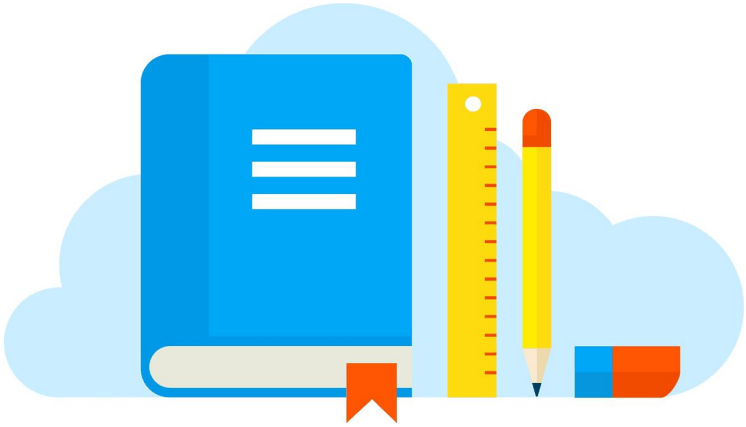


Class Notes

Lecture 7: Combinational Circuits - 1

Two classes of logic circuits:

- combinational
- Sequential

Combinational Circuit:

•A combinational circuit consists of logic gates whose outputs at any time are determined directly from the present combination of inputs without regard to previous inputs.

•A combinational circuit performs a specific information-processing operation fully specified logically by a set of Boolean functions.

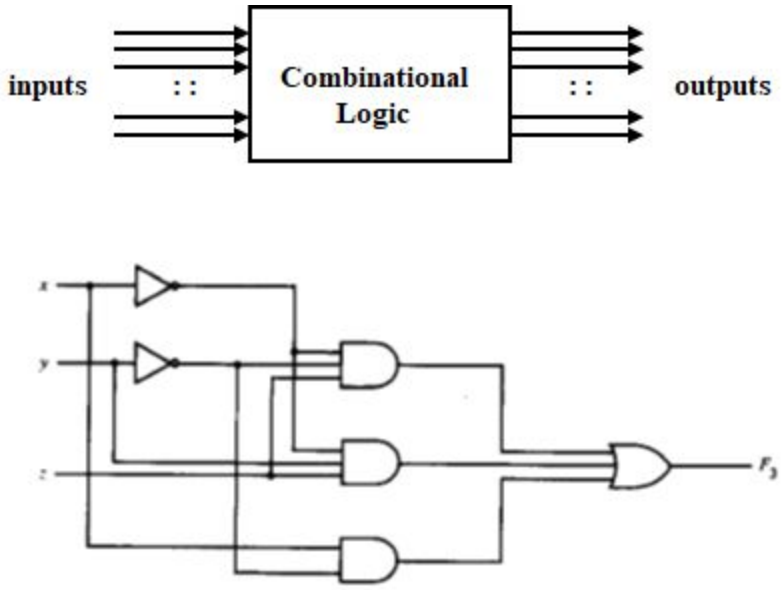


Figure showing a block level and a gate level diagram of combinational circuit

Sequential Circuit:

•Sequential circuits employ memory elements in addition to logic gates. Their outputs are a function of the inputs and the state of memory elements

•The state of memory elements, in turn, is a function of previous inputs. As a consequence, the outputs of a sequential circuit depend not only on present inputs, but also on past inputs.

•Sequential circuits will be covered in the future chapters.

In short,

•A combinational circuit consists of input variables, logic gates and output variables.

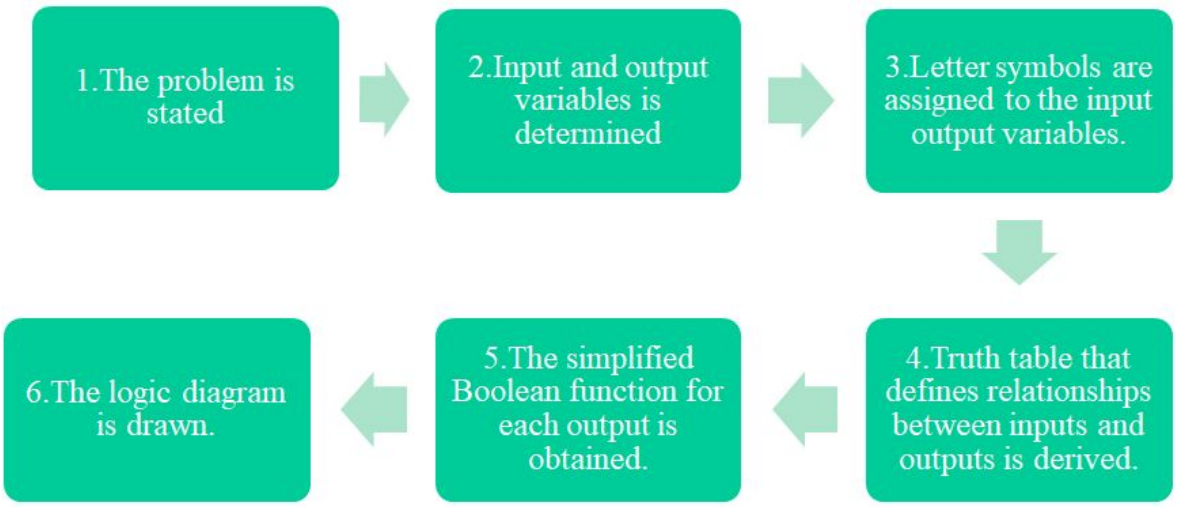
•A combinational circuit can be described by m Boolean functions, one for each output variable. Each output function is expressed in terms of the n input variables.

•There **is no feedback path** or presence of memory element

Design Procedure of a Combinational Circuit:

The design of combinational circuits starts from the verbal outline of the problem and ends in a logic circuit diagram.

The procedure involves the following steps:



Design Procedure of a Half Adder:

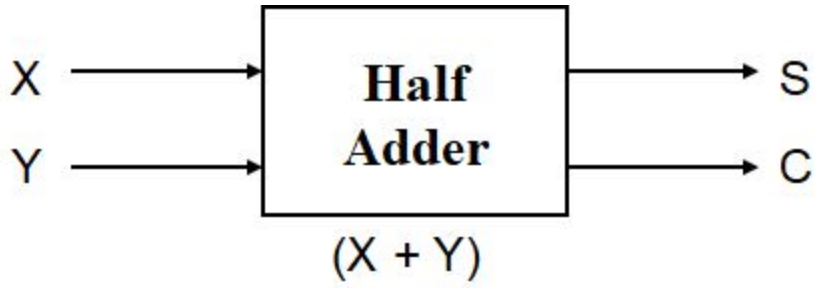
The combinational circuit that performs the addition of two bits is called a half-adder.

1) State Problem:

Example: Build a Half Adder to add two bits

2) Determine and label the inputs & outputs of the circuit.

Example: Two inputs and two outputs labeled, as follows:



3) Draw truth table.

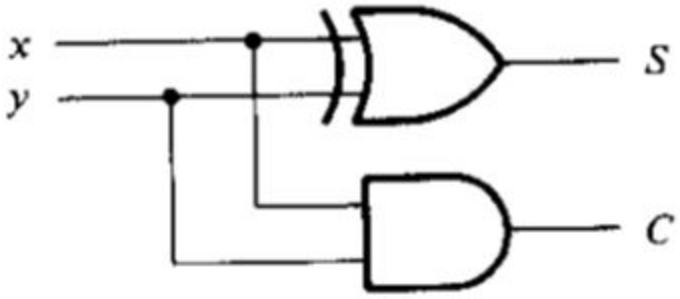
X	Y	C	S
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

4) Obtain a simplified Boolean function.

Example: C = XY

S = X'Y + XY' = X⊕Y

5) Draw a logic diagram.



(e) $S = x \oplus y$
$C = xy$

Design Procedure of a Full Adder

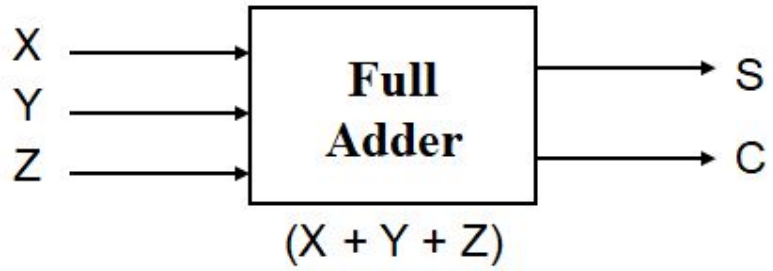
-Half-adder adds up only two bits.

-To add two binary numbers, we need to add 3 bits (including the carry). Example:

	1	1	1		carry
	0	0	1	1	X
+	0	1	1	1	Y
	1	0	1	0	S

-

-Need Full Adder (so called as it can be made from two half-adders).



-Truth Table:

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

Note:

Z - carry in (to the current position)

C - carry out (to the next position)

From truth table

C = X'YZ+XY'Z+XYZ'+XYZ

S = X'Y'Z + XYZ + X'YZ'+XY'Z'

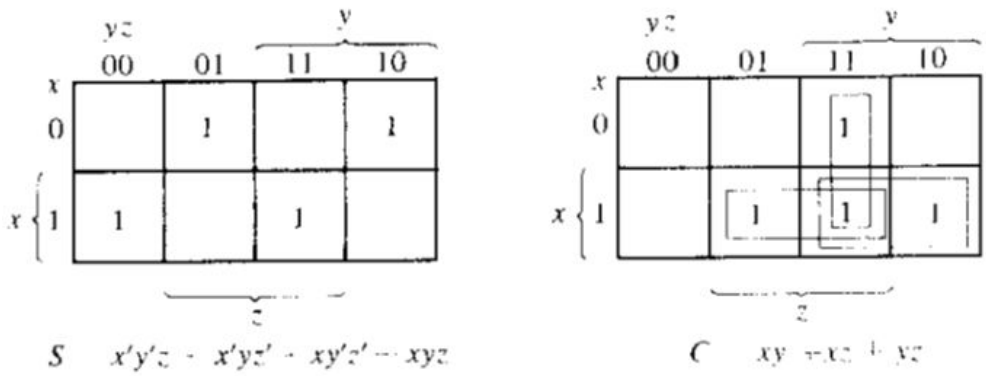


FIGURE 4-3
Maps for a full-adder

Alternative formulae using algebraic manipulation, Using K-map, simplified SOP form:

C = XY + XZ + YZ

S = X'Y'Z + XYZ + X'YZ'+XY'Z'

Note:

How to turn (X+Y) to ((X⊕Y) + XY)

If F=(X+Y)

=(X+Y)(Y+Y')

=XY+XY'+YY+YY'

=XY+XY'+Y

=XY'+XY+Y

=XY'+Y(X+1)

=XY'+Y

=XY'+Y(X'+X)

=XY'+X'Y+XY

=(X⊕Y) + XY)

C = XY + XZ + YZ

= XY + (X + Y)Z

= XY + ((X⊕Y) + XY)Z

= XY + (X⊕Y)Z + XYZ

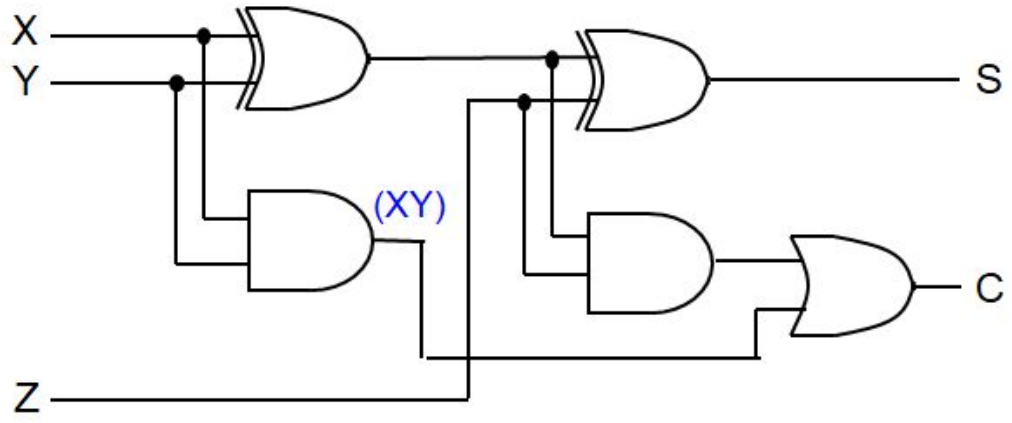
= XY(1+Z) + (X⊕Y)Z

= XY + (X⊕Y)Z

S = X'Y'Z + X'YZ' + XY'Z' + XYZ

= X'(Y'Z + YZ') + X(Y'Z' + YZ)
= X'(Y⊕Z) + X(Y⊕Z)'
= X⊕(Y⊕Z) or (X⊕Y)⊕Z

C = XY + (X⊕Y)Z
S = (X⊕Y)⊕Z



So, a Full Adder can be made from two Half-Adders (+ OR gate).

Design Methods:

Different combinational circuit design methods:

1. Gate-level method (with logic gates)
2. Block-level design method

Design methods make use of logic gates and useful functional blocks.

- These are available as Integrated Circuit (IC) chips.

So, Why is it better to use IC than the classical gate method?

- IC consists of gates all packed up together internally, which keeps the gates safe inside as well as makes it economical by reducing external interconnection wires

Type of IC chips (based on packing density) :

1. Small-scale integration (SSI): up to 12 gates
2. Medium-scale integration (MSI): 12-99 gates
3. Large-scale integration (LSI): 100-9999 gates
4. Very large-scale integration (VLSI): 10,000-99,999 gates
5. Ultra large-scale integration (ULSI): > 100,000 gates

Main objectives of circuit design:

(i) reduce cost

- reduce number of gates (for SSI circuits)
- reduce IC packages (for complex circuits)

(ii) increase speed

(iii) design simplicity (reuse blocks where possible)

Block-Level Design Method:

- More complex circuits can be built using block-level method.
- In general, block-level design method (as opposed to gate-level design) relies on algorithms or formulae of the circuit, which are obtained by decomposing the main problem to sub-problems recursively (until small enough to be directly solved by blocks of circuits).

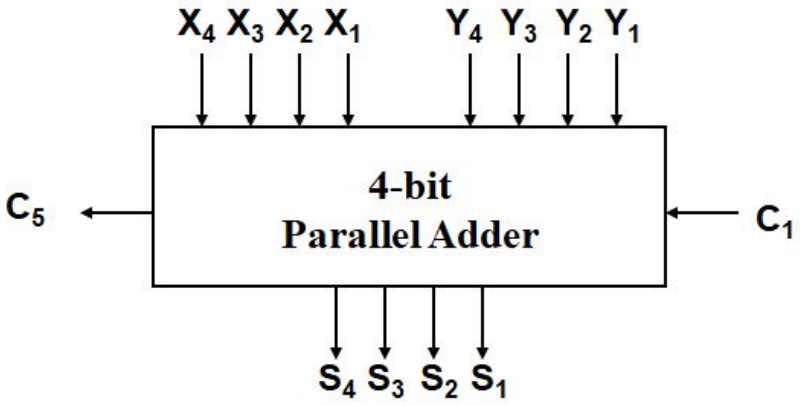
What if we want to add more than 3 bit?

N-bit Full adder Can be

- Series adder: consist of 1 full adder and a storage for saving Carry out (which becomes Carry in for next bits addition) (Not required in this chapter)
- Parallel adder: Consist of N adder and all N-bits of both numbers are given as input simultaneously. Carry input, in the least significant position, must be 0.

4-bit Parallel Adder:

Consider a circuit to add two 4-bit numbers together and a carry-in, to produce a 5-bit result:



Black-box view of 4-bit parallel adder

5-bit result is sufficient because the largest result is:

$$(1111)_2+(1111)_2+(1)_2 = (11111)_2$$

Traditional design techniques should not be used.

Truth table for 9 inputs very big, i.e. 2^9=512 entries:

$X_4X_3X_2X_1$	$Y_4Y_3Y_2Y_1$	C_1	C_5	$S_4S_3S_2S_1$
0 0 0 0	0 0 0 0	0	0	0 0 0 0
0 0 0 0	0 0 0 0	1	0	0 0 0 1
0 0 0 0	0 0 0 1	0	0	0 0 0 1
...	...	...	...	...
0 1 0 1	1 1 0 1	1	1	0 0 1 1
...	...	...	...	...
1 1 1 1	1 1 1 1	1	1	1 1 1 1

Simplification is very complicated as no. of input is 9 so 512(=2^9) possible combinations will be there in the truth table!

Thus an Alternative design is used.

Addition formulae for each pair of bits (with carry in),

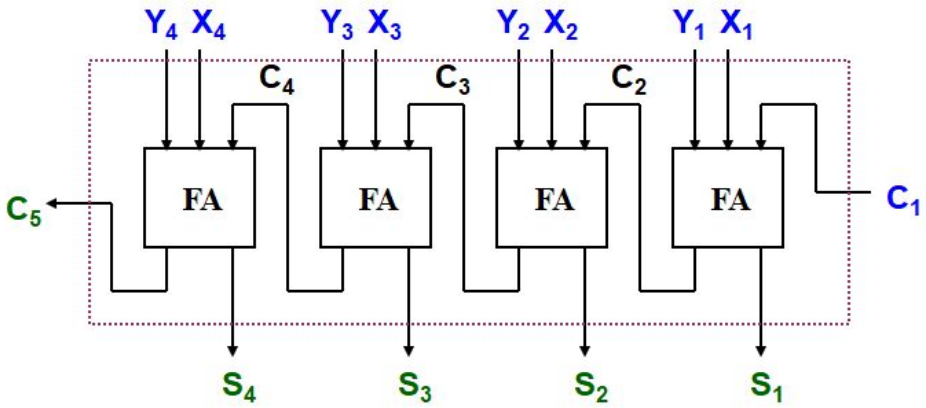
$$C_{i+1}S_i = X_i + Y_i + C_i$$

has the same function as a full adder.

$$C_{i+1} = X_i Y_i + (X_i \oplus Y_i) C_i$$

$$S_i = X_i \oplus Y_i \oplus C_i$$

Cascading 4 full adders via their carries, we get:



Input
Output

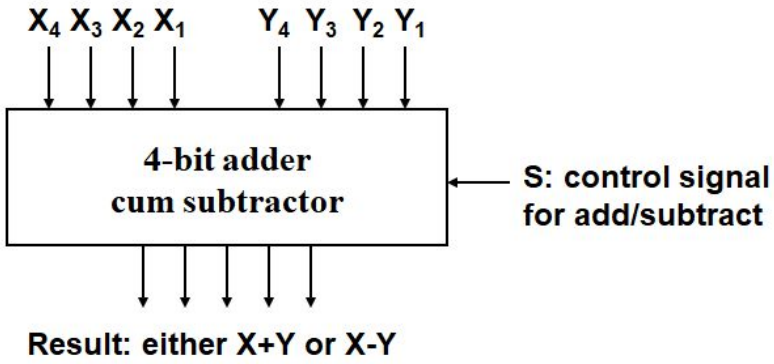
So, n-bit full adder requires n full- adders.

Practice Problems:

4-bit Parallel Adder cum Subtractor:

Subtraction can be performed through addition using 2s-complement numbers.

Hence, we can design a circuit which can perform both addition and subtraction, using a parallel adder.



The control signal **S=0** means add
S=1 means subtract

Recall that:

X-Y = X + (-Y)

= X + (2's complement of Y)

= X + (1's complement of Y) +1

X+Y = X + (Y)

Design requires:

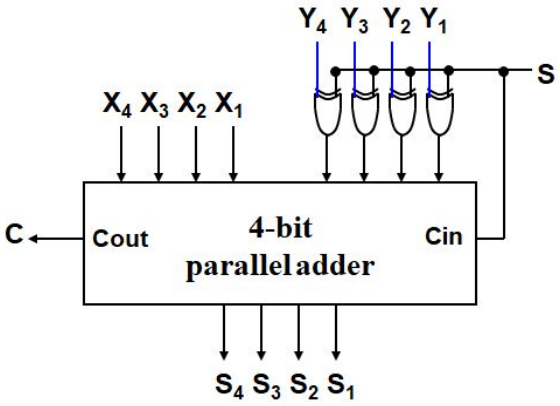
(i) XOR gates:

such that: output = Y when S=0
= Y' when S=1

(ii) S connected to carry-in.

S	Y	Output
0	0	0
0	1	1
1	0	1
1	1	0

Adder cum subtractor circuit:



A 4-bit adder cum subtractor

Analysis:

If S=1, then

X + (1's complement of Y) +1

appears as the result.

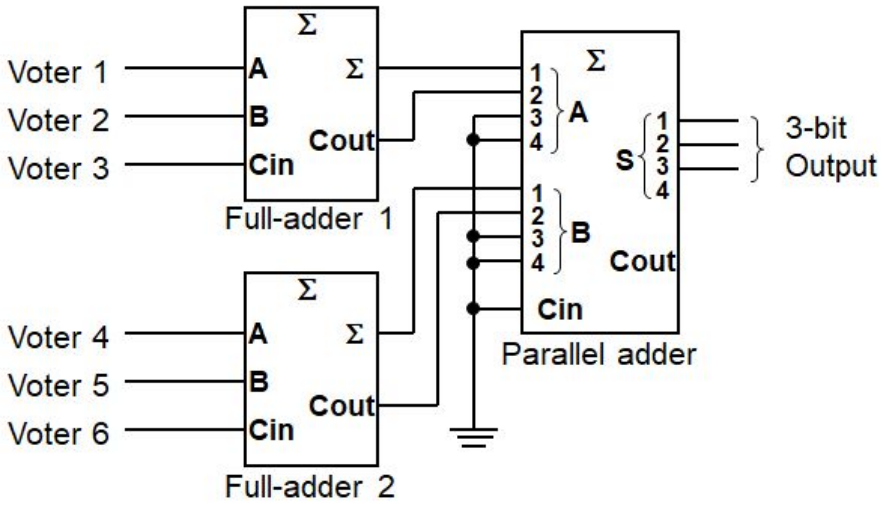
If S=0, then X+Y appears as

the result.

Application: 6-person voting system.

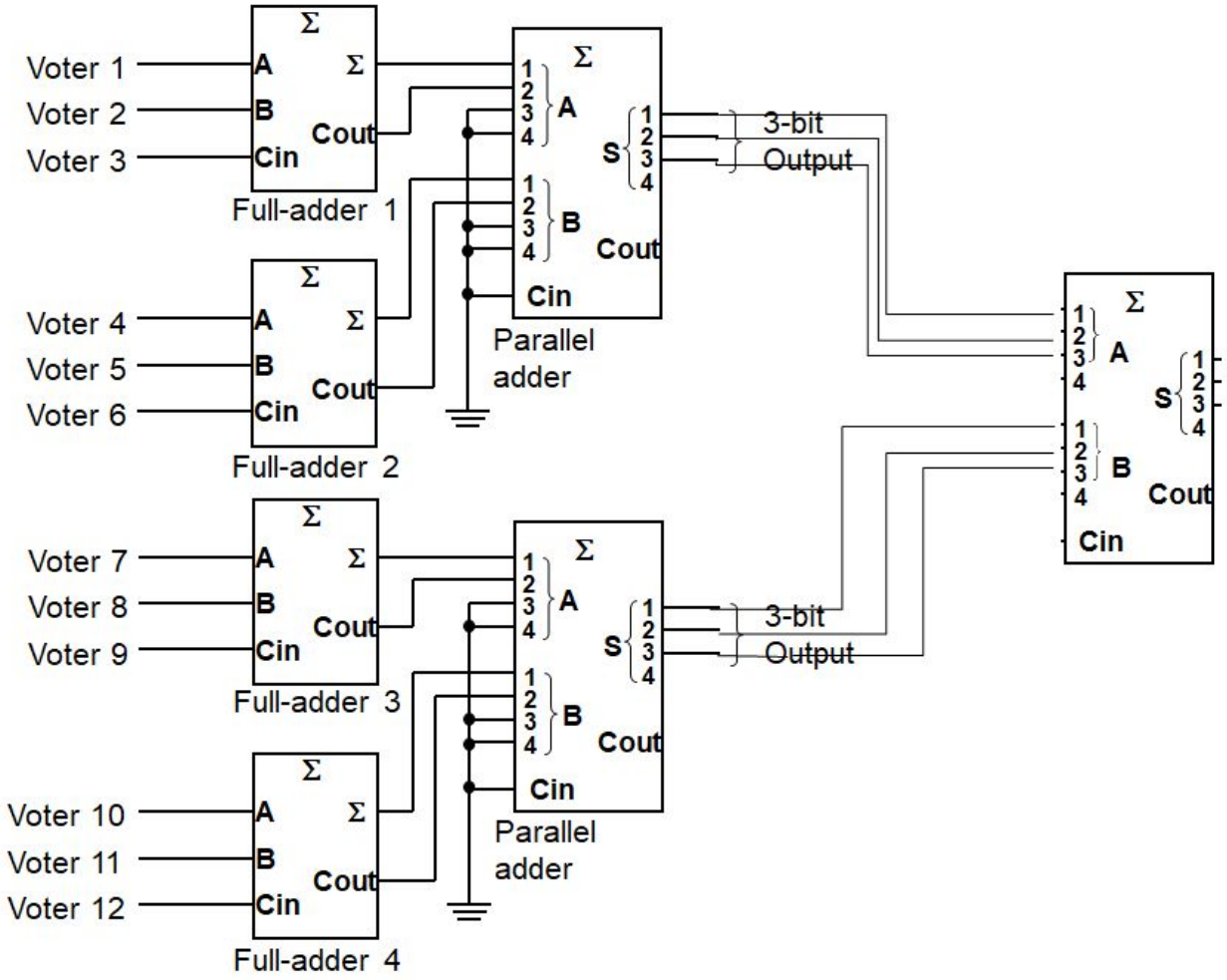
-Use FAs and a 4-bit binary parallel adder.

-Each FA can sum up to 3 votes



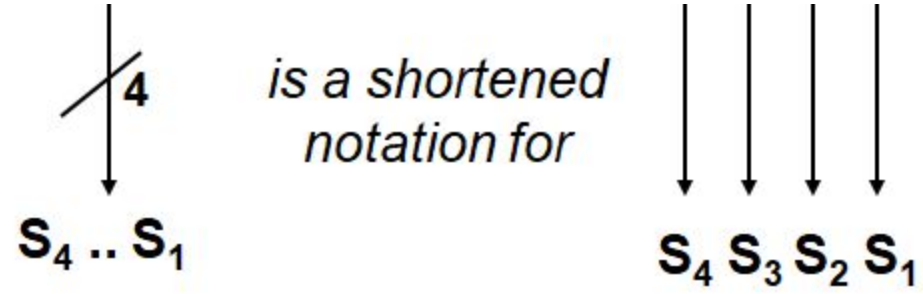
Pin number 4 contains the MSB bit and Pin 1 contains the LSB bit.

Application: 12-person voting system.



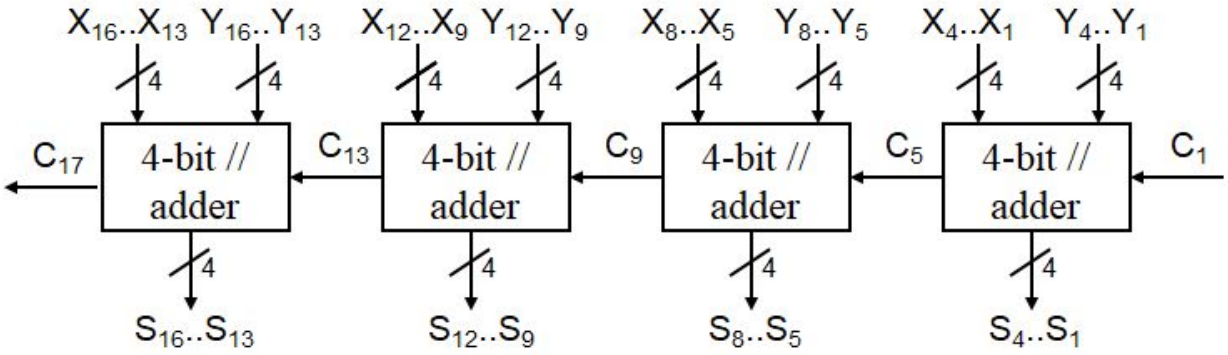
16-bit Parallel Adder

In a figure multiple lines can be shortened in this manner.



Larger parallel adders can be built from smaller ones.

Example: a 16-bit parallel adder can be constructed from four 4-bit parallel adders:



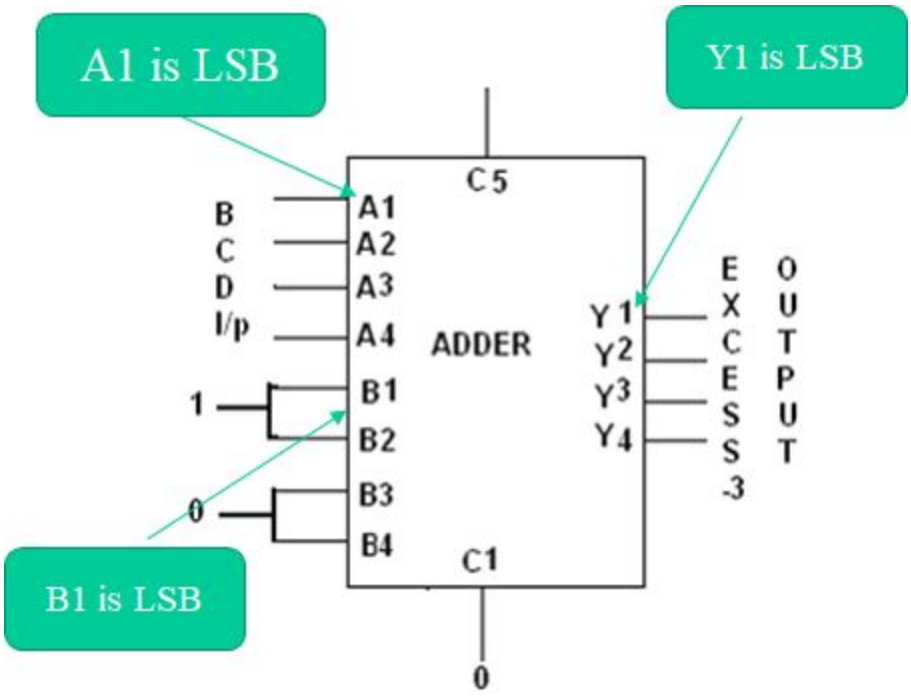
A 16-bit parallel adder

16-bit parallel adder ripples carry from one 4-bit block to the next.

Such ripple-carry circuits are “slow” because of long delays needed to propagate the carries.

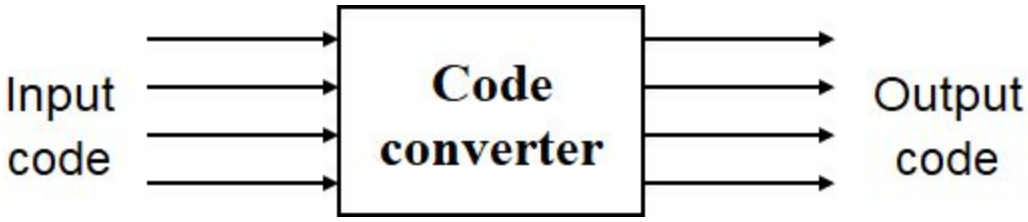
BCD to excess 3 code converter

Inspection of truth table of ‘excess 3 from bcd’ shows that we can get excess-3 code from bcd by adding ‘0011’ to each BCD number. This addition can be implemented by means of 4-bti full adder MSI circuit.



Code Converters

Code converters – take an input code, translate to its equivalent output code.



Example: For a BCD to Excess-3 Code Converter.

Input: BCD digit

Output: Excess-3 digit