ITI1100 Section Z
Digital Systems I

Chapter 4:

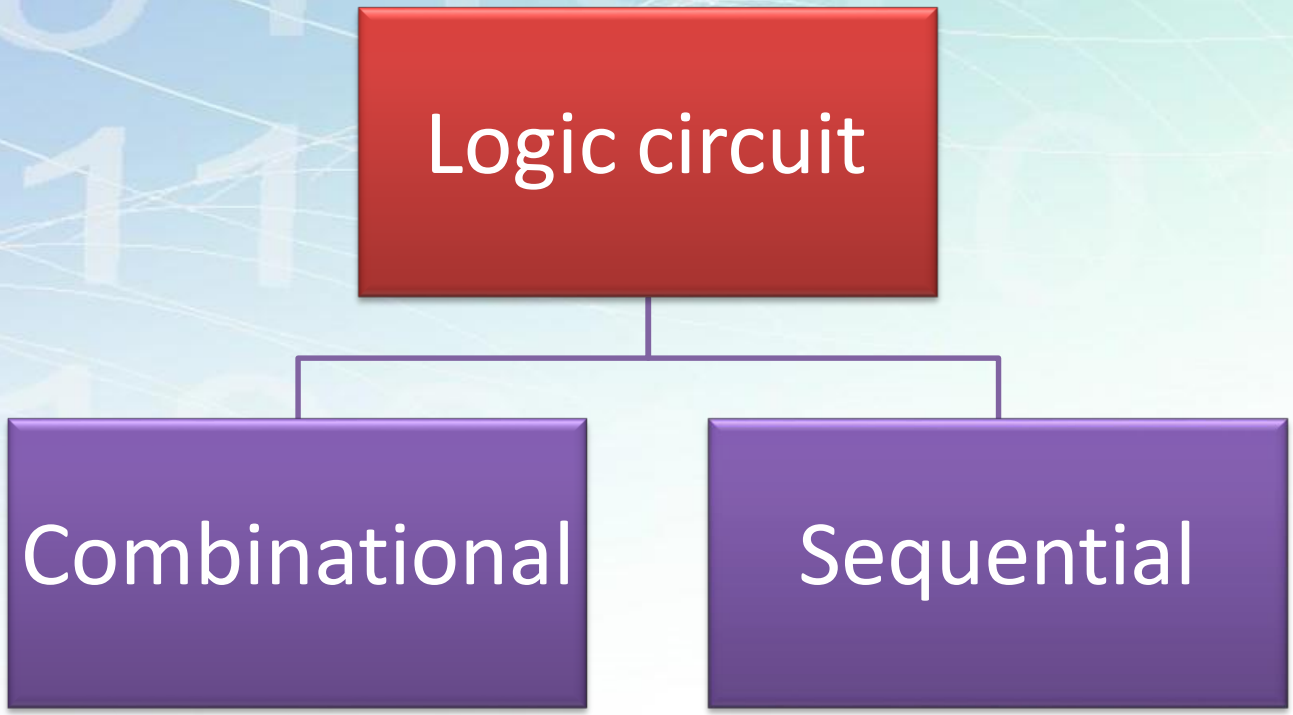
Combinational Logic

(1)

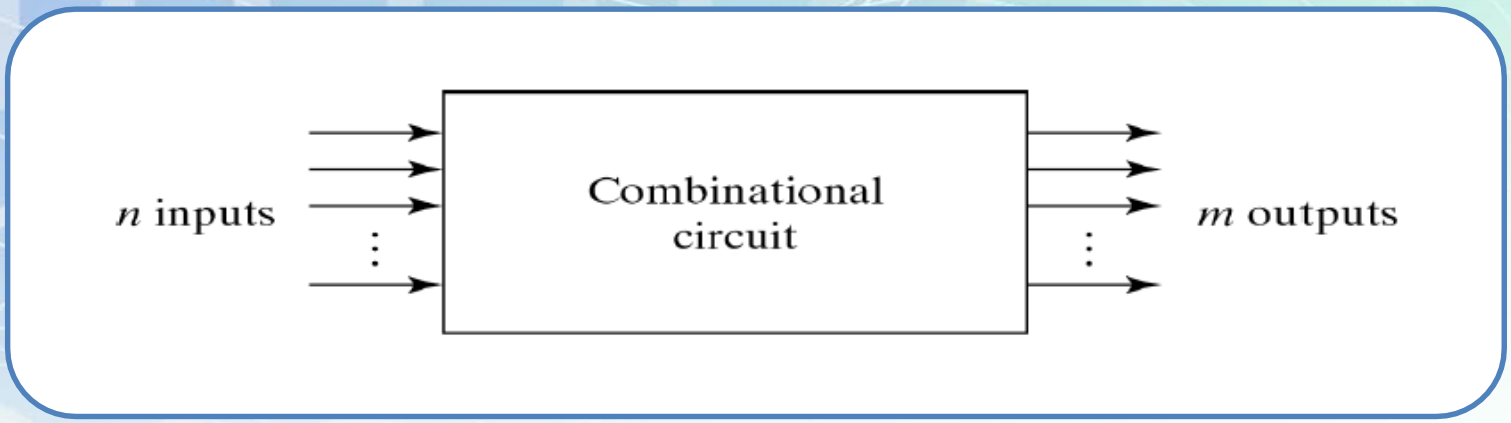
Prof. Mohammad Alja'Afreh

Summer 2019

Logic Circuits

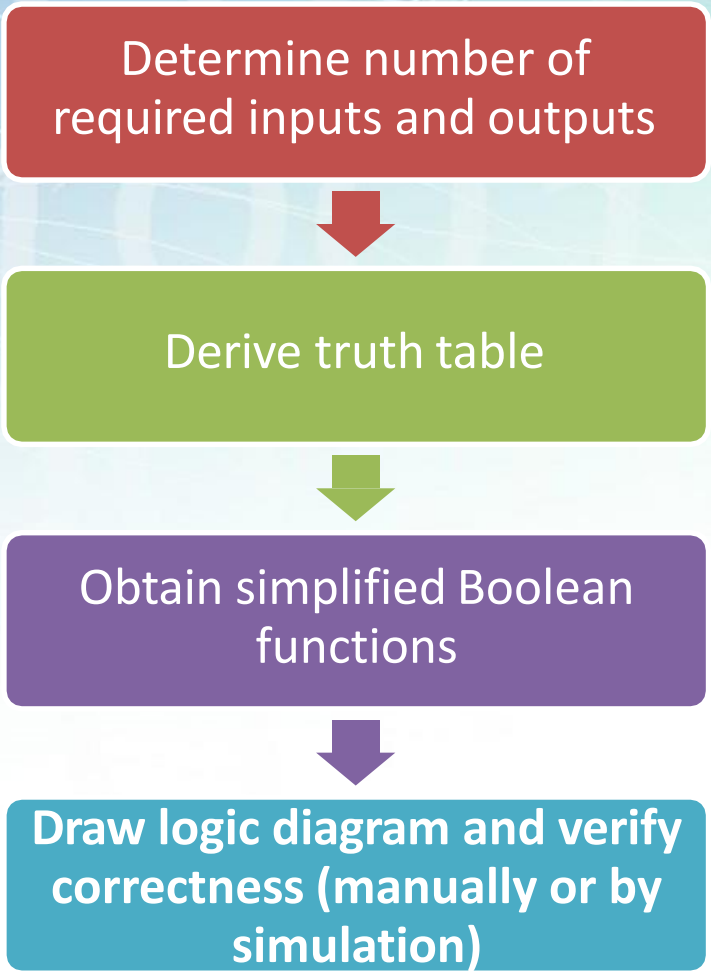


Combinational Logic Circuits



- Outputs logical functions of inputs
- New outputs appear shortly after changed inputs (propagation delay)
- No feedback loops
- No clock.

Design Procedure

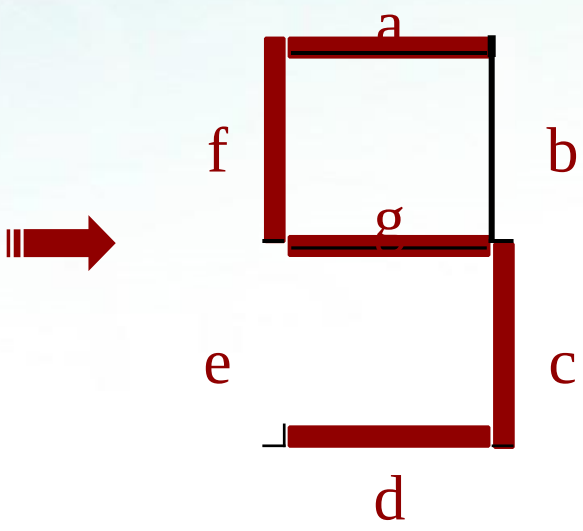


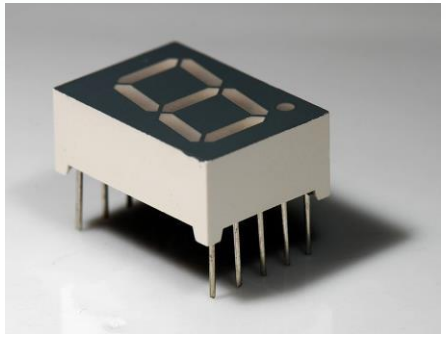
Combinational Circuit Examples

BCD to Seven Segment Decoder

- **Binary Coded Decimal (BCD)** uses **4 bits** to **represent decimal digits 0 - 9**
- **Decoder** is used to **display BCD** numbers using **seven illuminated segments LED**

0	0	0	0	0
1	0	0	0	1
2	0	0	1	0
.	.	.	.	.
7	0	1	1	1
8	1	0	0	0
9	1	0	0	1

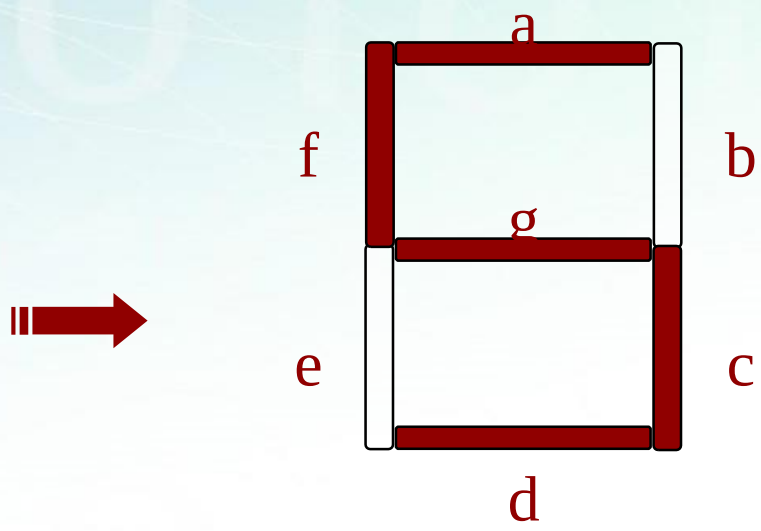




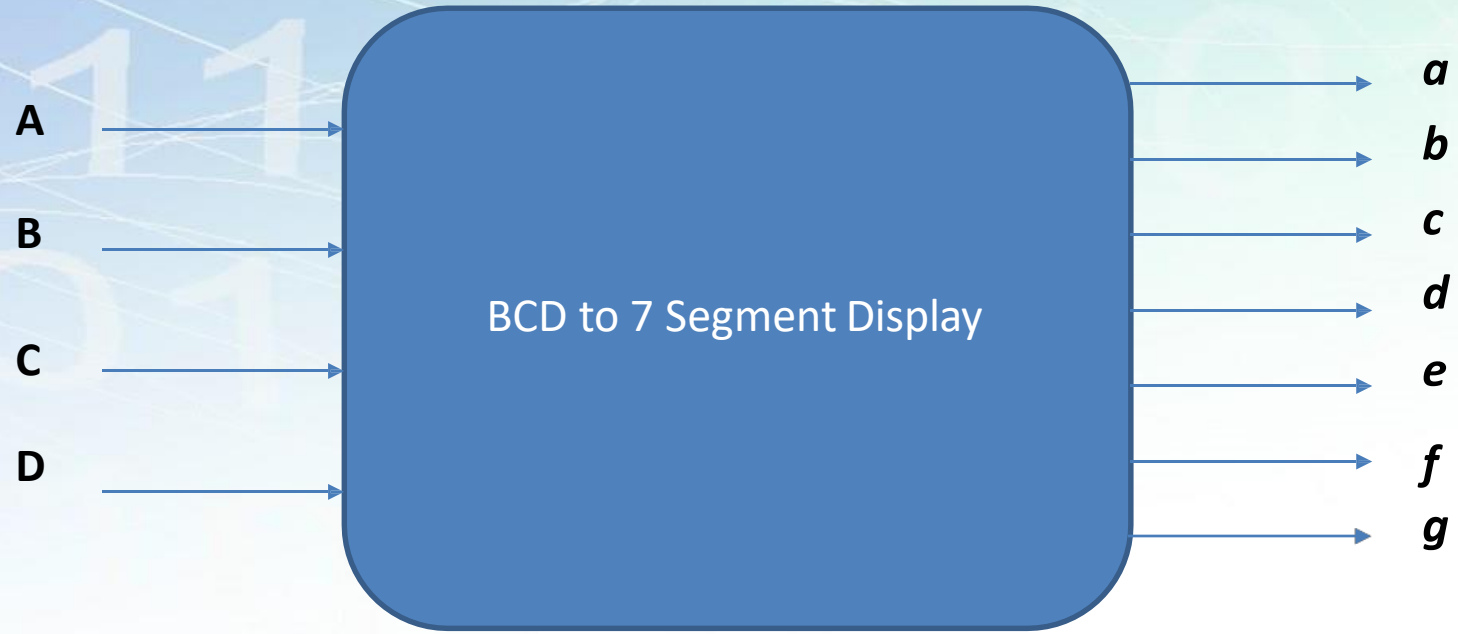
BCD to Seven Segment Decoder (cont'd)

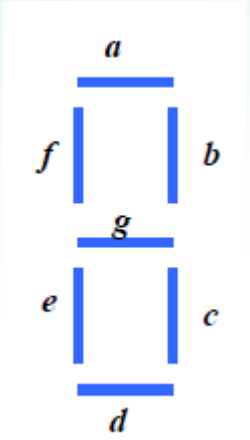
- The **segments** that should be **illuminated** for **each digit**

0	a,b,c,d,e,f
1	b,c
2	a,b,d,e,g
3	a,b,c,d,g
4	b,c,f,g
5	a,c,d,f,g
6	a,c,d,e,f,g
7	a,b,c
8	a,b,c,d,e,f,g
9	a,b,c,d,f,g



BCD to Seven Segment Decoder (cont'd)





BCD to Seven Segment Decoder (cont'd)

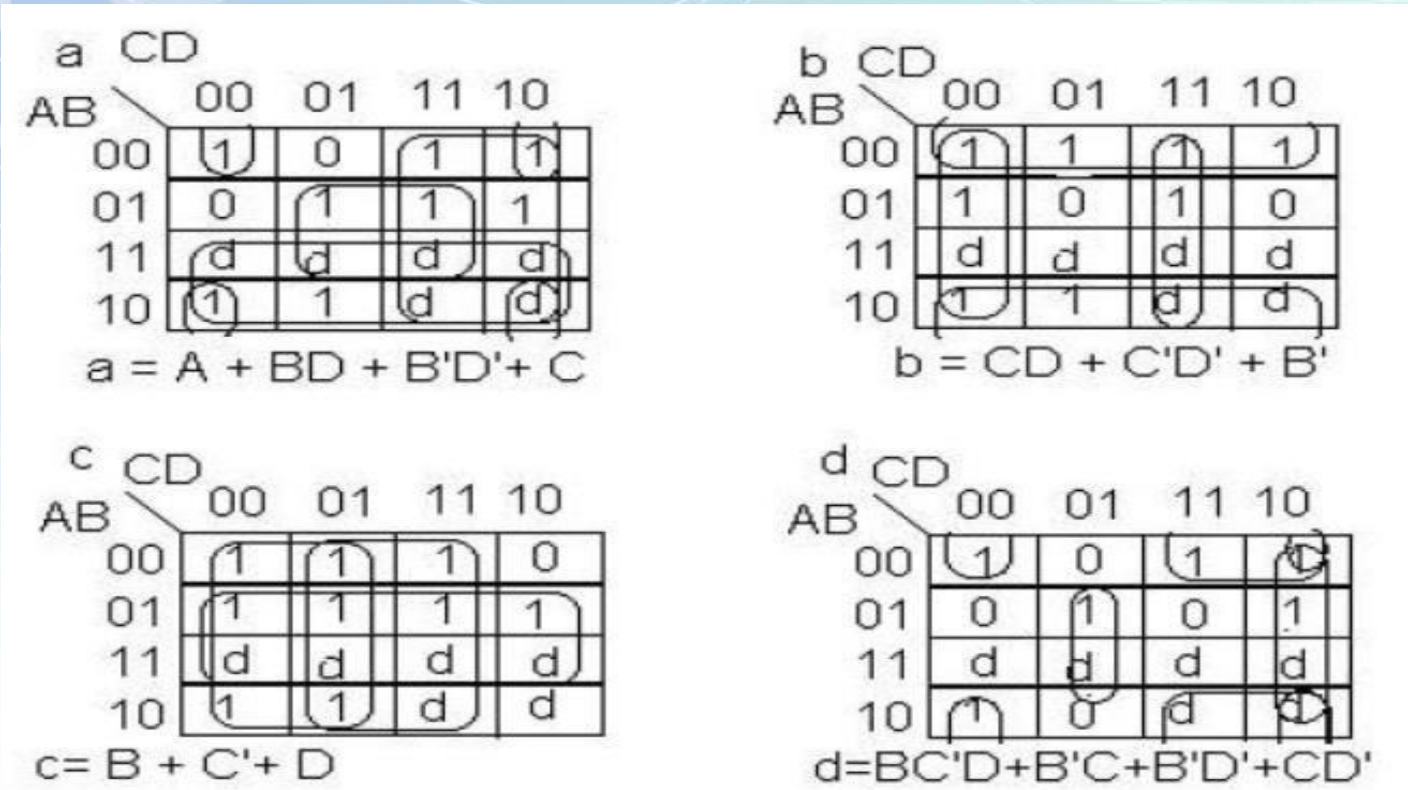
Truth Table

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
1	0	1	0	x	x	x	x	x	x	x
1	0	1	1	x	x	x	x	x	x	x
1	1	0	0	x	x	x	x	x	x	x
1	1	0	1	x	x	x	x	x	x	x
1	1	1	0	x	x	x	x	x	x	x
1	1	1	1	x	x	x	x	x	x	x

Don't care terms

BCD to Seven Segment Decoder (cont'd)

Simplification with K-map



- **Note: d = x**
- **Do k-maps for e, f, and g**

BCD to Excess-3 Code Converter

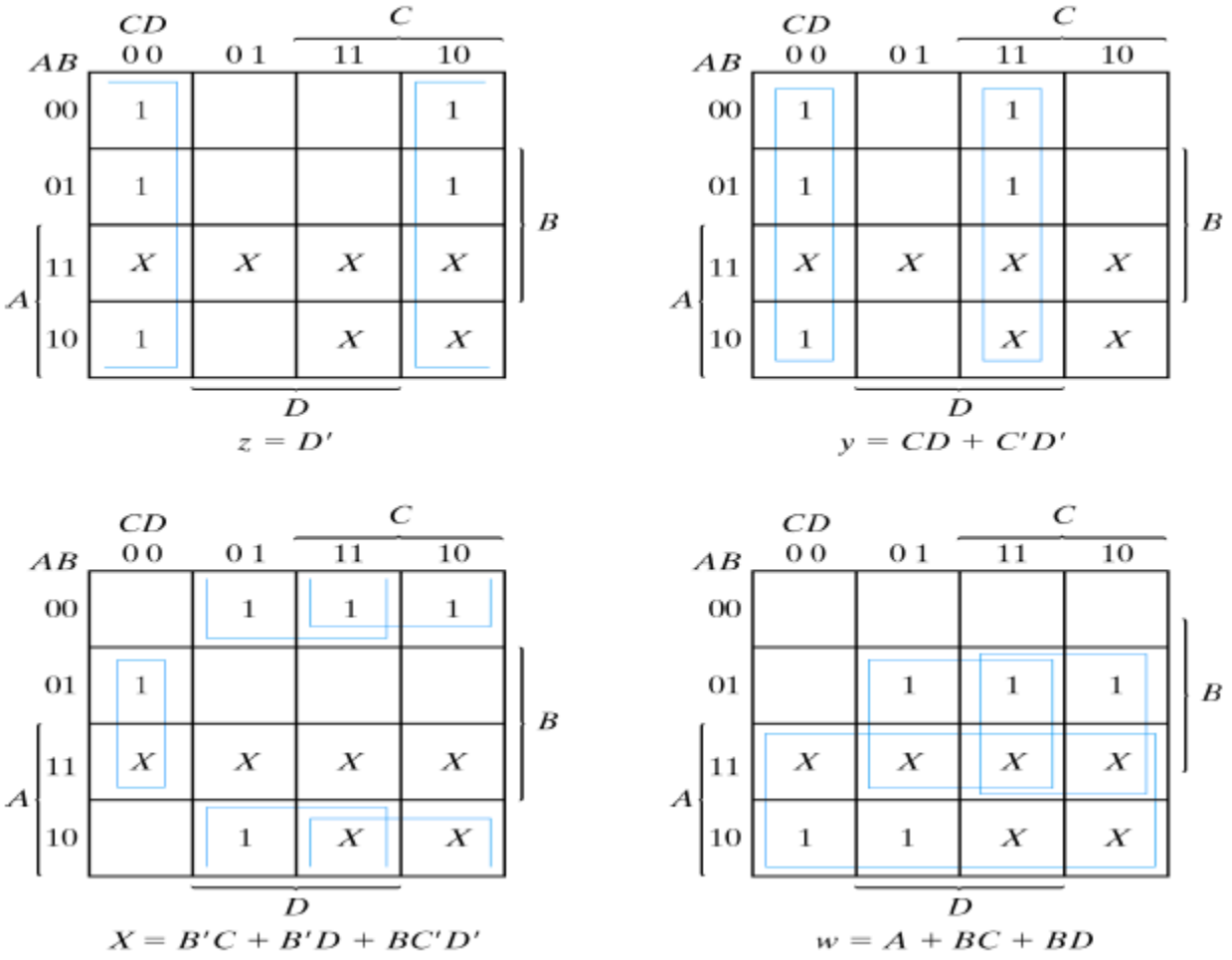
- A circuit that translates **BCD binary code** to **Excess-3 code**
- **Excess-3 code: BCD code + 3**
- BCD inputs **1010 to 1111** are **don't care** terms

Truth Table

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

BCD to Excess-3 Code Converter (cont'd)

Simplification with K-map



BCD to Excess-3 Code Converter (cont'd)

- **Two-level AND-OR implementation** for the circuit can be obtained directly **from the Boolean expression derived from the K-maps**
- **Further manipulation** can be done on the function to **allow use of common gates for multiple-output circuits**
- There are several possibilities for the implementation. The following shows the **implementation with 3 levels of gates**

$$W = A + BC + BD = A + B(C + D)$$

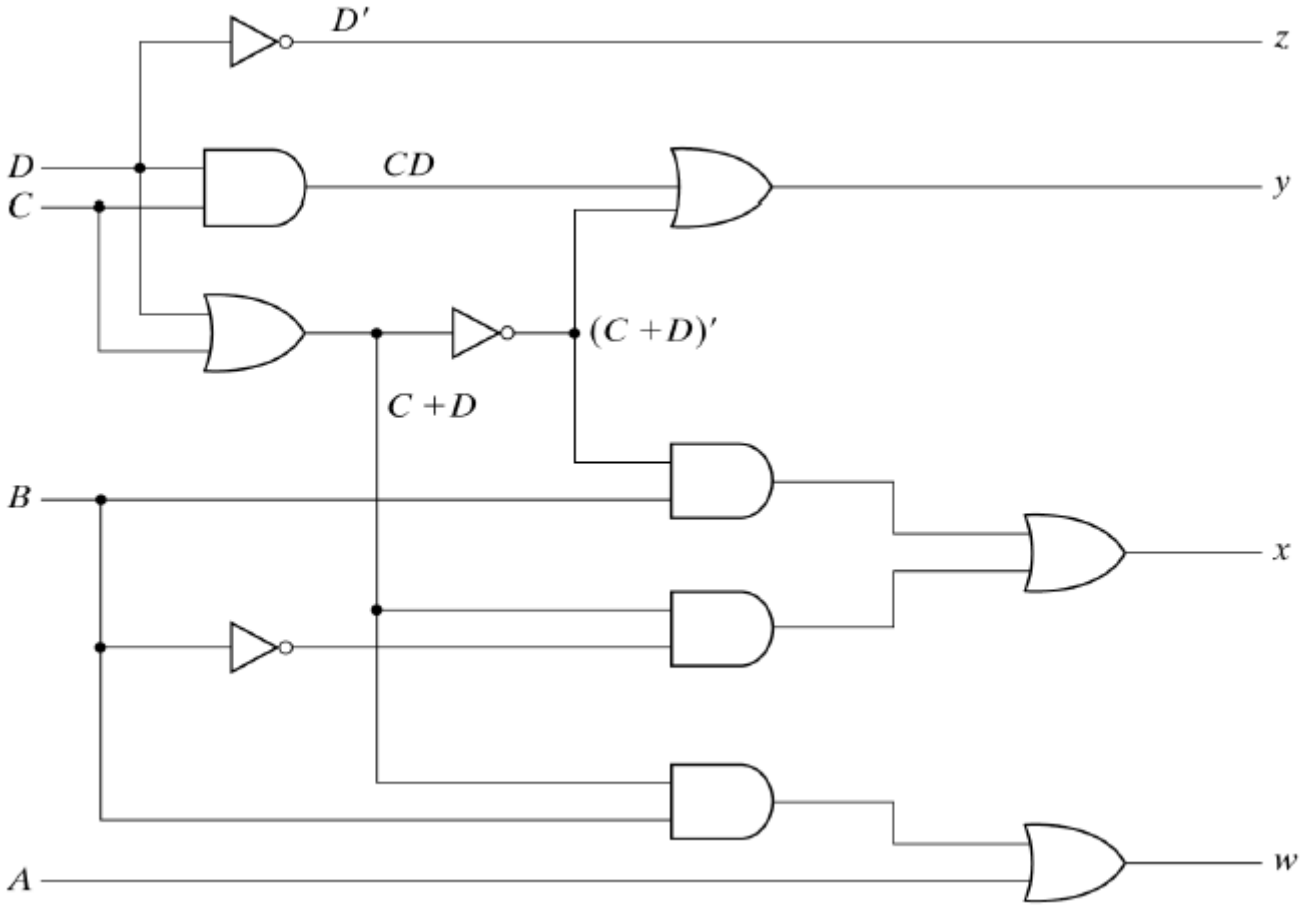
$$X = B'C + B'D + BC'D' = B'(C + D) + BC'D'$$

$$= B'(C+D) + B(C+D)'$$

$$Y = CD + C'D' = CD + (C + D)'$$

$$Z = D'$$

BCD to Excess-3 Code Converter (cont'd)



The 3-level logic circuit requires fewer gates

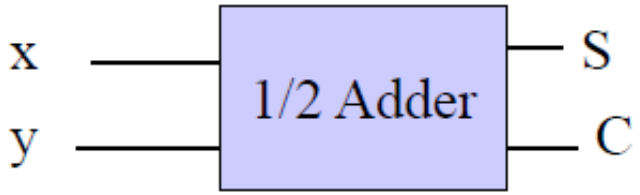
Binary Adders and Subtractors

Half Adder

- The **half-adder** accepts **two binary digits on its inputs** and produces **two binary digits on its outputs**: **a sum bit and a carry bit**

Truth Table

x	y	C	S
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

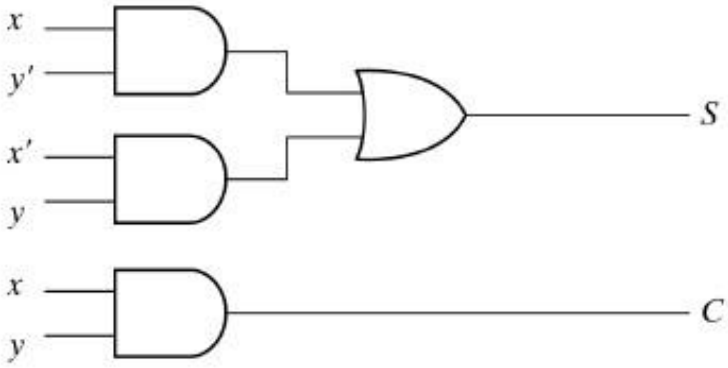


Some of products

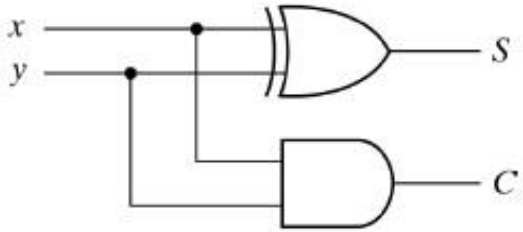
$$S = x'y + xy'$$

$$C = xy$$

Half Adder (cont'd)



(a) $S = xy' + x'y$
$C = xy$



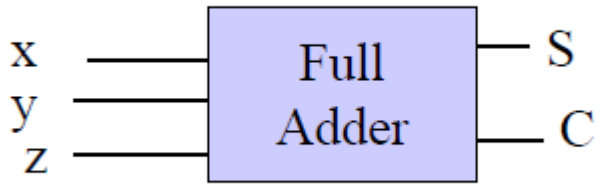
(b) $S = x \oplus y$
$C = xy$

Full Adder

- The **Full-adder** accepts **two input bits and an input carry** and generates a **sum output and an output carry**
- **Basic difference** between a full and a half adder is that the **full adder accepts an input carry**

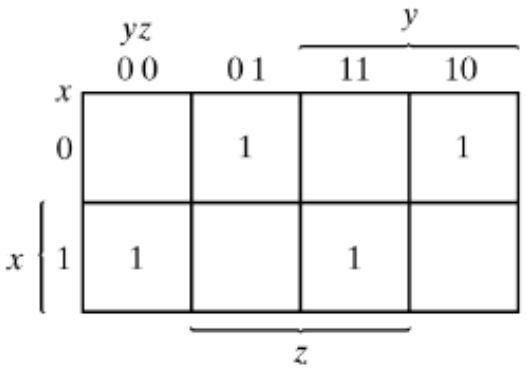
Truth Table

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

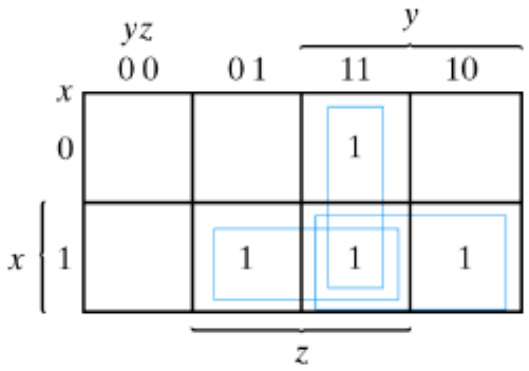


Full Adder (cont'd)

Simplification with K-map

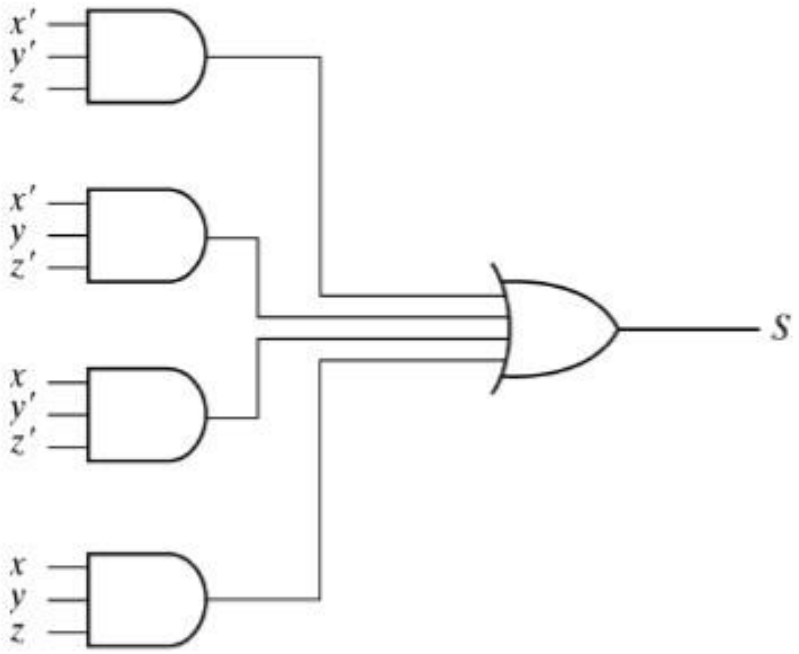


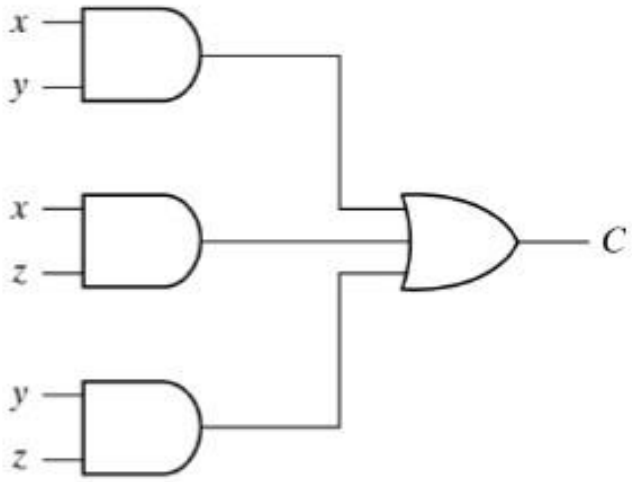
$S = x'y'z + x'yz' + xy'z' + xyz$



$C = xy + xz + yz$

Full Adder (cont'd)





Full-Adder

Truth Table				
X	Y	Z	C_o	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

$$C_o = X'YZ + XY'Z + XYZ' + XYZ$$

$$= Z[X'Y + XY'] + XY[Z' + Z]$$

$$= Z[X \oplus Y] + XY \cdot 1$$

$$= Z(X \oplus Y) + XY$$

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

$$= X'[Y'Z + YZ'] + X[Y'Z' + YZ]$$

$$= X'[Y \oplus Z] + X[Y \oplus Z]'$$

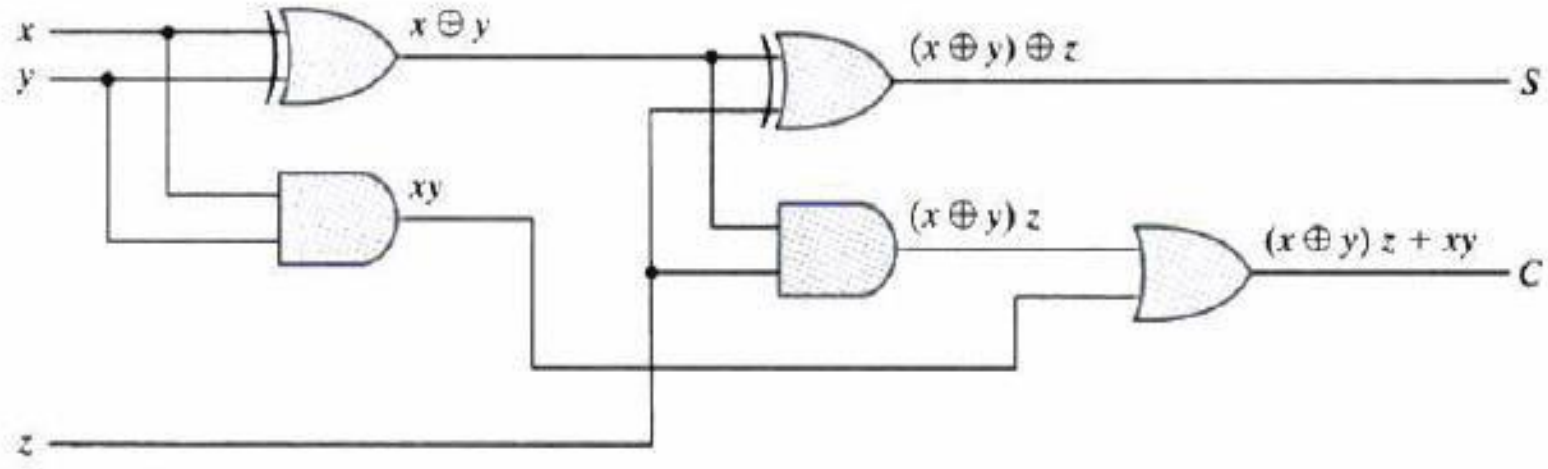
$$[X] \qquad [X]'$$

$$= X' X + X X'$$

$$= X \oplus X$$

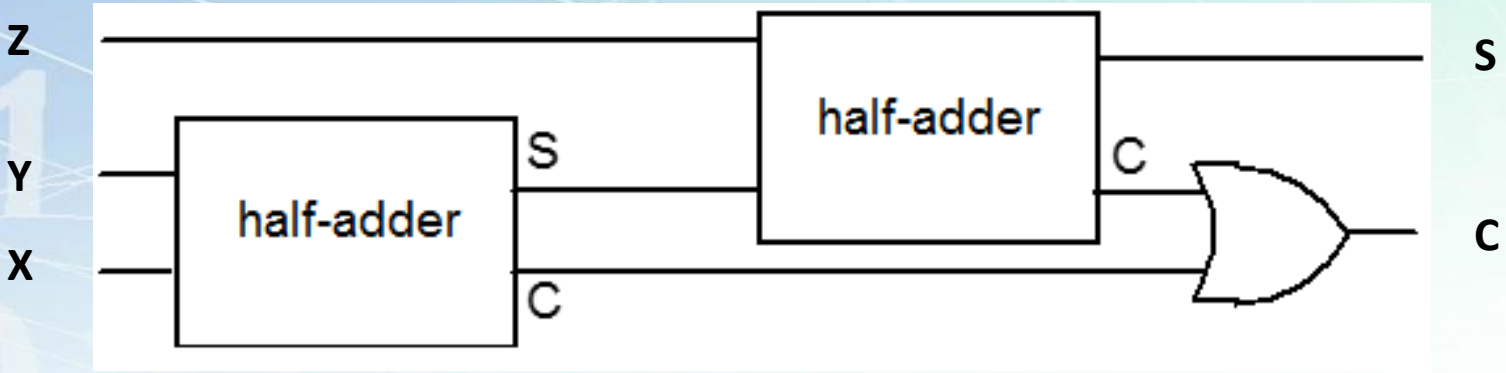
$$= X \oplus Y \oplus Z$$

Full Adder (cont'd)



The 3-level logic circuit with XOR gates

Full Adder (cont'd)



A full adder can be made from two half adders plus an OR gate