

Combinational Logic with MSI and LSI

In this topic, we'll learn:

- Binary Parallel Adder
- Encoders
- Decoders
- Multiplexers
- Demultiplexers

Introduction

- With integrated circuits it is not the count of the number of gates that determines the cost, but
 - The number of external interconnections needed to implement the given function.
- Classical method of previous chapter does not produce the best combinational circuit in many situations.
 - Truth table and simplification method is cumbersome if i/p variables are excessively large.
- Alternate design procedure
 - Depends on the problem
 - Ingenuity of the designer.
- Check if the function is already available in IC package.
 - Formulate a method to incorporate an MSI device in the circuit even if the function cannot be produced.

Binary Parallel adder

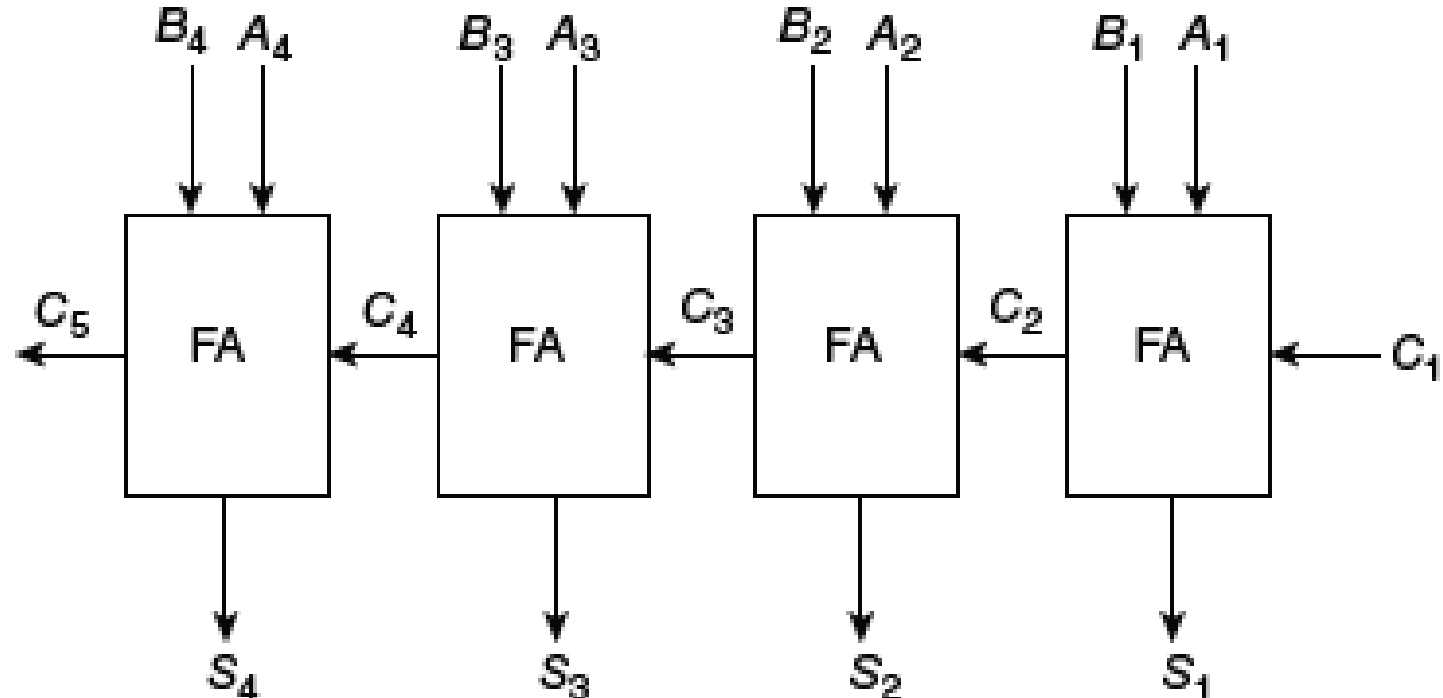
Subscript i	4	3	2	1		Full-adder of Fig. 4-5
Input carry	0	1	1	0	C_i	z
Augend	1	0	1	1	A_i	x
Addend	0	0	1	1	B_i	y
Sum	1	1	1	0	S_i	S
Output carry	0	0	1	1	C_{i+1}	C

A binary parallel adder is a digital circuit that produces the arithmetic sum of two binary numbers in parallel.

It consists of full adders connected in a chain, with the output carry from each full adder connected to the input carry of the next full adder in the chain

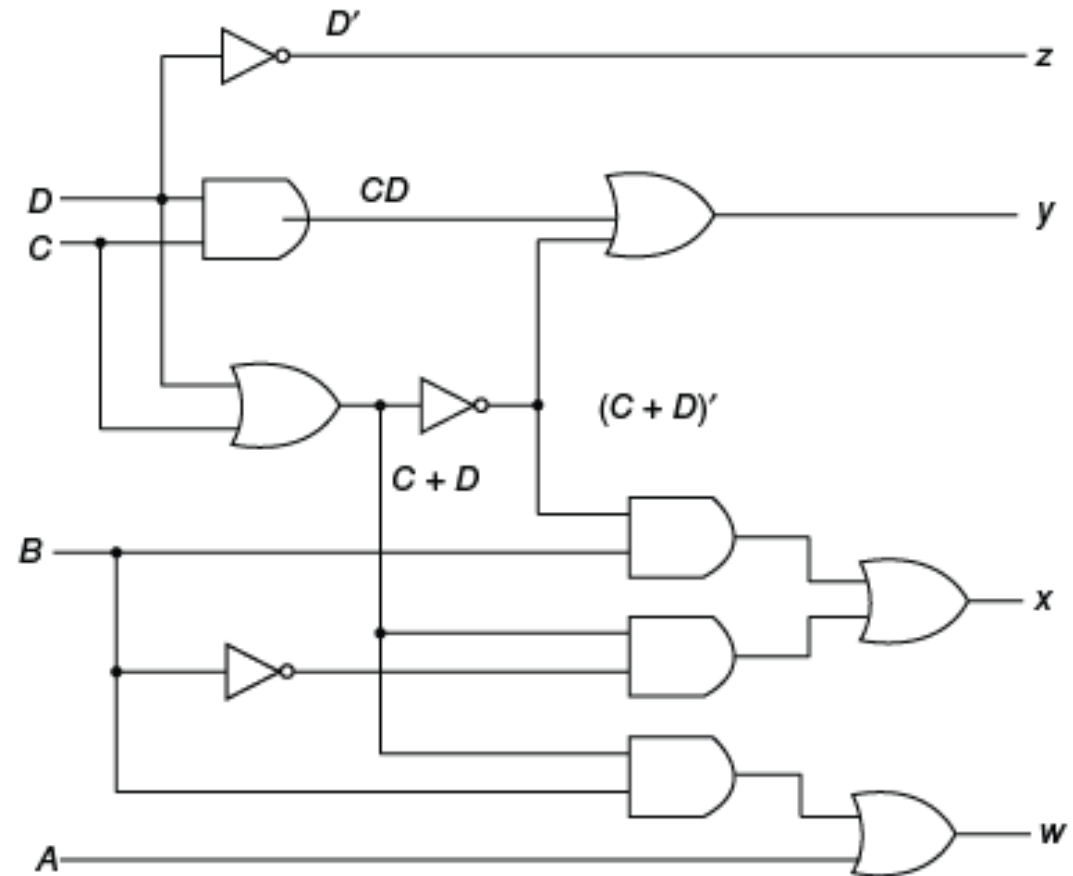
4-bit full-adder (MSI)

- Classical method would require a truth table with $2^9 = 512$ entries.

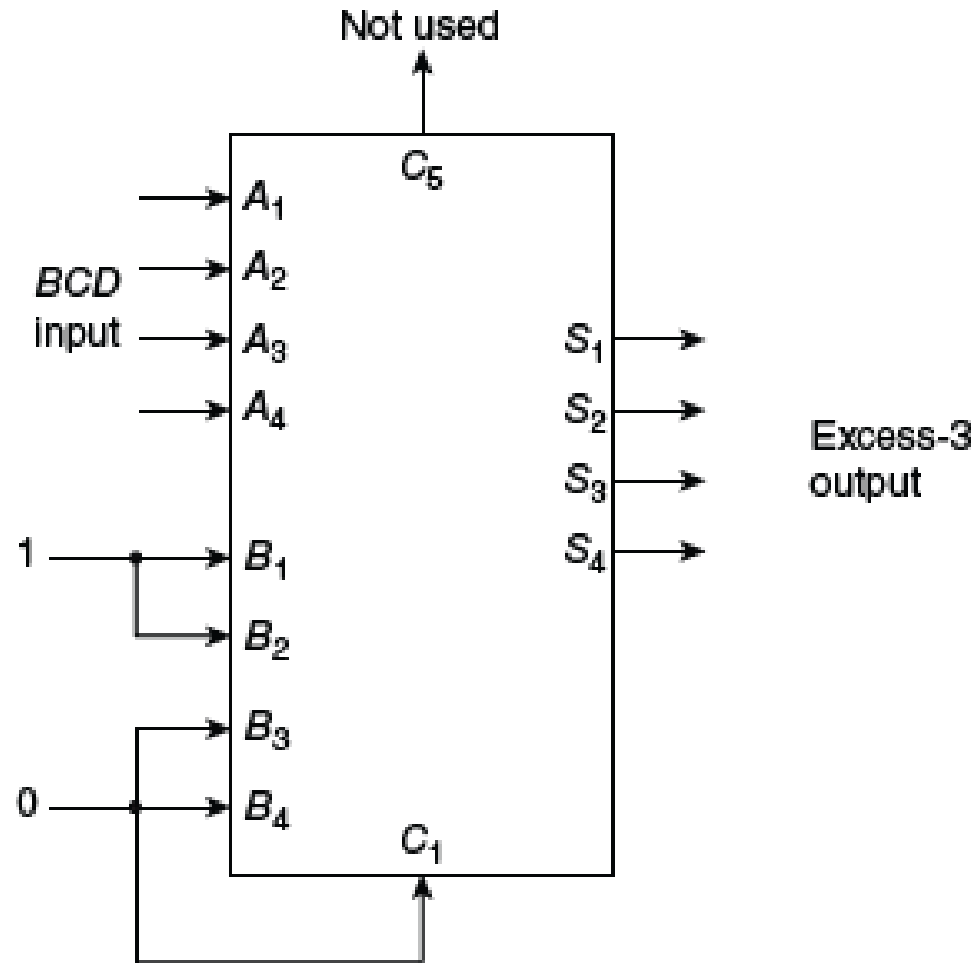


BCD-to-excess-3 code converter

- Classical method



BCD-to-excess-3 code converter



Carry Propagation

- All the bits of the augend and the addend are available for computation at the same time.
- Carry signals must propagate through the gates before the correct output sum is available.
- Total propagation time is equal to the propagation delay of a typical gate times the number of gate levels in the circuit.
 - Longest delay time in parallel adder is for the carry to propagate through the full-adder.

Full-adder

$$P_i = A_i \oplus B_i$$

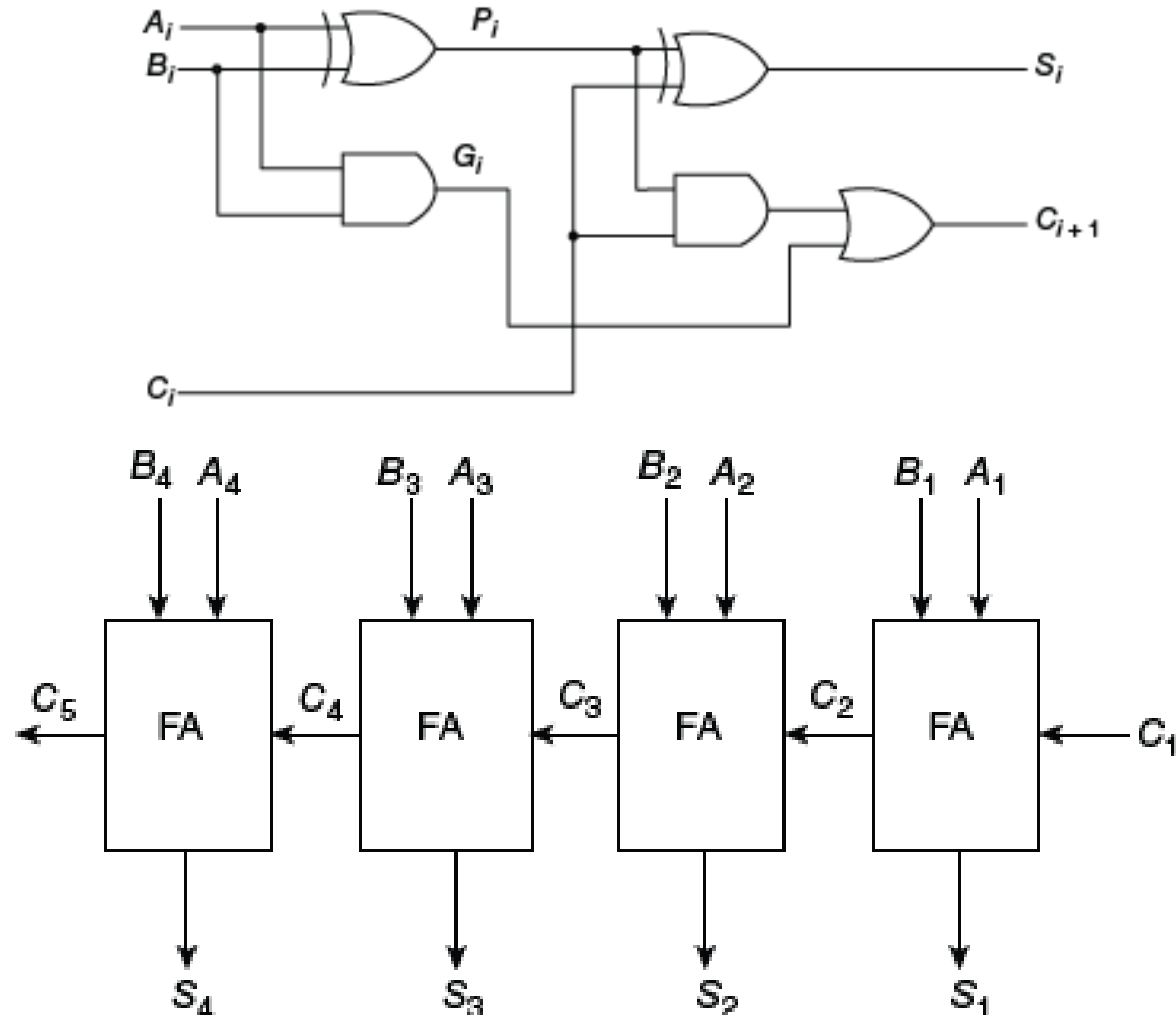
$$G_i = A_i B_i$$

$$S_i = P_i \oplus C_i$$

$$C_{i+1} = G_i + P_i C_i$$

In a FA the carry propagates through 2 gate levels

In a 4bit adder the carry propagates through $2 \times 4 = 8$ gate levels



Look-ahead carry

$$C_2 = G_1 + P_1 C_1$$

$$C_3 = G_2 + P_2 C_2 = G_2 + P_2 (G_1 + P_1 C_1) = G_2 + P_2 G_1 + P_2 P_1 C_1.$$

C_3 does not have to wait for C_2

Carries are generated using two gate levels (one level of AND followed by an OR gate) at each stage.

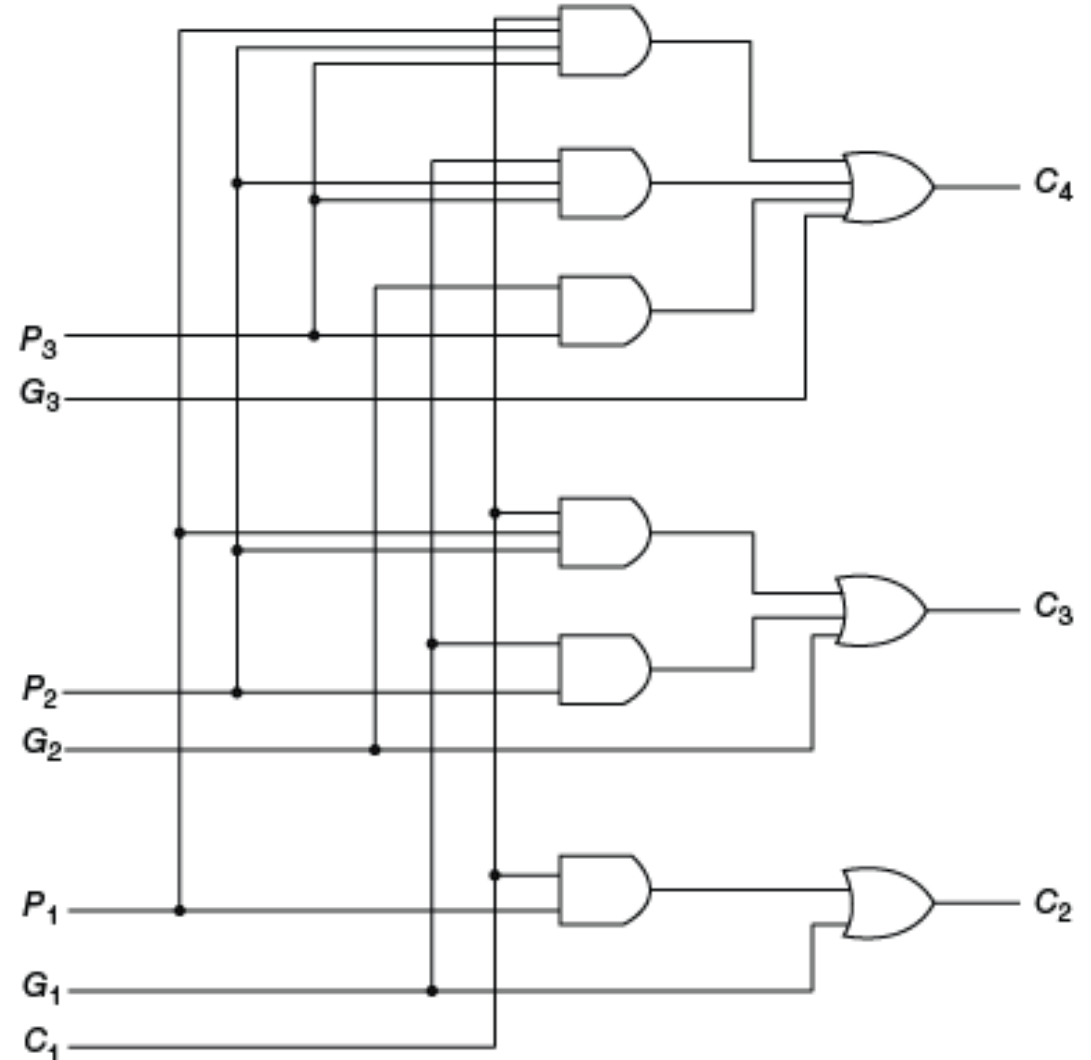
Look-ahead carry generator

$$C_2 = G_1 + P_1 C_1$$

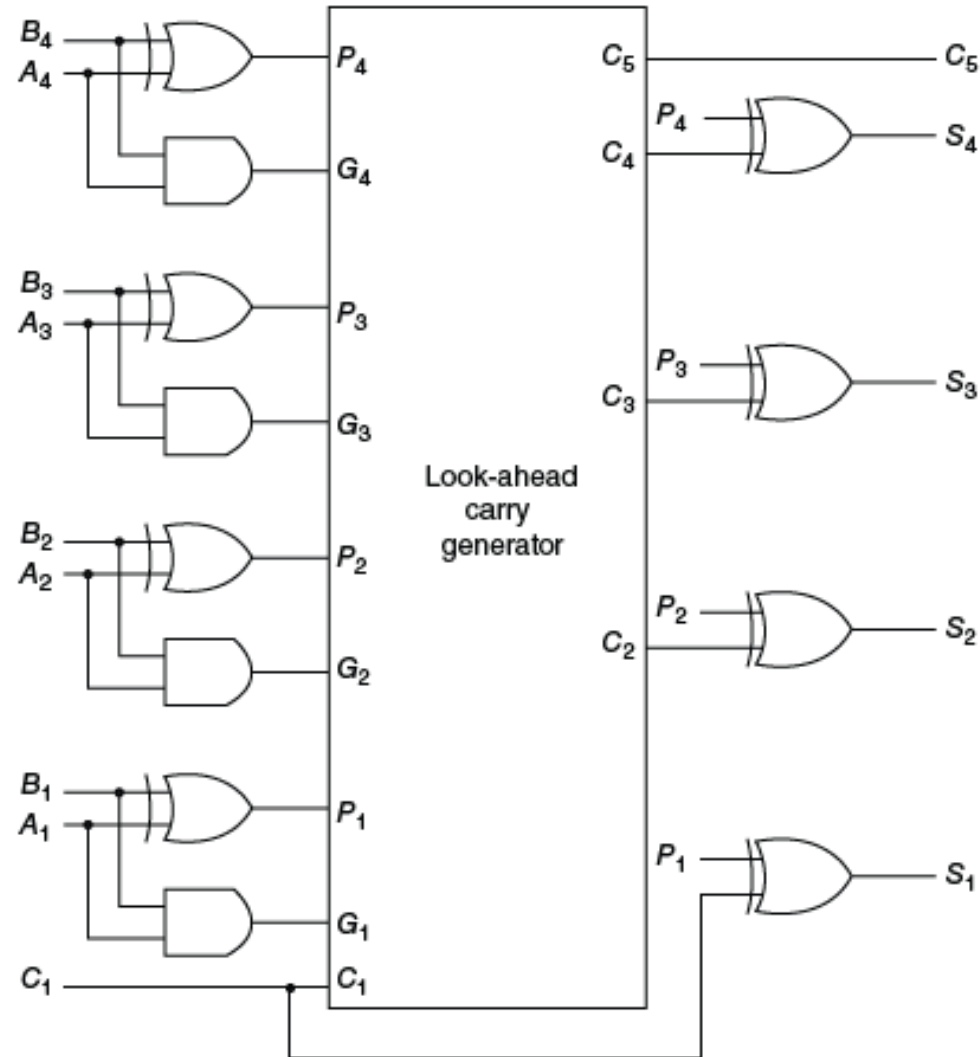
$$C_3 = G_2 + P_2 C_2$$

$$= G_2 + P_2(G_1 + P_1 C_1)$$

$$= G_2 + P_2 G_1 + P_2 P_1 C_1$$



4-bit full-adders with look-ahead carry



Decimal Adder

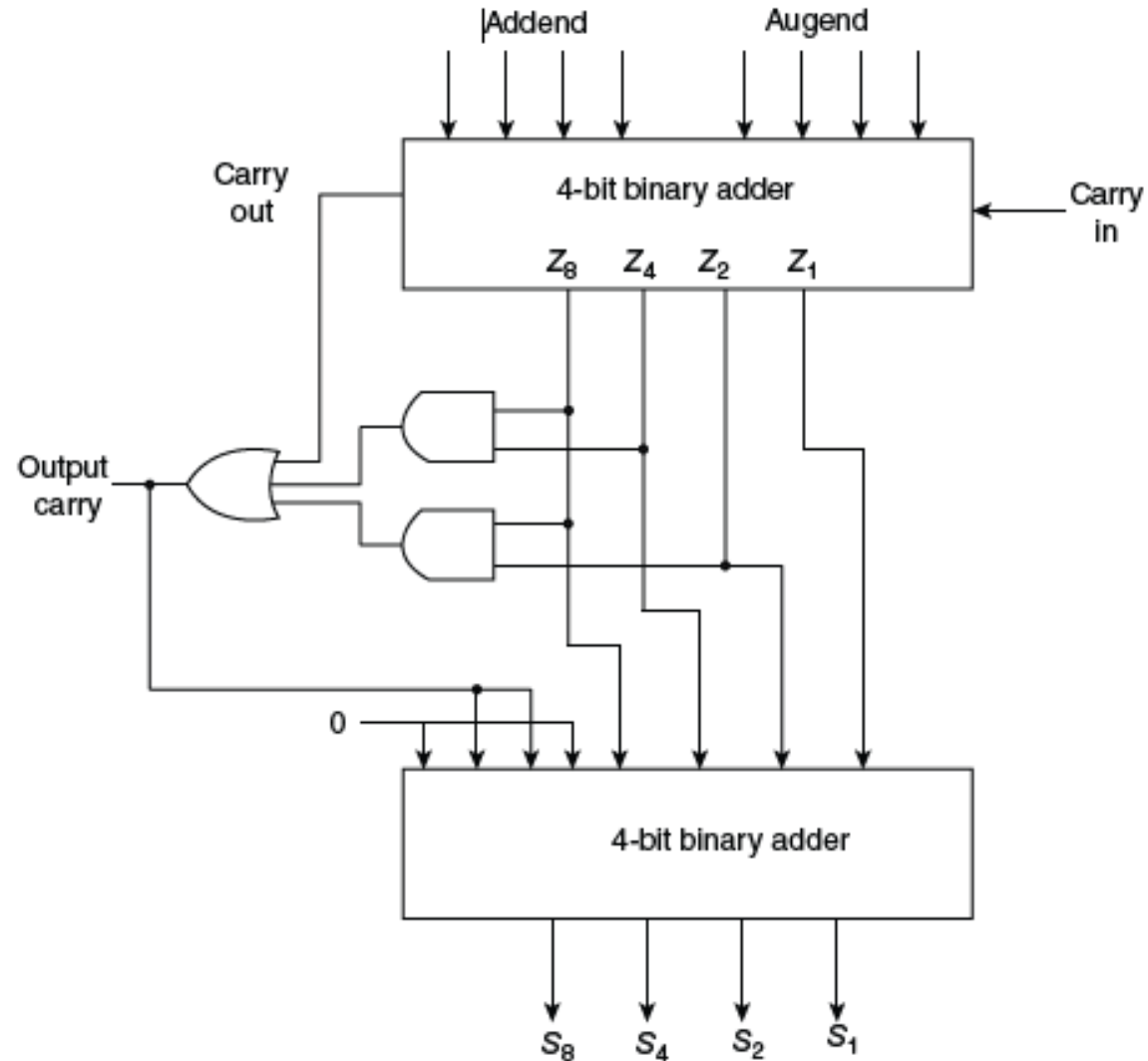
- Applications that perform arithmetic operations directly in decimal number systems
 - Represent decimal numbers in binary-coded form.
- Adder in such cases
 - Accept coded decimal numbers
 - Present results in the accepted code.
 - Requires 9 i/ps and 5 o/ps.
 - Classical method requires a truth table with $2^9 = 512$ entries (many don't care).
 - Alternatively, it can be designed using FA.

BCD adder

- BCD adder refers to a 4-bit binary adder that can add two 4-bit words of BCD format.
- The output of the addition is a BCD-format 4-bit output word, which defines the decimal sum of the addend and augend and a carry that is created in case this sum exceeds a decimal value of 9.
- Therefore, BCD adders can implement decimal addition.

Binary sum					BCD sum					Decimal
K	Z_8	Z_4	Z_2	Z_1	C	S_8	S_4	S_2	S_1	
0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	1	0	0	0	0	1	1
0	0	0	1	0	0	0	0	1	0	2
0	0	0	1	1	0	0	0	1	1	3
0	0	1	0	0	0	0	1	0	0	4
0	0	1	0	1	0	0	1	0	1	5
0	0	1	1	0	0	0	1	1	0	6
0	0	1	1	1	0	0	1	1	1	7
0	1	0	0	0	0	1	0	0	0	8
0	1	0	0	1	0	1	0	0	1	9
0	1	0	1	0	1	0	0	0	0	10
0	1	0	1	1	1	0	0	0	1	11
0	1	1	0	0	1	0	0	1	0	12
0	1	1	0	1	1	0	0	1	1	13
0	1	1	1	0	1	0	1	0	0	14
0	1	1	1	1	1	0	1	0	1	15
1	0	0	0	0	1	0	1	1	0	16
1	0	0	0	1	1	0	1	1	1	17
1	0	0	1	0	1	1	0	0	0	18
1	0	0	1	1	1	1	0	0	1	19

Block Diagram of a BCD adder



Magnitude comparator

- Comparing two n -bit numbers has 2^{2n} entries in the truth table.
- Regularity present is used to design by algorithmic procedure.
 - Finite set of steps are followed to obtain the solution to a problem.
- Design a 2-bit magnitude comparator.

Inputs				Outputs		
A_1	A_0	B_1	B_0	$A > B$	$A = B$	$A < B$
0	0	0	0	0	1	0
0	0	0	1	0	0	1
0	0	1	0	0	0	1
0	0	1	1	0	0	1
0	1	0	0	1	0	0
0	1	0	1	0	1	0
0	1	1	0	0	0	1
0	1	1	1	0	0	1
1	0	0	0	1	0	0
1	0	0	1	1	0	0
1	0	1	0	0	1	0
1	0	1	1	0	0	1
1	1	0	0	1	0	0
1	1	0	1	1	0	0
1	1	1	0	1	0	0
1	1	1	1	0	1	0

- **Expression for $A < B$**

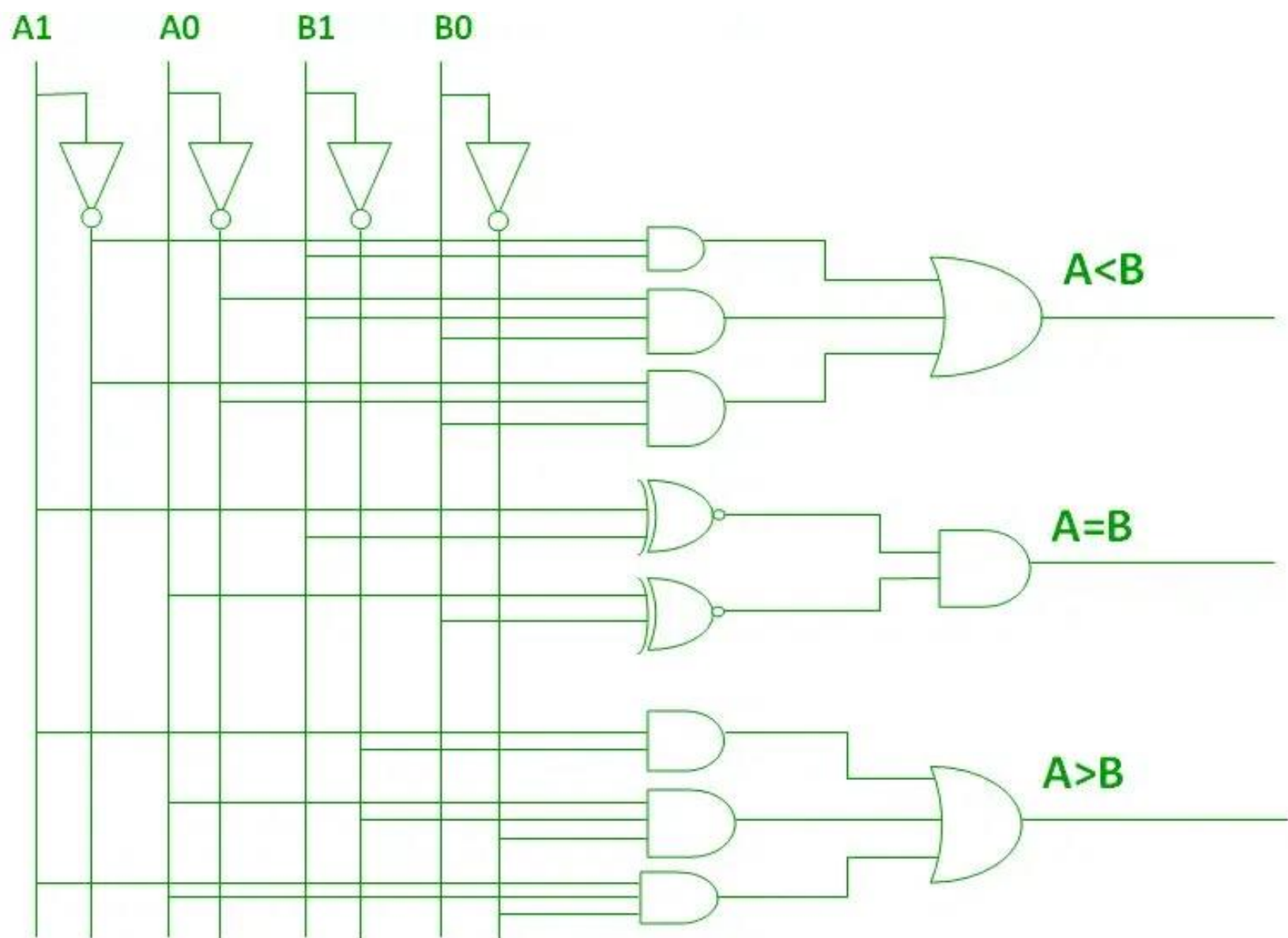
$$Y = A_1' B_1 + A_1' A_0' B_0 + A_0' B_1 B_0$$

- **Expression for $A = B$**

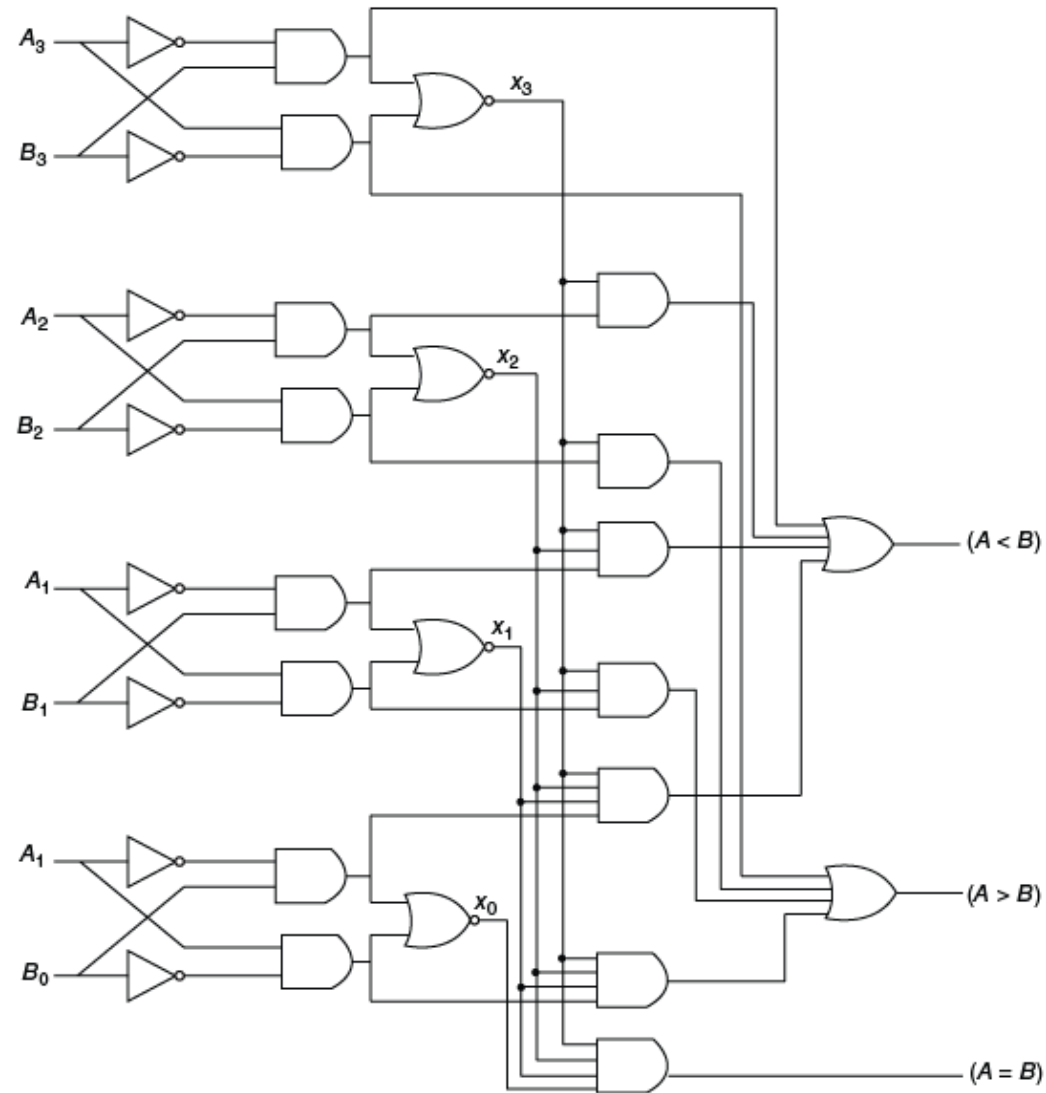
$$Y = A_1' A_0' B_1' B_0' + A_1' A_0 B_1' B_0 + A_1 A_0 B_1 B_0 + A_1 A_0' B_1 B_0'$$

- **Expression for $A > B$**

$$Y = A_1 B_1' + A_0 B_1' B_0' + A_1 A_0 B_0'$$



4-bit comparator



Decoder

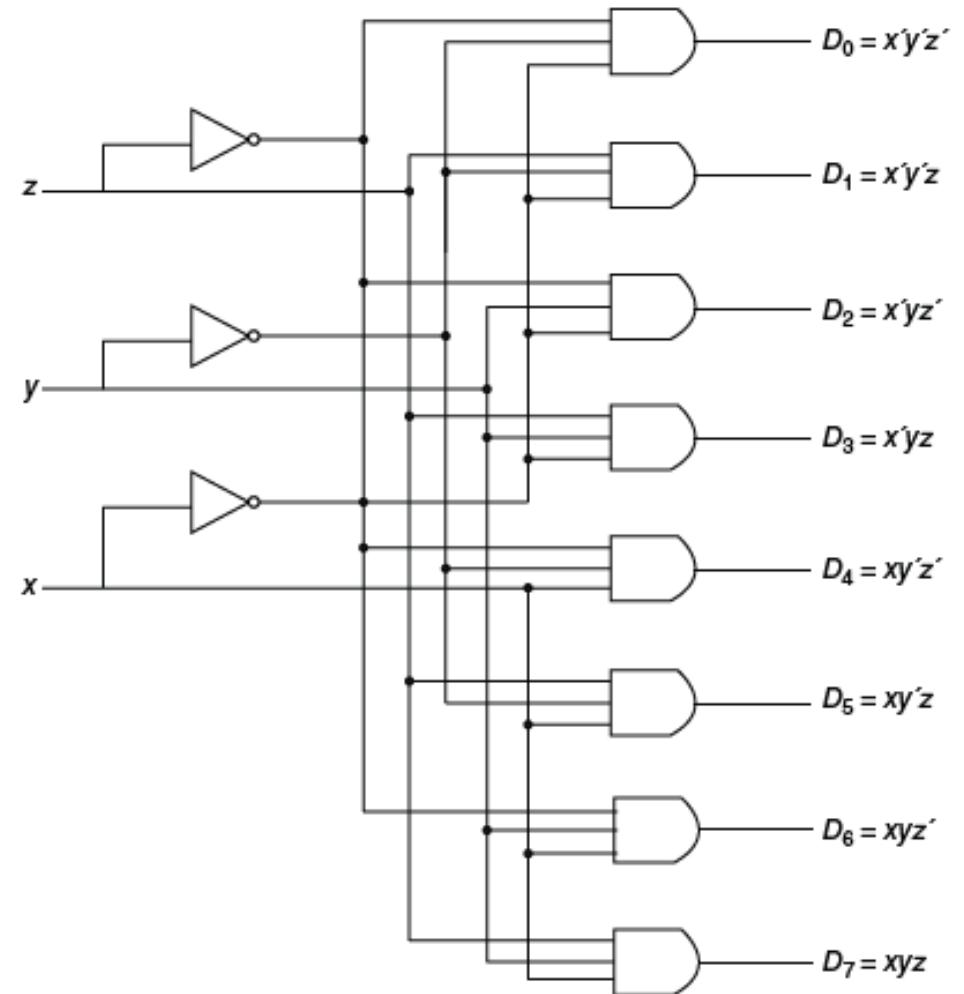
- Combinational circuit
 - that converts binary information
 - n i/p lines
 - maximum of 2^n o/p lines.
 - Decoder covered here are called n-to-m line decoders where $m \leq 2^n$.
 - Produces m minterms of n i/p variables.

3-to-8 line decoder

- Binary-octal conversion
 - i/ps represent a binary number.
 - o/ps represent the eight digits in the octal number system.

3-to-8 line decoder

- Binary-octal conversion
 - i/ps represent a binary number.
 - o/ps represent the eight digits in the octal number system.



Truth table of a 3-to-8 line decoder

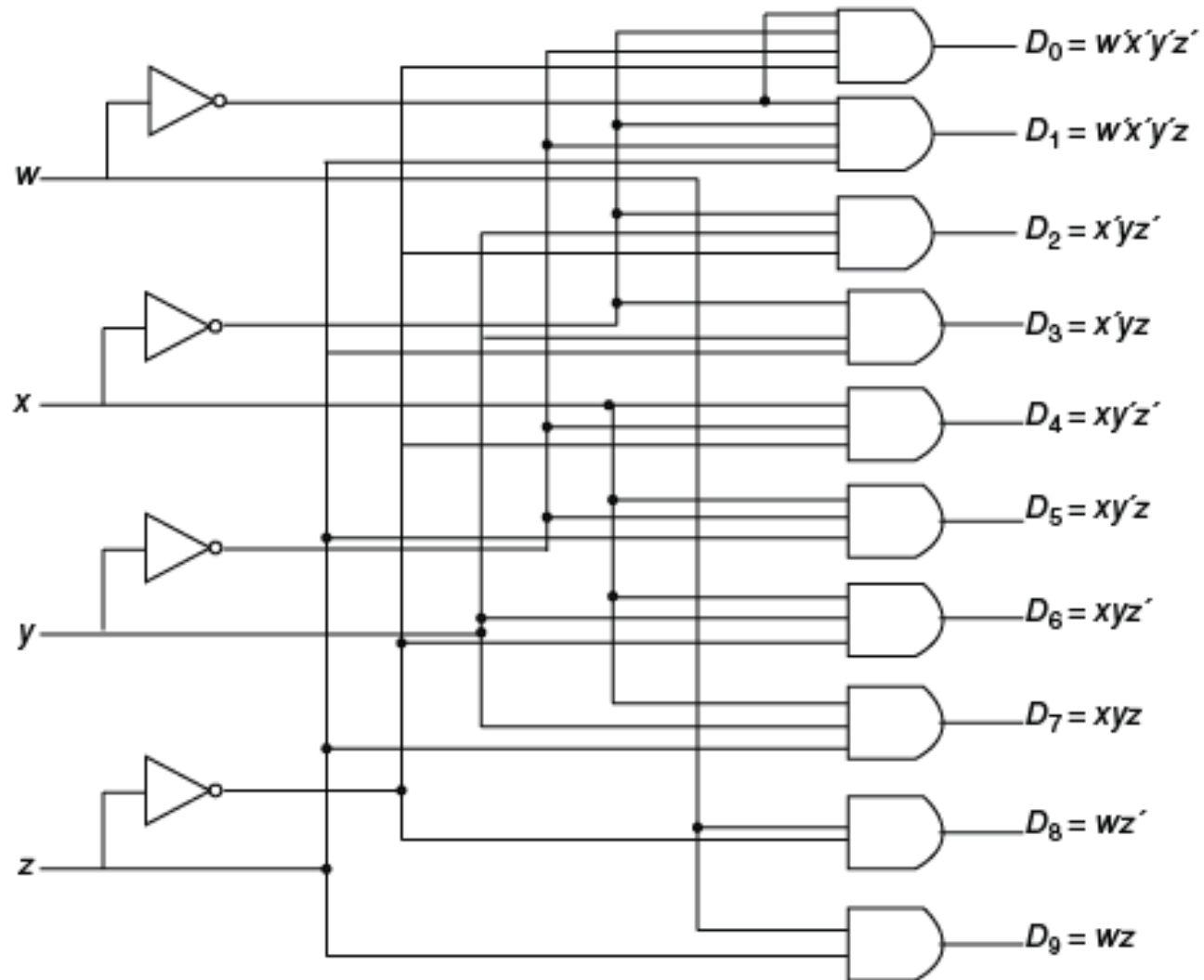
Inputs			Outputs							
x	y	z	D_0	D_1	D_2	D_3	D_4	D_5	D_6	D_7
0	0	0	1	0	0	0	0	0	0	0
0	0	1	0	1	0	0	0	0	0	0
0	1	0	0	0	1	0	0	0	0	0
0	1	1	0	0	0	1	0	0	0	0
1	0	0	0	0	0	0	1	0	0	0
1	0	1	0	0	0	0	0	1	0	0
1	1	0	0	0	0	0	0	0	1	0
1	1	1	0	0	0	0	0	0	0	1

BCD to decimal decoder

The input will be 4-line as the code has 4 bits.

The output will be 10 line, one for each decimal digit

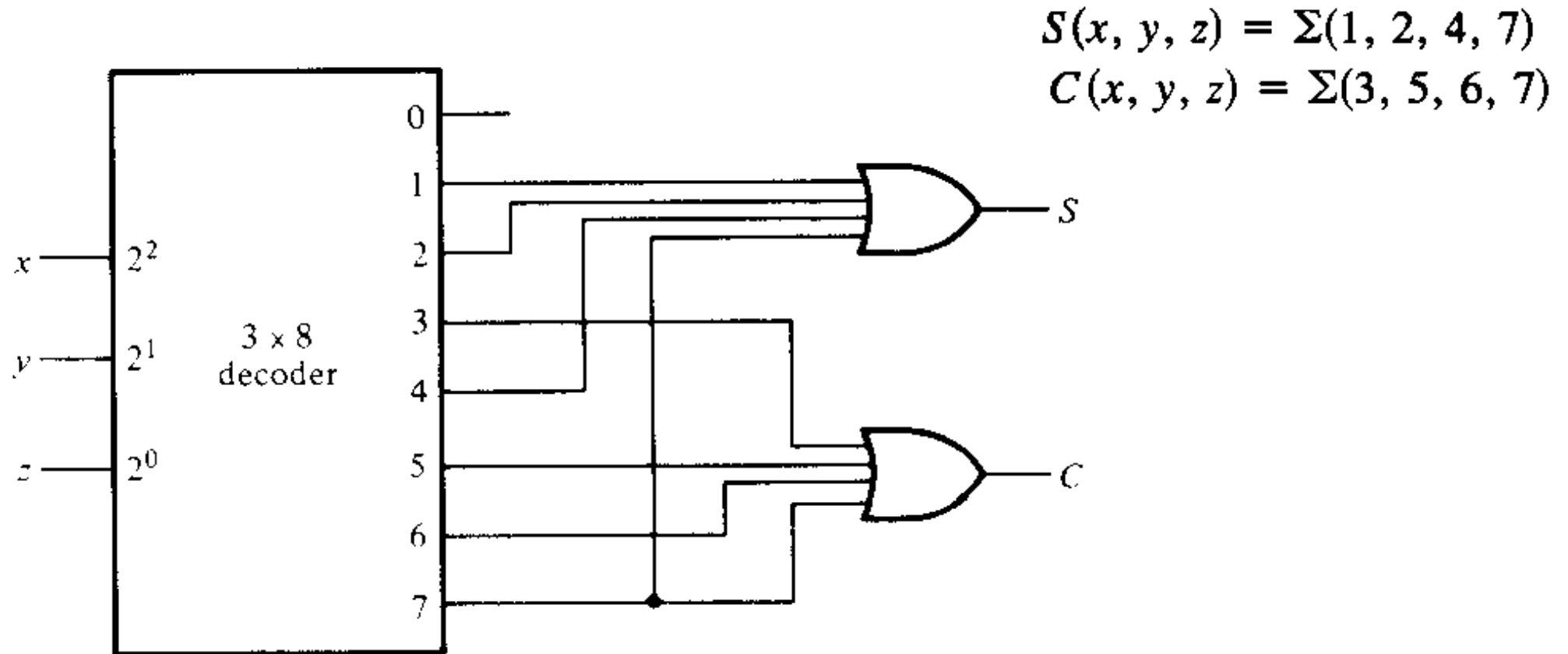
BCD to decimal decoder



Combinational Logic Implementation using decoders

- Decoder provides the 2^n minterm of n input variables.
 - any Boolean function can be expressed in sum of minterms canonical form.

Implement a full-adder circuit with a decoder and two OR gates.

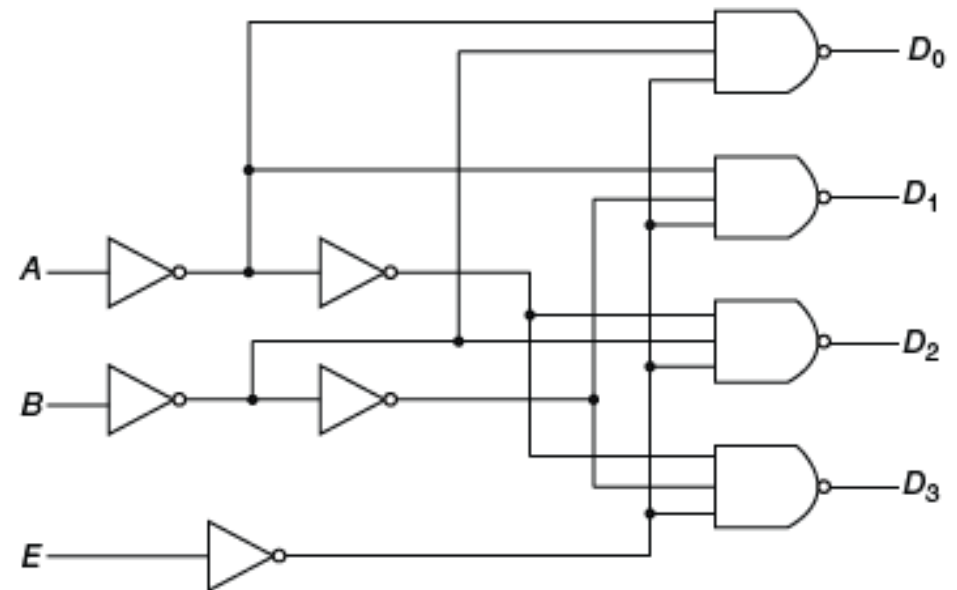


Decoder implementation

- Decoder gives the best implementation of the combinational circuit
 - If the combinational circuit has many i/ps.
 - If each o/p is expressed with small number of minterms.

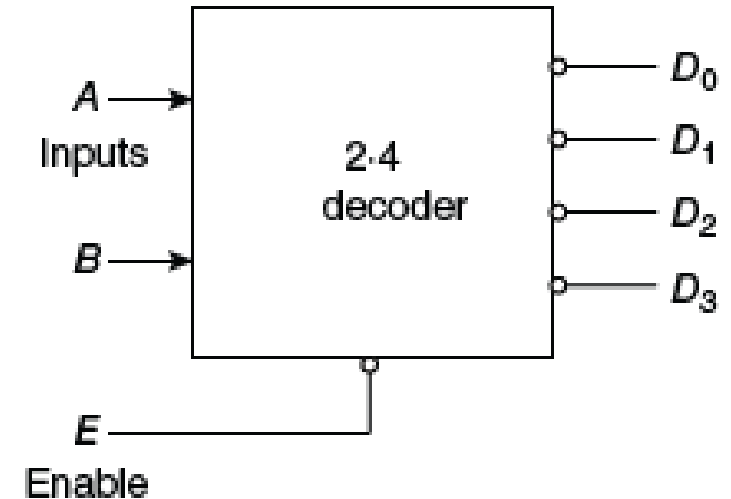
Decoder with enable i/p.

- IC decoders are constructed with NAND gates
 - economical if decoder minterms are generated in the complement form.
- Most IC decoders include one or more enable i/ps to control the circuit operation.
- Truth Table



Block diagram

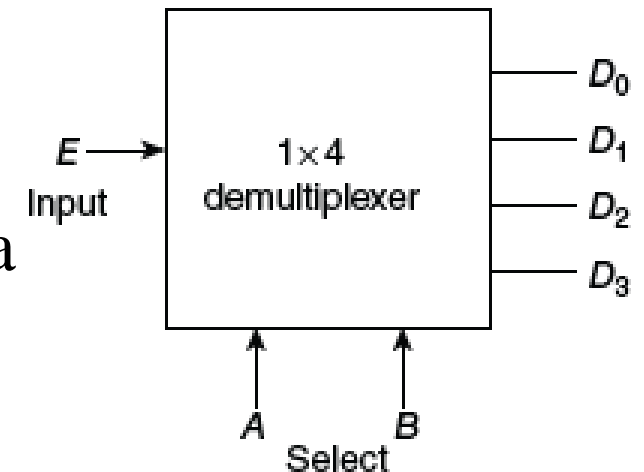
- Decoder is enabled when $E = 0$.
- small circles at the outputs indicate that
 - All outputs are complemented



Demultiplexers

- Decoder with an enable input can function as a demultiplexer.
 - receives information on a single line
 - transmits this information on one of 2 possible output lines.
- Selection of a specific o/p line is controlled by the n selection lines.
- Decoder works as demultiplexers
 - If E is taken as i/p.
 - I/p lines A, B are used as selection lines.
- Decoder with an enable input is referred to as a *decoder/demultiplexer*

E	A	B	D ₀	D ₁	D ₂	D ₃
1	X	X	1	1	1	1
0	0	0	0	1	1	1
0	0	1	1	0	1	1
0	1	0	1	1	0	1
0	1	1	1	1	1	0



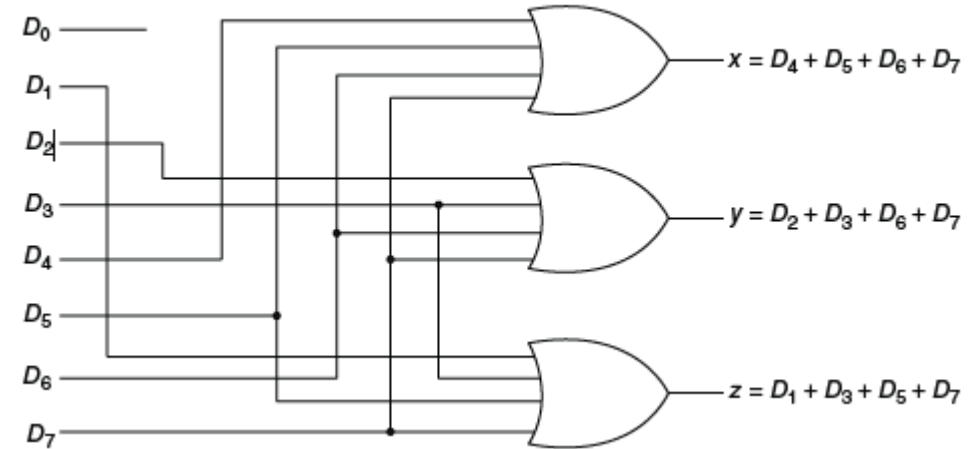
4×16 decoder using two 3×8 decoders with
enable inputs

Encoders

- Encoder: Digital function that produces a reverse operation from that of a decoder.
 - 2^n (or less) input lines and n output lines.
 - output lines generate the binary code for the 2^n i/p variables.
- Octal to binary encoder.

Octal-to-binary encoder

- Discrepancy when $D_0=1$ and when all i/ps are 0.
 - Can be resolved by using one more o/p to indicate all i/ps are not 0's.
- Assumes that only one input line can be equal to 1 at any time
 - Out of $2^8=256$, only 8 combinations have meaning.
 - Others are don't-care conditions.



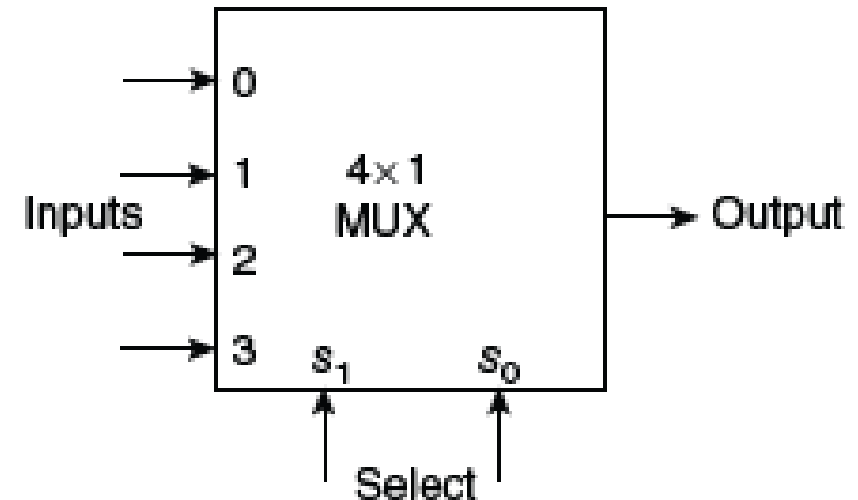
Inputs								Outputs		
D_0	D_1	D_2	D_3	D_4	D_5	D_6	D_7	x	y	z
1	0	0	0	0	0	0	0	0	0	0
0	1	0	0	0	0	0	0	0	0	1
0	0	1	0	0	0	0	0	0	1	0
0	0	0	1	0	0	0	0	0	1	1
0	0	0	0	1	0	0	0	1	0	0
0	0	0	0	0	1	0	0	1	0	1
0	0	0	0	0	0	1	0	1	1	0
0	0	0	0	0	0	0	1	1	1	1

Priority Encoders

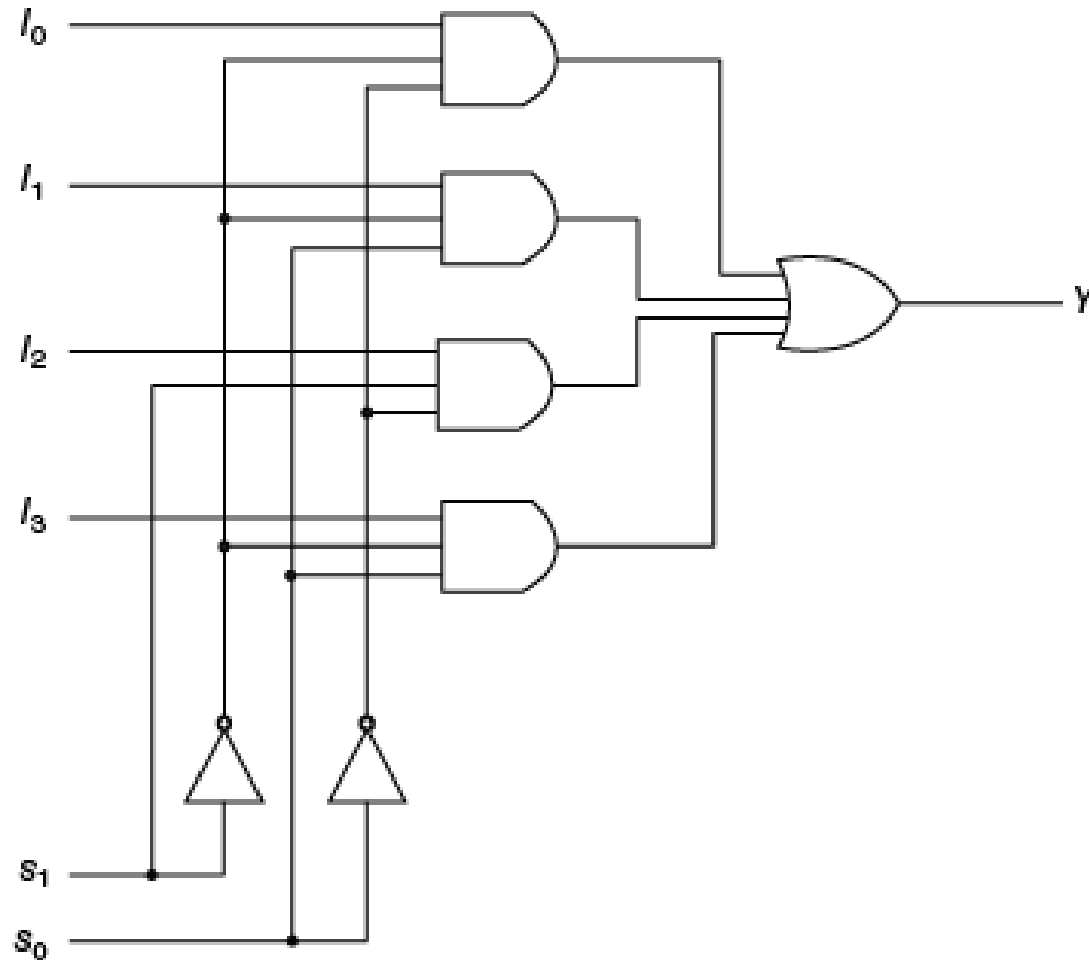
- Type of encoders available in IC are priority encoders.
- Only highest priority i/p line is encoded
 - if both $D2$ and $D5$ are logic-1 simultaneously then o/p is 101.
- Truth table of priority encoder is different.

Multiplexers (Data Selector)

- *Digital multiplexer* is a combinational circuit
 - that selects binary information from one of many input lines
 - directs it to a single output line
 - Selection lines.
- Normally 2^n i/p lines and n selection lines are present.
- **Truth table and Block diagram**



Truth Table and Block diagram



s_1	s_0	Y
0	0	I_0
0	1	I_1
1	0	I_2
1	1	I_3

Multiplexers using decoders

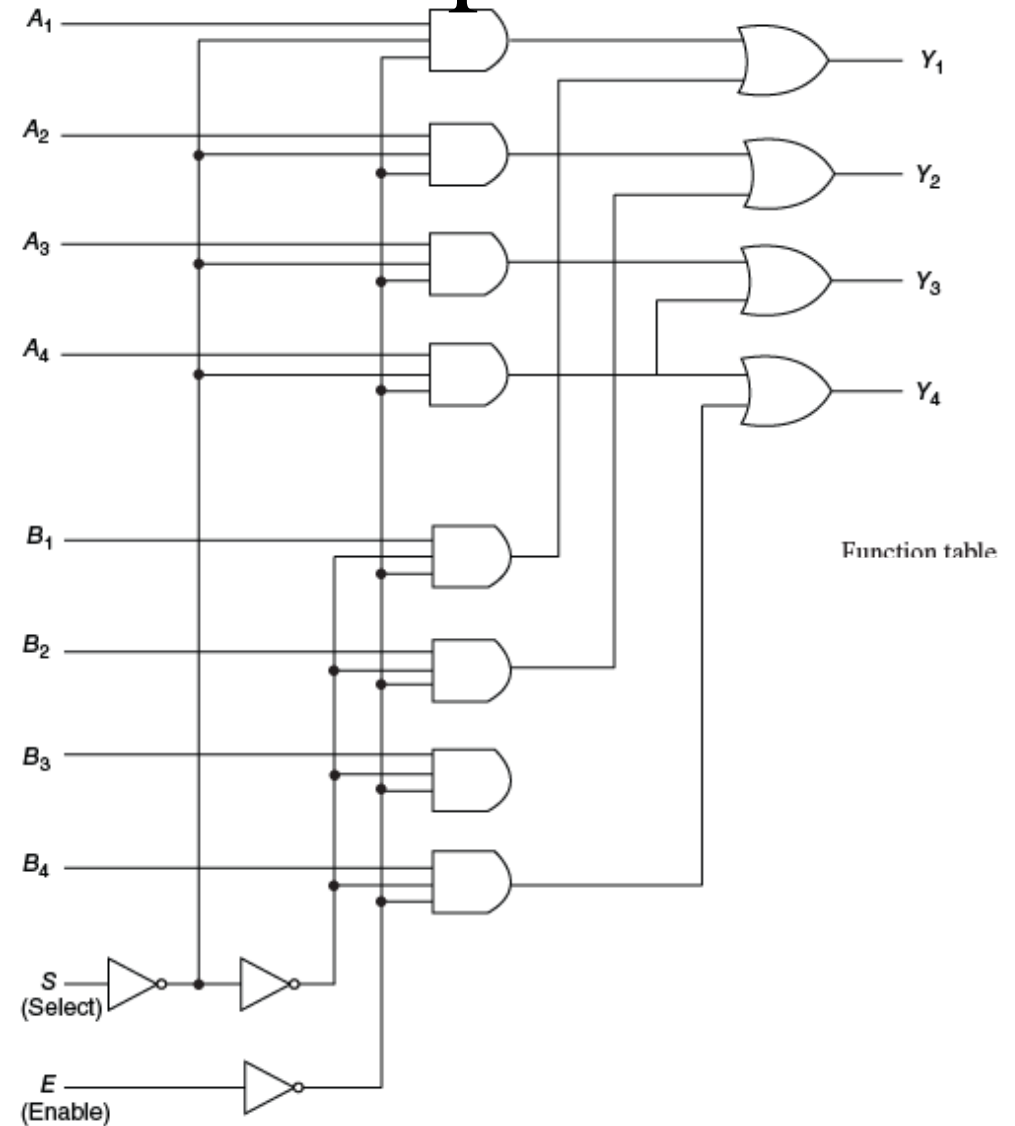
- 2^n -to-1 line multiplexer is constructed from an n -to- 2^n decoder
 - by adding an extra i/p line to each AND gate.
- o/ps of AND gates are applied to an OR gate to provide the o/p.
- Size of the Multiplexer (MUX) is specified by the number of i/p (2^n) lines and a single o/p line.

Multiplexer ICs with *enable* input

- Used to expand two or more multiplexer ICs to a digital multiplexer with a larger number of inputs.

Quadruple 2-to-1 line multiplexers

- In some cases two or more multiplexers are enclosed within one IC package
- Four multiplexers
 - each capable of selecting one of two input lines.



Applications of Multiplexer

- Very useful MSI function.
- Connecting two or more sources to a single destination among computer units
- Useful for constructing a common bus system.
- Used to implement any Boolean function.

Boolean Function Implementation

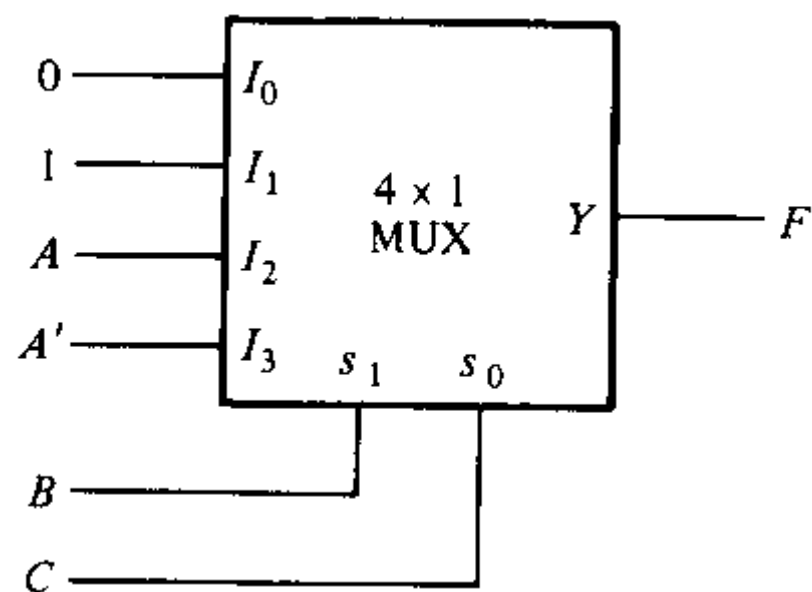
- Decoder can be used to implement a Boolean function by employing an external OR gate.
- A MUX is decoder followed by an OR gate.
- Minterms to be included in the function are made 1.

Boolean function of $n + 1$ using n selection lines

- n variables are used for selection lines
- Remaining variable is used as the i/p.
 - If A is this variable the i/ps to the multiplexer is either A or, A' or, 1 or 0.
- Implement $F(A, B, C) = \Sigma(1,3,5,6)$ with a 4-to-1 multiplexer.

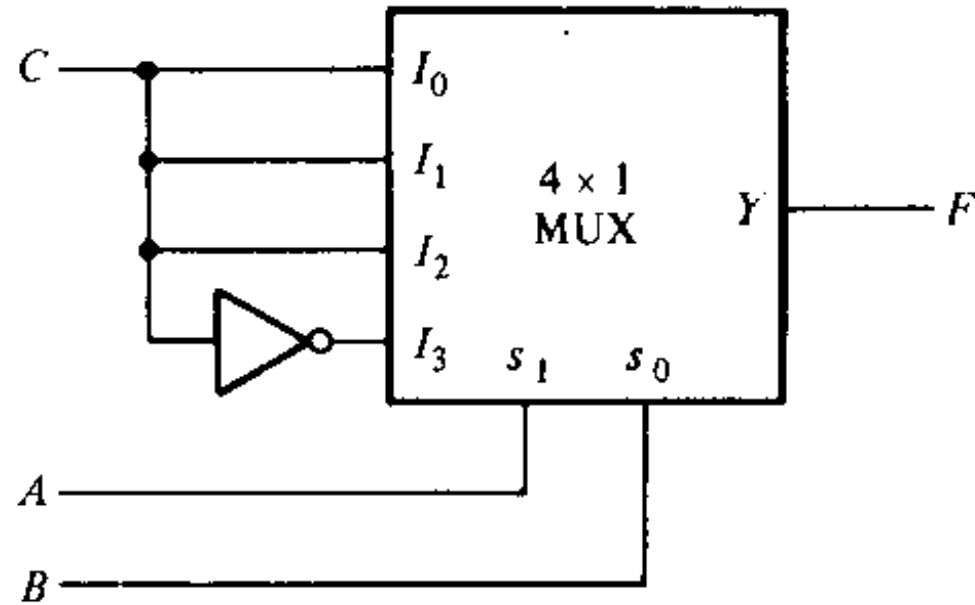
Minterm	A	B	C	F
0	0	0	0	0
1	0	0	1	1
2	0	1	0	0
3	0	1	1	1
4	1	0	0	0
5	1	0	1	1
6	1	1	0	1
7	1	1	1	0

	I_0	I_1	I_2	I_3
A'	0	①	2	③
A	4	⑤	⑥	7
	0	1	A	A'



Alternate implementation

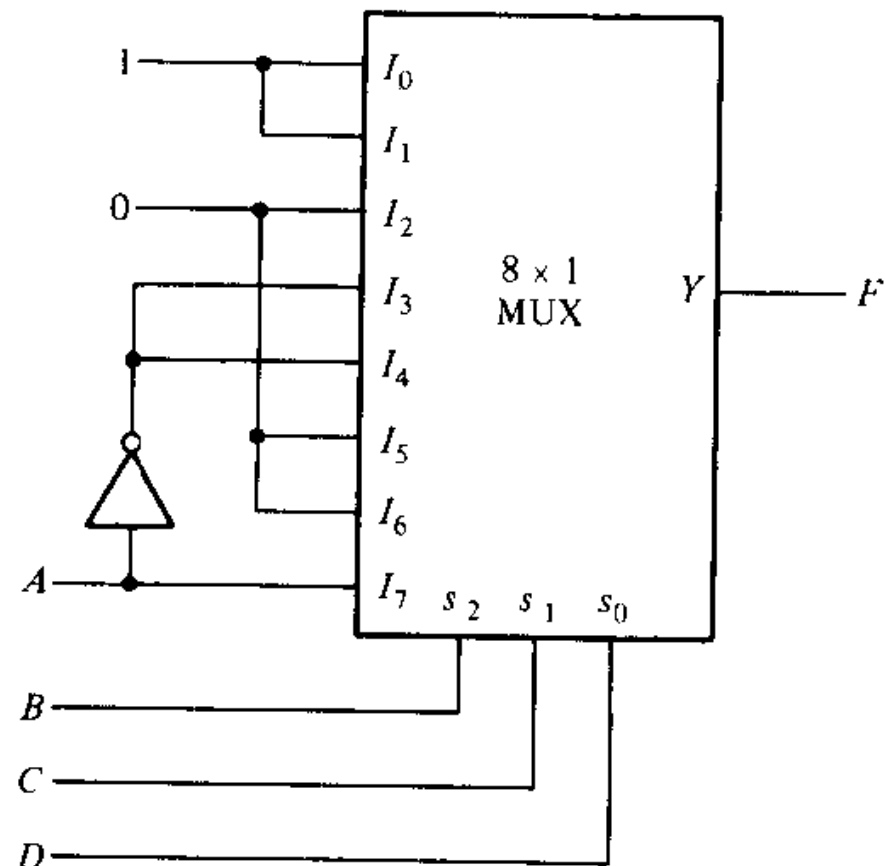
	I_0	I_1	I_2	I_3
C'	0	2	4	⑥
C	①	③	⑤	7
	C	C	C	C'



Implement the following function with a
multiplexer:

$$F(A, B, C, D) = \Sigma(0, 1, 3, 4, 8, 9, 15)$$

	I_0	I_1	I_2	I_3	I_4	I_5	I_6	I_7
A'	①	②	3	④	⑤	6	7	
A	⑧	⑨	10	11	12	13	14	⑮
	1	1	0	A'	A'	0	0	A



Multiplexer vs Decoder

- One multiplexer for each output function.
 - combinational circuits with a small number of outputs should be implemented
- Decoder method requires an OR gate for each output function
 - one decoder is needed to generate all minterms.
 - Combinational circuits with many output functions would probably use fewer Ics
- However,
 - decoders are mostly used for decoding binary information.
 - multiplexers are mostly used to form a selected path between multiple sources and a single destination.
 - considered when designing small, special combinational circuits.