

BINARY ADDER

- The most basic arithmetic operation is the addition of two binary digits.
- A combinational circuit that performs the addition of two bits is **half adder**
- An adder performs the addition of 2 significant bits and a previous carry is called a **full adder**



- (Arithmetic) Addition of two binary digits
 - $0 + 0 = 0$, $0 + 1 = 1$, $1 + 0 = 1$, and $1 + 1 = 10$
 - The result has two components
 - the sum (S)
 - the carry (C)
- (Arithmetic) Addition of three binary digits

0	+	0	+	0	=	0	0
0	+	0	+	1	=	0	1
0	+	1	+	0	=	0	1
0	+	1	+	1	=	1	0
1	+	0	+	0	=	0	1
1	+	0	+	1	=	1	0
1	+	1	+	0	=	1	0
1	+	1	+	1	=	1	1



BINARY ADDER

Half Adder

- Adds 1-bit plus 1-bit
- Produces **Sum** and **Carry**

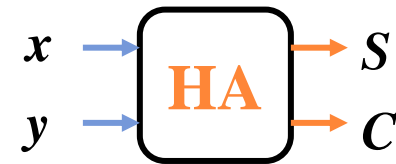


Fig: Block diagram of a half adder

Truth Table:

Input		Output	
x	y	C	S
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

$$\begin{array}{r} x \\ + y \\ \hline C \quad S \end{array}$$

Output equations:

$$S = x'y + xy' = x \oplus y$$

$$C = xy$$

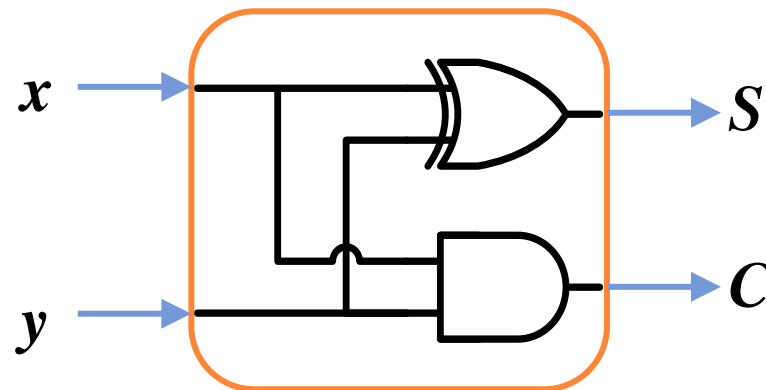


Fig: Circuit diagram of a half adder

- **Full Adder:** The full adder accepts 2 input bits and an input carry and generates a sum output and an output carry.

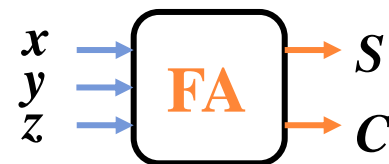


Fig: Block diagram of a full adder

Truth Table:

Input			Output	
x	y	z	C	S
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

Output equations:

	y	
	0	1
x	1	0
	z	

$$S = xy'z' + x'yz' + x'y'z + xyz = x \oplus y \oplus z$$

	y	
	0	1
x	0	1
	z	

$$C = xy + xz + yz$$



○ Full Adder

$$S = xy'z' + x'yz' + x'y'z + xyz = x \oplus y \oplus z$$

$$C = xy + xz + yz$$

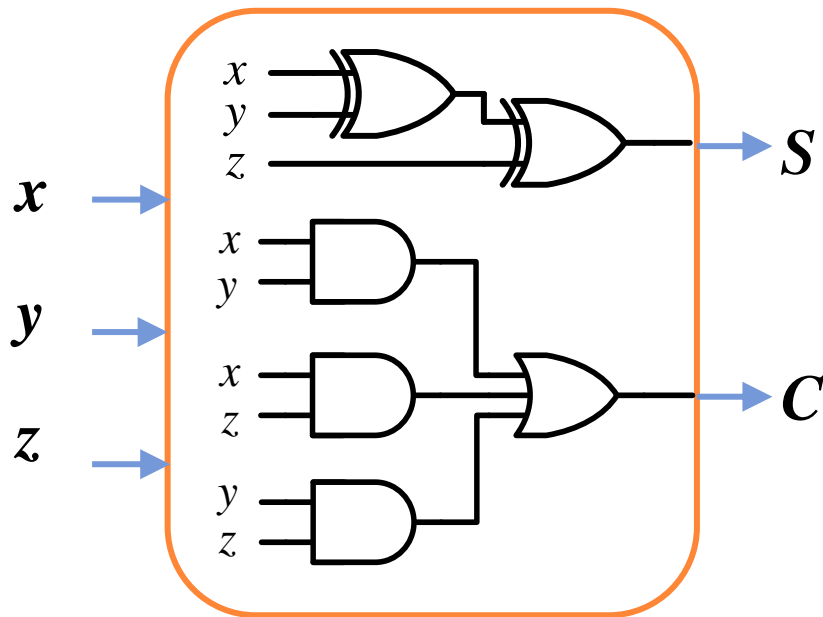


Fig: Circuit diagram of a full adder



ARRANGEMENT OF TWO HALF ADDERS TO FORM A FULL ADDER:

○ Full Adder

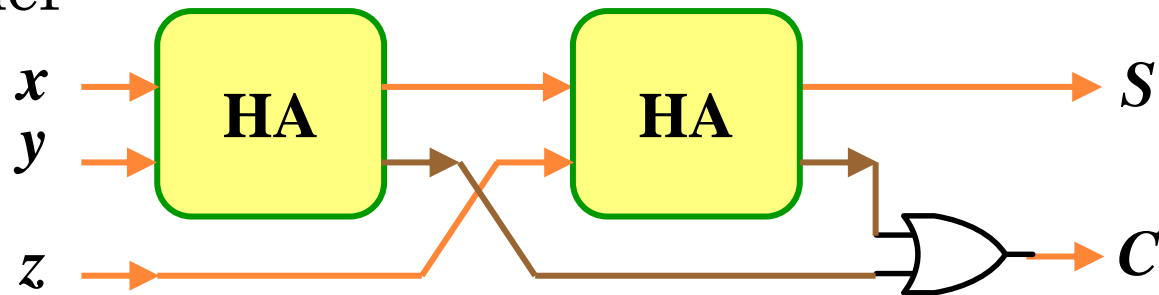


Fig: Block diagram of a full adder using 2 half adders

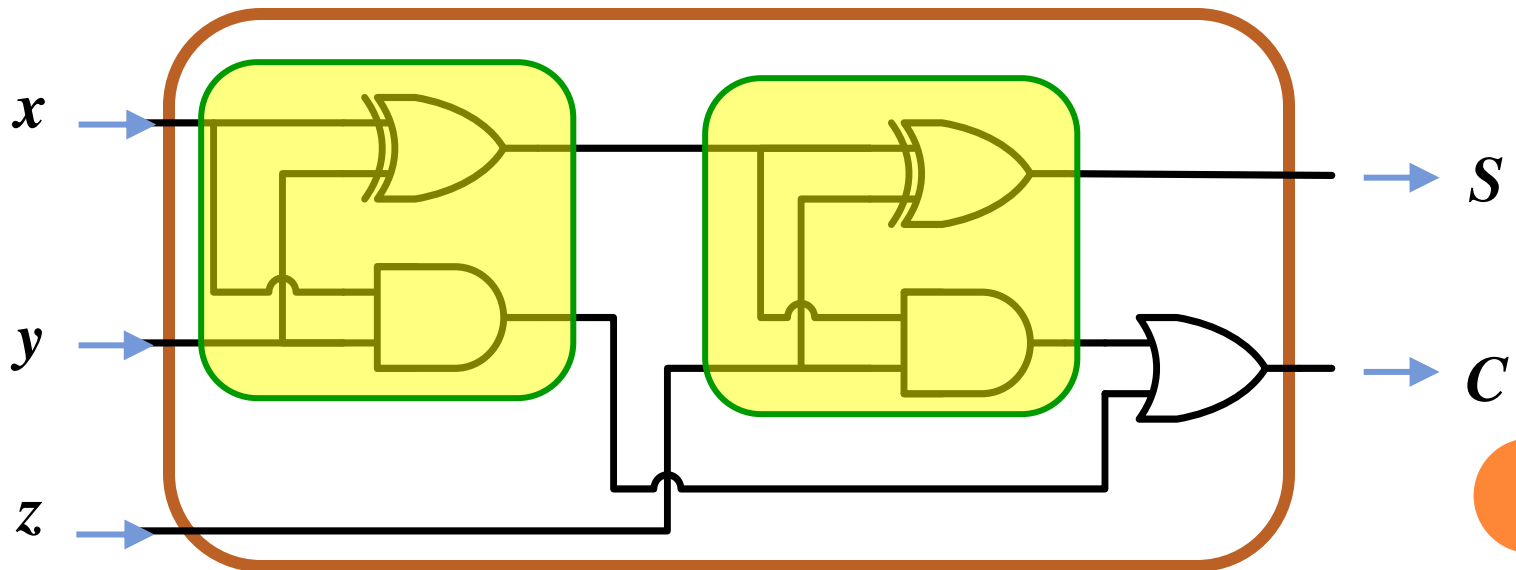


Fig: Circuit diagram of a full adder using 2 half adders

SELF STUDY:

Binary Subtractor:

- Half Subtractor
- Full Subtractor

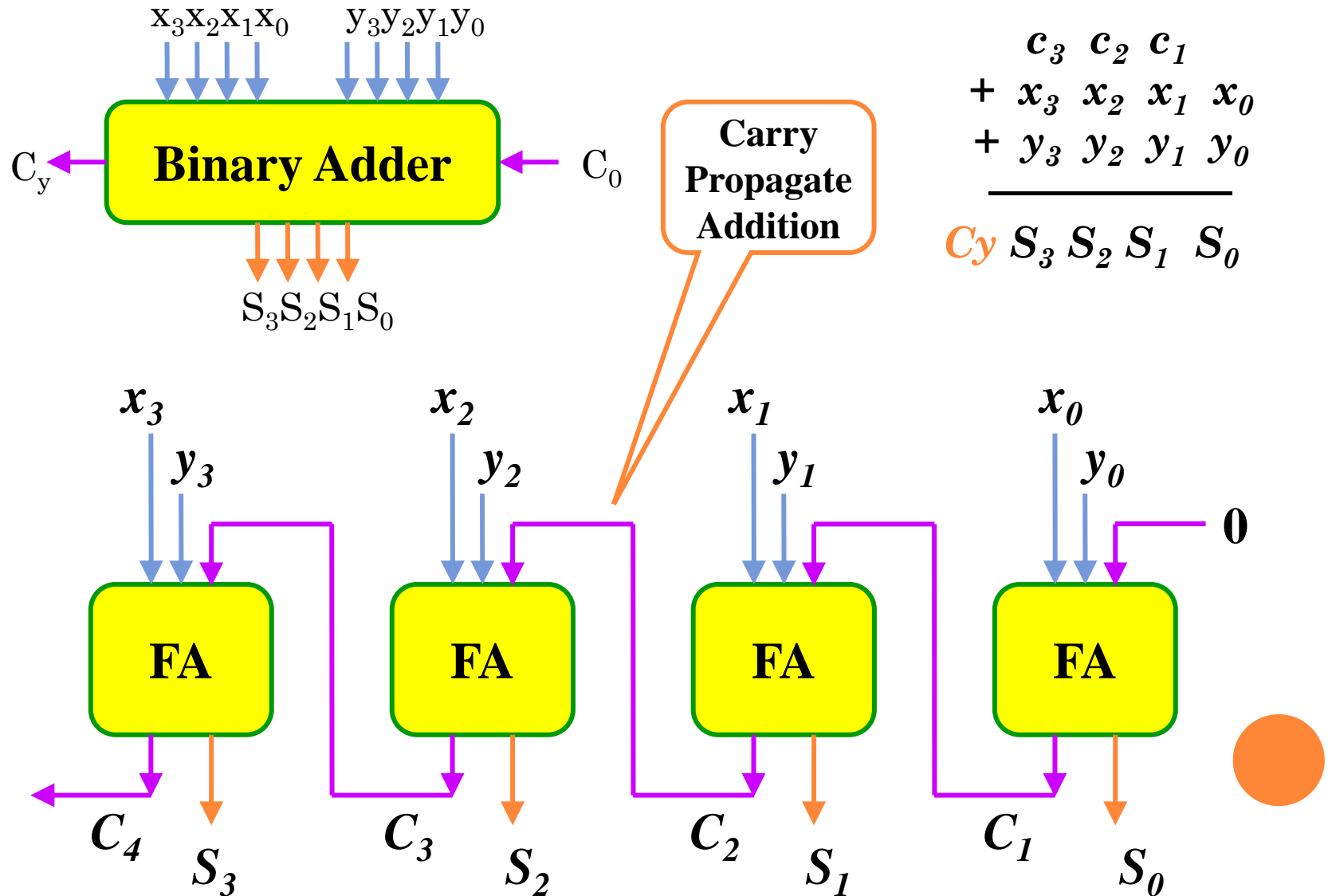


BINARY PARALLEL ADDER

- A binary parallel adder produces the arithmetic sum of two binary numbers in parallel.
- It consists of full adders connected in cascade, with the output carry from one full adder connected to the input carry of the next full adder.
- The parallel method uses n full adder circuits and all bits of A and B are applied simultaneously.



4 BIT BINARY PARALLEL ADDER



FOUR-BIT BINARY ADDER



- Carry bits must “ripple” through each stage of a multi-bit adder before the output settles down to the correct result.
- **Significantly slower --> Rippling effect of carry**
- For an n bit adder, Propagation delay = (Number of gate level x Average gate delay) x (number of bits)



CARRY LOOK AHEAD ADDER (CLA)

- In terms of the method used to handle carries in a parallel adder, there are 2 types:
 - i. The ripple carry adder
 - ii. The carry look ahead adder

A ripple carry adder is one in which the carry output of each full adder is connected to the carry input of the next higher order stage. The sum and the output carry of any stage can not be produced until the input carry occurs; this causes a time delay in the addition process. The carry propagation delay for each full adder is the time from the application of the input carry until the output carry occurs, assuming that the A and B inputs are already present.



- A method of speeding up the addition process by eliminating the ripple carry delay is called look ahead carry addition.
- The look ahead carry adder anticipates the output carry of each stage and based on the input bits of each stage, produces the output carry by either carry generation or carry propagation.
- In CLA, we define two new binary variables P_i and G_i where, $P_i = A_i \oplus B_i$, $G_i = A_i B_i$
- Here, G_i : carry generate, it produces an output carry when both A_i and B_i are one, regardless of the input carry.
- P_i : carry propagate, it is the term associated with the propagation of the carry from C_i to C_{i+1} .



CARRY LOOKAHEAD LOGIC

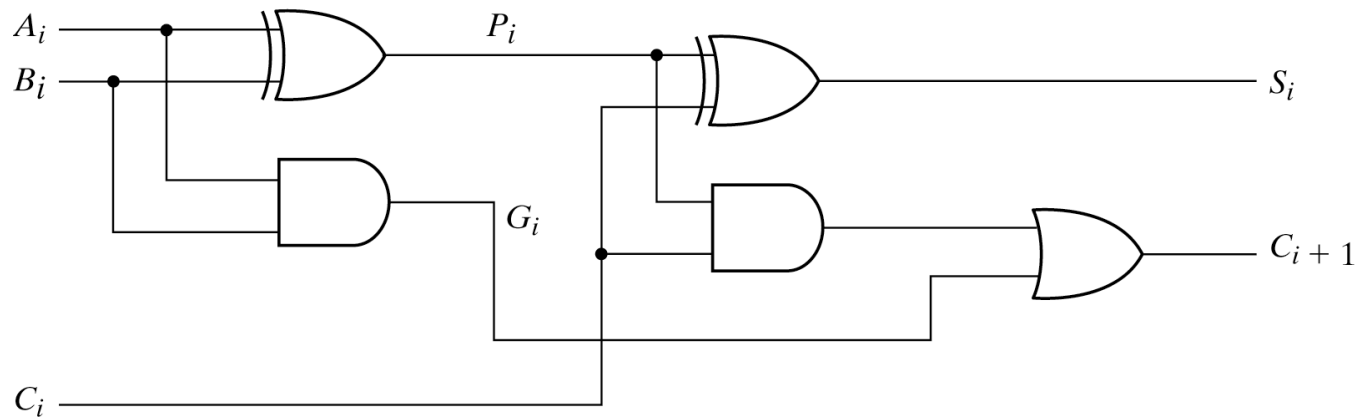


Fig. 4-10 Full Adder with P and G Shown

- So the output sum and carry of full adder can be expressed as:
- $S_i = P_i \oplus C_i$
- $C_{i+1} = G_i + C_i P_i$

- For example: for 3 bit CLA adder:
 - $S_1 = P_1 \oplus C_1 = P_1 \oplus 0 = P_1$ [$\because C_1 = \text{initial carry} = 0$]
 - $\therefore S_1 = A_1 \oplus B_1$ [$\because P_i = A_i \oplus B_i$],
 $C_1 = \text{initial carry} = 0$.
 - Similarly,
- $$C_2 = G_1 + P_1 C_1$$
- $$= A_1 B_1 + 0 \text{ } [\because C_{i+1} = G_i + C_i P_i \text{ and } G_i = A_i B_i]$$
- $$= A_1 B_1$$
- $$S_2 = P_2 \oplus C_2 = (A_2 \oplus B_2) \oplus A_1 B_1$$
- $$C_3 = G_2 + P_2 C_2 = A_2 B_2 + (A_2 \oplus B_2) A_1 B_1$$
- $$S_3 = P_3 \oplus C_3 = (A_3 \oplus B_3) \oplus [A_2 B_2 + (A_2 \oplus B_2) A_1 B_1]$$
- $$C_4 = G_3 + P_3 C_3$$
- $$= A_3 B_3 + (A_3 \oplus B_3) [A_2 B_2 + (A_2 \oplus B_2) A_1 B_1]$$



CARRY LOOKAHEAD LOGIC

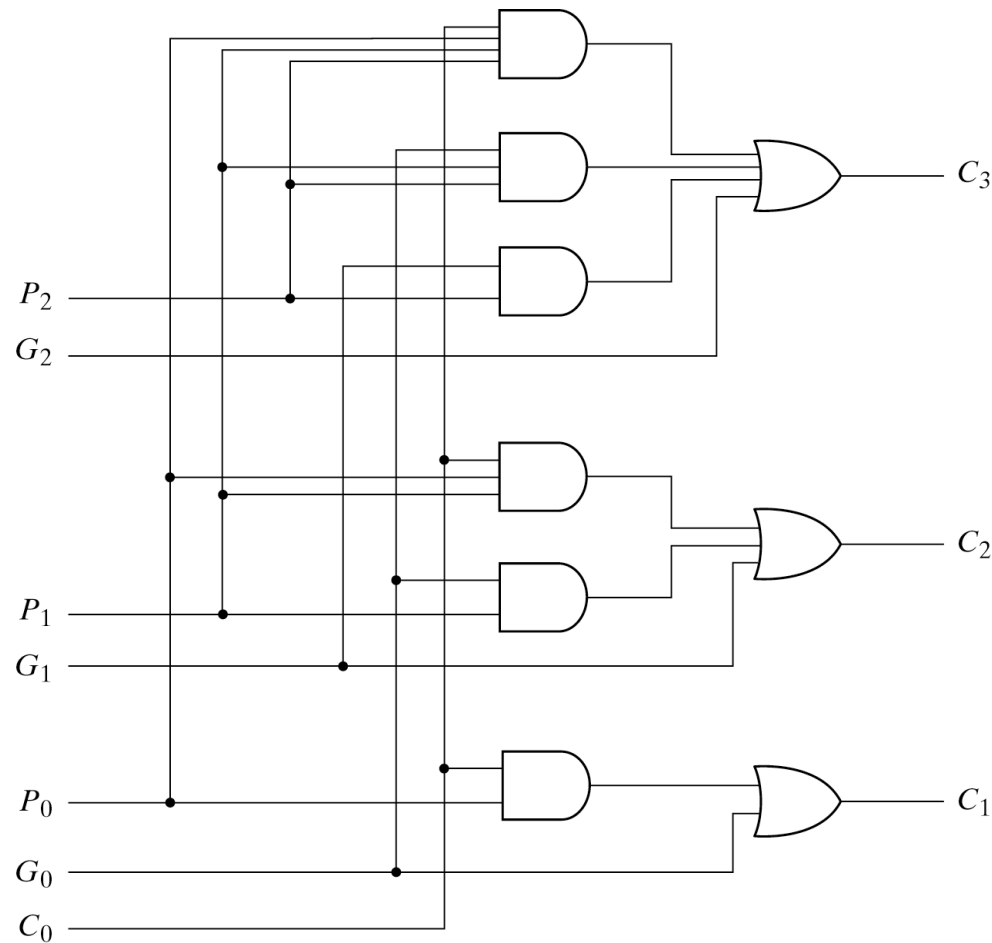


Fig. 4-11 Logic Diagram of Carry Lookahead Generator



4-BIT ADDER WITH CARRY LOOKAHEAD

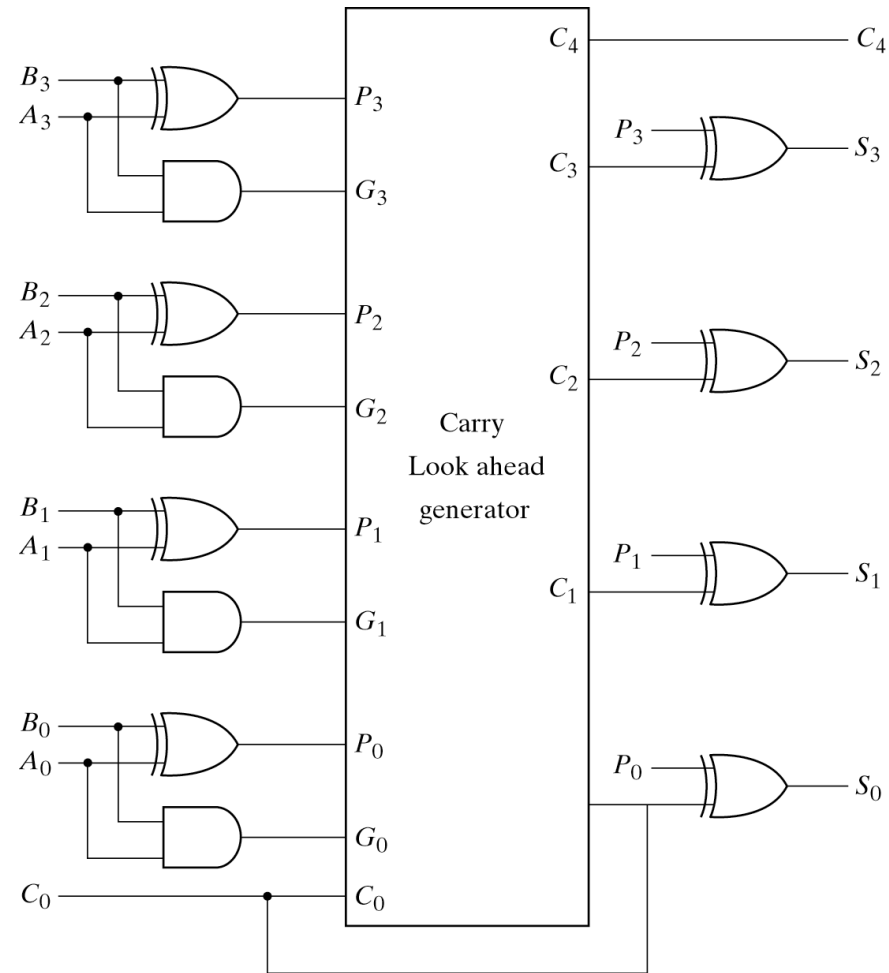
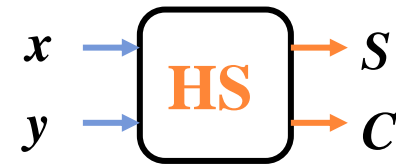


Fig. 4-12 4-Bit Adder with Carry Lookahead

BINARY SUBTRACTOR [SELF STUDY]

- Half Subtractor
 - Produces $x - y$
 - D – difference

x y	B D
0 0	0 0
0 1	1 1
1 0	0 1
1 1	0 0



$$\begin{array}{r} x \\ - y \\ \hline B \quad D \end{array}$$

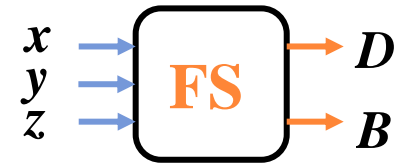
- $D = x'y + xy' = S$ of half adder
- $B = x'y$



[SELF STUDY]

Full Subtractor

- $(x - y) - z$; where z represents a borrow



x	y	z	B	D
0	0	0	0	0
0	0	1	1	1
0	1	0	1	1
0	1	1	1	0
1	0	0	0	1
1	0	1	0	0
1	1	0	0	0
1	1	1	1	1

	yz			
	00	01	10	11
$x=0$	0	1	0	1
$x=1$	1	0	1	0

$$D = xy'z' + x'yz' + x'y'z + xyz = x \oplus y \oplus z$$

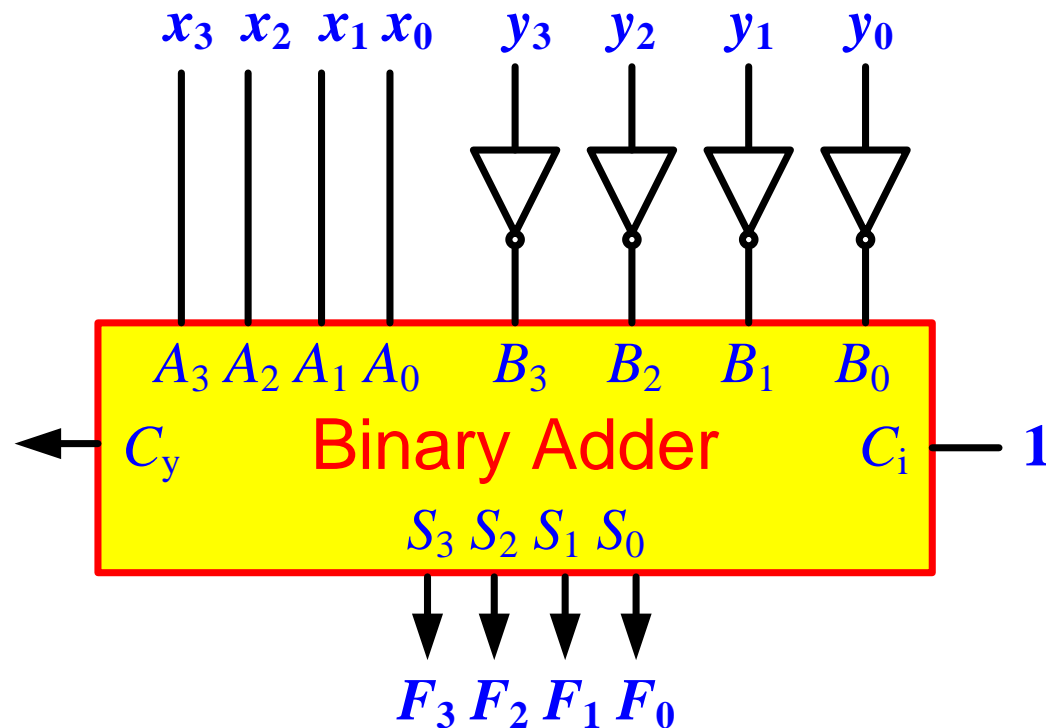
	yz			
	00	01	10	11
$x=0$	0	1	1	1
$x=1$	0	0	1	0

$$B = x'y + x'z + yz$$



BINARY SUBTRACTOR [SELF STUDY]

- Use 2's complement with binary adder
 - $x - y = x + (-y) = x + y' + 1$



BCD ADDER

- BCD adder
- A BCD adder is a circuit that adds two BCD digits in parallel and produces a sum digit also in BCD.
 - The output sum cannot be greater than 19 (9+9+1)



BCD ADDER

- 4-bits plus 4-bits
- Operands and Result: 0 to 9

$X + Y$	$x_3 x_2 x_1 x_0$	$y_3 y_2 y_1 y_0$	Sum	Cy	$S_3 S_2 S_1 S_0$
0 + 0	0 0 0 0	0 0 0 0	= 0	0	0 0 0 0
0 + 1	0 0 0 0	0 0 0 1	= 1	0	0 0 0 1
0 + 2	0 0 0 0	0 0 1 0	= 2	0	0 0 1 0
0 + 9	0 0 0 0	1 0 0 1	= 9	0	1 0 0 1
1 + 0	0 0 0 1	0 0 0 0	= 1	0	0 0 0 1
1 + 1	0 0 0 1	0 0 0 1	= 2	0	0 0 1 0
1 + 8	0 0 0 1	1 0 0 0	= 9	0	1 0 0 1
1 + 9	0 0 0 1	1 0 0 1	= A	0	1 0 1 0
2 + 0	0 0 1 0	0 0 0 0	= 2	0	0 0 1 0
9 + 9	1 0 0 1	1 0 0 1	= 12	1	0 0 1 0

0001 1000

$$\begin{array}{r}
 + x_3 x_2 x_1 x_0 \\
 + y_3 y_2 y_1 y_0 \\
 \hline
 \end{array}$$

Cy $S_3 S_2 S_1 S_0$

Invalid Code

Wrong BCD Value

BCD ADDER

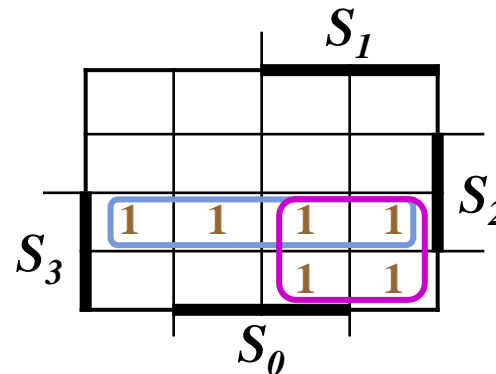
$X + Y$	$x_3 x_2 x_1 x_0$	$y_3 y_2 y_1 y_0$	Sum	Cy	$S_3 S_2 S_1 S_0$	<i>Required BCD Output</i>	<i>Value</i>
9 + 0	1 0 0 1	0 0 0 0	= 9	0	1 0 0 1	0 0 0 0 1 0 0 1	= 9
9 + 1	1 0 0 1	0 0 0 1	= 10	0	1 0 1 0	0 0 0 1 0 0 0 0	= 16 ✕
9 + 2	1 0 0 1	0 0 1 0	= 11	0	1 0 1 1	0 0 0 1 0 0 0 1	= 17 ✕
9 + 3	1 0 0 1	0 0 1 1	= 12	0	1 1 0 0	0 0 0 1 0 0 1 0	= 18 ✕
9 + 4	1 0 0 1	0 1 0 0	= 13	0	1 1 0 1	0 0 0 1 0 0 1 1	= 19 ✕
9 + 5	1 0 0 1	0 1 0 1	= 14	0	1 1 1 0	0 0 0 1 0 1 0 0	= 20 ✕
9 + 6	1 0 0 1	0 1 1 0	= 15	0	1 1 1 1	0 0 0 1 0 1 0 1	= 21 ✕
9 + 7	1 0 0 1	0 1 1 1	= 16	1	0 0 0 0	0 0 0 1 0 1 1 0	= 22 ✕
9 + 8	1 0 0 1	1 0 0 0	= 17	1	0 0 0 1	0 0 0 1 0 1 1 1	= 23 ✕
9 + 9	1 0 0 1	1 0 0 1	= 18	1	0 0 1 0	0 0 0 1 1 0 0 0	= 24 ✕

+ 6

BCD ADDER

- Correct Binary Adder's Output (+6)
 - If the result is between 'A' and 'F'
 - If $Cy = 1$

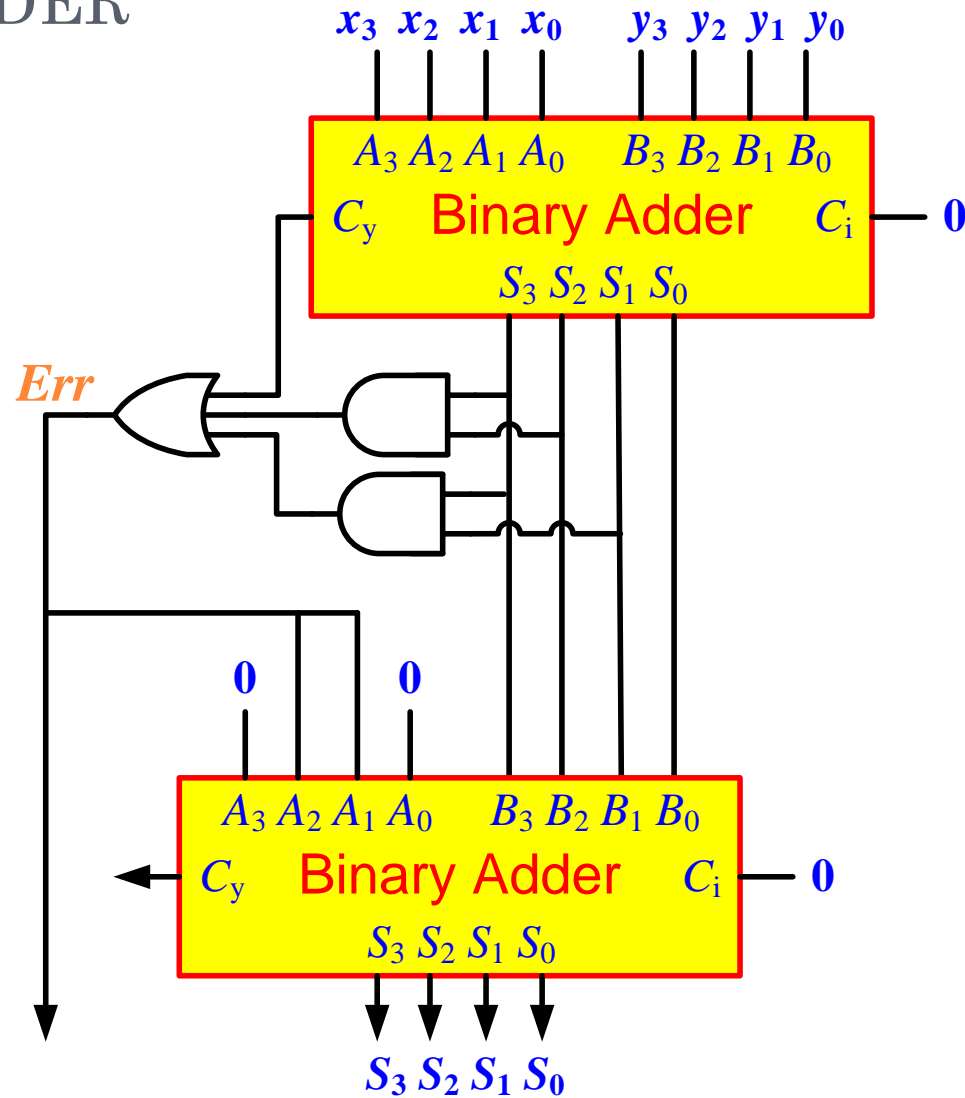
$S_3 S_2 S_1 S_0$	<i>Err</i>
0 0 0 0	0
1 0 0 0	0
1 0 0 1	0
1 0 1 0	1
1 0 1 1	1
1 1 0 0	1
1 1 0 1	1
1 1 1 0	1
1 1 1 1	1



$$Err = S_3 S_2 + S_3 S_1$$



BCD ADDER



MAGNITUDE COMPARATOR

- The comparison of two numbers is an operation that determines if one number is greater than, less than or equal to the other number.
- A magnitude comparator is a combinational circuit that compares two numbers, A and B, and determines their relative magnitudes.
- The outcome of the comparison is specified by three binary variables that indicate whether $A > B$, $A = B$ or $A < B$



THE 1-BIT MAGNITUDE-COMPARATOR

Input		Outputs		
X	Y	X>Y	X=Y	X<Y
0	0	0	1	0
0	1	0	0	1
1	0	1	0	0
1	1	0	1	0

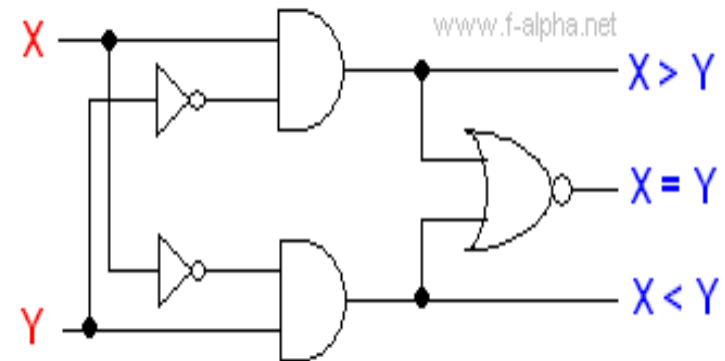


Fig: 1 bit Comparator Circuit

- Output Equations:
- F1: $X > Y = XY'$
- F2: $X = Y = X'Y' + XY = X \odot Y$
- F3: $X < Y = X'Y$

SELF STUDY

- 2 bit, 3 bit, 4 bit comparator [See class lecture and book]

