Combinational Logic Design

Instructor: Saraju P Mohanty

(Part 1)

- Positive and Negative Logic
- Transmission Gates
- Combinational Circuits
- Analysis of Combinational Circuits
- Design Procedure for Combinational Circuits
- Decoders
- Encoders

Positive Vs Negative Logic

Positive Logic : High Level represent logic 1

Negative Logic: Low Level represent logic 1

Signal value	Logic value
H	1
L	0

(a) Positive logic

Signal value	Logic value
H	0
L	1

(b) Negative logic

Fig. 2-42 Signal Assignment and Logic Polarity

Positive Vs Negative Logic (Formal ….)

- Positive Logic: The signal that activates the circuit (the 1 state) has a voltage level that is more POSITIVE than the 0 state.
- Negative logic: The signal that activates the circuit (the 1 state) has a voltage level that is more NEGATIVE than the 0 state

EXAMPLE 1	Active signal — TRUE, 1, HIGH	= +10 volts
	Complement — FALSE, 0, LOW	= 0 volts
EXAMPLE 2	Active signal — TRUE, 1, HIGH	= 0 volts
	Complement — FALSE, 0, LOW	= −10 volts

(Examples of positive logic)

Positive Vs Negative Logic …..

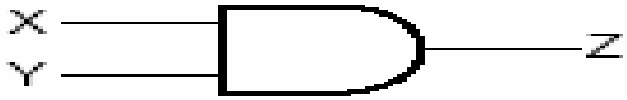
X	Y	Z
L	L	L
L	H	L
H	L	L
H	H	H

(a) Truth table with H and L

(b) Gate block diagram

X	Y	Z
0	0	0
0	1	0
1	0	0
1	1	1

(c) Truth table for positive logic



(d) Positive-logic AND gate

X	Y	Z
1	1	1
1	0	1
0	1	1
0	0	0

(e) Truth table for negative logic

(f) Negative-logic OR gate

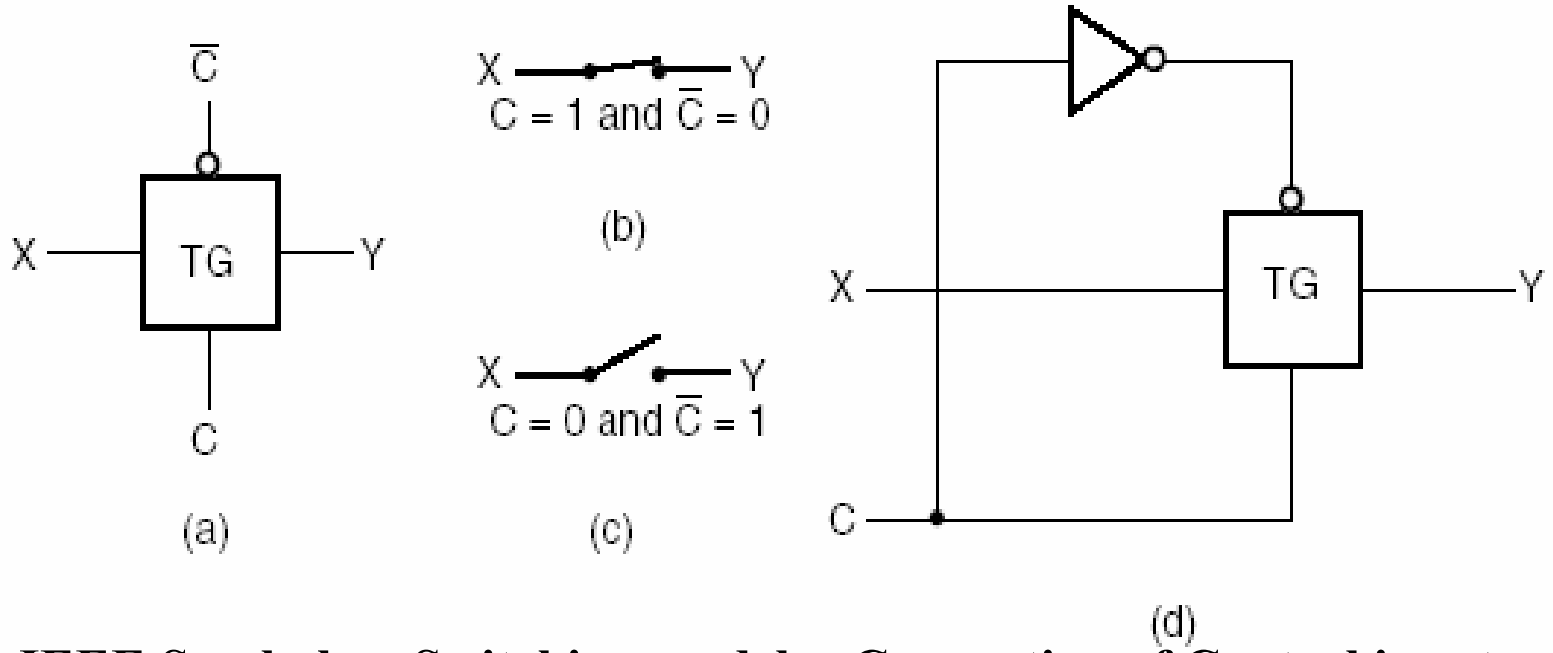
Fig. 2-43 Demonstration of Positive and Negative Logic

Positive Vs Negative Logic : Observations

- Small triangles on the inputs and outputs are polarity indicators.
- Presence of polarity indicator implies negative logic.
- Same physical gate can operate as positive AND gate or negative OR gate.
- The actual signal values (such as 5.0V) do not determine the type of logic, rather the relative amplitude of two signals.
- Conversion of positive to negative logic or vice versa: change 1's to 0's and 0's to 1's at inputs and outputs.

Transmission Gates

An electronic switch for connecting and disconnecting two points in a circuit.



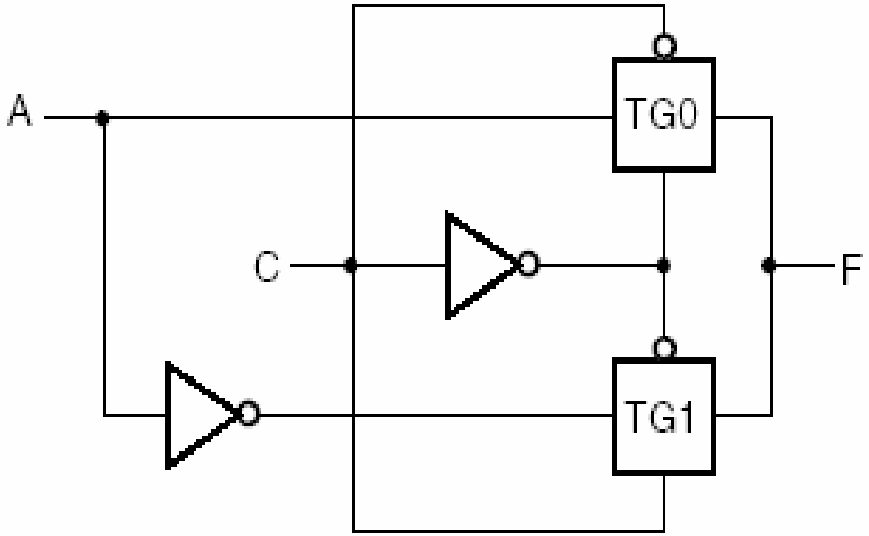
IEEE Symbol **Switching model** **Connection of Control inputs**

Fig. 2-44 Transmission Gate (TG)

Transmission Gates (XOR implementation)

Input A provides output and input C controls the path pf gates.

C=0 means TG0 is ON, and C = 1 means TG1 is ON.



(a)

A	C	TG1	TG0	F
0	0	No path	Path	0
0	1	Path	No path	1
1	0	No path	Path	1
1	1	Path	No path	0

(b)

Fig. 2-45 Transmission Gate Exclusive-OR

Combinational Circuit ??

Consists of input variables, output variables, logic gates and interconnections. For n input variables → 2^n possible input implementations. Thus, it can be specified by a truth table. Also by m ($m \leq 2^n$) Boolean functions, one for each possible output variable.

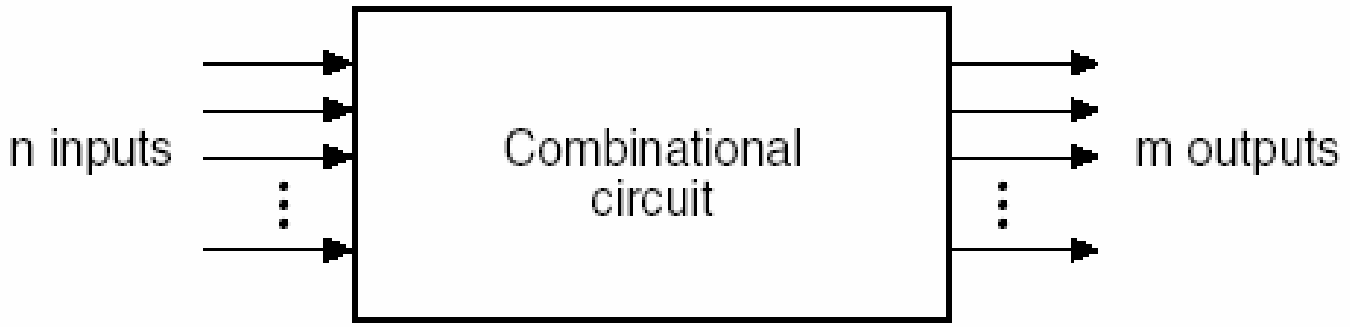


Fig. 3-1 Block Diagram of Combinational Circuit

A combinational circuit has no feedback paths and no storage elements.

Analysis of Combinational Circuits

Functional verification of the logic circuit.

Three Ways

- **Derivation of Boolean Functions:**
  - Label all gate outputs that are function only of input variables or their complements. Determine the Boolean functions for each gate output.
  - Label gates being functions of input variables and previously labeled gates. Find the Boolean functions for the outputs of these gates.
  - Repeat the previous process until circuit outputs are obtained in terms of input variables.
- **Derivation of the truth table:**
  - For n inputs list the binary numbers in a table from 0 to 2^n-1.
  - Break circuit into small single-output blocks by labeling each block output with an arbitrary symbol.
  - Obtain the truth table for the blocks depending on input variables only
  - Proceed until columns for all circuit outputs are determined.
- **Logic Simulation:**
  - Enter circuit as a netlist or schematic
  - Specify inputs as a file contents the simulator can read, or by entering inputs interactively into the simulator.
  - Proceed with simulation.

Combinational Analysis : Boolean functions

Gate output that are functions of input only : T_1=B'C and T_2=A'B

Other gates Outputs of : T_3=A+T_1=A+B'C, T_4=T_2XOR D, and T_5=T_2+D

Outputs : F_2=T_5=A'B+D and F_1=T_3+T_4 = A+B'C+BD'+B'D

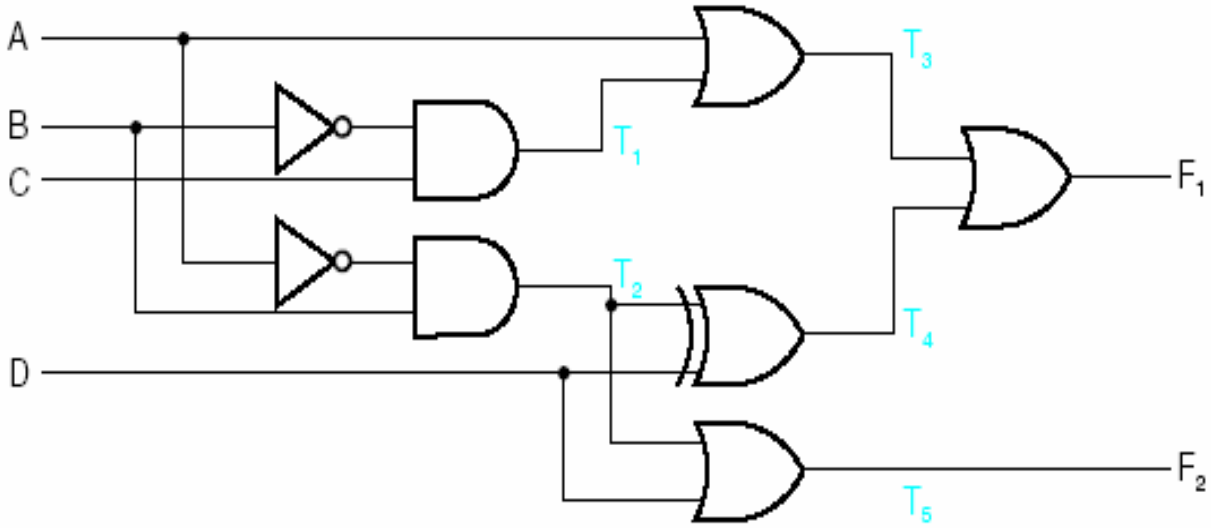


Fig. 3-5 Logic Diagram for Analysis Example

Combinational Analysis : Truth Table

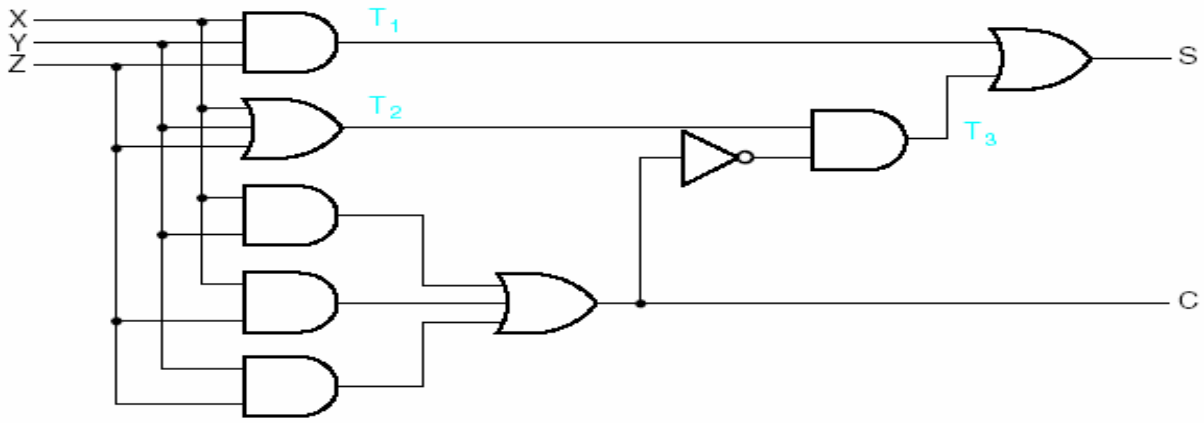


Fig. 3-6 Logic Diagram for Binary Adder

TABLE 3-1
Truth Table for Binary Adder

X	Y	Z	C	$\overline{C}$	T_1	T_2	T_3	S
0	0	0	0	1	0	0	0	0
0	0	1	0	1	0	1	1	1
0	1	0	0	1	0	1	1	1
0	1	1	1	0	0	1	0	0
1	0	0	0	1	0	1	1	1
1	0	1	1	0	0	1	0	0
1	1	0	1	0	0	1	0	0
1	1	1	1	0	1	1	0	1

Table 3-1 Truth Table for Binary Adder

Combinational Analysis : Simulation

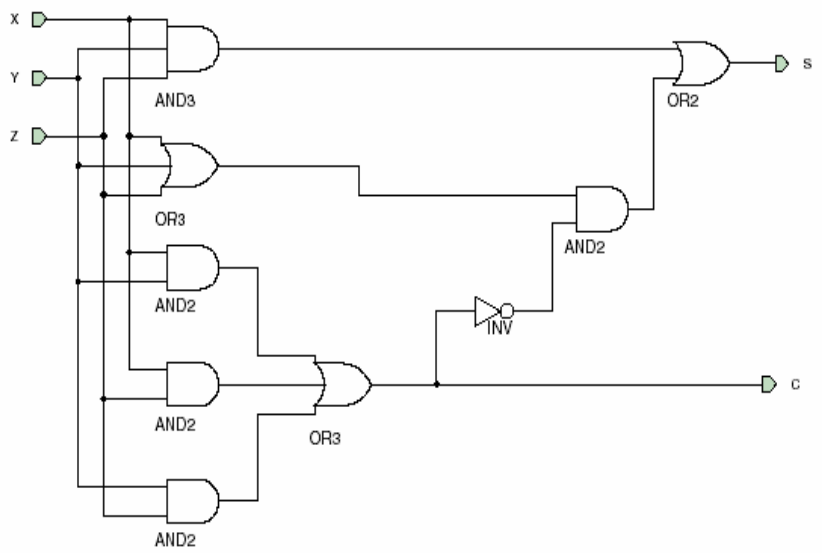


Fig. 3-7 Xilinx Foundation Schematic for Binary Adder in Figure 3-6

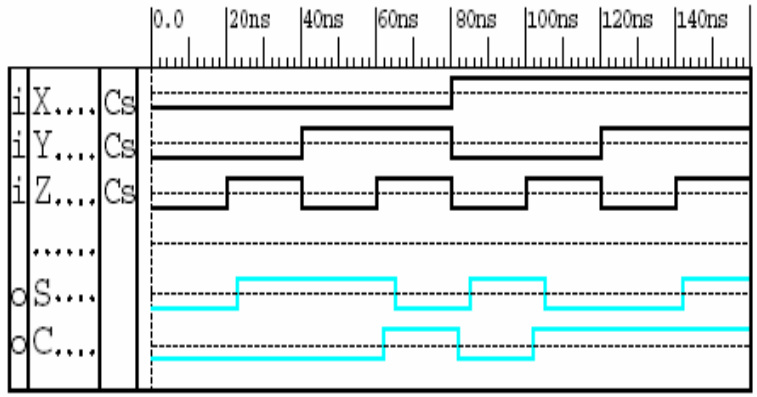


Fig. 3-8 Waveforms for the Binary Adder Schematic in Figure 3-7

To verify the correctness check the output waveform values against the truth table of full adder.

Combinational Circuits :Design Procedure

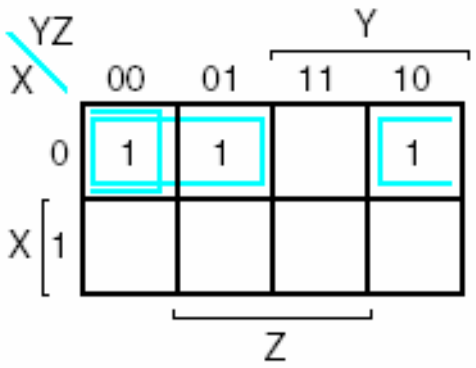
- Step 1: From circuit specifications determine required number of inputs and outputs, assign letter symbol.
- Step 2: Derive truth table that defines required relationship between inputs and outputs.
- Step 3: Obtain simplified Boolean functions for each output as a function of input variables.
- Step 4: Draw the logic diagram.
- Step 5: Verify design correctness.

Combinational Design Example

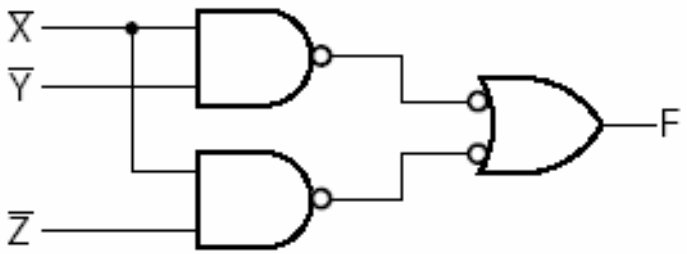
Design a combinatorial circuit with 3 inputs and 1 output, where the output must be logic 1 when the binary value of the inputs is less than 011 (3) and logic 0 otherwise. Use only NAND gates. (Input: X, Y, Z. Output: F)

X	Y	Z	F
0	0	0	1
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	0

(a) Truth table



(b) Map $F = \bar{X}\bar{Y} + \bar{X}Z$



(c) Logic diagram

Fig. 3-9 Solution to Example 3-1

Combinational Design Example : Code Converter

•For a combinatorial circuit with two or more outputs, each output must be expressed separately as a function of all the input variables.

•Example of a multi-output circuit is a code converter that translates info from one binary code to another. Inputs provide the elements bit combination as specified by the first code, the outputs generate the corresponding bit combination of the second code.

•Illustration via two examples:

- ➢BCD-to-Excess-3 code converter
- ➢BCD to the seven signals required to drive a seven-segment light-emitting diode (LED) display.

Comb. Exp. : BCD-to-Excess-3 code converter

Excess-3 code for a decimal digit is the binary combination corresponding to the decimal digit plus 3. Excess-3 code for decimal 5 is 8.

Only 10 out of 16 combinations are listed below. Rest treated as “don’t care conditions”. BCD and excess-3 use 4 bits → 4 input and 4 output variables.

TABLE 3-2
Truth Table for Code Converter Example

Decimal Digit	Input BCD				Output Excess-3			
	A	B	C	D	W	X	Y	Z
0	0	0	0	0	0	0	1	1
1	0	0	0	1	0	1	0	0
2	0	0	1	0	0	1	0	1
3	0	0	1	1	0	1	1	0
4	0	1	0	0	0	1	1	1
5	0	1	0	1	1	0	0	0
6	0	1	1	0	1	0	0	1
7	0	1	1	1	1	0	1	0
8	1	0	0	0	1	0	1	1
9	1	0	0	1	1	1	0	0

Table 3-2 Truth Table for Code Converter Example

Comb Exp : BCD-to-Excess-3 code converter ….

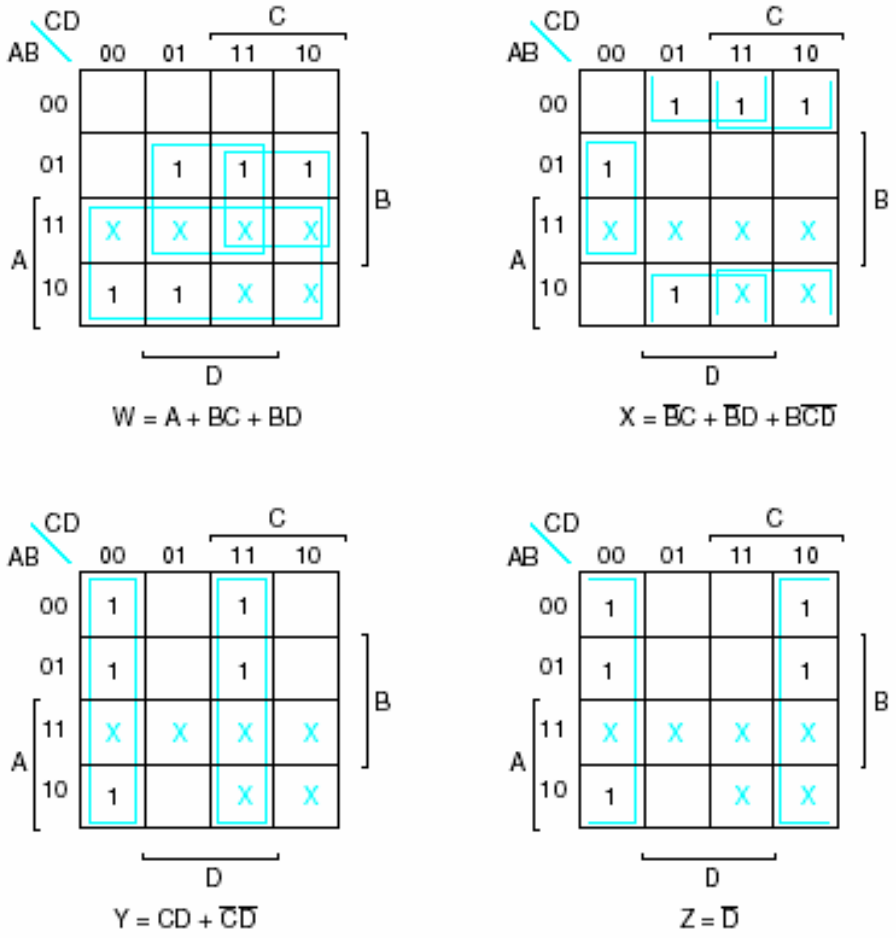


Fig. 3-10 Maps for BCD-to-Excess-3 Code Converter

Comb Exp : BCD-to-Excess-3 code converter …

W = A+BC+BD = A+B(C+D), Z = D', X = B'C+B'D+BC'D' = B'(C+D)+BC'D', and Y = CD+C'D' = (C XOR D)'

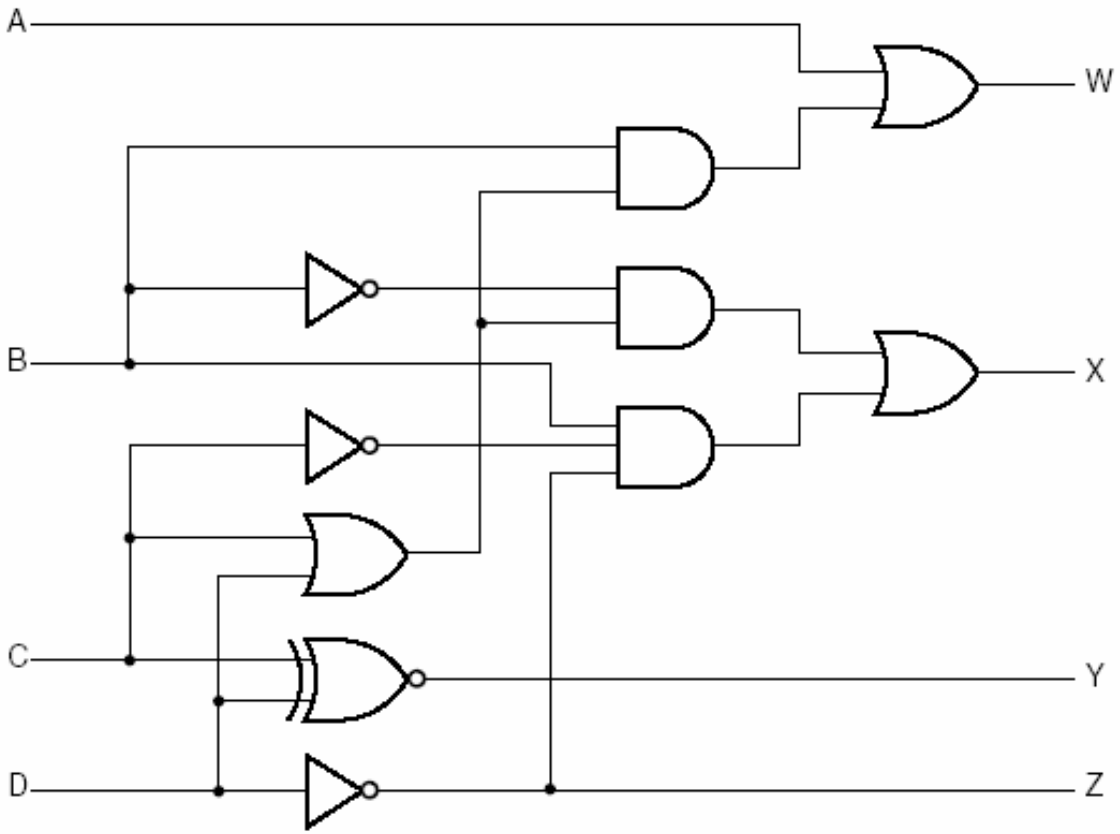
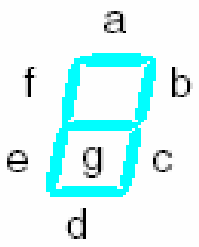


Fig. 3-11 Logic Diagram of BCD-to-Excess-3 Code Converter

Comb Exp : BCD-to-Seven-Segment Decoder

It accepts a decimal digit in BCD and generates the appropriate outputs for the selection of segments that display the decimal digit.



(a) Segment designation

(b) Numeric designation for display

Fig. 3-12 Seven-Segment Display

Comb Exp: BCD-to-Seven-Segment Decoder …

(a) Segment designation (b) Numeric designation for display

Fig. 3-12 Seven-Segment Display

BCD Input				Seven-Segment Decoder						
A	B	C	D	a	b	c	d	e	f	g
0	0	0	0	1	1	1	1	1	1	0
0	0	0	1	0	1	1	0	0	0	0
0	0	1	0	1	1	0	1	1	0	1
0	0	1	1	1	1	1	1	0	0	1
0	1	0	0	0	1	1	0	0	1	1
0	1	0	1	1	0	1	1	0	1	1
0	1	1	0	1	0	1	1	1	1	1
0	1	1	1	1	1	1	0	0	0	0
1	0	0	0	1	1	1	1	1	1	1
1	0	0	1	1	1	1	1	0	1	1
All other inputs				0	0	0	0	0	0	0

Table 3-3 Truth Table for BCD–to–Seven-Segment Decoder

Comb Exp: BCD-to-Seven-Segment Decoder …

One possible way of simplification results in the following Boolean expressions:

a= A'C+A'BD+B'C'D'+AB'C'

b= A'B'+A'C'D'+A'CD+AB'C'

c= A'B+A'D+B'C'D'+AB'C'

d= A'CD'+A'B'C+B'C'D'+AB'C'+A'BC'D

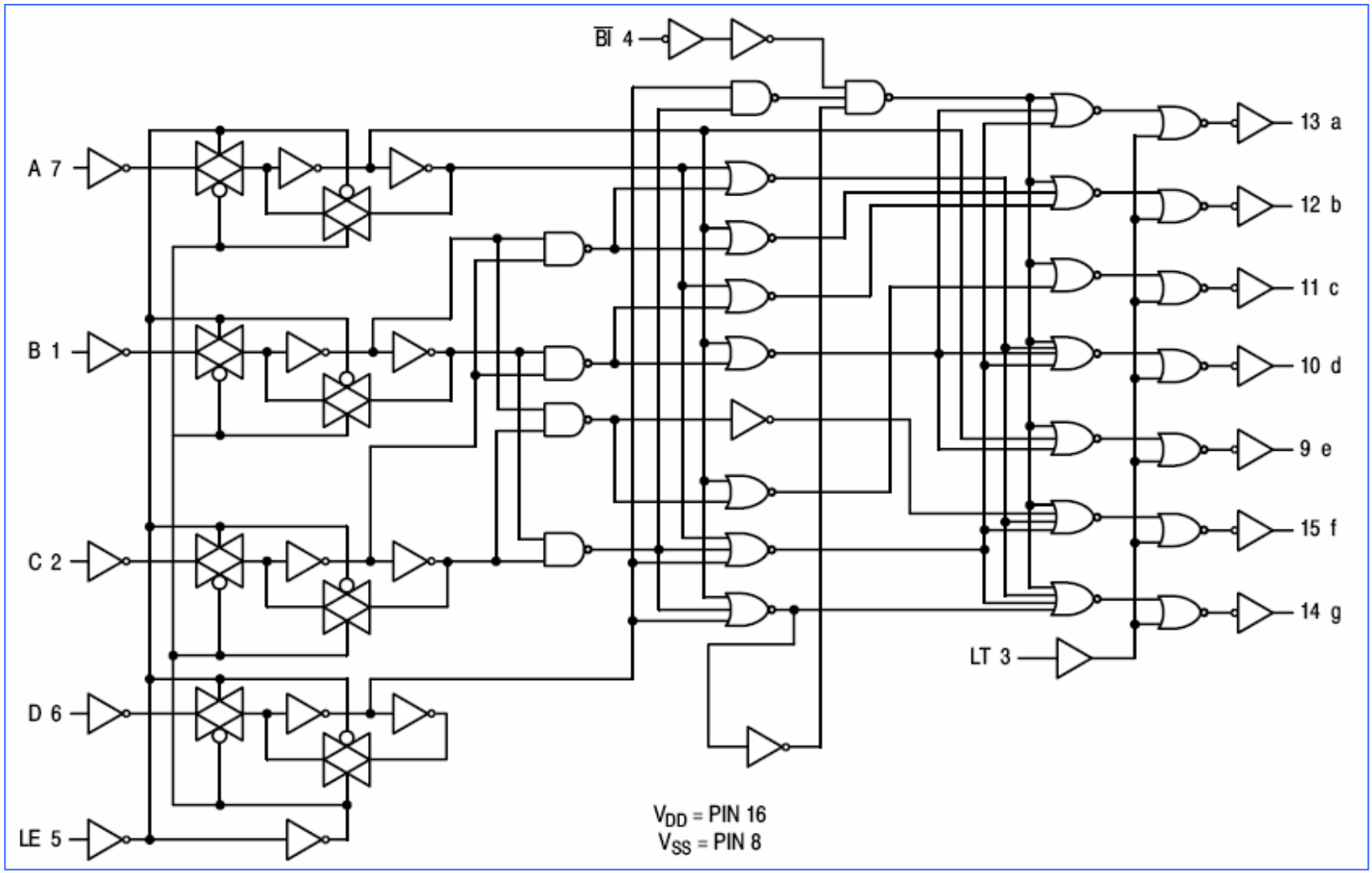
e= A'CD'+B'C'D'

f= A'BC'+A'C'D'+A'BD'+AB'C'

g= A'CD'+A'B'C+A'BC'+AB'C'

14 AND gates 7 OR gates. 27 product terms, 6 common terms.

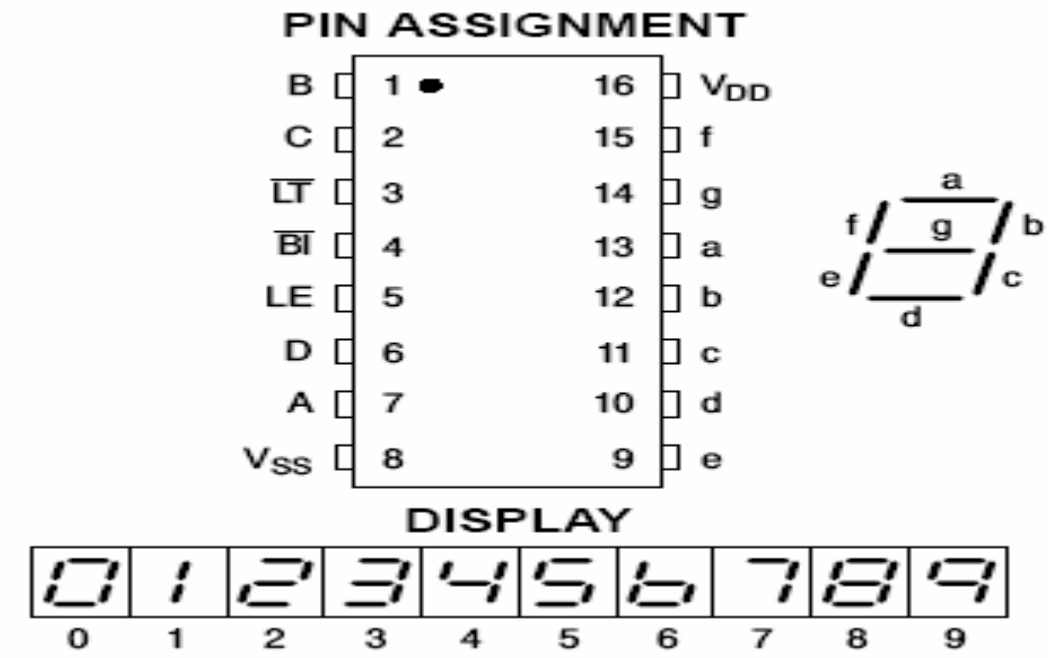
Comb Exp: BCD-to-Seven-Segment Decoder …



Comb Exp: BCD-to-Seven-Segment Decoder …

•The circuit shown in the previous slide is logic diagram of MC14511B from http://onsemi.com shown in the Fig. below.

•Lamp test (LT), blanking (BI), and latch enable (LE) inputs are used to test the display, to turn–off the display, and to store a BCD code, respectively.
•Applications: It can be used with LED, incandescent, fluorescent, gas discharge, or liquid crystal.



Decoders ??

•A decoder is a combinatorial circuit that converts binary information from the n coded inputs to a maximum of 2^n unique outputs.

•If the n bit coded information has unused bit combinations, then the decoder has $m \leq 2^n$ outputs.

•Decoders are called n-to-m line decoders. Their purpose is to generate the 2^n or fewer minterms of n input variables.

•Decoder operation may be clarified by its corresponding truth table.

•Some decoders are constructed directly using NAND gates since this is more economical.

Decoder Example: 3-to-8 line decoder

•3 inputs are decoded to 8 outputs.

•For each possible input combination, there are 7 outputs that are equal to 0 and only one that is equal to 1.

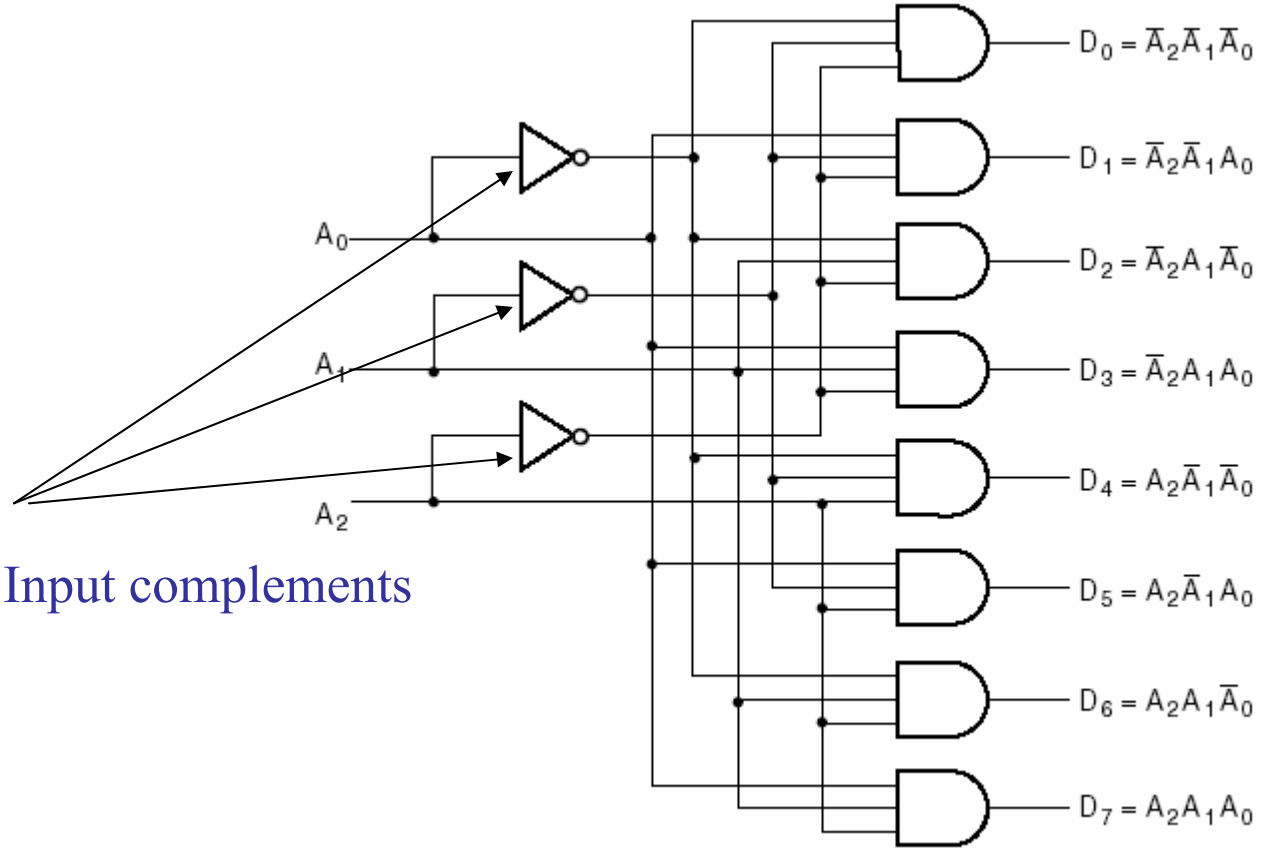
Truth Table for 3–to–8-Line Decoder

Inputs			Outputs							
A_2	A_1	A_0	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
0	0	0	0	0	0	0	0	0	0	1
0	0	1	0	0	0	0	0	0	1	0
0	1	0	0	0	0	0	0	1	0	0
0	1	1	0	0	0	0	1	0	0	0
1	0	0	0	0	0	1	0	0	0	0
1	0	1	0	0	1	0	0	0	0	0
1	1	0	0	1	0	0	0	0	0	0
1	1	1	1	0	0	0	0	0	0	0

Table 3-4 Truth Table for 3–to–8-Line Decoder

Decoder Example: 3-to-8 line decoder ….

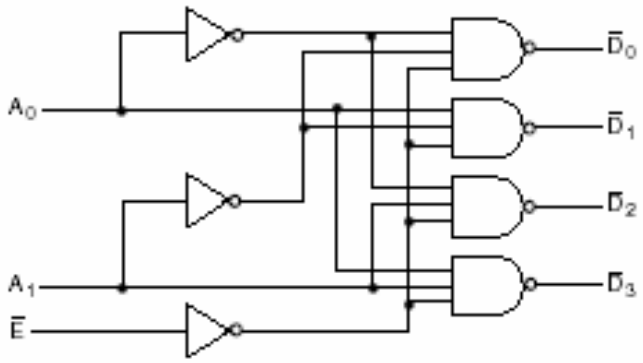
- 3 inverters provide complement of the outputs.
- Each one of the eight AND gates generate one minterm.



Input complements

Fig. 3-13 3-to-8-Line Decoder

Decoder Example: 2-to-4 line decoder



(a) Logic diagram

$\bar{E}$	A_1	A_0	$\bar{D}_0$	$\bar{D}_1$	$\bar{D}_2$	$\bar{D}_3$
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
1	X	X	1	1	1	1

(b) Truth table

$\bar{D}_0 = \overline{\bar{E}\,\bar{A}_1\,\bar{A}_0}$

$\bar{D}_1 = \overline{\bar{E}\,\bar{A}_1\,A_0}$

$\bar{D}_2 = \overline{\bar{E}\,A_1\,\bar{A}_0}$

$\bar{D}_3 = \overline{\bar{E}\,A_1\,A_0}$

(c) Logic Equations

Fig.3-14 A 2–to–4-Line Decoder

The decoder is enabled when E'=0. Only one output can be equal to 0 at any given time, all other outputs are equal to 1. Circuit is disabled when E'=1, regardless of the values of the other two inputs.

Decoder Expansion

When a certain size decoder is needed but only smaller sizes are available, combine two or more to get it.

Example: Two 2-to-4 line decoders are combined a 3-to-8 line decoder.

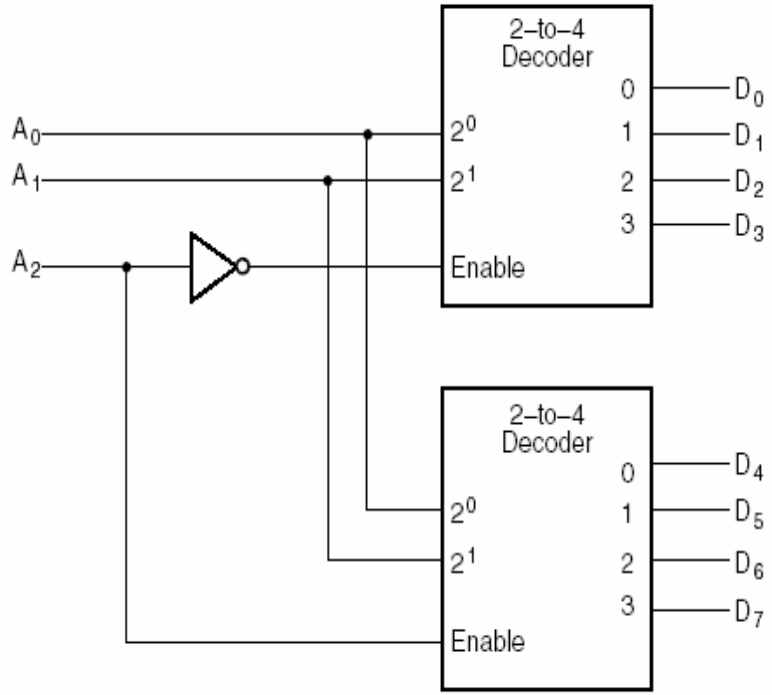


Fig. 3-15 A 3-to-8 Decoder Constructed with Two 2-to-4 Decoders

Decoder Expansion : Explanations……

•The two LSB of the input are connected to both decoders. The MSB is connected to the enable input of one decoder and through an inverter to the enable input of the other decoder.

•When A_2=0, the upper decoder is enabled, the lower is disabled. The outputs of the lower decoder become inactive and equal to 0. The upper decoder generates the minterms D_0 to D_3. When A_2=1, the enable conditions are reversed and the minterms D_4 to D_7 are generated.

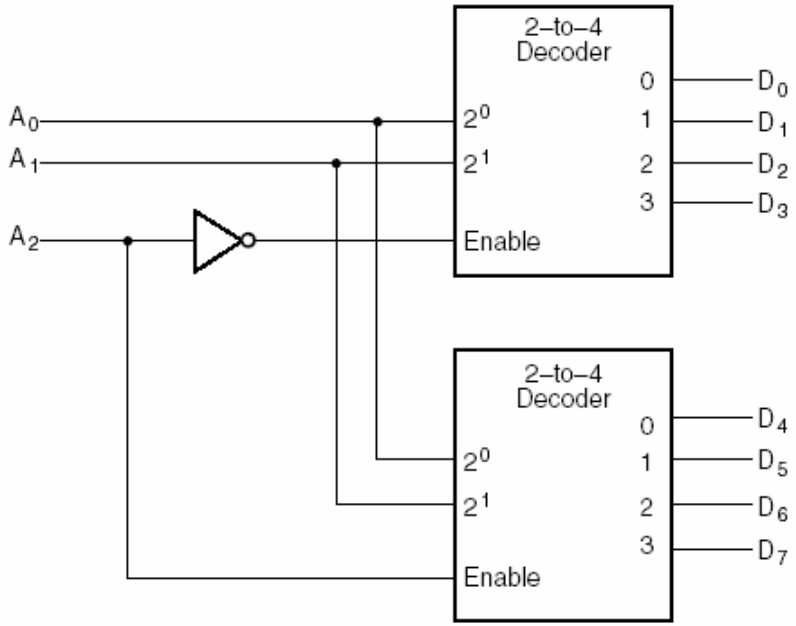


Fig. 3-15 A 3-to-8 Decoder Constructed with Two 2-to-4 Decoders

Decoder to Any Combinatorial Circuit

- Decoder provides the 2^n minterms of n input variables. Since any Boolean function can be expressed as a sum of minterms, one can use a decoder to generate the minterms and an external OR gate to form their logical sum.
- Thus, any combinatorial circuit with n inputs and m outputs can be implemented with an n-to-2^n-line decoder and m OR gates.
- This procedure for implementing a combinatorial circuit by means of a decoder and OR gates, requires that the Boolean functions be expressed as a sum of minterms. This form can be obtained from the Truth Table or by plotting each function on a map. The inputs to each OR gate are selected from the decoder outputs according to the list of minterms of each function.

Decoder to A Binary Full Adder

TABLE 3-1
Truth Table for Binary Adder

X	Y	Z	C	$\overline{C}$	T_1	T_2	T_3	S
0	0	0	0	1	0	0	0	0
0	0	1	0	1	0	1	1	1
0	1	0	0	1	0	1	1	1
0	1	1	1	0	0	1	0	0
1	0	0	0	1	0	1	1	1
1	0	1	1	0	0	1	0	0
1	1	0	1	0	0	1	0	0
1	1	1	1	0	1	1	0	1

Table 3-1 Truth Table for Binary Adder

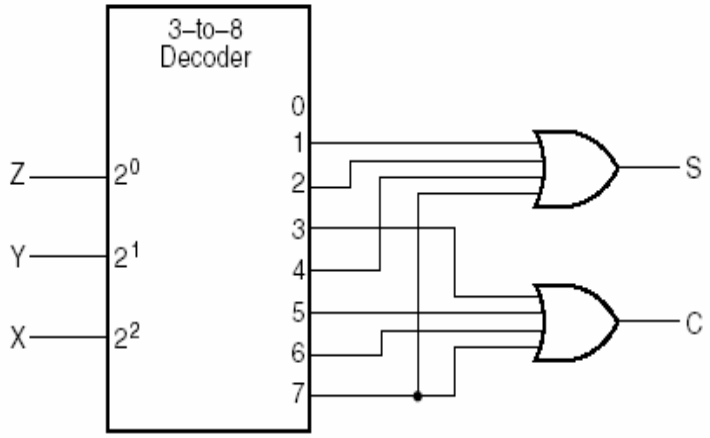


Fig. 3-16 Implementing a Binary Adder Using a Decoder

Full Adder as Sum-of-Minterms :

$S(X,Y,Z) = \Sigma m(1,2,4,7)$

$X(X,Y,Z) = \Sigma m(3,5,6,7)$

Encoders ??

Performs the inverse operation of a decoder. It has 2^n (or fewer) input lines and n output lines.

Example: Octal-to-binary encoder. Eight inputs corresponds to eight octal digits. Only one input has the value 1 at any given time. It can be implemented with 3 4-input OR gates.

Truth Table for Octal–to–Binary Encoder

Inputs								Outputs		
D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0	A_2	A_1	A_0
0	0	0	0	0	0	0	1	0	0	0
0	0	0	0	0	0	1	0	0	0	1
0	0	0	0	0	1	0	0	0	1	0
0	0	0	0	1	0	0	0	0	1	1
0	0	0	1	0	0	0	0	1	0	0
0	0	1	0	0	0	0	0	1	0	1
0	1	0	0	0	0	0	0	1	1	0
1	0	0	0	0	0	0	0	1	1	1

Table 3-5 Truth Table for Octal–to–Binary Encoder

Boolean Functions

$A_0 = D_1 + D_3 + D_5 + D_7$

$A_1 = D_2 + D_3 + D_6 + D_7$

$A_2 = D_4 + D_5 + D_6 + D_7$

NOTE: Design of a priority encoder can be considered similarly.