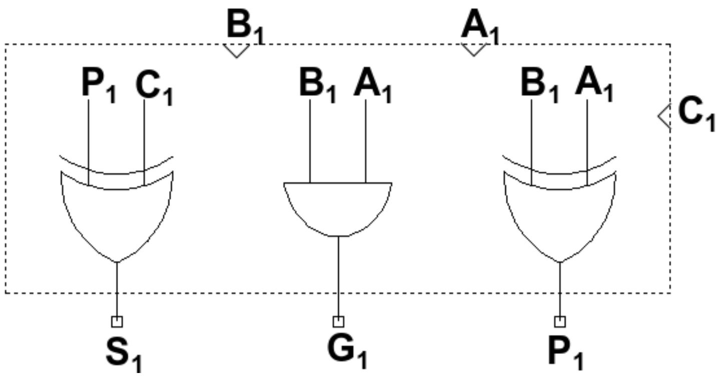


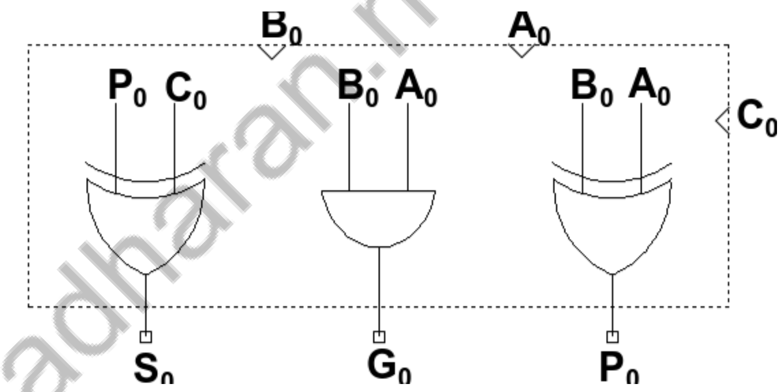
CLLB



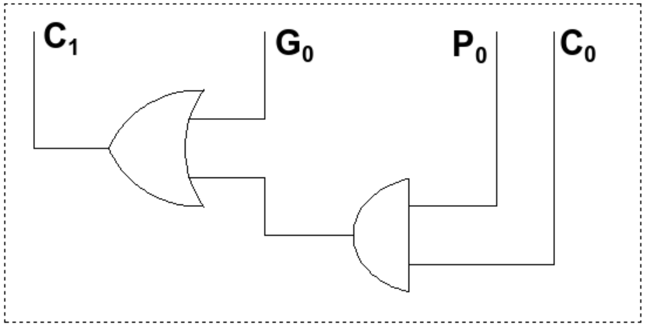
$C_1 = G_0 + P_0 C_0$

Carry Look-Ahead Adder



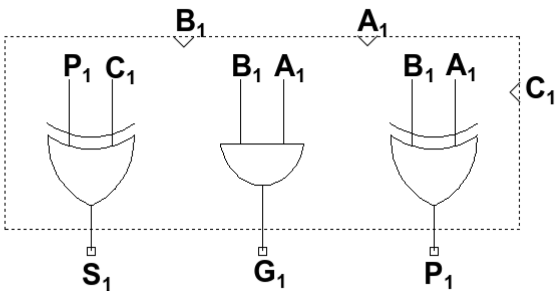


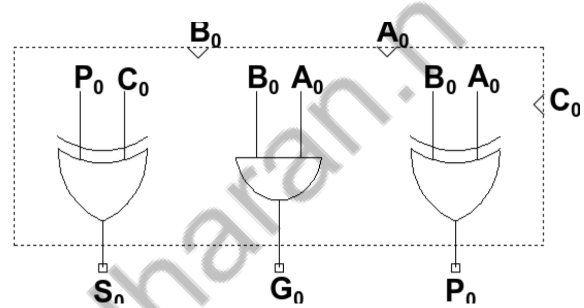
$$C_2=G_1+P_1C_1$$

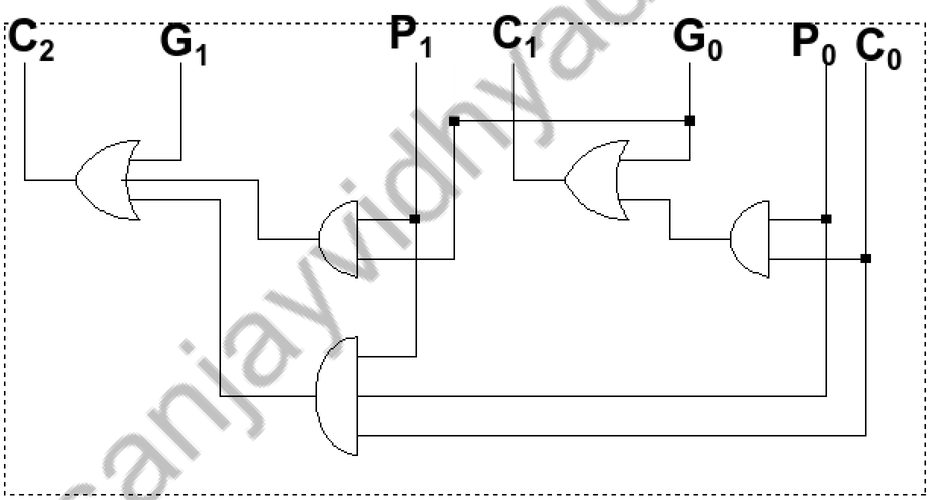


$$C_2 = G_1+P_1(G_0+P_0C_0)$$
$$= G_1+P_1G_0+P_1P_0C_0$$

Carry Look-Ahead Adder







$$C_2 = G_1 + P_1 G_0 + P_1 P_0 C_0$$

$$C_1 = G_0 + P_0 C_0$$

Carry Look-Ahead Adder

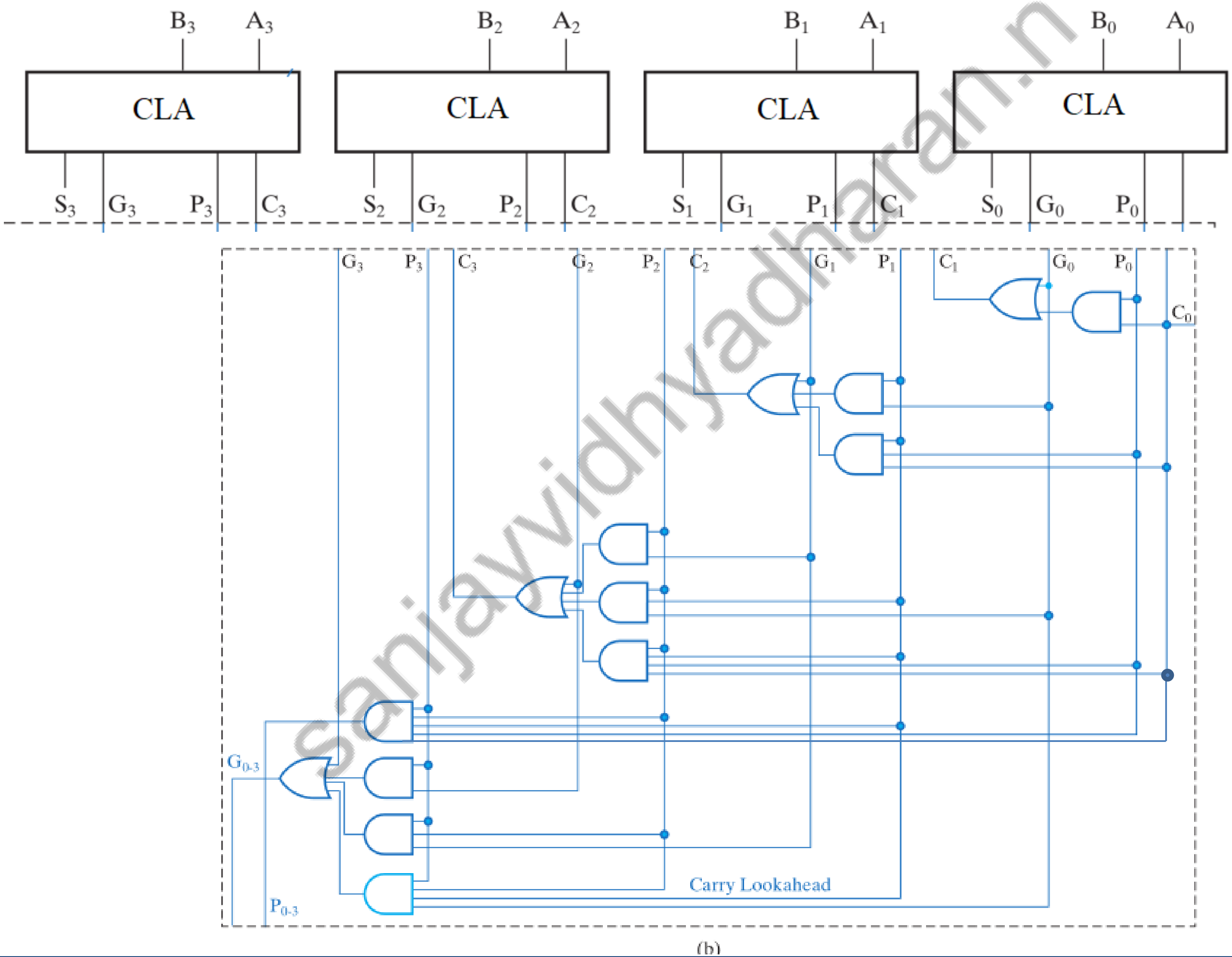
$C_1=G_0+P_0C_0$

$C_2= G_1+P_1G_0+P_1P_0C_0$

$C_3=G_2+P_2G_1+ P_2P_1G_0 +P_2P_1P_0C_0$

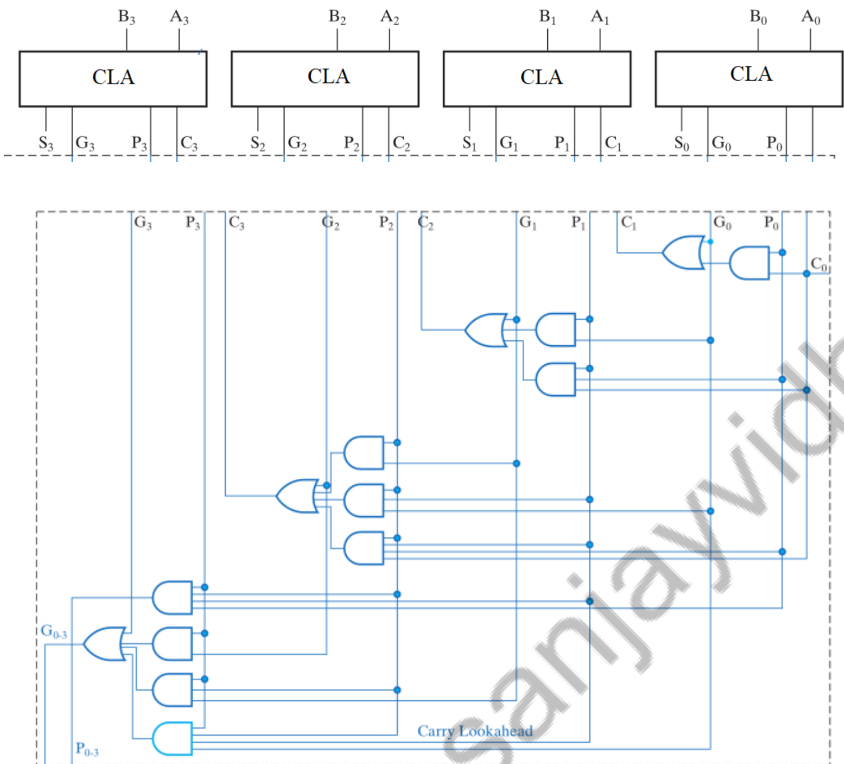
$C_4=G_3+P_3G_2+ P_3P_2G_1 +P_3P_2P_1G_0 +P_3P_2P_1P_0C_0$

Carry Look-Ahead Adder

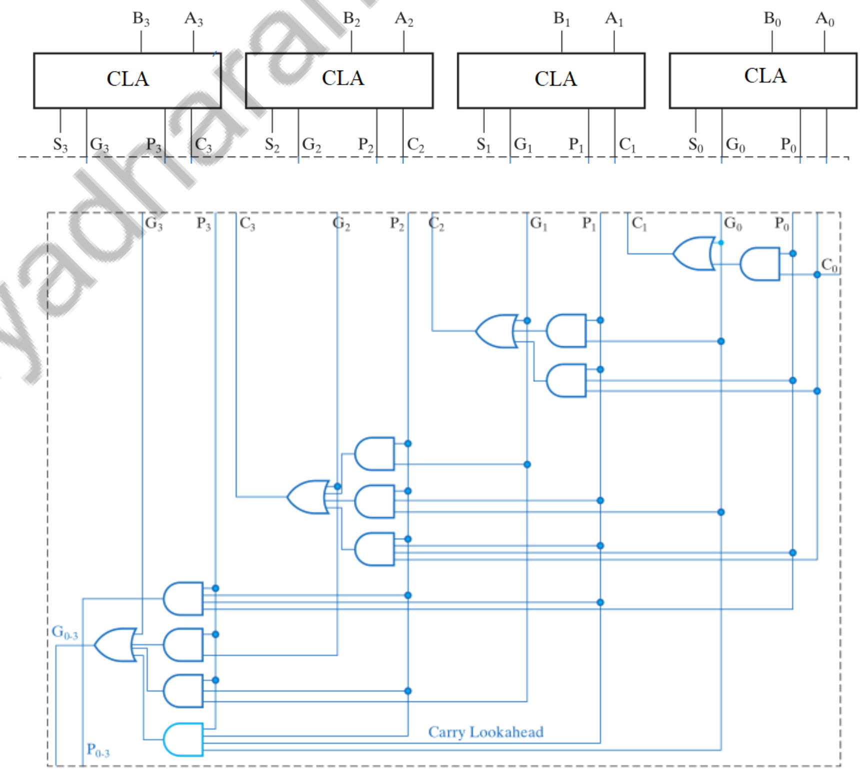


Carry Look-Ahead Adder

8 Bit Full Adder



(b)

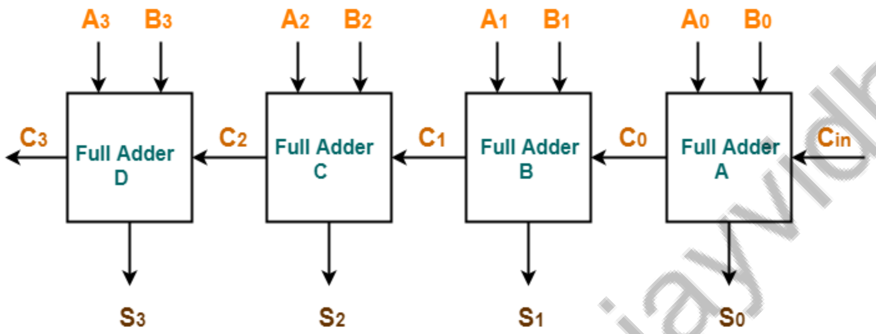


(b)

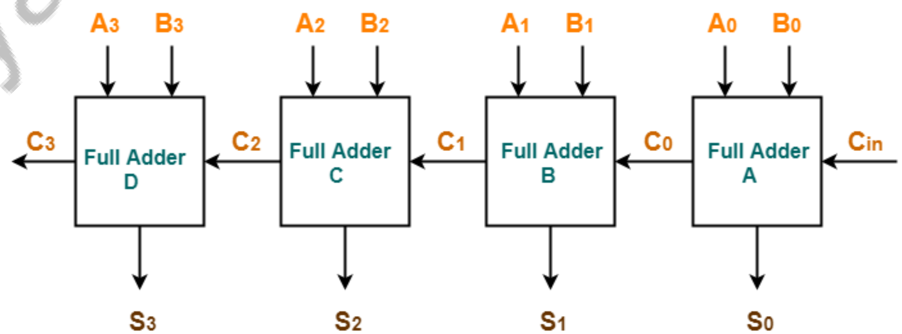
Delay = 2 X 3 = 6 Gate Delay

Ripple Carry Adder

8 Bit Full Adder



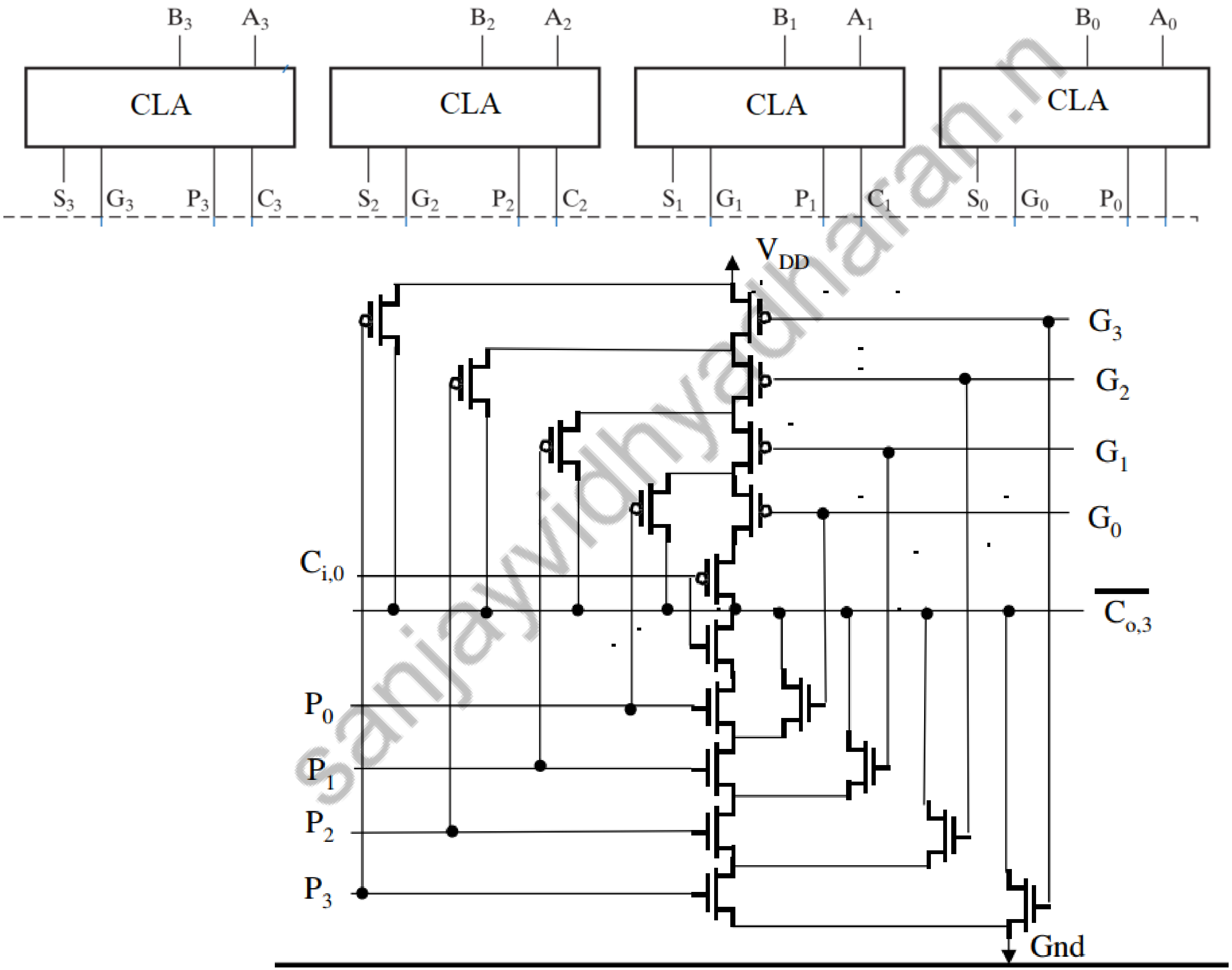
4-bit Ripple Carry Adder



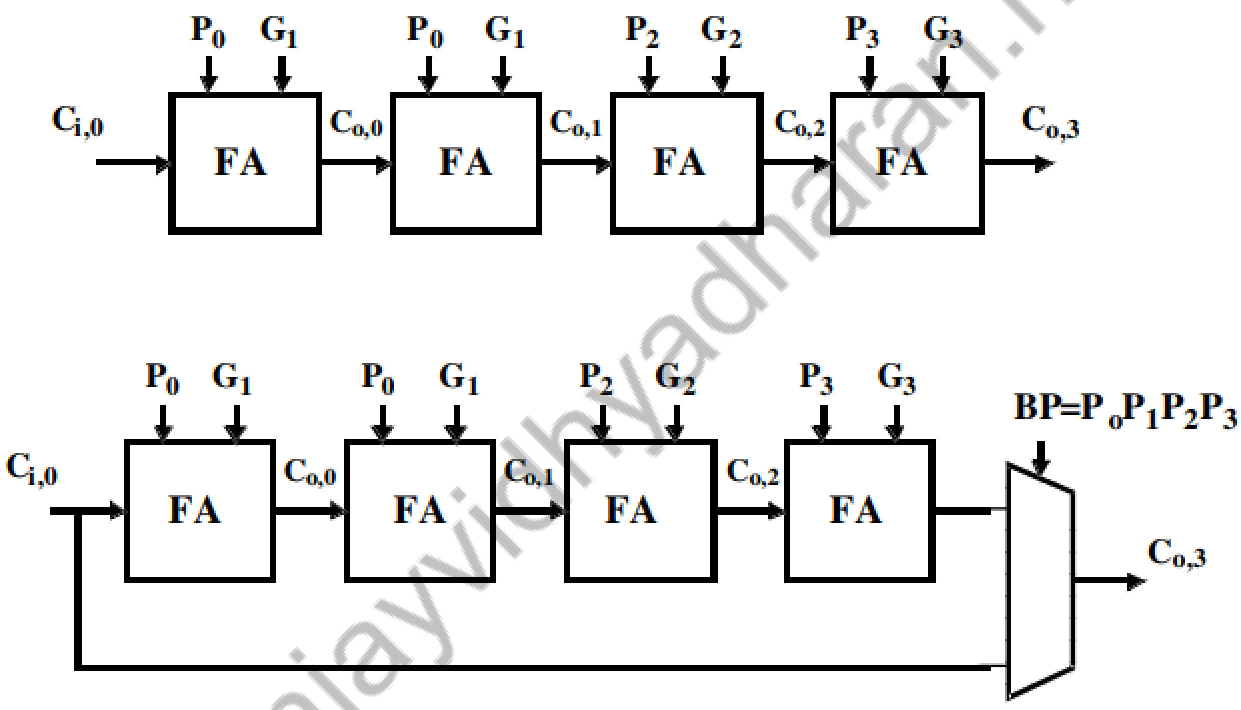
4-bit Ripple Carry Adder

Delay = 8 X 2 Gate Delay

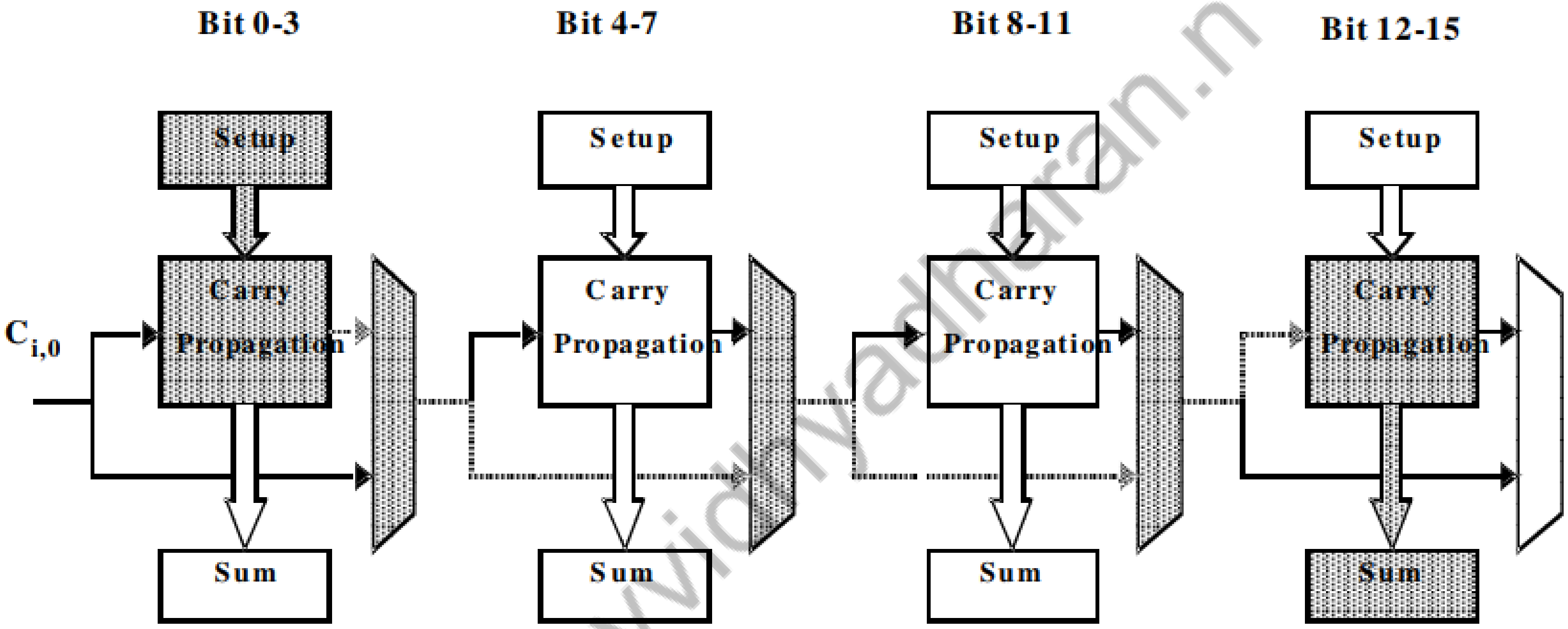
Carry Look-Ahead Adder



Carry Bypass or Carry Skip Adder



Carry Bypass or Carry Skip Adder



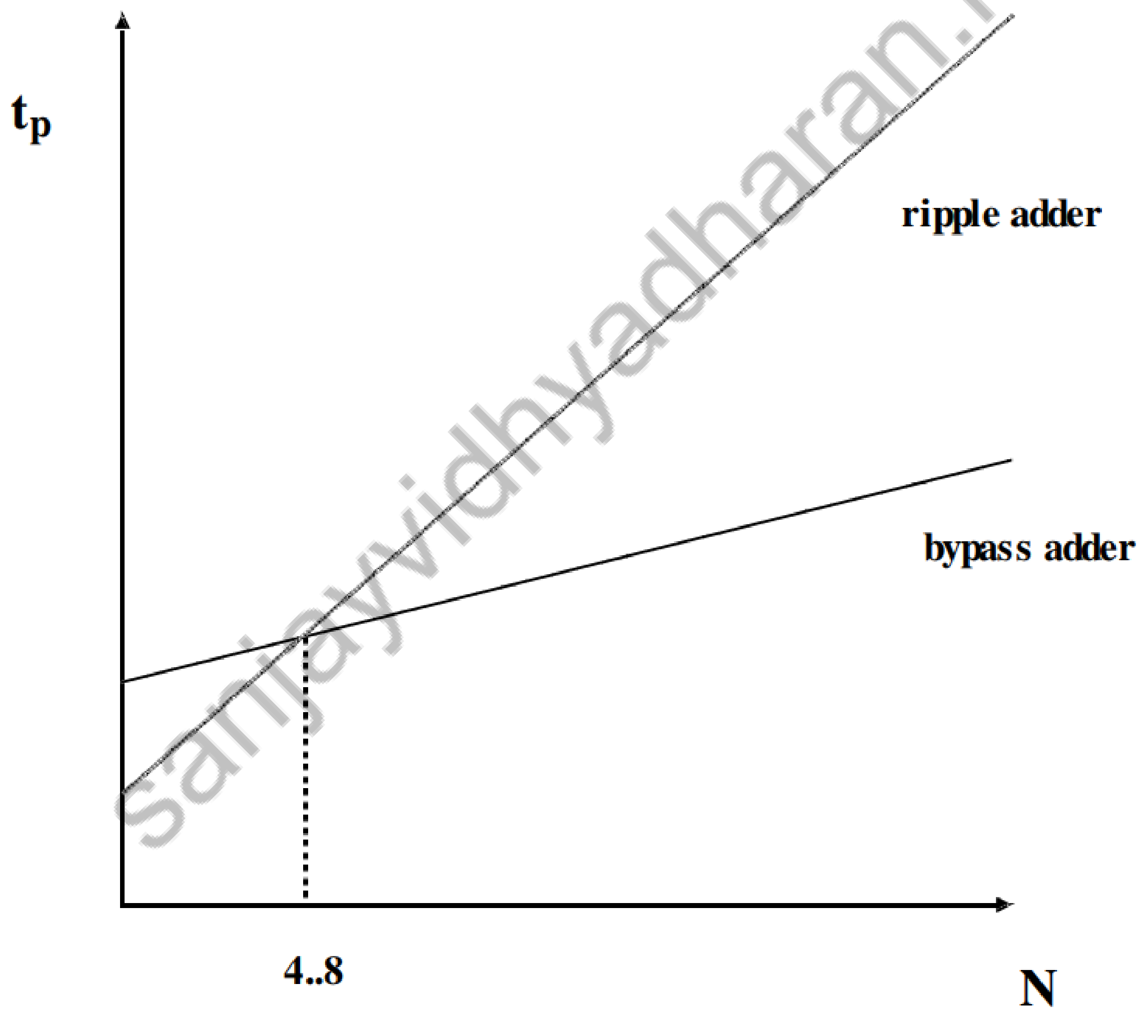
Design N-bit adder using N/M equal length stages

e.g. N = 16, M = 4

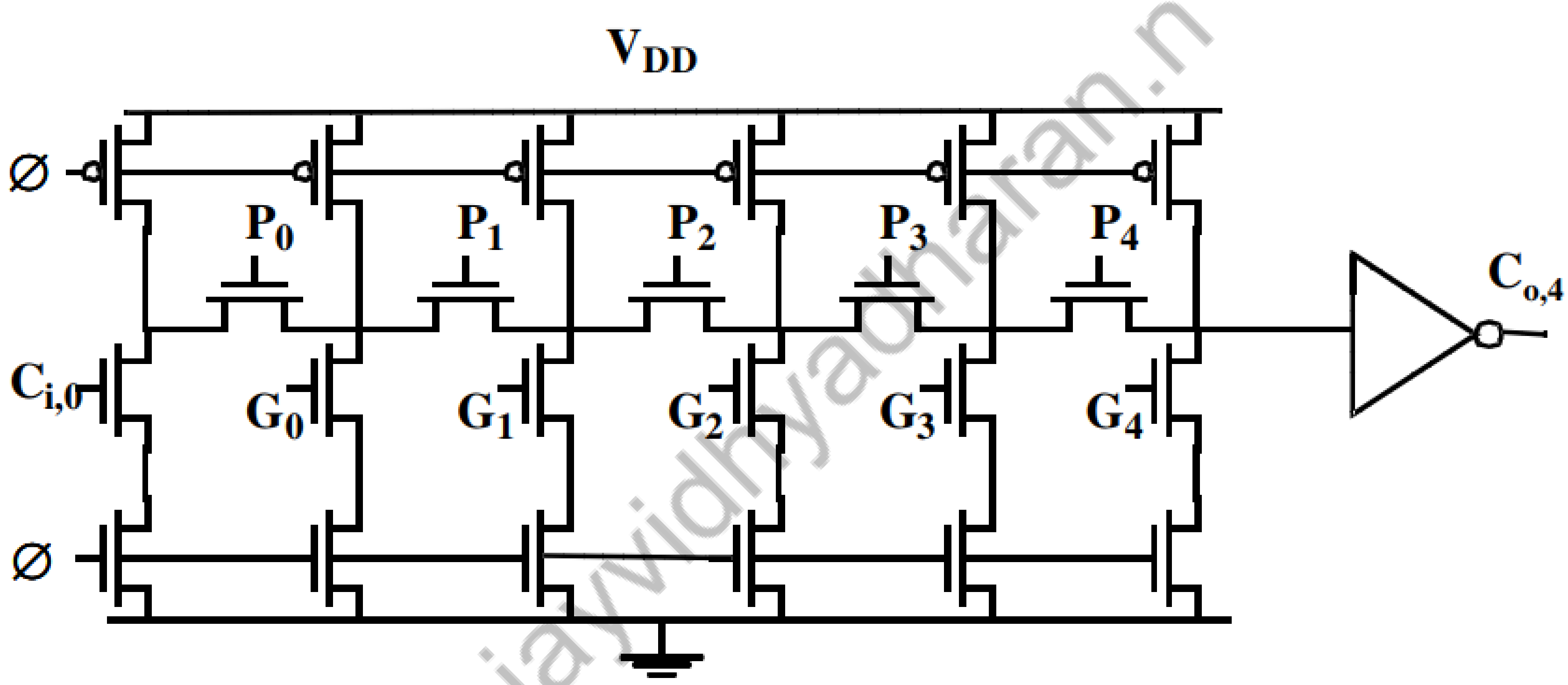
What is the critical path?

$$t_p = t_{setup} + Mt_{carry} + (N/M-1)t_{bypass} + Mt_{carry} + t_{sum}\text{ , i.e. } O(N)$$

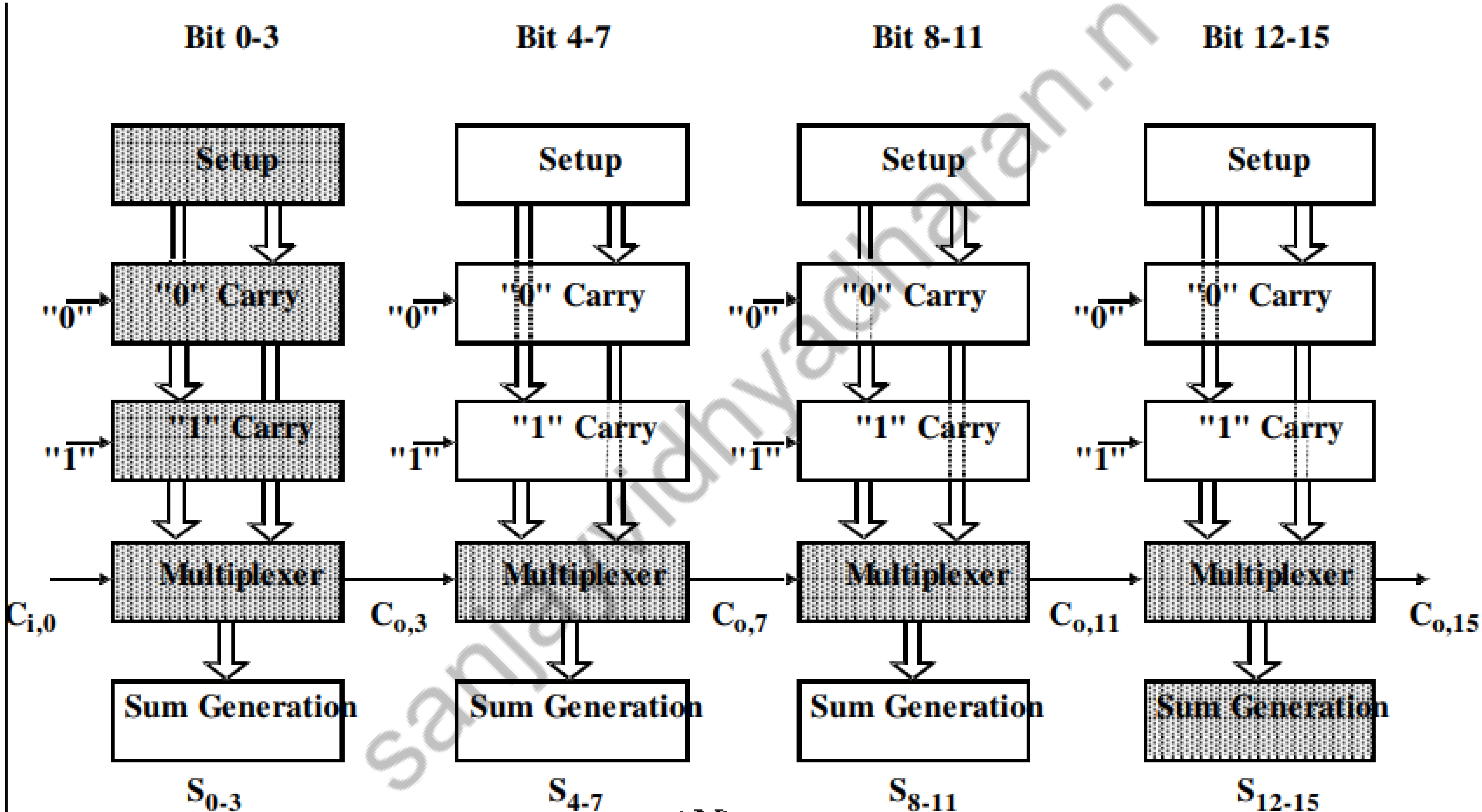
Carry Ripple versus Carry Bypass



Manchester Carry Chain

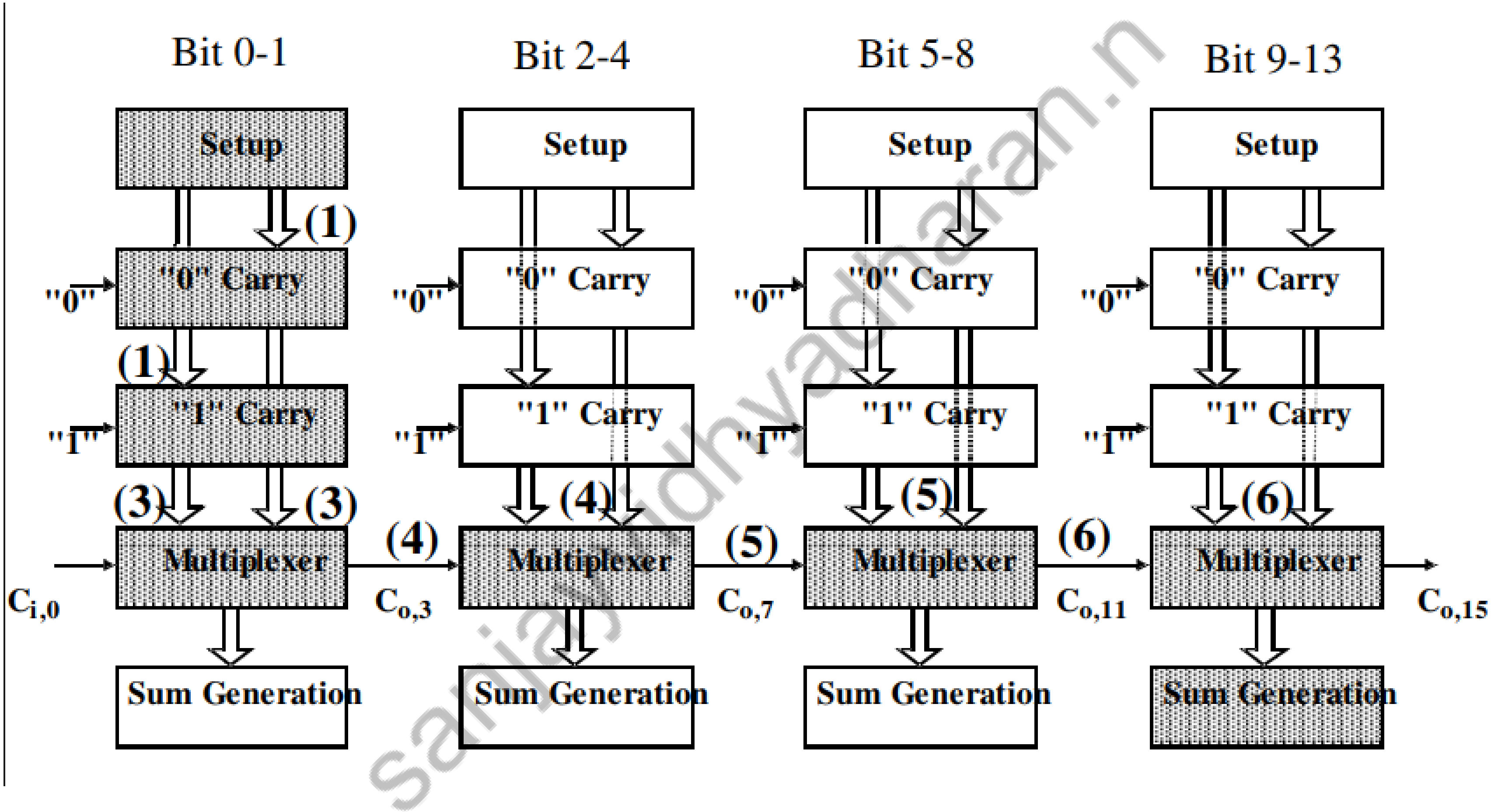


Linear Carry-Select Adder



$$t_{add} = t_{setup} + \left(\frac{N}{M}\right) t_{carry} + M t_{mux} + t_{sum}$$

Square Root Carry-Select Adder



$$t_{add} = t_{setup} + Mt_{carry} + \left(\sqrt{2N}\right)t_{mux} + t_{sum}$$

Square Root Carry-Select Adder

N Bit adder, $\quad M -$ Bits in First Stage , $\quad P -$ Number of Stages

$$N = M + (M+1) + (M+2) + (M+3) + \cdots . + (M+P-1)$$

Series: a, a+d, a+2d,……,a+(n-1)d

$s_n = n/2(2a + (n-1)d)$

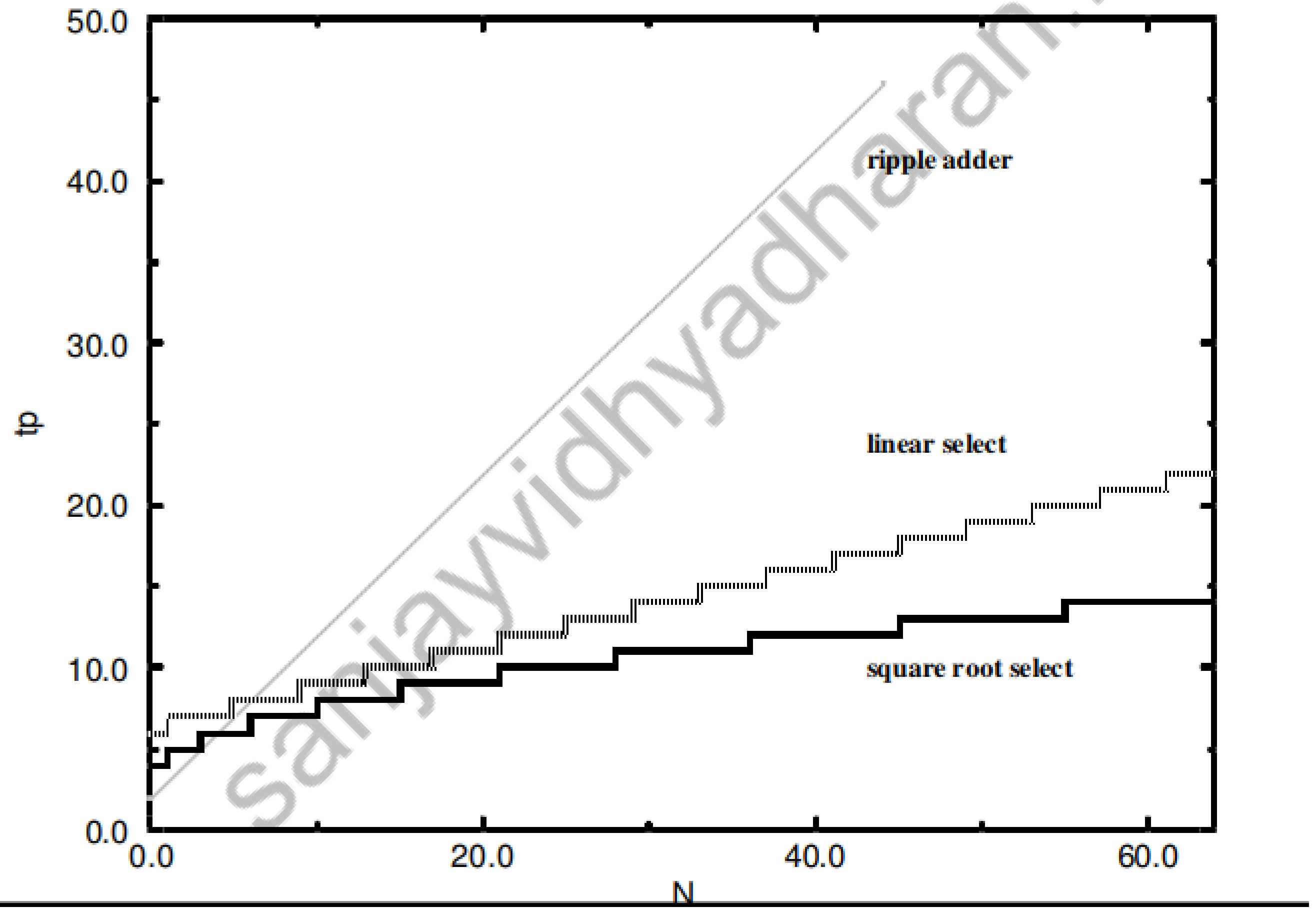
$$N = MP + \frac{P(P-1)}{2}$$

$$N = \frac{P^2}{2} + P(M - \frac{1}{2})$$

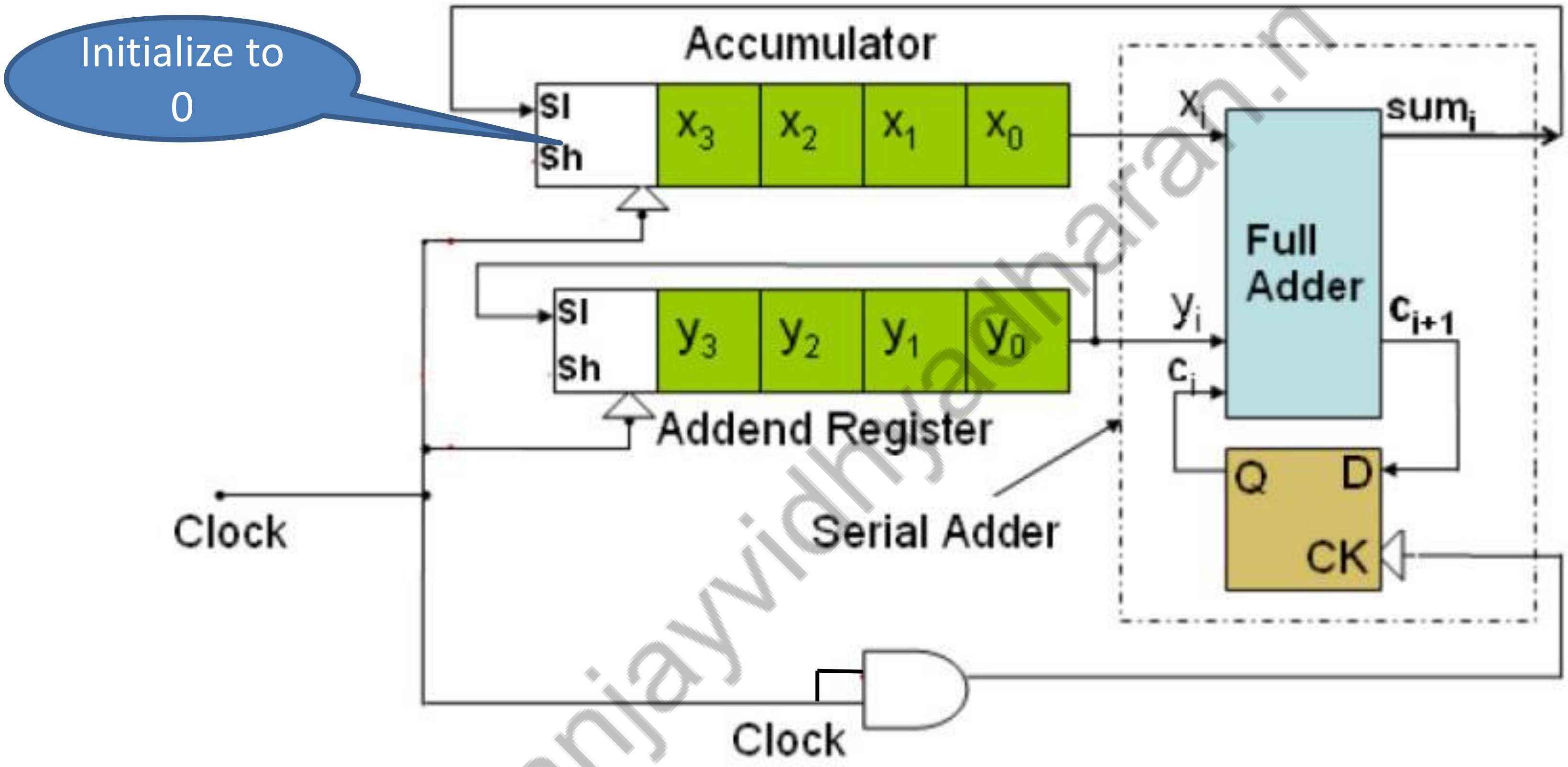
$$M \ll N \ (e.M = 2 \ and \ N = 64)$$

$$N \approx \frac{P^2}{2}$$

Adder Delays - Comparison



Serial Adder



Block Diagram of a 4-bit Serial Adder with Accumulator

4 Bit-Adder Subtractor

Add 4 & -3

```
  0100
  1101
1 0001
```

Add -4 & -5

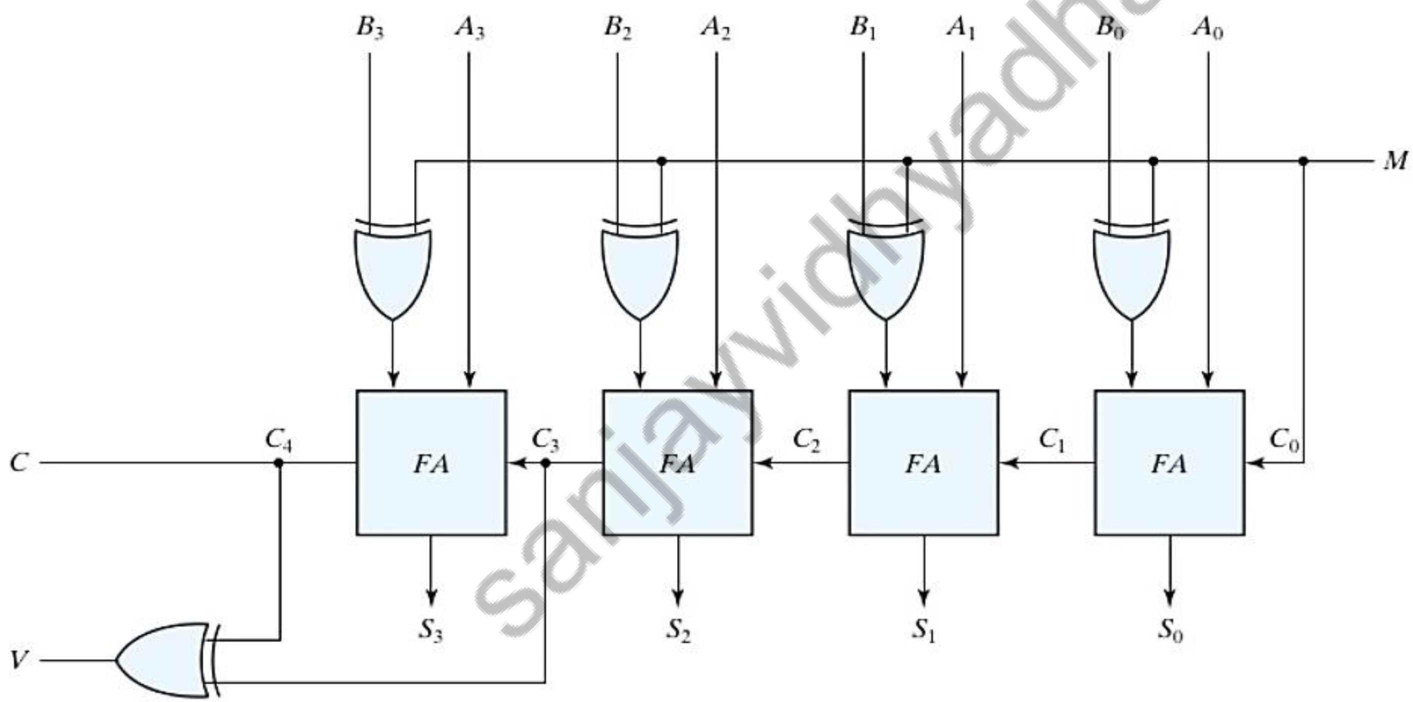
```
  1100
  1011
1 0111
```

Add -8 & 4

```
1000
0100
1100
```

Add 4 & 4

```
0100
0100
1000
```



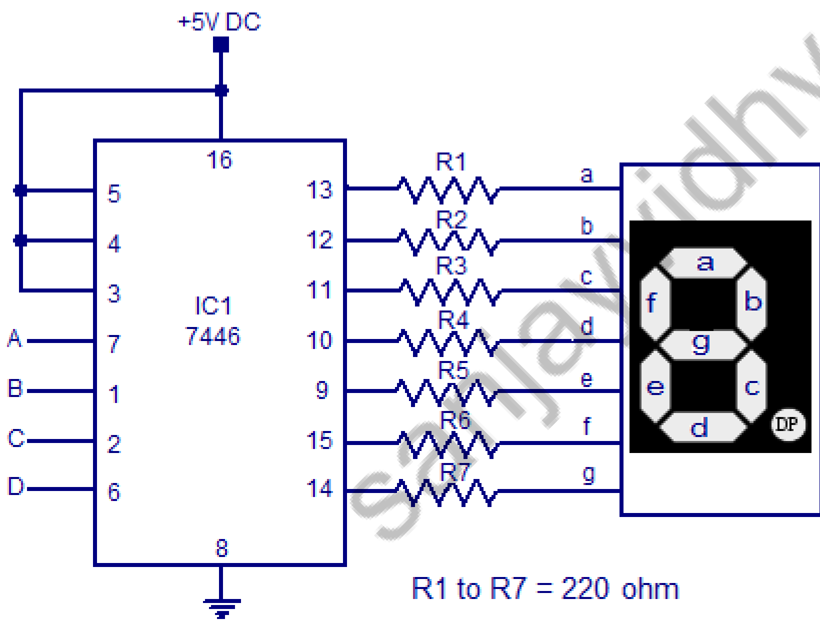
Overflow

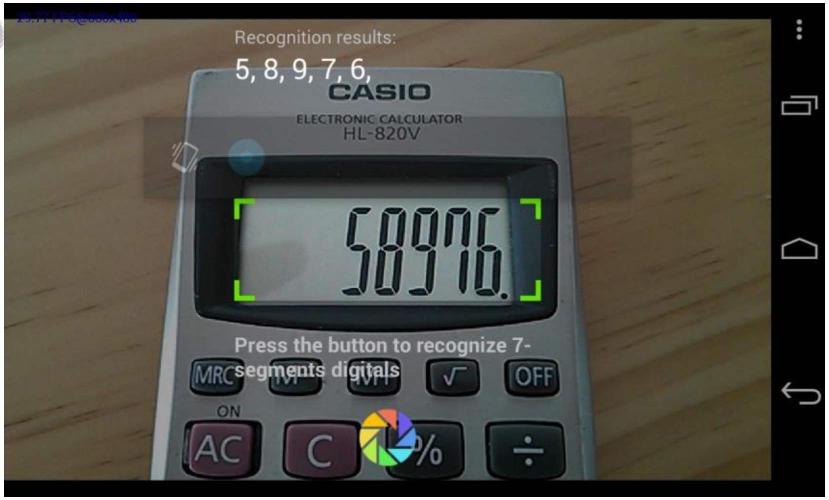
Decimal	2's comp.
7	0111
6	0110
5	0101
4	0100
3	0011
2	0010
1	0001
0	0000
-0	–
-1	1111
-2	1110
-3	1101
-4	1100
-5	1011
-6	1010
-7	1001
-8	1000

Binary Coded Decimal

General digital systems

User enters decimal → BCD i/p→ Binary i/p → compute in binary → Binary o/p → BCD o/p → Decimal output shown to user





Binary Coded Decimal

BCD addition

4 + 5	4	0 1 0 0	
	5	0 1 0 1	
	9	1 0 0 1	Expected Result
4 + 8	4	0 1 0 0	
	8	1 0 0 0	
		1 1 0 0	Is this expected Result ?

Expected answer is BCD of 12 0001 0010

Binary Coded Decimal

BCD addition

4 + 8	4	0 1 0 0
	8	1 0 0 0
Greater than 9		1 1 0 0
		0 1 1 0
		0 0 0 1 0 0 1 0
		1 2

Add correction of +6

= To skip 6 invalid states (10 - 15) BCDs

Binary Coded Decimal

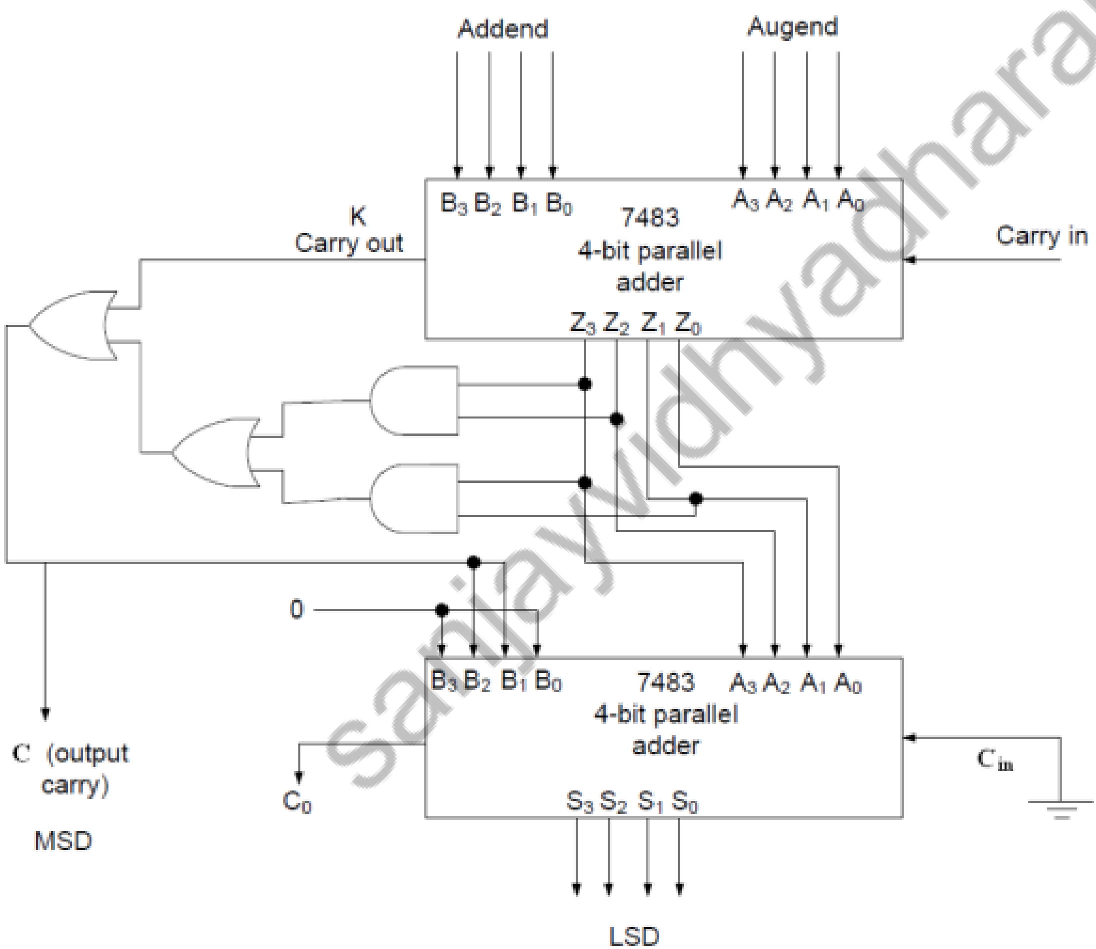
BCD addition

9 + 9	9 1 0 0 1		
	9 1 0 0 1		
Carry out generated	1 0 0 1 0	Expected result ?	
	0 1 1 0	Add correction of +6	
	0 0 0 1 1 0 0 0		
	1 8		

After addition if carry out is generated or if sum is greater than 9 there is need for correction

Binary Coded Decimal

BCD addition



0 0 0 0

0 0 0 1

0 0 1 0

0 0 1 1

0 1 0 0

0 1 0 1

0 1 1 0

0 1 1 1

1 0 0 0

1 0 0 1

Thank you