Digital Electronics & Logic Design

(EC 207)

Dr. Vivek Garg

Department of Electronics Engineering

S. V. National Institute of Technology (SVNIT)
Surat

email: vivekg@eced.svnit.ac.in; vivekgarg0101@gmail.com

Course Outline

- **PN DIODE AND TRANSITOR** **(04 Hours)**

  PN Diode Theory, PN Characteristic and Breakdown Region, PN Diode Application as Rectifier, Zener Diode Theory, Zener Voltage Regulator, Diode as Clamper and Clipper, Photodiode Theory, LED Theory, 7 Segment LED Circuit Diagram and Multi Colour LED, LASER Diode Theory and Applications, Bipolar Junction Transistor Theory, Transistor Symbols And Terminals, Common Collector, Emitter and Base Configurations, Different Biasing Techniques, Concept of Transistor Amplifier, Introduction to FET Transistor And Its Feature.

- **WAVESHAPING CIRCUITS AND OPERATIONAL AMPLIFIER** **(06 Hours)**

  Linear Wave Shaping Circuits, RC High Pass and Low Pass Circuits, RC Integrator and Differentiator Circuits, Nonlinear Wave Shaping Circuits, Two Level Diode Clipper Circuits, Clamping Circuits, Operational Amplifier OP-AMP with Block Diagram, Schematic Symbol of OP-AMP, The 741 Package Style and Pinouts, Specifications of Op-Amp, Inverting and Non-Inverting Amplifier, Voltage Follower Circuit, Multistage OP-AMP Circuit, OP-AMP Averaging Amplifier, OP-AMP Subtractor.

- **BOOLEAN ALGEBRA AND SWITCHING FUNCTIONS** **(04 Hours)**

  Basic Logic Operation and Logic Gates, Truth Table, Basic Postulates and Fundamental Theorems of Boolean Algebra, Standard Representations of Logic Functions- SOP and POS Forms, Simplification of Switching Functions-K-Map and Quine-Mccluskey Tabular Methods, Synthesis of Combinational Logic Circuits.

- **COMBINATIONAL LOGIC CIRCUIT USING MSI INTEGRATED CIRCUITS** **(07 Hours)**

Course Outline

Binary Parallel Adder; BCD Adder; Encoder, Priority Encoder, Decoder; Multiplexer and Demultiplexer Circuits; Implementation of Boolean Functions Using Decoder and Multiplexer; Arithmetic and Logic Unit; BCD to 7-Segment Decoder; Common Anode and Common Cathode 7-Segment Displays; Random Access Memory, Read Only Memory And Erasable Programmable ROMS; Programmable Logic Array (PLA) and Programmable Array Logic (PAL).

- **INTRODUCTION TO SEQUENTIAL LOGIC CIRCUITS** **(04 Hours)**

  Basic Concepts of Sequential Circuits; Cross Coupled SR Flip-Flop Using NAND or NOR Gates; JK Flip-Flop Rise Condition; Clocked Flip-Flop; D-Type and Toggle Flip-Flops; Truth Tables and Excitation Tables for Flip-Flops; Master Slave Configuration; Edge Triggered and Level Triggered Flip-Flops; Elimination of Switch Bounce using Flip-Flops; Flip-Flops with Preset and Clear.

- **SEQUENTIAL LOGIC CIRCUIT DESIGN** **(06 Hours)**

  Basic Concepts of Counters and Registers; Binary Counters; BCD Counters; Up Down Counter; Johnson Counter, Module-N Counter; Design of Counter Using State Diagrams and Table; Sequence Generators; Shift Left and Right Register; Registers With Parallel Load; Serial-In-Parallel-Out (SIPO) And Parallel-In-Serial-Out(PISO); Register using Different Type of Flip-Flop.

- **REGISTER TRANSFER LOGIC** **(04 Hours)**

  Arithmetic, Logic and Shift Micro-Operation; Conditional Control Statements; Fixed-Point and Floating-Point Data; Arithmetic Shifts; Instruction Code and Design Of Simple Computer.

- **PROCESSOR LOGIC DESIGN** **(03 Hours)**

  Processor Organization; Design of Arithmetic Logic Unit; Design of Accumulator.

- **CONTROL LOGIC DESIGN** **(04 Hours)**

  Control Organization; Hard-Wired Control; Micro Program Control; Control Of Processor Unit; PLA Control.

Course Text and Materials

1. Schilling Donald L. and Belove E., "Electronics Circuits- Discrete and Integrated", 3rd Ed., McGraw-Hill, 1989, Reprint 2008.
2. Millman Jacob, Halkias Christos C. and Parikh C., "Integrated Electronics", 2nd Ed., McGraw-Hill, 2009.
3. Taub H. and Mothibi Suryaprakash, Millman J., "Pulse, Digital and Switching Waveforms", 2nd Ed., McGraw-Hill, 2007.
4. Mano Morris, "Digital Logic and Computer Design", 5th Ed., Pearson Education, 2005.
5. Lee Samual, "Digital Circuits and Logic Design", 1st Ed., PHI, 1998.

Logic Operations and Logic Gates

- Basic building blocks of a digital circuit
- Data processing on the circuit is controlled using transistors
- Output depends on the logic gate and the input
- Input is one of two states – high or low
- Output is one of two states – high or low
- There are seven types of logic gates

Logic Operations and Logic Gates

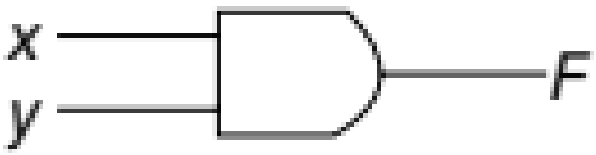
- Inverter (NOT Gate)
- AND Gate
- OR Gate
- Exclusive-OR (XOR) Gate

- NAND Gate = AND Gate + Inverter
- NOR Gate = OR Gate + Inverter
- Exclusive-NOR Gate = XOR Gate + Inverter

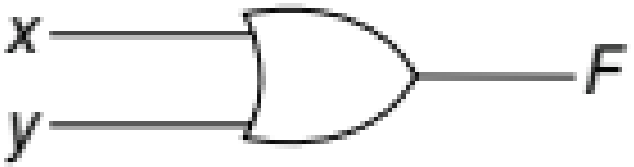
Logic Operations and Logic Gates

There are three basic types of logic gate.

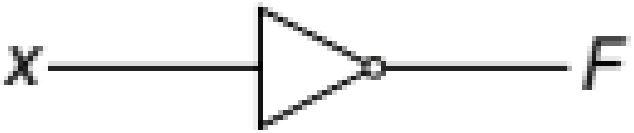
- AND gate



- OR gate



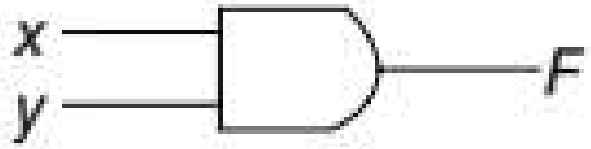
- NOT gate



+	-
HIGH	LOW
YES	NO
TRUE	FALSE
1	0

Logic Operations and Logic Gates

AND

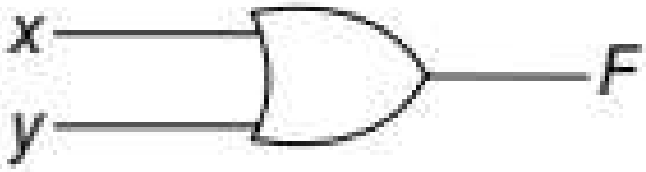


$F = xy$

x	y	F
0	0	0
0	1	0
1	0	0
1	1	1

Logic Operations and Logic Gates

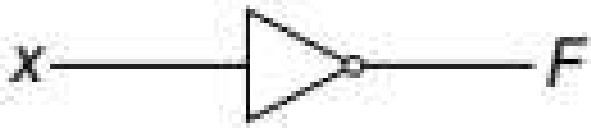
OR



$F = x + y$

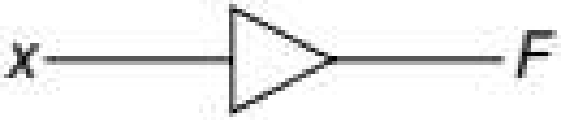
x	y	F
0	0	0
0	1	1
1	0	1
1	1	1

Logic Operations and Logic Gates

Inverter 

$F = x'$

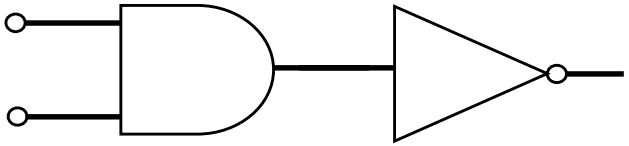
x	F
0	1
1	0

Buffer 

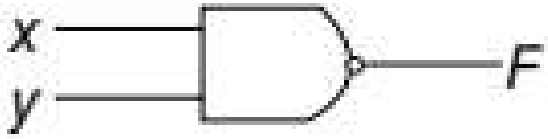
$F = x$

x	F
0	0
1	1

Logic Operations and Logic Gates



NAND

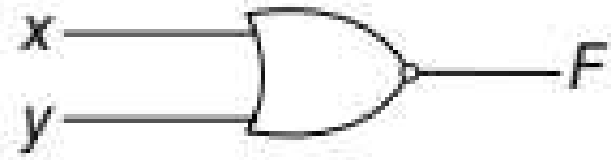


$F = (xy)'$

x	y	F
0	0	1
0	1	1
1	0	1
1	1	0

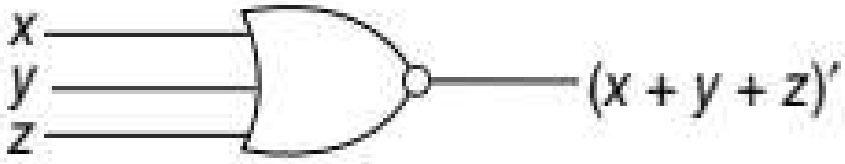
Logic Operations and Logic Gates

NOR



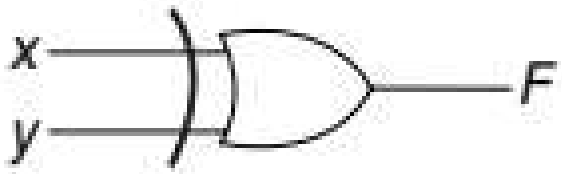
$F = (x + y)'$

x	y	F
0	0	1
0	1	0
1	0	0
1	1	0



Logic Operations and Logic Gates

Exclusive-OR
(XOR)

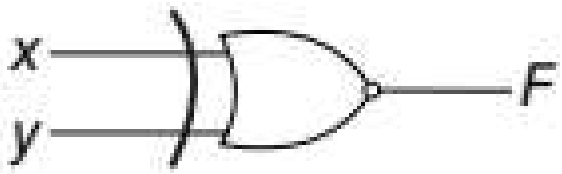


$F = xy' + x'y$
$= x \oplus y$

x	y	F
0	0	0
0	1	1
1	0	1
1	1	0

Logic Operations and Logic Gates

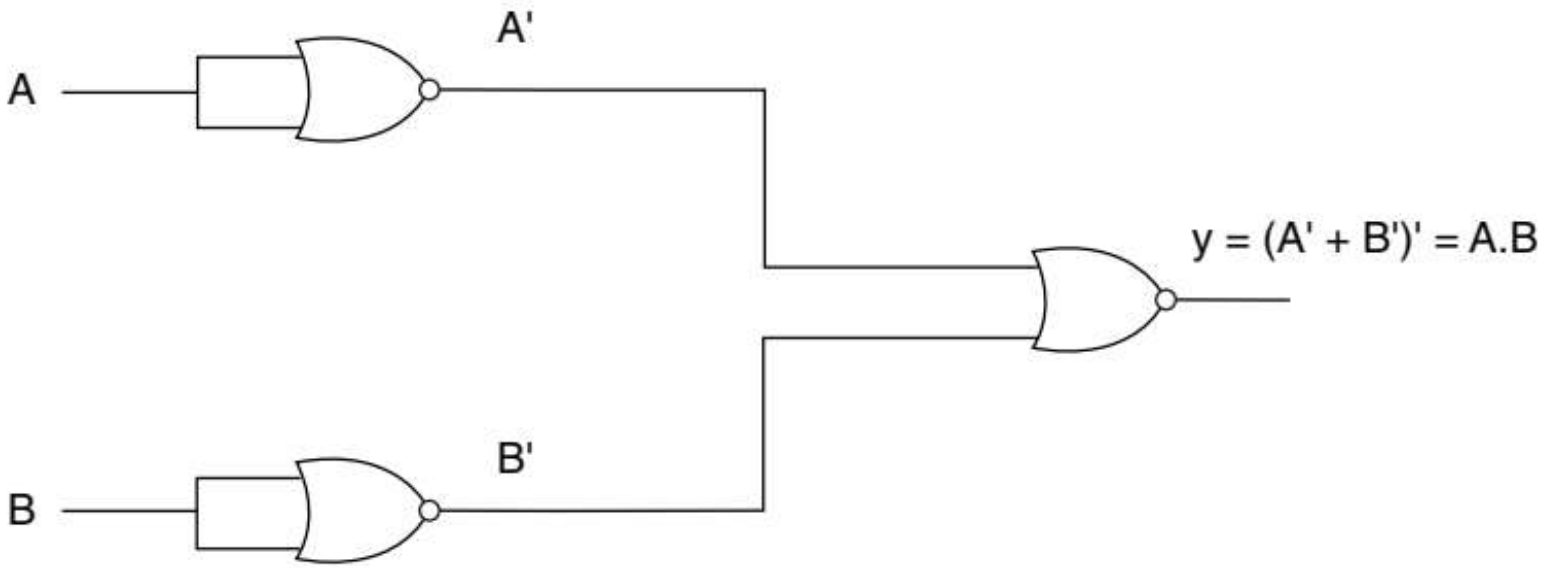
Exclusive-NOR
or
equivalence



$F = xy + x'y'$
$= x \odot y$

x	y	F
0	0	1
0	1	0
1	0	0
1	1	1

Logic Operations and Logic Gates



Basic Postulates and Theorems

Postulate 2	(a) $x + 0 = x$	(b) $x \cdot 1 = x$
Postulate 5	(a) $x + x' = 1$	(b) $x \cdot x' = 0$
Theorem 1	(a) $x + x = x$	(b) $x \cdot x = x$
Theorem 2	(a) $x + 1 = 1$	(b) $x \cdot 0 = 0$
Theorem 3, involution	$(x')' = x$	
Postulate 3, commutative	(a) $x + y = y + x$	(b) $xy = yx$
Theorem 4, associative	(a) $x + (y + z) = (x + y) + z$	(b) $x(yz) = (xy)z$
Postulate 4, distributive	(a) $x(y + z) = xy + xz$	(b) $x + yz = (x + y)(x + z)$
Theorem 5, DeMorgan	(a) $(x + y)' = x'y'$	(b) $(xy)' = x' + y'$
Theorem 6, absorption	(a) $x + xy = x$	(b) $x(x + y) = x$

Examples

$x\,(x' + y)$

$x'\,y'\,z + x'\,yz + xy'$

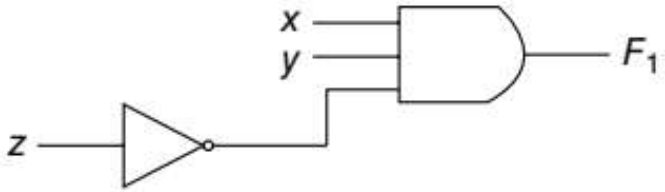
$xy + x'\,z + yz = xy$

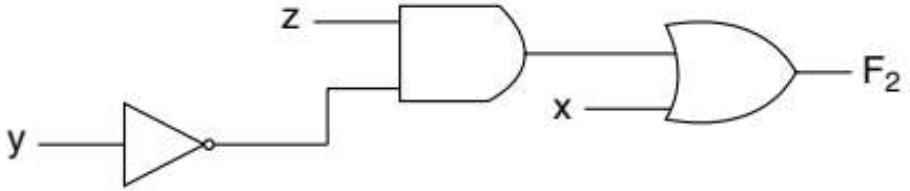
$(x'\,yz' + x'\,y'\,z)'$

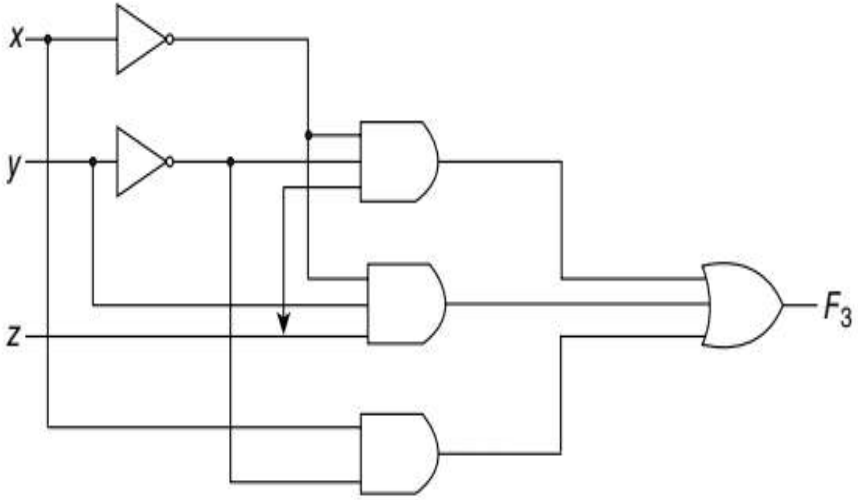
$[x\,(y'\,z' + yz)]'$

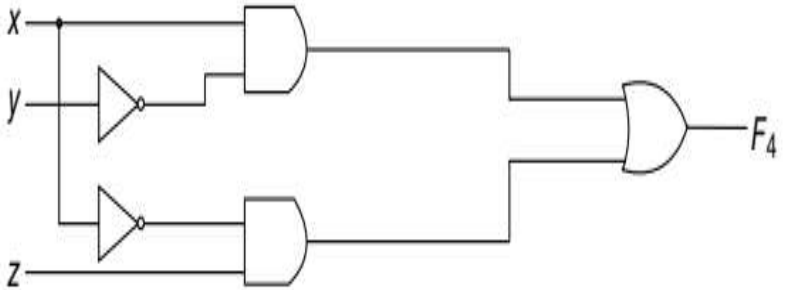
Logic Operations and Logic Gates

Examples



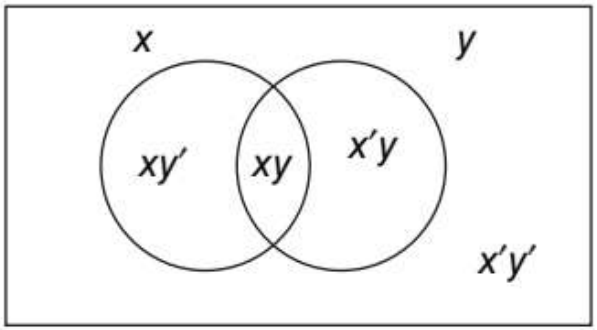


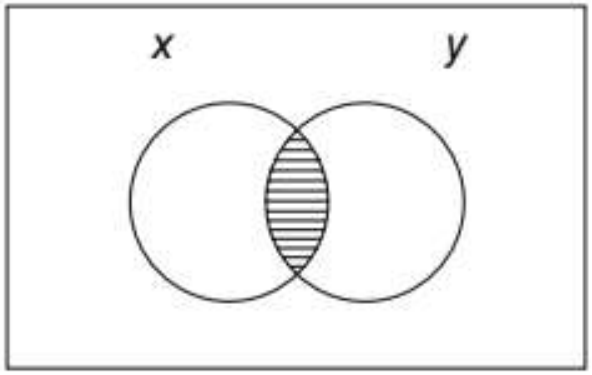


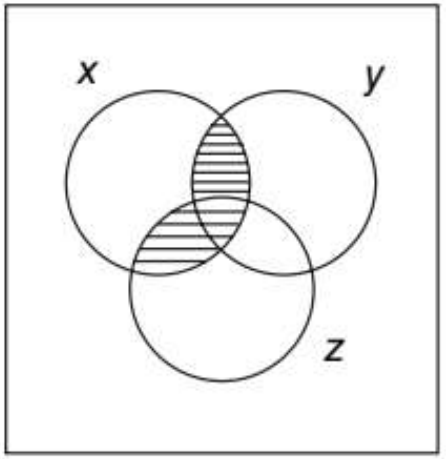


Logic Operations and Logic Gates

Venn Diagram







Canonical and Standard Forms

Minterms and Maxterms

			Minterms		Maxterms	
x	y	z	Term	Designation	Term	Designation
0	0	0	$x'y'z'$	m_0	$x+y+z$	M_0
0	0	1	$x'y'z$	m_1	$x+y+z'$	M_1
0	1	0	$x'yz'$	m_2	$x+y'+z$	M_2
0	1	1	$x'yz$	m_3	$x+y'+z'$	M_3
1	0	0	$xy'z'$	m_4	$x'+y+z$	M_4
1	0	1	$xy'z$	m_5	$x'+y+z'$	M_5
1	1	0	xyz'	m_6	$x'+y'+z$	M_6
1	1	1	xyz	m_7	$x'+y'+z'$	M_7

Canonical and Standard Forms

x	y	z	F_1	F_2	F_3	F_4
0	0	0	0	0	0	0
0	0	1	0	1	1	1
0	1	0	0	0	0	0
0	1	1	0	0	1	1
1	0	0	0	1	1	1
1	0	1	0	1	1	1
1	1	0	1	1	0	0
1	1	1	0	1	0	0

Canonical and Standard Forms

x	y	z	Function f_1	Function f_2
0	0	0	0	0
0	0	1	1	0
0	1	0	0	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	I	0	0	1
1	1	1	1	1

$$f_1 = (x+y+z)(x+y'+z)(x+y'+z')(x'+y+z')(x'+y'+z)$$
$$= M_0 \cdot M_2 \cdot M_3 \cdot M_5 \cdot M_6$$

$$f_2 = (x+y+z)(x+y+z')(x+y'+z)(x'+y+z) = M_0 M_1 M_2 M_4$$

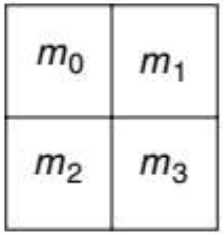
Canonical and Standard Forms

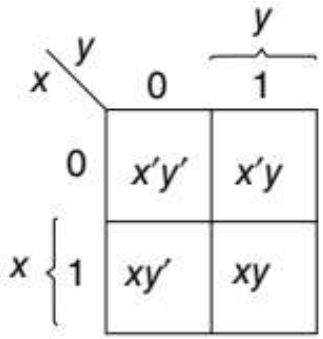
$$F = A + B'C$$

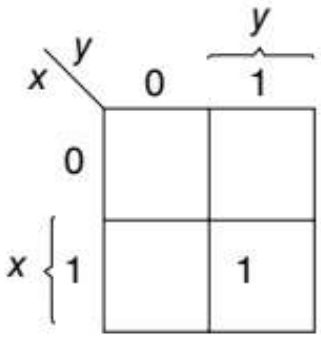
$$F = xy + x'z$$

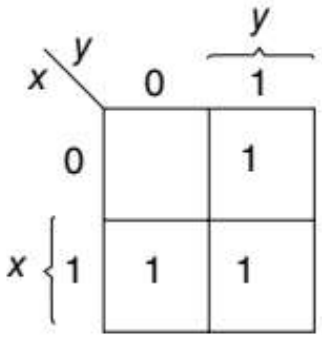
Simplification of Boolean Functions

K-Map





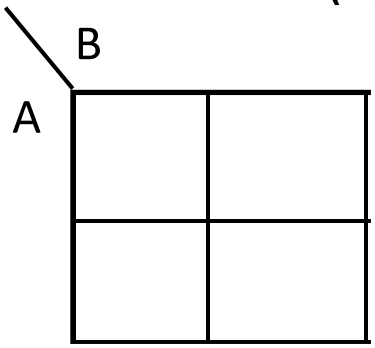




Simplification of Boolean Functions

K-Map

Fill the function Σm(0,1,3) in K- Map and simplify



1. Fill the 1s.
2. Form the largest rectangles of 1s
3. Check if any of the rectangles is not required. (largest should be selected)
4. Write the result.

Rectangle: Group of Squares of power 2, ie., 1, 2, 4, 8 etc.

Logic Operations and Logic Gates

Simplification of Boolean Functions

K-Map

F=A’B+AB’

Simplification of Boolean Functions

K-Map

m_0	m_1	m_3	m_2
m_4	m_5	m_7	m_6

x \ yz	00	01	11	10
0	$x'y'z'$	$x'y'z$	$x'yz$	$x'yz'$
1	$xy'z'$	$xy'z$	xyz	xyz'

$F = x'yz + x'yz' + xy'z' + xy'z$

x \ yz	00	01	11	10
0			1	1
1	1	1		

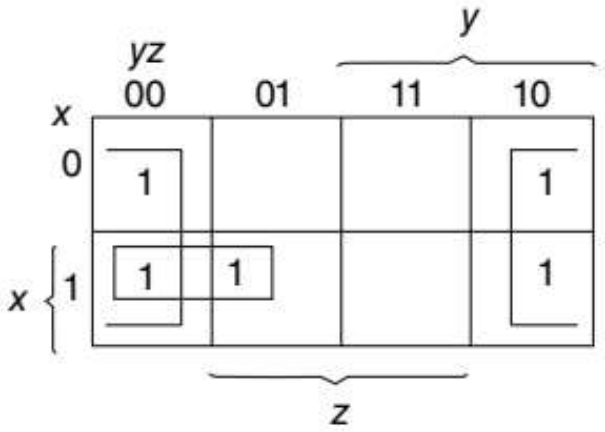
$F = x'yz + xy'z' + xyz + xyz'$

x \ yz	00	01	11	10
0			1	
1	1		1	1

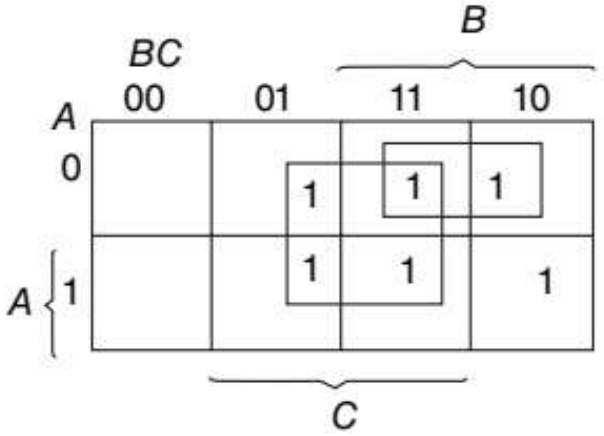
Simplification of Boolean Functions

K-Map

$F(x,y,z) = \sum(0, 2, 4, 5, 6)$



$F = A'C + A'B + AB'C + BC$



Simplification of Boolean Functions

K-Map

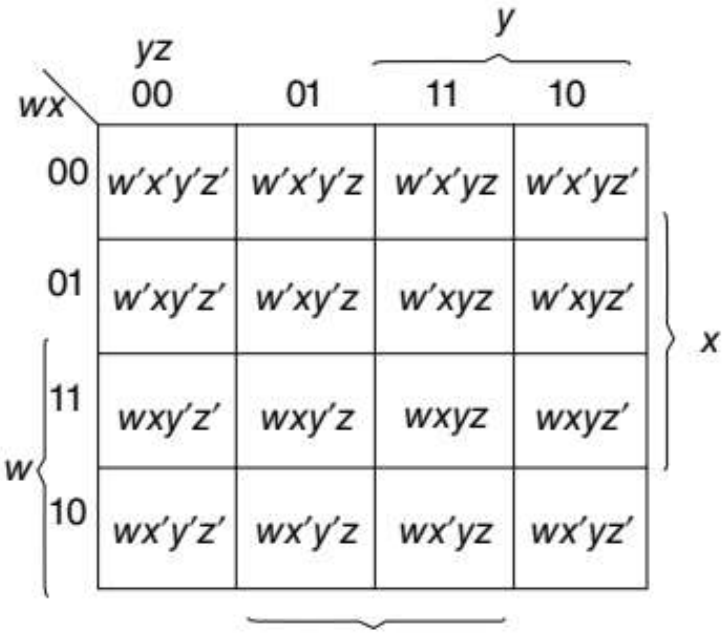
Simplify Σm(0,1,2,3,4,5) using K- Map

Simplify Σm (0,2,4,5,6) using K- Map

Simplification of Boolean Functions

K-Map

m_0	m_1	m_3	m_2
m_4	m_5	m_7	m_6
m_{12}	m_{13}	m_{15}	m_{14}
m_8	m_9	m_{11}	m_{10}

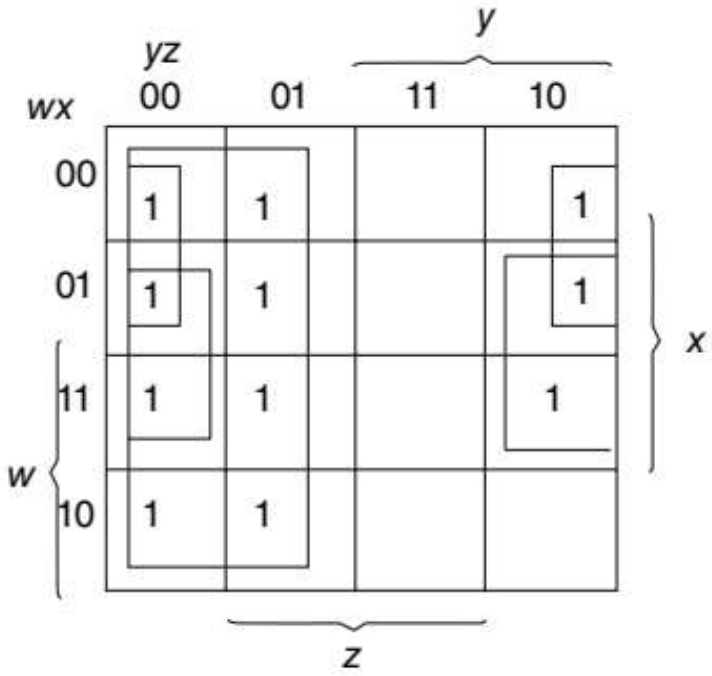


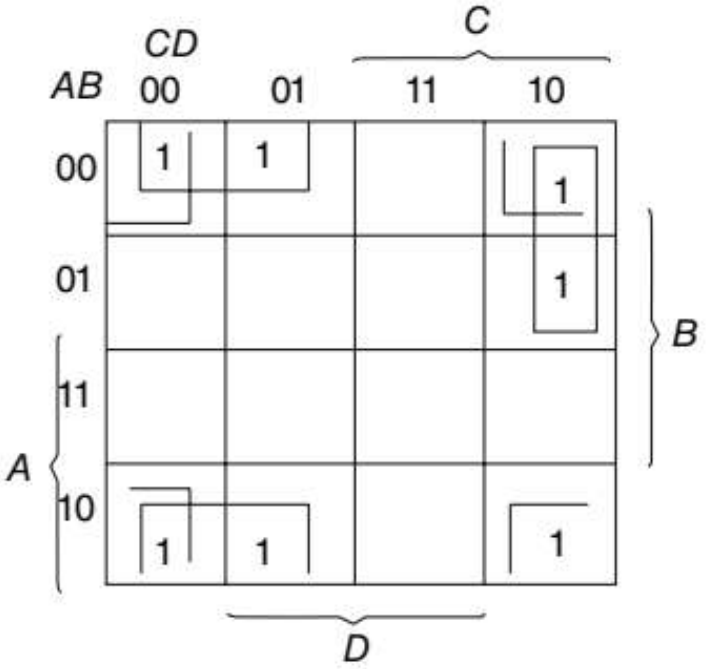
Simplification of Boolean Functions

K-Map

$F(w,x,y,z) = \Sigma(0, 1, 2, 4, 5, 6, 8, 9, 12, 13, 14)$

$F = A'B'C' + B'CD' + A'BCD' + AB'C'$





Simplification of Boolean Functions

K-Map

Simplify Σm (0,1,2,4,5,6,8,9,10,12,13) using K- Map

Simplification of Boolean Functions

Don't care conditions

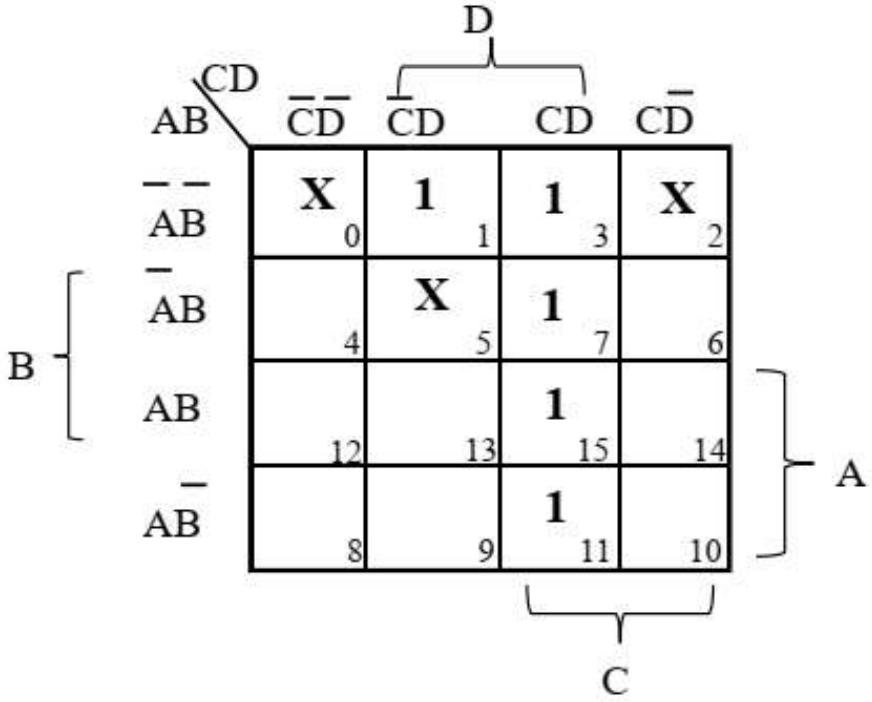
- Function is not specified in certain variable value combinations.
  - The input combinations never occur (e.g. BCD)
  - We do not care what the output is response to these inputs.
- Outputs are unspecified for the input combinations. Such functions are called incompletely specified functions.
- Unspecified minterms of the function: don’t care conditions
- Used to provide further simplification of the functions.
- Marked with a ‘X’ and be used as adjacent term for simplification.

Simplification of Boolean Functions

- $F(A,B,C,D)=\Sigma m(1,3,7,11,15)$
- $d(A,B,C,D)=\Sigma m(0,2,5)$

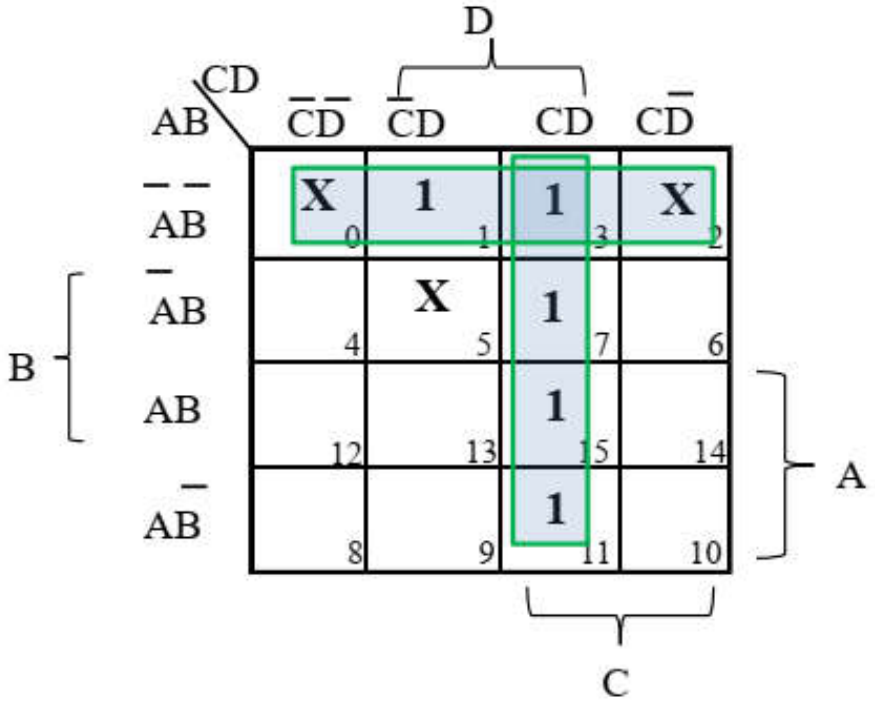
Simplification of Boolean Functions

- F(A,B,C,D)=Σm(1,3,7,11,15)
- d(A,B,C,D)=Σm(0,2,5)



Simplification of Boolean Functions

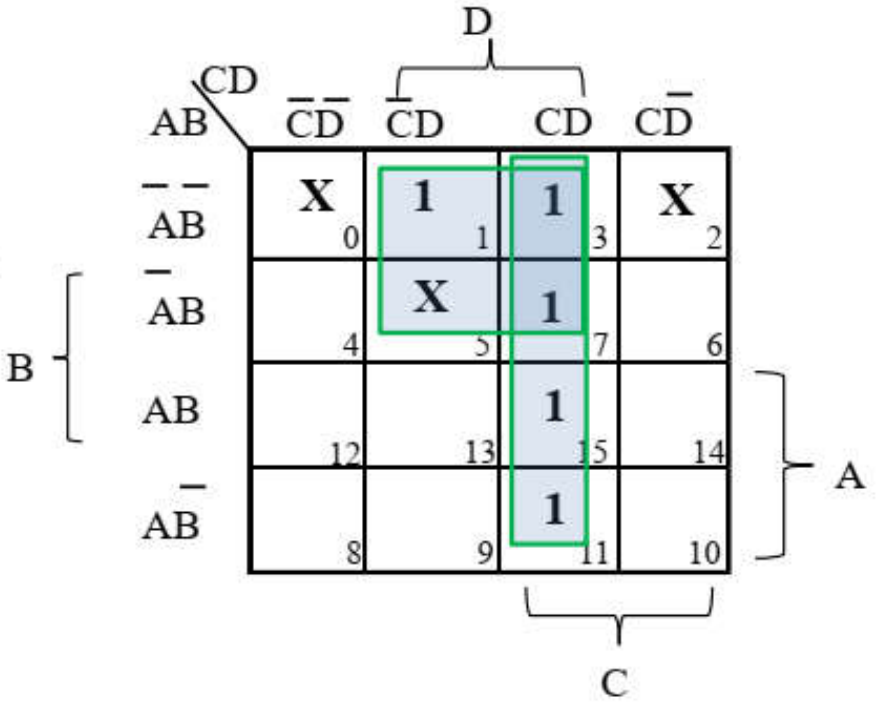
- F(A,B,C,D)=Σm(1,3,7,11,15)
- d(A,B,C,D)=Σm(0,2,5)



Simplification of Boolean Functions

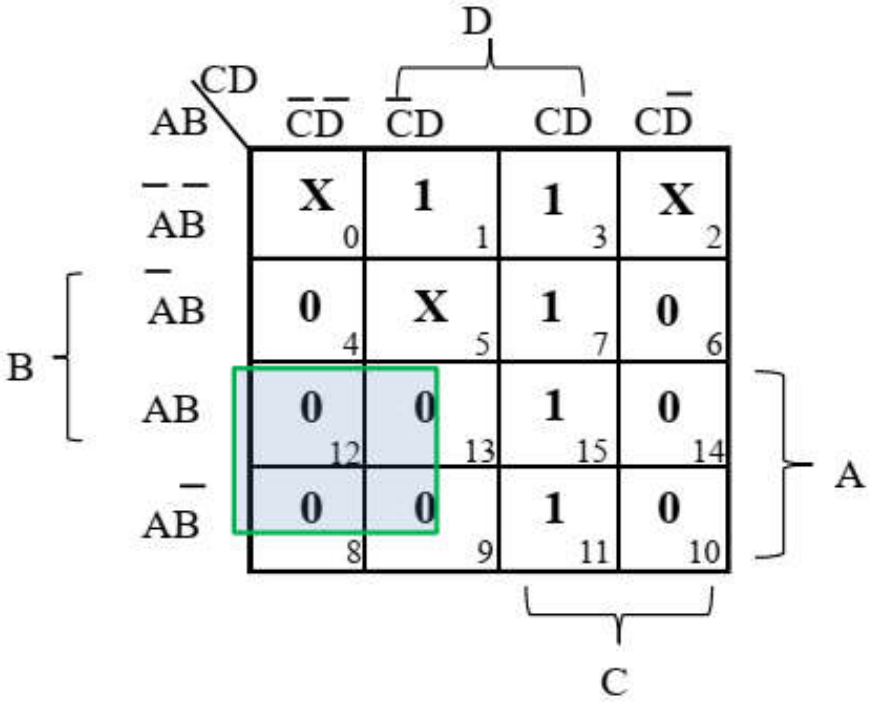
- F(A,B,C,D)=Σm(1,3,7,11,15)
- d(A,B,C,D)=Σm(0,2,5)

Incompletely specified function can be implemented using algebraically different expressions.



Problem (POS)

- $F(A,B,C,D)=\Sigma m(1,3,7,11,15)$
- $d(A,B,C,D)=\Sigma m(0,2,5)$



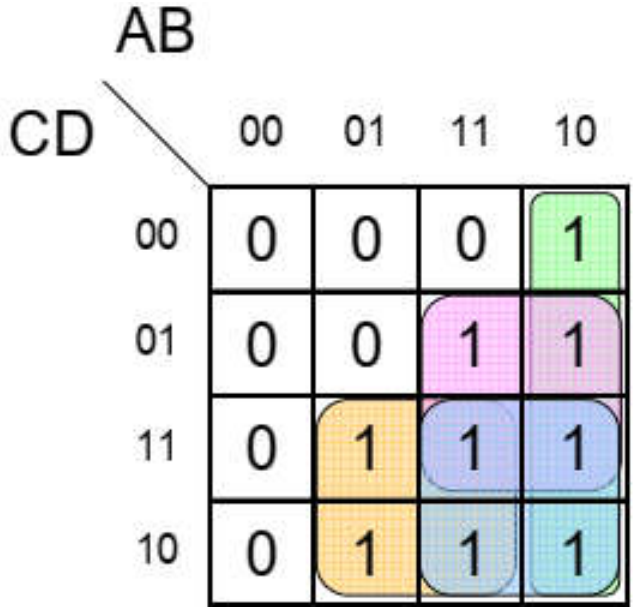
Logic Operations and Logic Gates

Simplification of Boolean Functions

Prime Implicants

- Can be obtained from the map
  - A single '1' on a map is a prime implicant if there is no adjacent square
  - Two adjacent 1s form a prime implicant if it is not part of rectangle containing 4 or more 1s.
  - 4 1s form a rectangle representing prime implicant if it is not part of a bigger rectangle containing 8 or more squares of 1s.
- Each essential prime implicant contains atleast one square of 1 which is not contained in any other prime implicant.

Quine-McClukey tabular method



Quine-McClukey tabular method

$F = \sum(0, 1, 2, 8, 10, 11, 14, 15)$

(a)

	$w\,x\,y\,z$
0	0 0 0 0 √
1	0 0 0 1 √
2	0 0 1 0 √
8	1 0 0 0 √
10	1 0 1 0 √
11	1 0 1 1 √
14	1 1 1 0 √
15	1 1 1 1 √

(b)

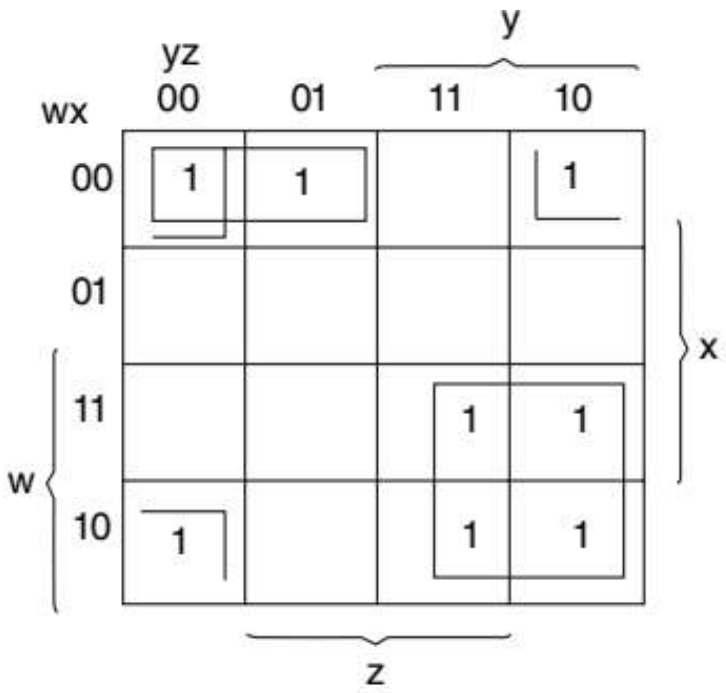
	$w\,x\,y\,z$
0, 1	0 0 0 −
0, 2	0 0 − 0 √
0, 8	− 0 0 0 √
2, 10	− 0 1 0 √
8, 10	1 0 − 0 √
10, 11	1 0 1 − √
10, 14	1 − 1 0 √
11, 15	1 − 1 1 √
14, 15	1 1 1 − √

(c)

	$w\,x\,y\,z$
0, 2, 8, 10	− 0 − 0
0, 8, 2, 10	− 0 − 0
10, 11, 14, 15	1 − 1 −
10, 14, 11, 15	1 − 1 −

$F = w'\,x'\,y' + x'\,z' + w\,y$

Quine-McClukey tabular method



$$F = w' x' y' + x' z' + w y$$

Quine-McClukey tabular method

(a)		(b)		(c)
0	√	0, 1 (1)		0, 2, 8, 10 (2, 8)
		0, 2 (2)	√	0, 2, 8, 10 (2, 8)
1	√	0, 8 (8)	√	
2	√	2, 10 (8)	√	10, 11, 14, 15 (1, 4)
8	√	8, 10 (2)	√	10, 11, 14, 15 (1, 4)
		10, 11 (1)	√	
10	√			
11	√	10, 14 (4)	√	
14	√			
15	√	11, 15 (4)	√	
		14, 15 (1)	√	

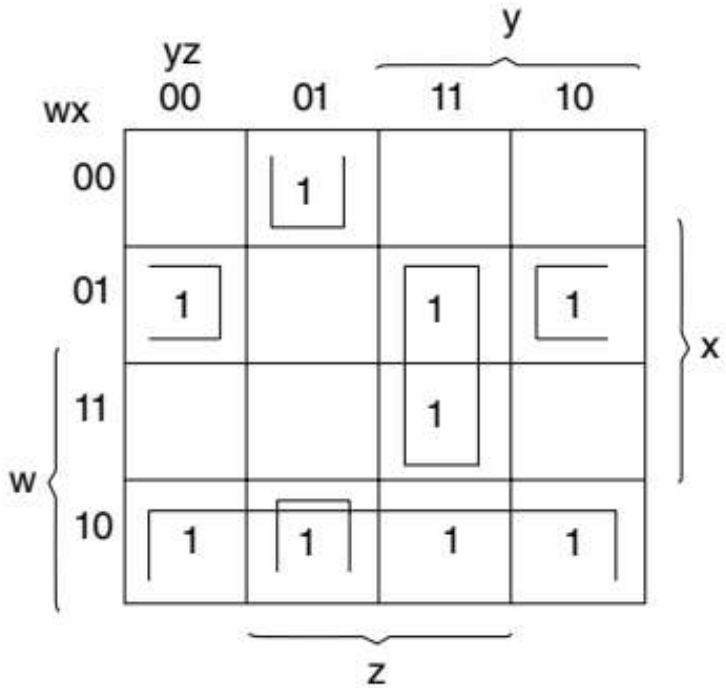
Quine-McClukey tabular method

(a)			(b)			(c)
0001	1	√	1, 9	(8)		8, 9, 10, 11 (1, 2)
0100	4	√	4, 6	(2)		8, 9, 10, 11 (1, 2)
1000	8	√	8, 9	(1)	√	
			8, 10	(2)	√	
0110	6	√				
1001	9	√	6, 7	(1)		
1010	10	√	9, 11	(2)	√	
			10, 11	(1)	√	
0111	7	√				
1011	11	√	7, 15	(8)		
			11,15	(4)		
1111	15	√				

Quine-McClukey tabular method

Prime-implicants		
Decimal	Binary $w\ x\ y\ z$	Term
1, 9 (8)	− 0 0 1	$x'\ y'\ z$
4, 6 (2)	0 1 − 0	$w'xz'$
6, 7(1)	0 1 1 −	$w'xy$
7, 15 (8)	− 1 1 1	xyz
11, 15(4)	1 − 1 1	wyz
8, 9, 10, 11 (1, 2)	1 0 − −	wx'

Quine-McClukey tabular method



$$F = x' y' z + w' x z' + x y z + w x'$$

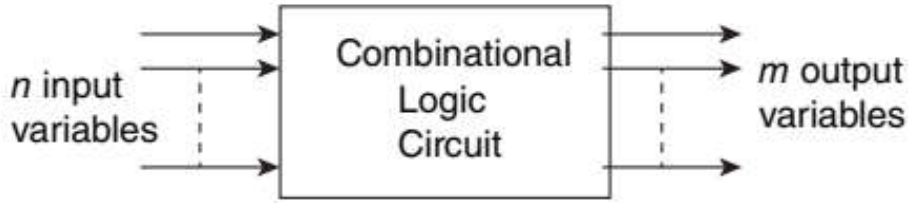
Quine-McClukey tabular method

		1	4	6	7	8	9	10	11	15
√ $x'\,y'\,z$	1, 9	X					X			
√ $w'\,x\,z'$	4, 6		X	X						
$w'\,x\,y$	6, 7			X	X					
$x\,y\,z$	7, 15				X					X
$w\,y\,z$	11, 15								X	X
√ $w\,x'$	8, 9, 10, 11					X	X	X	X	
		√	√	√		√	√	√	√	

$$F = x'\,y'\,z + w'\,x\,z' + w\,x' + x\,y\,z$$

Digital Logic Circuits

COMBINATIONAL CIRCUITS	SEQUENTIAL CIRCUITS
Output depends only on the present value of the inputs.	Output depends on both the present and previous state values of the inputs
These circuits will not have any memory as their outputs change with the change in the input value.	Sequential circuits have some sort of memory as their output changes according to the previous and present values.
There are no feedbacks involved.	In a sequential circuit the outputs are connected to it as a feedback path.
Used in basic Boolean operations.	Used in the designing of memory devices.
Implemented in: Half adder circuit, full adder circuit, multiplexers, de-multiplexers, decoders and encoders.	Implemented in: RAM, Registers, counters and other state retaining machines.



Design Process

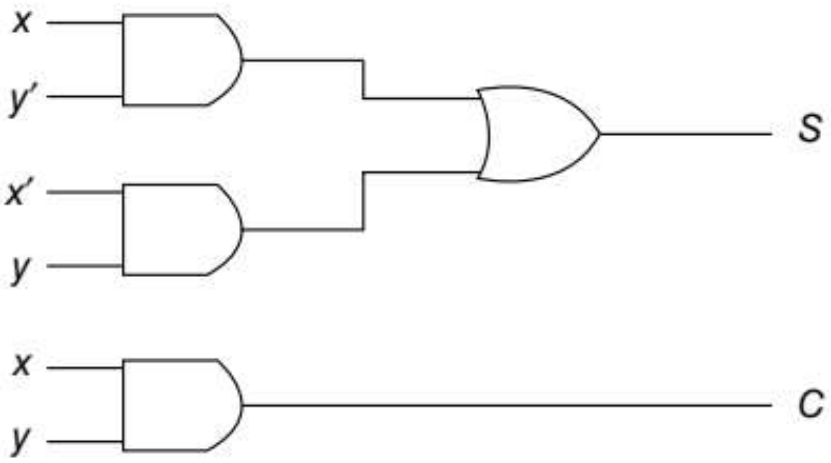
1. The problem is stated.
2. The number of available input variables and required output variables is determined.
3. The input and output variables are assigned letter symbols.
4. The truth table that defines the required relationships between inputs and outputs is derived.
5. The simplified Boolean function for each output is obtained.
6. The logic diagram is drawn.

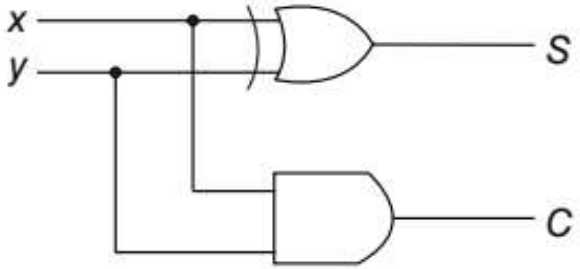
Adders

Half-Adder

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

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

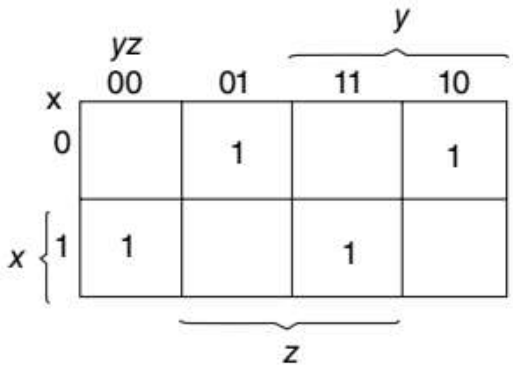




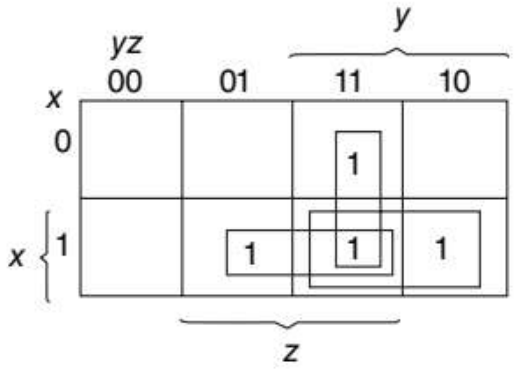
(e) $S = x \oplus y$
$C = xy$

Full-Adder

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



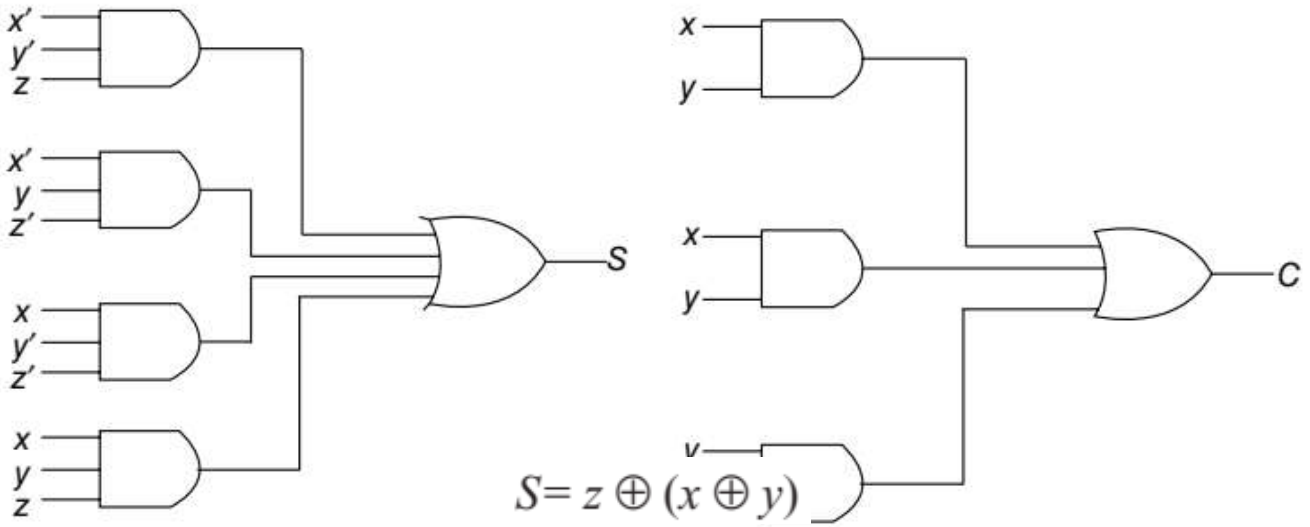
$S = x'y'z + x'yz' + xy'z' + xyz$



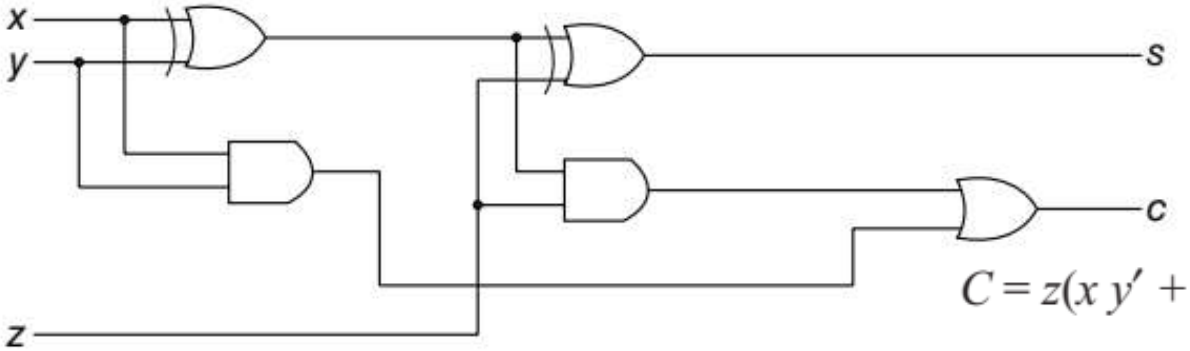
$C = xy + xz + yz$

$$S = z \oplus (x \oplus y)$$
$$= z'(x\,y' + x'y) + z(x\,y' + x'y)'$$
$$= z'(x\,y' + x'y) + z(x\,y + x'y')$$
$$= x\,y'z' + x'yz' + x\,yz + x'y'z$$

$$C = z(x\,y' + x'y) + x\,y = x\,y'z + x'yz + x\,y$$



$S = z \oplus (x \oplus y)$



$S = z \oplus (x \oplus y)$

$C = z(x\,y' + x'y) + x\,y = x\,y'z + x'yz + x\,y$

Subtractors

Half-Subtractor

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

$$D = x'y + x\,y'$$

$$B = x'y$$

Full-Subtractor

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

$$D = x'y'z + x'yz' + x\,y'z' + x\,yz$$

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