IT 2203
Combinational Logic

Dr. Md. Sazzadur Rahman
Professor
Institute of Information Technology
Jahangirnagar University, Savar, Dhaka.

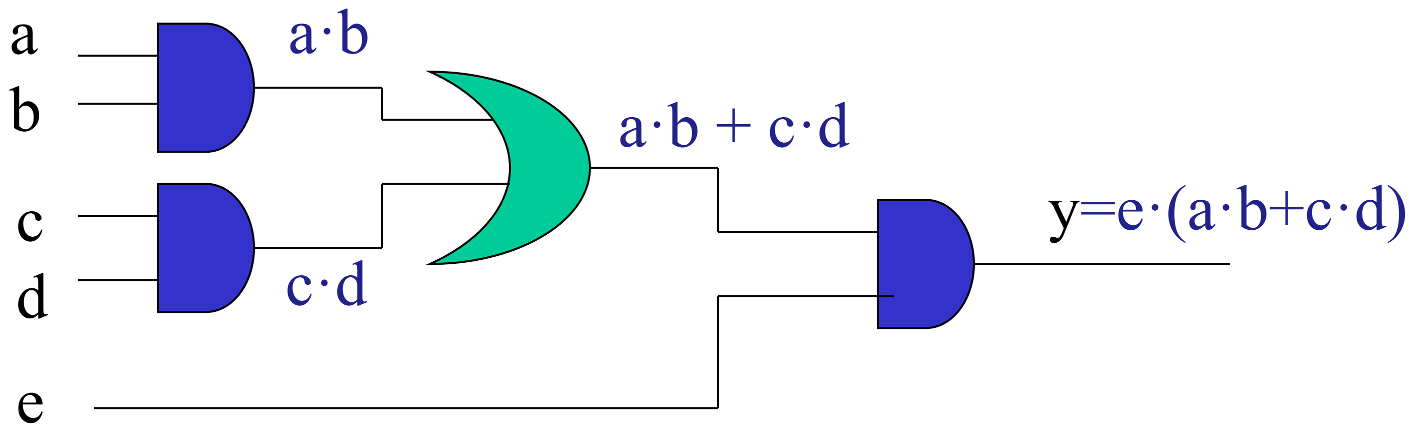
Outlines

- Introduction
  - Motivation
  - Review of Boolean Algebra
  - Transformation of Logic Gates
- Specification
- Synthesis

Motivation: Combinational Logic

- Description
  - Language
  - Boolean algebra
  - Truth table
- Schematic Diagram
  - Inputs, Gates, Nets, Outputs
- Goal
  - Validity: correctness, turnaround time
  - Performance: power, timing, cost
  - Testability: yield, diagnosis, robustness

Motivation: Combinational Logic vs. Boolean Algebra Expression



Schematic Diagram:

5 primary inputs, 1 primary output

4 components (gates)

9 signal nets

12 pins

Boolean Algebra:

5 literals

4 operators

Obj: min #terms
min #literals

Motivation: iClicker

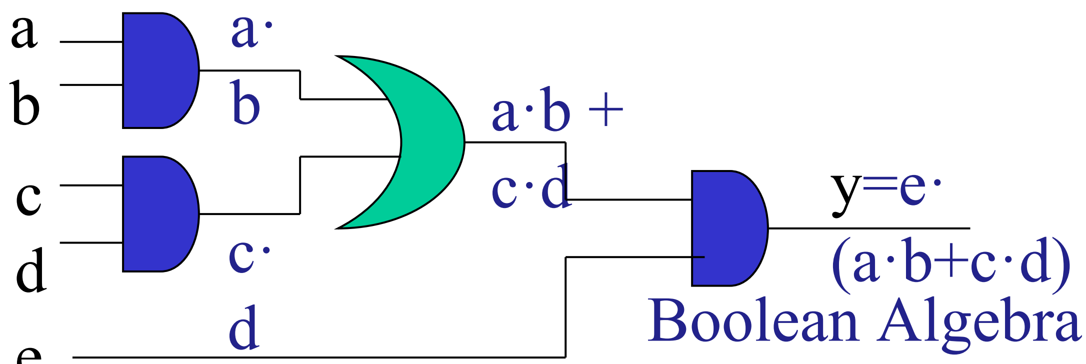
Schematic Diagram:

5 primary inputs

4 components (gates)

9 signal nets

12 pins



Boolean Algebra:

5 literals

4 operators

A. #inputs
B. #gates
C. #nets
D. #pins
E. None

I. #operators
II. #literals + #operators
III. #literals + 2 #operators

Schematic Diagram vs. Boolean Expression

- Boolean Expression: #literals, #operators
- Schematic Diagram: #gates, #nets, #pins

- One more example?

Some Definitions

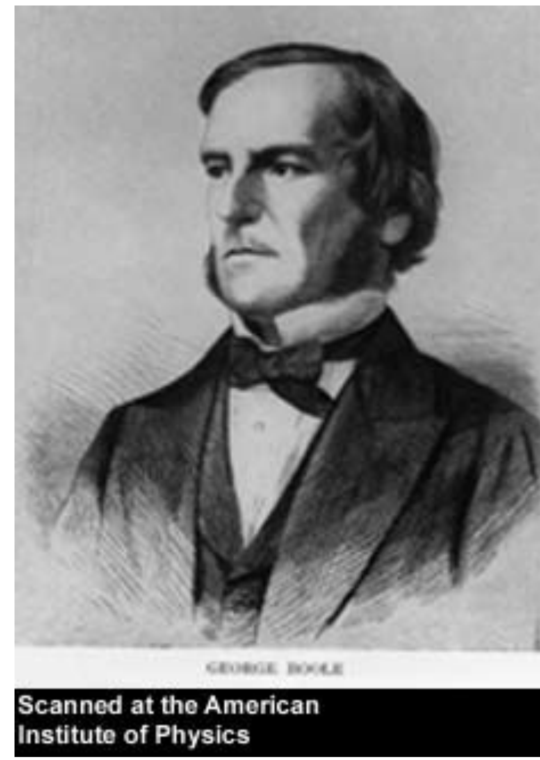
- Complement: variable with a bar over it
  $\bar{A}, \bar{B}, \bar{C}$
- Literal: variable or its complement
  $A, \bar{A}, B, \bar{B}, C, \bar{C}$
- Implicant: product of literals
  $A\bar{B}C, A\bar{C}, BC$
- Minterm: product that includes all input variables
  $\bar{A}BC, AB\bar{C}, A\bar{B}C$
- Maxterm: sum that includes all input variables
  $(A+\bar{B}+C), (\bar{A}+B+C), (\bar{A}+\bar{B}+\bar{C})$

Digital Discipline: Binary Values

- Typically consider only two discrete values:
  - 1’s and 0’s
  - 1, TRUE, HIGH
  - 0, FALSE, LOW
- 1 and 0 can be represented by specific voltage levels, rotating gears, fluid levels, etc.
- Digital circuits usually depend on specific voltage levels to represent 1 and 0
- *Bit*: *B*inary dig*it*

George Boole, 1815 - 1864

- Born to working class parents
- Taught himself mathematics and joined the faculty of Queen's College in Ireland.
- Wrote *An Investigation of the Laws of Thought* (1854)
- Introduced binary variables
- Introduced the three fundamental logic operations: AND, OR, and NOT.



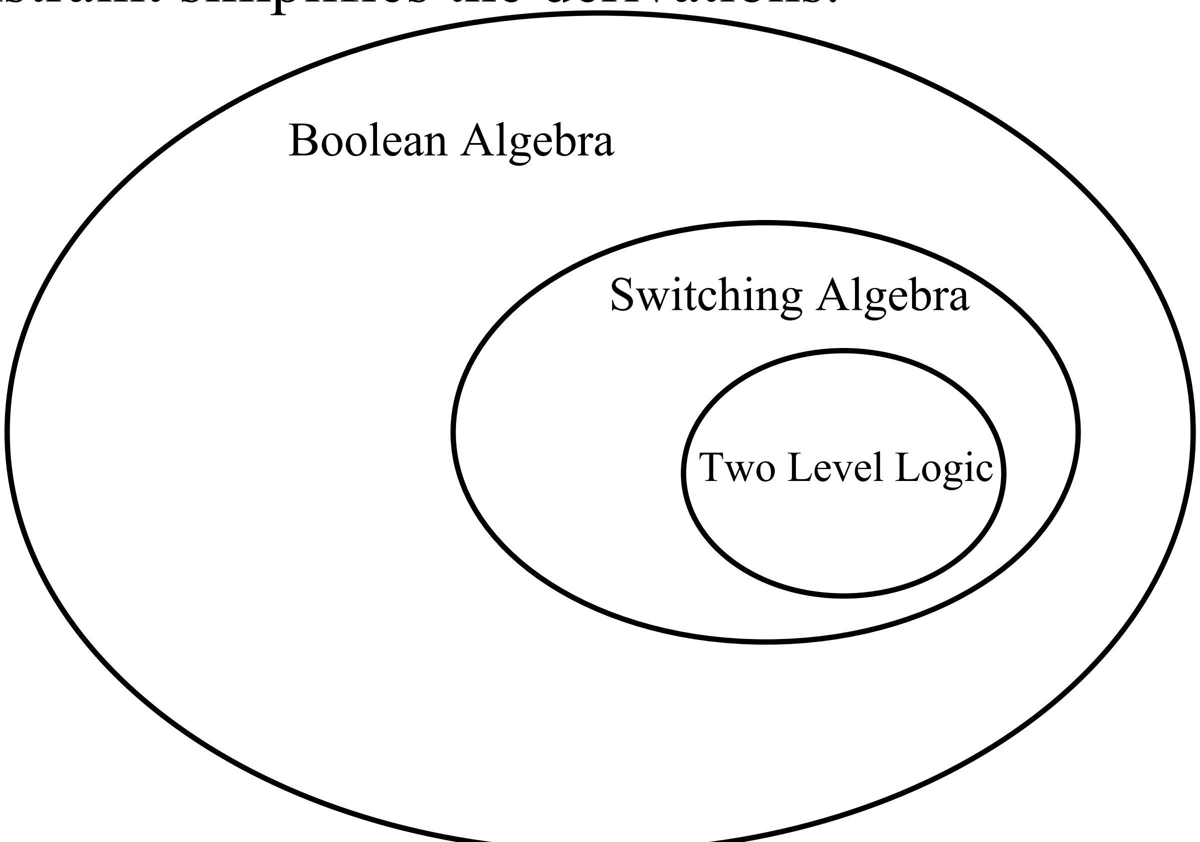
Review of Boolean Algebra

Let B be a nonempty set with two binary operations, a unary operation `, and two distinct elements 0 and 1. Then B is called a Boolean algebra if the following axioms hold.

- Commutative laws: $a+b=b+a$, $a\cdot b=b\cdot a$
- Distributive laws: $a+(b\cdot c)=(a+b)\cdot(a+c)$, $a\cdot(b+c)=a\cdot b+a\cdot c$
- Identity laws: $a+0=a$, $a\cdot 1=a$
- Complement laws: $a+a'=1$, $a\cdot a'=0$

Switching Algebra (A subset of Boolean Algebra)

- Boolean Algebra: Each variable may have multiple values.
- Switching Algebra: Each variable can be either 1 or 0. The constraint simplifies the derivations.



Switching Algebra

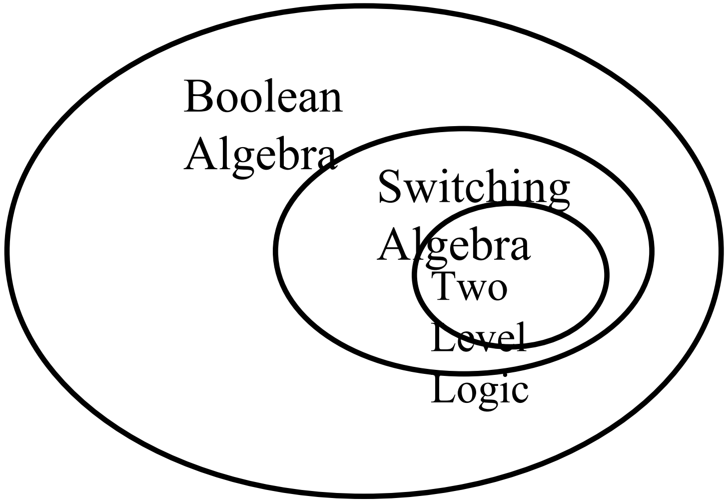
- Two Level Logic: Sum of products, or product of sums, e.g. ab + a’c + a’b’, (a’+c)(a+b’)(a+b+c’)
- Multiple Level Logic: Many layers of two level logic with some inverters, e.g. (((a+bc)’+ab’)+b’c+c’d)’bc+c’e

Features of Digital Logic Design

- Multiple Outputs
- Don’t care sets

Handy Tools:

- DeMorgan’s Law: Complements
- Consensus Theorem
- Shannon’s Expansion
- Truth Table
- Karnaugh Map (single output, two level logic)



Review of Boolean algebra and switching functions

AND, OR, NOT

AND

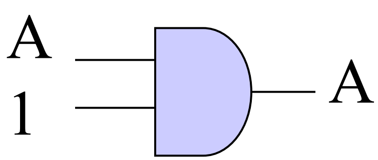
A	B	Y
0	0	0
0	1	0
1	0	0
1	1	1

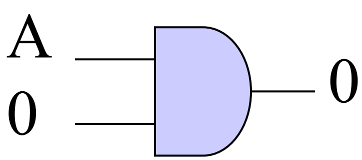
OR

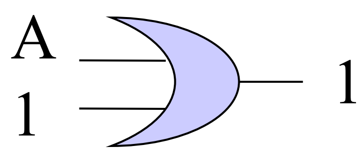
A	B	Y
0	0	0
0	1	1
1	0	1
1	1	1

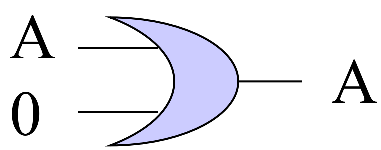
NOT

A	Y
0	1
1	0









0 dominates in AND
0 blocks the output
1 passes signal A

1 dominates in OR
1 blocks the output
0 passes signal A

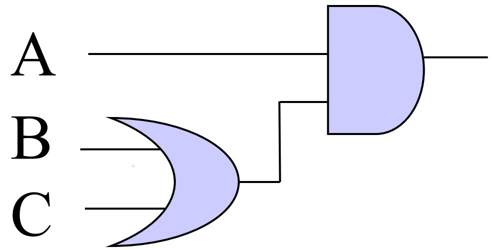
1. Identity $A * 1 = A \quad A + 1 = 1$

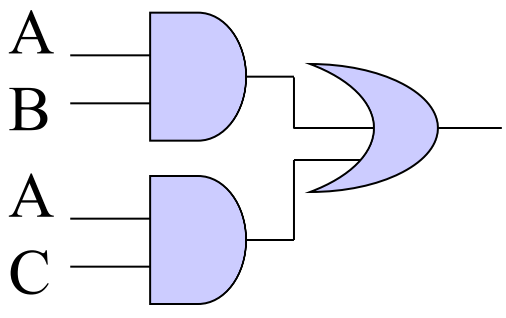
$A * 0 = 0 \quad A + 0 = A$

2. Complement $A + A' = 1 \quad A * A' = 0$

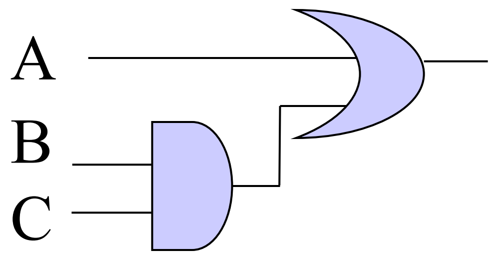
T8. Distributive Law

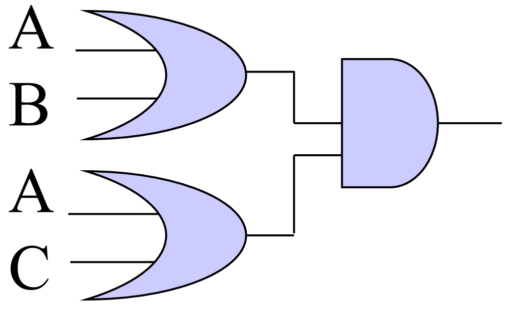
$A(B+C) = AB + AC$





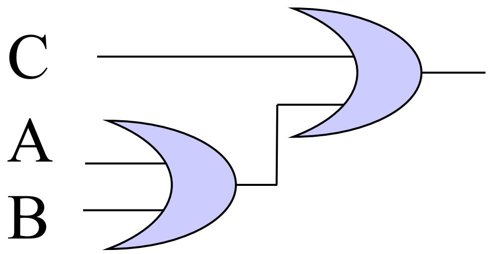
$A+BC = (A+B)(A+C)$

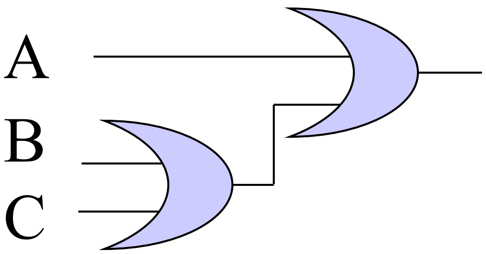




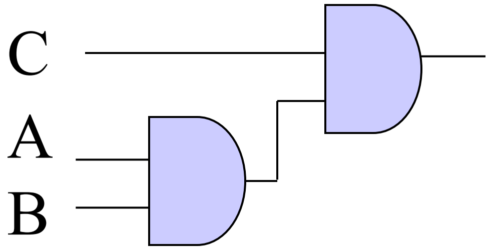
T7. Associativity

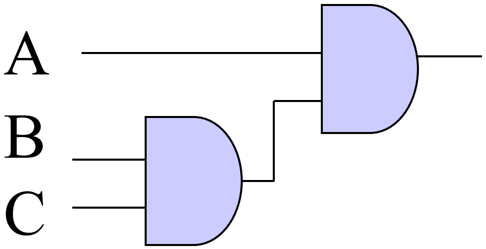
(A+B) + C = A + (B+C)





(AB)C = A(BC)



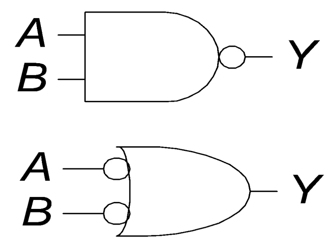


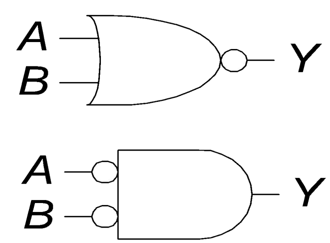
T12. DeMorgan’s Law (A+B)’ = A’B’ (AB)’ = A’ + B’

T11. Consensus Theorem AC + AB + BC’ = AC + BC’

DeMorgan’s Theorem

- $Y = \overline{AB} = \overline{A} + \overline{B}$

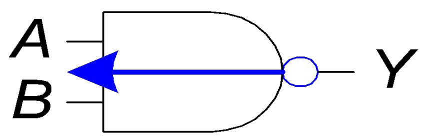


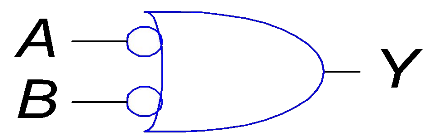


- $Y = \overline{A + B} = \overline{A} \bullet \overline{B}$

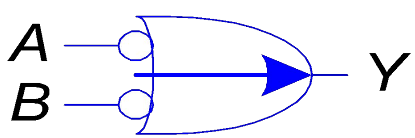
Bubble Pushing

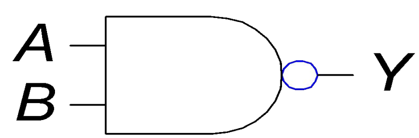
- Pushing bubbles backward (from the output) or forward (from the inputs) changes the body of the gate from AND to OR or vice versa.
- Pushing a bubble from the output back to the inputs puts bubbles on all gate inputs.





- Pushing bubbles on *all* gate inputs forward toward the output puts a bubble on the output and changes the gate body.





Consensus Theorem

- $AB+AC+B'C$

  $=AB+B'C$

- $(A+B)(A+C)(B'+C)$

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

Exercise: to prove the reduction using

(1)Boolean algebra,

(2)Logic simulation and

(3) Shannon's expansion

Consensus Theorem: iClicker

Which term in AB’+AC+BC can be deleted?

A. AB’

B. AC

C. BC

D. None of the above

Shannon's Expansion

- Shannon's expansion assumes a switching algebra system
- Divide a switching function into smaller functions
- Pick a variable x, partition the switching function into two cases: x=0 and x=1
  - $f(x)=xf(1)+x'f(0)$
  - $f(x,y)=xf(1,y)+x'f(0,y)$
  - $f(x,y,z,\ldots)=xf(1,y,z,\ldots)+x'f(0,y,z,\ldots)$

Shannon's Expansion: iClicker

$f(x,y,z)=xf(?,y,z)+x'f(?,y,z)$

A. $?=0$

B. $?=1$.

$f(x,y)=(x+f(?,y))(x'+f(?,y))$

- A. $?=0$
- B. $?=1$.

Shannon’s Expansion

- Decompose the Switching function into minterms

$f(x, y) = xf(1, y) + x'f(0, y)$

$= x\big(yf(1,1) + y'(1,0)\big) + x'(yf(0,1) + y'f(0,0))$

$= xyf(1,1) + xy'f(0,1) + x'yf(1,0) + x'y'f(0,0)$

Likewise, we can decompose the function into maxterms

$f(x, y) = \big(x + f(0, y)\big)(x' + f(1, y))$

$= (x + \big(y + f(0,0)\big)(y' + f(0,1)))(x' + (y + f(1,0))(y' + f(1,1)))$

$= (x + y + f(0,0))(x + y' + f(0,1))(x' + y + f(1,0)(x' + y' + f(1,1))$

Switching Algebra

- Shannon’s expansion and consensus theorem are used for logic optimization
- Shannon’s expansion divides the problem into smaller functions
- Consensus theorem finds common terms when we merge small functions
- Karnaugh map mimics the above two operations in two dimensional space as visual aids.

Part I. Combinational Logic

I) Specification

a. Language

b. Truth Table

c. Boolean Algebra

Canonical Expression: Sum of minterms and
Product of maxterms

d. Incompletely Specified Function

Binary Addition

$$\begin{array}{rr} & 5 \\ + & 7 \\ \hline & 12 \end{array}$$

Carry Sum

$$\begin{array}{rrrrl} 1 & 1 & 1 & & \text{Carry bits} \\ & 1 & 0 & 1 & 5 \\ + & 1 & 1 & 1 & 7 \\ \hline 1 & 1 & 0 & 0 & 12 \end{array}$$

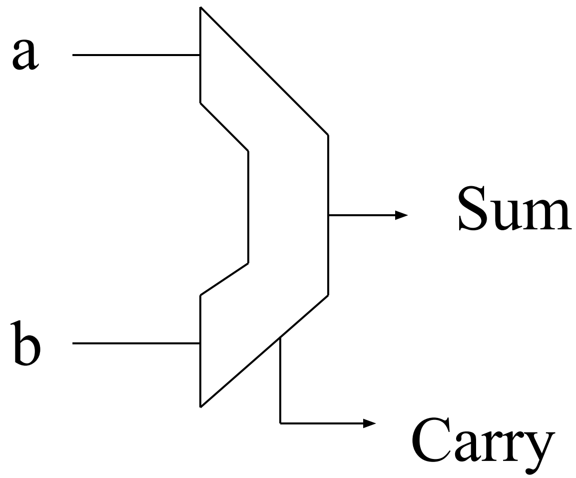
Carryout Sums

Binary Addition: Hardware

- Half Adder: Two inputs (a,b) and two outputs (carry, sum).

- Full Adder: Three inputs (a,b,c) and two outputs (carry, sum).

Half Adder



Truth Table

a	b	carry	sum
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

Switching Function

Switching Expressions:

$\text{Sum}(a,b) = a'\cdot b + a\cdot b'$

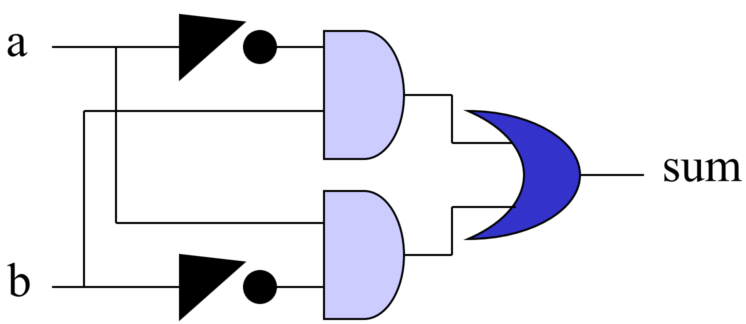
$\text{Carry}(a, b) = a\cdot b$

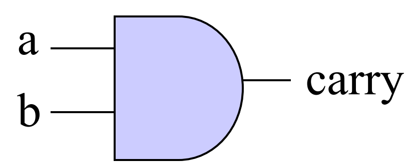
Ex:

$\text{Sum}(0,0) = 0'\cdot 0 + 0\cdot 0' = 0 + 0 = 0$

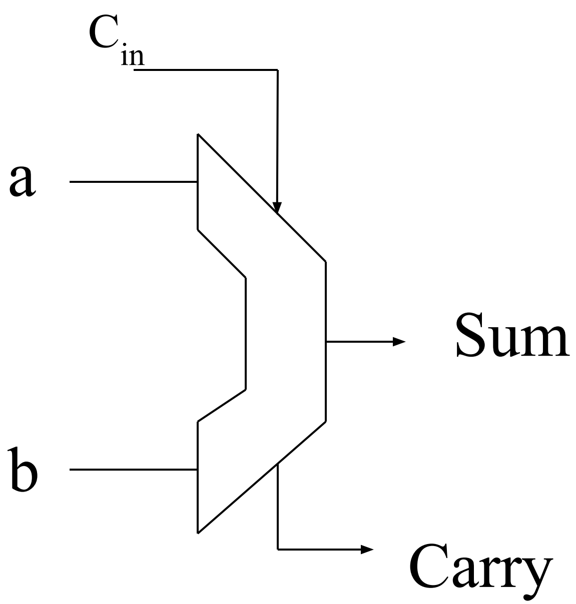
$\text{Sum}(0,1) = 0'\cdot 1 + 0\cdot 1' = 1 + 0 = 1$

$\text{Sum}(1,1) = 1'\cdot 1 + 1\cdot 1' = 0 + 0 = 0$





Full Adder



Truth Table

Id	a	b	c_{in}	carry	sum
0	0	0	0	0	0
1	0	0	1	0	1
2	0	1	0	0	1
3	0	1	1	1	0
4	1	0	0	0	1
5	1	0	1	1	0
6	1	1	0	1	0
7	1	1	1	1	1

Minterm and Maxterm

Id	a	b	c_{in}	carryout		
0	0	0	0	0		a+b+c
1	0	0	1	0		a+b+c’
2	0	1	0	0		a+b’+c
3	0	1	1	1	a’ b c	
4	1	0	0	0		a’+b+c
5	1	0	1	1	a b’c	
6	1	1	0	1	a b c’	
7	1	1	1	1	a b c	

maxterm

minterm

Minterms

$f_1(a,b,c) = a'bc + ab'c + abc' + abc$

$a'bc = 1$ iff $(a,b,c,) = (0,1,1)$
$ab'c = 1$ iff $(a,b,c,) = (1,0,1)$
$abc' = 1$ iff $(a,b,c,) = (1,1,0)$
$abc = 1$ iff $(a,b,c,) = (1,1,1)$

$f_1(a,b,c) = 1$ iff $(a,b,c) = (0,1,1), (1,0,1), (1,1,0)$, or $(1,1,1)$

Ex: $f_1(1,0,1) = 1'01 + 10'1 + 101' + 101 = 1$
$f_1(1,0,0) = 1'00 + 10'0 + 100' + 100 = 0$

Maxterms

$$f_2(a,b,c) = (a+b+c)(a+b+c')(a+b'+c)(a'+b+c)$$

$a + b + c = 0$ iff $(a,b,c,) = (0,0,0)$
$a + b + c' = 0$ iff $(a,b,c,) = (0,0,1)$
$a + b' + c = 0$ iff $(a,b,c,) = (0,1,0)$
$a' + b + c = 0$ iff $(a,b,c,) = (1,0,0)$

$f_2(a,b,c) = 0$ iff $(a,b,c) = (0,0,0), (0,0,1), (0,1,0), (1,0,0)$

Ex: $f_2(1,0,1) = (1+0+1)(1+0+1')(1+0'+1)(1'+0+1) = 1$
$f_2(0,1,0) = (0+1+0)(0+1+0')(0+1'+0)(0'+1+0) = 0$

$f_1(a,b,c) = a'bc + ab'c + abc' + abc$
$f_2(a,b,c) = (a+b+c)(a+b+c')(a+b'+c)(a'+b+c)$

$f_1(a, b, c) = m_3 + m_5 + m_6 + m_7 = \Sigma m(3,5,6,7)$
$f_2(a, b, c) = M_0M_1M_2M_4 = \Pi M(0, 1, 2, 4)$

iClicker: Does $f_1 = f_2$?

A. Yes
B. No.

The coverage of a single minterm. E.g. $m_4 = ab'c'$

Id	a	b	c_{in}	carry	minterm $_4 = ab'c'$
0	0	0	0	0	0
1	0	0	1	0	0
2	0	1	0	0	0
3	0	1	1	1	0
4	1	0	0	0	1
5	1	0	1	1	0
6	1	1	0	1	0
7	1	1	1	1	0

Only one row has a 1.

The coverage of a single maxterm. E.g. $M_4 = a'+b+c$

Id	a	b	c_{in}	carry	$maxterm_4 = a+b+c$
0	0	0	0	0	1
1	0	0	1	0	1
2	0	1	0	0	1
3	0	1	1	1	1
4	1	0	0	0	0
5	1	0	1	1	1
6	1	1	0	1	1
7	1	1	1	1	1

Only one row has a 0.

Incompletely Specified Function

Don't care set is important because it allows us to minimize the function

Id	a	b	f (a, b)
0	0	0	1
1	0	1	0
2	1	0	1
3	1	1	-

1) The input does not happen.
2) The input happens, but the output is ignored.

Examples:

-Decimal number 0… 9 uses 4 bits. (1,1,1,1) does not happen.

-Final carry out bit (output is ignored).

Incompletely Specified Function

Id	a	b	c	g(a,b,c)
0	0	0	0	0
1	0	0	1	1
2	0	1	0	-
3	0	1	1	1
~~4~~	~~1~~	~~0~~	~~0~~	~~1~~
5	1	0	1	-
6	1	1	0	0
7	1	1	1	1

$g_1(a,b,c) = a'b'c + a'bc + ab'b' + abc$

$= m_1 + m_3 + m_4 + m_7$

$= \sum m(1,3,4,7)$

$g_2(a,b,c) = (a+b+c)(a'+b'+c)$

$= M_0 M_6$

$= \prod M(0,6)$

iClicker: Does $g_1(a,b,c) = g_2(a,b,c)$?

A. Yes

B. No